SIEMENS

SIMATIC S5

Программируемый контроллер AG S5-135U (S и R процессоры)

Руководство

Номер для заказа:

6ES5 928-3UA12

Содержание

Инструкции, информации Предложения/исправления	C79000-D85633-C704-0112/89	
Рук-во по эксплуатации: Центральное устройство	C79000-B8500-C260-07	1
Центральный процессор 928	C79000-B8500-C632-01	2
RAM- модуль 377 (буферизированный)	C79000-B8500-C615-02	3
Координатор 923A	C79000—B8576-C263-07	4
Координатор 923 C	C79000-B8500-C349-05	5
CPU 928	C79000-B8500-C633-01	6
Многопроцессорный режим работы Программируемых контроллеров S5-135U и S5-155U	C79000-B8500-C500-02	7
Многопроцессорные коммуникации	C79000-B8500-C468-04	8
Программируемые контроллеры серии U (монтаж)	C79000-B8500-C452-03	9

Указания по технике безопасности

Данное руководство содержит указания, которые вы должны соблюдать для обеспечения собственной безопасности, а также защиты от повреждений продукта и связанного с ним оборудования. Эти замечания выделены предупреждающим треугольником и представлены, в соответствии с уровнем опасности следующим образом:



Опасность
указывает, что если не будут приняты надлежащие меры предосторожности, то это **приведет** к гибели людей, тяжким телесным повреждениям или существенному имущественному ущербу.



Предупреждение
указывает, что при отсутствии надлежащих мер предосторожности это **может привести** к гибели людей, тяжким телесным повреждениям или к существенному имущественному ущербу.



Осторожно
указывает, что возможны легкие телесные повреждения и нанесение небольшого имущественного ущерба при непринятии надлежащих мер предосторожности.

Осторожно
указывает, что возможно повреждение имущества, если не будут приняты надлежащие меры безопасности.

Замечание
привлекает ваше внимание к особо важной информации о продукте, обращении с ним или к соответствующей части документации.

Квалифицированный персонал

К монтажу и работе на этом оборудовании должен допускаться только **квалифицированный персонал**. Квалифицированный персонал – это люди, которые имеют право вводить в действие, заземлять и маркировать электрические цепи, оборудование и системы в соответствии со стандартами техники безопасности.

Надлежащее использование

Примите во внимание следующее:



Предупреждение
Это устройство и его компоненты могут использоваться только для целей, описанных в каталоге или технической документации, и в соединении только с теми устройствами или компонентами других производителей, которые были одобрены или рекомендованы фирмой Siemens.

Этот продукт может правильно и надежно функционировать только в том случае, если он правильно транспортируется, хранится, устанавливается и монтируется, а также эксплуатируется и обслуживается в соответствии с рекомендациями.

Товарные знаки

SIMATIC®, SIMATIC HMI® и SIMATIC NET® - это зарегистрированные товарные знаки SIEMENS AG. Некоторые другие обозначения, использованные в этих документах, также являются зарегистрированными товарными знаками; права собственности могут быть нарушены, если они используются третьей стороной для своих собственных целей.

Copyright © Siemens AG 2001 Все права защищены
Воспроизведение, передача или использование этого документа или его содержания не разрешаются без специального письменного разрешения. Нарушители будут нести ответственность за нанесенный ущерб. Все права, включая права, вытекающие из патента или регистрации практической модели или конструкции, сохраняются.

Siemens AG
Департамент автоматизации и приводов
Промышленные системы автоматизации
П\я 4848, D- 90327, Нюрнберг

Отказ от ответственности
Мы проверили содержание этого руководства на соответствие с описанным аппаратным и программным обеспечением. Так как отклонения не могут быть полностью исключены, то мы не можем гарантировать полного соответствия. Однако данные, приведенные в этом руководстве, регулярно пересматриваются, и все необходимые исправления вносятся в последующие издания. Мы будем благодарны за предложения по улучшению содержания.

©Siemens AG 2001

Technical data subject to change.

Excellence in Automation & Drives: Siemens

SIEMENS

Опасность при использовании плат других изготовителей, обладающих так называемой совместимостью с системой STMATIC.

″Изготовителю какого - либо продукта (в данном случае системы SIMATIC) вменяется в обязанность наблюдение за продуктом, т.е. он должен предупреждать об общем характере опасности, которая может возникнуть при использовании продукта. Действие данной обязанности наблюдения за продуктом было распространено последними правовыми нормами и на комплектующие детали других изготовителей. Согласно данным нормам изготовитель обязан наблюдать и регистрировать и такие опасности, которые возникают в результате соединения его продукта с продуктом других изготовителей.

Поэтому мы считаем своей обязанностью предостеречь наших заказчиков, использующих системы SIMATIC, от применения в системе автоматизации SIMATIC - в качестве запасных или дополнительных плат других изготовителей, обладающих так называемой совместимостью с системой SIMATIC.

Наши продукты контролируются с помощью системы обеспечения качества» отвечающей самым высоким требованиям. Нам не известно, осуществляют ли другие изготовители плат, обладающих так называемой совместимостью с системой SIMATIC, вообще какой-либо или примерно равноценный контроль качества. Так называемые совместимые с системой SIMATIC платы появляются на рынке без нашего согласия; не существует никаких рекомендаций фирмы Сименс АО об использовании плат других изготовителей с так называемой совместимостью с системой SIMATIC. Реклама других изготовителей плат с так называемой совместимостью с системой SIMATIC создаёт ошибочное впечатление о том, будто бы содержание рекламы в специальных журналах, каталогах или на выставках с нами согласовано. При применении в нашей системе автоматизации плат других изготовителей с так называемой совместимостью с системой SIMATIC речь идёт об использовании нашего продукта с нарушением рекомендаций. Из-за чрезвычайной многосторонности использования наших систем автоматизации SIMATIC и большого количества проданных на международном рынке изделий мы не в состоянии провести какой-либо конкретный анализ опасности, возникающей в результате использования данных плат, с так называемой совместимостью с системой SIMATIC. Анализ влияния всех плат с так называемой совместимостью с системой SIMATIC на наш продукт SIMATIC выходит за рамки возможностей изготовителя. Если при применении плат с так называемой совместимостью с системой SIMATIC в системах автоматизации SIMATIC возникают дефекты, то все гарантийные обязательства на такие системы ликвидируются.

За ущерб, причинённый в результате использования плат, с так называемой совместимостью с системой SIMATIC мы ответственности не несём, так как мы своевременно предупредили пользователей о возможной опасности, связанной с использованием плат с так называемой совместимостью с системой SIMATIC.”

C79000-D85633-C704-0112/89

Правила по использованию
электростатически разрушаемых модулей (EGB)

1. Что означает EGB?

Все модули SIMATIC/TELEPERM выполнены спомощью высокоинтегрированных модулей или элементов технологии КМОП. Эти электронные компоненты очень чувствительны к перенапряжениям и к электростатическим разрядам;

Сокращения таких элементов

Ektrostatisch Gefahrdeten Bauelemente / Baugruppen; "EGB"

Электростатически Разрушаемые Элементы / Модули

Кроме того, часто используется международное сокращение;

"ESD" (Electrostatic Sensitive Device)

Приведенный ниже рисунок на шильдиках шкафов, носителей модулей или упаковок указывает на применение электростатически разрушаемых элементов и, вместе с тем на чувствительность к контактам имеющихся модулей:

”EBG могут быть разрушены напряжением и энергией, которые находятся ниже границы ощущения людей. Такие напряжения возникают уже в том случае если к элементу или модулю дотрагивается неразряженный электростатически человек. Модули, подверженные таким перенапряжениям, невозможно, в большинстве случаев, сразу определить как неисправные, т.к. первые признаки неисправности возникают через продолжительное время эксплуатации.
Для того чтобы электростатический разряд

—почувствовать, необходимо 3500 вольт
—услышать, необходимо 4500 вольт
—увидеть — минимум 5000 вольт.

Однако, даже доля такого напряжения может повредить или разрушить электронный элемент.

Поврежденные, подверженные перенапряжению или ослабленные статическим разрядом элементы могут проявить свою дефектность при изменении типичных данных нагрузки, например, при:
—изменении температуры,
—ударах,
—сотрясениях,
—изменениях нагрузки..

Только при последовательном соблюдении правил защиты и ответственном использовании можно действительно избежать Функциональных нарушений и сбоев модулей EGB .

2. Когда возникает статическое напряжение?

Вы никогда не сможете обнаружить, что Вы сами или материалы, и инструмент с которыми Вы работаете не имеют статического заряда.

В обычных условиях, имеется маленький заряд до 100 В, однако, он может очень быстро возрасти до 35 000 В!

Например, из-за:

—хождения по ковровым дорожкам	до 35000 в
—хождения по искусственным полам	до 12000 в
—сидения на пластиковом стуле	до 18000 в
—пластикового насоса для припоя	до 8000 в
—пластиковой кофейной чашки,	до 5000 в
—пластиковых чехлов	до 5000 в
—книг и тетрадей с обложками из искусственных материалов	до 8000 в

3. Важнейшие меры защиты от статического разряда

* Большинство пластиков очень электризуется и поэтому должны быть отделены от разрушаемых таким образом элементов!
* При обращении с электростатически разрушаемыми элементами, обратите Внимание на хорошее заземление человека, рабочего места и упаковки!

4. Работа с модулями EGB

* К электронным модулям можно прикасаться только тогда, когда этого никак нельзя избежать при выполняемых работах. При этом ни *в* коем случае нельзя брать печатные платы, касаясь разъемов или проводящих дорожек,

* Дотрагиваться до элементов можно только в том случае, если
  —Вы заземлены с помощью EGB-браслетов
  или если
  —на Вас одета EGB-обувь или шины защитного заземления EGB-обуви связанные с EGB-полом.

* Перед прикосновением к электронному модулю Вы должны разрядить собственное тело. Это можно сделать простейшим образом - прикоснувшись к заземленной проводящей поверхности (например, к металлической поверхности электрического шкафа, трубопровода и т.д.).

* Для работы с модулями нельзя использовать разряжающиеся и высокоизолированные вещества, например, пластмассовые подложки, изолированные поверхности столов, частями одежды из синтетических волокон.

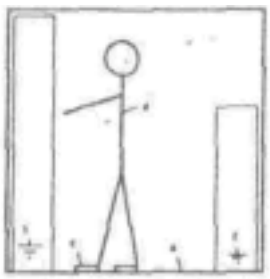
* Модули можно укладывать только на проводящие поверхности (стол с EGB-поверхностью, проводящий EGB-поролон, EGB-упаковочные средства, EGB-транспортные носители).

* Модули нельзя проносить вблизи от терминалов, мониторов, или телевизоров (минимальное расстояние от экрана > 10 см).

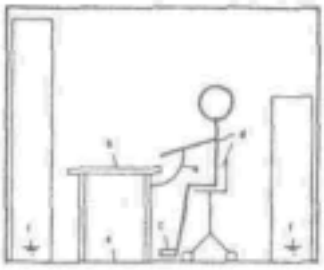
На следующих картинках еще раз пояснены необходимые средства защиты.



Сидячее рабочее место



Стоячее рабочее место



Сидячее/стоячее рабочее место

a токопроводящие полы
b EGB-стол
c EGB-обувь
d EGB-одежда
e EGB-браслет
f заземленные шкафы

5. Измерения и ремонт в EGB- модулях

* Измерения в ECB-модулях можно производить только если

—измерительный прибор заземлен (например, через защитный провод) или

—перед измерениями была проведена кратковременная разрядка измерительной головки прибора (например, касанием металлического кожуха прибора).

* При пайке можно использовать тальке заземленные паяльники.

6. Перевозка EGB- модулей

Модули и элементы хранятся или пересылаются, обычно, в токопроводящих упаковка (например, в металлизированных пластиковых пакетах, металлизированных ячейках).

В том случае, если упаковка непроводящая, необходимо перед упаковкой обернуть, модули проводящим материалом, что можно, например, сделать с помощью токопроводящей пенорезины, EGB-пакета, хозяйственной Фольги или бумаги (но ни в коем случае не пластиковых пакетов или оболочек).

При использовании модулей со встроенными батареями следует помнить, что токопроводящие упаковки не должны касаться места подключения батареи или замыкать её. Для этого место подключения батареи необходимо закрыть с помощью изолирующей ленты или других изолирующих материалов.

Указания для пользователя по технике безопасности

1. Общие указания

Эта документация является "техническим обоснованием поставляемым с продуктом (MtU), содержащим необходимую информацию для определенного использования описанного ниже продукта. Она используется технически квалифицированным персоналом, например, инженерами, программистами или специалистами по обслуживанию, которые имеют специальное образование и соответствующие знания в области измерительной, управляющей и регулирующей техники, а также в области техники автоматизации.

Знание и технически безукоризненное выполнение данных в MtU указаний по безопасности и обслуживанию являются предпосылкой для безаварийного запуска и эксплуатации, а также для безопасности при работе и профилактики описанного продукта. Только квалифицированный персонал, в смысле знания MtU (см. ПУНКТ 2) располагает необходимыми профессиональными знаниями для того, чтобы указания по технике безопасности, приведенные в данном положении в общем, виде, в конкретном случае правильно интерпретировать и использовать.

MtU является неотъемлемой частью общего объема поставки даже если по логическим соображениям предусмотрена раздельная поставка. MtU не содержат всех деталей для любого оборудования описываемого продукта и не могут иметь данные на каждый случай, который может возникнуть при запуске, эксплуатации и обслуживании. Если Вы хотите получить дальнейшую информацию или у Вас возникли какие-то особые проблемы, которые в данном руководстве не освещены или освещены недостаточно, то рекомендуем обратиться в ближайшее к Вам представительство фирмы SIEMENS.

Кроме того, необходимо указать, что содержание этой документации к продукту не является частью имеющихся соглашений, договоренностей или правовых отношений и не может их изменять. Все обязательства Фирмы SIEMENS можно получить из соответствующего контракта, который также содержит полные гарантийные правила. Эти контрактные гарантийные определения на основе положений данного руководства ни дополняются, ни ограничиваются.

2. Квалифицированный персонал

в смысле указаний по технике безопасности в данном MtU и на самом продукте: это люди, которые

* как проектирующий персонал хорошо знакомы с концепциями безопасности техники автоматизации;

* как обслуживающий персонал знакомый в полном объеме с правилами по технике автоматизации умеет использовать содержание этих MtU на практике;

* как сервисный персонал владеет соответствующими правилами по ремонту техники автоматизации и имеет право запуска в эксплуатацию токовых контуров и устройств/систем в соответствие со стандартами техники безопасности.

3. Указания по безопасности

Приведенные ниже указания с одной стороны служат вашей персональной безопасности, а с другой стороны - предохранению от разрушения описанного продукта или подключенных устройств.

В данных MtU подчеркнуты указания по безопасности и обслуживанию, по предотвращению опасности для жизни и здоровья пользователей или обслуживающего персонала и для избежания разрушения устройств путем ввода в текст таких слов как "Внимание" или "Опасность".

4. Определение области применения

* Устройство/система или компоненты системы могут быть использованы только для случаев установки, предусмотренных в каталоге или в техническом описании, и только в сочетании с рекомендованными или допустимыми фирмой SIEMENS чужими устройствами и компонентами.

* Описанный продукт был разработан, изготовлен, проверен и задокументирован с учетом соответствующих норм безопасности. Поэтому, если учитывать предписания по проектированию, монтажу и согласованной эксплуатации и обслуживанию, а также указания по технике безопасности, то продукт не может быть опасным с точки зрения разрушения оборудования или здоровья людей.

Опасно

* После удаления кожуха или защитной арматуры или после открытия шкафа с системой, определенные части этого устройства/системы, находящиеся под напряжением, становятся доступными.

* При неквалифицированном обращении с устройством/системой или при несоблюдении указанных в данном MtU или на самом устройстве/системе указаний по технике безопасности могут возникнуть тяжелые последствия для человека или оборудования.

* Только квалифицированный персонал может работать с данными устройствами/системами.

* Этот персонал должен быть подробно ознакомлен со всеми источниками опасности и с инструкциями, указанными в данном MtU.

* Безупречная и осторожная эксплуатация продукта включает в себя правильную транспортировку, хранение, установку и монтаж, а также правильное обслуживание и эксплуатацию.

5. Указания по проектированию и установке продукта

Т.к. продукт при его использовании, в основном, является составной частью большой системы или установки, то в этих указаниях должны. Задаваться направления по безопасной интеграции продукта в его окружение.

При этом особенно следует помнить следующие обстоятельства;
Для того чтобы при проектировании устройств техники автоматизации, например, многоканальной конструкции, достичь высокой концептуальной безопасности необходимо, точно, следовать, инструкциям, содержащимся, в, данных MtU, т.к. неправильные действия могут сделать все меры по избежанию опасных ошибок бесполезными или создать дополнительные источники опасности.

* В специальных случаях установки следует обратить внимание на предписания по безопасности и предупреждению несчастных случаев.

* При повышенных требованиях к функциональной безопасности продукта необходимо принять во внимание опасные ошибки (например, при неисправных элементах конструктивна) и с помощью дополнительных мер (например, многоканальная конструкция) блокировать их влияние.

* В страиваемое устройство можно эксплуатировать и обслуживать только во встроенном состоянии или с закрытым кожухом.

* В устройствах с жестким подключением (закрепленное по месту устройство/система) без сетевого разделителя и/или предохранителей, сетевой разделитель и / или предохранители встраиваются при создании установки; устройство подключается к защитному кабелю.

* Устройство/система с жестко подключенным, не отсоединяемым кабелем подключения и без всеполярного сетевого разделителя необходимо иметь рядом заземленный защитный разъем для простого подключения / отключения устройства.

* Устройства, работающие с сетевым напряжением, перед эксплуатацией необходимо проверить, совпадает ли установленная область номинальных напряжений с местным сетевым напряжением.

* При питании 24 В следует обратить внимание на предохраняющее электрическое разделение малых напряжений. Можно использовать только сетевые блоки питания, изготовленные в соответствие с IEC 364-4-41 или HD 384.04.41 (VDE 0100 часть 410).

* Кабели подключения и сигнальные кабели прокладываются таким образом, чтобы индуктивные и емкостные составляющие помех не являлись причиной нарушения функции автоматизации.

* Для того, чтобы обрыв кабеля или жилы на сигнальной стороне не приводил к неопределенном у состоянию устройства автоматизации, при подключении входов и выходов предусматриваются с аппаратной и программной стороны соответствующие меры предосторожности.

* Устройства автоматизации и их обслуживающие элементы устанавливаются таким образом, чтобы они были достаточно защищены от незапланированной деятельности.

* Колебания или отклонения сетевого напряжения от номинального значения не могут превышать границ допуска, указанных в технических данных, в противном случае, не исключаются сбой функций и опасные состояния в электрических модулях/устройствах.

* Следует принять меры, чтобы при прерывании/отключении питания, прерванная программа могла быть продолжена с того же самого места. При этом может также возникнуть кратковременный, не опасный режим работы. Например, использование "Not-Aus " .

* Устройство "Not - Aus" (т.е. Аварийное отключение) согласно EN 60204/IEC 204 (VDE 0113) должно оставаться действующим для всех режимов работы устройства автоматизации. Исключение устройства аварийного останова может привести к неконтролируемым или неопределённым последствиям .

6. Активные и пассивные ошибки устройства автоматизации

* В зависимости от постановки задач, электронные устройства управления могут иметь как активные, так и пассивные опасные ошибки. В устройстве управления приводам, например, опасны активные ошибки т.к. они ведут к неверному включению привода. В функциях сообщении, напротив, пассивные ошибки могут выдать сообщение об опасном режиме работы.

* Это различие возможных ошибок, в зависимости от постановки задач на опасные и неопасные имеет значение для всех соображений безопасности для поставляемого продукта. Везде, где возникновение ошибок в устройстве автоматизации может стать причиной больших разрушений материалов или даже нанесения вреда персоналу, необходимо предпринять дополнительные меры или установить дополнительные устройства, которые в случае ошибки обеспечат безопасное рабочее состояние (например, с помощью независимых переключателей граничного значения, механических блокировок и т.д.)

7. Действия при профилактике и поддержании работоспособности

* Ремонт автоматизированной техники может быть предпринят только специалистом по сервису Фирмы SIEMENS или квалифицированным персоналом (см. выше) для замены частей или компонентов используются только те, которые указаны в списке запасных частей или в главе "Запасные части" этого MtU. Некомпетентное вскрытие и неправильный ремонт могут привести к смерти или тяжелым телесным повреждениям, а также к значительным материальным потерям.

* Перед вскрытием устройства, необходимо всегда отсоединять сетевой разъем или выключать сетевой переключатель.

* При замене предохранителей можно использовать только тот тип, который указан в технических данных или в руководстве по обслуживанию.

* Батареи нельзя держать вблизи огня, а также нагревать корпус батареи т.к. возможен взрыв (максимальная температура окружающей среды 100^0 C). Литиевые батареи или батареи содержащие ртуть нельзя вскрывать и перезаряжать, при замене необходимо использовать тот же тип!

* Батареи и аккумуляторы при выбрасывании считаются спецмусором.

* Если необходимо проводить измерительные или контрольные работы на активном устройстве, то следует выполнять действия и инструкции согласно предписания по предотвращению несчастных случаев VBG 4,0 особенно пункт 8 "Допустимые отклонения при работе с активными частями". При этом используется с оответствующии электрический инструмент.

* При установке монитора:
некомпетентное вмешательство, особенно изменение высокого напряжения или установка других типов электронно-лучевых трубок может привести к тому, что рентгеновское излучение увеличится до недопустимых размеров. Измененное таким образом устройство не может считаться допустимым и не может эксплуатироваться.

SIEMENS

SIMATIC S5 **6ES5135—3KA13**
6ES5135—3KA21
6ES5135—3KA31
135U-центральное устройство **6ES5135—3KB41**

Инструкция **Каталоговый №. C79000-B8500-C260-07**

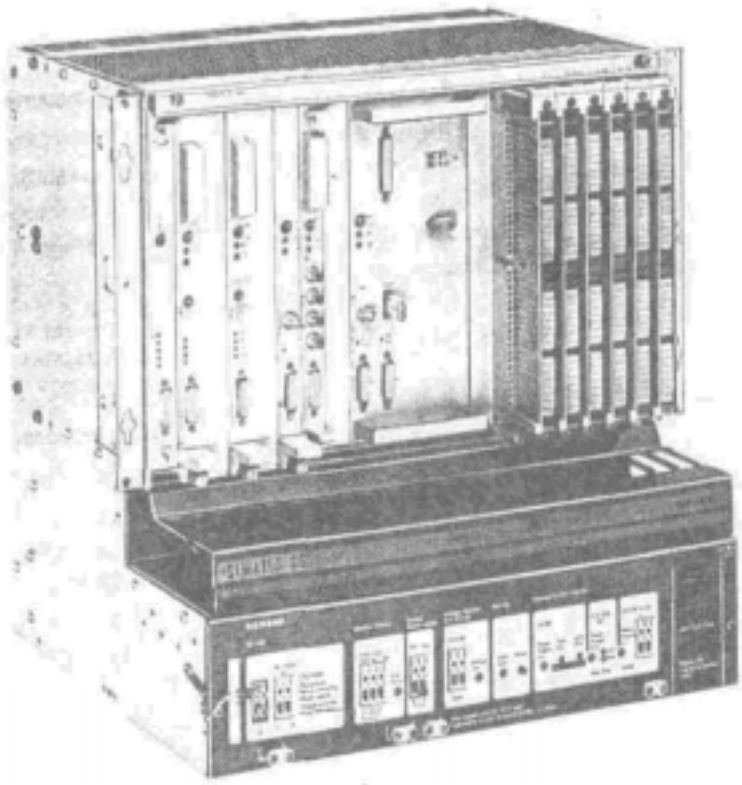


Рис.1 Центральное устройство S5-135U с S-процессором

Содержание **Страница**

1. Техническое описание
1.1. Область применения
1.2. Структура
1.3. Принцип действия
1.4. Технические- данные

2. Монтаж
2.1. Монтаж центрального устройства
2.2. Монтаж блока питания
2.3. Монтаж дополнительного 15-вольтового модуля питания
2.4. Подключение напряжения питания
2.5. Монтаж соединительных кабелей
2.6. Указания по монтажу

3. Эксплуатация
3.1. Общие замечания
3.2. Органы управления и элементы индикации
3.3. Назначение и расположение перемычек в блоке питания
3.4. Реакция блока питания при неисправностях
3.5 . Запуск и Функциональный тест
3.5.1. Запуск контроллера в однопроцессорном режиме
3.5.2. Запуск контроллера в многопроцессорном режиме

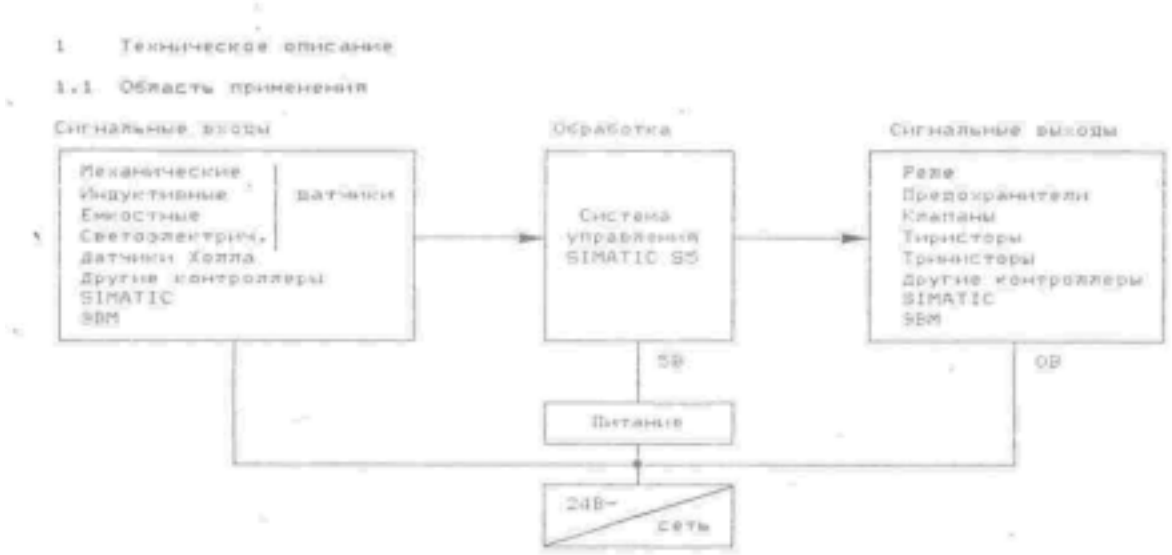
4. Обслуживание
4.1. Общие указания
4.2. Замена буферной батареи и вентилятора
4.3. Назначение контактов заднего разъема блока питания
4.4. Назначение контактов задних разъемов модулей
4.5. Расположение сигналов прерываний на внутренней шине

5. Комплектующие.....…….....38

1. Техническое описание

1. Техническое описание

1.1. Область применения



.

Рис.2 Область применения программируемого контроллера S5-135U

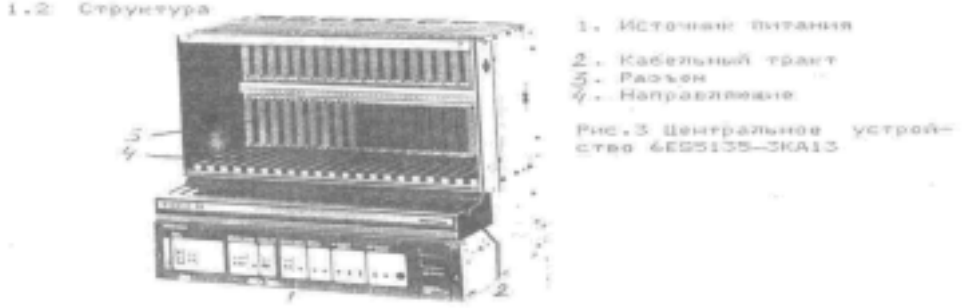
Программируемый контроллер S5-135U- представитель семейства программируемых контроллеров SIMATIC S5.

Возможность применения в контроллере различных процессоров (S-, R-, M- процессоров и/или CPU 928) позволяет решать самые разнообразные задачи и делает AG 135U универсальным. Контроллер без труда может быть приспособлен к решению определенной задачи благодаря своим широким функциональным возможностям (в нем можно использовать от 1 до 4 процессоров).

Этот контроллер может применяться в сочетании с другими типами устройств управления семейства SIMATIC S5 (как программируемыми, так и на жесткой логике).

Для программирования, документирования и тестирования программ применяются соответствующие программирующие устройства. Программа пользователя пишется на языке STEP 5, ассемблере или языках высокого уровня.

1.2. Структура



Контроллер S5-135U состоит из рамы с 21 установочным местом для плат и блока питания, который конструктивно встроен в один блок с вентиляторами. Платы вставляются непосредственно в каркас.

Кабельный тракт для укладки проводов расположен спереди над блоком питания.

Центральные устройства S5-135U поставляются в следующем исполнении (положение на 3/90):

Заказной №	Блок питания [1]) с вентиляторами	Заказной № блока питания
6 ES5 135-3KA13	Вход: 220/110B AC Выход: 5B/18A DC 24B/0.8A DC	6ES5 955-3LC14
6 ES5 135-3KA21	Вход: 220/110B AC Выход: 5B/18A DC 24B/0.8A DC	6ES5 955-3LF12
6 ES5 135-3KA31	Вход: 24B DC Выход: 5B/18A DC 24B/0.8A DC	6ES5 955-3NC13
6 ES5 135-3KA41	Вход: 24B DC Выход: 5B/18A DC 24B/0.8A DC	6ES5 955-3LA12

DC — постоянный ток
AC — переменный ток

Если установочных мест в центральном устройстве не хватает для того, чтобы решить задачу, то применяется устройство расширения (см. Руководство по эксплуатации устройств расширения).

При адресации области P- или Q- периферии надо помнить следующее:

Область Q можно адресовать только через модуль связи с устройством расширения (напр. AS 300, AS 301), поэтому модули входов и выходов с адресами в области Q можно устанавливать только в устройства расширения.

Адреса, которые используются в области Q, не могут быть использованы в центральном устройстве во избежание двойной адресации.

[1]) 15B-модуль (6ES5 956-0AA11) может быть вмонтирован в любой блок питания. Его наличие необходимо в том случае, когда в конфигурацию контроллера входит коммуникационный процессор CP535. Суммарный ток потребления источников на 24B и 15B не может превышать максимальный ток 0,8 A или 2,8 A.

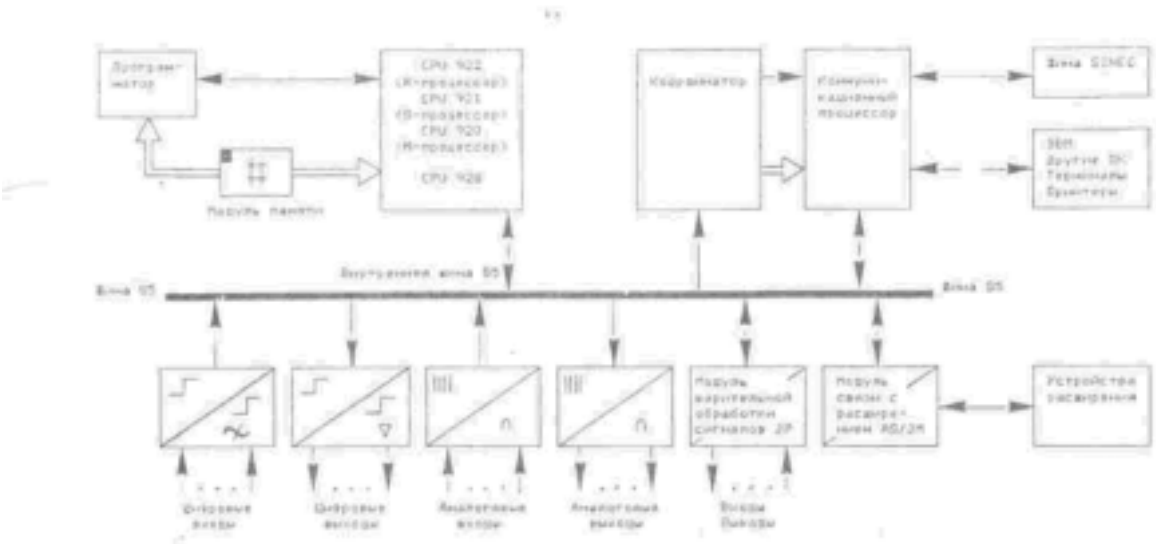
Схема возможного размещения плат



Координатор
CPU 920, CPU 922, CPU 928 [1])
Коммуникационные процессоры, процессоры диагностики
Модуль памяти CP 513 (память на магнитных доменах) память типа "винчестер" CP551 [2])
Модули предварительной обработки сигнала [4]) IP 241 USW, IP 242A, IP 246, IP 247, IP 252
Модули связи 300-5 или 301-5
Модули связи 300-3, 301-3, 304 и 308
Цифровые входы/выходы; аналоговые входы/выходы
Модули предварительной обработки сигнала IP 240, IP 241, IP 242, IP 242A, IP 243, IP 244, DE 432
Модули предварительной обработки сигнала IP 245, IP 257, [5]) IP 260, IP 261
Модули предварительной обработки сигнала IP 246, IP 247, IP 252
Прием сигналов аппаратного прерывания с цифровых входов и IP 241 по 245 возможен только на местах с 43 по 131

[1]) Модуль двойной ширины: занимает дополнительно еще одно установочное место справа от основного.
[2]) МОДУЛЬ пятикратной ширины: занимает ДОПОЛНИТЕЛЬНО еще четыре установочных места справа от основного,.
[4]) Соответствующие модули цифровых входов/выходов 482-4 занимают до четырех мест установки справа от основного.
[5]) Адресация буферов связи

1.3. Принцип действия



1) В многопроцессорном режиме можно установить до 4 CPU

Рис. 4 Структура центрального устройства S5 - 135U

CPU устроены так, что могут работать и в однопроцессорном и в многопроцессорном режимах.

о Однопроцессорный режим

Программа пользователя обрабатывается циклически. Доступ ко входам и выходам возможен по S5 — шине в любое время.

о Многопроцессорный режим

Программируемый контроллер S5—135U — многопроцессорное устройство, в котором каждый CPU решает свою специфическую задачу.

—CPU 921:

S-процессор — Предназначен в основном для быстрой обработки двоичных сигналов (задачи управления). Может также выполнять вычислительны функции и регулирование.

—CPU 922:

R-процессор — Предназначен в основном для быстрой обработки слов (вычисления и регулирование). Возможна также обработка двоичных сигналов.

—CPU 928: — Универсальный процессор. Может вести как быструю обработку двоичных сигналов (задачи управления), так и слов (вычисления и регулирование).

—CPU 92O:

M–процессор — Предназначен для обработки значений измерений, решения арифметических и статистических задач. Программируется на языках BASIC,PASCAL или ASM 86/136.

—Координатор:

KOR 923 — Предназначен для координации работы от 2 до 4 процессоров в многопроцессорном режиме.

—Коммуникационный процессор: Предназначен для диагностики и мониторинга процесса, связи с другими системами.

При использовании нескольких R, M или S — процессоров или CPU 928 задача автоматизации может быть наглядно разделена на составные части. Каждый процессор выполняет свою задачу независимо от других. Это повышает скорость обработки. Каждый CPU может быть запущен независимо от других. В контроллере могут работать до 4 CPU.

Обмен данными между CPU и модулями входов/выходов осуществляется по 35-шине. Принадлежность входных/выходных модулей отдельным CPU в многопроцессорном режиме устанавливается согласно спискам адресов, которые заносятся в блок данных DB 1. В однопроцессорном режиме DB1 может использоваться для оптимизации времени выполнения программы. Поэтому DB1 нельзя использовать как обычный блок данных.

Координатор в циклическом режиме предоставляет каждому процессору доступ к S5-шине. Обмен информацией между процессорами осуществляется через координатор, который имеет для этих целей область маркеров межпроцессорного обмена (923A) и память для "многопроцессорной коммуникации" (только 923 C).

o Память программ

Программа пользователя хранится в субмодулях памяти. В каждый процессор можно вставить по одному такому субмодулю. Субмодули могут быть типа:

—RAM (память со свободным доступом для чтения и записи)
—RAM с буферной батареей
—EPROM (память с ультрафиолетовым стиранием).

RAM — модуль вставляется в процессор и затем может быть запрограммирован программатором по линии связи контроллер—программатор. Модуль EPROM программируется только непосредственно на программаторе через специальный разъем.

o Связь с другими устройствами

—Программатор

Программаторы подключаются непосредственно к процессорам или координатору 923 C для программирования или запуска. Координатор 923 C нельзя использовать в качестве PG-мультиплексора в модификациях центрального устройства 6ES5 135-3KA12 и –3KB12.

—Стандартные периферийные устройства, компьютеры

Коммуникационные процессоры CP (максимум 8 шт.) позволяют организовать независимые друг от друга каналы связи с:

—стандартными периферийными устройствами, такими как принтер, клавиатура, терминал
—ЭВМ;
—другими программируемыми контроллерами.

Необходимые для текстов и изображений данные программируются в субмодулях RAM или EPROM для каждого коммуникационного процессора.

o Функции работы CPU (кроме M процессора)

Режим управления	Функция	Процессор 6ES5....
Ручной новый пуск со сбросом	Все маркеры, таймеры, счетчики и область отображения процесса стираются. Программа пользователя начинает обрабатываться с начала.	Все модули
Ручной новый пуск с сохранением маркеров	Состояние маркеров сохраняется; таймеры, счетчики и область отображения процесса стираются. Программа пользователя обрабатывается с начала.	921-3UA11 921-3UA12
Автоматический новый пуск с сохранением маркеров.	После сброса напряжения питания контроллер автоматически переходит в рабочий режим. В остальном этот режим напоминает ручной новый пуск с сохранение м маркеров.	921-3UA11 921-3UA12
Ручной перезапуск	Состояние маркеров, таймеров, счетчиков и область отображения процесса на время останова сохраняются. 0бработка программы пользователя продолжается из точки прерывания.	921-3UA21 922-3UA11
Автоматический перезапуск		
0бщий сброс	Вся память стирается. Контроллер переходит в начальное состояние. Если используется модуль RAM, необходимо загрузить програ м му, если EPROM, — то возможен новый пуск со сбросом. Общий сброс, может быть выполнен с помощью переключателя на модуле CPU, программа тора и прибора диагностики /0P.	Все модули

1.4. Технические данные

Соответствует правилам безопасности	VDE 0160
Класс защиты	1
Тип защиты	IP 20
Рабочая температура	от О до 55^{o} С
Допустимая температура транспортировки и хранения	от —40 до 70 "С
Относительная влажность	макс . 95% при 25 ^{o} С без выпадения росы
Допустимая высота над уровнем моря	макс. 3500 М
Требования к установке и размещению:	допускается встраивание в стационарные установки, которые не подвержены вибрации; допускается размещение на борту корабля или а/машины при соблюдении правил транспортировки (не допускается размещение непосредственно на двигателе).
Вес	не более 14 кг
Размеры	482.6 мм * 432 мм * 310 мм

Источник питания 6ESS 955—3LC14

<u>Входные характеристики</u> Напряжение питания, U_{EN}	230/115 В переменного тока +10% /-18.7% [1])
Уровень срабатывания сигнализации пониженного напряжения U_E	< 187 (93) В переменного тока
Частота питающего напряжения $f_{E,}$ Гц	от 48 до 63
Входной ток с номинальной нагрузкой при питающем напряжении 23.0 (115) В, I_{EN}	1.25 (2.5) А
Бросок тока при включении (пре дельный ток включения), I_{Emax}	100 (50) А
КПД при номинальной нагрузке с вентилятором	тип. 61 %
Максимальное время отключения питающего напряжения	> 5 мс
COS Ф питающего напряжения	0,65
Входной предохранитель	быстродействующий 4 А; 250В; 6,3 мм х 32мм; место установки F26.
<u>Выходные характеристики 1</u> Напряжение под нагрузкой U_{AN1}	5,1 В постоянного тока +/-0,5%
Диапазон выходных напряжений	(0,95 . . . 1,05) х U_{AN1}
Ток под. нагрузкой U_{AN1}	18 А
Пульсации напряжения	не более 1% U_{A1}
Отклонение напряжения при резком изменении нагрузки на 50%	не более 5% U_{A1}
Время восстановления при 100% 1_N	не более 5 мс
Защитное отключение при повышении напряжения U_{A1} более	6В + / —5 %
Уровень срабатывания сигнализации понижения напряжения U_{A1} менее	4.75В +5%
Ограничение тока при перегрузке по току	(1.05...1.15) U_{AN1}

[1]) Выбирается переключателем сетевого напряжения

Выходные характеристики 2

Номинальное напряжение U_{AN2}	24В пост. тока +25/—17%
Номинальный ток I_{AN2}	0,8 А
Суммарный ток нагрузки выходов 24АВ и 15В	не более 0.8 А
Пульсации напряжения	не более 0.5% U_{AZ}
Защита от перегрузки по току	1.5А быстродейств. предохр. 250В; 6.3мм х 32мм; место установки F90

Выходные характеристики 3 с дополнительным модулем.

Номинальное напряжение U_{AN3}	15В пост. тока +/—5%
Номинальный ток U_{AN3}	0.5 А DC [1])
Пульсации напряжения	не более 5% U_{A3}
Защитное отключение при повышении напряжения	U_{A3} > 18. 5В
Уровень срабатывания сигнализации понижения напряжения (светодиод на передней панели)	U_{A3} < 14В +/-3%
Защита от перегрузки по току (ограничение тока) I_{A3}	0.5. .. 1 .5 А

Выходные характеристики 4

Номинальное напряжение U_{AN4}	24В пост. тока + 6/-5В
Номинальный ток I_{AN3}	0.4 А [1])
Порог ограничения тока	О.44А
Порог срабатывания индикации понижения напряжения (светодиод на передней панели)	16Б +/-20%
Емкость нагрузки	max. 100 нф

[1]) Сумма выходных токов (I_{A2}+I_{A3}+I_{A4})< 0,8 А DC

Вентиляторы	Два осевых вентилятора
Питающее напряжение	115В-перем. Тока
Производительность каждого	160 $М^3$/час (свободный поток)
Контроль за вентиляторами	Воздушный поток контролируется термисторами и датчиками; останов одного или обоих вентиляторов сигнализируется светодиодом, и контакты соответствующего реле отключают выходные напряжения (этот режим может быть исключен перемычкой F—R) .
Наработка на отказ	тип. 30000... 40000 час при 35 ^{0}C тип. 40000... 50000 час при 30 ^{0}C
Дополнительный контроль Контроль напряжения 24В (напряжения внешнего монитора)	от 15 до 30 В
Электрическая изоляция первичной/вторичной цепи	Есть
Потенциальная развязка	Нет
Буферная батарея Емкость	5 Ач
Напряжение	3.4 В
Долговечность	10 лет без разрядки

Источник питания 6ES5 955—3LF12

<u>Выходные характеристики</u> Напряжение питания U_{EN}	230/115 В AC +10/-18.7% [1])
Уровень срабатывания сигнализации пониженного напряжения U_E	< 187 В AC (93 В AC.)
Частота питающего напряжения f_E	48 . . .63 Гц
Входной ток I_{EN} с номинальной на грузкой при питающем напряж. U_{EN}= 230 В(115 В)	2.95 (5.9) А
Бросок тока при включении (предельное значение) I_{Emax}	200 А (100 А)
КПД с номинальной нагрузкой	тип >70% с. вентилятором
Максимальное время отключения питающего напряжения	› 5мс
COS ф питающего напряжения	0,73
Входной предохранитель	быстродействующий 6 А; 250В 6.3 мм х 32 мм; место установки F107
<u>Входные характеристики 1</u> Номинальное напряжение под нагрузкой U_{AN1}	5.1 В DC +/-0.5%
Диапазон установки выходного напряжения	(0.95 . . . 1.05) х U_{AN1}
Ток под нагрузкой I_{AN1}	40 А DC
Пульсации напряжения	не более 1 % от U_{A1}
Отключение напряжения при изменении нагрузки на 50% при максимальном токе	не более 5% от U_{A1}
Время восстановления	не более 5 мс
Защитное отключение при повышении напряжения U_{A1} более	6В+ /- 5 %
Уровень срабатывания сигнализации понижения напряжения U_{A1} менее (светодиод на передней панели)	4.75 В + 5 %
Ограничение тока при перегрузке по току	(1.05. . . 1.15) х I_{AN1}

[1]) Выбирается переключателем сетевого напряжения

Выходные характеристики 2	
Номинальное напряжение U_{ANZ}	24В DC +25/—17%
Номинальный ток I_{ANZ}	2.8 А DC [1])
Пульсации напряжения	< 0.5% от Uanz
Защита от перегрузки по току	А быстродеиств. предохр.; 250В: 6,3мм х 32мм; место установки F 255
Выходные характеристики 3 с дополнительным модулем	
Номинальное напряжение U_{AN3}	15В DC +/—5%
Номинальный ток I_{AN3}	2 А DC [1])
Пульсации напряжения	< 5% от U_{AN3}
Защитное отключение при повышении напряжения	U_{A3} > 18,5В
Уровень срабатывания сигнализации понижения напряжения (светодиод на передней панели)	U_{A3} < 14В +/—3%
Защита от перегрузки по току I_{A3} (ограничение тока)	2. . .3 А
Выходные характеристики 4: фронт 24В	
Номинальное напряжение U_{AN4}	24В DC + 6 В/—5В
Номинальный ток I_{AN4}	0.4 А [1])
Порог ограничения тока	› 0.44 А
Уровень срабатывания сигнализации понижения напряжения (светодиод на передней панели)	U_{A4} < 16В +/—20%
Емкость нагрузки	макс 100 нФ
Вентиляторы	Два осевых
Питающее напряжение	230/115В пер. тока переключ.
Производительность каждого	160 $М^3$ / час (свободный поток)
Контроль за вентиляторами	Воздушный поток контролируется термисторами и датчиками; останов одного, или обоих вентиляторов сигнализируется светодиодом, и контакты соответствующего реле отключают выходные напряжения (этот режим может быть исключен перемычкой F—R).

[1]) Сумма выходных токов (I_{AZ} + I_{A3} + I_{A4}) < 0.8 А DC

Наработка на отказ	тип. 30000 ... 40000 час при 55 ^{0}C тип. 40000. . .50000 час при 30^{0}C
Дополнительный контроль Контроль напряжения 24В (напряжения внешнего монитора)	› 14. . .20 В
Электрическая изоляция первичной/вторичной цепи	Есть
Потенциальная развязка	-
Буферная батарея Емкость	5 Ач
Напряжение	3.4.В
Долговечность	ок. 10 лет без разрядки
Источник питания 6ES5 955—3NA12.	
Входные характеристики Напряжение питания U_{EN}	24 В DC +25 / — 17 %
Уровень срабатывания сигнализации пониженного напряжения U_E	< 20 В пост. Тока
Входной ток I_{EN} с номинальной на грузкой при напряжении питания 24 В	4.5 А
Бросок тока при включении	100 А
КПД с номинальной нагрузкой	тип. 60%
Максимальное время отключения питающего напряжения	4 мс 5
Предохранитель	быстродеиствующии 6 А; 250 В ; 6.3 мм х 32 мм ; место установки F2
Выходные характеристики 1 Номинальное напряжение под нагрузкой U_{AN1}	5.1В DC: +/- 0.5%
Диапазон установки выходного напряжения	(0,95. . . 1.05) х U_{AN1}
Ток под нагрузкой	10 А DC
Пульсации напряжения	< 1% от U_{AN1}

Отклонение напряжения при изменении нагрузки на 50% при 100%I_N	<5% от U_{AN1}
Время восстановления	< 5 мс
Защитное отключение при повышении напряжения U_{A1} более	6В+/-5%
Уровень срабатывания сигнализации понижения напряжения U_{AN1} менее (светодиод на передней панели)	4.75В+/-5%
Ограничение тока при перегрузке	(1.05…1.15)х U_{AN1}

Выходные характеристики 2

Номинальное напряжение U_{ANZ}	24В DC +25/—17%
Номинальный ток I_{ANZ}	2.8 А DC [1])
Суммарный ток нагрузки выходов 24В и 15В	<0.8А
Пульсации напряжения	< 0.5% от U_{ANZ}
Защита от перегрузки по току	Быстродеиств. предохр. 1,5А 250В: 6,3мм х 32мм; место установки F 490

Выходные характеристики 3 с дополнительным модулем

Номинальное напряжение U_{AN3}	15В DC +/—5%
Номинальный ток I_{AN3}	0.5 А DC [1])
Пульсации напряжения	< 5% от Uan3
Защитное отключение при повышении напряжения	U_{A3} > 18,5В
Уровень срабатывания сигнализации понижения напряжения (светодиод на передней панели)	U_{A3} < 14В +/—3%
Защита от перегрузки по току I_{A3} (ограничение тока)	0.5. . .1.5 А

Выходные характеристики 4: фронт 24В

Номинальное напряжение U_{AN4}	24В DC + 6 В/—5В
Номинальный ток I_{AN4}	0.4 А [1])
Порог ограничения тока	> 0.44 А
Уровень срабатывания сигнализации понижения напряжения (светодиод на передней панели)	U_{A4} < 16В +/—20%
Емкость нагрузки	макс 100 н Ф

[1]) Сумма выходных токов (I_{A2}+I_{A3}+I_{A4})< 0,8 А DC

Вентиляторы	Два осевых
Питающее напряжение	24 В пост. Тока
Производительность каждого	160 M^3 / час (свободный поток)
Контроль за вентиляторами	Воздушный поток контролируется термисторами и датчиками останов одного, или обоих вентиляторов сигнализируется светодиодом, и контакты соответствующего реле отключают выходные напряжения (этот режим может быть исключен перемычкой F—R).
Наработка на отказ	тип. 30000 ... 40000 час при 55 ^{0}C тип. 40000. . .50000 час при 30^{0}C
Дополнительный контроль	
Контроль напряжения 24В	-
Электрическая изоляция первичной/вторичной цепи	-
Потенциальная развязка	Есть
Буферная батарея	
Емкость	5 Ач
Напряжение	3.4.В
Долговечность	ок. 10 лет без разрядки

Источник питания 6ES5 955—3NA13.

Входные характеристики

Напряжение питания U_{EN}	24 В DC +25 / — 17 % DC
Уровень срабатывания сигнализации пониженного напряжения U_E	< 20 В DC
Входной ток Ien при номинальной нагркзке U_{EN}= 24 В	6.9 А
Бросок тока при включении	100 А
КПД с номинальной нагрузкой	тип. 67%
Максимальное время отключения питающего напряжения	> 5 мс

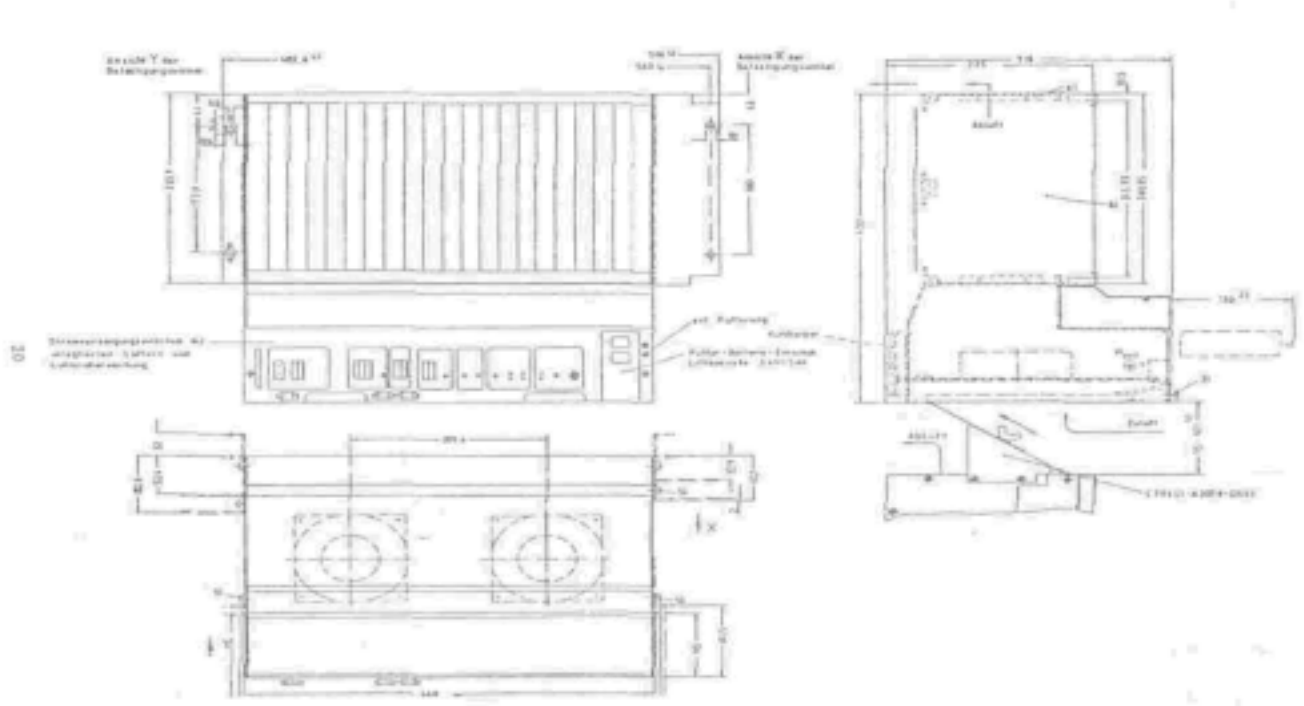
Входной предохранитель	15 А; 250 В ; 6.3 мм х 32 мм ; место установки F1
<u>Выходные характеристики 1</u>	
Номинальное напряжение под нагрузкой U_{AN1}	5.1В DC: +/- 0.5%
диапазон установки выходного напряжения	(0,95. . . 1.05) х U_{AN1}
Ток под нагрузкой	18 А DC
Пульсации напряжения	< 1% от U_{AN1}
Отклонение напряжения при изменении нагрузки на 50% при 100%I_N	<5%
Установочное время	<5мс
Защитное отключение при повышении напряжения U_{A1} более	6В+/-5%
Уровень срабатывания сигнализации понижения напряжения U_{AN1} менее (светодиод на передней панели)	4.75В+/-5%
Ограничение тока при перегрузке	(1.05…1.15)х U_{AN1}
<u>Выходные характеристики 2</u>	
Номинальное напряжение U_{ANZ}	24В DC +25/—17%
Номинальный ток I_{ANZ}	0.8 А DC [1])
Суммарный ток нагрузки выходов 24В и 15В	<0.8А
Пульсации напряжения	< 0.5% от U_{ANZ}
Защита от перегрузки по току	быстродеиств. предохр. 1,5А; место установки F 90
<u>Выходные характеристики 3 с дополнительным модулем</u>	
Номинальное напряжение U_{AN3}	15В DC +/—5%
Номинальный ток I_{AN3}	0.5 А DC [1])
Пульсации напряжения	< 5% от U_{AN3}
Защитное отключение при повышении напряжения	Ua3 > 18,5В
Уровень срабатывания сигнализации понижения напряжения (светодиод на передней панели)	Ua3 < 14В +/—3%
Защита от перегрузки по току I_{A3} (ограничение тока)	0.5. . .1.5 А

[1]) Сумма выходных токов (I_{A2}+I_{A3}+I_{A4})< 0,8 А DC

Выходные характеристики 4: фронт 24В	
Номинальное напряжение U_{AN4}	24В DC + 6 В/—5В
Номинальный ток I_{AN4}	0.4 А [1])
Порог ограничения тока	›0.44 А
Уровень срабатывания сигнализации понижения напряжения (светодиод на передней панели)	16В +/—20%
Емкость нагрузки	макс 100 н Ф
Вентиляторы	Два осевых
Питающее напряжение	230/115В пер.тока переключ
Производительность каждого	160 $М^3$ / час (свободный поток)
Контроль за вентиляторами	Воздушный поток контролируется термисторами и датчиками ; останов одного, или обоих вентиляторов сигнализируется светодиодом, и контакты соответствующего реле отключают выходные напряжения (этот режим может быть исключен перемычкой F—R).
Наработка на отказ	тип. 30000 ... 40000 час при 55 ^{0}C тип. 40000. . .50000 час при 30^{0}C
Дополнительный контроль Контроль напряжения 24В (напряжение внешнего монитора)	>14…20В
Электрическая изоляция первичной/вторичной цепи	Есть
Потенциальная развязка	-
Буферная батарея Емкость	5 Ач
Напряжение	3.4.В
Долговечность	ок. 10 лет без разрядки

[1]) Сумма выходных токов ($I_{A2}+I_{A3}+I_{A4}$)< 0,8 А DC

Установочные размеры



а—вид по оси Y крепящего уголка
б—вид по оси X крепящего уголка
в—приточный воздух
г—отводимый воздух
д—выдвижной блок буферной батареи
литиевая батарея 3,4 В / 5 Ач
е—радиатор
ж—внешняя буферизация
з—выдвижной блок питания со встроенными вентиляторами
и системой контроля вентиляторов
и—отводимый воздух

1) Свободное пространство для приточного воздуха при настенной установке или при монтаже в шкафах
2) Свободное пространство для замены буферной батареи
3) Шина снятия натяжения экранированных проводов
4) Ширина корпуса при произвольном положении крепящего уголка
5) Крепящий уголок привинчен к соединительным шинам
6) Крышка
7) Общая ширина корпуса с крепящими винтами (с учетом высоты головки винта)
8) Базовая несущая конструкция в соответствии с ES 902 для двойного европейского формата 233,4 x 160.
Установочная ширина 28 SEP (SEP - стандартное установочное место)

2. Монтаж

2.1. Монтаж центрального устройства

S5—135U — центральный контроллер предназначен для установки в шкафы управления, открытые рамы или консоли.

Для закрепления прибора требуются винты и шайбы М6.

Сверху с боков и сзади должно быть оставлено пространство не менее 88,9 мм для циркуляции охлаждающего воздуха. Если необходимо разместить контроллеры один над другим, то рекомендуется использовать специальный воздухопровод (6ES5 981 — ODA11) .

Отвод тепла в самом шкафу должен обеспечиваться естественной конвекцией или вентиляцией.

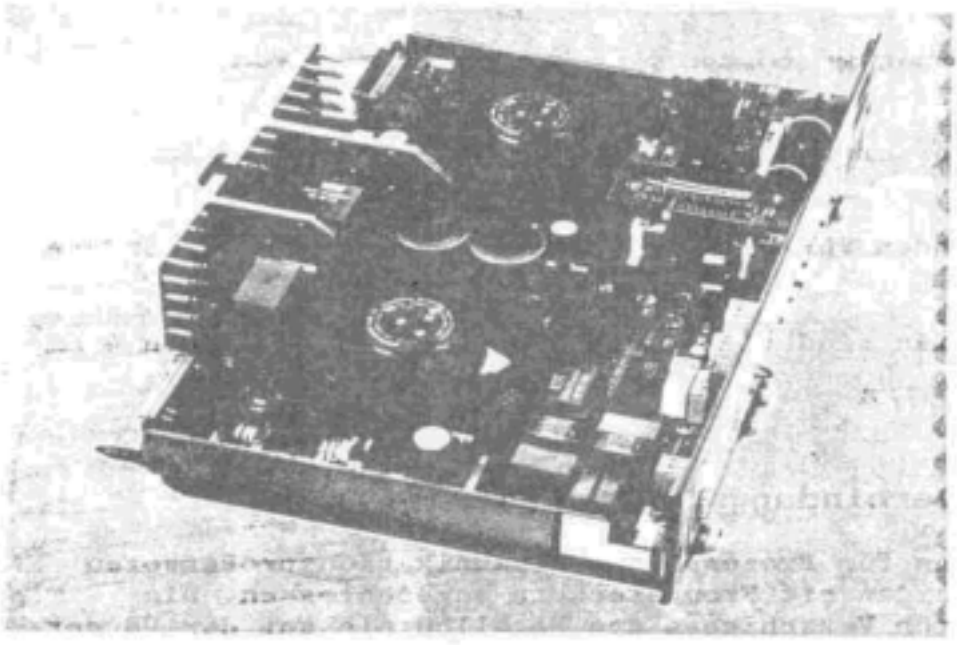
2.2. Монтаж блока питания

Блок питания можно устанавливать только при отключенном напряжении питания. Если Вы хотите снять блок питания с контроллера — открутите два винта слева и справа передней панели, и, после этого, выдвигая на себя, можете вынуть блок питания .

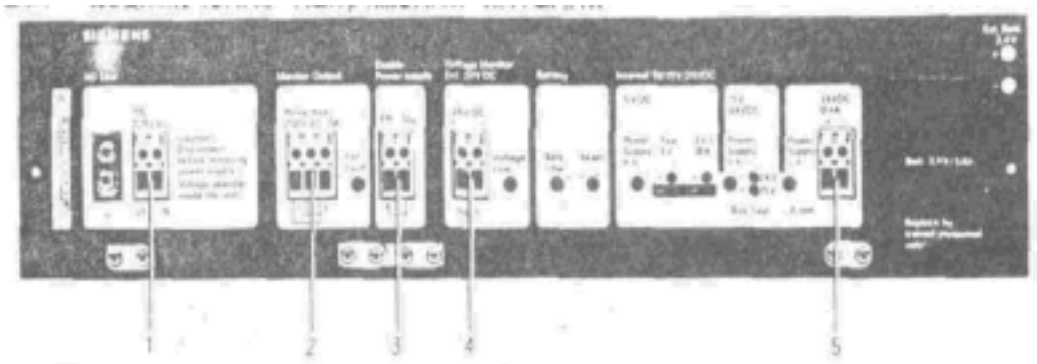
2.3. Монтаж дополнительного 15—вольтового модуля питания

Дополнительный модуль питания можно устанавливать только при отключенном напряжении.

Порядок снятия модуля описан в разделе 2.2.



2.4. Подключение напряжения питания



1. Клеммник напряжения питания :
   В зависимости от типа источника питания подключается либо 220В переменного тока, либо 24В постоянного.

2. Выход контрольного устройства :
   Если один или оба вентилятора не работают, то это сигнализируется светодиодом на передней панели, и выходные напряжения отключаются, (эта Функция может быть устранена посредством перемычки F—R; в этом случае задействованными остаются лишь контакты сигнального реле и светодиод).

   1 2 3 (о подключении см. Указания по монтажу)

3. Напряжение блокировки ;
   Отсутствие напряжения на EN-входе приводит к отключению источника питания. С помощью выхода U_H можно управлять максимум семью анодами EN (расположены на передней панели). (о подключении см. Указания по монтажу).

4. Контроль напряжения;
   Вход для контроля внешнего напряжения 24 В; должен быть нагружен или отключен перемычкой BA—EX в источнике питания (за исключением модификации источника питания 6ES5 955—3NA11).

5. Выход 24В; 0.4.А постоянного тока:
   Предназначен для запитывания деблокировочных вводов U-периферии .

Рис.7 Подключение напряжения питания

При подключении необходимо руководствоваться стандартами VDE, особенно VDE 0100.

Сечение подключаемых к передним клеммам проводов не должно превышать 4 мм^2.

2.5. Монтаж соединительных кабелей

Подключение соединительных кабелей к модулям центральных процессоров, коммуникационных процессоров и интерфейсов расширения производится через передние разъемы. После подключения, передние разъемы необходимо закрепить скользящим металлическим фиксатором. Обратите внимание на соответствие типа переднего разъема типу модуля, т.к. несоблюдение этого требования чревато непредсказуемыми последствиями.

2.6 Указания по монтажу

Кабели напряжения питания 220 В переменного тома внутри шкафов могут прокладываться в одних коробах с остальными соединительными кабелями только при условии экранирования сигнальных кабелей.

Заземляющий провод от внешнего источника питания к контроллеру должен быть как можно более коротким, с поперечным сечением не менее 10 мм^2, хорошо закреплен и проложен отдельно от сигнальных проводников.

Если программируемый контроллер устанавливается в шкафу, то его корпус должен иметь хороший контакт с несущей консолью шкафа.

Внешний источник питания желательно устанавливать в верхней части шкафа.

Металлические части шкафа (боковые стенки, дверь и т.д). должны быть соединены между собой низкоомными проводами (сечением 10 — 16мм^2). Сам шкаф необходимо соединить с защитным проводником.

Если для подключения входов/выходов используются экранированные кабели, то экран должен быть подключен с обеих сторон к экранной шине, которая с низким импедансом соединена с корпусом контроллера.

3. Эксплуатация

3.1. Общие замечания

— Недопустимо напряжение более 50В между выгодными проводниками и потенциалом защитного проводника блока питания.

— Защитный проводник должен быть всегда подключен.

— Если произошла перегрузка по напряжению, то источник питания блокирует выводы (выходные напряжения 5 и 24 В ограничиваются на уровне менее О . 5 В) .

Блок питания можно опять запустить в работу, если выключить и включить напряжение питания, при условии, что перегрузка не была обусловлена внутренней неисправностью источника питания.

— Источник питания функционирует правильно только при токе нагрузки не менее 1А (для моди4'икации 3LF12 — не менее 2 А').

— Выгодное напряжение должно достигать номинального уровня максимум за 200 мс.

— В основании корпуса источника питания можно установить воздушный фильтр.

3.2. Органы управления и элементы индикации

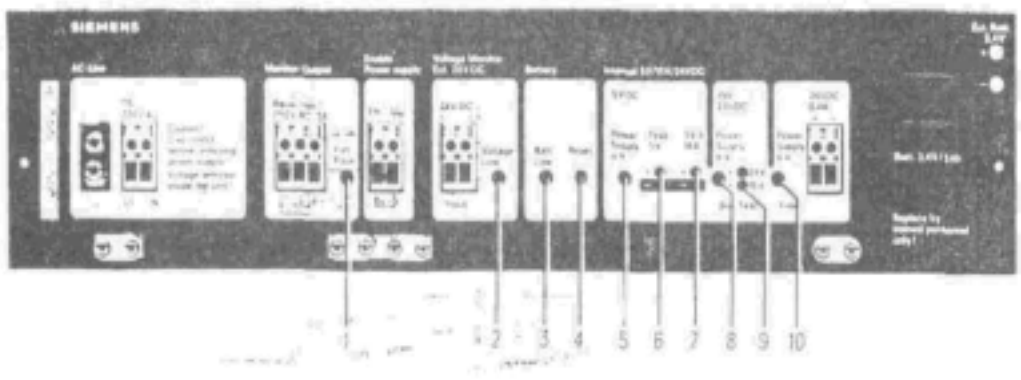


Рис. 3 Источник питания 6ES5 955—3...

1 Красный светодиод "FAN FAULT"
Этот светодиод загорается, если вентилятор неисправен. В этом случае при наличии перемычки F-R через &--10 сек, блок питания выключается. Если по техническим причинам контроллер нельзя тотчас же отключить , то перемычку F-R необходимо снять, но помнить, что во избежание перегрева блок питания должен быть выключен не позднее, 'чем через 60 сек. после сообщения.

2 Красный светодиод "VOLTAGE LOW"
Этот светодиод загорается, если выходное напряжение понизилось до недопустимого уровня (это не относится к модификации источника питания — 3 NA 12).

3 Желтый светодиод "BATT. LOW”
Желтый светодиод загорается, если напряжение буферной батареи упало до 2.7В, при выключении напряжения питания данные, буферизированные в RAM, теряются, после замены батареи СВЕетодиод "BATT. LOW " сразу гаснет.

4 Кнопка "RESET"
Если напряжение питание на контроллер не подано, то надо подать его и, если светодиод "BATT. LOW" горит, необходимо заменить батарею. После успешной замены батареи необходимо нажать кнопку квитирования, иначе контроллер при включении питания войдет в состояние "СТОП".

5 Зеленый светодиод "POWER SUPPLY O.K."
Этот светодиод горит, если источник питания обеспечивает требуемое выходное напряжение 5В.

6 Контрольные клеммы "TEST 5 V"
Предназначены для проверки выходного напряжения (стандартная величина 5.1В +/ -О . 5 %') .

7 Контрольные клеммы "3 V = 18 A":
Контроль выходного тока (3В при максимальном токе нагрузки соответствующего источника питания).

8 Зеленый светодиод "РОИER SUPPLY О.К „ "
Этот светодиод горит, если источник питания обеспечивает требуемые выходные напряжения 24В и 15В (при УСЛОВИИ, что задействован дополнительный 15-вольтовый модуль)

9 Контрольные клеммы "15 V/24 DC";
а) для контроля выходного напряжения 24В +25/—17%:
б) для контроля выходного напряжения 15В +/—5 % (при условии, что имеется дополнительный модуль).

10 Зеленый светодиод "POWER SUPPLY O.K.":
Этот светодиод горит, если источник питания обеспечивает требуемое выходное напряжения 24В.

3.3 Назначение и расположение перемычек в блоке питания

Функция	Перемычка
Контроль батареи (/BAU)=вкл. Контроль батареи (/BAU)=выкл.	NN- MM замкн.* NN- MM разомкн.
Выключение блока питания при неисправности вентилятора При неисправности вентилятора блок питания не отключается (остается только светодиод и реле)	F-R замкн. * F-R разомкн.
Работа с контролем напряжения нагрузки Работа без контроля напряжения нагрузки	BA- EX разом * BA- EX замкн.
Управление сигнальным реле (замкнуты контакты 2—3)	
—по ошибке вентилятора	не зависит от других перемычек
—при сообщении об ошибке батареи [2)]	RR- LL замкн.
—без сообщения об ошибке батареи	RR- LL разом к.*
—при сообщении о понижении напряжения [1)] (BASPA=low)	BB- AA замкн. BB- AA разомк.*
—без сообщения о понижении напряжения	

* состояние поставки

1) При понижении напряжения на контрольном входе (Uu < 20В <. 25%) (отключается перемычкой BA-EX) и при пониженном напряжении на выходе (Ua< 4.75В) выдается сигнал BASPA низкого уровня.

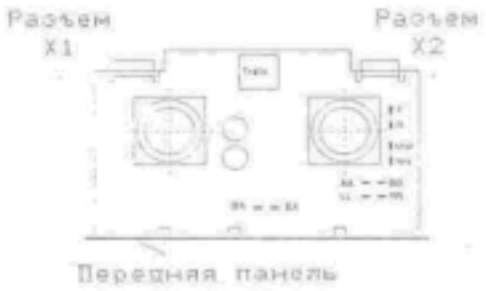
2) Понижение напряжения батареи (Uбат < 2.7В) ведет к сообщению об ошибке батареи (отключается перемычкой MM—NN). Кроме индикации "Batt Low" и выдачи сигнала /BAU можно дополнительно опрашивать реле сообщений (при перемычке RR-LL) для приведенных ниже блоков питания.
Блок питания:
6ES5 95'5- 3NA11 начиная с версии 9
6ES5 955-3NA11 начиная с версии 5
6E55 955-3NA11 начиная с версии 7

При других источниках питания перемычка RR-L.L не действует.

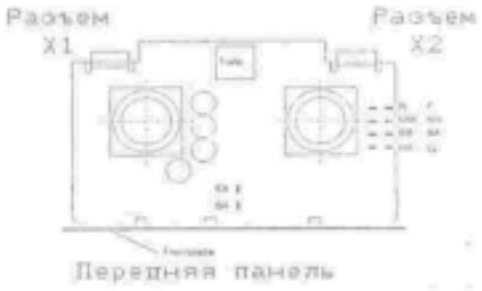
Расположение перемычек

На рисунке показано стандартное расположение перемычек (заводская установка) .

Источник питания 6ES5 955-3LC14



Источник питания 6ES5 955-3LF12



Источник питания 6.ES5 955-3NA12



Источник питания 6ES5 955-3NC13



3.4 Реакция блока питания при неисправностях

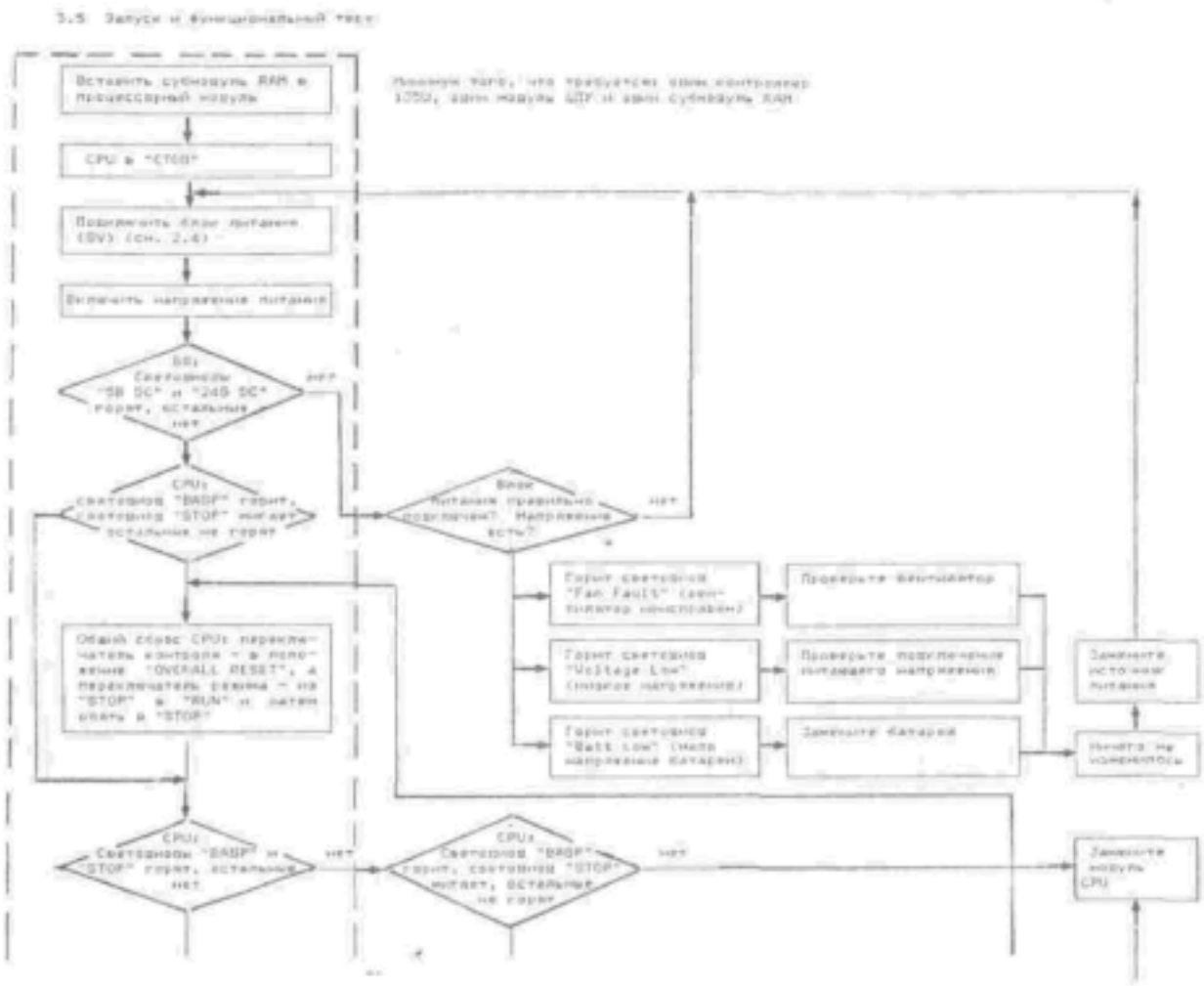
При выключенном блоке питания контакты реле 2-3 замкнуты, а контакты 1—2 разомкнуты. При безошибочной работе контакты реле 1-2 замкнуты, а контакты 2 - 3 разомкнуты.

Кроме неисправностей вентилятора можно, при соответствующей установке перемычек, получать на реле сообщений информацию о других неисправностях в состоянии покоя (контакты реле 2-3 замкнуты) (смотри описание перемычек').

Следующая таблица описывает реакцию блоков питания при неисправностях (условия; перемычка MM-NN .замкнута, перемычка BA-EX разомкнута) .

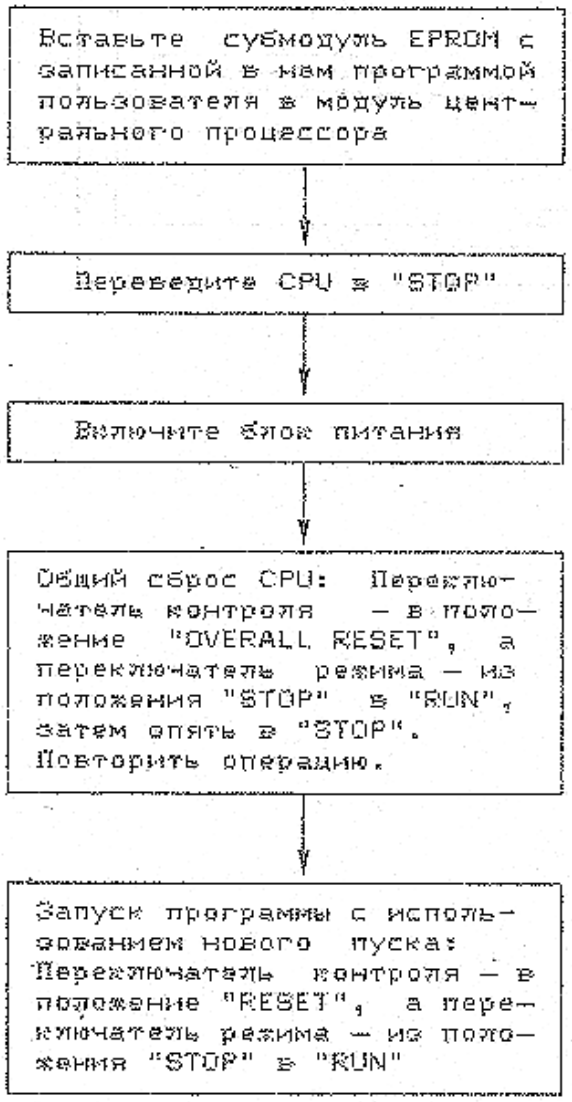
				Сигнализация		Выход. напряжен выключено	
				Светодиод	замкн. контакты реле	F-R: разомкн.	F-R замкн*
деблокировка (EN-UH Присутст вует	Нет ошибки вентилятора	нет ошибки батареи и нет сообщения о пониженном напряжении		не горит	1-2	нет	нет
		ошибка батареи	RF-LL Замк.1)		2-3		
			RR-LL Разомк.		1 -2		
		Сообщ.о пониженн. напряжени и	В Б-А А замк.		2-3		
			В В-А А разомк.		1-2		
	Ошибка вентилятора			горит	2-3	да	нет
Деблокировка (UH-EN отсутствует	Нет ошибки вентилятора	нет ошибки батареи		не горит	1-2	Да	Да
		Ошибка батареи	RR-LL замк 2)				
			RR-LL разомк		2-3		
		BB-AA замкн			2-3		
		BB—AA разомкн			1-2		

[1]) Смотри замечание [1]) Раздел 3.3
[2]) Зависит от версии. Смотри замечание [2]) Раздел 3.3

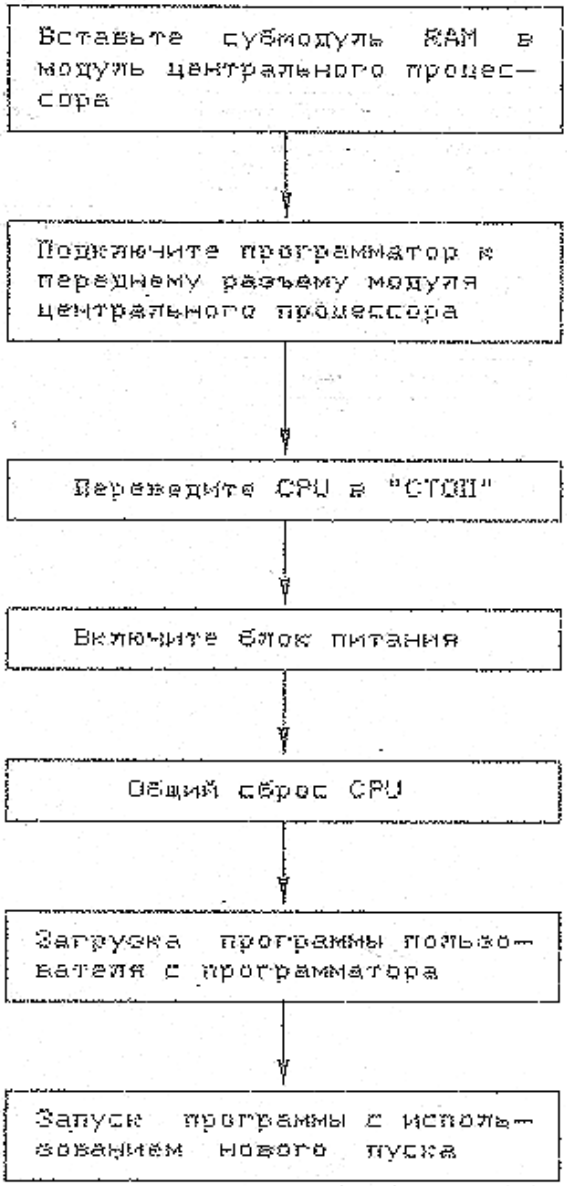

3.5 Запуск и функциональный тест
Встановіть субмодуль RAM в процесорний модуль
Помните того, что требуется один контроллер 1752, один модуль ЦПУ и один субмодуль RAM
CPU в "STOP"
Подключить блок питания (5V) (см. 2.6)
Включить напряжение питания
Светодиоды "5B SC" и "24S SC" горят, остальные нет
нет
CPU: светодиод "BASP" горит, светодиод "STOP" мигает, остальные не горят
Блок питания правильно подключен? Напряжение есть?
нет
Горит светодиод "Fan Fault" (вентилятор неисправен)
Проверьте вентилятор
Горит светодиод "Voltage Low" (низкое напряжение)
Проверьте подключение питающего напряжения
Горит светодиод "Batt Low" (мало напряжение батареи)
Замените батарею
Замените источник питания
Начать не изменилось
Общий сброс CPU: переключатель контроля – в положение "OVERALL RESET", а переключатель режима – из "STOP" в "RUN" и затем опять в "STOP"
CPU: Светодиоды "BASP" и "STOP" горят, остальные нет
нет
CPU: Светодиод "BASP" горит, светодиод "STOP" мигает, остальные не горят
нет
Замените модуль CPU
Запуск CPU: переключатель контроля – в положение "RESET", переключатель режимов – из "STOP" в "RUN"
Субмодуль "RAM" вставлен?
нет
Вставьте правильно модуль "RAM"
Замените модуль "RAM"
CPU: Светодиод "RUN" горит, а "BASP" – нет
нет
Контроллер запущен

3.5.1 Запуск контроллера в однопроцессорном режиме

Последовательность действий при использовании субмодуля EPROM:

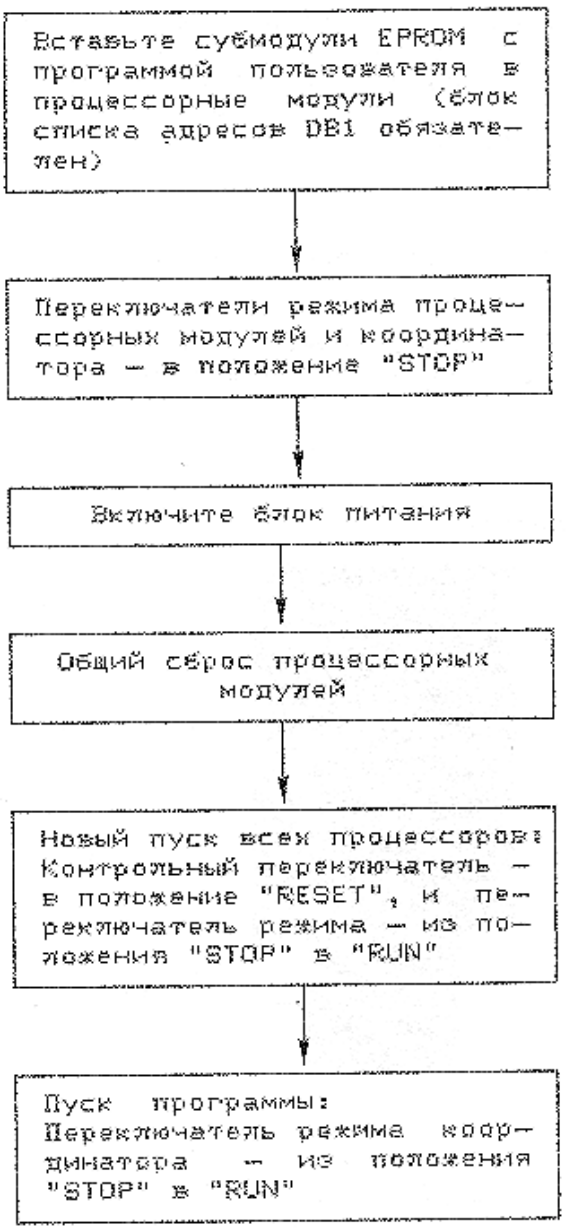


Последовательность действий при использовании субмодуля RAM:

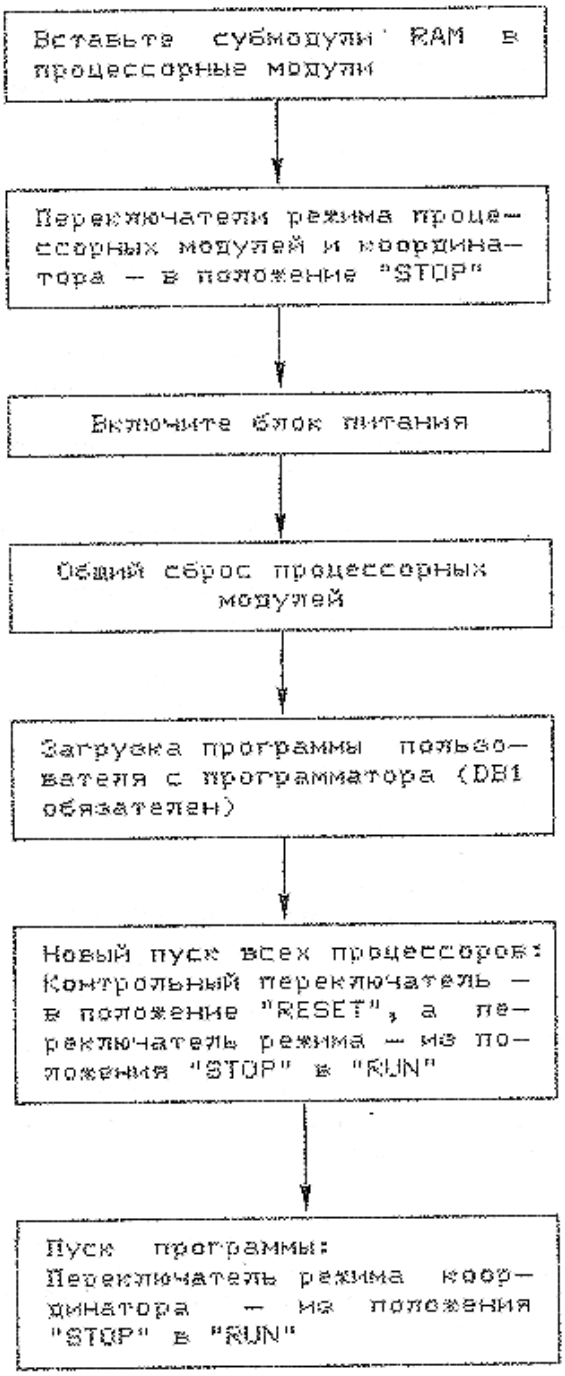


3.5.2 Запуск контроллера в многопроцессорном режиме

Последовательность действий при использовании субмодуля EPROM:



Последовательность действий при использовании субмодуля RAM:



4. Обслуживание

4.1 Общие указания

o Установка и снятие модулей

Запрещается вставлять и вынимать модули из контроллера, если он включен, т.к. это может привести к непредсказуемым последствиям (исключение составляет U- периферия). Однако, если это по каким-либо причинам необходимо сделать, то центральный процессор при этом обязательно должен находиться в состоянии "стоп".

o Блок питания

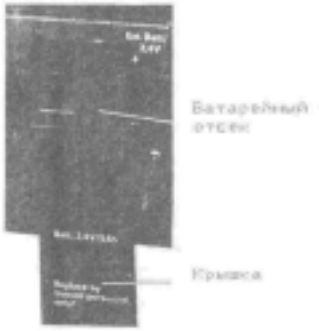
Блок питания можно демонтировать только при отключенном напряжении.

При снятии блока питания надо сохранить связь буферной батареи с шиной контроллера, чтобы не потерять программу пользователя.
Профилактические работы не требуются.

4.2 Замена буферной батареи и вентилятора

Замена буферной батареи не приведет к потере информации в памяти, если контроллер в этот момент будет включен или если к клеммам "Ext. Batt ." будет подведено внешнее напряжение (3.4В) В последнем случае необходимо проследить за правильностью соблюдения полярности.

Буферную батарею следует менять раз в 3 года независимо от объема памяти и режима ее использования (емкость батареи 5Ачас).



Чтобы заменить батарею, необходимо:

—открыть крышку (сдвинуть ее вниз)
—вынуть батарею
—заменить батарею

Наработка вентилятора на отказ (см. Технические данные) зависит от длительности работы, температуры и условий окружающей среды. При остановке вентилятора во время работы включение контроля вентилятора (перемычка F-R замкнута) позволяет избежать нежелательного воздействия, например, на модули; блок питания отключается.

В отдельных случаях необходима профилактическая замена вентиляторов через заданный промежуток времени.

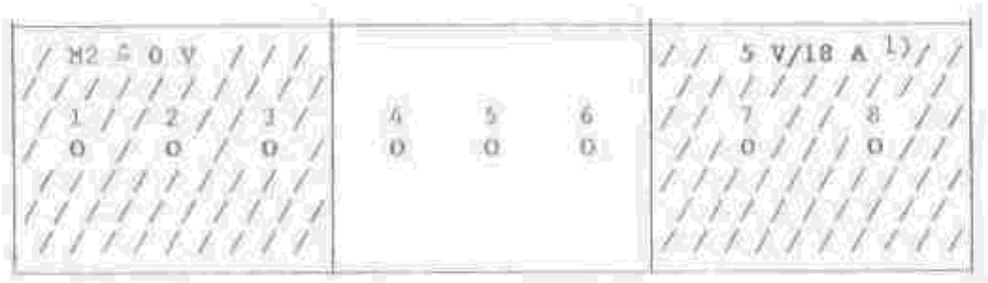
Для замены вентилятора блок питания в обесточенном состоянии снимается, крепежные винты вентилятора откручиваются, питание вентилятора (штекерные контакты) отключается. Установка вентилятора производится в обратной последовательности.

4.3. Назначение контактов заднего разъема блока питания

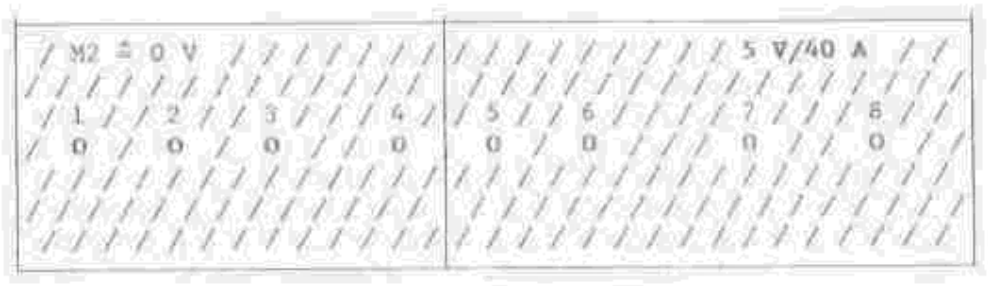
o Подключение нагрузки

миниатюрный разъем X1, 8-полюсный, с 5/8 сильноточными контактами класса D по MIL-C24308, вид сзади

Для –3LC14, -3NA12, -3NC13:

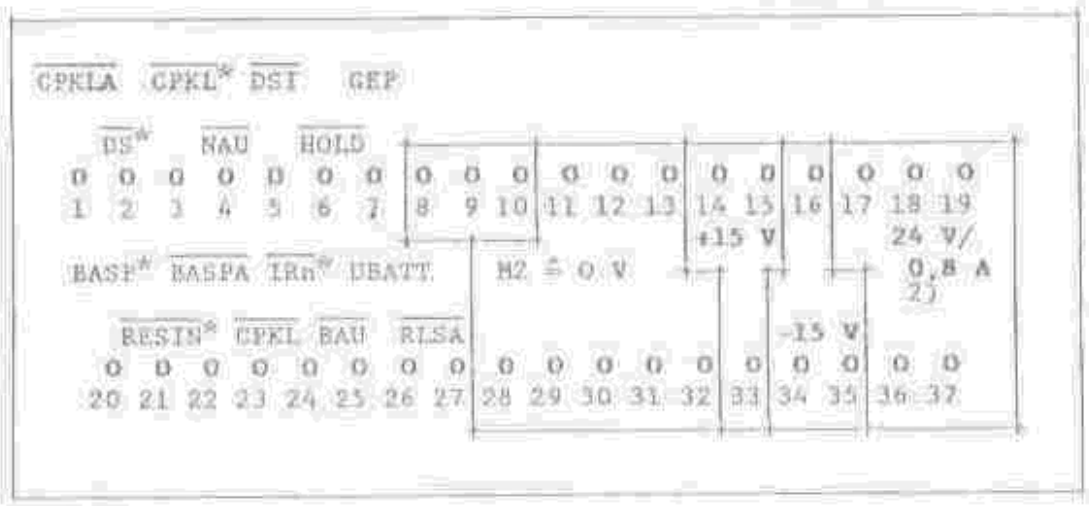


Для –3LF12:



o Подключение сигналов (включая 24В/0,8А или 24В/2,8А в случае 3LF12)

Миниатюрный разъем X2, 37-полюсный, класса D по MIL-C24308, вид сзади:



[1]) 5В/10А при блоке питания –3NA12
[2]) 2.8А при блоке питания –3LF12
* сигнал для MC – 210

4.4. Назначение контактов задних разъемов модулей

Координатор (позиция 3)

	z	b	d
2	+ 5В	М 5В	
4	PL	PESPI	UBAT
6	CPKL	ADB 0	ADB 12
8	MEMR	ADB 1	ADB 13
10	MEMW	ADB 2	ADB 14
12	RDY	ADB 3	ADB 15
14	DB 0	ADB 4	PL
16	DB 1	ADB 5	PL
18	DB 2	ADB 6	PL
20	DB 3	ADB 7	PL
22	DB 4	ADB 8	PL
24	DB 5	ADB 9	PL
26	DB 6	ADB10	PL
28	DB 7	ADB11	DSI
30		BASP	PL
32	PL	М 5В	BASPA
2	+ 5В	М 5В	
4	DB 12	DB 8	
6	DB 13	DB 9	PL
8	DB 14	DB 10	PL
10	DB 15	DB 11	PL
12		PL	PL
14	NAU	PL	PL
16	BAU	PL	PL
18	CPKLA	PL	PL
20		PL	PL
22	PEU	PL	PL
24	GEP	PL	PL
26		PL	PL
28	PL	PL	PL
30	M24/15В	M24/15В	M24/15В
32	+ 24В	М 5В	+ 15В

Процессор (позиции 11,19,27,35)

	z	b	d
2	+ 5В	М 5В	
4	PL	PESPI	UBAT
6	CPKL	ADB 0	ADB 12
8	MEMR	ADB 1	ADB 13
10	MEMW	ADB 2	ADB 14
12	RDY	ADB 3	ADB 15
14	DB 0	ADB 4	IRx
16	DB 1	ADB 5	
18	DB 2	ADB 6	
20	DB 3	ADB 7	
22	DB 4	ADB 8	
24	DB 5	ADB 9	
26	DB 6	ADB 10	
28	DB 7	ADB 11	DSI
30	PL	BASP	PL
32	PL	М 5В	BASPA
2	+ 5В	М 5В	
4	DB 12	DB 8	
6	DB 13	DB 9	М 5В
8	DB 14	DB 10	
10	DB 15	DB 11	
12	М 5В		
14	NAU		
16	BAU		
18		М 5В	
20	PL	PL	PL
22	PL	PL	PL
24	GEP	М 5	
26		PL	PL
28	PL	PL	PL
30	M24/15В	M24/15В	M24/15В
32	+ 24В	М 5В	+ 15В

PL = резерв

Коммуникационный процессор (CP 525) позиции 43,51,59,67

	z	b	d
2	+ 5B	M 5B	
4	PL	PESPI	UBAT
6	CPKL	ADB 0	ADB 12
8	MEMR	ADB 1	ADB 13
10	MEMW	ADB 2	ADB 14
12	RDY	ADB 3	ADB 15
14	DB 0	ADB 4	IRA
16	DB 1	ADB 5	IRB
18	DB 2	ADB 6	IRC
20	DB 3	ADB 7	IRD
22	DB 4	ADB 8	
24	DB 5	ADB 9	
26	DB 6	ADB10	
28	DB 7	ADB11	DSI
30		BASP	
32	PL	M 5B	BASPA

2	+ 5B	M 5B	
4	DB 12	DB 8	
6	DB 13	DB 9	
8	DB 14	DB 10	
10	DB 15	DB 11	
12			
14	NAU		
16	BAU		
18			
20		PL	PL
22		PL	PL
24	GEP		
26		PL	PL
28	PL	PL	PL
30	M24/15B	M24/15B	M24/15B
32	+ 24B	M 5B	+ 15B

Периферийные модули позиции 75-131

	z	b	d
2	+ 5B	M 5B	
4	PL	PESPI	UBAT
6	CPKL	ADB 0	ADB 12
8	MEMR	ADB 1	ADB 13
10	MEMW	ADB 2	ADB 14
12	RDY	ADB 3	ADB 15
14	DB 0	ADB 4	IRA
16	DB 1	ADB 5	IRB
18	DB 2	ADB 6	IRC
20	DB 3	ADB 7	IRD
22	DB 4	ADB 8	
24	DB 5	ADB 9	
26	DB 6	ADB 10	
28	DB 7	ADB 11	DSI
30		BASP	PL
32	PL	M 5B	BASPA

PL = резерв

Подключение EG
(позиции 139,147, 155)

	z	b	d
2	+ 5B	M 5B	
4	PL	PESPI	
6	CPKL	ADB 0	ADB 12
8	MEMR	ADB 1	ADB 13
10	MEMW	ADB 2	ADB 14
12	RDY	ADB 3	ADB 15
14	DB 0	ADB 4	
16	DB 1	ADB 5	
18	DB 2	ADB 6	M 5B
20	DB 3	ADB 7	M 5B
22	DB 4	ADB 8	M 5B
24	DB 5	ADB 9	M 5B
26	DB 6	ADB10	M 5B
28	DB 7	ADB11	M 5B
30		BASP	M 5B
32		M 5B	BASPA
2	+ 5B	M 5B	
4	DB 12	DB 8	
6	DB 13	DB 9	
8	DB 14	DB 10	
10	DB 15	DB 11	
12			
14			
16			
18	CPKLA	PEU	
20			
22	M 5B	M 5B	
24	M 5B	M 5B	
26	M 5B	M 5B	
28	M 5B	M 5B	
30	M 5B	M 5B	
32	M 5B	M 5B	

Подключение EG
(позиция 163)

	z	b	d
2	+ 5B	M 5B	
4	PL	PESPI	+ 5B
6	CPKL	ADB 0	ADB 12
8	MEMR	ADB 1	ADB 13
10	MEMW	ADB 2	ADB 14
12	RDY	ADB 3	ADB 15
14	DB 0	ADB 4	+ 5B
16	DB 1	ADB 5	+ 5B
18	DB 2	ADB 6	M 5B
20	DB 3	ADB 7	M 5B
22	DB 4	ADB 8	M 5B
24	DB 5	ADB 9	M 5B
26	DB 6	ADB 10	M 5B
28	DB 7	ADB 11	M 5B
30		BASP	M 5B
32		M 5B	BASPA
2	+ 5B	M 5B	
4	DB 12	DB 8	
6	DB 13	DB 9	
8	DB 14	DB 10	
10	DB 15	DB 11	
12	+ 5B	+ 5B	
14	+ 5B	+ 5B	
16	+ 5B	+ 5B	
18	CPKLA	PEU	
20			
22	M 5B	M 5B	
24	M 5B	M 5B	
26	M 5B	M 5B	
28	M 5B	M 5B	
30	M 5B	M 5B	
32	M 5BB	M 5B	

PL = резерв

4.5 Расположение сигналов прерываний на внутренней шине

Модуль	CPU 1	CPU 2	CPU 3	CPU 4	PE/CP
Номер ячейки	11	19	27	35	ф3-131
Сигнал					
IRA	конт.14				конт.14
IRB		конт.14			конт.16
IRC			конт.14		конт.18
IRD				конт.14	конт.20

PE — модуль периферии
CP — коммуникационный процессор

5. Комплектующие

Наименование	Каталоговый номер	Ремонто-пригод-ность [1]	Место заказа
Блок питания с вентилятором	6ES5 955-3LC13	R	GWK
	6ES5 955-3LF11	R	GWK
	6ES5 955-3NA11	R	GWK
	6ES5 955-3NC12	R	GWK
Пакет вентиляторов (2 штуки)			
для 6ES5 955-3LC14 и 6ES5 955-3LF12	6ES5 988-3LB21	N	GWK
для 6ES5 955-3NA12 и 6ES5 955-3NC13	6ES5 988-3NB11	N	GWK
Дополнительный 15-В модуль	6ES5 956-0AA11	R	GWK
Батарейный отсек	6XW79084-L1001-B5	N	GWK
Буферная батарея	W79084-L1001-B5	N	GWK
Крышка вентилятора	C79451-A3079-D501	N	GWK
Предохранители (6.3 x 32мм)		N	GWK
15 A	299461		
6 A	300095		
4 A	291963		
1.5A	287268		
Держатель фильтра	6ES5 981-0FA11	N	GWK
Фильтр (10 штук)	6ES5 981-0EA11	N	GWK

1) R-ремонтопригодно
N-неремонтопригодно

SIEMENS

SIMATIC S5

ЦЕНТРАЛЬНЫЙ ПРОЦЕССОР 928

6ES5 928-3UA12

руководство по эксплуатации **Заказной №: C79000-B8500-C632-01**

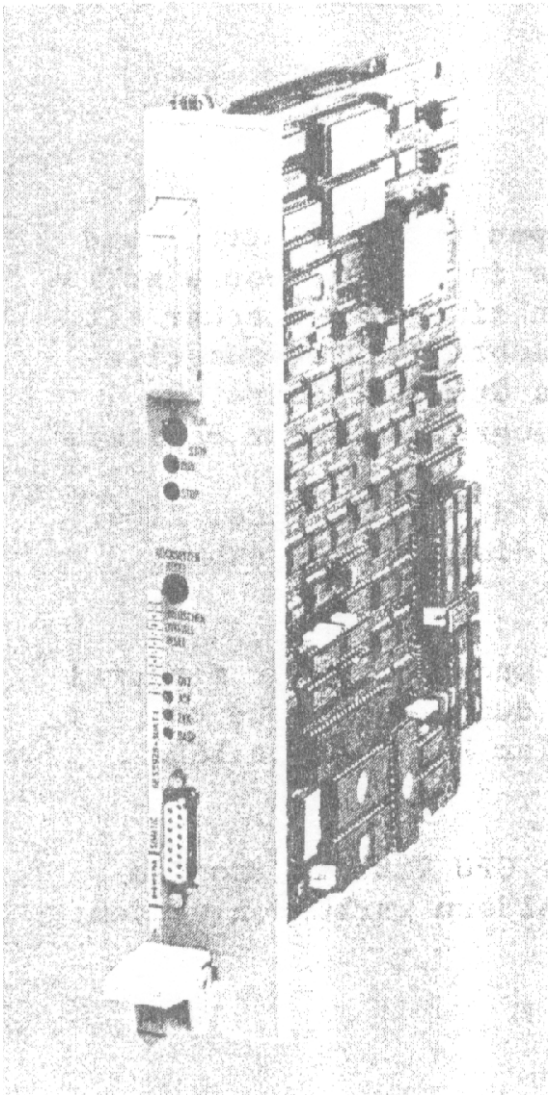


Рис. 1 CPU 928

Оглавление

1. Техническое описание
1.1. Область применения
1.2. Структура
1.3. Принцип действия
1.3.1. Пояснения к принципиальной схеме
1.3.2. Модуль памяти пользователя
1.3.3. Обработка прерываний от процесса
1.4. Распределение памяти
1.5. Технические данные

2. Монтаж
2.1. Снятие и установка модулей
2.2. Установочные места центрального устройства

3. Эксплуатация
3.1. Возможные режимы работы
3.2. Органы управления и элементы индикации
3.3. Запуск
3.3.1. Новый пуск
3.3.2. Ручной перезапуск
3.3.3. Автоматический перезапуск

4. Обслуживание
4.1. Назначение контактов базового разъема
4.2. Назначение контактов переднего разъема

1. Техническое описание

1.1 . Область применения

CPU 928применяется в устройствах автоматизации (AG) S5-135U и S5-155U. Мощность CPU наращивается модульно, благодаря этому обеспечивается возможность многопроцессорного режима работы с использованием до 4 CPU без изменения имеющихся соединений.
CPU 928 является универсальным устройством, обеспечивающим, наряду с быстрой обработкой двоичных сигналов (ориентация на решение задач управления), быструю обработку слов (ориентация на решение задач регулирования).
В качестве языка программирования используется язык STEP 5.

1.2 . Структура

CPU 928 состоит из двух плоских модулей (печатная плата 1 и печатная плата 2) двойного европейского формата, которые выполнены по многослойной технологии (конструкция типа «сандвич»). Оба плоских модуля соединяются друг с другом через разъем и поставляются в виде единой, скрепленной винтами, сборной конструкции. При этом печатная плата 2 является базовым модулем, т.к. на ней смонтированы два базовых разъема для шины S5 базовый несущей конструкции.
Ширина передней панели равна ширине 2 2/3 стандартного установочного места. Прямоугольное гнездо в лицевой панели предназначено для установки модуля памяти пользователя.
Органами управления являются: двухпозиционный переключатель (селектор) и трехпозиционная клавиша. Для индикации рабочих состояний используется один красный и один зеленый светодиод. Сигнализация об ошибках и индикация реакций на ошибки выполняется четырьмя маленькими красными светодиодами.
Подключение CPU 928 к программаторам, диагностическим приборам и пультам управления выполняется через 15-контактный передний разъем.

1.3 . Принцип действия

1.3.1. Пояснения к принципиальной схеме

CPU 928 состоит из нескольких функциональных модулей:

Печатная плата 1 (LP 1):

—Главный CPU
—Интерфейсный CPU (AS-CPU)
—Квазидвухпортовая RAM

Печатная плата 2(LP 1):

—Процессор для обработки двоичных сигналов
—Органы управления и элементы индикации

o Главный CPU

—Основой главного CPU является 16-битовый микропроцессор 80186. Этот микропроцессор имеет прямой доступ к следующим областям памяти:
—Память операционной системы
—Память DB (блоков данных). Она используется в качестве памяти пользователя для хранения блоков данных.
—Модуль памяти пользователя. В этом модуле содержатся пользовательские команды в коде MC-5.
—Память для системных данных. Она содержит данные для операционной системы, содержит отображение процесса на цифровых входах и выходах и, кроме того, используется для маркеров, таймеров и счетчиков.
—Контрольный модуль. В этом модуле аппаратно реализована девятая дорожка и, так же на аппаратном уровне, реализована логика проверки готовности (READY- LOGIK). Девятая дорожка используется для контроля доступа к областям отображения процесса цифровой периферии, а логика проверки готовности контролирует квитирования периферийных модулей, к которым осуществлялось обращение.

o Квазидвухпортовая RAM

—Главный и интерфейсный CPU могут обмениваться друг с другом информацией через квазидвухпортовую RAM.

o Интерфейсный CPU

—Интерфейсный CPU 8031 выполняет пересылку информации на последовательный интерфейс и получение информации с последовательного интерфейса.

o Процессор обработки двоичных сигналов

—Основой центрального устройства процессора обработки двоичных сигналов являются три интегральных схемы:

Устройство управления
Центральное устройство управления обрабатывает все двоичные команды языка STEP 5 и частично или полностью обрабатывает подмножество команд, оперирующих словами. Те команды над словами, которые не могут быть обработаны полностью или частично этим устройством, передаются на процессор, предназначенный для обработки слов (т.е. на главный CPU).

Логическое устройство
В логическом устройстве выполняются, в соответствии с обработанными двоичными командами, логические операции над двоичной информацией. Для поддержки некоторых команд над таймерами и счетчиками в логическом устройстве выполняется предварительная обработка, благодаря этому процессор, предназначенный для обработки слов, переводится в активное состояние только в том случае, когда он должен осуществлять обработку.

Адресное устройство

Шаговый адресный счетчик (SAZ), используемый для адресации команд прикладной программы, находится в адресном регистре. Модуль поддержки функции часов генерирует по системному такту синхронизирующий сигнал актуализации содержимого таймера. Необходимость обработки таймера определяется по состоянию признаков таймера.

о Органы управления и элементы индикации (см. Главу 3.)

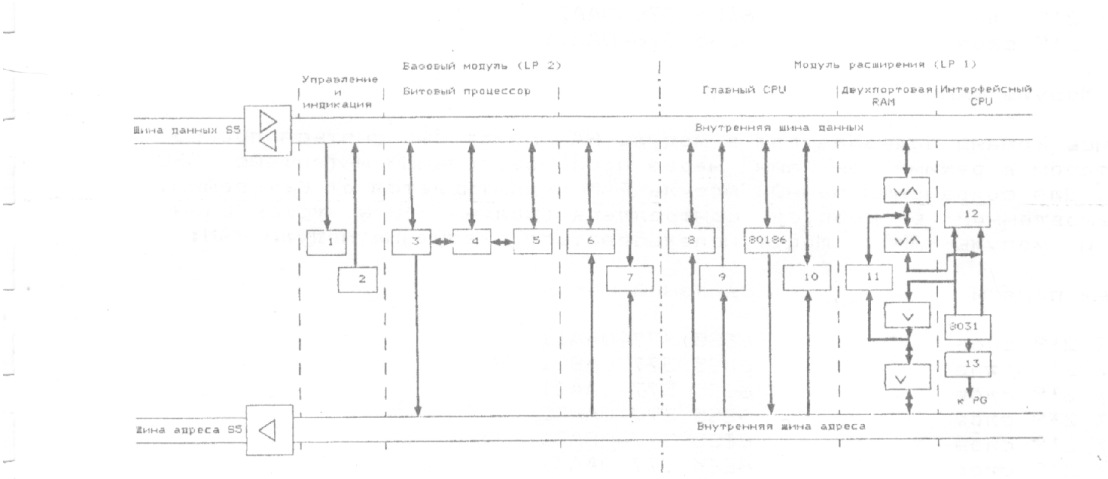


Рис. 2 Функциональная схема

1 —Индикация
2 —Управление
3 —Адресное устройство
4 —Устройство управления
5 —Логическое устройство
6 —Модуль пользователя
7 —Контрольный модуль
8 —Память системных данных
9 —Память операционной системы
10—Память блоков данных
11—Двухпортовая RAM
12—Память устройства управления интерфейсного CPU
13—Ттл/телетайп

1.3.2. Модуль памяти пользователя

К CPU 928 можно подключать в качестве памяти пользователя модуль EPROM (6ES5 376. . .) или модуль RAM (6ES5 377. . .).

o Модуль EPROM

Модуль EPROM программируется непосредственно на программаторе, программирование выполняется в автономном режиме. Могут использоваться следующие модули EPROM.

Объем памяти	Заказной номер
$8x2^{10}$ слов	6ES5 376-0AA11
$16x2^{10}$ слов	6ES5 376-0AA21
$32x2^{10}$ слов	6ES5 376-0AA31

o Модуль RAM

Запись команд пользователя в модуль RAM может выполняться программатором в режиме «он-лайн» через последовательный интерфейс CPU 928. Для сохранения данных модуль RAM подпитывается от батарейки, установленной в корпусе контроллера (должны быть установлены CPU и модуль RAM). Могут использоваться следующие модели RAM.

Объем памяти	Заказной номер
$8x2^{10}$ слов	6ES5 377-0AA11
$16x2^{10}$ слов	6ES5 377-0AB21
$16x2^{10}$ слов	6ES5 377-0AA21
$32x2^{10}$ слов	6ES5 377-0AA32
$32x2^{10}$ слов	6ES5 377-0BA31
$64x2^{10}$ слов	6ES5 377-0AA41

1.3.3. Обработка прерываний от процесса

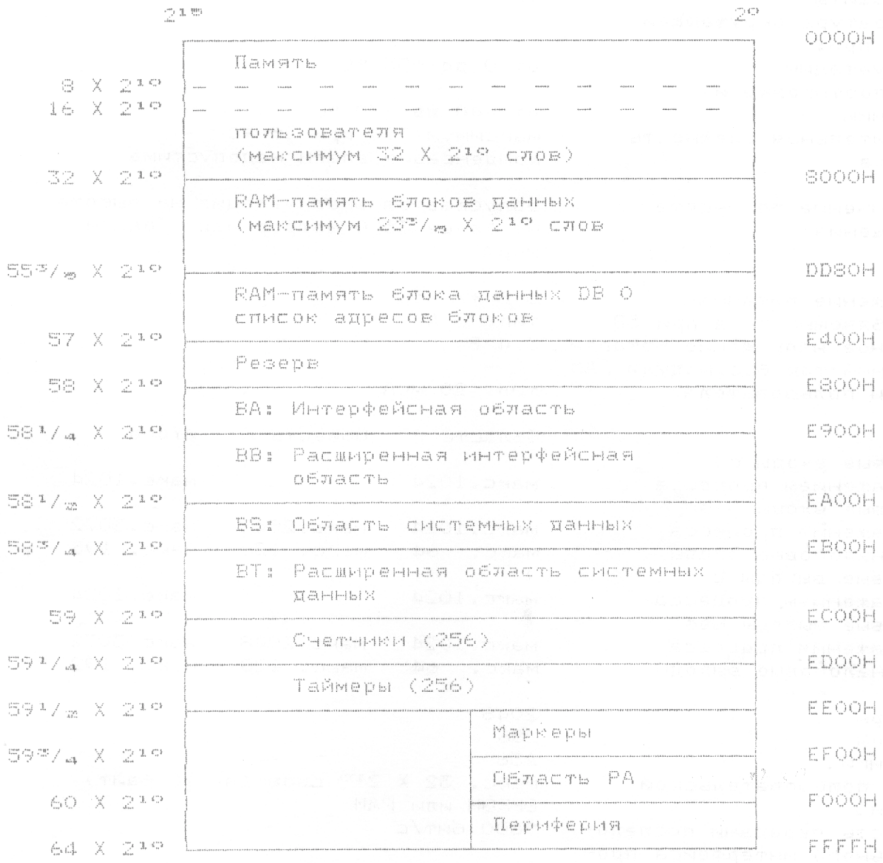
BAG (программируемом контроллере) 135U для каждого процессора предусмотрен один сигнал прерываний (IR). Этот сигнал может быть использован в случае, если реакция на одно или несколько событий должна иметь более высокий приоритет по сравнению с реакцией на другие события.

При обработке прерывания от процесса прекращается циклическая обработка программы, и управление передается программе, находящейся в организационном блоке 2, т.е. в OB 2 (более подробно этот вопрос рассматривается в Руководстве по программированию C79000-B8500-C398).

Эта ,так называемая обработка программ управляемых по прерыванию, возможна только при использовании управляемых по прерыванию модулей цифровых входов (например, 6ES5 432-. . . .) или при использовании работающего аналогичным образом «модуля IP»

[1]) Ппрограммирование на программаторе PG 675 возможно только при использовании адаптера MEP 6ES5 984 OUA21. Кроме того необходимо наличие операционной системы S5-DOS.
[2]) Удлиненное конструктивное исполнение, применяется только для ввода в эксплуатацию.
[3]) Модуль RAM с собственной буферной батареей.

1.3 Распределение памяти



1.5 Технические данные.

Тип защиты	IP 00
Температура окружающей среды при: эксплуатации	от 0 до +55^{0}C
транспортировке и хранении	от-40 до+70^{0}C
Относительная влажность воздуха	максимум 95% при 25^{0}C д опускается эксплуатация на высоте
Ограничение по высоте размещения	не более 3500м над уровнем моря
Напряжение питания	5В+/-5%
Потребляемый ток при 5В	тип. 3А
Буферное напряжение батареи	3,4В
Буферный ток без модуля RAM памяти пользователя	тип. 20мкА

	область P	область Q	Итого
Цифровые входы с отображением процесса	макс. 1024	-	макс. 1024
Цифровые входы без отображения процесса	макс. 1024	макс. 1048	макс. 3072
или аналоговые входы	макс. 64	макс. 128	макс. 193
Цифровые выходы с отображением процесса	макс. 1024	-	макс. 1024
Цифровые входы без отображения процесса	макс. 1024	макс. 2048	макс. 3072
или аналоговые выходы	макс. 64	макс. 128	макс. 192

Маркеры	2048
Таймеры	256
Счетчики	256
Объем пользовательской памяти	макс. 32x2^{10} слов (по 16 байт) EPROM или RAM
Скорость передачи последовательного интерфейса программатора	9600 бит/с
Программные блоки PB	256
Шаговые блоки SB	256
Функциональные блоки FB	256
Функциональные блоки FX	256
Блоки данных DB	256
Блоки данных DX	256
Организационные блоки OB	с OB 1 по 39
Интегрированные организационные блоки специальных функций SF-OB	с 40 по 255

Стандартные функциональные блоки SFB	для цифровых функций (например, 32-битовый двоичный делитель, блок извлечения квадратного корня из числа с плавающей точкой, регистр сдвига, буферная память)
Габариты (В x Н x Т)	40,64 мм x 233,4 мм x 160 мм
Масса	ок. 1 кг.

2. Монтаж

2.1. Снятие и установка модулей

Для того чтобы снять модуль, установленный в центральном устройстве, необходимо потянуть его на себя за специальные ручки, слегка покачивая вверх вниз. Снятие и установка модулей может производиться только при отключенном центральном устройстве.

2.2. Установочные места центрального устройства

Однопроцессорный режим работы:

CPU 928 должен устанавливаться на установочном месте 11 программируемого контроллера 135 U.
При этом CPU 928 займет также и установочное место 19, находящееся справа от установочного места 11.

Многопроцессорный режим работы:

CPU 928, как и R-, S-, и/или M-процессоры должен в соответствии с их количеством, устанавливаться без пропусков на установочных местах 11, 19,и27, и 35, начиная с установочного места 11.

3. Эксплуатация

3.1. Возможные режимы работы

o Однопроцессорный режим работы

Установлен один CPU 928:

Постоянная деблокировка шины S5 и непрерывный цикл.

o Многопроцессорный режим работы

На установочных местах, предусмотренных для многопроцессорного режима работы, могут устанавливаться от двух до четырех 928, R-, M-, и/или S-процессоров. Процессоры перечисленных типов могут устанавливаться в любой комбинации, при этом под CPU 928 отводится два из перечисленных ранее установочных мест. Одно установочное место занимает базовый модуль CPU 928, а модуль расширения занимает установочное место справа от установочного места базового модуля. Так, например, можно установить CPU 928, подключив его через установочное место 35, в этом случае CPU 928 будет занимать установочные места 35 и 43.

При использовании CPU 928 в сочетании с процессорами других типов необходимо соблюдать следующее правило:
CPU 928 подключается справа от других процессоров, при этом между подключенными процессорами и CPU 928 не должно быть свободных установочных мест.
Деблокировка шины выполняется через координатор 923. Назначение входов и выходов маркеров сопряжения, а также входов и выходов периферийных модулей, должно описываться для каждого процессора адресным списком в DB 1. Отсутствие адресных списков приводит к тому, в состояние останова переходит весь программируемый контроллер. Адресный список вводится с помощью программатора в DB1, который хранится в модуле RAM процессора, запись выполняется в режиме «он-лайн». В случае использования модуля EPROM адресный список заводится в этот модуль EPROM непосредственно с программатора, запись выполняется в автономном режиме (см. Руководство по программированию C79000-B8500-C398).

Все процессоры под управлением координатора 923 выполняют синхронный запуск программы.

В случае, когда один из процессоров перед циклической обработкой программы переходит в состояние останова, в состояние останова переходят также и все остальные процессоры, а цифровые входы блокируются сигналом BASP.

Для асинхронного или независимого запуска процессоров можно с помощью координатора 923 выбрать функцию “TEST” («Тестирование») (см. Руководство по программированию C79000-B8500-C398 и руководство по эксплуатации координатора 923). Рекомендуется распределить, с учетом специфики задачи, выполнение прикладной программы между несколькими процессорами.

3.2. Органы управления и элементы индикации.

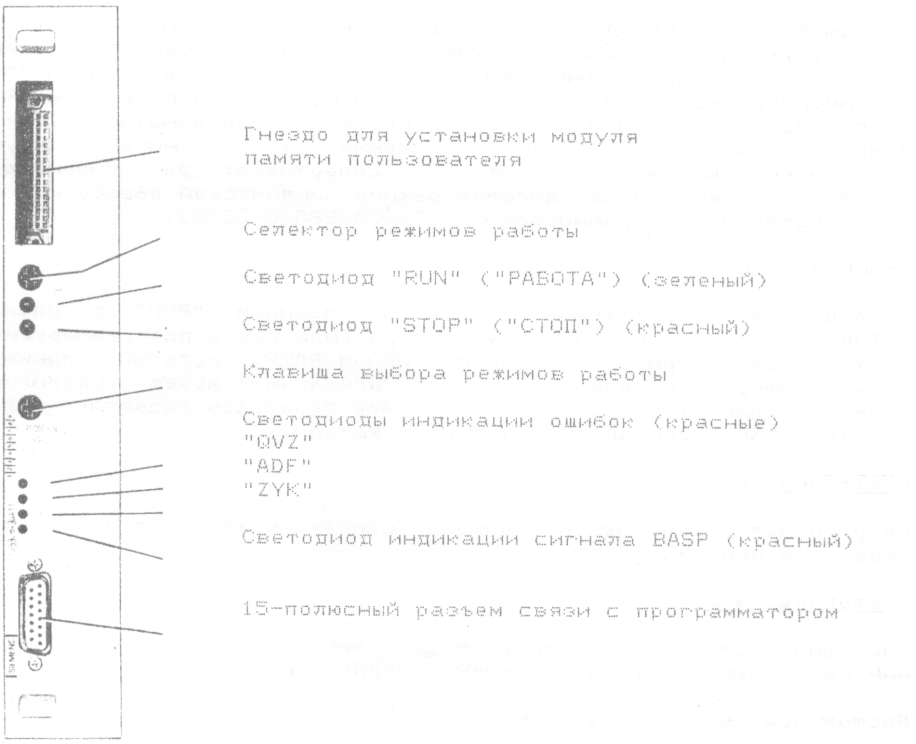


Рис. 3 Передняя панель CPU 928

Селектор режимов работы

Позиция “RUN” («Работа»)

Если селектор находится в позиции “RUN” (“Работа”), то CPU 928 будет находится в режиме циклической обработки, что должно подтверждаться свечение светодиода “RUN”. В циклическом режиме осуществляется непрерывное повторение (=цикл): считывание отображение процесса на входах, обработка прикладной программы в соответствии с последовательностью вызова в OB 1 или в FB 0, формирование отображения процесса на выходах, обновление состояния маркеров связи (если это запрограммировано в DB 1), запуск средств контроля цикла.

Циклическая обработка прерывается: при обновлении таймеров, при прерываниях от процесса, при прерываниях по времени и их обработке, при активизации контура регулятора и при вызове функций по интерфейсу программатора.

Циклическая обработка прекращается при обнаружении ошибки или при сообщении об ошибке в системе, устройстве или программе. Такими ошибками, например, являются: сбой напряжения сети (NAU) и ошибка при выполнении программы (LFZ).

«Автоматический перезапуск» при включении напряжения питания выполняется при соблюдения следующих трех условий: селектор режимов работы при включении напряжения питания находится в позиции “RUN” (в многопроцессорном режиме он должен находится в позиции “RUN” у всех 928, S-, M- и/или R-процессоров и у координатора 923); в памяти зафиксирован «сбой напряжения сети», никаких других записей об ошибках в памяти не содержится; до исчезновения напряжения сети AG находился в режиме циклической обработки (см. Руководство по программированию C79000-B8500-C398).

Позиция “STOP”.

Перевод селектора режимов работы из позиции “RUN” в позицию “STOP” приводит к останову процессора (или всего программируемого контроллера). При этом выдается сигнал BASP, который блокирует модули цифровых выходов. Функция «УПРАВЛЕНИЕ» может выполняться через программатор, т.к. в этом случае процессор подавляет выдачу сигнала BASP и деблокирует цифровые выходы.

Светодиод “RUN”

Непрерывное свечение светодиода свидетельствуют о циклической обработке программы.

Светодиод “STOP”

Состояние останова индицируется состоянием светодиода: непрерывным свечением, быстрым миганием и медленным миганием.

Постоянное свечение светодиода:
—имеет место после включения напряжения питания, если селектор режимов работы находится в позиции “STOP” и нет ошибок при инициализации. В этой ситуации возможен запуск.

Состояние останова после прерывания обработки прикладной программы означает:

—Прекращение обработки в однопроцессорном режиме из-за перевода селектора режимов работы из позиции “RUN” в позицию “STOP”.
—Прекращение обработки в однопроцессорном режиме посредством функции программатора “AG-STOP” («ОСТАНОВ ПРОГРАММИРУЕМОГО КОНТРОЛЛЕРА»).
—Прекращение обработки в однопроцессорном режиме из-за аппаратного сбоя, причиной которого не является какой-либо отдельный процессор (BAU, PEU, NAU).
—Прекращение обработки в однопроцессорном режиме из-за полного сброса.
—Прекращение обработки в многопроцессорном режиме из-за перевода селектора режимов работы координатора в позицию “STOP”.
—Прекращение обработки в многопроцессорном режиме обусловлено другим процессором или координатором 923.
—Прекращение обработки в многопроцессорном режиме посредством функции программатора “AG-STOP”, выполненной на одном из процессоров.
—Прекращение обработки в многопроцессорном режиме после завершения функции программатора «КОНЕЦ КОНТРОЛЯ ОБРАБОТКИ» на одном из других процессоров.

Быстрое мигание светодиода:

—запрошен общий сброс (см. Описание клавиши режимов работы). Запуск возможен только после выполнения общего сброса или после устранения ошибки, приведшей к общему сбросу. Возможные причины выдачи запроса на общий сброс приводятся в Руководстве по программированию.

Медленное мигание светодиода:

—Ошибка в ходе циклической обработки программы этим процессором. Модуль переводится в состояние останова в случае, если не запрограммирована соответствующая обработка ошибки. После перевода селектора режимов работы из позиции “RUN” в позицию “STOP” светодиод переходит в постоянное свечение.
—При неправильной эксплуатации (например, при выборе недопустимого режима работы, ошибки в DB 1 и т.д.).
—При наличии команды останова в программе пользователя.
—При ошибках программирования и при аппаратных сбоях (вызов незагруженного блока, ошибка адресации, задержка квитирования, ошибка в коде команды и т.д.) выдаются сигналы, уточняющие причину ошибки. Эти сигналы выдаются следующими светодиодами:
светодиодом “ADF”
светодиодом “QVZ”
светодиодом “ZYK”
—При выполнении функции программатора «КОНЕЦ КОНТРОЛЯ ОБРАБОТКИ» на этом процессоре.

Клавиша выбора режимов работы

Общий сброс (“OVERALL RESET”)

Стирание содержимого всех модулей RAM с последующей инициализацией всех модулей RAM. Исходная ситуация:
Светодиод “STOP” постоянно горит.

—зафиксировать клавишу выбора режимов работы в позиции “OVERALL RESET”, а селектор режимов работы перевести из позиции “STOP” в позицию “RUN”, после чего селектор вернуть в позицию “STOP”.
Результат: Светодиод “STOP” начинает быстро мигать. Запрашивается выполнение общего сброса [1]).

—клавишу выбора режимов работы вновь зафиксировать в позиции “OVERALL RESET”, селектор режимов работы перевести из позиции “STOP” в позицию “RUN”, после чего селектор вернуть в позицию “STOP”.
Результат: Общий сброс выполнен. Процессор остается в состоянии останова. Светодиод “STOP” горит постоянно. Теперь можно выполнить перезапуск.

Сброс (“RESET”): см. Раздел 3.3

[1]) На этом этапе можно отказаться от выполнения общего сброса: Селектор режимов работы переводится из позиции “STOP” в позицию “RUN”, а затем снова переводится в позицию “STOP”, без использования при этом клавиши выбора режимов работы.
Результат: Общий сброс предотвращен. Процессор остается в состоянии останова. Светодиод “STOP” горит постоянно.

Светодиоды индикации ошибок и сигналов

Светодиод “QVZ”

Светодиод горит, когда периферийный модуль, к которому обращается программа, не выдает квитирующего сигнала:

—отсутствует квитирование периферийного модуля, который в однопроцессорном режиме работы при перезапуске CPU 928 подтвердил бы в области отображения процесса (входы с 0 по 127, выходы с 0 по 127) свое наличие и был зарегистрирован в девятой дорожке как существующий модуль или
—не квитируется периферийный модуль, который в многопроцессорном или однопроцессорном режиме был зарегистрирован в DB1 (адресный список) и был идентифицирован при перезапуске как существующий или
—не квитируется периферийный модуль, к которому осуществляется прямой доступ (команды L/T P …, L/T Q…).

Возможны следующие причины отсутствия квитирования:

—отказ модуля
—модуль снимался во время эксплуатации, в состоянии останова или при выключенном устройстве, и новый пуск после снятия модуля не был осуществлен.

Пользователь может запрограммировать реакцию на QVZ в интерфейсных организационных блоках (см. Руководство по программированию C79000-B8500-C398).

Светодиод “ADF”

При перезапуске CPU 928 операционная система формирует т.н. девятую дорожку. Все физически имеющиеся периферийные модули помечаются логическими единицами в области отображения процесса, а все отсутствующие – логическими нулями в девятой дорожке. В многопроцессорном режиме работы ил программировании адресного списка в DB1 девятая дорожка формируется в соответствии информацией, которая содержится в DB1. CPU 928 прерывает циклическую обработку программы в случае, когда прикладная программа обращается по адресу периферии, которому не соответствует ни один из подключенных модулей. Пользователь может запрограммировать реакцию на ADF в интерфейсных организационных блоках (см. Руководство по программированию C79000-B8500-C398).

Светодиод “ZYK”

Этот светодиод выдает световой сигнал в случае повышения максимального времени цикла. Время цикла определяется как сумма времени выполнения всех сегментов программы пользователя (циклическая обработка + обработка прерывания по времени + обработка прерывания от процесса) и времени выполнения системной программы. Сообщение об ошибке, которое выдается светодиодом “ZYK”, прерывает циклическую обработку программы. Пользователь может запрограммировать реакцию на ADF в интерфейсных организационных блоках (см. Руководство по программированию C79000-B8500-C398).

Светодиод «BASP»

Этот светодиод зажигается в случае блокировки выдачи команд. Цифровые входы непосредственно переключаются в безопасное состояние. По сигналу BASP не осуществляется сброс области отображения цифровой периферии. Сигнал BASP выдается при включении и выключении блока питания, при пониженном напряжении, при переходе программируемого контроллера в состояние останова и при выполнении функции программатора «КОНТРОЛЬ ОБРАБОТКИ».

3.3. Виды запуска

В ходе запуска операционная система определяет и формирует данные, которые необходимы для циклического режима работы. Затем начинается циклическая обработка программы (см. также Руководство по программированию C7900-B8500-C398). Операционная система поддерживает три вида запуска.

3.3.1. Новый пуск

Содержимое маркеров, таймеров, счетчиков и область отображения процесса стирается. Обработка программы пользователя выполняется сначала.

Программируемый контроллер должен находиться в состоянии останова. При работе в многопроцессорном режиме селектор координатора 923 должен находиться в положении «СТОП».

Процессор сбрасывается и переходит в режим циклической обработки программы в случае, когда:

—клавиша выбора режима работы CPU 928 зафиксирована в положении «RESET» и при этом
—селектор режимов работы переводится из позиции «STOP» в позицию «RUN», после чего
—в многопроцессорном режиме работы селектор координатора 923 переводится из позиции «STOP» в позицию «RUN».

3.3.2. Ручной перезапуск

Во время останова сохраняется содержимое маркеров, таймеров, счетчиков и область отображения процесса. Обработка программы пользователя продолжается с точки прерывания.
Перед переходом в состояние останова программируемый контроллер должен находится в циклическом режиме работы или в режиме запуска. В многопроцессорном режиме селектор контроллера 923 должен находится в позиции “STOP”. Процессор переходит в режим циклической обработки программы в случае, когда:
—клавиша выбора режимов работы CPU 928 находится в среднем положении,
—селектор режимов работы переводится из позиции “STOP” в позицию “RUN” и/или после чего
—в многопроцессорном режиме работы селектор координатора 923 переводится из позиции “STOP” в позицию “RUN”

3.3.3. Автоматический перезапуск

Во время останова сохраняется содержимое маркеров, таймеров, счетчиков и область отображения процесса. Обработка программы пользователя продолжается с точки прерывания.

Автоматический перезапуск выполняется после включения напряжения питания в случае, когда:

—программируемый контроллер до возникновения ситуации “Netz Aus” («Отключено сетевое питание») находился в режиме циклической обработки или в режиме запуска;
—селектор режимов работы CPU 928 (в многопроцессорном режиме - селектор режимов работы 928 и S-, M- и/или R- процессоров и координатора 923) находился в позиции “RUN” без каких-либо переключений в другие позиции;
—модуль памяти пользователя не снят
—буферная батарея находится в работоспособном состоянии (должно обеспечивается сохранение данных в модуле RAM).

Текстовый режим работы: см. Руководство по программированию C79000-B8500-C398.

4. Обслуживание

4.1. Назначение контактов базового разъема

Базовый разъем 1

	d	b	z
2		M 5 B	+5 B
4	UBAT	PESP	
6	ADB 12	ADB 0	CPKL
8	ADB 13	ADB 1	MEMR
10	ADB 14	ADB 2	MEMW
12	ADB 15	ADB 3	RDY
14	IR	ADB 4	DB0
16		ADB 5	DB1
18		ADB 6	DB2
20		ADB 7	DB3
22		ADB 8	DB4
24		ADB 9	DB5
26		ADB10	DB6
28	DSI	ADB11	DB7
30	BUSEN	BASP	
32	BASPA	M 5 B	HALT

Базовый разъем 2

	d	b	z
2		M 5 B	+5 B
4		DB 8	DB 12
6	M 5 B	DB 9	DB 13
8		DB 10	DB 14
10		DB 11	DB 15
12			M 5 B
14			NAU
16			BAU
18		M 5 B	
20		STEU	
22	TxD	STOPPA	PEU
24		M 5 B	GEP
26	TEST	RxD	
28		PERO	
30		M 24 B	M 24 B
32		M 5 B	M 5 B

4.2. Назначение контактов переднего разъема

1. Корпус/масса/M_{ent}
2. RxD
3. VP +5B_
4. +24ВВ шины
5. Масса/M_{int}
6. TxD
7. TxD
8. Корпус/масса/M_{ent}
9. RxD
10. 24 В - масса
11. 20 мА/передатчик
12. Масса/M_{int}
13. 20 мА/приемник
14. VPG +5 В_
15. Масса/M_{int}

SIEMENS

SIMATIC S5
Координатор 923A **6ES5 923-3UA11**
Руководство по эксплуатации **Заказной No. C79000—B8576-C263-07**



Рис. 1 Координатор 923A

Оглавление Страница

1. Техническое описание

1.1. Область применения

1.2. Структура

1.3. Принцип действия

1.3.1. Арбитраж шин

1.3.2. Буфер обмена данными

1.4. Технические данные

2. Монтаж

2.1. Установка и снятие модулей

2.2. Установочные места в AC 135U

3. Эксплуатация

3.1. Органы управления

3.2. Режимы работы

3.3. Кодирование числа процессоров

3.4. Адресация буфера обмена данными

3.5. Назначение перемычек

4. Назначение контактов задних разъемов

5. Комплектующие

1. Техническое описание

1.1. Область применения

Координатор 923A устанавливается в программируемый контроллер (AG) 135U. При этом он выполняет две независимые функции:

- Арбитраж шин

Координация работы CPU (центральных процессорных устройств) в многопроцессорном режиме, т.е. при одновременной работе нескольких (от 2 до 4 S-, R- и/или M-) процессоров.

- Буфер обмена данными.

Это функция обмена данными между процессорами

1.2. Структура

Координатор (KOR) 923 A представляет собой печатную плату двойного стандартного европейского Формата.

На задней стороне модуля имеются два 48-полюсных разъема типа "2 ряда" для связи с внутренней S5-шиной контроллера.

Ширина передней панели - $1^{1/3}$ размера стандартного установочного места.

На передней панели имеется трехпозиционный переключатель режимов работы (селектор режимов).

1.3. Принцип действия

1.3.1. Арбитраж шин

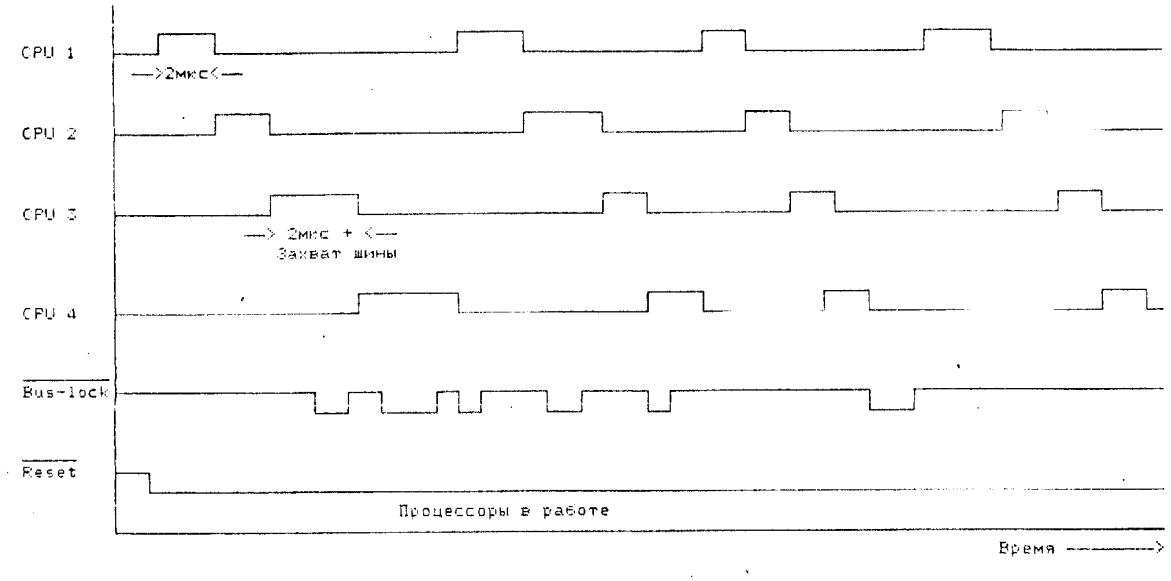
- Сигналы доступа к шине

KOR 923A циклически разрешает доступ к шине контроллера каждому из 2/4 CPU. CPU может использовать шину только в этот момент.

Доступ к шине определяется Функцией временного мультиплексирования (переключения). Номер CPU задается перемычками на плате координатора. Время доступа к S5—шине ограничено 2мс для каждого CPU. Время доступа к шине можно увеличить с помощью захвата шины.

Сразу после сигнала сброса, инициированного источником питания. шина предоставляется первому по порядку CPU и далее последовательно каждому в соответствии с присвоенными порядковыми номерами:

CPU1, CPU2, CPU3, CPU4, CPU1, CPU2, и т.д.



Bus-lock = Захват шины
Reset = Сброс

Рис. 2 Последовательность сигналов управления шиной

1.3.2. Буфер обмена данными

Буфер обмена данными KOR 923A берет на себя Функции маркеров связи (маркеров межпроцессорного обмена). Маркеры связи осуществляют циклический обмен данными между CPU (центральными процессорами) и между CPU и коммуникационными процессорами (CP) в AG 1351.1.

Буфер обмена данными представляет собой память RAM, буферизованную непосредственно от AG.

В руководствах по программированию процессоров объясняется, как программировать эту функцию.

1.4. Технические данные

Класс защиты	IP 00
Рабочая температура	0...55°C
Температура транспортировки и хранения	—40. . .700C
Относительная влажность	95% при 25°C (без конденсации)
макс. высота над уровнем моря	3500 м
Напряжение питания	5В +/—5%
Номинальный потребляемый от источника 5В ток	0.5 А
Напряжение буферной батареи	2.7В от аккумулятора блока питания
Потребляемый от буферной батареи ток	100 нА
Размеры	20.32 x 233.4 x 160 мм
Вес	не более 0.3 Кг

2. Монтаж

2.1. Установка и снятие модулей

Модули снимаются при помощи легкого покачивания вверх/вниз и одновременного с этим выдвижения на себя за ручки. Модули можно вставлять и вынимать, только если контроллер выключен.

2.2. Установочные места в AG 135U

KOR 923 A устанавливаются на 3-е установочное место в AG 135U.

3. Эксплуатация

3.1. Органы управления

Единственным элементом управления координатора является трехпозиционный селектор выбора режима с позициями "RUN", "STOP" и "TEST", расположенный на передней панели модуля.

3.2. Режимы работы

o Состояние останова

Если селектор выбора режима после включения питания находится в положении "STOP" или если имеет место какая-нибудь другая ситуация останова, то все CPU находятся в состоянии останова.

o Новый пуск

Новый пуск происходит в том случае, если селектор режимов координатора переводится из положения "STOP" в "RUN" (обеспечивая тем самым подачу питания), при условии, что все CPU сброшены.

o Новый пуск с запоминанием, перезапуск

После перевода селектора режимов из положения "STOP" в "RUN" все CPU в принципе готовы к циклической работе, но это возможно лишь при условии, что до остановки контроллер работал в цикле, что селекторы режимов всех процессоров — тоже в положении "RUN" и что отсутствует ситуация, требующая остановки контроллера.

o Автоматический новый пуск, автоматический перезапуск

Если при включении литания селектор режимов находится в положении "RUN", то происходит новый пуск с запоминанием (S—процессор —3UAII и —3UA12) или автоматический перезапуск (Б—процессор —3UA21 и R—процессор), но это при условии, что селекторы режимов всех CPU тоже в положении "RUN", и до останова контроллер работал в циклическом режиме.

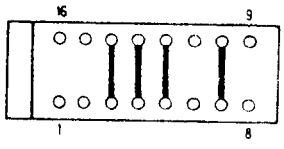
o Тест

Тест-Функция используется для инициализации системы при многопроцессорной работе. Тест-Функция активируется при установке перемычки 3-14 на штеккере 45. При переключении селектора режимов из положения "STOP" в "TEST" контроллер входит в режим циклической работы. Сигнал "BASP" подавляется для тех CPU, которые находятся в состоянии останова благодаря своим селекторам режимов. Однако, некоторые CPU могут быть в состоянии работы без блокирования сигналом "BASP" цифровых выходных модулей.

Если в работе CPU, включенного в "RUN", обнаруживается ошибка, которая вводит его в состояние останова, то Тест—Функция отключается, и выдается сигнал "BASP", блокирующий цифровые модули выходов.

После того, как старт системы полностью подготовлен. Тест должен быть отключен во избежание ошибок управления.

Штеккер 45



Перемычка 3—14 включена:
Тест—функция установлена

Рис. 3 Установка Тест—Функции

3.3. Кодирование числа CPU

Каждому S—, R— и/или M—процессору в многопроцессорном режиме с помощью перемычек на плате координатора 923A присваивается порядковый номер.

Номер CPU	Перемычки штеккера 62
2	7—10, 8—9
3	7—10
4	8—9

При установке в контроллер CPU 92S с другими процессорами необходимо помнить, что CPU 928 всегда устанавливается справа _от остальных процессоров. Если используются два CPU '^28, то для них надо освободить три установочных места, т.е. должна быть установлена перемычка 7-10.

3.4. Адресация буфера обмена данными

Область маркеров межпроцессорной связи занимает в памяти пространство адресов с F200H до F2FFH, всего 256 байт. Она может заполняться блоками по 32 байта. Если коммуникационные процессоры не используются, вся эта область доступна для контроллера и может использоваться им для Функций маркеров межпроцессорной связи.

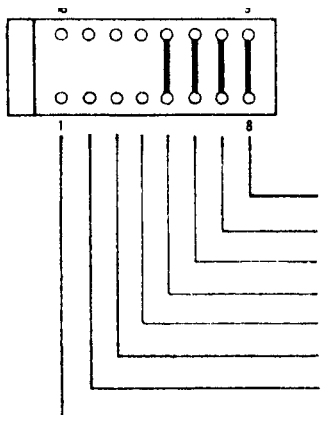
Удалением перемычек на штеккере 7 один или несколько 32—байтовых блоков могут маскироваться, т.е. защищаться от записи.

В случае использования маркеров межпроцессорной связи в CP должно применяться маскирование соответствующих областей в координаторе.

Пример:

Необходимо маскировать 4 32-байтовых блока маркеров связи со старшими адресами:

Штеккер 7



Маркеры связи Байтовая область	Адреса (шестнадцатирич.)
0 ... 31	F200H ... F21FH
32 ... 63	F220H ... F23FH
64 ... 95	F240H ... F25FH
96 ... 127	F260H ... F27FH
128 ... 159	F280H ... F29FH
160 ... 191	F2A0H ... F2BFH
192 ... 223	F2C0H ... F2DFH
[illegible]	F2E0H ... F2FFH

3.5. Назначение перемычек

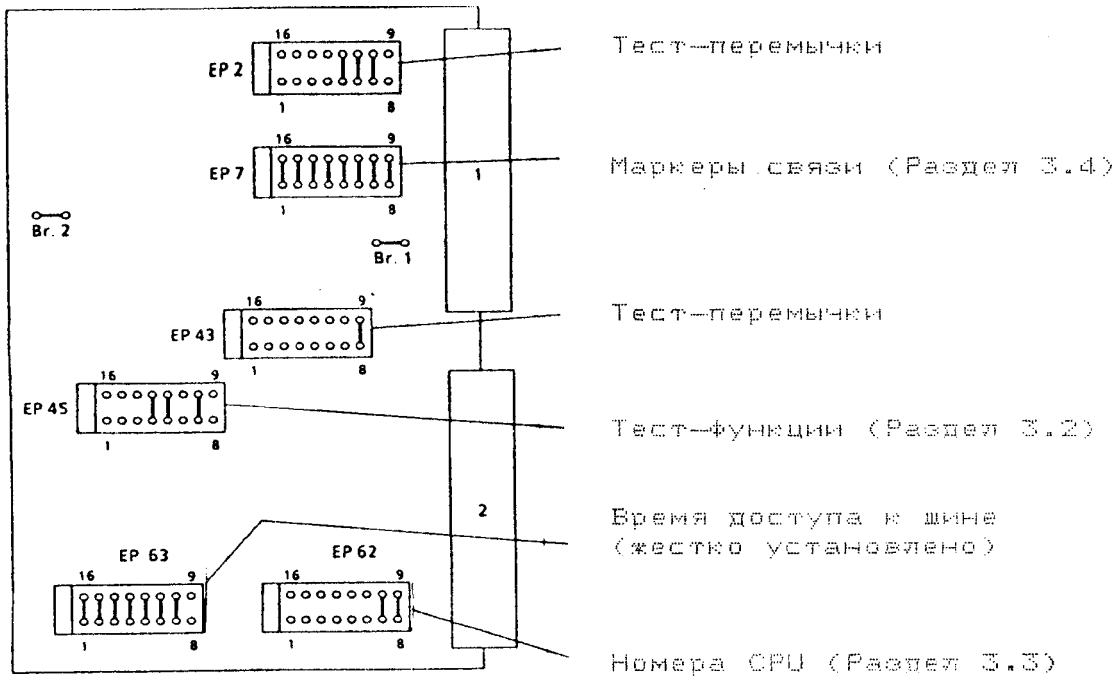


Рис. 4 Размещение перемычек и копирующих штеккеров
(показан вариант стандартной поставки)

4. Назначение контактов задних разъемов

Задний разъем 1

	d	b	z
2		M 5B	+ 5B
4	UBAT		
6		ADB 0	CPKL
8		ADB 1	MEMR
10		ADB 2	MEMW
12		ADB 3	RDY
14	BUSEN 1	ADB 4	DB 0
16	BUSEN 2	ADB 5	DB 1
18	BUSEN 3	ADB 6	DB 2
20	BUSEN 4	ADB 7	DB 3
22		ADB 8	DB 4
24		ADB 9	DB 5
26		ADB10	DB 6
28	DSI	ADB11	DB 7
30			
32		M 5B	HALT

Задний разъем 2

	d	b	z
2		M 5B	+ 5B
4			
6			
8			
10			
12			
14			NAU
16			
18			
20		STEU	
22		STOPPA	
24			
26	TEST		
28		PERO	
30			
32		M 5B	

5. Комплектующие

Кодирующий штеккер C79334—A3011—B12

SIEMENS

SIMATIC S5

Координатор 923C ***6ES5 923-3UC11***

Руководство по эксплуатации Заказной No. C79000-B8500-C349-05

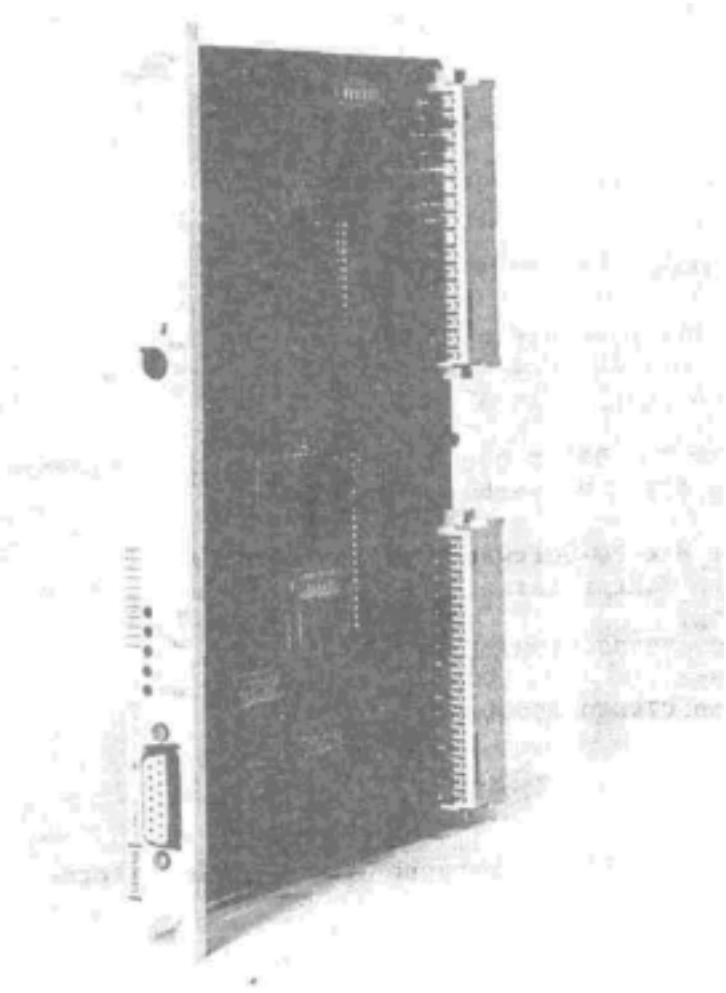


Рис.1 Координатор 923 C

Оглавление **Страница**

1 Техническое описание
1.1 область применения
1.2 Структура
1.3 Принцип действия
1.3.1 Арбитраж шин
1.3.2 Контроль времени доступа к шине
1.3.3 Буфер обмена данными
1.3.4 PG-мультиплексор (PG-MUX)
1.4 Технические данные

2 Монтаж
2.1 Установка и снятие модулей
2.2 Установочные места в контроллере

3 Эксплуатация
3.1 Органы управления и элементы индикации
3.2 Режимы работы
3.3 Предустановка координатора
3.4 Предустановка PG-мультиплексора
3.5 Адресация буфера обмена данными
3.6 Перемычки блокировки сигналов координации
3.7 Регистр ошибок
3.8 Размещение перемычек и кодирующих устройств

4 Обслуживание
4.1 Назначение контактов разъемов
4.2 Комплектующие

1 Техническое описание

1.1 Область применения

Координатор (KOR) 923 C устанавливается в программируемый контроллер (AG) 135U (кроме модификаций -3KA12 и -3KB 12), а также в устройство расширения (EG) 185 .Он обеспечивает выполнение независимых функций:

о Арбитраж шин

Координация работы CPU (центральных процессорных устройств) в многопроцессорном режиме, т.е. при одновременной работе нескольких (от 2 до 4) CPU (S-процессор/CPU921, R-процессор/CPU 922 M-процессор / CPU 920, CPU 928 или CPU 946 / 947) .

о Буфер обмена данными.

Это Функция обмена данными между процессорами

о Мультиплексор программатора (PG - MUX)

Централизованный доступ с соответствующих программаторов (PG по последовательному интерфейсу KOR 923C к последовательным PG- интерфейсам максимум 8 модулей AG.

Эта Функция позволяет связать KOR 923 C через коммуникационный процессор (CP) 535 с фирменной шиной SINEC H1.

Для управления PG- мультиплексором необходима операционная система S5-DOC.

1.2 Структура

KOR 923 C представляет собой печатную плату двойного стандартного европейского формата .

На задней стороне модуля имеются два 48-полюсных разъема типа "2 ряда" для связи с внутренней S5-шиной контроллера.

Ширина передней панели – 1 1/3 размера стандартного установочного места.

В верхней трети передней панели находится гнездо с крышкой. Открыв крышку, Вы получаете доступ к кодирующим штекерам, с помощью которых можно произвести установку параметров модуля.

На передней панели имеется также трех позиционный переключатель режима работы (селектор режимов).

Здесь же 5 небольших светодиодов, которые служат для индикации
Ошибок и ошибочного состояния.

Координатор 923 C через передний 15-полюсный разъем можно соединить с программатором, прибором диагностики, панелью оператора или коммуникационным процессором.

1.3 Принцип действия

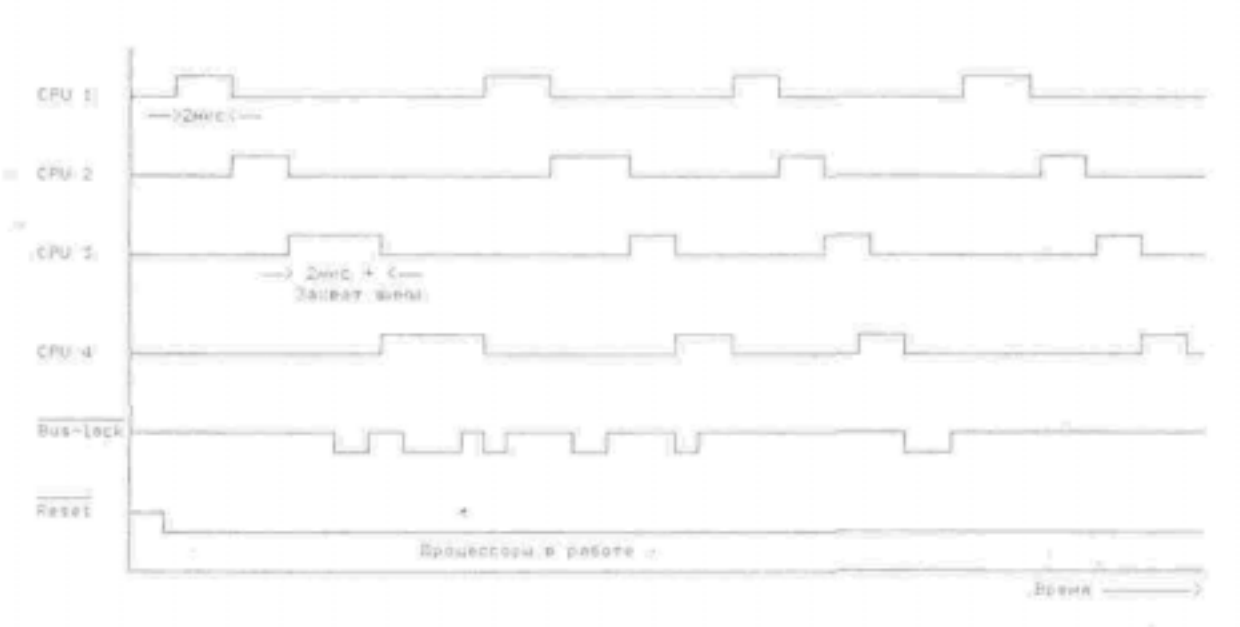
1.3.1 Арбитраж шин

- Сигналы доступа к шине

KOR 923 C циклически разрешает доступ к внутренней шине каждому из 2...4 CPU в AG 135U или 155U. Процессор может использовать шину только во время своего доступа к ней.

Доступ к шине определяется функцией временного мультиплексирования (переключения). Номер процессора задается перемычками на кодирующем штекере передней панели. Время доступа к S5-шине ограничено для каждого процессора 2-мя мкс. Время доступа к шине можно увеличить с помощью захвата шины.

Сразу после сигнала сброса, инициированного источником питания, шина предоставляется первому по порядку процессору и далее последовательно каждому в соответствии с присвоенными порядковыми номерами:
CPU 1, CPU 2, CPU 3, CPU 4, CPU 1, CPU 2 и т.д.



Bus-lock = Захват шины
Reset = Сброс

Рис. 2 Последовательность сигналов управления шиной.

1.3.2 Контроль времени доступа к шине

Сигнал захвата шины декодируется из 4-к старших адресных битов и равен 0, когда все четыре старших бита равны 1 (адресное пространство от 60 к до б4к). Сигнал выдается самим CFU после того, как он уже получил разрешение на доступ к шине от координатора. Время доступа CPU к шине увеличивается на время длительности сигнала захвата шины (см. Рис. 2). Фирма в стандартной поставке устанавливает контроль длительности времени захвата шины на уровне 2мс. Если этот сигнал превышает 2мс, то координатор 923С останавливает все CPU.

Номер процессора, который превысил максимальное время захвата шины помещается в регистр по адресу FEFFH, который доступен контроллеру, при этом загорается светодиод "BUS FAULT" на передней панели координатора. Этот регистр и светодиод сбрасываются, когда пропадает сигнал, вызывающий состояние останова.

1.3.3 Буфер обмена данными

Буфер обмена данными KOR 923 C берет на себя функции маркеров межпроцессорной связи. Маркеры межпроцессорной связи осуществляют циклический обмен данными между CPU в AG 135 U.

Буфер обмена данными содержит четыре свободных страницы памяти (буфера связи) для обмена блоками данных между CPU.

В руководствах по программированию процессоров объясняется, как программировать эти функции.

Буфер обмена данными представляет собой память RAM, буферизованную непосредственно от AG .

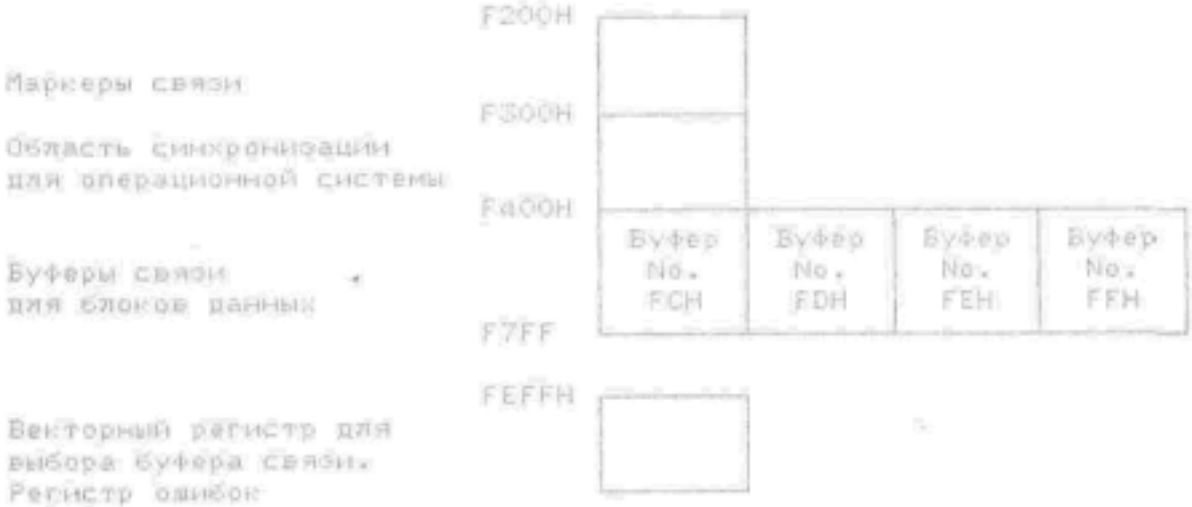


Рис –3. Распределение памяти буфера обмена данными по S5-шине

- Процедуре адресации буфера связи (векторного регистра).

Векторный регистр используется для выбора отдельных областей в общем пространстве памяти. Это 8-битовый регистр, который может быть записан по адресу FHFFH.

Имеются четыре буфера связи, каждый объемом 1К. Каждому буферу присвоен идентификационный номер. Эти номера установлены в координаторе и не могут быть изменены.

Каждый доступ к буферу связи предваряется загрузкой соответствующего номера в векторный регистр.

Эти номера не должны использоваться в других модулях того же контроллера, чтобы избежать дублирования адресов.

В момент включения блока питания векторный регистр стирается , то есть он содержит номер 0Н.

Для обмена данными с этой областью памяти используются специальные процессорные функции которые описаны в руководстве по программированию R-процессора . S-процессор А, 6ES5 921- 3UA11не может оперировать с этими функциями.

1.3.4 PG- мультиплексор (PG-MUX)

TTY - интерфейс на передней панели координатора 923 С может подключать к 8-ми последовательным интерфейсам в контроллере. Эту функцию выполняет микропроцессор 8031 с помощью мультиплексора.

Эти мультиплексированные интерфейсы имеют TTL- уровень и соединяются с другими модулями через задний разъем 2 и шину контроллера.

- Процедура выбора последовательного интерфейса

Все модули контроллера обслуживаются мультиплексором в соответствии с присвоенными им номерами (от 1 до 31 в десятичном счислении). Минимальный номер (базовый адрес)' устанавливается в двоичном коде на кодирующем штекере S2. Этим максимум 8 номерам соответствуют в AG 135 U установочные места под номерами 11, 19, 27, 35, 43, 51, 59 и 67 (младший номер у 11-го установочного места).

Все 8 номеров (или установочных мест) назначаются кодирующим штекером S3; младший номер переключателем S 3.1, старший - S 3.8.

Если какие то места не заняты, или предполагается с какими то модулями работать через их передние интерфейсные разъемы, то соответствующие места необходимо маскировать при помощи штекера S3»

Передний разъем модуля, который предполагается использовать в режиме мультиплексирования, не должен быть занят.

3 Эксплуатация

3.1 Органы управления и элементы индикации

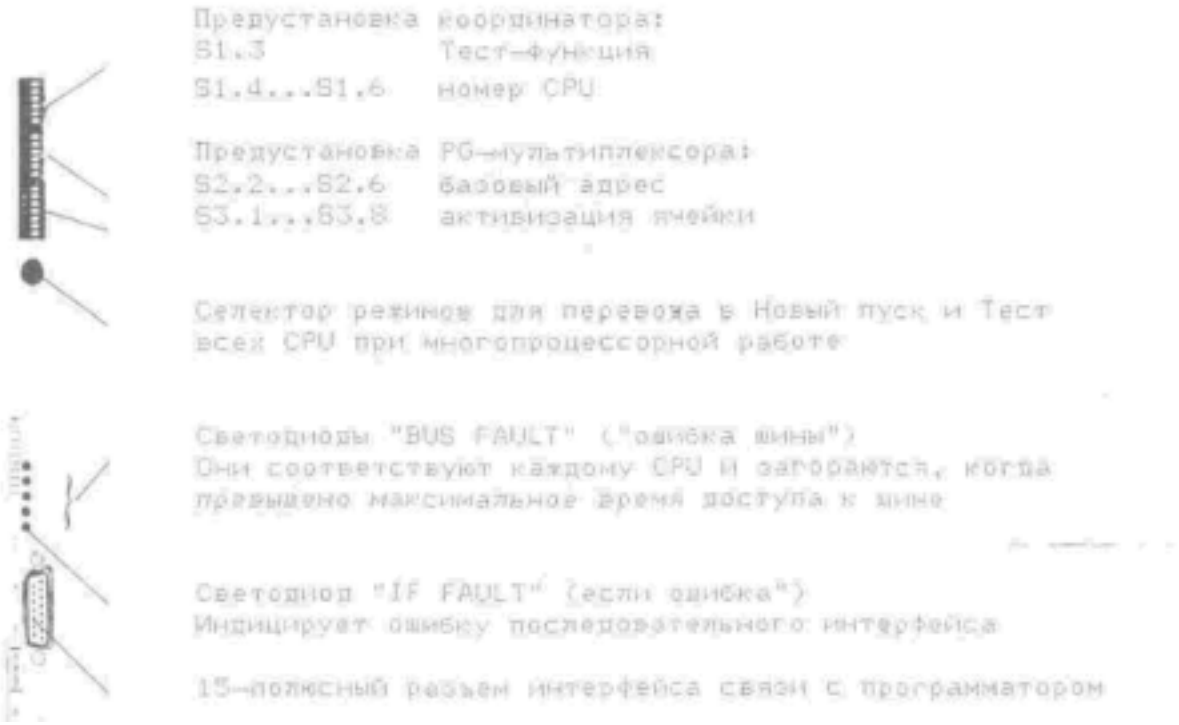


Рис. 4 Передняя панель координатора KOR 923 C

3.2 Режим работы

Состояние останова

Если селектор выбора режима после включения питания находится в положении “STOP” или если имеет место какая-нибудь другая ситуация, то все CPU находятся в состоянии останова.

Новый пуск

Новый пуск происходит в том случае, если селектор режимов координатора переводится из положения “STOP” в “RUN” (после подачи напряжения питания), при условии, что все CPU перед этим сброшены.

Новый пуск с запоминанием, перезапуск

После перевода селектора режимов из положения “STOP” в “RUN” все CPU в принципе готовы к циклической работе, но это возможно лишь при условии, что до остановки контроллер работал в цикле, что селекторов режимов всех процессоров – тоже в положении “RUN”и что отсутствует ситуация, требующая остановки контроллера.

Автоматический новый пуск, автоматический перезапуск

Если при включении питания селектор режимов находится в положении “RUN”, то происходит новый с запоминанием (S-процессор – 3UA11 и –3UA12) или автоматический перезапуск (S-процессор – 3UA11 и R-процессор), но это при условии, что селекторы режимов всех CPU тоже в положении “RUN” и до останова контроллер работал в циклическом режиме.

Тест

Тест-функция используется для инициализации системы при многопроцессорной работе. Тест-функция активируется при установке переключателя S1.3 кодирующего штекера. При переключении селектора режимов из положения “STOP” в “TEST” контроллер входит в режим циклической работы. Сигнал “BASP” подавляется для тех CPU, которые находятся в состоянии останова благодаря своим селекторам режимов. Однако некоторые CPU могут быть в состоянии работы без блокирования сигналов “BASP” цифровых выходных модулей.

Если в работе CPU, включенного в “RUN”, обнаруживается ошибка, которая вводит его в состояние останова, то Тест-функция отключается, и выдается “BASP”-сигнал, блокирующий цифровые модули выходов, работающие с этим CPU. Другие CPU при этом продолжают работу в циклическом режиме.

После того, как пуск системы полностью подготовлен, Тест должен быть отключен во избежание ошибок управления.

3.3 Предустановка координатора

Стандартная поставка:

	ВКЛ	ВЫК	
S1.1		X	
2		X	
3		X	Тест (см. Раздел 5.2)
4	X		количество CPU = 2 *)
5		X	количество CPU = 3 *)
6		X	количество CPU = 4 *)

Может быть включён только 1 из 3 переключателей, т.к. они показывают количество процессоров, установленных в контроллере.

3.4 Предустановка PG-мультиплексора

o Базовый адрес

Кодирующий штекер S2 предназначен для установки базового адреса, который может иметь значение между 1 и 31 (он необходим для адресации процессоров контроллера, имеющих выход на внешний разъем координатора). Мультиплексируемый модуль может иметь базовый адрес или любой иэ 7 следующих адресов. Базовый адрес вычисляется как сумма двоичных разрядов, выбранных кодирующими переключателями.

	выкл	вкл	
S2.1	X		–
2	X		16
3	X		8
4	X		4
5	X		2
6		X	1

(базовый адрес=1)

o Активизация адресов.

Адреса модулей и номера установочных мест, которые управляются координатором KOR 923 C, устанавливаются кодирующим штекером S3.

	выкл	вкл		Установочное место (в AG 135U)
S3.1	X		базовый адрес + 0	11/CPU 1
2	X		базовый адрес + 1	19/CPU 2
3	X		базовый адрес + 2	27/CPU 3
4	X		базовый адрес + 3	35/CPU 4
5	X		базовый адрес + 4	43
6	X		базовый адрес + 5	51
7	X		базовый адрес + 6	59
8	X		базовый адрес + 7	67

Пример: Необходимо установить базовый адрес 10 для модулей контроллера 135U, расположенных в ячейках 11, 35, 43 и 67, управляемых координатором KOR 923 C .

Установка базового адреса:

	вык	вкл	
S2.1	X		1
2	X		16
3		X	8 ——) 8
4	X		4
5		X	2 ——> +2
6	X		1
			10 ——> базовый адрес

Активизация требуемых мест установки:

	вык	вкл		Устан. место (в AG 135U)	Номер управляемого места	Адрес
S3.1		X	баз.ад.+ 0	11	11	10
2	X		баз.ад.+ 1	19		
3	X		баз.ад.+ 2	27		
4		X	баз.ад.+ 3	35	35	13
5		X	баз.ад.+ 4	43	43	14
6	X		баз.ад.+ 5	51		
7	X		баз.ад.+ 6	59		
8		X	баз.ад.+ 7	67	67	17

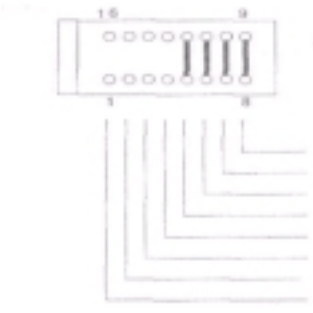
3.5 Адресация буфера обмена данными

Область маркеров межпроцессорной связи (маркеров связи) занимает в памяти пространство адресов с F200H до F2FFH, всего 256 байт. Она может заполняться зонами по 32 байта. Если коммуникационные процессоры не используются, вся эта область доступна для контроллера и может использоваться им для функций маркеров связи.

Размещением перемычек на штекере 60 одна или несколько 32-байтовых зон могут маскироваться, т.е. защищаться от записи.

В случае использования коммуникационными процессорами маркеров связи должно применяться маскирование соответствующих областей в координаторе.
Пример:
Необходимо маскировать 4 32-байтовых зоны маркеров связи со старшими адресами:

Штекер 60

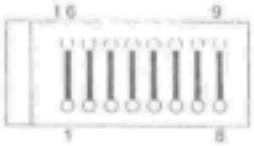


Маркеры связи Байтовая область	Адреса (шестнадцатирич.)	
0 ... 31	F200H ... F21FH	
32 ... 63	F220H ... F23FH	
64 ... 95	F240H ... F25FH	
96 ... 127	F260H ... F27FH	
128 ... 159	F280H ... F29FH	}
160 ... 191	F2A0H ... F2BFH	}-маска
192 ... 223	F2C0H ... F2DFH	}
224 ... 255	F2E0H ... F2FFH	}

3.6 Перемычки блокировки сигналов координации

Координатор 923C при работе в качестве PC --мультиплексора может устанавливаться в устройстве расширения EG 185 U и EG 186U В этом режиме координационные сигналы должны быть отключены.

штекер 61

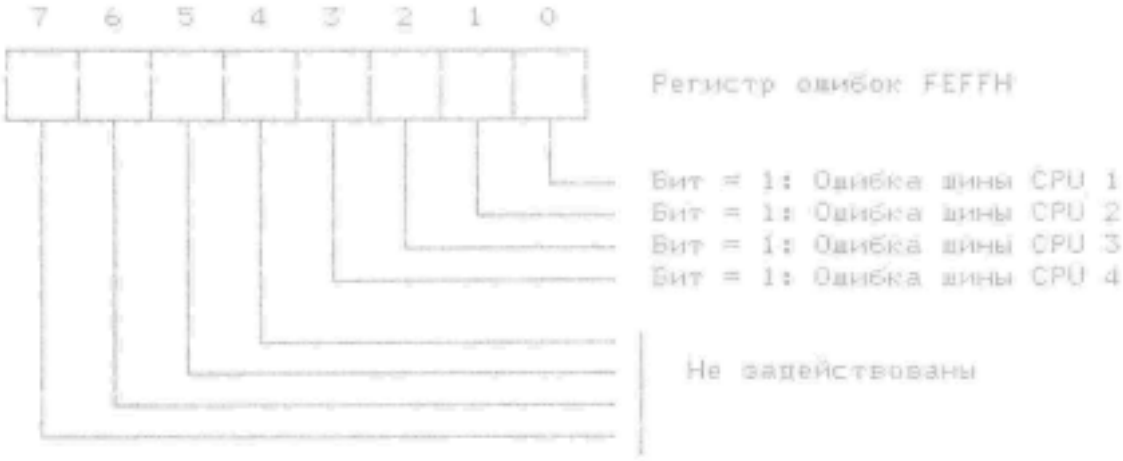


Координация	
действует	блокирована
все перемычки замкнуты	все перемычки разомкнуты

В состоянии поставки все перемычки замкнуты.

3.7 Регистр ошибок

Регистр ошибок имеет 8 разрядов и расположен по адресу FEFFH. Обращение к нему происходит если появляется ошибка шины. Каждому CPU в регистре ошибок соответствует один бит, и он устанавливается в 1, если на шине данного CPU обнаружена ошибка. Регистр стирается каждый раз, когда сигнал останова будет активирован.



Регистр ошибок может быть считан любым CPU, чтобы выполнить центральные функции.

3.8 Размещение перемычек и кодирующих штеккеров

На рисунке показано стандартное положение перемычек

Предустановка координатора

(см. Раздел 3.3.)

Предустановка PG -мультиплексора

(см. Раздел 3.4)

X_4. . .X_6 - тест – перемычки

Штеккер 60: Адресация буфера
обмена данными (см. Раздел 3.5)

Штеккер 61: Блокировка сигналов
Координации (см. раздел 3.6)

Штеккер 62: Контроль шины
(см. раздел 1.3.2)

Штеккер 63: арбитраж шины
(см. Раздел 1.3.1)

Штеккер64: - тест - перемычек
Положение перемычек X_4 . . .X_6 и штекеров 62. . . 64 изменять нельзя.

4. Обслуживание

4.1. Назначение контактов задних разъемов

Задний разъем 1

	d	b	z
2		M 5B	+ 5B
4	UBAT		
6	ADB 12	ADB 0	CPKL
8	ADB 13	ADB 1	MEMR
10	ADB 14	ADB 2	MEMW
12	ADB 15	ADB 3	RDY
14	BUSEN 1	ADB 4	DB 0
16	BUSEN	ADB 5	DB 1
18	BUSEN	ADB 6	DB 2
20	BUSEN	ADB 7	DB 3
22		ADB 8	DB 4
24		ADB 9	DB 5
26		ADB 10	DB 6
28	DSI	ADB 11	DB 7
30		BASP	
32		M 5B	HALT

Задний разъем 2

	d	b	z
2		M 5B	+ 5B
4			
6	RXD 8		
8	TXD 8		
10	RXD 7		
12	TXD 7	RXD 6	
14	RXD 5	TXD 6	NAU
16	TXD 5	RXD 4	
18	RXD 3	TXD 4	
20	TXD 3	STEU	
22	RXD 1	STOPA	
24	TXD 1	RXD 2	
26	TEST	TXD 2	
28		PERO	
30			M 24B
32		M 5B	+ 24B

Передний разъем

Контакт	Назначение
1	Масса (M_{zht})
2	Приемник TTY (-)
3	PL [1])
4	+24B
5	PL [1])
6	Передатчик TTY (+)
7	Передатчик TTY (-)
8	Масса (M_{zht})
9	Приемник (M_{zht})
10	24B масса (источник тока (-) 20 мА)
11	Источник тока (+) 20 мА
12	PL [1])
13	Источник тока (+) 20 Ма
14	PL [1])
15	PL [1])

PL [1]) = резерв

4.2. Комплектующие

Кодирующий штеккер	C79334-A3011-B12
Передняя заглушка	C79451-A3079-C251

SIEMENS

SIMATIC S5

CPU 928

Руководство по программированию Каталог № C79000-B8500-C633-01

Оглавление

1. AG S5-135U – принцип действия и область применения…

1.1. Новые свойства функции CPU 928

2. Программа пользователя

2.1. Язык программирования STEP5

2.1.1. Виды представления KOP, FUP, AWL

2.1.2. Структурированное программирование

2.1.3. Инструкции языка STEP5

2.1.4. Представление чисел

2.1.5. Блоки языка STEP5

2.2. Организационные, программные и шаговые блоки

2.2.1. Программирование

2.2.2. Вызов

2.2.3. Специальные организационные блоки

2.2.4. Организационные блоки спецфункций

2.3. Функциональные блоки

2.3.1. Структура функциональных блоков

2.3.2. Программирование функциональных блоков

2.3.3. Вызов и параметрирование функциональных блоков

2.3.4. Специальные функциональные блоки

2.4. Блоки данных

2.4.1. Структура блока данных

2.4.2. Программирование блока данных

2.4.3. Открытие блока данных

2.4.4. Специальные блоки данных

3. Обработка программ

3.1. Обзор

3.1.1. Организация программы

3.1.2. Хранение программы

3.1.3. Обработка программы пользователя на языке STEP5

3.1.4. Предустановки для обработки программ

3.2. Набор операций языка STEP5 с примерами программ

3.2.1. Основной набор операций

3.2.2. Дополнительный набор операций

4. Режим работ

4.1. Режим работ и уровни обработки программ

4.2. Режим работы стоп

4.3. Режим работы запуск

4.3.1. Новый пуск или ручной перезапуск

4.3.2. Автоматический перезапуск

4.3.3. Прерывания во время запуска

4.4. Режим работы обработка

4.4.1. ЦИКЛ: Циклическая обработка программы

4.4.2. ПРЕРЫВАНИЕ ПО ВРЕМЕНИ: Обработка программ управляемых по времени

4.4.3. ПРЕРЫВАНИЕ ОТ РЕГУЛЯТОРА: обработка регулятора

4.4.4. ПРЕРЫВАНИЕ ОТ ПРОЦЕССА: Обработка программы управляемой по прерыванию от процесса

5. Обработка прерываний и ошибок

5.1. Наиболее распространенные ошибки в программе пользователя

5.2. Обработка информации об ошибках

5.3. Управляющие биты и стек прерывания (USTACK)

5.4. Обработка ошибок с помощью организационных блоков

5.5. Ошибки при ЗАПУСКЕ

5.5.1. DB0-FE (ошибки DB0);

5.5.2. DB1-FE (ошибки DB1);

5.5.3. DB2-FE (ошибки DB2);

5.5.4. DX0-FE (ошибки DX0);

5.6. Ошибки в режимах ОБРАБОТКА и ЗАПУСК

5.6.1. BCF (Ошибка кода команды)

5.6.2. LZF (Ошибка выполнения команды)

5.6.3. ADF (Ошибка адресации)
5.6.4. QVZ (Задержка квитирования)
5.6.5. ZYK-FE (Ошибка времени цикла)
5.6.6. WECK-FE (Ошибка “Будильника”)
5.6.7. REG-FE (Ошибка регулятора)
5.6.8. ABR (Прерывание)

6. Встроенные спецфункции

6.1. Работа с регистрами
6.1.1. Обращение к байту индикации (OB 110)
6.1.2. Стирание аккумуляторов АКК 1, АКК 2, АКК 3, и АКК 4 (OB111)
6.1.3. Пролистывание аккумуляторов вперед (OB 112) и назад (OB 113)
6.2. Команды управления структурой
6.2.1. Счетчик времени цикла (OB 160…OB 163)
6.3. Считывание стека блоков
6.4. Обработка блоков
6.4.1. Управляемый доступ к блоку данных (OB 130)
6.4.2. Тестирование блоков данных DB/DX (OB 181)
6.4.3. Передача маркеров в блок данных (OB 190 и ОБ 192)
6.4.4. Передача из блока данных в область маркеров (OB 191 и OB 193)
6.4.5. Передача блоков данных в область памяти DB-RAM (OB 254, 255)
6.5. Многопроцессорная связь (OB200…OB205)
6.6. Доступ к буферам связи
6.6.1. Запись в буфер связи (OB 216)
6.6.2. Чтение из буфера связи (OB 217)
6.6.3. Захват буфера связи (OB 218)
6.7. Расширение знак
6.8. Системы функции
6.8.1.
6.8.2.
6.8.3. Установка времени цикла (OB 221)
6.8.4. Новый запуск времени цикла (OB 222)
6.8.5. Проверка видов запуска процессоров (OB 223)
6.8.6. Передача маркера межпроцессорной связи в блок (OB 224)
6.8.7. Чтение слова из системы программы (OB 226)
6.8.8. Считывание контрольной суммы (OB 227)
6.8.9. Чтение информации о состоянии и уровне выполняемой программы (OB 228)

6.9. Функции для стандартных функциональных блоков (OB230…OB237)
6.10. Регистр сдвига
6.10.1. Инициализация регистра сдвига (OB240)
6.10.2. Обработка регистра сдвига (OB241)
6.10.3. Стирание регистра сдвига (OB242)
6.11. Регулирование: PID- алгоритмы
6.11.2. Инициализация PID-алгоритма (OB250)
6.11.3. Обработка данных по PID-алгоритму (OB251)

7. Предустановки системных функций в блоке данных DX 0

8. Заполнение и организация памяти
8.1. Распределение адресного пространства в CPU928
8.1.1. Распределение адресного пространства системной области RAM
8.1.2. Распределение адресного пространства периферии
8.1.3. Организация памяти в CPU 928
8.1.4. Заголовок блока в памяти пользователя и DB-RAM
8.1.5. Список адресов блоков в блоке данных DB0
8.1.6. Область BA/BB
8.1.7. Область BS/BT

9. Обращение к памяти по абсолютным адресам
9.1. Обращение к регистрам и памяти по адресу в аккумуляторе 1
9.2. Передача блоков памяти
9.3.Операции под регистром BR
9.3.1. Загрузка BR-регистров
9.3.2. Сдвиг содержимого регистров
9.3.3. Обращение к локальной памяти
9.3.4. Обращение к глобальной памяти
9.3.5. Обращение к буферам связи

10. Многопроцессорный режим работы
10.1. Общие замечания
10.2. Обмен данными между процессорами
10.2.1. Маркеры связи
10.2.2. Межпроцессорные связи

10.2.3. Передача взаимозависимых блоков данных
10.3. Разделение доступа к шине периферийных устройств
10.3.1. Блок данных DB1
10.4. Запуск в многопроцессорном режиме
10.5. Тестовый режим

11. Вспомогательные средства тестирования: On-line- функции
11.1. On-line- функция “СТАТУС ПЕРЕМЕННЫХ”
11.2. On-line- функция “СТАТУС”
11.3. On-line- функция “КОНТРОЛЬ ОБРАБОТКИ”
11.4. On-line- функция “УПРАВЛЕНИЕ”
11.5. On-line- функция “УПРАВЛЕНИЕ ПЕРЕМЕННЫМИ”
11.6. On-line- функция “СЖАТИЕ ПАМЯТИ”
11.7. Функции PG “СТАРТ/СТОП”
11.8. On-line- функция “ПОЛНОЕ СТИРАНИЕ AG”
11.9. On-line- функция “ ВЫВОД АДРЕСА”
11.10. On-line- функция “ЗАПОЛНЕНИЕ ПАМЯТИ”
11.11. Таблица: Действие на точках контроля

ПРИЛОЖЕНИЯ
A Технические данные AG S5-135U
B Обзор меток ошибок
C Обзор операций STEP-5
D Обзор операций языка STEP-5 (в алфавитном порядке)
E Команды языка STEP-5 (отсортированные по кодам команд)
F Команды языка STEP-5, которых нет в CPU928
G Обзор меток уровней обработки программ
H Пример использования стекает прерываний

Глоссарий

Список рисунков, примеров и обзоров.

Важно:

Предлагаемое руководство по программированию действительно для CPU 928 с MLFB-номером –3UA12 (12 МГц)

Что где искать?

Глава 1 описывает в общем принципы работы и внутреннюю структуру процессора. Дан набросок типичной структуры установки с использованием программируемого контроллера AG S5-135U и указания на новые свойства и функции CPU 928.

Глава 2 поясняет структуру программы пользователя и подробнее останавливается на особенностях языка программирования STEP5. В заключение дана характеристика различных программных блоков STEP5 и описано их программирование.

Глава 3 содержит информацию по циклической обработке программ в CPU 928, по организации программы и по хранению программ в памяти. Описан весь набор команд языка STEP5 и приведено множество примеров программирования. (Дополнительную информацию о командах языка STEP5 вы найдете в списке операций языка STEP5. Кроме того обратите внимание на список литературы).

Глава 4 описывает различные режимы работы CPU 928 (Запуск, Работа, Останов) и определяет понятие «Уровень обработки программ». Пояснены возможности уровней обработки программ в разных режимах работы процессора. Кроме того, здесь вы найдете важную информацию по обработке программ, управляемых по времени и по сигналам от процесса.

Глава 5 подробно излагает тему «Поиск и обработка ошибок». Приведены типичные ошибки при запуске и работе, и даны указания как обнаружить ошибку и как на нее реагировать. Пояснены на примерах структура и использование стека прерываний (USTACK)

Глава 6 описывает встроенные спецфункции CPU 928 и содержит множество примеров программы пользователя.

Глава 7 посвящена структуре и программированию блока данных DX0, с помощью которого вы можете легко изменить по своему желанию некоторые функции CPU 928 (тоже с примерами программ)

Глава 8 содержит детальную информацию по отдельным областям памяти CPU 928. Знакомый с операционной системой пользователь найдет размещение системных данных.

Глава 9 интересна для опытных пользователей, которые уже имеют хорошие системные знания. Она содержит все команды языка STEP5, с помощью которых вы сможете обращаться по абсолютным адресам к любой области памяти и работать с различными регистрами CPU 928/

Глава 10 даст вам дополнительную информацию о многопроцессорном режиме и описывает структуру и программирование блока данных DB1, который необходим для многопроцессорного режима. Кроме того, пояснены особенности тестового режима.

Глава 11 описывает функции ON-LINE, которые вы можете вызвать на программаторе для тестирования вашей программы, и указывает на особенности, связанные с CPU 928.

Сокращения:

ABBR	Прерывание
ADF	Ошибка адресации
AG	Устройство автоматизации, программируемый контроллер
AKK1 (2,3,4)-L	Младшее слово аккумулятора 1 (2,3,4), 16 бит
AKK1 (2,3,4)-H	Старшее слово аккумулятора 1 (2,3,4), 16 бит
AKK1 (2,3,4)-LL	Младший байт младшего слова аккумулятора 1,8 бит
AKK1 (2,3,4)-LH	Старший байт младшего слова аккумулятора 1, 8бит
ANZ0, ANZ1	Слово состояния, кодированное
AWL	Список инструкций
BASP	Блокировка выдача команд
BSTACK	Стек блоков
BCD	Двоично кодированное десятичное число
BSF	Ошибка командного кода
CP	Коммуникационный процессор
D, DL/DW, DW/ DD	Данные (1бит), 1байт данных левый/правый (8бит), слово данных (32бита)
DB	Блок данных
DBA	Начальный адрес блока данных (в регистре данных)
DBL	Длина блока данных (в регистре 3)
DX	Дополнительные блоки данных
EPROM	Перепрограммируемое ПЗУ
ERAB	Первый опрос (бит состояния)
FB	Функциональный блок
FUP	Функциональный план
FX	Дополнительный функциональный блок
IP	Модуль интеллигентной периферии
KOP	Контактный план
KOR	Координатор
LZF	Ошибка времени цикла
M, MB, MW, MD	Маркерный бит, байт, слово, двойное слово
OB	Организационный блок
OR	Или (бит состояния)
OS	Overflow с запоминанием (состояние слова)
OV	Overflow (состояние слова)
PA	Отображение процесса
PAA	Область отображения выходов
PAE	Область отображения входов
PB	Программный блок
PB, PW	Байт периферии (программатор PG675), слово периферии
PG	Программатор
CPU	Процессор
PY	Периферийный байт (программатор PG685)
QB, QW	Байт, слово в области дополнительной периферии
QVZ	Задержка квитирования
RAM	Память со свободным доступом
SAZ	Счетчик step-адреса (в регистре 15)
SB	Шаговый блок
SF	Специальная функция
STA	Статус (бит состояния)
T	Таймер
USTACK	Стек прерываний
VKE	Результат логической операции (бит состояния)
Z	Счетчик
ZYK	Ошибка цикла

Список литературы

Введение в программирование на языке STEP5 и пояснения по программированию контроллеров SIMATIC S5-135U вы можете найти в следующих руководствах:

S5-135U Программирование на языке STEP5

SIEMENS AG, ISBN 3-8009-1461-1

(S-и R-процессоры)

Автоматизация с помощью устройства SIMATIC S5-135U

SIEMENS AG, ISBN 3-8009-1522-7

(S- и R-процессоры, а также CPU928)

Кроме того, посмотрите в вашем руководстве по устройству остальные главы, относящиеся к CPU928 (руководство по эксплуатации, список операций языка STEP5)

1. ВВЕДЕНИЕ: AG 135U –принцип действия и область использования

Эта глава предназначена для пользователей, которые в первый раз сталкиваются с использованием программируемых контроллеров, однако, имеются знания по другим микрокомпьютерным системам.

Структура устройства.

Программируемый контроллер (AG) является компьютерной системой, которая разработана специально для компьютерной промышленности, например, для управления деревообрабатывающими станками. ПК построен по модульному принципу и состоит из рамы для установки модулей минимум с одним процессорным модулем, который в дальнейшем коротко называется CPU, и некоторым числом периферийных модулей. Какие и сколько процессоров периферийных модулей может быть установлено, зависит от задач автоматизации.

Программируемый контроллер S5-135U принадлежит к семейству свободно программируемых систем с памятью SIMATIC S5. Он является мощным процессорным устройством для автоматизации производственных процессов (управления, выдачи сообщений, контроля, регулировки, протоколирования) и может использоваться как в простой системе с двоичными сигналами, так и для решения сложных комплексных задач автоматизации.

Центральное устройство S5-135U может иметь по выбору:

—Один процессор в однопроцессорном режиме или

—Координатор (KOR) и до 4 процессоров в многопроцессорном режиме:

—И кроме того коммуникационные процессоры (CP) до 7 CP в однопроцессорном режиме или от 4 до 6 (7) CP в многопроцессорном режиме.

Остальные свободные места в центральном устройстве S5-135U могут быть заняты модулями входов и выходов. Расширение периферии возможно с помощью подключения к центральному устройству устройства расширения (EG).

Обратите внимание на каталог «программируемый контроллер S5-135U ST54.1,каталоговый номер: E86010-K4654-A111-A3.

На рисунке приведена типичная конфигурация устройства S5-135U. Для однопроцессорного режима достаточно модулей, выделенных на рисунке жирным шрифтом.

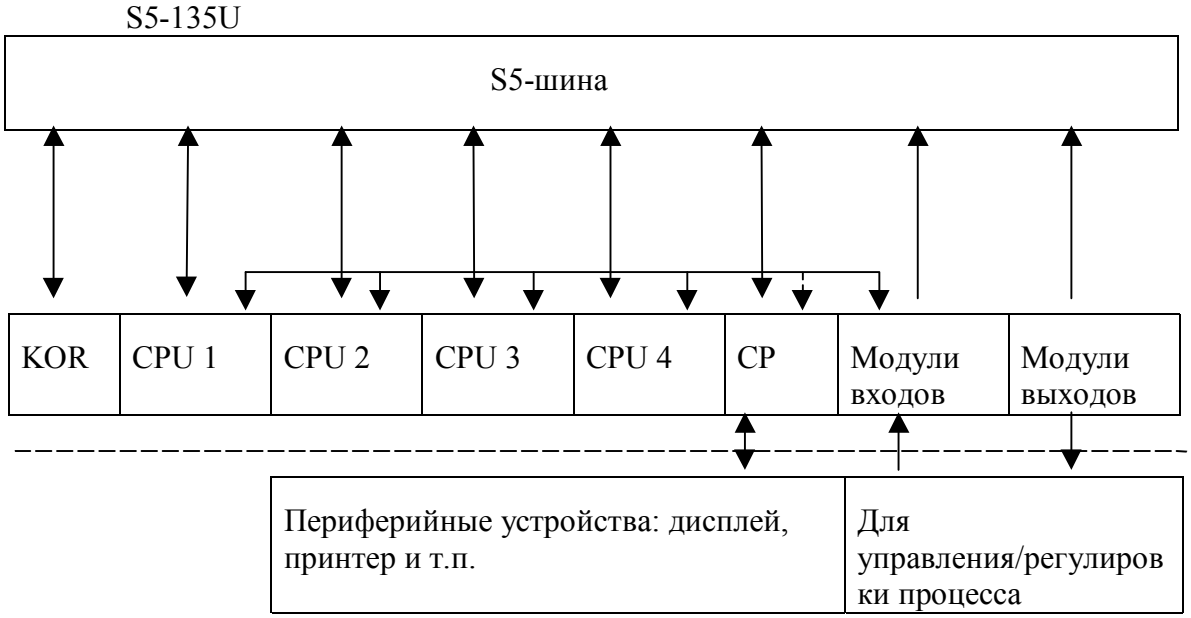


Рис. 1.1. Стандартная структура устройства AG S5-135U.

Область применения.

В однопроцессорном режиме можно использовать устройство для решения простых задач автоматизации. Особенности:

—S-процессор особенно удобен для решения задач управления (быстрая обработка битовых операций)

—R-процессор наиболее подходит для задач регулирования (быстрая обработка слов)

—M-процессор больше всего годится для обработки измеренных значений; программируется на языке Ассемблера и на языках высокого уровня (BASIC, C)

—CPU 928 универсален, устанавливается в случае обработки битовых операций со словами.

Для решения комплексных задач автоматизации, в отличие от многих других AG, центральное устройство S5-135U можно использовать с одновременной установкой нескольких процессоров (как многопроцессорное устройство):

Многопроцессорный режим имеет смысл тогда, когда программа обработки процесса слишком велика для одного процессора, и когда этот процесс можно разбить на несколько независимых друг от друга задач. В этом случае каждая отдельная задача может быть выполнена своим процессором (см. выше). При этом каждый процессор обрабатывает индивидуальную программу независимо от других процессоров.

По общей шине (S5-шина) процессоры обращаются попеременно к модулям периферии. Координатор разделяет доступ процессоров к шине S5 на жестко заданные отрезки времени. Только процессор, который допущен к шине S5, может общаться с периферией.

По шине S5 процессоры могут обмениваться между собой данными. Этот обмен данных производится с помощью промежуточной памяти координатора.

Принцип действия

Процессор постоянно повторяет следующий цикл:

1. Опрашиваются все модули входов, занятые в процессе, и их значения считываются в область отображения входов (PAE).

2. Содержащиеся в PAE значения обрабатываются программой пользователя, и результирующие значения заносятся в область отображения выходов (PAA)

3. Значения выходов, которые находятся в PAA, выдаются процессором в модули выходов.

Время, необходимое процессору для выполнения этих трех задач, называется временем цикла.

Цикл должен выполняться достаточно быстро. Состояние процесса не может изменяться быстрее, чем процессор может на него реагировать. В противном случае процесс может выйти из под контроля. В качестве максимального времени реакции следует брать удвоенное время цикла. Время цикла зависит от вида и объема программы пользователя (см. ниже) и, обычно, не постоянно.

Для процессов, которым необходима обработка информации через определенные промежутки времени, может быть предусмотрена дополнительная программа, управляемая по времени. После обработки циклической программы в течение определенного отрезка времени, она прерывается для обработки программы, управляемых по времени. В CPU 928 можно обрабатывать до 9 программ прерываний по времени. Общее время цикла увеличивается из-за времени обработки программы, управляемой по времени.

Сигнал процесса, на который нужно реагировать особенно быстро, может быть обработан процессором в программе, управляемой по прерываниям. При поступлении сигнала прерывания процессор прекращает обработку циклической программы или программы, управляемой по времени, для обработки программы, управляемой по прерываниям. Время цикла увеличивается из-за времени обработки программы, управляемой по прерыванию.

В некоторых случаях время цикла составляет из времени обработки циклической программы и времён обработки многократно вызываемых программ, управляемых по времени и прерываниям. Каждый процессор контролирует время цикла. Превышение запрограммированного граничного значения ведёт к прерыванию обработки программы, переводит процессор в состояние «СТОП» и блокирует выходные сигналы.

Программы

Имеющаяся в каждом процессоре программа разделяется на программу пользователя и операционную систему.

Программа пользователя для AG 135U создаётся на специально разработанном для программируемых контроллеров языке программирования STEP 5 (исключение – M-процессор). Программа пользователя построена по модульному принципу и состоит минимум из одного программного модуля (блока). Различают два основных типа блоков:

a) Блоки, содержащие коды

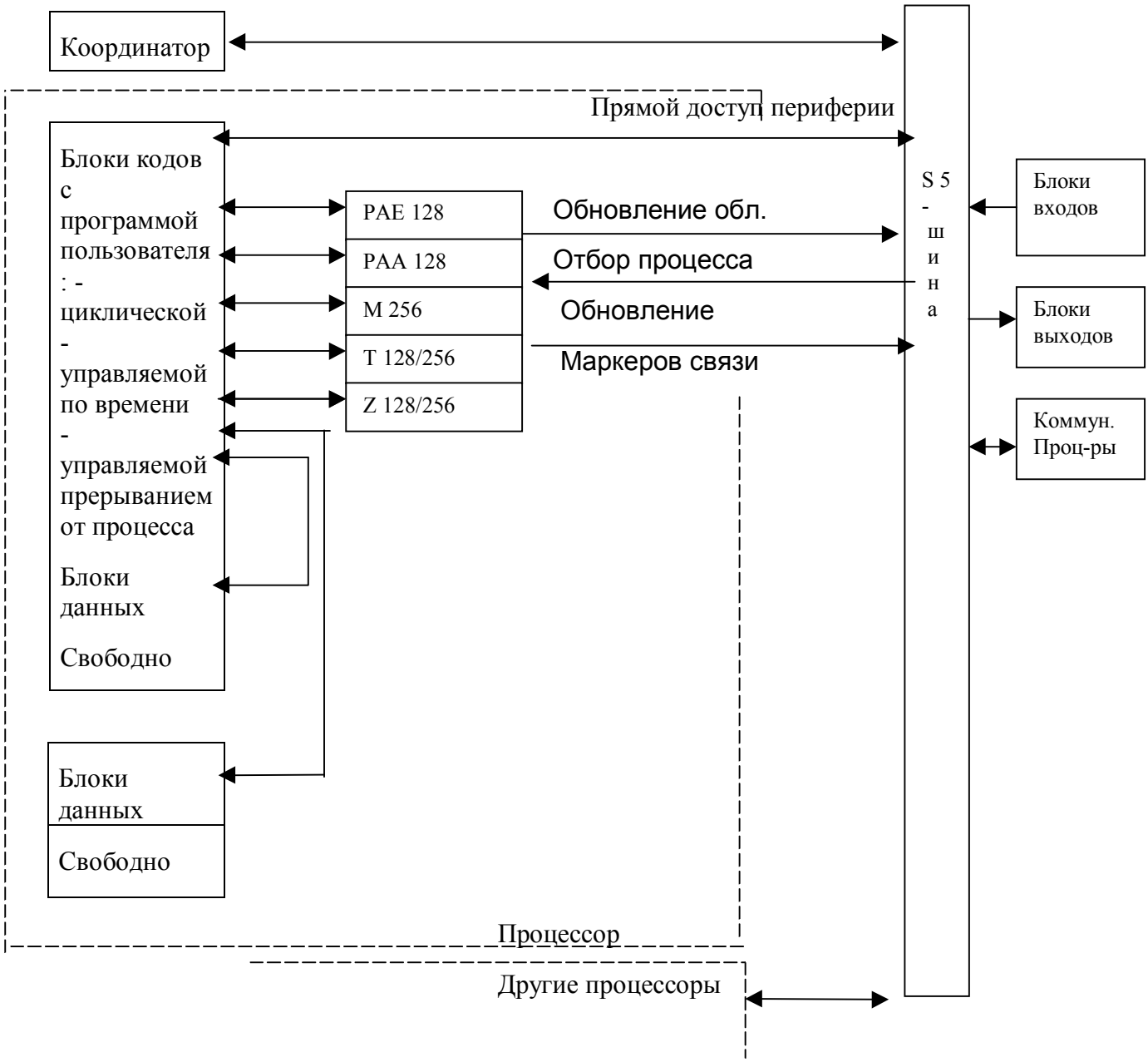
b) Блоки данных: блоки, хранящие константы и переменные для STEP 5 программы.

К операционной системе пользователь не имеет доступа. Операционная система поддерживает все функции программируемого контроллера. К ним относятся:

— обновление области отображения процесса (входов, выходов, маркеров связи),

— обновление таймеров,

— вызов программ, обрабатываемых циклически, либо управляемых по времени и по прерыванию от процесса.



Структура одного процессора S5-135U (многопроцессорный режим).

Внутренняя структура процесса.

Память процессора разделена на множество областей. Важнейшими из них являются:

— память пользователя (максимум 32 килослова).

Память пользователя находится на подключаемом модуле RAM или EPROM и содержит кодовые блоки и блоки данных

— RAM блоков данных (=DB-RAM, максимум 23,375 килослов)

DB-RAM является областью памяти для хранения блоков данных; блоки данных, чье содержание должно изменяться программой пользователя, должны быть скопированы из модуля EPROM в DBRAM.

— Область маркеров M (256 байт).

Область маркеров – это область памяти, время доступа к которой со стороны программы пользователя мало. Область маркеров должна быть использована, прежде всего, для наиболее часто используемых рабочих данных. Возможно обращение к следующим типам данных: отдельные биты, байты, слова и отдельные слова. Отдельные маркерные байты могут использоваться в качестве маркеров связи для обмена данными между процессорами. Маркеры связи обновляются операционной системой в конце цикла с использованием буферной памяти координатора.

— Область отображения входов и выходов PAE/PAA (по 128 байт)

Область отображения процесса программы пользователя может обращаться таким же образом как и к маркерной области. Область отображения процесса обновляется в конце цикла операционной системой.

— Область периферии (512 байт)

Программа пользователя может помимо области отображения процесса непосредственно обращаться к шине S5 к модулям периферии. Возможны следующие типы данных: байты и слова.

— Таймеры T (128 таймеров для S и R процессоров, 256 таймеров для CPU 928)

Таймеры загружаются программой пользователя значением времени в промежутке от 10 мс до 9990 с и обрабатываются операционной системой с частотой 10 мс.

— Счетчики Z (128 счетчиков для S и R процессоров, 256 счетчиков для CPU 928)

Счетчики загружаются программой пользователя начальным значением (максимум 999).

Команды языка STEP5 могут обращаться к следующим областям операндов:

- область маркеров
- область отображения входов и выходов
- область периферии
- таймеры
- счетчики
- текущий блок данных

Для обращения к этим областям операндов используются команды языка STEP 5 двух различных типов:

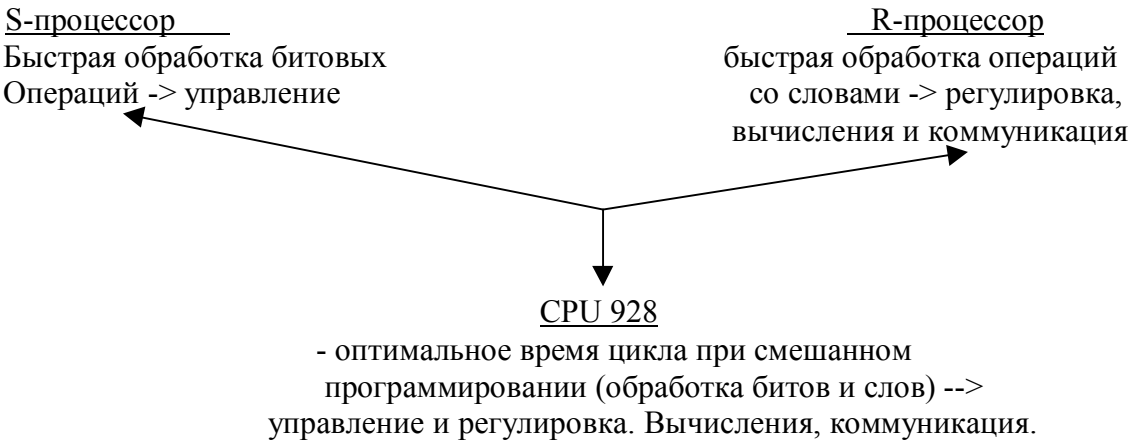
- основная часть команд языка STEP 5 адресует ячейки памяти относительно начала области операндов. Пока вы работаете исключительно с этими командами, программа отделена от области операндов и не может в случае ошибки самостоятельно переписываться
- небольшое количество команд STEP 5 работает с абсолютной адресацией. С помощью этих команд можно производить обращение ко всей области памяти.

По сравнению с другими областям операндов текущий блок данных не имеет твердого начального адреса и длины. Текущий блок данных – это блок данных, чей начальный адреса длина занесены в специальные регистры (см. ниже). Программа пользователя может в случае, если не используются команды с абсолютной адресацией, обращаться только исключительно к текущему блоку данных. Возможны следующие типы данных: отдельные биты, байты, слова и двойные слова. Обращение к текущему блоку данных происходит медленнее, чем обращение к области маркеров.

Кроме вышеназванных областей памяти, процессор содержит несколько регистров:

- 4 аккумулятора (по 32 бита), которые служат в качестве многоцелевых регистров, например, как вспомогательные регистры при передачи память-память или как регистры для операндов и результатов вычислений.
- 1 счетчик команд (счетчик адреса STEP, SAZ), который содержит адрес следующей выполняемой команды.
- 1 регистр хранения блока (указатель стека блоков = BSP), который управляет вводом в стек блоков.
- 1 DBA-регистр (DBA= начальный адрес блока данных), который содержит начальный адрес текущего блока данных.
- 1 DBL-регистр (DBL=длина блока данных), который содержит количество слов данных текущего блока данных.
- 1 регистр состояния.
- 1 BR-регистр (BR-регистр базового адреса), который используется для адресации при абсолютном обращении к памяти.

1.1. Новые свойства и функции CPU 928 (для пользователей S- или R- процессоров)



CPU 928 представляет собой модуль шириной 40мм и занимает, поэтому, два установочных места в центральном устройстве 135U. Он имеет два преимущества S-процессора (быстрая обработка битовых операций, оптимальных для задач управления) и R-процессора (быстрая обработка операций со словами, оптимальных для задач регулировки). Поэтому CPU 928 особенно удобен для задач контроля и выдачи сообщений, для коммуникации в многопроцессорном режиме и для функций диагностики и контроля. Но CPU 928 можно использовать и как универсальное средство для решения различных задач автоматизации.

Если вы уже знакомы с S- или R-процессором S5-135U, - обратите внимание на описание новых свойств и функций CPU 928 в следующих главах (курсив выделяет указания действительные только для версии 3UA12!):

Глава 3.1.1: Организация программ.

Максимальная глубина вложения блоков увеличена до числа ‘62’ (CPU 928-3UA11:’30’, R-процессор: ‘20’).

Максимально допустимое время цикла составляет теперь 6000 мс (CPU 928-3UA11 и R-процессор: 4000 мс)

Глава 3.3: Дополнительные операции.

Набор операций языка STEP 5 расширен. Появились следующие новые команды:

- Команды сложения и вычитания 32-битовых чисел с фиксированной запятой: +D, -D, AAD, DF (системные операции) [1]).
- Команды загрузки и передачи слов в области BB или BT: LBB, TBB, LBT, T BT (дополнительные операции).

[1]) Программирование этих команд зависит от типа PG и от версии операционной системы PG.

Глава 4.4.2: Прерывания по времени.

Вы можете теперь обрабатывать до 9 программ управляемых по времени. Отдельные программы находятся в организационных блоках с OB10 по OB18. Каждый OB вызывается в своём временном растре: OB10, например, обрабатывается каждые 10мс, OB15-каждые 500мс. OB прерываний по времени с более коротким временным растром имеют более высокий приоритет, чем прерывания по времени с более длинным временным интервалом и могут при необходимости в них вкладываться.

Глава 5.6: Ошибки при запуске (ANLAUF) и работе (RUN).

Метки ошибок в аккумуляторах АКК 1 и 2 были дополнены.

Глава 6: Встроенные спецфункции.

В CPU 928 имеются новые спецфункции. Это:
OB110 : доступ к байту индикации
OB111 : стирание содержимого АКК 1, АКК 2, АКК 3 и АКК 4
OB112 : пролистывание аккумуляторов вперед
OB113 : пролистывание аккумуляторов назад
OB120 : Вкл./выкл. «блокировка всех прерываний»
OB121 : Вкл./выкл. « блокировка отдельных прер-й по времени»
OB122 : Вкл./выкл. «задержка всех прерываний»
OB123 : Вкл./выкл «задержка отдельных прер-й по времени.
OB160-OB163 : Счетчики циклов
OB170 :Чтение стека блоков (BSTACK)
OB180 : Переменное обращение к блоку данных
OB181 : тестирование блоков данных
OB190 и OB192: передача маркеров в блоки данных
OB191 и OB193: Передача блоков данных в область маркеров
OB228 :Чтение информации о состоянии уровней обработки программ.

Глава 7 : Блок данных DX0.

В CPU 928 может быть установлено до 256 таймеров и счетчиков (в R-процессоре: 128 счетчиков, 128 таймеров).

Дополнительные параметры обработки программ управляемых по прерываниям от процесса.

Для арифметики с плавающей запятой Вы можете в DX0 задать, должен ли работать процессор с 16-битовой или 24-битовой мантиссой.

Глава 8.1: Разделение области адресации в CPU 928.

Блок данных RAM CPU 928 может быть расширен до 23,375 килослов (R-процессор: 11,125 килослов). Поэтому Вы можете обрабатывать больше блоков данных, чем до сих пор.

Вам предлагается, кроме того, две новых области операндов с длиной в 256 слов, которые могут быть свободно использованы: область BB и BT. Для обращения к этим областям имеются новые команды STEP 5.

Глава 9: Операции с регистром BR.

Область в адресном пространстве, которую имеет смысл опрашивать с помощью команд STEP 5 LIR, TIR, TNB и TNW, в CPU 928 расширена и имеет меньше пробелов.

Глава 9.3 Операции с регистром BR.

Для простой адресации при абсолютном доступе к памяти имеется регистр BR (регистр базового адреса).

Имеются

- новые команды, позволяющие загружать и изменять регистр BR (см. 9.3.1), [1]).
- Новые команды, для сдвига содержимого отдельных регистров (см.9.3.2.), [1]).
- Новые команды, позволяющие доступ к локальным и глобальным областям памяти (см. 9.3.3. и 9.3.4.), [1]).
- Новые команды для доступа к буферам связи, [1]).

ПРИЛОЖЕНИЕ A и список операций:

В версии 3UA12 (12 МГц) -по сравнению с версией 3UA11-улучшено время выполнения команд CPU 928 на 1/3.

ПРИЛОЖЕНИЕ G:

Метки отдельных уровней обработки программ изменены или дополнены.

1) Программирование этих команд зависит от типа PG и от версии операционной системы.

2. Программа пользователя

2.1. Язык программирования STEP 5

Язык программирования STEP 5 предназначен для программирования контроллеров семейства SIMATIC S5, с помощью которых можно решать практически любые задачи автоматизации. В STEP 5 Вы найдете как простые двоичные функции, так и сложные цифровые функции и арифметические операции.

Команды языка программирования STEP 5 делятся на

Основные операции:

- используются во всех блоках
- виды представления: контактный план (KOP), функциональный план (FUP)? Список инструкций (AWL)

дополнительные операции:

- Могут быть использованы только в функциональных блоках
- Вид представления: только список инструкций (AWL)

Системные операции:

- принадлежат к дополнительным операциям
- используются только в функциональных блоках
- вид представления: только список инструкций (AWL).
- Могут применяться только пользователем с хорошими знаниями в области системного программирования.

2.1.1. Виды представления KOR, FUP и AWL

При программировании на языке STEP 5 можно выбрать любой из трех видов представления: контактный план (KOR), функциональный план (FUP) или список инструкций (AWL), таким образом, метод программирования в различных ситуациях может быть различен.

Создаваемый при этом программатором (PG) машинный язык идентичен для всех трех видов представления.

При соблюдении определенных правил, ваша программа с помощью PG может быть переведена из одного вида в другой.

Если вы работаете с функциональным или контактным планом, то ваша программа будет представлена в графическом виде, а если вы работаете со списком инструкций, то ваша программа будет представлена в виде набора отдельных команд языка STEP 5.

Релейно-контактная схема	Список команд	Функциональная схема
Программируется с помощью графических символов, как схема прохождения тока. Соответствует DIN 19239 (проект)	Программируется с помощью мнемонических сокращений функциональных операций Соответствует DIN 19239 (проект)	Программируется помощью графических символов. Соответствует IEC 117-15 DIN 40700 DIN 40719 DIN 19239 (проект)



Рис. 2.1: Виды представления языка программирования STEP 5.

GRAPH 5 – это язык программирования с графическим представлением управления процессом. Он является управляющим по отношению к видам представления KOR, FUP, и AWL. Написанная в виде изображений на языке GRAPH 5 программа автоматически преобразуется программатором в программу STEP 5.

2.1.2 Структурированное программирование

В полную программу процесса входят

Операционная система: содержит общую часть всех инструкций и соглашений для реализации внутренних функций (например, сохранение данных при сбросе напряжения питания, управление реакцией пользователя при прерывании и т. д.)

Она расположена на так называемом EPROMe (Erasable Programmable Read Only Memory) и является фиксированной составной частью процессора. Как пользователь, вы не имеете возможности обращаться к операционной системе.

Программа пользователя: содержит набор всех написанных пользователем инструкций и соглашений для обработки сигналов, с помощью которых производится управление установкой (процессом). Программа пользователя распределяется на блоки.

Всю программу пользователя можно разделить на отдельные завершенные участки программы (блоки). Деление вашей программы на блоки значительно проясняет структуру программы и подчеркивает программно-технические связи отдельных частей установки.

Этот метод «структурного программирования» имеет следующие преимущества:

- облегчает программирование больших программ
- дает возможность стандартизации отдельных частей программы
- упрощает организацию программы
- делает простым внесение изменений
- предоставляет возможность раздельного (по блоком) тестирования программы
- облегчает эксплуатацию

Что такое блок?

Блоком называется часть программы пользователя, ограниченная функционально и структурно или по целям использования. Различают блоки, которые содержат инструкции для обработки сигналов (организационные блоки, программные блоки, функциональные блоки, шаговые блоки), и блоки, содержащие данные (блоки данных),

2.1.3. Инструкции языка STEP 5.

Инструкция языка STEP 5 является наименьшей самостоятельной единицей программы пользователя. Она является предписанием для работы процессора. Инструкция языка STEP 5 состоит из операции и операнда.

Пример	:0	M 54.1
	Операция	Операнд
	(Что делать?)	(Объект воздействия)

Операнд может быть представлен абсолютно или символически (через список соответствия).

Пример абсолютного представления: :U E 1.4

Пример символического представления: :U –Motor 1

Более подробно абсолютное и символическое представление можно посмотреть в руководстве пользователя «устройство программирования 685», каталоговый номер C79000-B8500-C373-02.

Объем операций языка STEP 5 позволяет вам:

- логически связывать между собой двоичные значения,
- загружать, хранить и передавать значения,
- сравнивать значения между собой и выполнять над ними арифметические операции,
- преобразовывать вид представления чисел,
- структурировать программу пользователя,
- влиять на обработку программы и т.д.

Большинство операций языка STEP 5 используют в качестве источников и приемников операндов и в качестве приемников для результатов операций два регистра: аккумулятор 1 (АКК1) и аккумулятор 2 (АКК2). Эти аккумуляторы имеют длину 32 бита (1 двойное слово).

Весь набор операций языка STEP 5 описан полностью в главе 6. Вы найдете там примеры программирования отдельных команд языка STEP 5.

Приложение C содержит дополнительно обзор всех имеющихся операций STEP 5 и их допустимые параметры в форме списка.

2.1.4. Представление чисел

Для того, чтобы процессор мог логически связывать между собой числа, изменять или сравнивать их, необходимо загружать их в двоично-кодированном представлении в аккумулятор.

В зависимости от выполняемых операций в STEP 5 допустимы следующие представления чисел:

Двоичные числа: a) 16-битовые числа с фиксированной запятой

b) 32-битовые числа с фиксированной запятой

c) числа с плавающей запятой

Десятичные числа: d) числа в виде кода BCD

При вводе и выводе значений чисел вы устанавливаете на программаторе формат данных (например, KF для формата с фиксированной запятой), в котором вы будете вводить или хотите вывести числовые значения. При этом PG берет на себя преобразование внутреннего представления чисел в требуемый вид.

В случае 16-битовых чисел с фиксированной запятой и чисел с плавающей запятой вы можете выполнять все арифметические операции: сравнение, сложение, вычитание, умножение и деление.

Числа в виде кодов BCD используются только при вводе и выводе: с их помощью нельзя непосредственно производить арифметические операции.

Над 32-разрядными числами с фиксированной точкой можно производить операции сравнения. Кроме того, при преобразовании чисел из кодов BCD в числа с плавающей точкой в качестве промежуточной ступени можно использовать 32-разрядное представление с фиксированной точкой. С помощью новых команд +D и –D Вы можете производить над ними операции сложения и вычитания.

Язык STEP 5 содержит операции преобразования, с помощью которых Вы можете непосредственно произвести преобразования чисел из одного вида представления в другой.

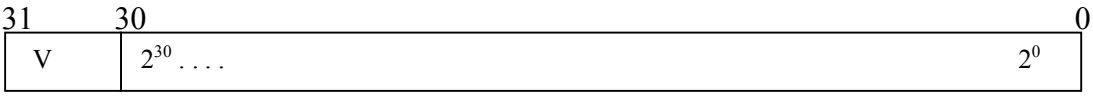
16-битовые и 32-битовые числа с фиксированной запятой

Числа с фиксированной запятой

Числами с фиксированной запятой называются все двоичные числа со знаком, Они могут быть длиной в одно слово (16 бит) или два слова (32 бита), причем биты 15 или 31 содержат информацию о знаке: «0» = положительное число, «1» = отрицательное число.

Отрицательные числа представляются в виде двоичного дополнения.

32-битовое число с фиксированной запятой:



ввод формата данных для 16-битового числа с фиксированной запятой на PG: KF

ввод формата данных для 32-битового числа с фиксированной запятой на PG: Только KH

допустимые области значений -32768 . . . .+32767 (16 бит)

-2147483648 . . . .+2147483647 (32 бита)

(для преобразования 16-битового числа с фиксированной запятой в 32-битовое число с фиксированной запятой смотри Раздел 6)

Числа с фиксированной запятой используются для простых задач вычисления и для сравнения числовых значений. Так как числа с фиксированной запятой всегда целые, то необходимо помнить, что при выполнении операций деления невозможно получение остатка!

Числа с плавающей запятой.

Числа с плавающей запятой – положительное или отрицательное число, разбитое на части, оно всегда занимает двойное слово (32-бита). Число с плавающей запятой представляется в виде экспоненциального числа. Мантисса занимает 24 бита, экспонента - 8 бит.

Экспонента задает порядок числа с плавающей запятой. Знак экспоненты показывает, является ли число с плавающей запятой больше 1.

Мантисса указывает точность числа с плавающей запятой:

—точность при 24-битовой мантиссе: $2^{-24} = 0{,}000000059604$

(соответствует 7 знакам после точки)

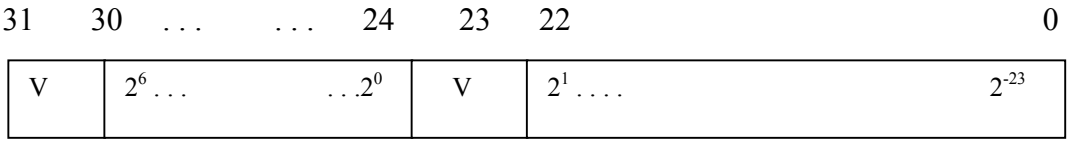
—точность при 16-битовой мантиссе: $2^{-16} = 0{,}000015258$

(соответствует 4 знакам после точки)

если знак мантиссы «0» – то число положительное:

если знак мантиссы «1» – то число отрицательное в виде второго дополнения.

Число с плавающей запятой:



экспонента мантисса

По предустановке при сложении, вычитании, умножении и делении CPU 928 вычисляет только 16-битовую мантиссу (биты с 8 по 23). Младшие значения (стоящие справа) битов с 0 по 7 имеют значения «0».

Если Вы хотите при вычислениях с плавающей запятой иметь большую точность (и можете для этого пожертвовать небольшой потерей во времени выполнения), то Вы можете установить в DX0 «арифметику с плавающей запятой с 24-битовой мантиссой» (см. Раздел7)

Ввод формата данных для чисел с плавающей запятой на программаторе PG : KG.

Допустимая область чисел:

+/- 0,1469368 x 10^{-38} . . . +/- 0,1701412 x 10^{39}

Ввод чисел с плавающей запятой Z с помощью программатора

Z = 12.34567

L KG + 1234567 + 02

Мантисса экспонента (по основанию 10) со знаком

Z= + 0,1234567x10^{+2} = 12,34567

Z= -0,005

L KG – 50000000 - 02

Мантисса экспонента (по основанию 10) со знаком

Z= -0,5 x 10^{-2} = 0,005

Если вы используете числа с плавающей запятой для решения задач сложных вычислений, особенно при перемножении и делении, то можете работать или с очень большими, или с очень маленькими числами.

Числа в коде BCD

Десятичные числа представляются в виде чисел в коде BCD. Трехзначное число со знаком занимает в аккумуляторе 16 бит (1 слово):

Бит 15…12 11…8 7…4 3…0

Знак сотни десятки единицы

15 0

знак	102	101	100

Отдельные цифры представляются в виде положительных четырехбитовых двоичных чисел от 0000 до 1001 (от 0 до 9).

Допустимая область чисел: -999…+999

Левые биты резервируются для знака.

Знак положительного числа : «0000»

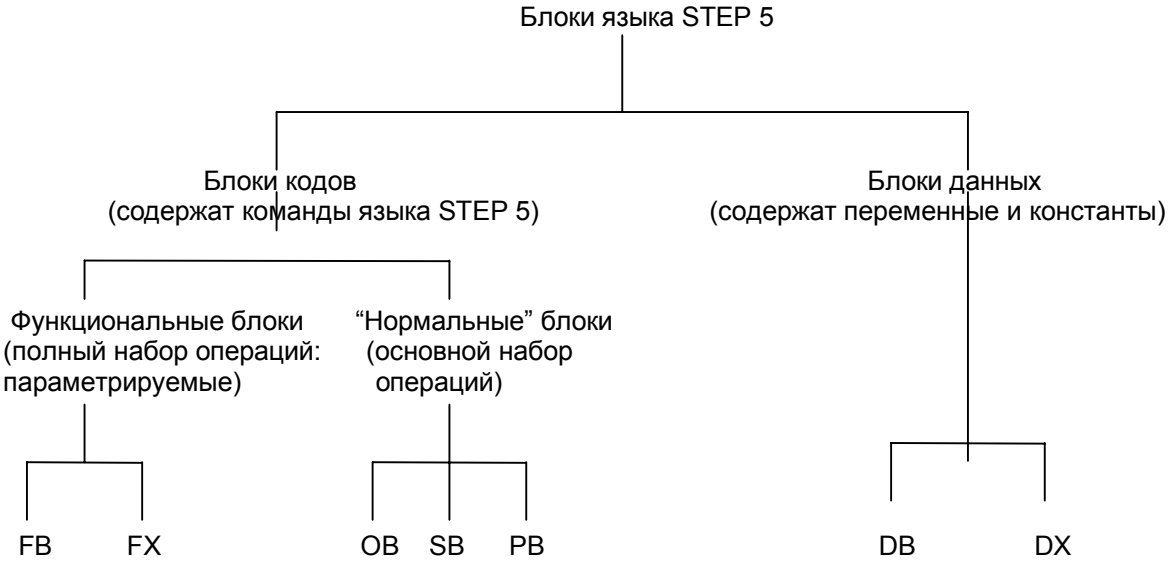
Знак отрицательного числа : «1111»

2.1.5. Блоки языка STEP-5

Блоком называется часть программы пользователя, ограниченная по функциональным, структурным целям или целям применения.

Блоки идентифицируются:

- типом блока (OB, PB, SB, FB, FX, DB, DX)
- номером блока (число от 0 до 255)



Язык программирования STEP-5 различает следующие типы блоков:

- Организационные блоки (OB)

Организационные блоки являются интерфейсом между операционной системой и программой пользователя. Они могут быть разделены на две группы:

OB1…OB39 вызываются операционной системой и управляют обработкой программы, действиями процессора при запуске и в случае ошибки. Эти OB программируются пользователем.

OB40…OB255 содержат специальные функции операционной системы. Они не могут быть вызваны пользователем.

- Программные блоки (PB)

Они используются для структурирования программы пользователя и содержат части программы, разделены по технологическим или функциональным признакам.

Обычно программные блоки содержат большую часть программы пользователя.

- Шаговые блоки (SB)

Это специальные программные блоки, которые применяются для шаговой обработки цепочек управления.

- Функциональные блоки (FB/FX)

Они служат для программирования часто повторяющихся или очень сложных функций (например, цифровые функции, управление процессом, регулирование, функции сообщений). Функциональный блок может быть многократно вызван из любого другого блока и при каждом вызове ему можно задать новые операнды (параметрирование).

- Блоки данных (DB/DX)

В блоках данных хранятся (жестко фиксированные или изменяемые) данные, с которыми работает программа пользователя. Эти блоки не содержат инструкций языка STEP 5 и отличаются по своим функциям от остальных блоков.

Блок строится следующим образом:

Все блоков состоят из - заголовка блока и
- тела блока

Заголовок блока всегда имеет длину 5 слов данных. Программатор автоматически заносит туда
- метку начала блока
- тип блока (OB, FB….)
- номер блока
- метку программатора
- библиотечный номер
- длину блока (включая заголовок блока).

Заголовок блока в программе пользователя:

Начальная	Метка
Тип блока	Номер блока
Метка PG	Библиотечный
Табличный номер	
Длина блока с заголовком (слов)	

15 0

Точные метки типов блоков и номера блоков можно найти в разделе 8.2.1

В теле блока – в зависимости от типа блока – содержатся:
- команды языка STEP 5 (OB, PB, SB, FB, FX)
- изменяемые и неизменяемые данные (DB, DX)
- список формальных операндов (FB, FX)

Для блоков DB, DX, FB и FX программатор дополнительно создает предзаголовок блока (DV, DXV, FV и FXV соответственно). Этот предзаголовок блока содержит информацию о формате данных (для DB и DX) или о метках переходов (для FB и FX), которые обрабатываются только программатором, поэтому предзаголовки в память контроллера не передаются. Как пользователь вы не можете иметь непосредственного влияния на содержимое предзаголовков блоков.

Блок языка STEP 5 может занимать максимально 4096 слов в памяти программ процессора. При вводе или передачи блоков с помощью программатора обращайте внимание и проверяйте объем памяти используемого программатора!

Отдельные типы блоков, которые предлагаются для программирования:

OB 1 …39
FB 0 …255
FX 0 …255
PB 0 …255
SB 0 …255
DB 3 …255
DX 1 …255

Блоки данных DB1, DB2, и DX0 содержат параметры. Они зарезервированы для определенных функций и поэтому не могут быть использованы в общих целях.

Все запрограммированные блоки передаются с помощью в любой последовательности в память программ (смотри рисунок), которая представляется в виде модулей RAM или EPROM, устанавливаемых в процессор. Начальные адреса запоминаемых блоков заносится в блок данных DB0.

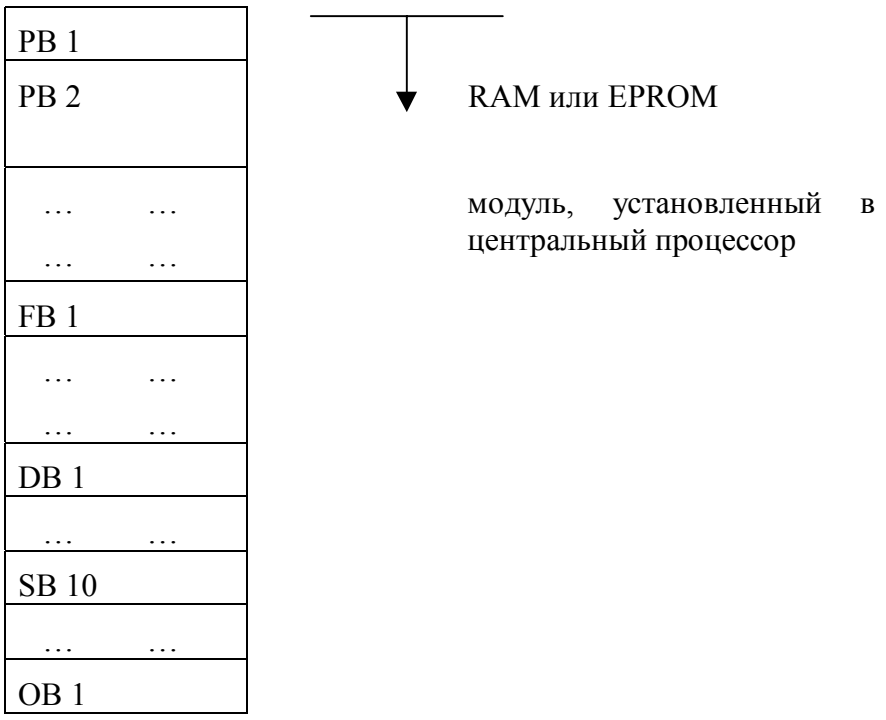


Рис.2-2: Расположение блоков памяти программ

При коррекции блоков старый блок в памяти объявляется недействительным, а новый заносится в память. Таким образом, при стирании блоков блок удаляется, а только объявляется недействительным.

Важно!

Стертые или корректированные блоки продолжают занимать место в памяти!

С помощью функции программатора «сжатие памяти» можно освободить место в памяти для новых блоков: функция удаляет все недействительные блоки в памяти, а действительные компонует (смотри Раздел 11.6).

2.2. Организационные, программные и шаговые блоки

Эти три вида блоков не различаются по способу программирования и вызова.

Все три можно программировать в виде KOP, FUP и AWL по выбору.

2.2.1. Программирование.

При программировании организационных, программных и шаговых блоков следует идти следующим путем:

- Сначала введите тип, затем номер блока, который хотите программировать.
  В Вашем распоряжении имеются следующие номера блоков:
  Программные блоки 0 255
  Шаговые блоки 0 255
  Организационные блоки 1 39
- Введите свою программу пользователя на языке STEP 5.

Важно !

При программировании PB, SB и OB можно использовать только основной набор команд языка STEP 5!

o Завершите ввод программы командой BE (конец блока).

Важно!

Блок языка STEP 5 всегда должен содержать законченную программу. Логические цепочки должны быть завершены внутри блока.

В блоке можно записать до 4000 инструкций (в зависимости от программатора).

Заголовок блока автоматически создается программатором и занимает 5 слов в памяти программ.

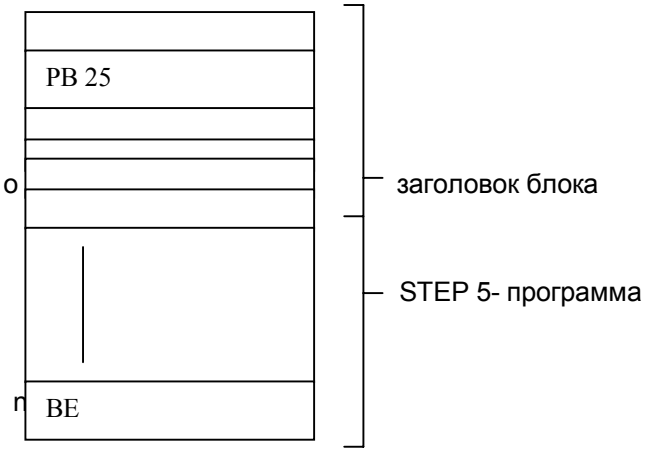


Рис. 2-3: структура организационных, программных и шаговых блоков.

2.2.2 Вызов

Блоки должны быть разрешены для обработки. Это достигается с помощью вызовов блоков (рисунок).

Вызов блока может быть запрограммирован внутри другого организационного, программного или шагового блока. Вызов блока соответствует переходу в программу. Каждый переход является причиной смены блока. Переходы могут быть как условные, так и безусловные.

- Безусловный вызов блока: SPA xx

  Соответствующий блок обрабатывается <u>независимо</u> от предыдущего результата логической операции (VKE).

  VKE представляет собой состояние сигнала в процессоре, который используется для дальнейшей двоичной обработки сигналов. VKE может связываться с состоянием сигнала операнда или операция может выполняться в зависимости от состояния предыдущего результата VKE: Так называемые «безусловные» операции выполняются всегда , так называемые «условные» операции выполняются <u>только</u> в случае, если VKE=1.

  Команда SPA принадлежит к «безусловным» операциям. На нее не влияет состояние VKE. VKE используется для перехода в новый блок, там он может быть обработан, однако не может иметь дальнейших логических связей.

- Условный вызов блока: SPB xx

  Инструкция перехода SPB принадлежит к «условным» операциям, то есть переход к соответствующему блоку происходит только в том случае ,если VKE = 1. При VKE = 0 перехода к блоку не происходит. При этом VKE устанавливается в «1»!

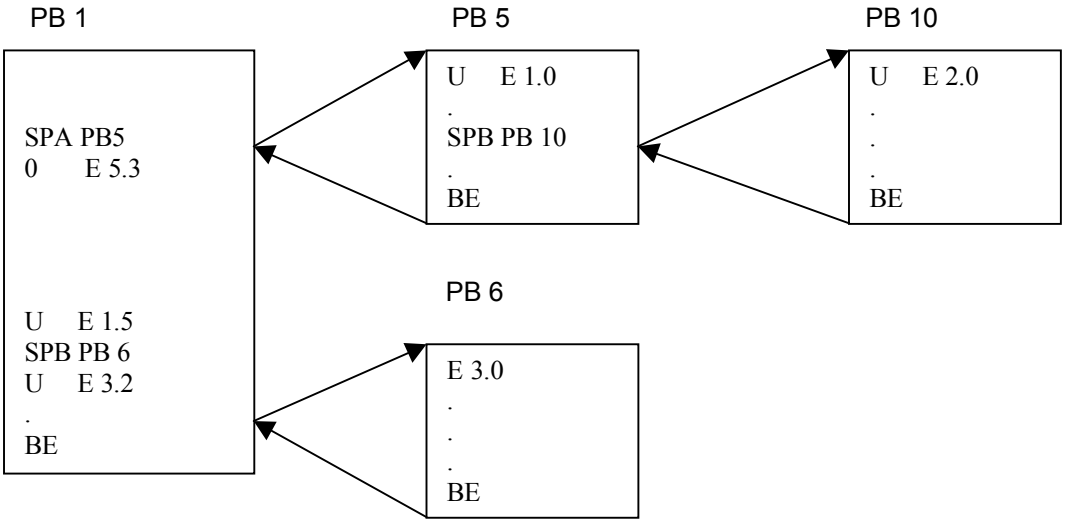


Рис. 2.4. <u>Вызов блока который разрешает разрешает обработку</u>.

После инструкции BE происходит возврат в блок, из которого вызван данный блок. Обработка программы продолжается с первой STEP 5 инструкции, следующей за инструкцией вызова.

Инструкция конца блока BE выполняется независимо от результата логической операции. После BE результат логической операции (VKE) не может иметь дальнейших логических связей. Непосредственно перед обработкой команды BE имеющийся результат VKE/ результат вычислений передается в вызывающий блок и может быть там обработан. При возврате из блока содержимое аккумуляторов АКК1, АКК2, АКК3, АКК4, битов индикации ANZ0 и ANZ1 и VKE не меняется.

2.2.3.Специальные организационные блоки

Интерфейсом между операционной системой и программой пользователя являются организационные блоки. Организационные блоки с OB1 по OB39 являются частями программы пользователя, которые можно запрограммировать так же, как программные, функциональные или шаговые блоки. Программируя данные OB можно определить действия процессора при запуске, во время обработки программы и в случае ошибки. Организационные блоки становятся действительными, как только они загружаются в память контроллера. Это можно выполнять и в режиме работы. Важно то, что данные OB вызываются операционной системой как реакция на определенные условия.

Организационный блок	Функция и критерий вызова
OB 1	Организация вызова циклической обработки программы после окончательного запуска
OB 2	Организация вызова программы, управляемой по прерыванию от сигнала шины S 5 (прерывание от процесса)
OB 10 – 18 OB 10 OB 11 OB 12 OB 13 OB 14 OB 15 OB 16 OB 17 OB 18	Организация обработки программы, управляемой по времени (прерывание по времени) Время обработки 10 мс Время обработки 20 мс Время обработки 50 мс Время обработки 100 мс Время обработки 200 мс Время обработки 500 мс Время обработки 1 мс Время обработки 2 мс Время обработки 5 мс
OB 20 - OB 22 OB 20 OB 21 OB 22	Организация выполнения запуска Вызов по требованию «нового пуска» Вызов по требованию «ручной перезапуск» Вызов при восстановлении напряжения питании («автоматический перезапуск»)
OB 19, 23 – 24 OB 19 OB 23 OB 24 OB 25 OB 26 OB 27	Реакция на следующие аппаратные или программные ошибки: [1]) Ошибка выполнения: Вызов незагруженного блока. Задержка квитирования в программе пользователя (при непосредственном обращении к модулям периферии или к другим адресам шины S 5) Задержка квитирования при обновлении области отображения процесса и при передаче маркеров межпроцессорной связи Ошибка адресации Превышение времени цикла Ошибка кодов команд: Ошибка замещения

[1]) Если в случае ошибки OB не запрограммирован, процессор переходит в состояние останова. Исключение: При отсутствии OB 23 и OB 24 (задержка квитирования) нет никакой реакции!

Организационный блок	Функция и критерий вызова
OB 28	Останов от функции PG /переключателя STOP/ шины S5 [1])
OB 29	Ошибка кодов команд: Недопустимый код команды
OB 30	Ошибка кодов команд: Недопустимый параметр
OB 31	Специальные ошибки выполнения
OB 32	Ошибка выполнения: Ошибка передачи в блок данных
OB 33	Ошибка «будильника» (требование обработки прерывания по времени, в то время как предыдущее прерывание по времени еще обрабатывается)
OB 34	OB 34 Ошибка при обработке функции регулятора

1) OB 28 вызывается перед переходом в состояние STOP. Контроллер переходит в состояние «СТОП» в любом случае – запрограммирован блок OB 28 или нет.

После того, как операционная система вызовет соответствующий организационный блок, обрабатывается программа пользователя, содержащаяся в нем. Как правило, после выполнения организационного блока ошибки происходит возврат в место прерывания программы (исключение: OB 28). О реакции при отсутствии OB ошибки см. Раздел 5.4).

Для тестирования эти организационные блоки можно вызывать из программы пользователя (SPA/SPB OB xxx). Однако невозможно, например, с помощью вызова OB 28 добиться перехода в состояние останова и путем вызова OB 22 – обеспечить автоматический перезапуск!

Важно!

Специальные организационные блоки программируются пользователем и вызываются операционной системой автоматически!

2.2.4. Организационные блоки спецфункций

Приведенные ниже организационные блоки содержат специальные функции операционной системы. Они не могут быть запрограммированы пользователем (это касается всех организационных блоков с номерами от 40 до 255!), однако могут быть вызваны. Эти блоки не содержат программ на языке STEP 5. OB спецфункций могут быть вызваны из всех кодовых блоков.

Обзор 2 – 5: Организационные блоки спецфункций в CPU 928.

OB110	Доступ к байту индикации
OB 111	Стирание содержимого АКК 1, АКК 2, АКК 3 и АКК 4
OB 112	Пролистывание аккумуляторов вперед
OB 113	Пролистывание аккумуляторов назад
OB 120	Вкл -/выключение «блокировки всех прерываний»
OB 121	Вкл -/выключение «блокировки отдельных прерываний по времени»
OB 122	Вкл -/выключение «задержки всех прерываний»
OB 160 – OB 163	Счетчики цикла
OB 170	Чтение стека блоков (BSTACK)
OB 180	Переменное обращение к блоку данных
OB 181	Тестирование блоков данных
OB 190, 192	Передача маркеров в блок данных
OB 191, 193	Передача блоков данных в область маркеров
OB 200, 202-205	Межпроцессорная коммуникация
OB 216-218	Обращение к буферам связи
OB 220	Преобразование содержимого АКК 1 из 16-битового значения в 32-битовое число с фиксированной запятой путем расширения знака.
OB 221	Установка и запуск нового времени цикла
OB 222	Перезапуск времени цикла
OB 223	Останов при различных видах запуска в многопроцессорном режиме работы контроллера.
OB 224	Передача блоков маркеров связи в многопроцессорном режиме работы
OB 226	Побайтовое чтение содержимого ячеек памяти операционной системы
OB 227	Считывание контрольной суммы области памяти операционной системы
OB 228	Считывание информации состояния уровня обработки программ
OB 230-237	Функции стандартных функциональных блоков
OB 240	Инициализация регистра сдвига
OB 241	Вызов регистра сдвига
OB 242	Стирание регистра сдвига
OB 250	Инициализация PID-регулятора
OB 251	Обработка PID-регулятора
OB 254-255	Передача блока данных в DB-RAM

Более детально описание этих функций можно найти в Разделе 6.

2.3. Функциональные блоки

Функциональные блоки (FB/FZ) являются частью программы пользователя, так же как, например, программные блоки. Функциональные блоки FX имеют одинаковую структуру с блоками FB, и программируются точно так же. С помощью функциональных блоков можно реализовать часто повторяющиеся или очень сложные функции.

Функциональные блоки по сравнению с программными, организационными и шаговыми блоками имеют четыре основных отличия:

- Функциональный блок можно параментировать, т. е. Формальные операнды функционального блока при разных вызовах можно заменять на различные фактические операнды. При этом для всех случаев применения многократно используется один и тот же функциональный блок.

- Функциональный блок может быть запрограммирован с помощью полного набора команд языка STEP 5. К нему кроме основного набора операций, которым может быть использован во всех видах блоков, относятся также дополнительные операции и системные операции.

  Важно!

  Дополнительные и системные операции могут быть использованы исключительно в функциональных блоках!

- Функциональный блок можно программировать и документировать только в виде списка команд (AWL).

  Однако вызвать функциональный блок можно в представлениях КОР и FUP в виде графического изображения «черного ящика»

- Функциональному блоку может быть присвоено имя длиной до 3 символов.

  Каждый функциональный блок представляет собой комплексную завершенную функцию внутри программы пользователя. Функциональные блоки могут быть:

- стандартными программными продуктами фирмы SIEMENS (стандартные функциональные блоки на мини дискетах); программа пользователя с данными функциональными блоками может быстро и надежно выполнять функции управления, регулировки, протоколирования.

- Запрограммированы пользователем.

2.3.1. Структура функциональных блоков

Заголовок блока (5 слов) в функциональном блоке построен также как заголовки других блоков STEP 5.

Тело функционального блока по своей структуре отличается от других типов блоков. Оно содержит действительную программу функционального блока. Выполняемые функции представляются в виде списка команд на языке программирования STEP 5. Между заголовком и, собственно, программой пользователя языка STEP 5 резервируется место для указания его имени и списка формальных операндов. Так как этот список не содержит команд для процессора, то он обходится с помощью команды безусловного перехода (SPA), которая автоматически вставляется программатором. Эта команда перехода при вызове блока на программатор не индицируется!

Операнды могут быть введены как в абсолютном виде (например, M 2.5), так и в символьном (например, - MOTOR 1). Соответствие символьных операндов должно быть указано в списке назначений.

При вызове функционального блока выполняется только программа тела блока.

Так выглядит функциональный блок в памяти программируемого контроллера:

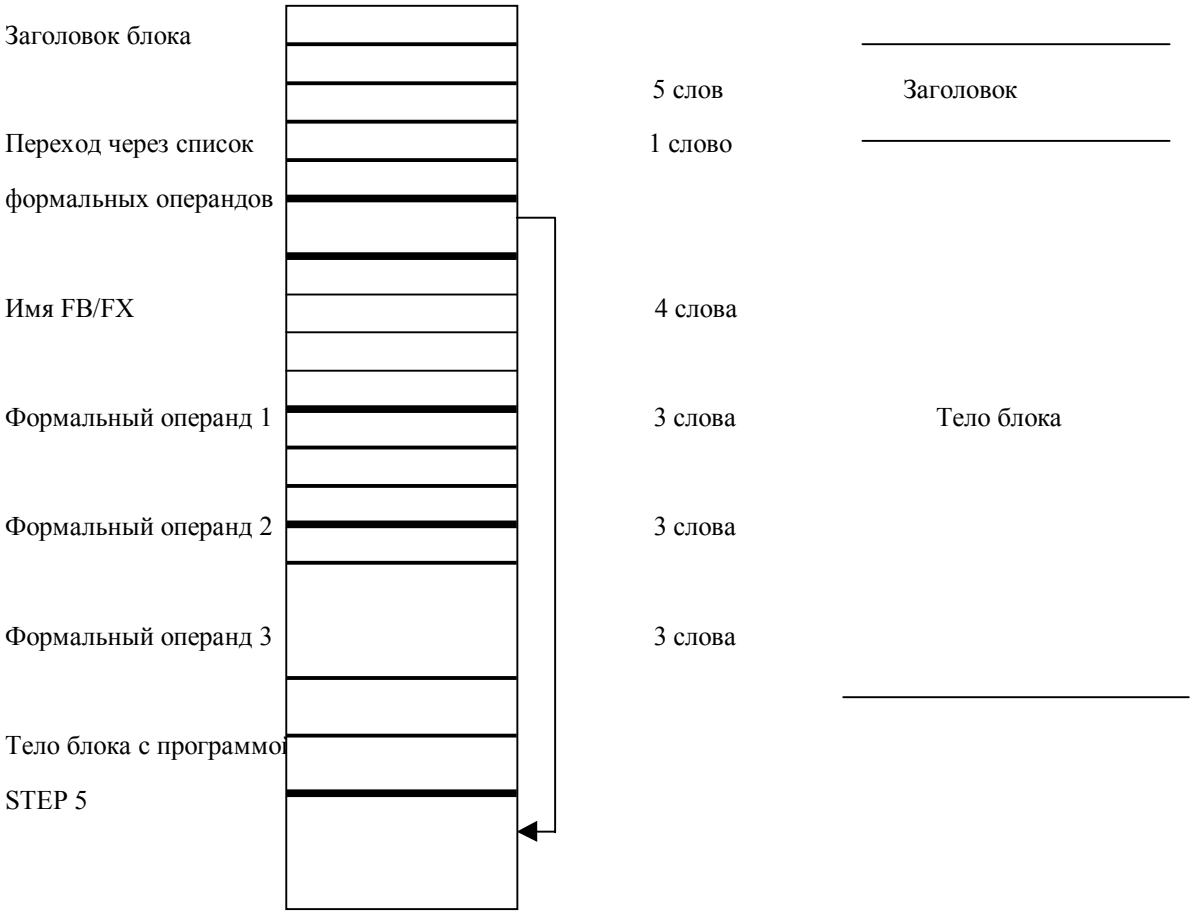


Рис. 2.6: Структура функционального блока

В памяти хранятся все данные, которые необходимы программатору, чтобы иметь возможность представить функциональный блок при вызове графически и для того, чтобы проверить операнды при параметрировании функционального блока. Неверный блок отвергается программатором.

Важно!

При работе с функциональным блоком следует различать:

а) программирование FB/FX и

б) вызов и параметрирование FB/FX

При программировании устанавливается функция блока. Указанные операнды являются формальными операндами, которые выполняют функцию резервирования места.

При вызове блока из предыдущего блока (OB, PB, SB, FB, FX) формальные операнды заменяются на фактические, функциональный блок параметрируется.

Далее можно найти более подробное описание.

2.3.2. Программирование функциональных блоков

При вводе функционального блока с помощью программатора следует:

o Ввести номер вводимого функционального блока

Важно!

Функциональные блоки пользователя должны иметь номера от FB255 со стандартными функциональными блоками, имеющими номера от FB1 по FB199.

o Можно ввести в качестве библиотечного номера число от 0 до 99999, функциональный блок индентифицируется этим библиотечным номером независимо от номера FB и имени.

Вы должны задать определенный библиотечный номер только один раз, чтобы можно было идентифицировать различные функциональные блоки.

o Введите имя функционального блока. Оно может быть длиной до 8 символов.

o Введите формальные операнды, которые будут использованы в блоке (максимально 40 формальных операндов)

Для каждого формального операнда надо ввести:

1. Имя параметра
2. Вид параметра
3. Тип параметра

Имя может состоять максимально из 4 символов.

В качестве вида параметра блоков можно выбрать:

E- входной параметр

A- выходной параметр

D- данные

B- команда

T- таймер

Z- счетчик

E, D, B, T или Z являются параметрами, которые при графическом изображении функционального блока выводятся слева. Параметры, отмеченные как A выводятся справа от графического изображения функционального блока.

Для параметров блока E, A и D дополнительно следует указать тип параметра.

BI/BY/W/D	для параметров E, A
KM/KN/KY/KC/KF/KT/KZ/KG	для параметров D

Тип параметра указывает, является параметр E или A битовым, байтовым, словным или двойным словным и какой данных (например, двоичный или шестнадцатеричный) для параметра D.

Вид параметра	Тип параметра	Допустимые фактические операнды
E, A	BI– для операндов с битовым адресом	E n.m входы A n.m выходы M n.m маркеры
	BY– для операндов с байтовым адресом	EB n входные байты AB n выходные байты MB n байты маркеров DL n старший брат слова данных
	W– для операндов со словным адресом	DR n младший байт слова данных PB n периферийные байты EW n входные слова AW n выходные слова MW n слова маркеров DW n слова данных PW n периферийные слова
	D– для операндов с двухсловным адресом	ED n двойное слово входов AD n двойное слово выходов MD n двойное слово маркеров DD n двойное слово данных
D	KM– для двоичного числа (16 разрядов) KY– для двух байтовых чисел от 0 до 255 KH– для 16-ричных чисел (макс. 4 позиции) KC– для символов (макс. 2 буквенно-цифровых знака) KT– для времени (BCD-код) в диапазоне 1…999.3	константы
D	KZ –для счетчиков (BCD) диапазон: 0…999 KF –для чисел с фиксированной запятой диапазоне от –32768 до +32768 KG –для чисел с плавающей запятой	
B	Указание типа не требуется	DB N – блоки данных, выполняется команда ADB FB N – функц. Блоки (допустимы только без параметров) вызываются безусловно SPA PB N – прогр. блоки вызываются безусловно SPA SB N – шагов. блоки вызываются безусловно SPA

T	Указание типа не требуется	T 0 … 127 Таймер [1])
Z	Указание типа не требуется	Z 0 … 127 Таймер [1])

[1]) Значение параметра таймера или счетчика должно быть указано в качестве данных или должно быть запрограммировано в качестве константы в функциональном блоке.

o Затем введите в программу на языке STEP 5 в виде списка команд.

При этом формальные операнды помечаются заранее заданными символами (например, U=X1). Можно несколько раз в различных местах программы функционального блока опрашивать их.

Важно!

Если Вы в списке операндов меняете число или последовательность формальных операндов, то необходимо соответственно переправить команды замещения в программе на языке STEP 5 при вызове функционального блока и список параметров блока в вызываемом блоке.

Важно!

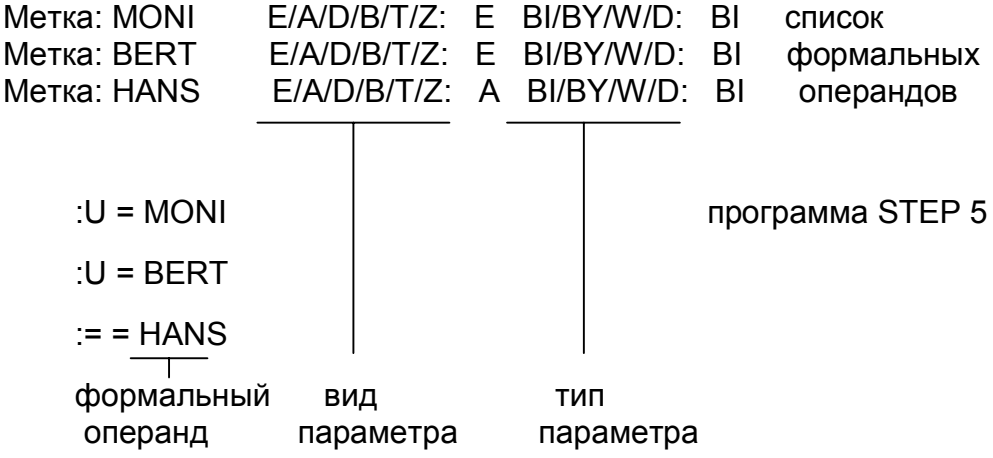
Программирование и изменение функционального блока производите в основном на дискете или «винчестере» и после этого передавайте в программируемый контроллер!

o Завершите ввод программы командой BE (конец блока).

Пример 2 - 7: Программирование функционального блока

FB202

Имя: BEISPIEL



2.3.3. Вызов и параметрирование функциональных блоков

Каждый функциональный блок можно вызвать в любом месте программы на языке STEP 5 с любой частотой. В то время как, программа на языке STEP 5 может быть вся написана в виде списка команд. Вызов функционального блока можно представить в графическом виде (FUP или KOP).

При вызове и параметрировании следует предусмотреть:

o Введите в блок, из которого осуществляется вызов, команду вызова функционального блока.

Вызов функционального блока может быть запрограммирован внутри организационного, программного или шагового блока или внутри другого функционального блока.

Вызов может быть условный или безусловный:

- Безусловный вызов (SPA, FBN для функционального блока или BA FXN для дополнительного функционального блока):

  Соответствующий функциональный блок обрабатывается независимо от предыдущего результата выполнения логической операции (VKE).

- Условный вызов (SPB FBN для функциональных блоков или BAB FXN для дополнительных функциональных блоков):

  Соответствующий функциональный блок обрабатывается только в том случае, если предыдущий результат логической операции VKE = 1. При VKE = 0 инструкция перехода не выполняется, однако, при этом VKE устанавливается в «1».

После условного или безусловного вызова нельзя производить дальнейшие логические связи с результатом логической операции. Однако, при переходе в соответствующий функциональный блок, VKE переносится туда и может там быть обработан.

После того как была введена команда вызова (например, SPA FB200), автоматически на экране программатора высвечивается имя и список формальных операндов соответствующего функционального блока:

o Теперь каждому формальному операнду должны быть присвоены фактические операнды для данного вызова, т.е. функциональный блок должен быть запараметрирован.

Эти фактические операнды при различных вызовах могут быть различны. При первом вызове FB200, например, входы и выходы, а при втором вызове – маркеры.

Соответствующему списку формальных операндов может при каждом вызове функционального блока соответствовать максимально до 40 фактических операндов.

Важно!

Прежде, чем вы вызовете и отпараметрируете функциональный блок, следует запрограммировать данный функциональный блок и переписать его на дискету или непосредственно в память программ пользователя в контроллер!

После перехода в функциональный блок и обработки программы функционального блока, вместо соответствующих формальных операндов используются фактические операнды, указанные при вызове блока.

На основании этих свойств можно использовать функциональный блок в программе пользователя, задевая лишь параметры.

Пример: Вызов и параметрирование функционального блока в представлении AWL и KOP/FUP в программном блоке.

- Представление AWL

```
PB 25
      : SPA  FB 201
ИМЯ : E – ANTR
ZU-E :        DW1
RME :         E 3.5
ESB :         M 2.5
UEZ :         T 2
ZEIT:         KT10.1
ZU-A:         DW1
BEA :         A 2.3
LSL :         A 6.0
```

Формальный операнд — Фактический операнд

- Представление KOP/FUP

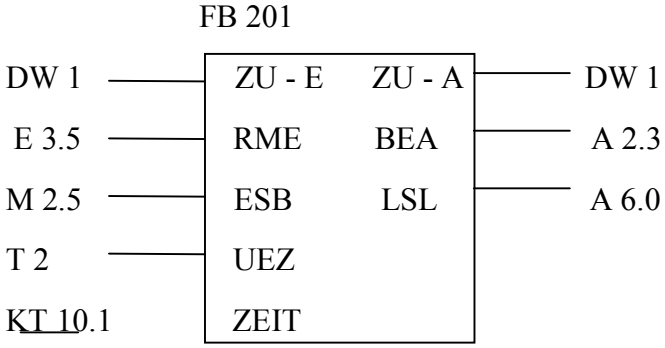


Рис. 2 – 8: Вызов и параметрирование функционального блока.

Следующий (полный) пример должен еще раз пояснить Вам программирование, вызов и параметрирование функционального блока. Вы легко можете его выполнить.

Программирование функционального блока FB 202:

FB 202

ИМЯ : BEISPIEL

МЕТКА : MONI E/A/D/B/T/Z: E BI/BY/W/D: BI список

МЕТКА : BERT E/A/D/B/T/Z: E BI/BY/W/D: BI формальных

МЕТКА : HANS E/A/D/B/T/Z: A BI/BY/W/D: BI операндов

: U = GABI

: U = BERT

: = = HANS

STEP 5-программа

Вид параметра

Тип параметра

формальный операнд

Вызов и параметрирование функционального блока FB 202 в программном блоке PB 25:

—Вид представления AWL

PB25

: SPA FB202

ИМЯ : BEISPIEL

GABI : E13.5

BERT : M17,7

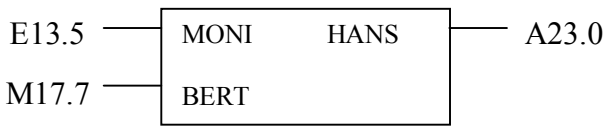
HANS : A23.0

Фактические операнды

Формальные операнды

—Вид представления KOP/FUP

FB202



Выполняемая программа

:U E 13.5

:U M 17.7

:= A 23.0

2.3.4. Специальные функциональные блоки

- Стандартные функциональные блоки.

Кроме функциональных блоков, которые самостоятельно программируются пользователем, имеются стандартные функциональные блоки, которые поставляются в виде готового программного продукта. Они содержат общие часто используемые стандартные функции (например, выдача сообщений, управление выполнением процессов и т.д.).

Стандартные функциональные блоки имеют номера от FB1 до FB199.

Если Вы используете стандартный функциональный блок, обратите особое внимание на специальные указания в соответствующем описании этого блока (используемая область, соглашения и т.д.).

Стандартные функциональные блоки для S5 – 135U, их исполнение, потребность в памяти и используемые ими переменные указаны в каталоге ST 57 «Программное обеспечение для программируемых контроллеров серии U»

Пример стандартного функционального блока

Извлечение корня из числа с плавающей запятой	FB6 для 115U
RAD : GP	FB6 для 135U
	FB19 для 150U

Функциональный блок RAD : GP извлекает квадратный корень из числа с плавающей запятой (8 – битовая экспонента и 24 – битовая мантисса). Результат – также число с плавающей запятой (8 – битовая экспонента и 24 – битовая мантисса), как бы младший бит мантиссы не округлялся.

Если необходимо, функциональный блок для последующих операций устанавливает идентификатор «корень отрицательный»

Числовые значения:

Радикал - 0,1469368 Exp – 38. . . . . +0,1701412 Exp + 39

Корень + 0,3833234 Exp – 19. . . . . + 0,1304384 Exp + 20

Функция: $Y = \sqrt{A}$

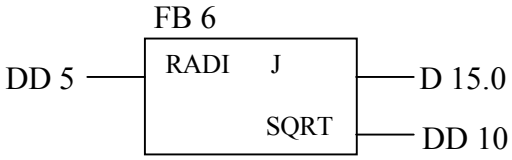
Y = SQRT; A = RADI

Вызов функционального блока FB 6:

В виде AWL:

```
      :  A DB 17
      :
      :
      :  SPA FB 6
ИМЯ   :  RAD : GP
RADI  :  DD 5
J     :  D 15.0
SQRT  :  DD 10
```

В виде KOP/FUP



DD – двойное слово данных

В приведенном выше примере извлекается корень из числа с плавающей запятой, которое записано в двойном слове данных DD 5 в виде 8-битовой экспоненты и 24-битовой мантиссы.

Результат, который тоже является 32-битовым числом с плавающей запятой, записывается в

DD 10. Перед выполнением функции должен быть вызван соответствующий блок данных.

Параметр J (вид параметра M, тип параметра BI) определяет знак результата: J = 1 для отрицательного результата. Используемые слова маркеров: с MW 238 по MW 254.

- Функциональный блок FB0

Если не запрограммирован организационный блок OB1, то операционная система вместо OB 1 циклически обрабатывает программу в FB0.

Важно!

FB0 должен использоваться только для обработки циклической программы! (Он не может содержать параметров).

Поскольку Вы в функциональном блоке можете использовать полный набор команд языка STEP 5, то программирование FB0 вместо OB1 удобно в том случае, если Вы хотите обработать короткую и критичную по времени программу.

Если запрограммированы и FB0 и OB1, то циклически обрабатывается только OB1.

2.4. Блоки данных

В блоках данных (DB/DX) располагаются фиксированные или переменные данные, которые используются программой пользователя. В блоках данных не могут обрабатываться команды языка STEP 5.

Данные в блоках данных могут быть:

- в виде набора битов, например, для отображения состояния установки
- в виде чисел (шестнадцатеричных, двоичных, десятичных) для значений таймеров, результаты вычислений
- алфавитно-цифровые символы, например, для текстовых сообщений

2.4.1 Структура блока данных

Блок данных состоит из следующих частей:
- предзаголовок блока (DV, DXV)
- заголовок блока
- тело блока

Предзаголовок блока создается автоматически. Он содержит информацию о форматах данных, указанных в теле блока. Как пользователь, вы не можете влиять на содержимое предзаголовка блока данных.

Важно!

Если вы передаете блок данных из AG или EPROM на дискету, то соответствующий предзаголовок блока стирается. Поэтому не надо блоки данных с различными форматами данных изменять в AG перед тем, как передавать их на диск, поскольку все слова данных этого DB автоматически имеют формат данных, который вы выбрали в маске предустановки.

Заголовок блока занимает 5 слов памяти и содержит:
- метку блока
- метку программатора
- номер блока
- библиотечный номер блока
- длину блока (включая длину заголовка блока).

Тело блока содержит в порядке возрастания, начиная со слова DW0 слова данных, с которыми работает программа пользователя. Каждое слово данных занимает в памяти 1 слово (16 бит).

Блок данных может занимать в памяти процессора до 2000 слов. Обратите внимание на объем памяти программатора при вводе и передаче блоков данных с помощью программатора!

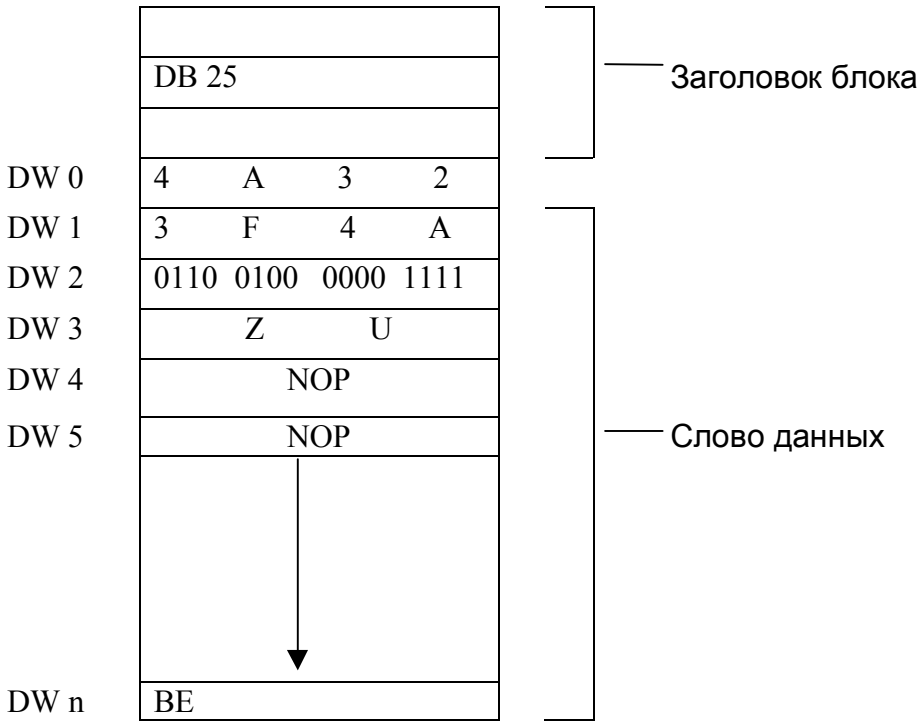


Рис. 2.9. Структура блока

2.4.2. Программирование блока данных

Блоки данных создаются таким образом:

- Вводится номер блока данных от 3 до 255 (при блоках данных DB) или от 1 до 255 (при блоках данных DX).

  Важно!
  Блоки данных DB 0, DB 1, DB 2 и DX 0 зарезервированы для определенных функций и поэтому не могут свободно использоваться (см. Раздел 2.4.4)!

- Вводятся отдельные слова данных в желаемые форматах данных.

Допустимые форматы данных	Примеры
KM = битовое представление	00100110 00111111
KH = шестнадцатеричное число	263F
KY = байт	38, 63
KF = число с фиксированной запятой	+9791
KG = число с плавающей запятой	+13566123+12
KC = символы	? ! ABCD ! ” # - + .. %
KT = значение таймеров	055.2
KZ = значение счетчиков	234
ZL = соответствие для списка соответствия (не для версии программного обеспечения PG S5DOS)	MOTOR1 = A 12.5

ВАЖНО!
Ввод слов данных не завершается инструкцией конца блока BE!

2.4.3. Открытие блока данных

Блок данных (DB/DX) может быть открыт только безусловно. Это возможно внутри организационного, программного, шагового или функционального блока. Какой-либо блок данных может быть многократно открыт в программе.

Блок данных открывается следующим образом:

o блок данных DB по инструкции	A DB ..
o блок данных DX по инструкции	AX DX ..

Доступ к хранимым данным в вызванном блоке данных производится при обработке программы по командам загрузки и передачи:

С помощью команд загрузки можно содержимое адресованного слова данных передать в АКК1, где его обрабатывает процессор.

Команды загрузки:	L DW..	(слово)
	L DR..	(правый байт)
	L DL ..	(левый байт)
	LDD ..	(двойное слово)

С помощью команд передачи данные передаются из АКК1 в адресованное слово данных.

Команды передачи:	T DW ..
	T DR ..
	T DL ..
	T DD..

При загрузке содержимое слова данных не изменяется.

При передаче старое слово данных не изменяется.

Важно!

- **Перед обращением к слову данных необходимо открыть в программе пользователя желаемый блок данных, т.к. только в этом случае процессор найдет правильное слово данных! Адресуемое слово данных должно содержаться в открытом блоке, в противном случае операционная система опознает по команде T DX ошибку передачи или загрузит по команде L DX неверное значение.**
- **С помощью команд загрузки и передачи вы можете работать только до номера слова данных 255!**

Пример: Передача слов данных
Необходимо слово данных 1 блока данных DB 10 передать в слово данных DW 1 блока данных DW 20 (см. рисунок).
Для этого необходимо ввести следующие инструкции:

A	DB10	(вызов DB 10)
L	DW1	(DW1 передается в аккумулятор)
A	DB20	(вызов DB 20)
T	DW1	(DW1 передается из аккумулятора в DW1)

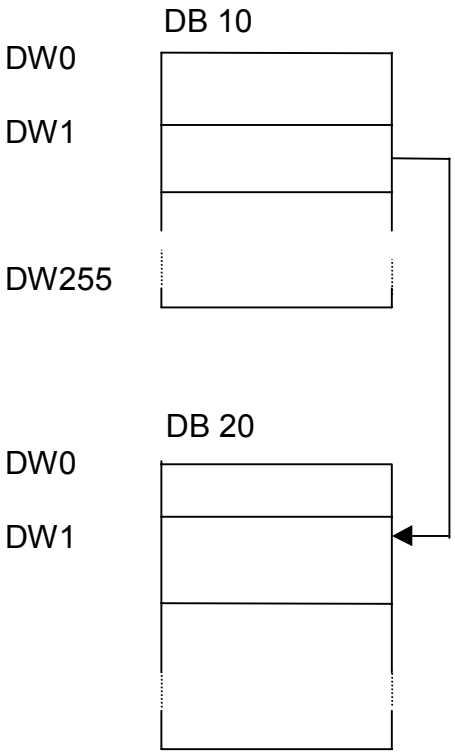


Рис. 2-10: Открытие блока данных и обращение к словам данных

После открытия блока данных все последующие инструкции работают с областью операндов D открытого блока.

Открытый блок данных остается действительным до тех пор, пока по инструкции перехода (например, SPA/SPB PB20)не будет продолжена обработка программы в другом блоке.

Если в этом блоке теперь открыт другой блок данных, то это действительно только в вызванном блоке (PB 20). После возврата в вызывающий блок снова становится действительным старый блок данных.

Важно!

Открытый блок данных остается действительным пока:

a) не открыт другой блок данных

b) не произведен возврат а вызывающий блок данных

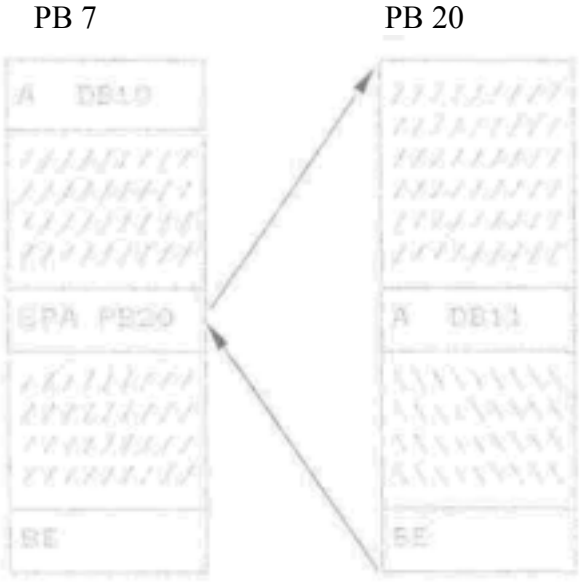
c) вызванный блок не завершается BE.

Пример: Область действия блоков данных

В программном блоке PB 7 открыт блок данных DB 10 (A DB 10). С этого момента обрабатываются данные этого блока данных.

После вызова (SPA PB 20)программного блока PB 20 начинается обрабатываться он. Блок данных DB 10, однако, действителен как и прежде. Сразу после открытия блока данных DB 11 (A DB 11) меняется область данных. До конца программного блока PB 20 (BE) действителен теперь блок данных DB 11.

После перехода назад в программный блок PB 7 снова становится действительным DB 10.



/ / / Зона действия DB 10
\ \ \ Зона действия DB 11

Рис. 2-11: Область действия вызываемых блоков данных

2.4.4. Специальные блоки данных

Блоки данных DB 0, DB 1, DB 2, DX 0 зарезервированы для определенных функций. Они обрабатываются операционной системой и не могут быть использованы для других целей.

- **Блок данных DB 0 (см. Раздел 8.2.2)**

  Блок данных DB 0 содержит список адресов с начальными адресами всех блоков, которые находятся в памяти пользователя или в RAM блоков данных процессора. Этот список адресов создается операционной системой при инициализации (после каждого включения напряжения и после полного стирания), а при вводе или изменении блока с помощью PG автоматически обновляются.

- **Блок данных DB 1 (см. Раздел 10.3)**

Блок данных DB 1 содержит список цифровых входов и выходов (P-периферия с относительными адресами байтов от 0 до 127), а также входных и выходных маркеров связи, которые соответствуют данному процессору и, в данном случае, длину таймерного набора.

В многопроцессорном режиме пользователь должен создавать DB 1 для каждого участвующего в работе процессора. В однопроцессорном режиме DB 1создается для того, чтобы уменьшить время цикла, так как только указанные в DB 1входы и выходы, входные и выходные маркеры связи или таймеры актуализируется.

- **Блок данных DB 2 (см. Раздел 4.4.3)**

Блок данных DB 2 служит для параметрирования структуры регулятора R64 пользователем. Функции регулировки могут поставляться как программный продукт и работать при поддержке операционной системы.

По регулировке смотрите Описание “Компактный регулятор в R-процессоре AG 135 U”, каталоговый номер C79000-B8500-C365-03.

- **Блок данных DX 0 (см. Раздел 7)**

Блок данных DX 0 содержит предварительные установки определенных функций операционной системы (например, при обработке запуска). Пользователь может эти предустановки в DX 0 изменить, с тем, чтобы задать некоторым функциям операционной системы свои требования.

3. Обработка программ

3.1. Обзор

Программа пользователя на языке STEP 5 может обрабатываться различными способами.

Большую часть времени программа работает в цикле. При этом организационный блок OB1 обрабатывается циклически, и управляемая им программа пользователя выполняется с самого начала последовательно блок за блоком.

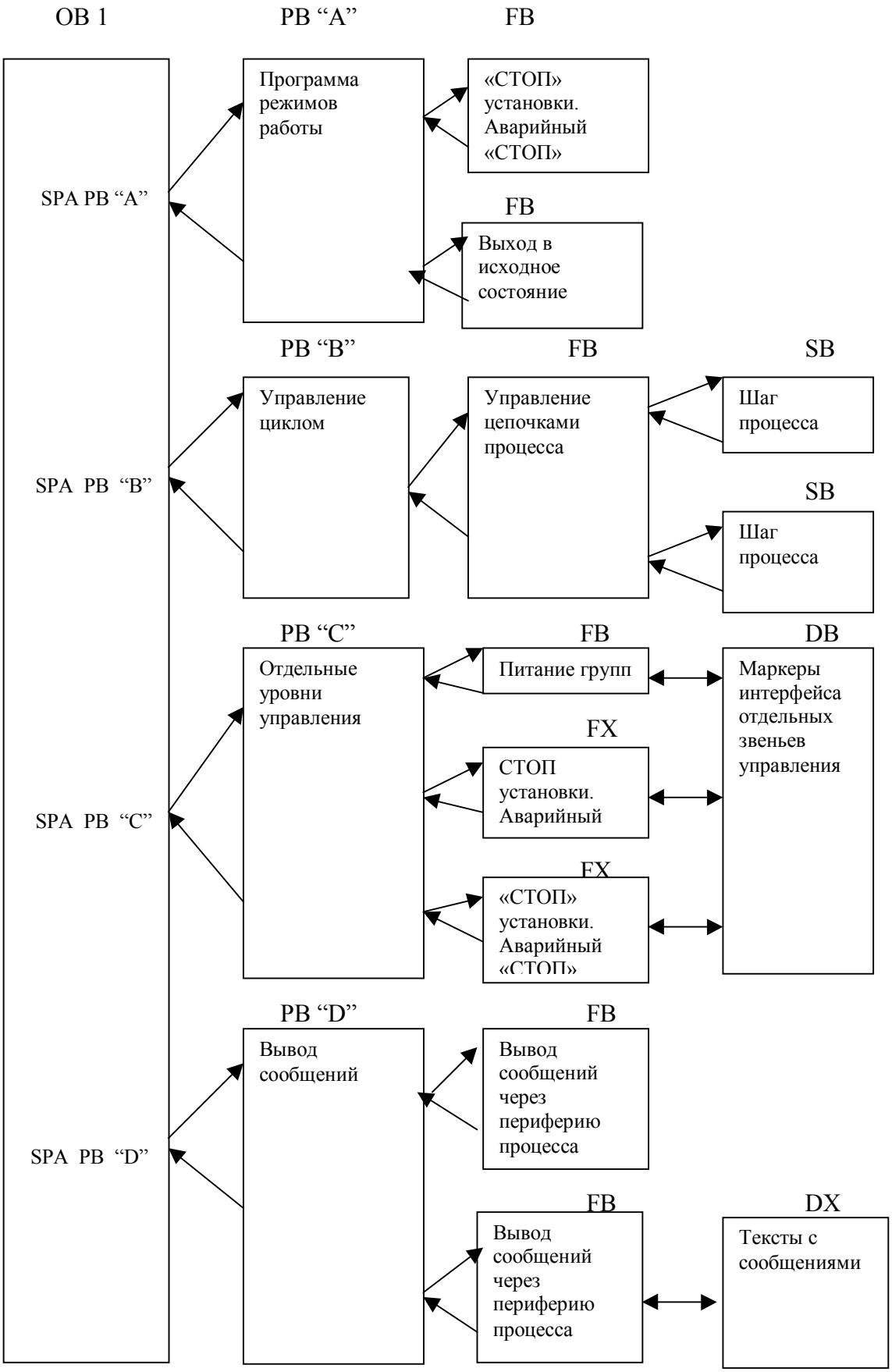
3.2. Организация программы

Организуя программу, вы устанавливаете, в какой последовательности должны обрабатываться написанные вами блоки. Для этого вы программируете в организационных блоках условные или безусловные вызовы требуемых блоков.

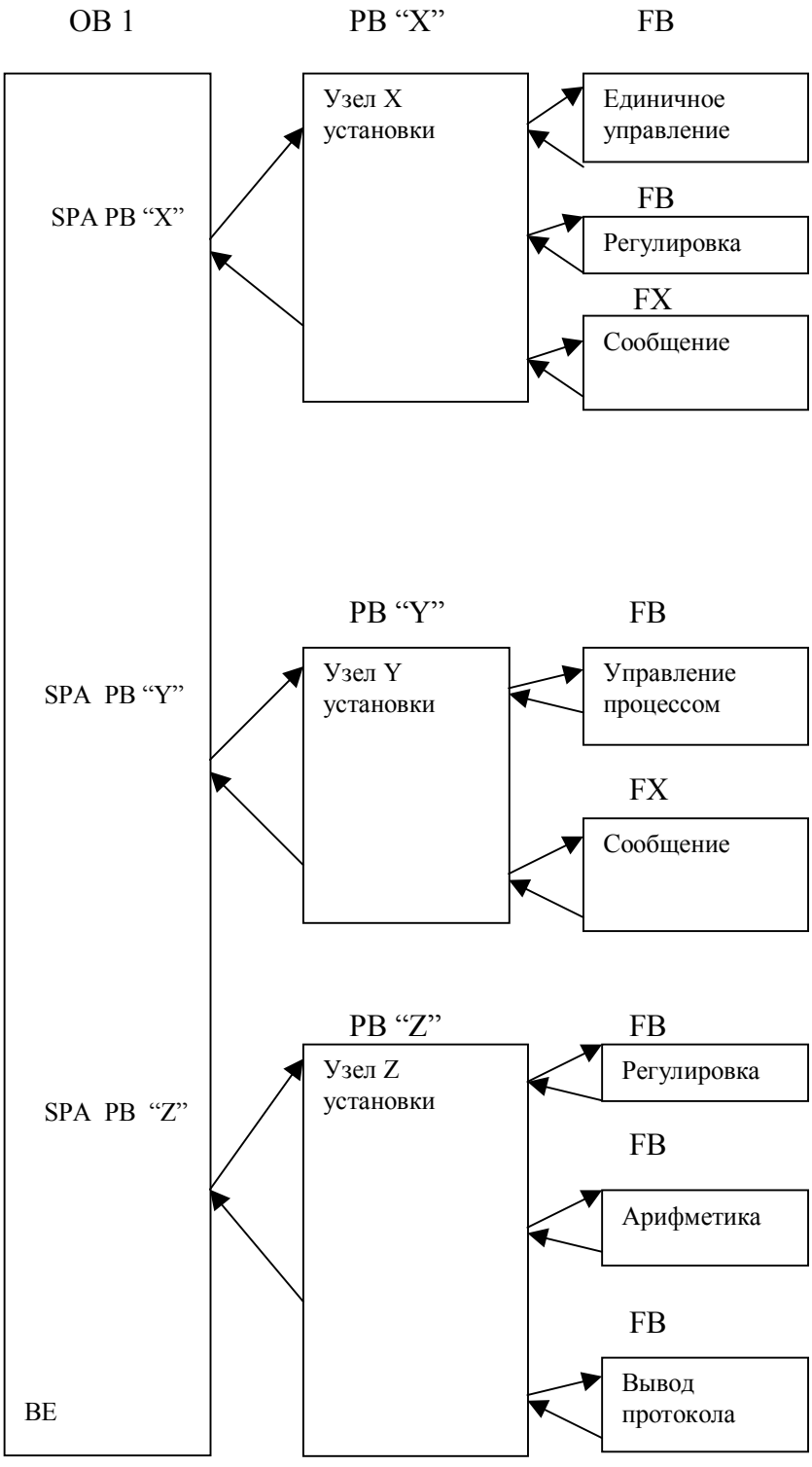
В программе любой организационный, программный, функциональный или шаговый блок может вызвать другие программные, функциональные или шаговые блоки в любой комбинации (друг за другом или один из другого путем вложения).

Программу пользователя целесообразно строить таким образом, чтобы ее структура по возможности соответствовала функциональной схеме установки (тогда каждому функциональному узлу установки будет соответствовать законченная часть программы).

Пример 3-1: Организация программы пользователя по структуре программы.



Пример 3-2: Организация программы пользователя по структуре установки.



Внимание!

Вы можете осуществить вложения максимально 62 блоков. Если вызвано более 62 блоков, процессор выдаст ошибку.

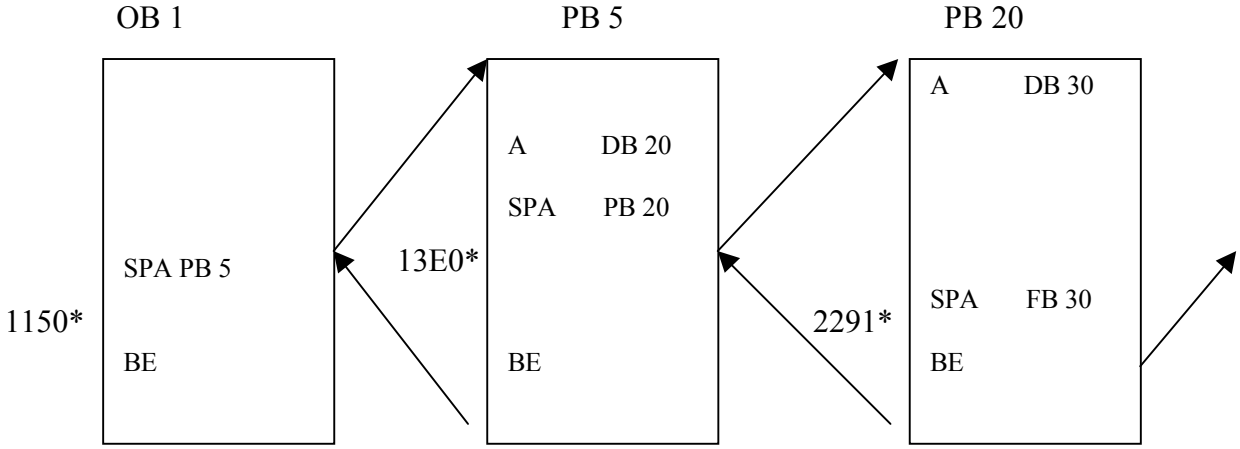
Глубину вложения в программе вы должны определить следующим образом:

- Сложите все запрограммированные вами блоки (в примере на следующей странице это 4 OB)
- Прибавьте глубину вложения в каждом организационном блоке (в примере: 2+2+1+0=5).
- Сложите обе суммы и получите глубину вложения вашей программы (в следующем примере: 4+5= глубина вложения 9). Глубина вложений не может превысить значение 62.

Расположение блоков в памяти пользователя (или DB-RAM)устанавливается путем указания адреса начала блока: этот адрес в кодовых блоках является адресом соответствующей ячейки памяти, в которой находится первая команда языка STEP 5 данного блока (для FB и FX это команда SPA в списке формальных операндов); в блоках данных это адрес ячейки памяти, в которой находится слово данных DW0 данного блока.

Для того, чтобы процессор при вызове блока (SPA/SPB xx, A DB) мог найти его в памяти, все начальные адреса запрограммированных блоков блока данных DB0. DB0 обрабатывается системной программой и не может быть вызван пользователем!

Для того, чтобы после обработки вызванного блока найти обратный путь в вызывающий блок, процессор сохраняет при каждом вызове нового блока адрес возврата. Адрес возврата – это адрес ячейки памяти, в которой расположена инструкция STEP 5, следующая после команды вызова блока. Кроме того, процессор запоминает начальный адрес и длину блока данных, действительного в данный момент.

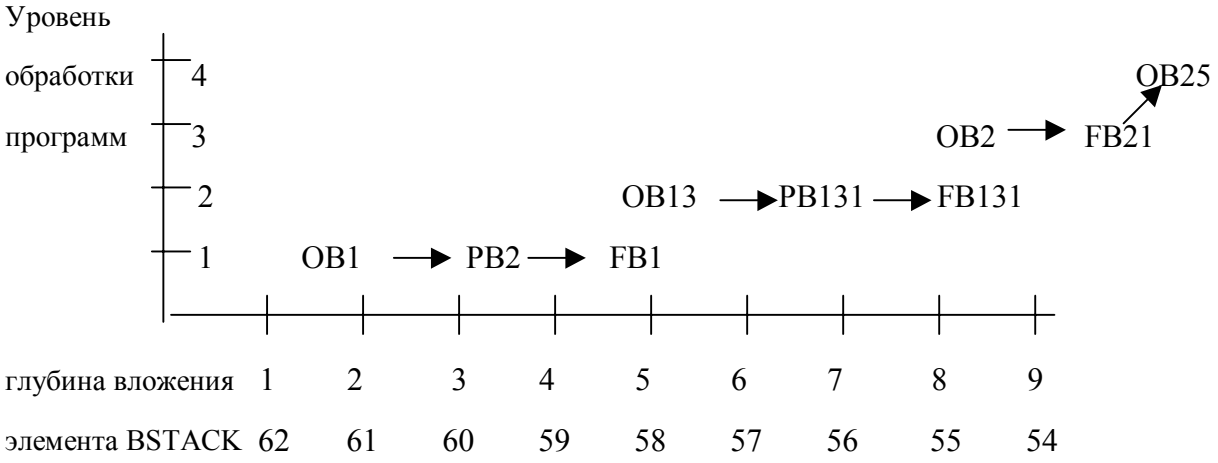


*Адрес возврата

Эти данные заводятся в стек блоков (BSTACK). BSTACK заполняется снизу вверх: первый ввод соответствует элементу В- стека 62, второй ввод- элементу В- стека 61, и т.д. Если вызванный блок полностью восстановлен и осуществлен обратный переход в вызывающий блок, то соответствующий ввод удаляется из стека.

После 62 вводов (элемент В- стека 1) стек блока заполнен. При превышении допустимой глубины вложения процессор переходит в состояние «СТОП».

Пример: Глубина вложения блоков и стек блоков (BSTACK)



3.1.2. Хранение программы

Для того, чтобы процессор мог обрабатывать программу пользователя, необходимо загрузить её в память программ. Для этого имеется две возможности:

a) Если Вы используете вставляемый модуль (субмодуль) RAM, то можете свою программу перевести непосредственно из программатора в процессор.

Использование модуля RAM позволяет быстро изменять содержимое памяти. Буферная батарея предотвращает стирание программы пользователя в памяти при отключении напряжения питания (0 буферной батареи см. Руководство по эксплуатации центрального устройства S5-135U).

Запрограммированные блоки запоминаются в модуле RAM в любой последовательности. При изменении блока изменяется последовательность блоков памяти.

Блоки данных DB и DX заносятся в модуль RAM до заполнения, а затем в RAM блоков данных процессора.

b) Вся программа пользователя заносится в субмодуль EPROM. В модуле EPROM программа полностью защищена при отключении напряжения питания и неисправностях буферной батареи.

Содержимое модуля EPROM не может в дальнейшем меняться. Поэтому блоки, которые содержат изменяемые в процессе выполнения программы данные, при перезапуске необходимо копировать из модуля EPROM в модуль RAM процессора (см. Специальные функции OB254, OB255, Раздел 6.4.5).

Если процессор находит ошибку в памяти пользователя, то он требует полного стирания и переходит в режим «стоп». После полного стирания Вы должны заново загрузить в память свою программу.

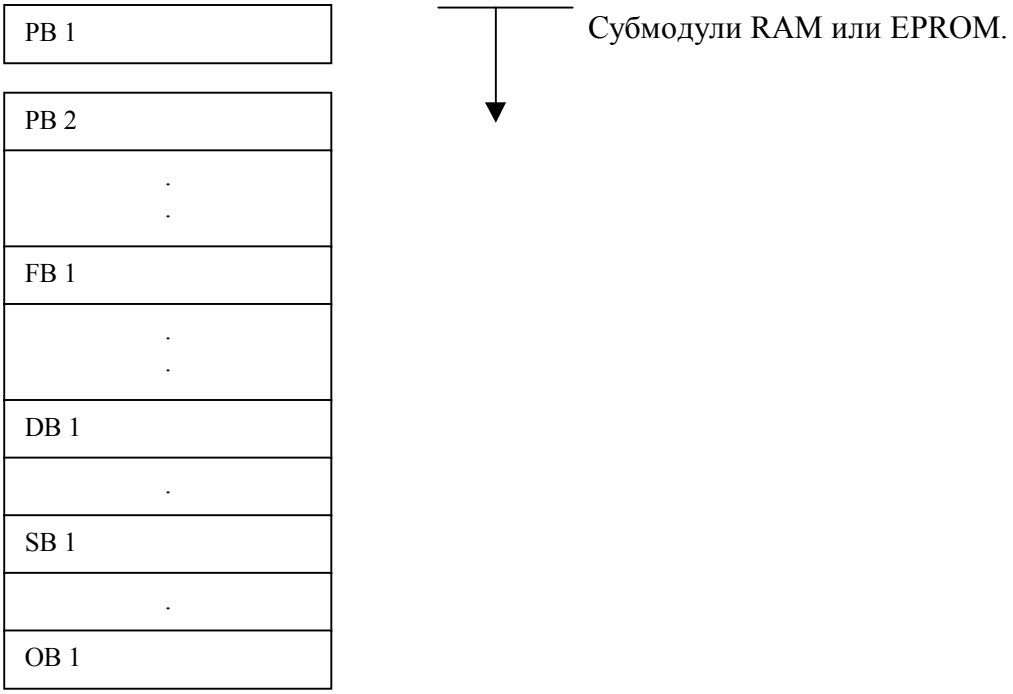


Рис. 3.4: Расположение блоков в памяти программ

3.1.3. Обработка программ пользователя на языке STEP 5

Программу пользователя можно обрабатывать различными способами. Обычно, в свободно-программируемых системах управления преобладает циклическая обработка программ.

В конце запуска операционная система автоматически вызывает организационный блок OB1 (или FBO). Процессор начинает с первой инструкции STEP5 программы и обрабатывает их всех последовательно. Если он достигает конца программы, то снова начинает следующий цикл с начала программы.

В каждом цикле операционная система выполняет следующие действия:

- Запускается время цикла.
- Обновляется область отображения входов.
- Обновляет входные маркеры связи.
- Вызывает интерфейс пользователя.
- Обновляет область отображения выходов
- Обновляет выходные маркеры связи.

Время цикла

Операционная система контролирует время, которое необходимо процессору для обработки программы пользователя. В начале обработки операционной системой запускается контролируемое время цикла.

Стандартное максимально допустимое значение устанавливается на 150ms.

Как пользователь, вы имеете возможность самостоятельно установить время цикла или во время циклической обработки программы заново его запустить (см. DX0, специальные функции OB 221 и 222).

Общее время цикла состоит из времени обработки программы пользователя и времени, необходимого для выполнения циклической части операционной системы (см. рисунок на следующей странице).

Время выполнения программы пользователя, в свою очередь, состоит из суммы времен выполнения всех вызываемых блоков при одном проходе программы (от вызова OB1 или FB0 до конца обработки). Таким образом, если вы определенный блок вызываете n раз, то должны в своей сумме предусмотреть данное время n раз.

Область отображения входов и выходов (PAE и PAA)

В начале обработки программы STEP 5 состояние сигналов модулей входов периферии считывается и передается в область отображения входов (в системной памяти процессора). Область отображения входов обрабатывается программой пользователя с помощью инструкций STEP 5, и в конце цикла создается область отображения выходов. Состояние сигналов области отображения выходов передается на модули периферийных выходов.

Область отображения процесса является, таким образом, областью памяти, содержание которой считывается с периферии или выдается на периферию один раз в цикл.

Важно!

Область отображения процессов существует только для байтов входов и выходов P-периферии с адресацией от 0 до 127!

Маркеры связи

Маркеры связи служат для обмена данными между отдельными процессорами (в многопроцессорном режиме) или между процессором и коммуникационным процессором.

Перед началом обработки программы на языке STEP 5 считываются входные маркеры связи процессора. После обработки программы на языке STEP 5 выходные маркеры связи передаются на координатор и на коммуникационные процессоры.

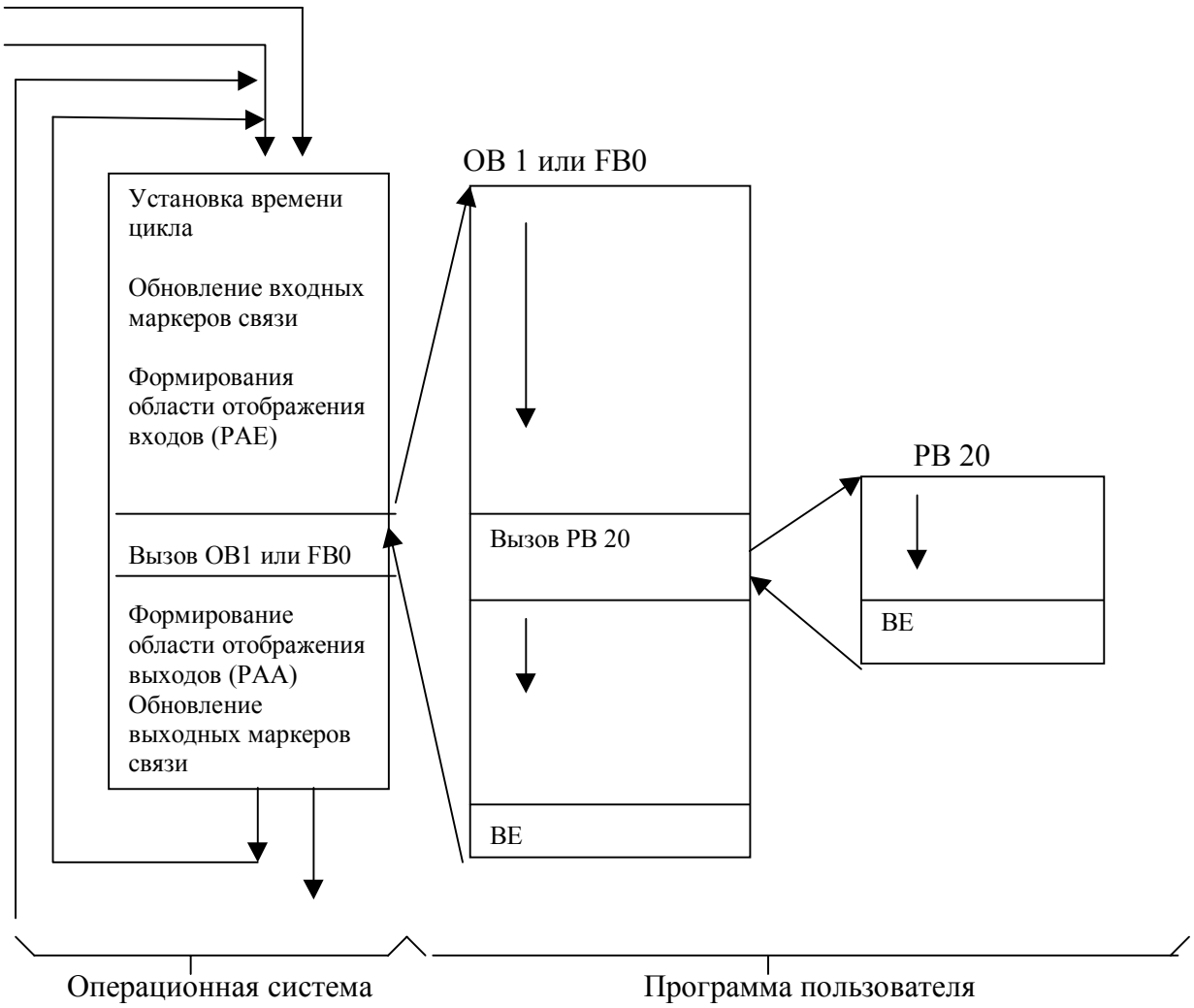


Рис. 3.5: Циклическая обработка программы.

Точки прерывания

Циклическая обработка программы может быть прервана на короткое время из-за:

- обработки программы, управляемой по прерыванию от процесса.
- Обработки программы, управляемой по времени.

Вы можете прервать или совсем прекратить обработку:

- при возникновении аппаратной или программной ошибки
- с помощью средств управления (PG-функции, селектор режимов с STOP).

3.1.4. Предустановки для обработки программ

Как пользователь, Вы имеете две возможности управления работой процессора при запуске, во время циклической обработки программы и в случае ошибки.

a) Путем программирования организационных блоков с OB1 по OB34 (интерфейсы между операционной системой и программой пользователя, см. Раздел2.2.3) и

b) путем программирования дополнительного блока данных DX 0 (см. Главу 7).

Организационные блоки с OB1 по OB34 являются интерфейсами между операционной системой и программой пользователя, т.к. они, с одной стороны, вызываются из операционной системы, а с другой стороны, как обычные программы пользователя, могут быть заполнены командами на языке STEP5. В этих организационных блоках можно осуществить вызов других блоков. Пользователь с помощью записанных в эти OB программ может задать реакцию процессора на определенную ситуацию.

Организационные блоки с OB1 по OB34 становятся действительными, как только Вы их загрузите в память программ (также в режиме работы).

Если они не запрограммированы пользователем, то либо не возникает никакой реакции процессора, либо, в большинстве случаев появления ошибки, процессор переходит в состояние «СТОП» (см. раздел 5.4).

Другая возможность управления процессором состоит в программировании блока данных DX0.

Функции, которые выполняет операционная система, обычно заданы заранее.

Путем ввода определенных параметров DX0 Вы можете изменить стандартные предустановки некоторых функций операционной системы.

Как и организационный блок, DX0 загружается в процессе работы в память программ. Однако он будет действителен в следующем новом пуске.

Если DX0 не запрограммирован пользователем, то предустановки остаются действительными.

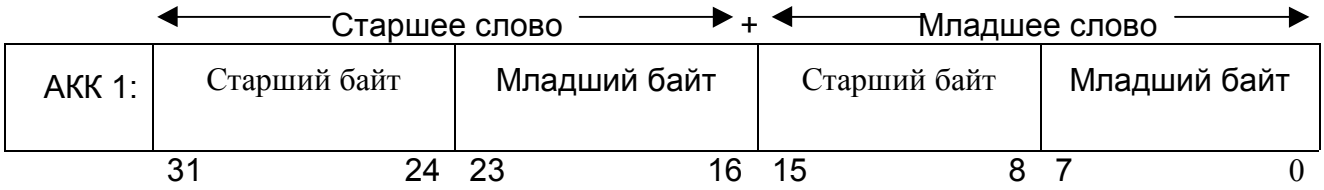
3.2 Набор операций языка STEP5 с примерами программ

Операции STEP5 разделяются на различные группы:

- К двоичным операциям относятся двоичные функции, операции запоминания, операции с таймерами и счетчиками.
- К цифровым операциям относятся операции загрузки и передачи, операции сравнения и арифметические операции.
- К организационным функциям принадлежат операции переходов, операции останова и конца блока, инструкции для создания и вызова блока данных и т.д.

Аккумуляторы как вспомогательные регистры

Большая часть операций языка STEP5 использует в качестве операндов и как место хранения результата два регистра (32-битовые) – аккумулятор 1 (АКК 1) и аккумулятор 2 (АКК 2).



Содержимое аккумуляторов всегда зависит от обработанной команды STEP5.

Примеры:

- При операциях загрузки в качестве приемника всегда используется АКК 1. Старое содержимое АКК 1 перемещается в АКК 2 (STACK LIFT). Содержимое аккумуляторов 3 и 4 при выполнении любой операции загрузки не изменяется.
- Арифметические команды объединяют содержимое АКК 1 и АКК 2, записывают результаты в АКК 1 и передают содержимое АКК 3 в АКК 2, а содержимое АКК 4 в АКК 3 (STACK DROP).
- При прибавлении констант (ADD BF/KF) к содержимому АКК 1 содержимое АКК 2, АКК 3 и АКК 4 не меняется.

Индикация результата

Есть команды для обработки информации, состоящей из отдельных битов, и команды для обработки информации в виде байтов и слов (8, 16 и 32 бита).

В обеих группах есть команды, которые устанавливают биты индикации и команды, которые интерпретируют их (см. Список инструкций, влияние на биты индикации). Есть биты индикации результата битовых команд (биты 0 . . . 3) и биты индикации результата словных команд (биты 4. . . 7), которые соответствуют группам команд. Байт индикации может быть введен с помощью программатора в виде:

	Биты индикации словные				Биты индикации битовые			
	ANZ1	ANZ0	OV	OS	OR	STA	VKE	ERAB
Бит	7	6	5	4	3	2	1	0

Биты индикации результата битовых команд:

$\overline{\text{ERAB}}$ Первый опрос

Начало логической операции. В конце цепочки логической операции (например, операций с памятью) ERAB устанавливается в 0. Команды, которые устанавливают ERAB в 0 (например, команда присвоения = A2.4), ограничивают VKE (см. соответствующую колонку в списке команд в разделе 8), т.е. в дальнейшем результат логической операции может быть интерпретирован (например, в командах, зависимых от VKE), но не может быть изменен. Только после первой логической команды (= начало цикла сканирования)VKE устанавливается в новое значение, и ERAB устанавливается в 1.

VKE Результат логической операции

Результат битовой логической операции. Команда «Истинно» при выполнении команды сравнения (смотри приложение: Список команд, двоичные логические операции или операции сравнения).

STA Статус

В битовых командах определяется логическое состояние бита, который только что был опрошен или установлен. Статус обновляется при двоичных логических операциях, исключая U(, 0(,),0) и операциях с памятью.

OR Или

Информация для CPU, что следующая логическая операция «И» должна быть выполнена пред логической операцией «ИЛИ» (И перед ИЛИ).

Биты индикации результата словных команд:

OV Переполнение

Определяется, произошло ли переполнение (превышение допустимых границ) при выполнении последней арифметической операции.

OS Запоминание переполнения

Бит переполнения запоминается. Этот бит используется для индикации уровня выполнения нескольких арифметических операций, на котором произошло переполнение.

ANZ1 и ANZ0

Биты кодируемого результата, которые интерпретируются в соответствии со следующей таблицей:

Биты результата		Результат вычисления с фиксированной запятой	Цифровая логическая операция	Сравнение содержимого АКК1 и АКК2	Сдвиг: последний сдвинутый бит
ANZ1	ANZ0				
0	0	Результат = 0	= 0	АКК2 = АКК1	0
0	1	Результат < 0	-	АКК2 < АКК1	-
1	0	Результат > 0	< > 0	АКК2 > АКК1	1

Для непосредственной обработки битов индикации в вашем распоряжении имеются операции переходов по условию (см. Раздел 3.2.2).

3.2.1. Основной набор операций

о Двойные логические операции

Операция		Параметр	Функция
)			Скобку закрыть
U(			Операция “И” для выражения в скобках
O(			Операция “ИЛИ” для выражения в скобках
O			Операция “ИЛИ” над функциями “И”
U			Операция “И”
O			Операция “ИЛИ”
	E	0.0 . . .127.7	с опросом входа на состояние «1»
	A	0.0 . . .127.7	с опросом выхода на состояние «1»
	M	0.0 . . .255.7	с опросом маркера на состояние «1»
N	E	0.0 . . .127.7	с опросом входа на состояние «0»
N	A	0.0 . . .127.7	с опросом выхода на состояние «0»
N	M	0.0 . . .255.7	с опросом маркера на состояние «0»
N	D	0.0 . . .255.115	с опросом слова данных на состояние «0»
	T	0 . . .255	с опросом таймера на состояние «1»
N	T	0 . . .255	с опросом таймера на состояние «0»
	Z	0 . . .255	с опросом счетчика на состояние >0
N	Z	0 . . .255	с опросом счетчика на состояние =1

Двоичные логические операции создают результат логической операции (VKE).

В начале выполнения цепочки логических операций (первый опрос) результаты первой логической операции зависят только от состояния опрашиваемых сигналов и от того, используется или нет их инверсное состояние (N-отрицание); однако, они не зависят от типа логической операции (U = UND <”И “>, O = ODER <” ИЛИ“>).

Во время выполнения цепочки логических операций VKE формируется из типа логических операций, предыдущего состояния VKE и состояния опрашиваемого сигнала. Цепочка логических операций завершается командой, ограничивающей VKE (ERAB = 0) (например, операцией установки).

VKE остается без изменений до тех пор, пока не начинается следующий «первый опрос». После этого VKE может быть интерпретирован, но не может быть установлен.

Пример:

Программа	Статус	VKE	/ERAB	
:				
= A 0.0	0	0	0 ←	Ограничение VKE, конец цепочки логической операции
U E 1.0	1	1	1 ←	Первый опрос
U E 1.1	1	1	1	
U E 1.2	0	0	0	
= A 0.1	0	0	0 ←	Ограничение VKE, конец цепочки логической операции

о Операция установки/сброса

Операция	Параметр	Функция
S		Установка
R		Сброс
F		Присвоение
E	0.0 … 127.7	входов в PAE
A	0.0 … 127.7	выходов в PAA
M	0.0 … 255.7	битов маркеров
D	0.0 … 255.15	битов слова данных

о Функции загрузки, передачи и сравнения

Операция	Параметр	Функция
L		Загрузка
T		Передача
EB	0…127	входного байта в/из PAE
EW	0…126	входного слова в/из PAE
ED	0…124	входного двойного слова в/из PAE
AB	0…127	выходного байта в/из PAA
AW	0…126	выходного слова в/из PAA
AD	0…124	выходного двойного слова в/из PAA
MB	0…255	маркерного байта
MW	0…254	маркерного слова
MD	0…252	маркерного двойного слова
DR	1…255	младшего байта слова данных из DB, DX
DL	1…255	старшего байта слова данных из DB, DX
DW	1…255	слова данных
DD	0…254	двойного слова данных
PB/	0…127	периферийного байта модулей цифровых
PY		входов/выходов (зависит от PAE/PAA)
PB/	128…255	периферийного байта модулей аналоговых
PY		входов/выходов
QB	0…255	байта расширенной периферии (область Q)

о Функции загрузки, передачи и сравнения (продолжение)

Операция	Параметр	Функция
L		Загрузка
T		Передача
P W	0…126	периферийного слова модулей цифровых входов/выходов (область P)
	128…254	периферийного слова модулей аналоговых
Q W	0…254	входов/выходов (область P)
		слова расширенной периферии (область Q)
L		Загрузка
K M	любая битовая комб. (16 бит)	константы в виде битовой комбинации
K H	0. . . FFFF	константы в 16-ричном коде
K F	от –32768 до +32768	константы в виде числа с фиксированной запятой
K Y	0. . .255для каждого байта	двухбайтовые константы
K B	0. . .255	однобайтовые константы
K C	2 любых алфав. -цифр. символа	константы (2 символа в коде AS I I)
K T	0.0. . .999,3	константы времени
K Z	0. . .999	константы счета
K G	[1])	константы в виде числа с плавающей запятой (32бита)
T	0. . .255	значения таймера
Z	0. . .255	значения счетчика
LC T	0. . .255	BCD-кодированное значение таймера
LC Z	0. . .255	BCD-кодированное значение счетчика
! =		сравнение на равенство
><		сравнение на неравенство
>		сравнение на больше
> =		сравнение на больше равно
<		сравнение на меньше
< =		сравнение на меньше равно
F		двух чисел с фиксированной запятой (16 бит)
D		двух чисел с фиксированной запятой (32бита)
G		двух чисел с плавающей запятой (32бита)

Команда загрузки записывает адресуемое значение в АКК 1, предыдущее значение которого переносится в АКК 2 (STACK LIFT).

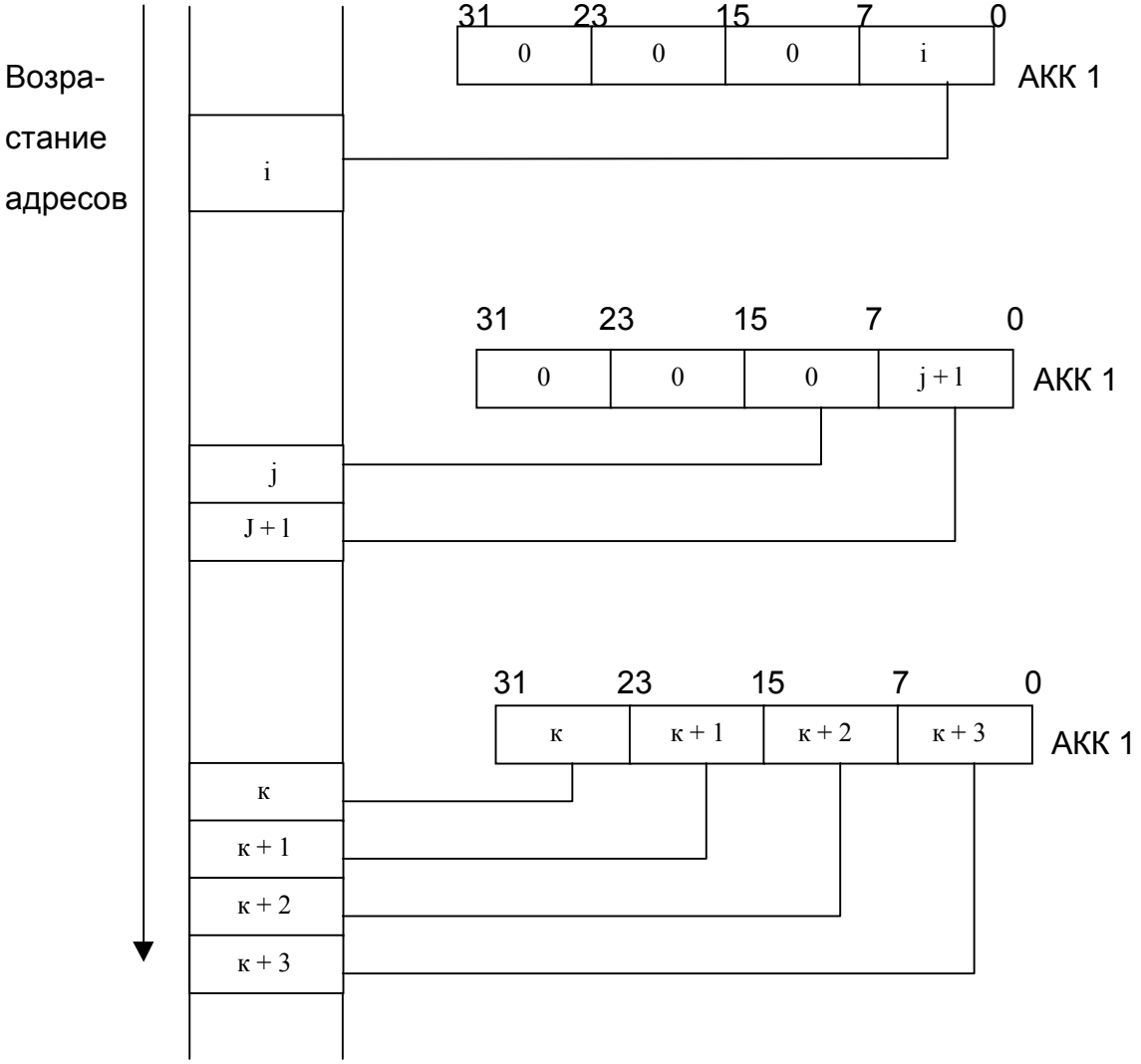
Команда передачи переносит содержимое аккумулятора АКК 1 в адресуемую ячейку памяти.

[1]) ё 0,1469368x10^{-38} . . . ё 0,1701412x10^{39}

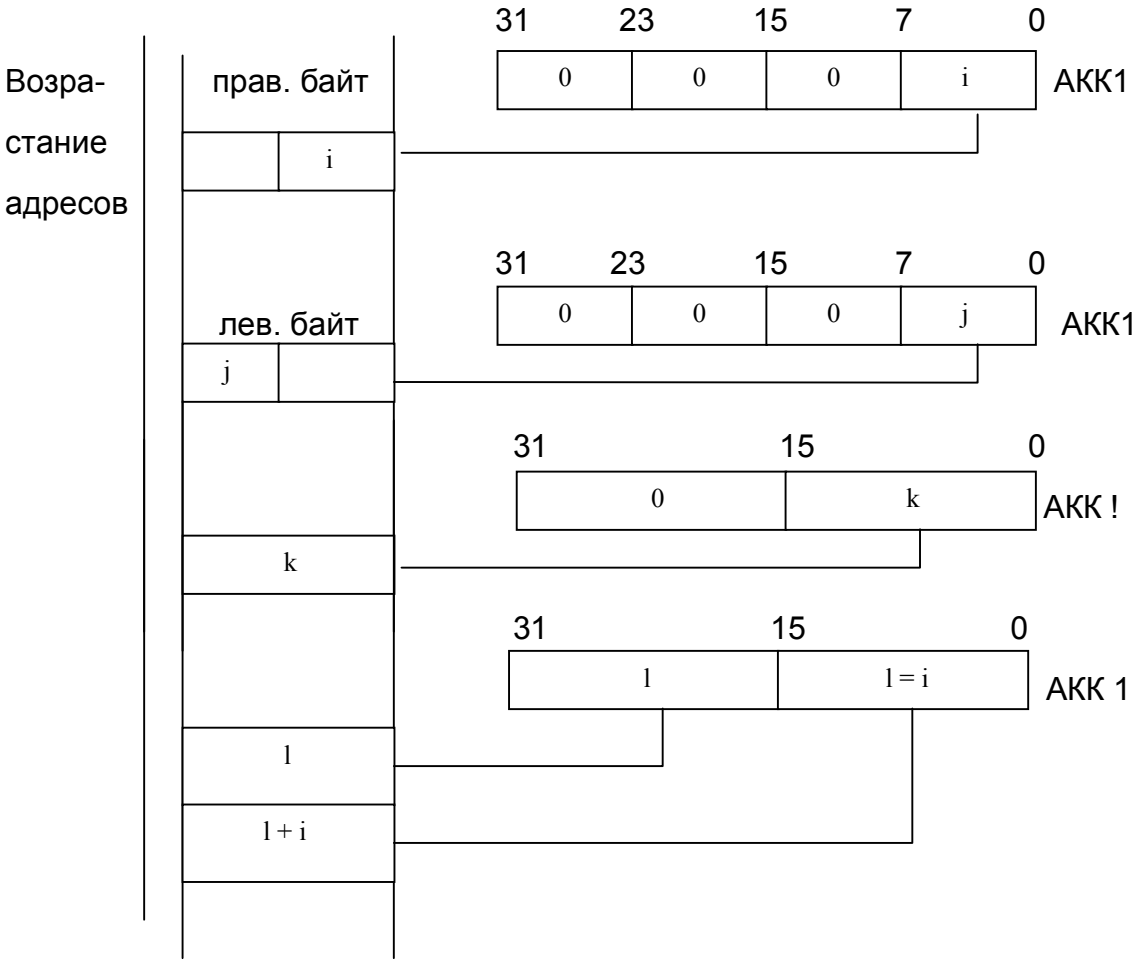
Пример: Загрузка/передача байта, слова, двойного слова из/в область памяти с байтовой организацией (PAE, PAA, маркер, периферия).

:L EW 5 Загрузка байтов 5 и 6 из области PAE в АКК 1.

:L MD 10 Загрузка маркерных байтов с 10 по 13.



Пример: Загрузка/передача байта, слова, двойного слова из/в область памяти с пословной организацией.



Слово или двойное слово внутри памяти располагается, начиная со старшего байта или слова в направлении увеличения адресов.

При загрузке байта или слова неиспользуемые биты в АКК 1 обнуляются.

Операции загрузки не влияют на биты индикации, а операции передачи имеют в качестве результата VKE и индикацию слов ANZ0 и ANZ1. Сравнивается всегда содержимое АКК1 и АКК2 (см. примеры программ и список операций).

Периферия может опрашиваться с помощью операций загрузки и передачи:

1. непосредственно:

   с помощью команд L/T PY, PW, QB, QW или

2. через отображение процесса:

   с помощью команд L/T EB, EW, AB, AW, AD и с помощью логических операций

При операциях передачи T PY 0...127 и T PW 0...126 параллельно обновляется состояние выходов в области отображения процесса.

Область отображения процесса представляет собой область памяти, содержимое которой выдается на периферию только один раз в цикл (Область отображения процесса выходов, PAA) или считывается с периферии (область отображения процесса входов. PAE). Это сделано для избежания частого изменения логического состояния битов внутри одного цикла программы, которые вызвали бы колебания соответствующего выхода периферии.

Обратите внимание на следующие пункты:

- Область отображения входов и выходов имеется для 128 байтов входов и 128 байтов выходов. P- периферии с адресами байтов с 0 по 127.

- Для всей области Q-периферии и области P- периферии с относительными адресами байтов со 128 по255 области отображения процессов не существует! (О разделении области адресов см. Раздел 8.1.2).

- Модули входов/выходов Q-периферии могут быть установлены только в устройстве расширения (не в центральном устройстве).

- В одном устройстве расширения может использоваться либо только P-, либо только Q-периферии.

- В случае, если в устройстве расширения используется P- периферия с адресами P-периферии, то эти адреса для модулей периферии в центральном устройстве не могут быть использованы (двойная адресация!).

о Операции с таймерами и счетчиками.

Перед тем, как загрузить таймер или счетчик, необходимо предварительно значение времени или счета загрузить в АКК1.

Имеют смысл следующие операции загрузки:

Для таймеров: L KT, L EW, L AW, L MW, L DW

Для счетчиков: L KZ, L EW, L AW, L MW, L DW

Операция	Параметр	Функция
SI T	0…255	Запуск таймера в качестве импульса
SV T	0…255	Запуск таймера в качестве удлиненного импульса
SE T	0…255	Запуск таймера в качестве задержки включения
SS T	0…255	Запуск таймера в качестве запоминающей задержки включения
SA T	0…255	Запуск таймера в качестве задержки включения
R T	0…255	Сброс таймера
S Z	0…255	Установка счетчика
R Z	0…255	Сброс счетчика
ZV Z	0…255	Прямой счет
ZR Z	0…255	Обратный счет

При использовании операций таймеров SI, SE, SV, SS, SA и S или счетчиков, значение, записанное в АКК1 переносится в область памяти, соответствующую данному таймеру/счетчику (соответствует команде передачи), и начинает выполняться соответствующая операция.

Если значение таймера или значение счетчика загружается при помощи EW, AW, MW, DW, то соответствующее слово должно иметь следующую структуру:

Для значения таймера

Номер бита



Пример: Установка во времени 127 секунд.

Значение битов.



Важно!

Каждый запуск таймера связан с погрешностью, равной единице временного растра таймера!

Таким образом, если вы запускаете таймер с временным растром «1» (=100мс) n раз, то вы получите погрешность с n x 100мс.

Поэтому следует выбирать по возможности наименьший базис времени (растр) таймера (временный растр<< значения времени)!

Пример: длительность таймера 4сек.

Неправильно: 1сек x 4

Следует: 10мс x 400

Для значения счетчика



Пример: должно быть задано значение счетчика 127

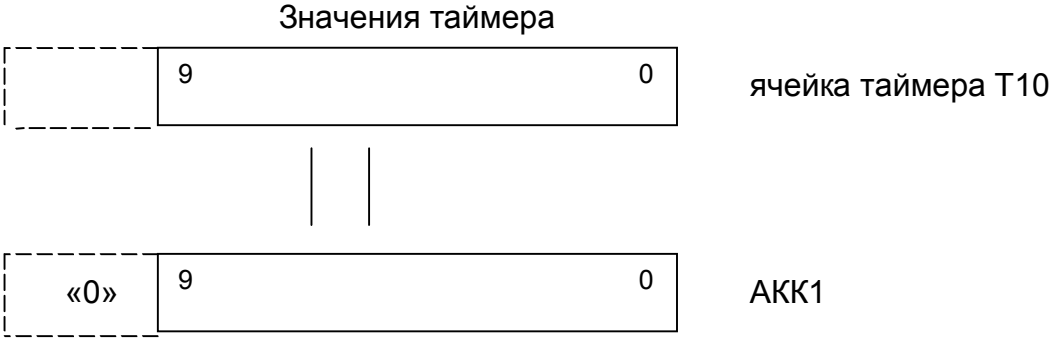
Значение битов:



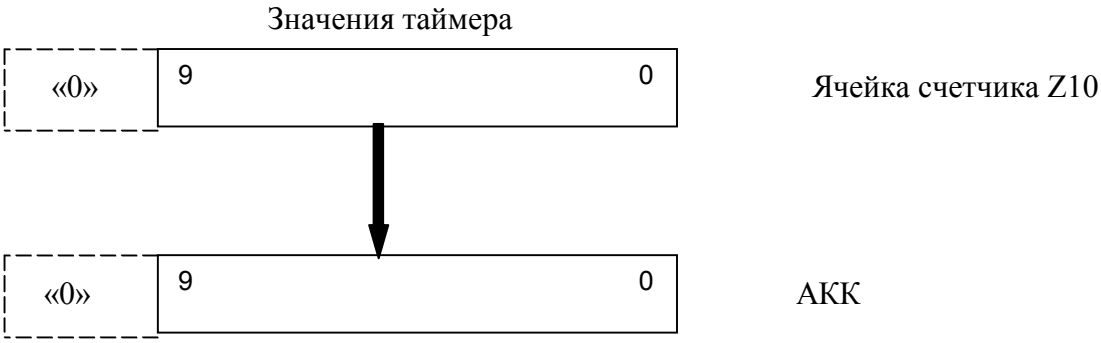
Значение таймера или счетчика хранится в области памяти, отведенной для этих целей, в виде двоичных кодов. Для того, чтобы опросить таймер или счетчик, значение из области таймеров или счетчиков должно быть перенесено в АКК1 непосредственно или в коде BCD.

Пример:

Непосредственная загрузка значения таймера:

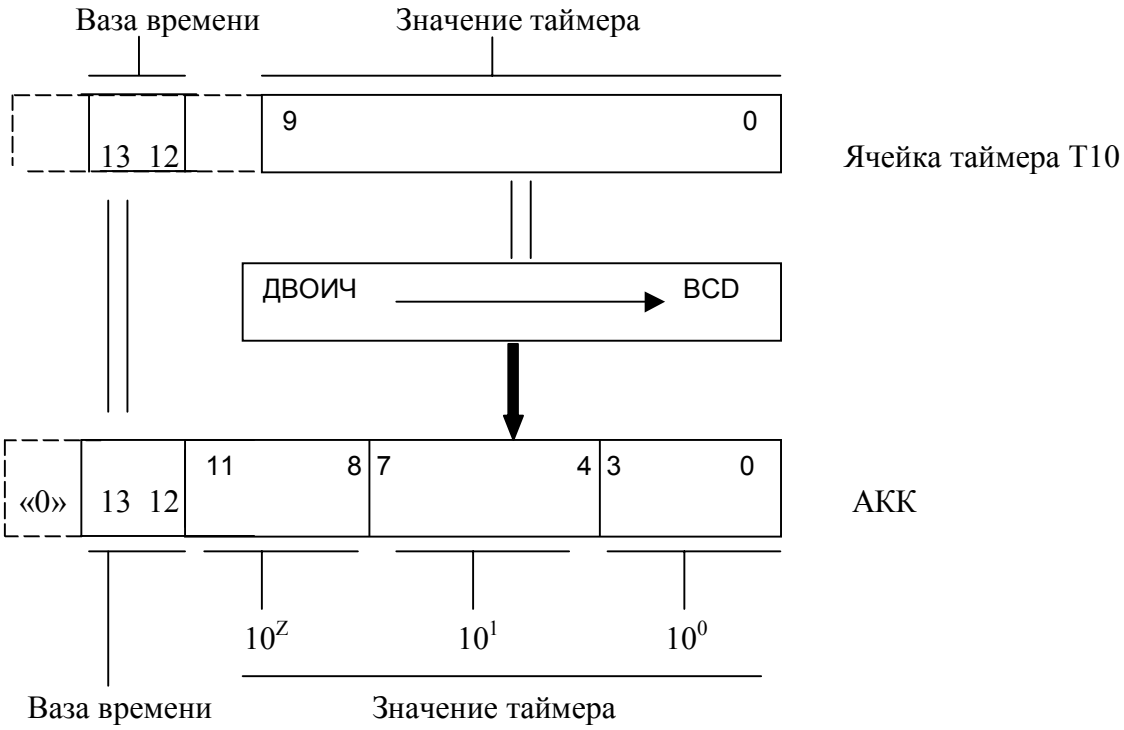


L T 10 непосредственная загрузка двоичного значения таймера T 10 в аккумулятор.
Значение базы времени здесь не загружается
Непосредственная загрузка значения счетчика:



L Z 10 непосредственная загрузка двоичного значения счетчика Z10 в АКК

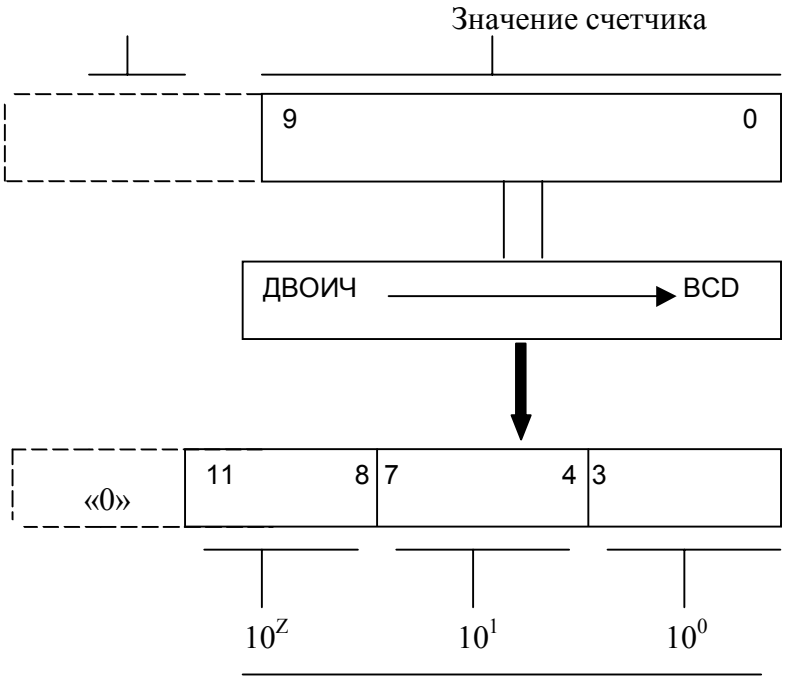
Кодированная загрузка значения таймера:



LC T 10 Кодированная загрузка значения таймера T 10 и базы времени в АКК

Значение базы времени здесь тоже загружается.

Кодированная загрузка значения счетчика:



LC Z 10 кодированная загрузка значения счетчика Z 10 в аккумулятор

При кодированной загрузке значение битов 14 и 15 ячейки таймера и значение битов с 12 по 15 ячейки счетчика не загружается. На их место в АКК записывается «0». После этого содержимое АКК может быть нужным образом обработано.

o Арифметические операции

Операция	Параметр	Функция
+ F		Сложение двух чисел с фиксированной запятой (16 бит)
- F		Вычитание двух чисел с фиксированной запятой (16 бит)
x F		Умножение двух чисел с фиксированной запятой (16 бит)
: F		Деление двух чисел с фиксированной запятой (16 бит)
+ G		Сложение двух чисел с плавающей запятой (32 бита)
- G		Вычитание двух чисел с плавающей запятой (32 бита)
x G		Умножение двух чисел с плавающей запятой (32 бита)
: G		Деление двух чисел с плавающей запятой (32 бита)

Арифметические операции выполняются над содержимым аккумуляторов 1 и 2 (см. список операций, Раздел 8). Затем результат помещается в АКК 1. Арифметические регистры изменяются при арифметических операциях следующим образом:

Состояние до операции <AKK 1> <AKK 2> <AKK 3> <AKK 4>

Состояние после операции < результат> <AKK 2> <AKK 3> <AKK 4>

Предыдущее содержимое аккумулятора <AKK 2> теряется!

Помните, что среди дополнительных операций имеются команды сложения и вычитания двойных слов с фиксированной запятой.

о Вызов блока

Операция		Параметр	Функция
SPA			Переход безусловный
SPB			Переход по условию (при VKE = 1)
	O B	1…39	к организованному блоку
	O B	40…255	к системной программе специальной функции
	P B	0…255	к программному блоку
	F B	0…255	к функциональному блоку
	S B	0…255	к шаговому блоку
BA	F X	0…255	безусловный переход к функциональному блоку FX
BAB	F X	0…255	условный переход к функциональному блоку FX (только если
A	D B	3…255	вызов блока данных DB
A	D X	1…255	вызов блока данных DX
BE			конец блока
BEB			конец блока условный (только если VKE=1)
BEA			безусловный конец блока

о Пустые операции

Операция	Параметр	Функция
NOP	0	Пустая операция
NOP	1	Пустая операция
BLD	0…255	Команды формирования изображения на экране программатора (не интерпретируется как команда процессора)

о Команда остановки

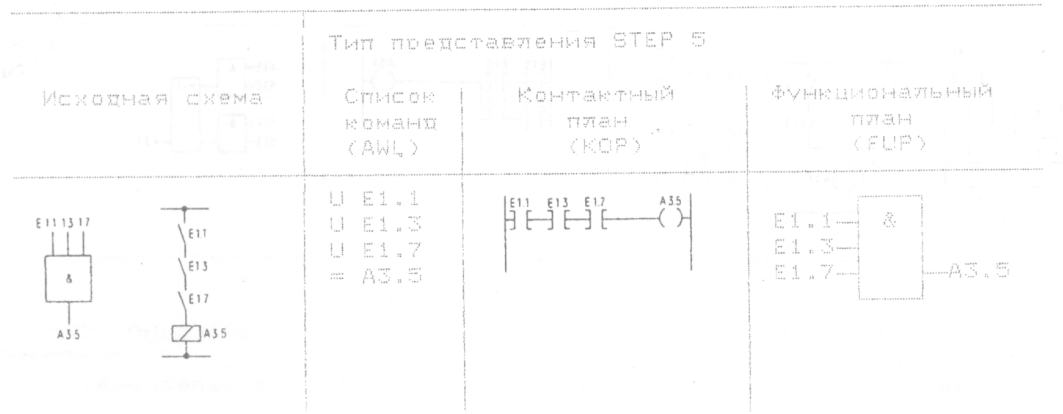
Операция	Параметр	Функция
STP		Процессор переходит в состояние «СТОП»

Примеры программирования логических функций, таймеров, счетчиков и функций сравнения.

о Логические операции

Операция «И»

Операция "И"

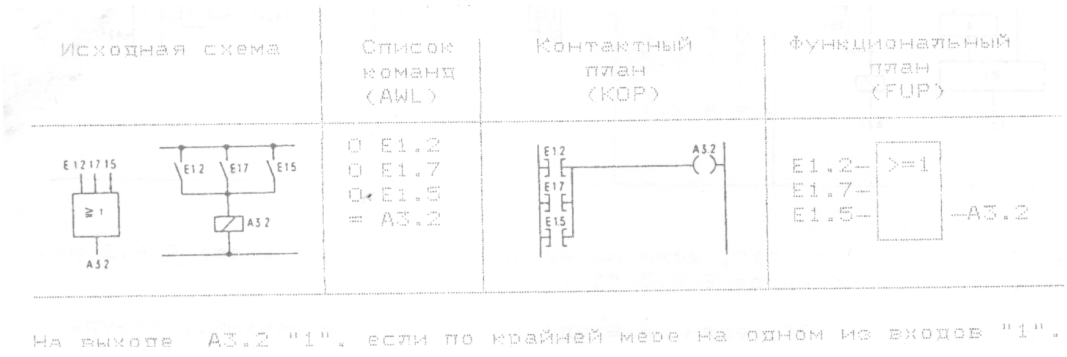


На выходе А3.5 индицируется «1», если на всех входах одновременно присутствует «1».

На выходе А3.5 «0», если хотя бы на одном из входов «0».

Количество опросов и последовательность программирования являются произвольными.

Операция «ИЛИ"

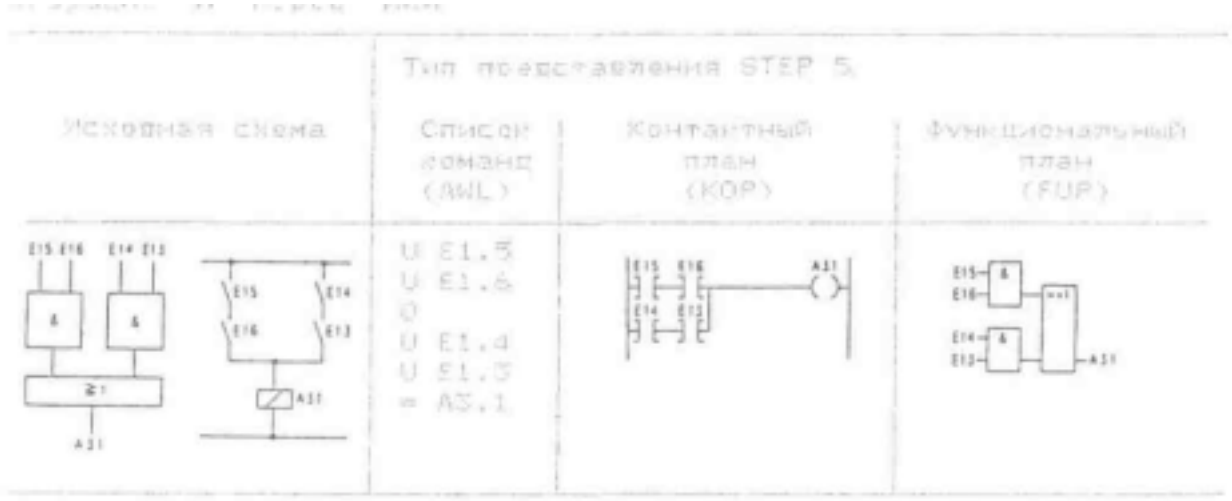


На выходе А3.2 «1», если, по крайней мере на одном из выходов «1».

На выходе А3.2 «0», если на всех входах одновременно «0».

Количество опросов и последовательность программирования произвольны.

Операция "И" перед "ИЛИ"



На выходе А3.1 «1», если по крайней мере одна из операций «И» выполнена (имеет на выходе «1»).

На выходе А3.1 «0», если ни одна из операций «И» не выполнена.

Операция «ИЛИ»- «И»- «ИЛИ»



На выходе А2.1 «1», если на выходе Е6.0 или на выходе Е6.2 и, одновременно, на выходе Е6.3 «1».

На выходе А2.1 «0», если на выходе Е6.0 и, одновременно, результат операции «И» «0».

Операция "ИЛИ" перед "И"



На выходе A3.0 "1", если обе операции "ИЛИ" одновременно выполнены.

На выходе A3.1 "0", если по крайней мере одна из операций "ИЛИ" не выполнена.

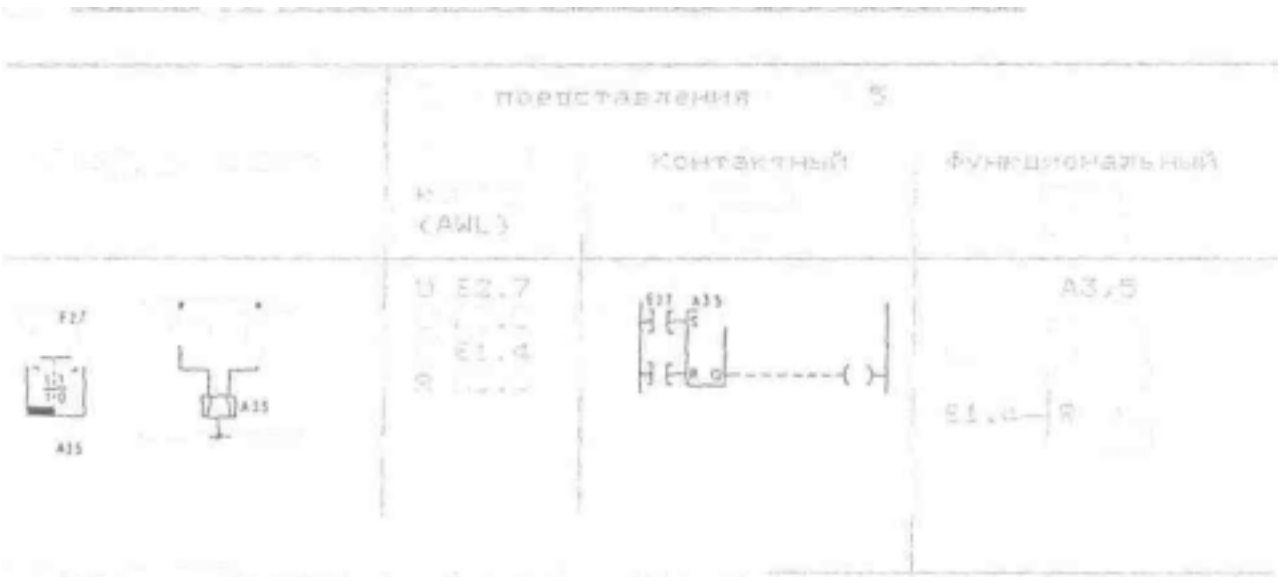
Опрос на "0"состояние сигнала



На выходе A3.0 "1" только в том случае, если на выходе E1,5 "1" (нормально разомкнутый контакт активирован), а на входе E1.6 "0" (нормально замкнутый контакт не активирован).

o Операции с памятью

RS-триггер (элемент памяти) для запоминания сигналов



Состояние сигнала "1" на выходе E2.7 вызывает установку элемента памяти ("1" на выходе A3.5). Если состояние сигнала на выходе E2.7 изменится на "0", то это не повлияет на состояние RS-триггера, т.е. запоминается предыдущее состояние.

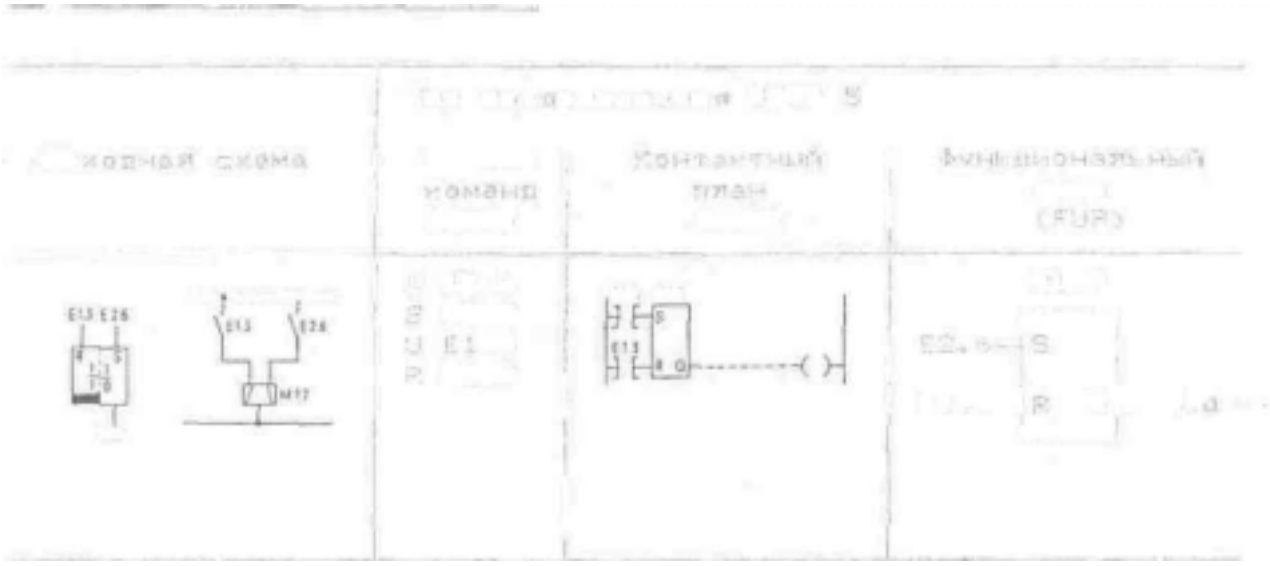
Появление "1" на выходе E1.4 приводит к сбросу состояние RS-триггера ("0" на выходе A3.5).

Если состояние сигнала на выходе E1.4 изменится на "0", то повлияет на состояние триггера, т.е. запоминается предыдущее состояние.

При одновременном поступлении "1" на оба выхода- установки (E2.7) и сброса (E1.4) приоритет у последнего опроса, т.е. в данном случае у сброса (E1.4)

o Операция с памятью (продолжение)

RS- элемент памяти с маркером



Состояние сигнала "1" на выходе E 2.6 вызывает установку элемента памяти. Если состояние сигнала на выходе E 2.6 изменится на "0", то это не повлияет на состояние RS-_триггера, т.е. запоминается предыдущее состояние.

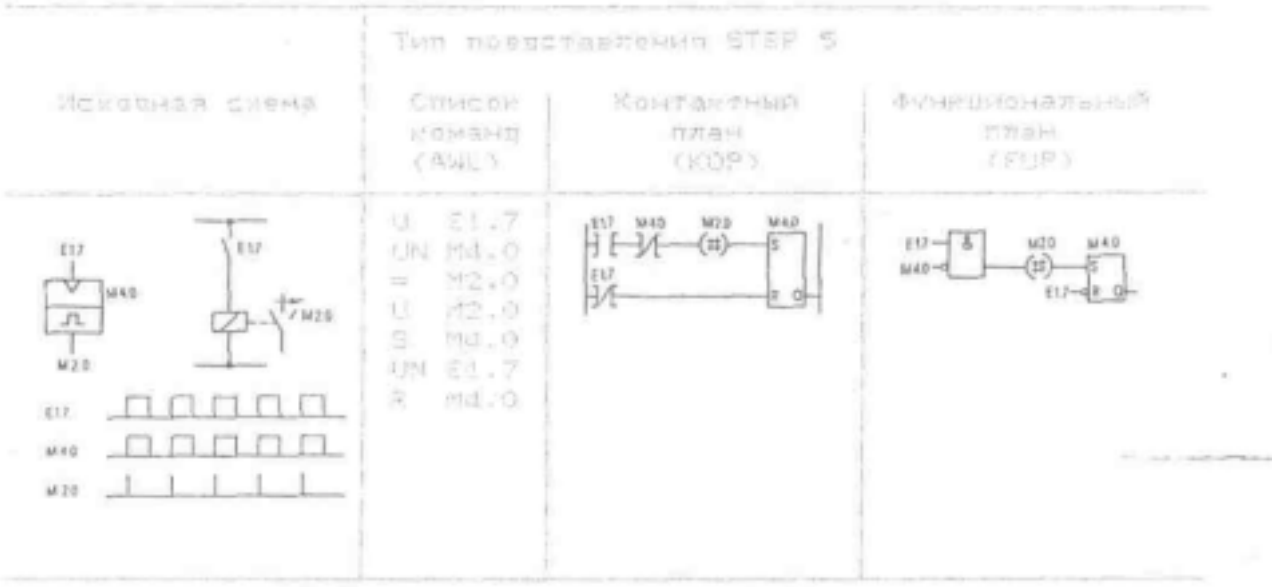
Появление "1" на выходе E 1.3 приводит к сбросу RS-_триггера.

Если состояние сигнала на выходе E 1.3 изменится на "0", то это не повлияет на состояние триггера, т.е. запоминается предыдущее состояние.

Если сигнал установки (выход E 2.6) и сигнал сброса (E1.3) появляются одновременно, имеет приоритет операция, запрограммированная последней (в данном случае U E 1.3)- сброс имеет приоритет.

o Операции с памятью (продолжение)

Эквивалентная система импульсивного контакта (анализ положительного фронта)



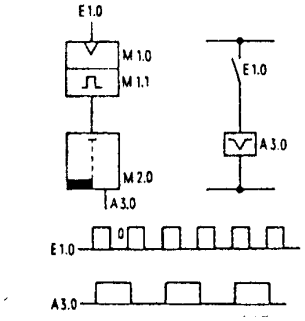
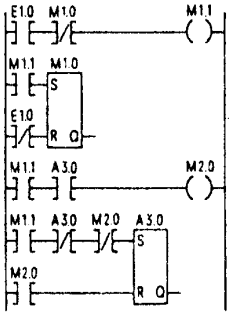
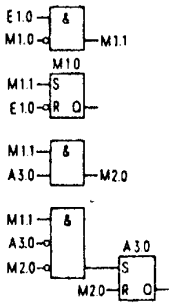
При любом положительном фронте на выходе E1.7 выполняется операция «И» (U E1.7 и UN M4.0) и результатом логической операции VKE=1 устанавливаются маркеры M4.0 *маркер фронта) и M2.0

При следующем цикле обработки программы операция «И» (U E1.7 и UN M4.0) не выполняется, т.к. маркер M4.0 установлен.

Маркер M2.0 только во время одного единственного цикла программы имеет состояние “1”.

Делитель частоты на 2 (Т—триггер)

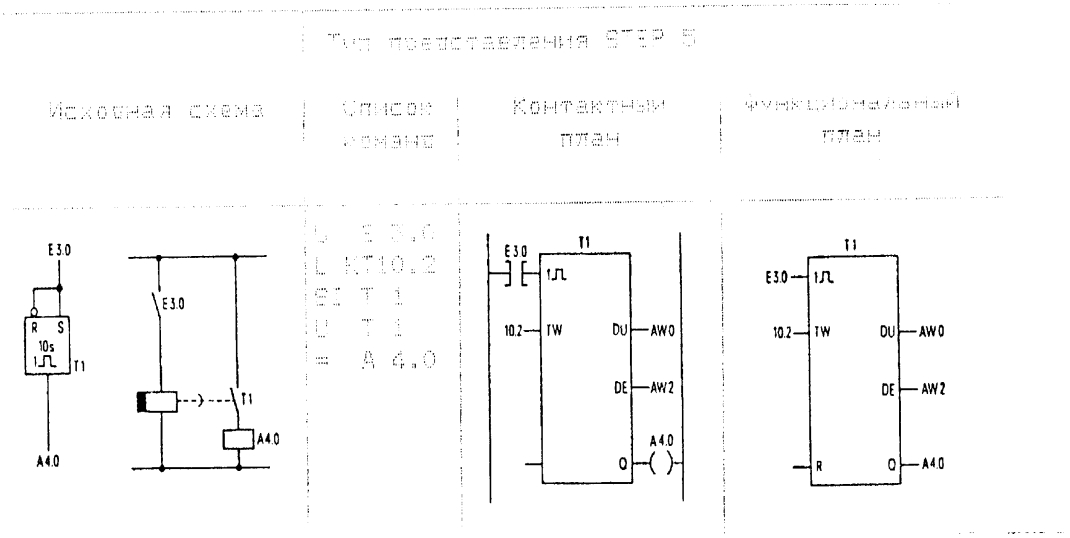
Тип представления STEP 5

Исходная схема	Список команд (AWL)	Контактный план (KOP)	Функционально-блочный план (FUP)
	U E1.0 UN M1.0 = M1.1 U M1.1 S M1.0 UN E1.0 R M1.0 U M1.1 U A3.0 = M2.0 U M1.1 UN A3.0 UN M2.0 S A3.0 U M2.0		

Делитель частоты на 2 (выход А3.0) меняет свое состояния при каждом положительном фронте сигнала на входе Е1.0. Поэтому на выходе элемента память частота сигнала в два раза меньше, чем на входе.

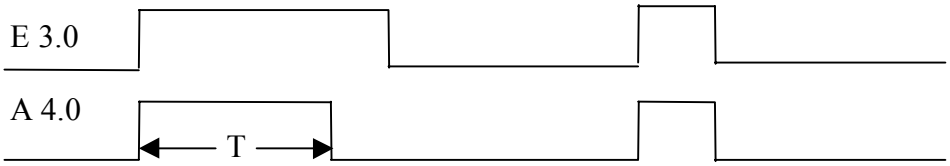
о Функции таймера

Импульс



таймер запускается во время первого же цикла обработки программы если результат логической операции (VKE)=1. Таймер не меняет своего состояния, пока VKE=1, и не истекло запрограммированное время.

Таймер сбрасывается в «0», когда VKE становится равным «0» или истекло время, на которое таймер запрограммирован.



Значение КТ 10.2:

Таймер загружается заданным значением (10)

Число справа от точки указывает на масштаб времени:

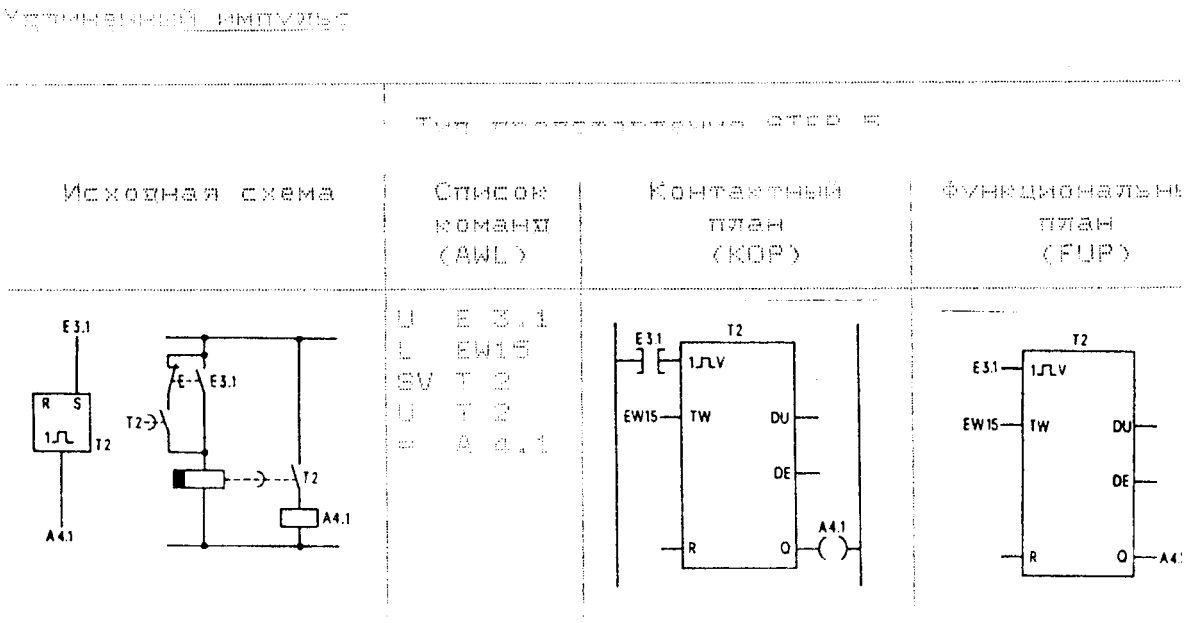
0=0,01 с 2=1 с

1=0,1 с 3=10 с

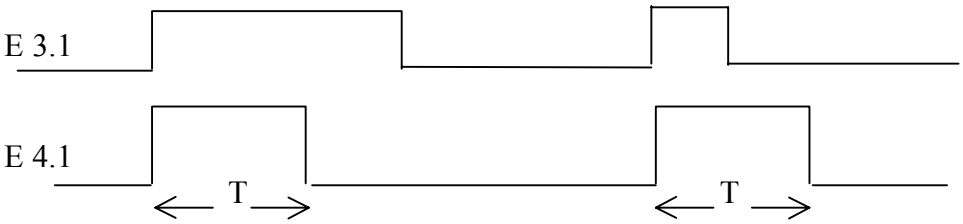
выходы DU и DE –цифровые выходы ячейки таймера. С выхода DU выдается состояние таймера в двоичном виде, а DE- в двоично-десятичном виде с базой времени.

о Функции таймера (продолжение)

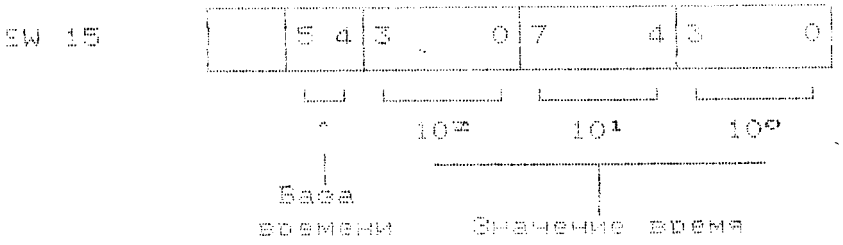
Удлиненный импульс



Таймер запускается во время первого же цикла обработки программы если результат логической операции (VKE)=1.
Таймер не меняет своего состояния, пока не истекло запрограммированное время даже если VKE стал равен "0".
Таймер сбрасывается в "0" по истечении времени, на которое он запрограммирован.

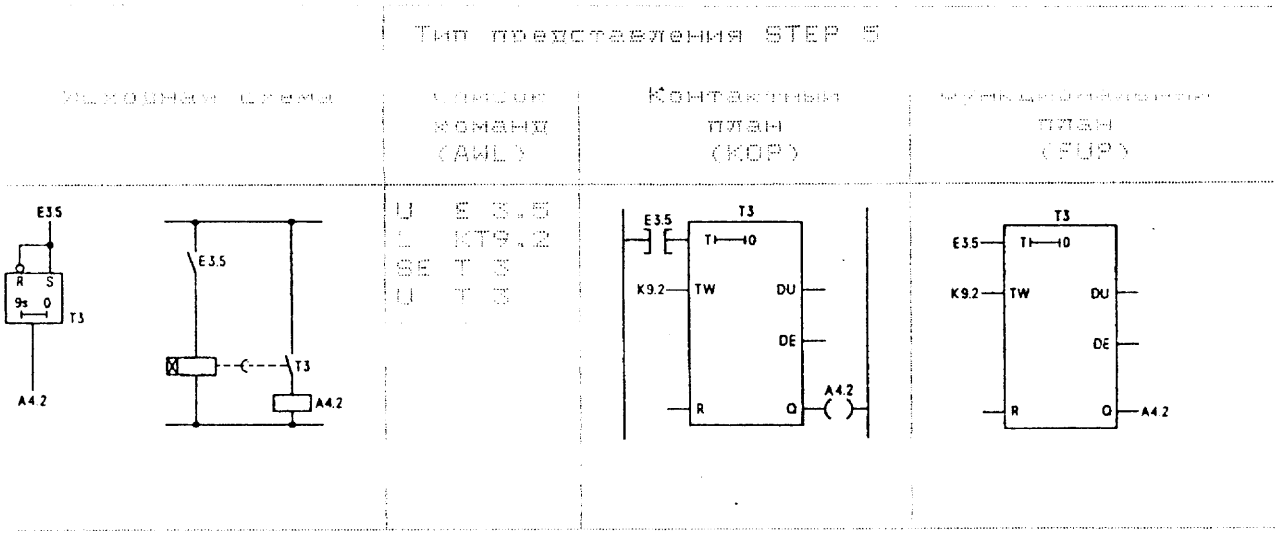


Установка значения, величины времени с помощью заданного в двоично-десятичном коде (BCD) значения операндов E, A, M или D (в примере - слово входов 15: EW15)

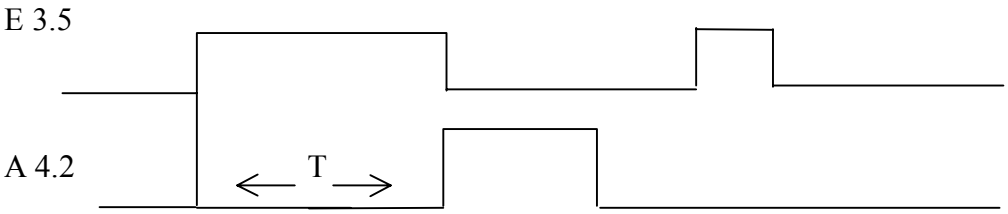


о Функции таймера (продолжение)

Задержка включения



Таймер запускается во время первого же цикла обработки программы, если результат логической операции (VKE)=1. Но он не устанавливается в "1" до тех пор, пока не истечет время, на которое запрограммирована задержка включения.
Таймер сбрасывается в "0", когда VKE устанавливается в "0".



KT 9.2

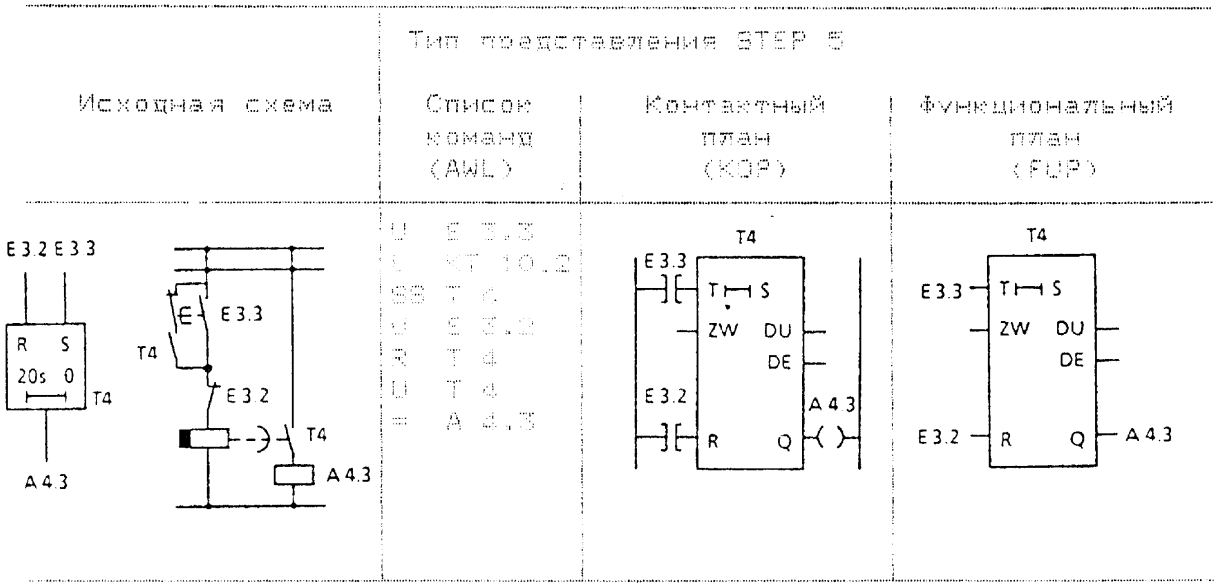
Таймер загружается заданным значением (9 сек.). Число справа от точки указывает на базу времени:

0 = 0,01сек. 2 = 1сек.

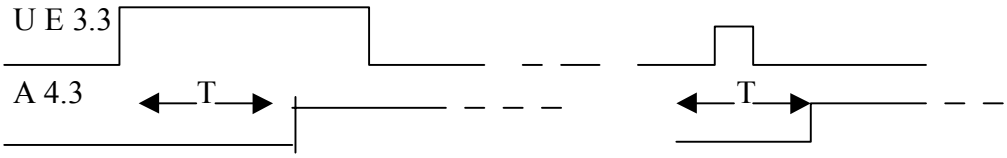
1 = 0,1сек. 3 = 10сек.

о Функции таймера продолжение

<u>задержка включения с запоминанием</u>

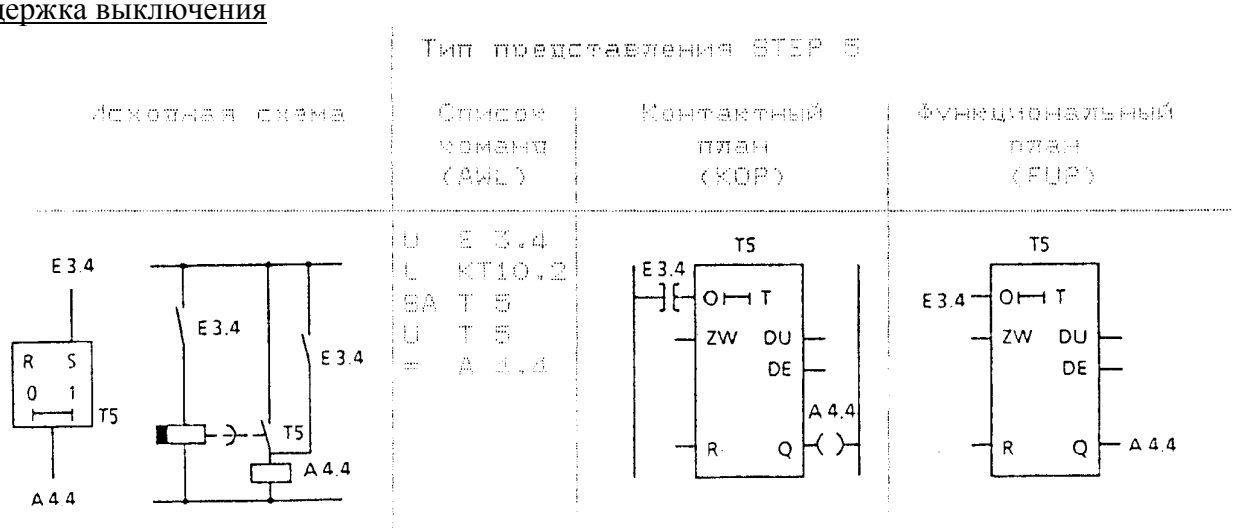


По положительному фронту сигнала на входе E3.3 начинается отсчет времени, на которое запрограммирована задержка включения.
По истечении этого времени таймер устанавливается в «1».
Таймер сбрасывается в «0» только функцией R T.

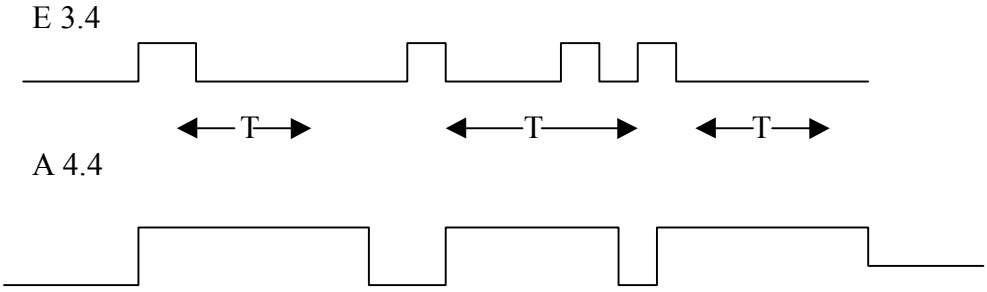


o Функции таймера (продолжение)

Задержка выключения

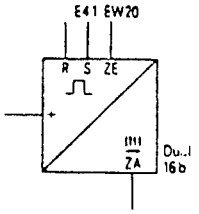
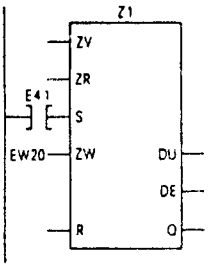
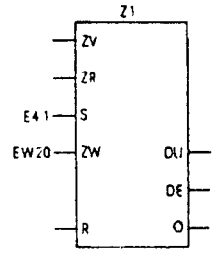


Таймер устанавливается в «1» во время первого же цикла обработки программы положительным фронтом сигнала на входе E 3.4.
По отрицательному фронту сигнала на входе E 3.4 начинается отсчет времени, на которое запрограммирована задержка включения.
До истечения этого времени таймер сбрасывается в «0».



о Операции со счетчиками

Установка счетчика

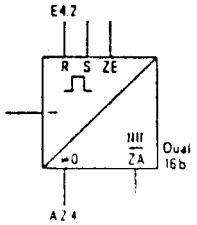
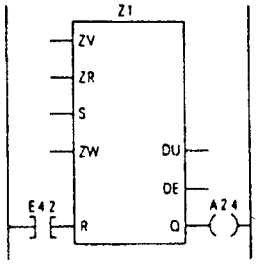
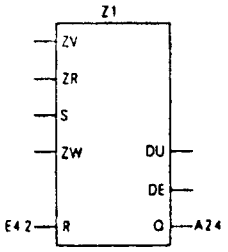
Исходная схема	Тип представления STEP 5		
	Список команд (AWL)	Контактный план (KOP)	Функциональный план (FUP)
	U E 4.1 L EW20 S Z 1		

Счетчик устанавливается по входу S во время первого же цикла обработки программы и в течение этого цикла не меняет своего состояния независимо от VKE. Счетчик перезапускается в начале следующего цикла, если к его началу VKE=1.

Указание для работы с положительным фронтом содержится в слове счетчика, которым может быть слово входов EW, слово выходов AW, слово маркеров MW или слово данных DW. В слове счетчика задано также значение счета. Слово счетчика задается в BCD—коде и имеет формат 16 бит.

Выходы DU и DE - цифровые выходы ячейки счетчика. С выхода DU выдается состояние счетчика в двоичном виде. а DE - в двоично-десятичном виде.

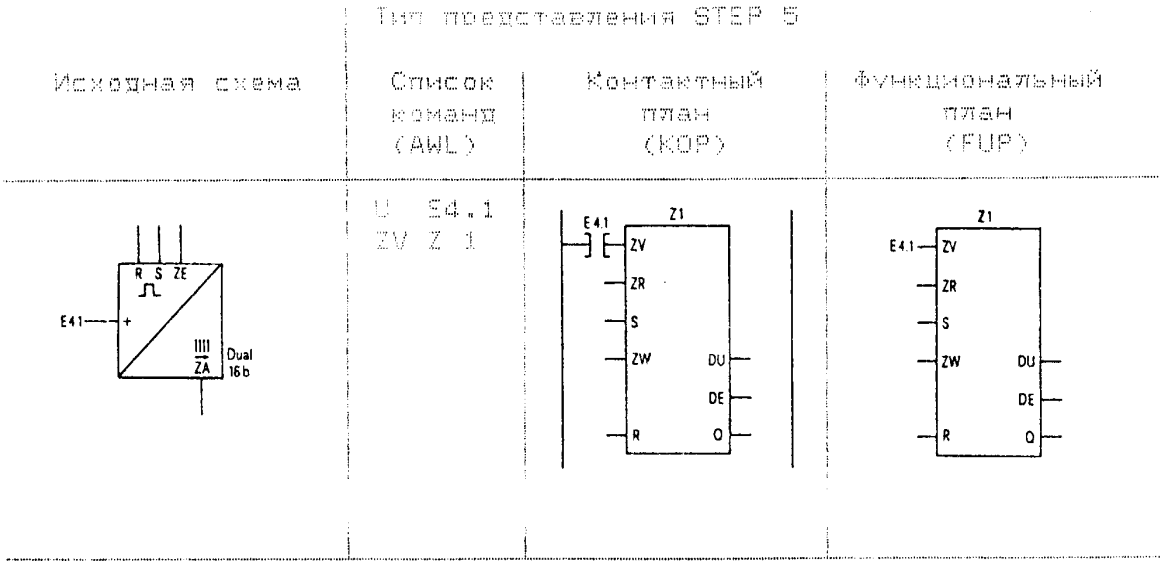
Сброс счетчика

Исходная схема	Тип представления STEP 5		
	Список команд (AWL)	Контактный план (KOP)	Функциональный план (FUP)
	U E 4.2 R Z 1 U Z 1 = A 2.4		

Счетчик сбрасывается по входу R, если VKE = 1 . Счетчик после этого остается без изменений, даже если в результате логической операции становится «0» на входе R.

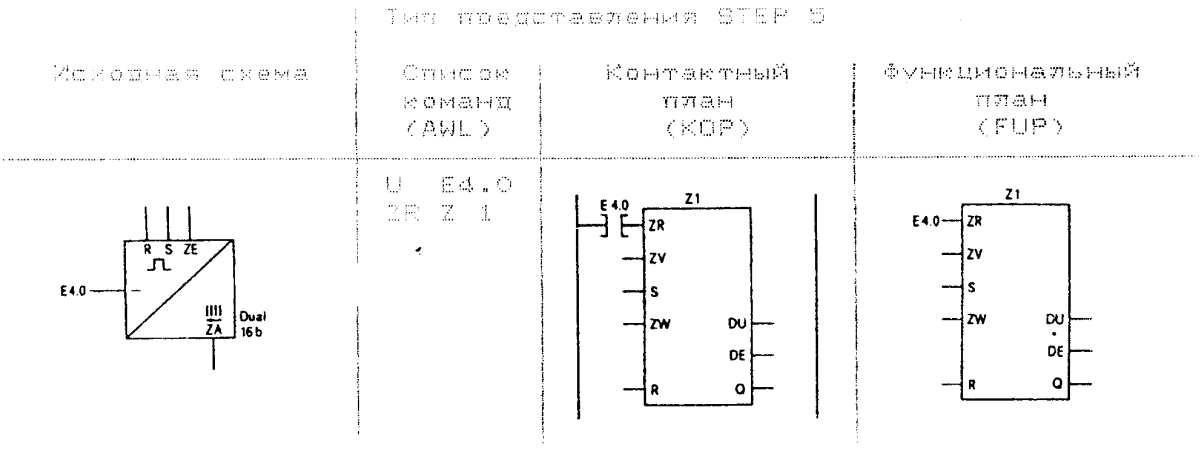
o Операции со счетчиками (продолжение)

Прямой счет



Значение счетчика увеличивается на 1 при поступлении на счетный вход ZV положительного фронта сигнала от операнда запрограммированного перед оператором ZV. Необходимые для работы с положительными фронтами указания содержатся в слове счетчика. Используя два различных маркера для работы со счетными входами ZV и ZR. Счетчик можно использовать для прямого и обратного счета.

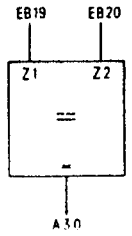
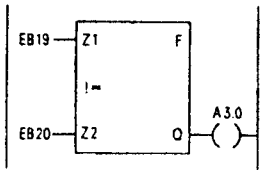
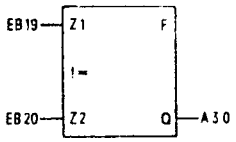
Обратный счет



Значение счетчика уменьшается на 1 при поступлении на счетный вход ZP положительного фронта сигнала от операнда, запрограммированного перед оператором ZR. Необходимые для работы с положительными фронтами метки содержатся в слове счетчика. Используя в различных маркерах для работы со счетными входами ZV и ZR. счетчик можно использовать для прямого и обратного счета.

o Функции сравнения

Сравнение на равенство

Исходная схема	Тип представления STEP 5		
	Список команд (AWL)	Контактный план (KOP)	Функциональный план (FUP)
	L EB19 L EB20 != F = A3.0		

Операнд, заданный первым сравнивается с последующим операндом в соответствии с функцией сравнения (Z1=Z2). По результату сравнения выставляется VKE:
VKE = 1, если операнды равны (содержимое EB19 равно содержимому EB20)

АКК 1-L= АКК 2 - L

VKE = 0, если операнды не равны.

АКК 1 - L = АКК 2 -L.

Коды условий ANZ 1 и ANZ 0 устанавливаются, как описано в разделе 6.1

АКК 2 - L
АКК 1 – L

0	EB 19

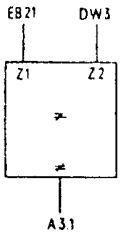
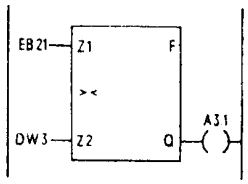
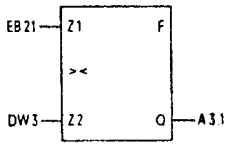
0	EB 20

АКК 2 - H и АКК 1 - H остаются без изменений во время сравнения чисел с фиксированной запятой,
При сравнении чисел с фиксированной запятой (!=F) и чисел с плавающей запятой (!=G) сравниваются полное содержимое аккумуляторов АКК 1и АКК 2 (32 бита)
При сравнении учитывается численное представление операндов, т.е. содержимое аккумуляторов АКК 1 - L и АКК 2 - L интерпретируется как числа с фиксированной запятой.

o Функции сравнения (продолжение).

Сравнение на неравенство

Тип представления STEP 5

Исходная схема	Список команд (AWL)	Контактный план (KOP)	Функциональный план (FUP)
	L EB21 L DW3 >< F = A3.1		

Операнд заданный первым, сравнивается с последующим операндом в соответствии с функцией сравнения (Z1xZ2). По результату сравнения выставляется VKE:
VKE =1, если операнды не равны (содержимое EB 21 не равно содержимому EB 22).
| АКК 1- L= АКК 2 - L.
VKE =0. если операнды равны.
АКК 1 – L = АКК 2 - L
Коды условий ANZ 1 и ANZ 0 устанавливаются как описано в разделе 6.1

0	EB 21	АКК 2 – L
DW	3	АКК 1 - L

АКК 2 - Н и АКК 1 " Н остаются без изменений во время 16-битового сравнения чисел с фиксированной запятой.

При сравнении 32-битовых чисел с фиксированной запятой сравнивается полное содержимое аккумуляторов АКК 1 и АКК 2 (включая АКК 1-Н и АКК 2 –L)
Это применимо и к сравнению на больше, больше или равно, меньше. меньше или равно (смотри список команд).
При сравнении учитывается численное представление операндов т.е. содержимое аккумуляторов АКК 1 - L и АКК 2 интерпретируется как числа с фиксированной запятой.

3.2.2. Дополнительный набор операций

Дополнительный набор операций может быть использован только в функциональных блоках (FB и FX). Общий набор операций для функциональных блоков, таким образом. состоит из набора основных и дополнительных операций.

К дополнительным операциям принадлежат системные функции: С помощью системных функций можно вмешиваться в работу операционной системы, память может быть перезаписана в любой момент. и может быть изменено содержимое рабочих регистров центрального процессора. Поэтому системные функции следует использовать, если это необходимо, с большой осторожностью. Программировать системные функции Вы сможете только сделав соответствующую установку в маске предустановок на программаторе.

По теме «Системные операции» обратитесь к Разделу 9 « обращение к памяти».

Программа функциональных блоков может быть представлена только в виде AWL. Поэтому они не могут быть запрограммированы в графическом виде представления (КОР и FUP).

Следующее описание показывает дополнительные операции и системные функции, которые могут быть использованы только в функциональных блоках. Кроме того, показана возможная комбинация операций дополнительного набора с фактическими операндами.

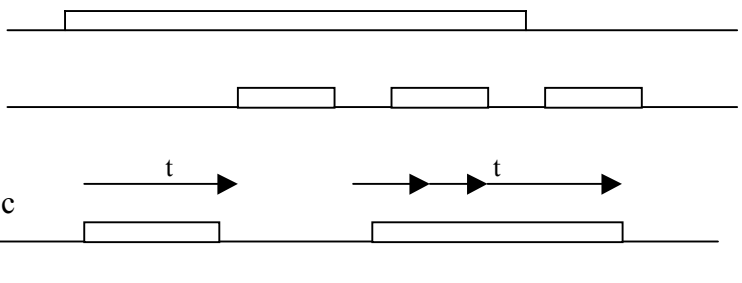
o Двоичная логика

Операция	Описание
U = ———	Функция "И" опрос формального операнда на состояние "1".
UN = ———	Функция "И" опрос формального операнда на состояние "0".
O = ———	Функция "ИЛИ" опрос формального операнда на состояние "1
ON = ———	Функция "ИЛИ"\ опрос формального операнда на состояние "0"
	Вставить формальный операнд
	В качестве фактического операнда допустимы двоично-адресованные входы/выходы и маркеры (параметры вида E.A: тип данных BI), таймеры и счетчики (вид параметра: T, Z).

o Функции запоминания

Операция	Описание
S = _____	Установка (двоичная) формального операнда
RB = _____	Сброс (двоичный) формального операнда
= = _____	Вывод результата соединения по формальному операнду
	Вставить формальный операнд
	В качестве фактического операнда допустимы двоично-адресованные входы, выходы, данные и маркеры (вид параметра E, A : тип параметра: BI)

о Функции таймеров и счетчиков

Операция	Описание
T 0. . .255	Деблокировка таймера для запуска Операция выполняется только по переднему фронту результата логической операции VKE. Таймер перезапускается, если VKE = «1» во время операции запуска VKE при SI T VKE при FT Опрос по U T 
FR Z 0. . .255	Деблокировка счетчика Операция выполняется только по переднему фронту результата логической операции VKE Счетчик устанавливается (счет «вперед» или «назад»). Если VKE=1 В соответствующей операции.
FR =_____	Деблокировка формального операнда для нового пуска (см. «Описание FR T или FR Z» в зависимости от формального операнда, вид таймера: T, Z).
RD = ----	Сброс (цифровой) формального операнда (вид параметра: T,Z)
SI = -----	Запуск таймера, заданного формальным операндом, в качестве импульса с длительностью, закодированной в АКК (вид параметра: T)
SE = ----	Запуск таймера, заданного формальным операндом, в качестве задержки включения с длительностью, закодированной в АКК (вид параметра: T)
SVZ = ----	Запуск таймера, заданного формальным операндом, в качестве удлиненного импульса с длительностью, закодированной в АКК или установка счетчика, заданного формальным операндом, с указанным впоследствии счетным значением (вид параметра: T,Z)
SSV = ----	Запуск таймера, заданного формальным операндом, в качестве запоминающей задержки включения со значением, закодированным в АКК, или счет впереди счетчика, заданного формальным операндом (вид параметра: T,Z) Установка значения таймера или счетчика в виде значения в коде BCD с помощью операндов AW, MW, DW, указанных в качестве формальных операндов (вид параметра: E, тип параметра: W), или с помощью константы (вид параметра: D, тип параметра: KT, KZ)

о Функции таймеров и счетчиков (продолжение)

Операция	Описание
SAR = ---- └	Запуск таймера, заданного формальным операндом, в качестве задержки включения с длительностью, закодированной в АКК или счет назад счетчика, заданного формальным операндом (вид параметра: T, Z) Вставить формальный операнд Таймеры и счетчики можно использовать в качестве актуальных операндов. Исключения SI и SE (только таймеры). Значения таймера или счетчика может быть установлено как с помощью основного набора операций, так и с помощью формальных операндов следующим образом: Установка значения таймера или счетчика в виде значения в коде BCD с помощью операндов EW, AW, MW, DW, указанных в качестве формальных операндов (вид параметра: E. тип параметра: W), или с помощью константы (вид параметра: D. тип параметра: KT, KZ)

Примеры:

Вызов функционального блока	Программа в функциональном блоке	Выполняемая программа
: SPA FB203 ИМЯ : BEISPIEL ANNA : E 10.3 BERT : T 17 HANS : A 18.4	: U = ANNA : L = KT 010.2 : SSV = BERT : U =BERT : = = HANS	U E 10.3 L KT 010.2 SS T 17 U T 17 = A 18.4
: SPA FB204 ИМЯ : BEISPIEL MAXI : E 10.5 IRMA : E 10.6 EVA : E 10.7 DORA : Z 15 EMMA : M 58.3	: U = MAXI : SSV = DORA : U = IRMA : SAR = DORA : U = EVA : L KZ 100 : SVZ = DORA : UN = DORA : = =EMMA	U E 10.5 ZV Z 15 U E 10.6 ZR Z 15 U E 10.7 L KZ 100 S Z 15 UN Z 15 = M 58.3
: SPA FB205 ИМЯ : BEISPIEL KURT : E 10.4 CARL : T 18 EGON : EW20 MAUS : M 100.7	: U = KURT : L = EGON : SVZ = CARL : U = CARL : = = MAUS	U E 10.4 L EW20 SV T 18 U T 18 = M 100.7

o Операции загрузки и пересылки

Операция	Описание
L =_____	Загрузка формального операнда Значение, заданное в качестве формального операнда, загружается в АКК (вид параметра: E, A, T, Z; тип параметра: BY, W, D).
LC =_____	Кодированная загрузка формального операнда Значение временной или счетной ячейки в качестве формального операнда загружается в АКК в BCD-коде (вид параметра: T, Z)
LW =_____	Загрузка набора битов формального операнда Формальный операнд в битовом виде загружается в АКК (вид параметра: D; тип параметра: KF, KH, KM, KY, KC, KT, KZ)
LD =_____	Загрузка набора битов формального операнда Формальный операнд в битовом виде загружается в АКК (вид параметра: D; тип параметра: KG)
T =_____ Вставить формальный операнд	Пересылка в формальный операнд Содержимое АКК пересылается в формальный операнд (вид параметра: A, E; тип параметра: W, D.)

В качестве фактических операндов допускаются соответствующие операнды основных операций. В операции LW допускается использование данных в форме двоичного(KM) или шестнадцатеричного (KH) набора битов, 2 байтового числа (KY), символов (KC), числа с фиксированной запятой (KF), значения таймера (KT) и значения счетчика (KZ). В операции LD в качестве данных допускаются числа с плавающей запятой.

Операция	Параметры	Описание
L BA	0…255	Загрузка слова в АКК 1 из области «интерфейса данных» (область BA)
L BB	0…255	Загрузка слова в АКК 1 из области «интерфейса данных» (область BB)
L BS	0…255	Загрузка слова в АКК 1 из области «системных данных» (область BS)(BS 60…63 свободны)
L BT	0…255	Загрузка слова в АКК 1 из области «системных данных» (область BT)
L BA	0…255	Загрузка слова в АКК 1 в область «интерфейса данных» (область BA)
L BB	0…255	Загрузка слова в АКК 1 в область «интерфейса данных» (область BB)
L BS [1)]	0…255	Загрузка слова в АКК 1 в область «системных данных» (область BS) (BS 60…63 свободны)
L BT [1)]	0…255	Загрузка слова в АКК 1 из области «системных данных» (область BT)

[1)] Системная операция

В противоположность областям BA, BB и BT область BS позволяет свободно использовать для целей пользователя только слова BS60…BS63. Обратите внимание на Раздел 8.2.4 «Область BS и BT»

o Арифметические операции

Операция	Описание
ENT	Ввод данных в память арифметики Результат команды ENT загружается в АКК 3 и АКК 4, которые в арифметических операциях используются тоже. АКК 4: = АКК 3 АКК 1: = АКК 1 АКК 3: = АКК 2 АКК 2: = АКК 2

Происходит STACK LIFT в аккумуляторах АКК 3 и АКК 4
<АКК 4> : = <АКК 3>
<АКК 3> : = <АКК 2>
<АКК 2> : = <АКК 2>
<АКК 1> : = <АКК 1>
Содержимое аккумуляторов АКК 1 и АКК 2 не меняется. Старое содержимое АКК 4 теряется.
Пример:
Необходимо вычислить выражение: (30 + 3 x 4)/6= 7

	АКК 1	АКК 2	АКК 3	АКК 4
Содержимое аккумулятора до выполнения арифметической операции	a	b	c	d
L KF 30	30	a	c	d
L KF 3	3	30	c	d
ENT	3	30	30	c
L KF 4	4	3	c	d
* F	12	30	c	c
+ F	42	c	c	c
L KF 6	6	42	c	c
/ F	7	c	c	c

Операция	Параметры	Описание
ADD BF [1)]	-127 . . . + 127	Прибавление байта константы (с фиксированной запятой) к содержимому АКК1
ADD KF [1)]	-3276 . . . +32767	Прибавление слова константы (с фиксированной запятой) к содержимому АКК1

[1)] Системная операция

Операция	Параметры	Описание
ADD DF [1) 2)]	-2147483648 … +2147482347	Прибавление константы с фиксированной запятой (двойное слово) к АКК 1.
—D [1) 2)]		Сложение двух двухсловных констант с фиксированной запятой (АКК 1 + АКК 2)
—D [1) 2)]		Вычитание двух двухсловных констант с фиксированной запятой (АКК 1 + АКК 2)
TAK [1)]		Обмен содержимым аккумуляторов АКК 1 АКК 2

1) Системная операция

2) Программирование возможно в зависимости от типа PG и версии программного обеспечения PG

о Цифровые логические операции

Операция	Описание
UW	Операция цифрового логического «И» над содержимым АКК 1 и АКК 2
OW	Операция цифрового логического «ИЛИ» над содержимым АКК 1 и АКК 2
XOW	Операция цифрового логического «ИСКЛ. ИЛИ» над содержимым АКК 1 и АКК 2

Аккумуляторы 3 и 4 не используются, а используются биты индикации ANZ1 и ANZ0 (смотри слово индикации результатов).

При использовании двух операций загрузки аккумуляторы могут быть загружены соответствующими операндами. Затем содержимое обоих аккумуляторов может быть использовано для цифровых логических операций.

Пример:

	AKK 1 – L	AKK 2 – L
L EW 1	EW 1	[AKK 1 – L]

Предыдущее содержимое аккумулятора АКК1-L

L EW 2	EW 1	EW 2

Операция логического цифрового «И» над EW 2 и EW 1

UW	Результат	EW 1

Организационные функции

о Функции перехода

Назначение условного или безусловного перехода указывается символически (максимально 4 знака, начинающиеся с буквы). Символический параметр команды перехода идентичен символическому адресу команды, к которой осуществляется переход. При программировании следует принимать во внимание , что абсолютное состояние перехода не может превышать 127 слов и что команда языка STEP 5 может состоять из более чем одного слова. Переход может выполняться только внутри одного блока; переход между сегментами недопустим.

Важно!
Команда перехода и назначение должны быть в одном сегменте. Для сегмента в качестве назначения можно указывать только один символический адрес.
Примечание: Эти условия не применимы для команды перехода SPR, для которой в качестве параметра указывается абсолютный адрес расстояния перехода.

Операция	Описание
SPA = адр	Безусловный переход Безусловный переход выполняется при всех условиях
SPB = адр	Переход по условию Условный переход выполняется, если VKE = 1. Если VKE = 1, команда не выполняется, и результат логической операции устанавливается в VKE = 1
SPZ = адр	Переход по условию: ANZ 1, ANZ 0 Переход выполняется, если только ANZ 1=0 и ANZ 0=0. Результат логической операции не изменяется.
SPN = адр	Переход по условию: ANZ 1, ANZ 0 Переход выполняется, если ANZ 1 == ANZ 0. Результат логической операции не меняется.
SPP = адр	Переход по условию: ANZ 1, ANZ 0 Переход выполняется, если только ANZ 1=1и ANZ 0=0. Результат логической операции не изменяется.
SPM = адр	Переход по условию: ANZ 1, ANZ 0 Переход выполняется, если только ANZ 1=0 и ANZ 0=1. Результат логической операции не изменяется.

адр = символический адрес (максимально 4 символа)

Операция	Описание
SPO = адр	Переход по условию переполнения (overflow) Переход выполняется, если только условие Переполнения OV=1. Если переполнения нет, (OV=0), переход не выполняется. Результат логической операции не изменяется. Если происходит превышение допустимых границ Результирующего числа при выполнении арифметических операций, происходит переполнение.
SPS = адр	Переход, если установлен код условия OS (запоминание переполнения) (OS=1).
SPR [1]) -32768…+32767	Любой переход внутри функционального блока: Выполняется всегда независимо от условий.

адр=символический адрес (максимально 4 символа)
[1]) Системная операция

о Функции сдвига

Операция	Описание
SLW 0 . . . 15	Сдвиг влево (справа заполнение нулями)
SRW 0 . . . 15	Сдвиг вправо (слева заполнение нулями)
SLD 0 . . . 32	Сдвиг двойного слова влево (справа заполнение нулями)
SVW 0 . . . 15	Сдвиг вправо со знаком
SVD 0 . . . 32	Сдвиг двойного слова вправо со знаком (знак устанавливается слева)
RLD 0 . . . 32	Циклический сдвиг влево
RRD 0 . . . 32	Циклический сдвиг вправо

При операциях сдвига используется только аккумулятор АКК 1.
Параметр команды определяет на сколько позиций производится сдвиг содержимого аккумулятора. В операциях SLW, SRW, SVW используется только младшее слово аккумулятора, в операциях SLD, SVD, RLD используется полное содержимое аккумулятора АКК 1 (32-бита).
Функции сдвига выполняются всегда независимо от условий.

Если последний сдвинутый бит равен нулю, используется команда перехода SPZ: По команде SPZ может происходить переход, если последний сдвинутый бит равен 0, а по команде SPN – если бит равен 1.

Влияние на биты индикаций ANZ 0 и ANZ 1:

ANZ 1	ANZ 0	Сдвиг : последний сдвинутый бит
0	0	0
0	1	-
1	0	1

Пример:

Программа STEP 5	Содержимое слова данных
: L DW 52	H = 14 AF
: SLW 4	H = 4 AF0

Программа STEP 5	Содержимое АКК 1 (16-ричн.)
: L ED0	2348 ABCD
: SLW 4	2348 BCD0
: SRW 4	2348 OBCD
: SLD 4	3480 BCD0
: SVW 4	3480 FBCD
: SVD 4	0348 OFBC
: RLD 4	3480 FBC0
: RRD 4	0348 OFBC
: BE	

Применение : Умножение с двойной точностью
например новое значение = старое значение x 8

```
: L   MW 10
: SLW 3
: T   MW 10      Внимание! Положительная граница
                 области не может быть превышена!
```

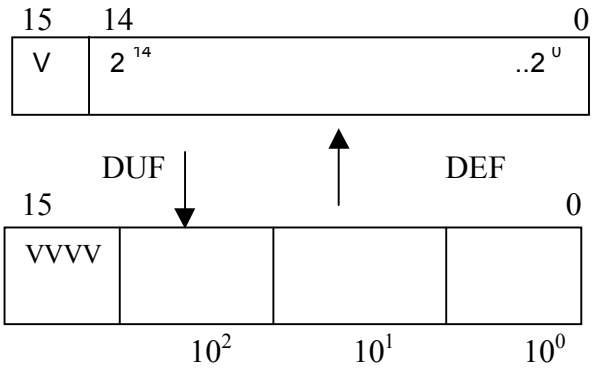
деление с двойной точностью
например новое значение = старое значение: 4

```
: A   DB 5
: L   DV0
: SRW 2
: T   DW0
```

о Операции преобразования

Операция	Описание
KEW	Формирование первого дополнения АКК 1(16 бит)
KZW	Формирование двоичного дополнения АКК 1(16 бит) тора 1
KZD	Формирование двоичного дополнения АКК 1(32 бит) тора 1
DEF	Преобразование числа с фиксированной запятой (16 бит) из кода BCD в двоичный вид
DUF	Преобразование числа с фиксированной запятой (16 бит) из двоичного вида в код BCD
DED	Преобразование числа с фиксированной запятой (32 бит) из кода BCD в двоичный вид
DUD	Преобразование числа с фиксированной запятой (32 бит) из двоичного вида в код BCD
FDG	Преобразование числа с фиксированной запятой (16 бит) в число с плавающей запятой
GFD	Преобразование числа с плавающей запятой (32 бит) в число с фиксированной запятой

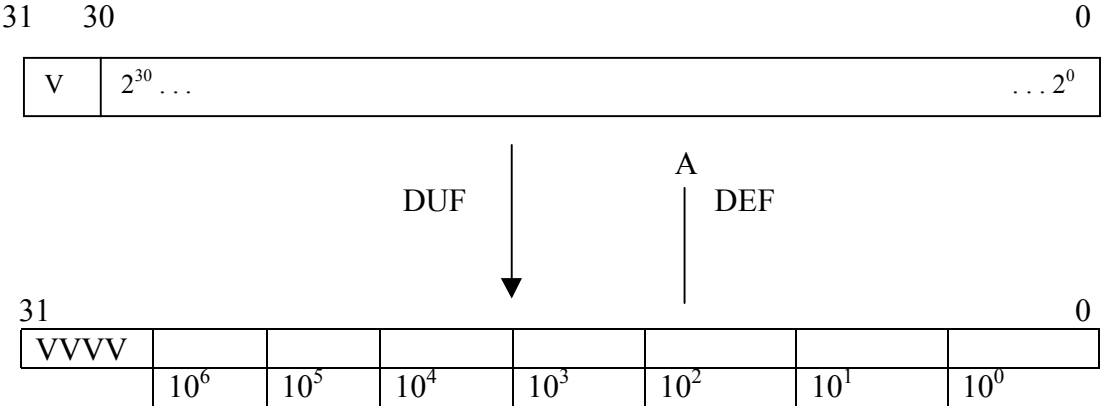
DEF:
Значение, находящееся в АКК 1- L (биты 0…15), интерпретируется как BCD-кодированное число. После преобразования в АКК 1- L находится 16-битовое число с фиксированной запятой.
DUF:
Значение, находящееся в АКК 1- L (биты 0…15), интерпретируется как 16-битовое число с фиксированной запятой. После преобразования в АКК 1- L находится BCD-кодированное число.



V (знак): 0 = положительное число
1 = отрицательное число

DED:
Значение, находящееся в АКК 1 (биты 0…31), интерпретируется как интерпретируется как BCD- кодированное число. После преобразования в АКК 1 находится 32 – битовое число с фиксированной запятой.

DUD:
Значение, находящееся в АКК 1 (биты 0…31), интерпретируется как 32 - битовое число с фиксированной запятой. После преобразования в АКК 1 находится 32 - битовое число с фиксированной запятой.



V (знак) : 0= положительное число
1= отрицательное число

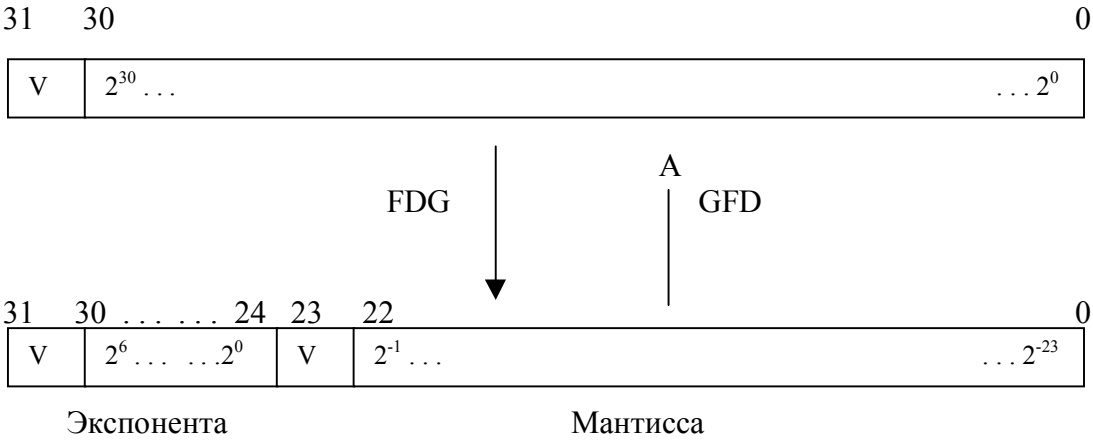
FDG:
Значение, находящееся в АКК 1 (биты 0…31), интерпретируется как BCD- кодированное число. После преобразования в АКК 1 находится число с плавающей запятой (экспонента и мантисса).

GFD:
Значение, находящееся в АКК 1 (биты 0 . . . 31),интерпретируется как 32 - битовое число с фиксированной запятой. После преобразования в АКК 1 находится 32 – битовое число с фиксированной запятой.

При этом числа с плавающей запятой >0 или < - 1, если необходимо, округляются до меньшего целого числа.

При этом числа с плавающей запятой < 0 или > - 1 округляются до 0.

Примеры: +5,7------> 5
-2.3------> -3
-0.6------> 0
+0.9------> 0



KEW, KZW:

Примеры

Содержимое слова данных 64 инвертируется и хранится в слове данных 78.

Программа STEP 5:	содержимое слова данных:
: L DW64	KM = 0011111001011011
: KEW	
: T DW78	KM = 1100001101001000

Содержимое слова данных 207 интерпретируется в виде числа с фиксированной запятой и записывается в слово данных 51 с противоположным знаком.

Программа STEP 5:	содержимое слова данных
: L DW207	KF : + 51
: KZW	
: T DW51	KF : - 51

o Декремент/Инкремент

операция	описание
D 1 . . . 255	Декремент
I 1. . . 255 Параметр	Инкремент

Содержимое АКК 1 уменьшается (декремент) или увеличивается (инкремент) на число, указанное в качестве параметра. Выполнение операции безусловное. Операция ограничивается правым байтом (без переноса).

пример

Программа STEP 5:	содержимое слова данных
: L DW7	KH: 1010
: I 16	
: T DW8	KH: 1020
: D 33	
: T DW9	KH: 10FF

о Функции обработки

операция	Описание
B DW0 . . . 255 (операция)	Обработка слова данных Следующая указанная операция будет скомбинирована с параметром, указанным в слове данных и затем выполнена.
в MW0 . . . 254 (операция)	Обработка слова маркеров Следующая указанная операция будет скомбинирована с параметром, указанным в слове данных и затем выполнена.
B = _____ _____ | формальный операнд	Обработка формального операнда (вид параметра: B): могут быть замешаны только операции A DB, SPA PB, SPA FB, SPA SB
BI [1] [2]	Обработка, использующая формальный операнд (косвенная). Номер формального операнда находится в АКК 1.
B BS [1] [2] 60 . . . 63	Выполнение команды из системной области данных (BS) (свободные системные данные: BS 60 . . .63).

[1]) системная функция

[2]) значение, которое в системных данных или в формальном операнде в функциональном блоке интерпретируется в качестве кода команды STEP 5, которая затем выполняется. Операция допустима как с командой B MW, так и с командой B DW.

Все операции, за исключением перечисленных ниже, могут комбинироваться с командой B MW или B DW.

— Все двух- и трехсловные команды (смотри приложение D)
(допустимы E DB, EX DX, SES, SEF, AX DX, BA FX и BAB FX).

— Операции с формальными операндами в функциональных блоках.

— SPA/SPB OB, SPA/SPB PB, FB, . . .

Программатор не проверяет допустимость комбинирования.

B MW 14
L EB 0 | 120 | MW 14

--------->L EB 120 (= выполненная команда)

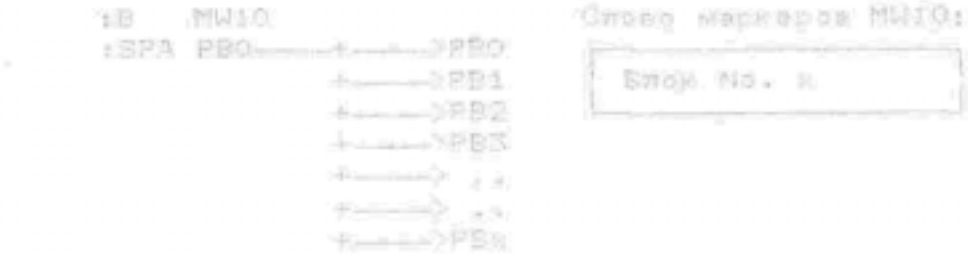
Пример (обработка слова данных)
Содержимое слов данных с DW20 по DW100 устанавливается в состоянии "О". Индексный регистр для параметров слов данных - DW1.



Примечание: Разветвитель переходов для использования подпрограмм



Применение: Разветвитель переходов для вызовов блоков



o Команды блокировки/деблокировки прерываний от процесса

AS	Блокировка прерывания от процесса
AF	Деблокировка прерывания от процесса

Функция "Блокировка/Деблокировка прерываний от процесса» может быть использована, например, если äî время выполнения программы обработки прерываний по времени необходимо подавить обработку прерываний от процесса. В той части программы, которая находится между инструкциями AS и AF обработка прерываний от процесса блокируется

Обратите внимание на специальную функцию OB 120 «Блокировка прерываний» Раздел 6.8.1

o Специальные операции

Операция	Параметр	Описание
E DB	3...255	Создание блока данных DB в DB-RAM
EX DX	1...255	Создание блока данных DX в DB-RAM

E DB: Создание блока данных

Команда E DB xxx создает блок данных с номером xxx (в пределах от 3 до 255) во внутренней памяти RAM процессоров.

Перед программированием команды необходимо загрузить в АКК 1 – L значение количества слов данных, которое должен иметь создаваемый блок. Соответствующий заголовок блока создается командой E DB/EX DX. Блок данных может зачитать в памяти максимум 4091 слов (включая заголовок блока)

Если создаваемый блок данных уже существует, слишком велика длина DB или недостаточно места в области DB-RAM, то операционная система вызывает OB 31. Если OB 31 не запрограммирован, то процессор переходит в состояние "СТОП" по ошибке выполнения программы. В этом случае в АКК 1 заносится метка ошибки.

Команда EXXDXxxx создает в DB-RAM блок данных DX и работает как команда E DBxxx (допустимые параметры от 1 до 255).

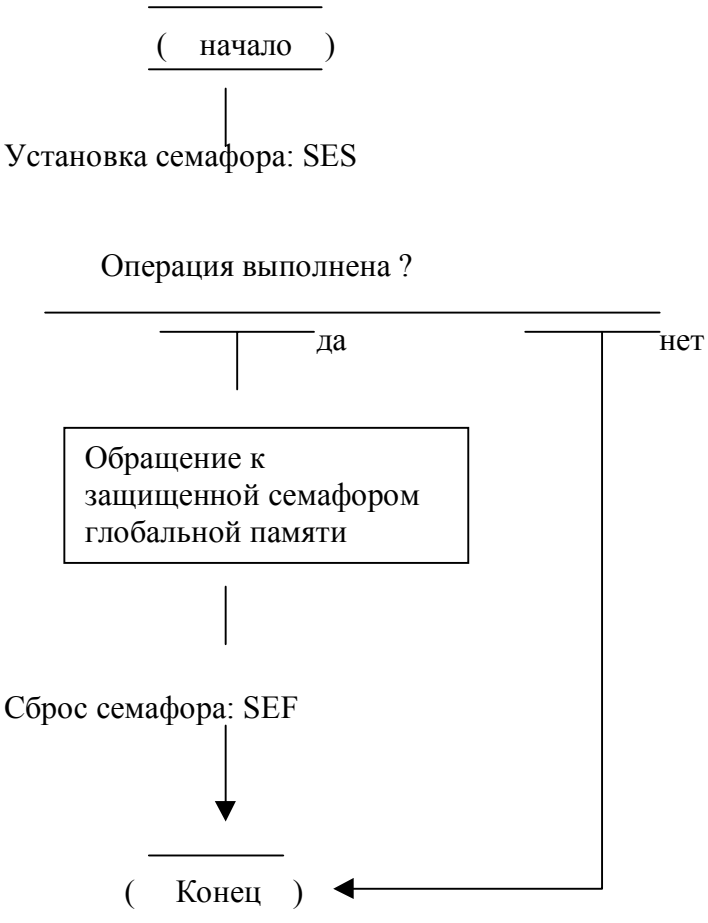
Операция	Параметр	Описание
SES	0…31	Установка семафора
SEF	0…31	Сброс семафора

SES/SEF: Установка/сброс семафора

Если в одном программируемом контроллере используются два или более процессора, некоторые глобальные области памяти (периферия CP, IP) окажутся общими, что представляет собой опасность, т.к. может оказаться, что процессоры переписывают одни и те же данные или считываются недействительные промежуточные состояния данных. Поэтому иногда требуется координировать доступ процессоров к общей области памяти.

Координирование отдельных процессоров возможно с помощью семафоров и команд SES и SEF. Только после успешной установки соответствующего семафора (SES) каждый из процессоров, участвующих в многопроцессорном режиме работы, может обращаться к общей области памяти. Семафор xx может при этом быть установлен только одним процессором. Если процессор не смог установить семафор, то он должен отказаться от попытки обращения к памяти. Точно также процессор не должен обращаться к памяти после сброса своего семафора (SEF).

Все участвующие в работе процессоры должны иметь функциональный блок со следующей структурой программы:

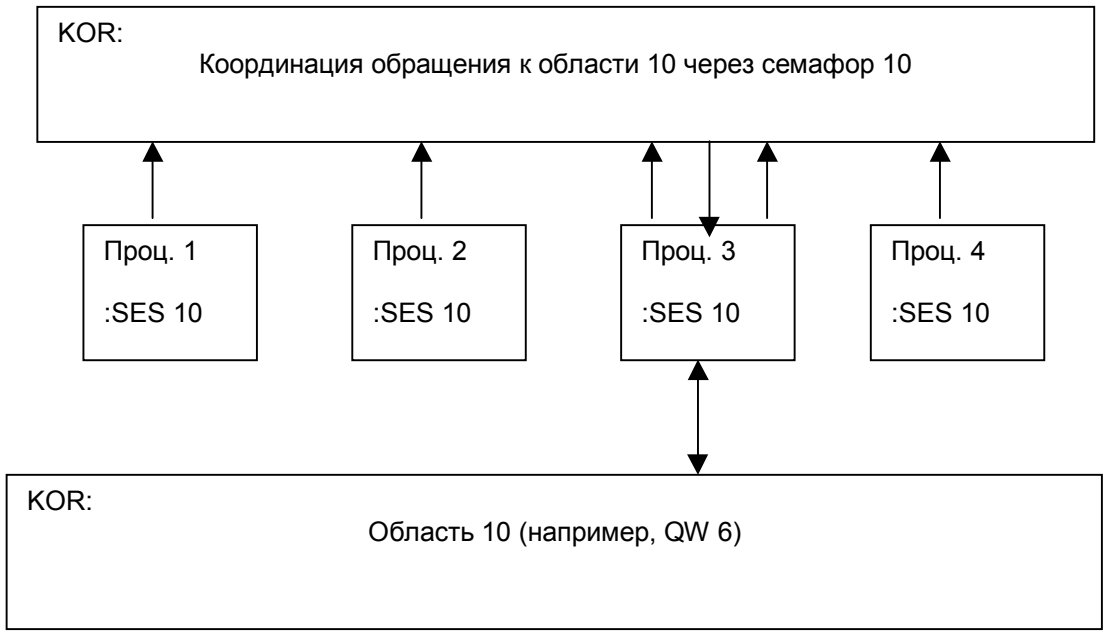


При использовании команд SES и SEF имеется возможность передачи процессорами совместно используемой информации защищенным образом в определенную область памяти, и считывать ее оттуда без вероятности прерывания данного процесса другими процессорами.

Важно!
Команды SESxxx и SEFxxx должны использоваться всеми процессорами, которые должны обращаться к общей глобальной области памяти (адрес >F000H).

Команда SESxxx (установка семафора) занимает для процессора, выставившего команду, определенный байт в координаторе (считаем, что он не занят другим процессором). Пока этот процессор держит свой байт, ни один из других процессоров не может обратиться к области памяти, защищенной семафором (номера от 0 до 31). При этом область блокирована для всех остальных процессоров.

Команда SEFxxx (деблокировка семафора) сбрасывает байт в координаторе. При этом защищенная область в памяти становится для чтения и записи всеми другими процессорами. Семафор может быть снят только тем же самым процессором, который его установил.



SES 10 Запрос на разрешение обращения к области 10

Разрешение на обращение для процессора 3 от KOR

Обращение к данным в области 10

SEF 10 Отмена разрешения на обращение

Перед каждой установкой или сбросом определенного семафора команда SES или SEF проверяет состояние (=статус) данного семафора. Биты индикации ANZ0 и ANZ1 зависят от этого следующим образом:

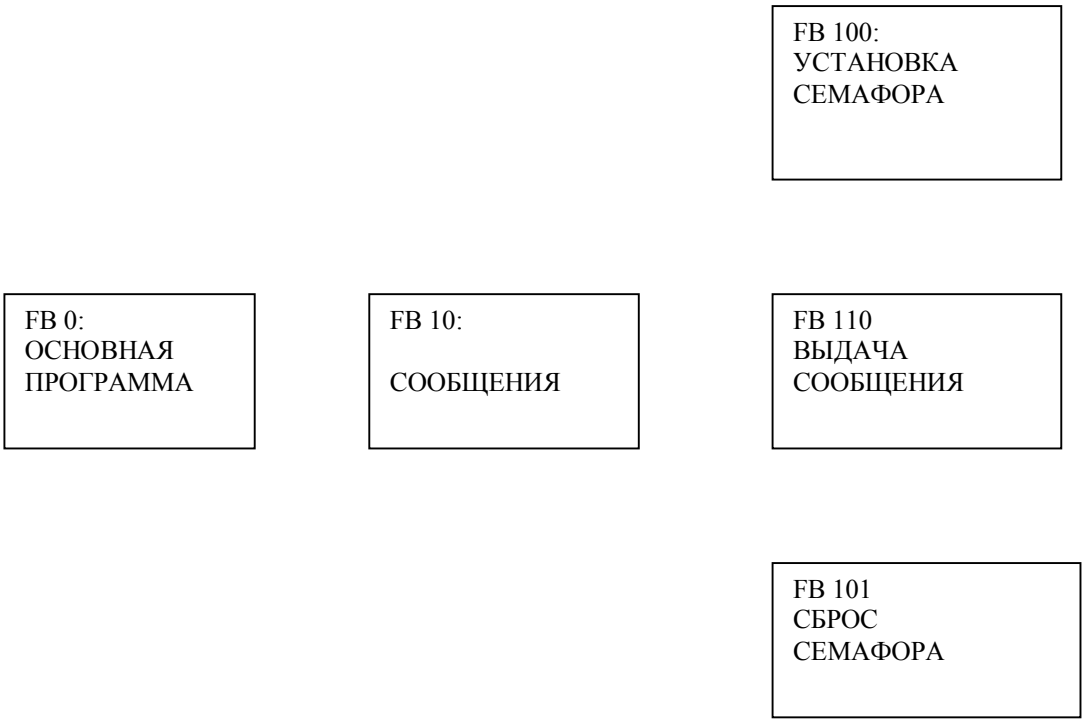
ANZ1	ANZ0	Значение	Обработка
0	0	Семафор был установлен другим процессором и не может быть установлен или сброшен	SPZ
1	0	Семафор устанавливается/сбрасывается	SPN,SPP

Важно!
Процесс опроса определенного семафора (= процесс чтения) и процесс установки пути сброса семафора (= процесс записи) составляет единое целое. Во время этого процесса никакой другой процессор не может обратиться к данному семафору!

При использовании семафора помните следующие моменты:
—Семафор является глобальной величиной, т.е. семафор с номером 16 при использовании в системе, например, трех процессоров может быть использован только один раз.
—Команды SES и SEF должны использоваться всеми процессорами, которые должны производить координированное обращение к общей области памяти.
—Все участвующие в работе процессоры должны иметь одинаковый режим запуска. При новом пуске все семафоры стираются, при автоматическом или ручном повторном пуске – сохраняют свое значение.
—Запуск в многопроцессорном режиме должен осуществляться синхронно. На этом основании недопустим тестовый режим.

Примеры использования семафоров

В программируемом контроллере S5-135U установлены процессоры, которые через общую область памяти Q-периферии (QW6) передают сообщение на устройство выдачи сообщений о состоянии. Каждое сообщение статуса должно выдаваться в течение 10 секунд и затем может быть заменено новым сообщением того же или другого процессора.
Использование периферийного слова QW6 (расширенная периферия, не имеет области отображения процесса) управляет с помощью семафора. При этом каждый из процессоров может записать в QW6 свое сообщение только в том случае, если он сумеет установить соответствующий семафор этой области. Семафор остается занятым (установленным) в течение 10 секунд (таймер T10). После завершения работы таймера процессор сбрасывает семафор и, таким образом, деблокирует занятую область для других процессоров. QW6 может быть использовано для нового сообщения.
Если при попытке процессора установить семафор будет обнаружено, что семафор уже занят другим процессором, то процессор повторяет попытку установить семафор и выдать свое сообщение в следующем цикле.
Следующая программа может работать во всех четырех процессорах с различными сообщениями. Должны быть загружены следующие блоки.



Используются 5 маркеров:
M10.0 = 1: сообщение запрашивается или обрабатывается
M10.1 = 1: семафор установлен
M10.2 = 1: таймер запущен
M10.3 = 1: сообщение передано
M10.4 = 1: семафор сброшен

```
FB0
Имя : MAIN

        : U     M  10,0          В случае, если нет активных сообщений
        : SPB = M001
        :
        :UN   E  0.0
        :BEB
        :
        :L   KH2222              Сообщение совпадает и
        :T   MW12
        :UN M 10.0
        :S     M 10.0            устанавливается маркер “сообщение”
        :
M001    :SPA  FB10               Вызывается FB “MELDE” (сообщение)
Имя     :MELDE
        :
        :BE

FB10

Имя     :MELDE
        :UN  M  10.1             Если семафор не установлен, вызывается
        :SPB FB100               FB “Установка семафора”
Имя     :SEMASET
        :
        :U     M 10.1            Если семафор установлен, и
        :UN  M  10.2             Таймер не запущен,
        :S     M  10.2
        :L     KT010.2           Запускается таймер
        :SV    T   10
        :
        :U      M  10.2          Если таймер запущен и
        :UN   M  10.3            Сообщение передано,
        : SPB   FB110            вызывается FB “Выдача сообщений”
Имя     :MELDAUSG
        :
        :U    M  10.2            Если таймер запущен и
        :UN  M  10.4             семафор не сброшен, и
        :UN   T  10              таймер завершил работу,
        :SPB  FB101              вызывается FB «Сброс семафора»
Имя     :SEMARESE
        :
        :UN    M  10.4           Если семафор сброшен
        :BEB
        : L   KH0000             все маркеры сброшены.
        :T    MB10               и все сообщения сброшены
        :BE
```

FB100

```
Имя    :SEMASET
       :SES     10                  Семафор No.10 Установлен
       :SPZ     = M001
       :UN      M  10.1             Если семафор установлен,
       :S       M  10.1             Устанавливается маркер
M001   :BE                         «Семафор установлен»
```

FB110

```
Имя    :MELDAUSG

       :L       MW12                Сообщение передается на периферию
       :T       QW6
       :UN      M  10.3
       :S       M   10.3            Установлен маркер «Передача сообщения»
```

FB101

```
Имя    :SEMARESE

       :SEF       10                Семафор No.10 сброшен
       :SPZ     = M001
       :UN      M   10.4
       :S       M   10.4            Установлен маркер «Семафор сброшен»
M001   :BE
```

4. Режим работ

4.1. Режимы работ и уровни обработки программы

Процессор может находиться в одном из трех режимов:

режим «СТОП» (STOP)
режим «ЗАПУСК» (ANLAUF)
режим «ОБРАБОТКА» (RUN)

Как описано в главе 2 , операционная система при возникновении определенной ситуации вызывает предусмотренные для этого организационные блоки (с OB 1 по OB 34), в которых пользователь может установить дальнейшую реакцию процессора. Если, например, при обновлении области отображения процесса в режиме RUN возникает задержка квитирования, то операционная система вызывает OB 24.

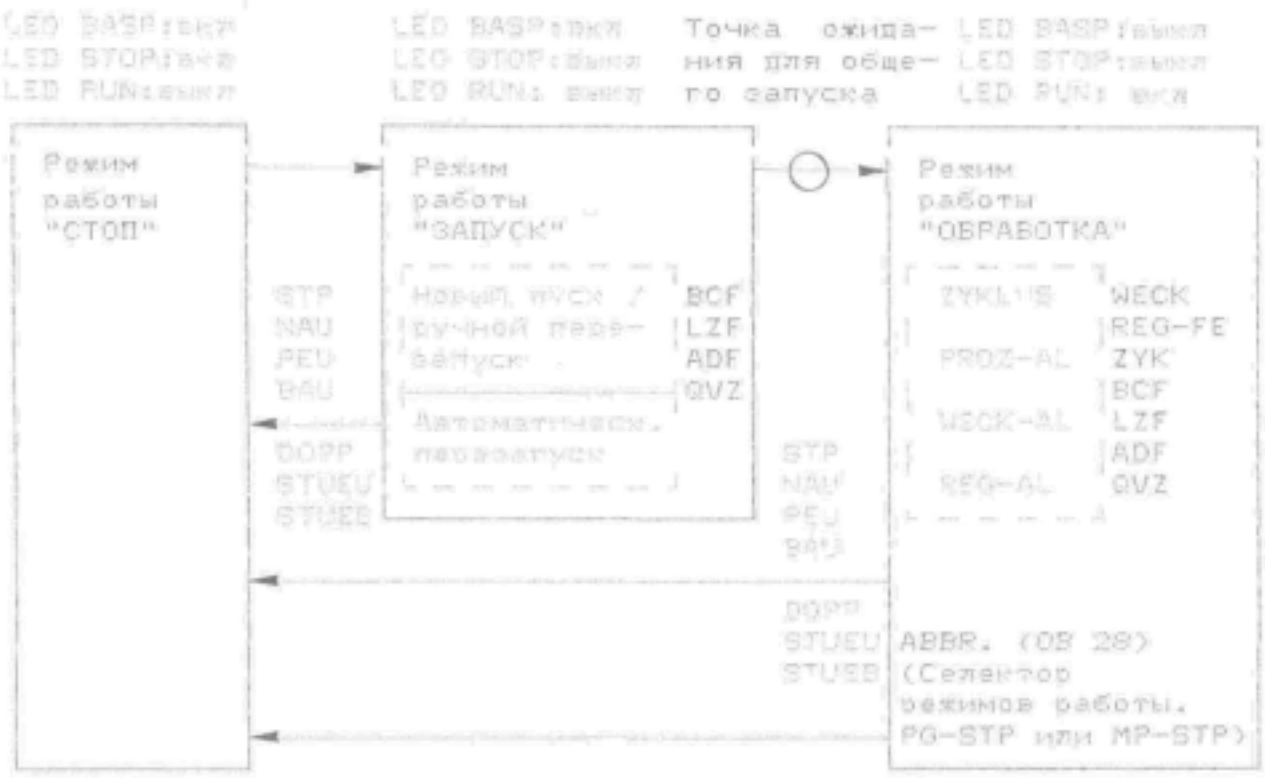
Одни реакции могут возникать только в режиме запуска, другие – только в режиме обработки, третьи – как при запуске, так и при обработке (см. следующую страницу).

После вызова организационного блока процессор выполняет содержащуюся в блоке STEP5 – программу пользователя. При этом устанавливается новое состояние регистров (регистры: аккумуляторы с 1 по 4, указатель стека блоков, начальный адрес блока данных, счетчик адресов STEP). Если из-за возникновения какой-либо ситуации будет прервана обычная обработка программы, то процессор после обработки OB включает все вызываемые из него блоки и продолжает обработку программы с места прерывания. В этот момент станет действительным старое содержимое регистров.

Одному или нескольким таким организационным блокам соответствует так называемый уровень обработки программ: если, например, вызывается OB 2, то активируется уровень обработки программ «прерывание от процесса». Уровень обработки программ «ошибка коды команды» активируется при вызове OB 27, OB 29 или OB 30.

На следующей странице вы найдете обзор рабочих состояний и уровней обработки программ в AG 135U, CPU 928.

Обзор 4-1: Рабочее состояние и уровни обработки программ



Уровни обработки программы при «ЗАПУСКЕ»:
НОВЫЙ ПУСК/РУЧНОЙ ПЕРЕЗАПУСК
АВТОМАТИЧЕСКИЙ ПЕРЕЗАПУСК
BCF (Ошибка кода команды)
LZF (Ошибка выполнения команды)
ADF (Ошибка адресации)
QVZ (Задержка квитирования)

Уровни обработки программы в режиме RUN:

ЦИКЛ	(Циклическая обработка программы)
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ 5 СЕК	(Обработка программы прерывания по времени)
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ 2 СЕК	(Обработка программы прерывания по времени)
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ 1 СЕК	(Обработка программы прерывания по времени)
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ 500 МС	(Обработка программы прерывания по времени)
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ 200 МС	(Обработка программы прерывания по времени)
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ 100 МС	(Обработка программы прерывания по времени)
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ 50 МС	(Обработка программы прерывания по времени)
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ 20 МС	(Обработка программы прерывания по времени)
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ 10 МС	(Обработка программы прерывания по времени)
ПРЕРЫВАНИЕ ОТ РЕГУЛЯТОРА	(Обработка программы прерывания от регулятора)
ПРЕРЫВАНИЕ ОТ ПРОЦЕССА	(Обработка программы прерывания от процесса)

WECK	(Ошибка Weck – “БУДИЛЬНИКА”)
REG	(Ошибка регулятора)
ZYK	(Ошибка цикла)
BCF	(Ошибка кода команды)
LZF	(Ошибка времени обработки)
ADF	(Ошибка адресации)
QVZ	(Задержка квитирования)
ABBR	(Прерывание)

Уровни обработки программы характеризуются следующими особенностями:

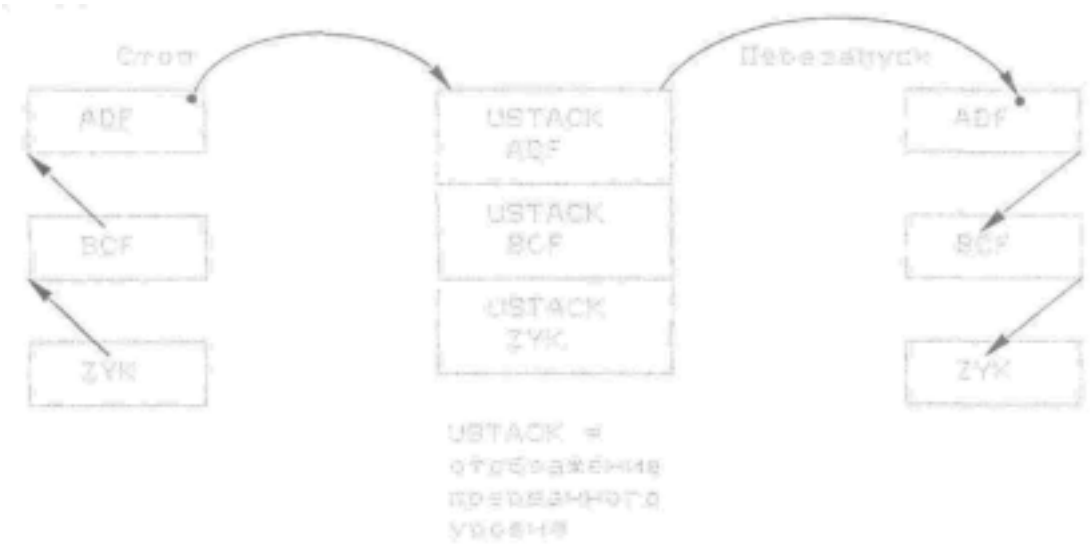
— Каждый уровень обработки программ имеет свою специфическую системную программу.

Пример:

На уровне обработки ЦИКЛ актуализируется системная программа области отображения входов и выходов, устанавливается время цикла и вызывается управление интерфейсом PG (точка контроля системы)

— Для каждого уровня обработки программы системная программа в случае прерывания создает некоторый стек прерываний и запоминает для него и для последующих уровень, в котором произошло прерывание (см. EBENE – уровень)

Пример:



— Уровни обработки программы имеют жесткий приоритет. От него зависит, будут ли прерывания обрабатываться друг за другом или путем вложений (см. следующую страницу)

«Перезапуск и уровень ошибок» отличается от «основного уровня» тем, что они, в основном, вызываются сразу и на границах команд, как только возникает соответствующая ситуация. Они могут вызываться как на «Основном уровне», так и друг из друга. При нескольких ошибках наивысший приоритет имеет та, что возникла последней.

«Основной уровень», наоборот, может вызываться только на границах блоков, однако это можно изменить при помощи соответствующей предустановки в DX0 (см. Главу ?).

Приоритет «Основных уровней»:
ЦИКЛ
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ [1)]
ПРЕРЫВАНИЕ ОТ РЕГУЛЯТОРА
ПРЕРЫВАНИЕ ОТ ПРОЦЕССА ↓ наивысший приоритет

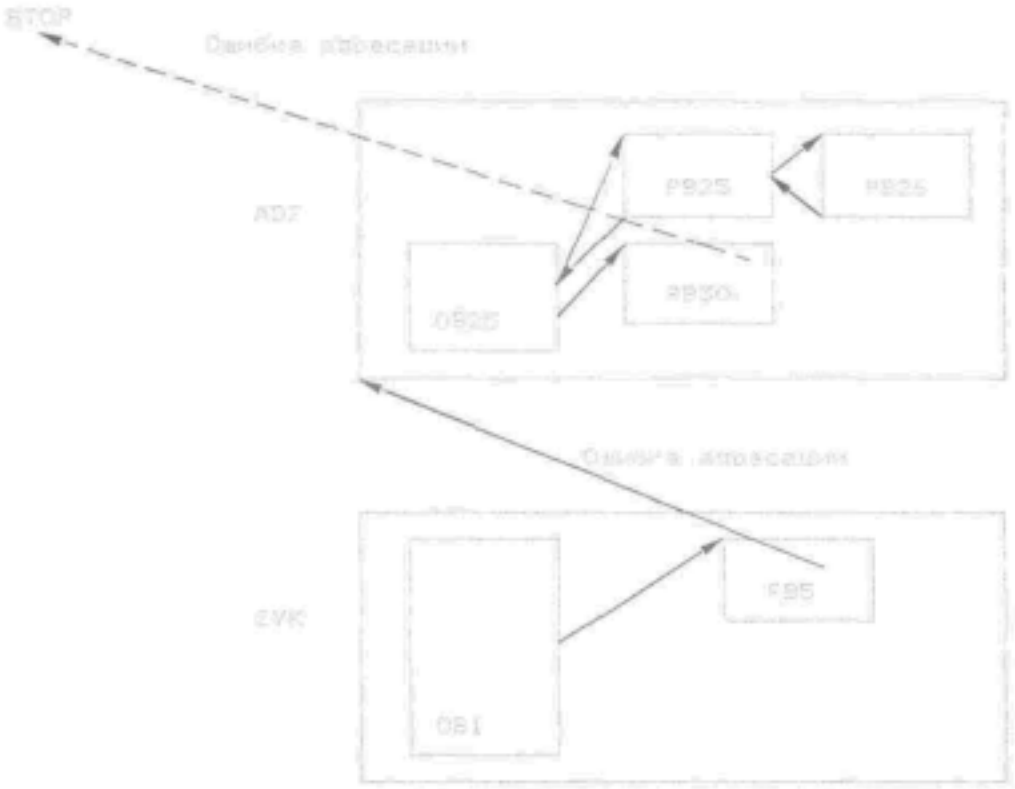
[1)] Прерывания по времени имеют и свой внутренний приоритет, причем, более короткие прерывания имеют более высокий приоритет.

Пример:
При обработке прерываний по времени возникает прерывание от процесса. Т.к. прерывание от процесса имеет большой приоритет, то обработка уровня прерывания по времени прерывается на следующей границе блока, и вызывается уровень обработки программы ПРЕРЫВАНИЯ ОТ ПРОЦЕССА. Но если при обработке прерывания от процесса возникает теперь, например, неверная адресация, то обработка прерывания от процесса будет прервана на следующей границе команды, для того, чтобы произошел вызов уровня ADF.

— Уровень обработки ошибок (ADF, BCF, LZF, QVZ, REG, ZYK), который активирован однократно и не до конца обработан, не может быть активирован еще раз. В этом случае AG переходит на уровень обработки повторного вызова (в USTACK: ‘DOPP’) и входит в состояние «СТОП» (исключение: ошибка будильника). В USTACK с глубиной вложения ‘01‘ метка ‘DOPP’, а также вызванный повторно уровень ошибки, помечаются крестиками.

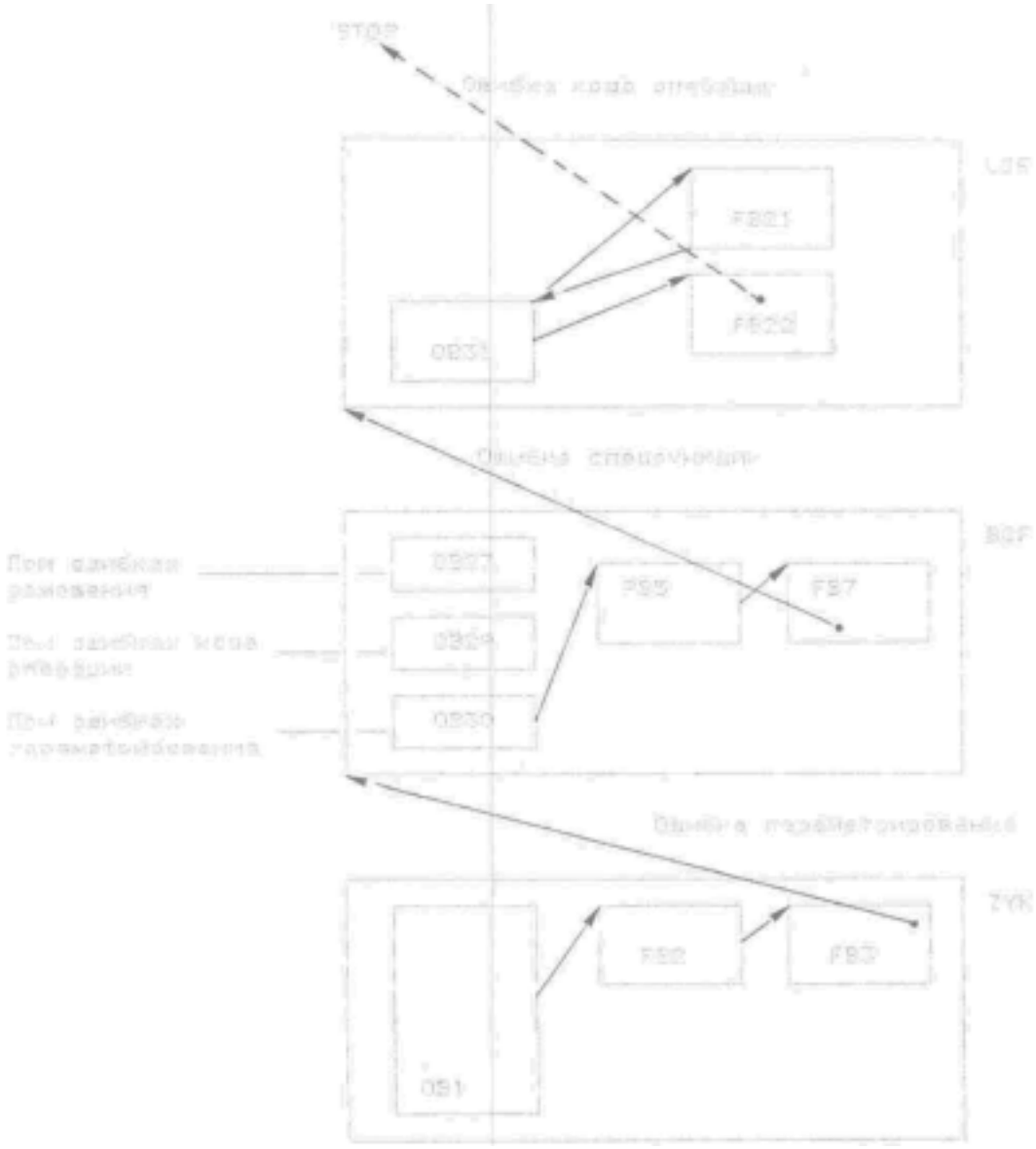
Пример 1:

При обработке уровня ADF (интерфейс пользователя OB 25) возникла другая ошибка адресации. Т.к. уровень адресации еще активирован, то не может быть активизирован еще раз: процессор переходит в «СТОП»



Пример 2 :

При возникновении ошибки кода операции на уровне обработки программы LZF операционная система пытается вызвать уровень BCF (интерфейс пользователя OB 29). Однако он уже активирован при возникновении ошибки параметризации (интерфейс пользователя OB 30) и не был полностью обработан. Поэтому новый вызов уровня BCF на этом месте недопустим: переходит в «СТОП».



Отдельные уровни обработки программ с соответствующими интерфейсами более подробно описаны в следующих разделах:

Раздел 4.3 «основной уровень» при ЗАПУСКЕ
Раздел 4.4 «основной уровень» при ОБРАБОТКЕ
Раздел 4.5 «уровни ошибок» при ЗАПУСКЕ и ОБРАБОТКЕ

4.2. Режим работы “СТОП”

Состояние “СТОП” характеризуется следующими особенностями:

- Отсутствует обработка программы пользователя
- Если перед этим имела место циклическая обработка программы, то значения счетчиков, таймеров, маркеров и области отображения процесса при переходе в состояние “СТОП” сохраняются.
- Выдается сигнал BASP (блокировка выходов). При этом блокируются все цифровые выходы.

  Исключение: В режиме тестирования сигнал BASP не выдается!
  (Режим тестирования см. Раздел 10.5)
- Если перед этим происходила обработка программы, то в состоянии “СТОП” должен иметься стек прерываний (USTACK).

Светодиоды на передней панели центрального модуля:

RUN : Выкл

STOP : Вкл. (постоянно или мигает)

BASP : Вкл. (кроме режима тестирования)

Светодиод STOP сигнализирует Вам, по какой причине произошел останов. Горение светодиода (постоянное или мигающее) имеет определенное значение, которое в дальнейшем коротко описано.

Светодиод STOP горит постоянно

Режим работы “СТОП” может быть вызван в однопроцессорном режиме:

a) при переключении селектора режима работ из “RUN” и “STOP”

b) при функции PG “AG-STOP”

c) при аппаратной ошибке (BAU, PEU, NAU)

d) после полного стирания

в многопроцессорном режиме:

a) при переключении селектора режима работ на координаторе в “STOP”

b) другой процессор из-за неисправности перешел в “СТОП” (на процессоре, не имеющем ошибки, светодиод горит постоянно)

c) PG - функция “AG-STOP” на процессоре

d) при PG – функции “КОНЕЦ КОНТРОЛЯ ОБРАБОТКИ” на другом процессоре

Светодиод STOP <u>медленно мигает</u>: (1 раз в 2 секунды)

Медленное мигание светодиода STOP сигнализирует Вам в большинстве случаев об ошибке.

В многопроцессорном режиме медленное мигание светодиода обозначает, что данный процессор явился причиной остановки (из-за возникновения ошибки).

Светодиод STOP медленно мигает:

a) при программировании команды “СТОП” в программе пользователя

b) при ошибке обслуживания (ошибка DB1, выбор недопустимого режима запуска, и т. д.)

c) при программной и аппаратной ошибках (вызов незагруженного блока, ошибка адресации, задержка квитирования, ошибка кодов команд и т. д.) в качестве дополнительного указания на причину ошибки горят:

светодиод ADF
светодиод QVZ
светодиод ZYK

d) при функции PG «КОНЕЦ КОНТРОЛЯ ОБРАБОТКИ» на данном процессоре

Светодиод STOP <u>мигает быстро</u>: (2 раза в секунду)

Быстрое мигание светодиода STOP сигнализирует Вам о сообщении:
“Требуется общий сброс”!

Требования общего сброса

a) Система требует общего сброса:

После каждого включения напряжения питания и после выполнения общего сброса процессор выполняет программу инициализации.
Если теперь при инициализации возникает ошибка, процессор переходит в состояние “СТОП” с быстрым миганием.

Возможные ошибки: 1. Содержимое RAM пусто

Устранение: общий сброс процессора

2. ERROM пользователя пуст или не установлен

Устранение: установить запрограммированный ERROM и произвести общий сброс процессора.

Причина неисправности должна быть устранена, и процессор должен быть полностью очищен.

b) Пользователь требует общего сброса:

Вы можете произвести общий сброс, если выполните следующую последовательность действий:

- Переведите селектор режимов работ из ‘RUN’ в ‘STOP’.
  Результат: Процессор находится в состоянии «СТОП».
  Светодиод STOP горит постоянно.

- Поставьте переключатель выбора в положение ‘OVERALL RESET’; одновременно переведите селектор режимов работ из ‘STOP’ в ‘RUN’ и снова в ‘STOP’.
  Результат: Требуется общий сброс.
  Светодиод ‘STOP’ быстро мигает.

Важно!
Если вы не хотите выполнить общий сброс, то в этот момент вы должны выбрать режим запуска.

Выполнение общего сброса

Независимо от того, затребован ли общий сброс системой или пользователем, вы должны производить его следующим образом:

- Установите переключатель выбора в положение ‘OVERALL RESET’: одновременно переведите селектор режимов работы из ‘STOP’ в ‘RUN’ и снова в ‘STOP’
  Результат: Общий сброс выполнен
  Светодиод STOP горит постоянно

- Или: с помощью функции PG ‘СТИРАНИЕ’
  (При общем сбросе с помощью PG можно исключить ручной общий сброс переключателями! Положение переключателей выбора и режима работ не имеет значения)
  Результат: Общий сброс выполнен
  Светодиод STOP горит постоянно

Важно!
Если вы произвели общий сброс, то в качестве режима запуска допускается только <u>новый пуск</u>!

Выход из положения «СТОП»:-Выберите режим запуска
(см. Раздел 4.3)
-Общий сброс, с новым пуском
-Режим тестирования (при многопроцессорном режиме, см. Раздел 10.5)

4.3. Режим работы ЗАПУСК

Для режима "ЗАПУСК" характерны следующие особенности:
—«ЗАПУСК» является переходом из состояния STOP в состояние RUN.

—Имеются различные режимы запуска: новый пуск ручной перезапуск и автоматический перезапуск.
После нового пуска циклическая программа пользователя обрабатывается с начала. После перезапуска циклическая программа пользователя продолжается с места прерывания.

—Для всех трех режимов запуска операционная система вызывает один из организационных блоков, в котором может быть заложена программа запуска. Длина программы запуска на языке STEP 5 в OB не ограничена. По времени она не контролируется. Из OB запуска могут быть вызваны другие блоки.

—Значения счетчиков, таймеров, маркеров и область отображения процесса при различных режимах запуска обрабатываются по разному.

—Выдаётся сигнал BASP (блокировка выходов). При этом все цифровые выходы блокируются.
Исключение: В тестовом режиме BASP не выдается! (о тестовом режиме см. раздел 10.5)

Светодиоды на передней панели центрального модуля:

RUN:	выкл
STOP:	выкл
BASP:	вкл (за исключением тестового режима)

Примечание:
Указания по "Запуску в многопроцессорном режиме" Вы найдете в Разделе 10.4.

<u>Новый пуск Вы можете выполнить следующим образом</u>:

- Установите переключатель выбора в положение “RESET” ; одновременно переведите селектор режимов работ из 'STOP” в "RUN”
- Или с помощью PG-Функции 'AG-СТАРТ' (—> новый пуск).

Новый пуск необходим после:

- Переполнения стека (USTACK: “STUEU,STUEB”)
- Повторного вызова одного и того же уровня обработки программ (USTACK: “DOPP”)
- Общего сброса (бит управления: “URGELOE”)
- Прерывание запуска (бит управления: “ANL-ABB”)
- Стоп после PG-функции «Конец контроля обработки»

Новый пуск допустим всегда, если система не требует полного стирания:

Ручной перезапуск Вы можете осуществить следующим образом:

o Переключатель выбора находится в среднем положении.

o Переведите селектор режима работ из «STOP» в “RUN”.

o Или: с помощью функции PG “AG--СТАРТ” (→ Перезапуск).

Ручной перезапуск допустим всегда. За исключением тех случаев, в которых допустим исключительно новый пуск (см. выше) или система требует общий сброс.

Выполнение автоматического перезапуска:

При сбое сетевого напряжения во время «Запуска» или «Обработки» и повторного включения напряжения процессор выполняет программу инициализации и затем автоматически осуществляет перезапуск.

Предустановки:

- Переключатели на всех процессорах и координаторе должны неизменно стоять в положении “RUN”
- При инициализации не должно возникать ошибок.

Если Ваш процессор при сбое напряжения питания и последующем включении питания должен выполнять автоматический новый пуск, то Вы должны изменить предустановки в блоке данных DX 0.

ВАЖНО!
Ручной или автоматический перезапуск допустимы, только если программа пользователя в состоянии «СТОП» не изменялась.

4.3.1.Новый пуск и ручной перезапуск

В следующей таблице содержится обзор видов запуска: новый пуск и ручной перезапуск.

Временная последов.	Новый пуск	Ручной перезапуск
Причина: | | | | V | | | |	Селектор режимов из положения STOP в положение RUN, а переключатель выбора в положение RESET PG-функция AG-СТАРТ (новый пуск) Включение питания если в DX0 установлен автоматический новый пуск	Селектор режимов из положения STOP в RUN и переключатель выбора в среднее положение PG-функция AG-СТАРТ (ручной перезапуск)
| V Возможности системной программы | | V | | | V | | | V | | | V | | |	- список блоков содержится в DX0 - область отображения входов стирается - область отображения выходов стирается - маркеры, таймеры, счетчики стираются - цифровая/аналоговая периферия стирается (2 x 128 байт) - маркеры связи стираются (256 байт) - семафоры стираются (все 32) - имеется DB1: установленные в нем цифровые входы/выходы и входные/ выходные маркеры связи передаются в список PA - DB1 отсутствует: действительно существующие модули (только цифровые входы/ выходы) передаются в список PA (маркеры связи игнорируются) - обрабатывается блок параметрирования регулятора DB2	- В DB0 сохраняется список блоков - Маркеры, таймеры, счетчики сохраняются - Маркеры, связи сохраняются - Семафоры сохраняются
| V	- вызывается интерфейс пользователя OB 20 (если имеется)	- Вызывается интерфейс пользователя OB 20 (если имеется)
| V	- запуск в многопроцессорном режиме синхронизирован	- Запуск в многопроцессорном режиме синхронизирован

Новый пуск: Программирование организационного блока OB 20

Если процессор выполняет новый пуск, то автоматически вызывается OB 20. Вы можете там расположить программу на языке STEP5, которая будет выполняться однократно перед началом циклической обработки программы:

Вы можете, например,

- вставить маркер
- запустить таймер
- с помощью прямого доступа к периферии установить выходы
- подготовить обмен данными между процессором
- произвести синхронизацию с CP (блок управления SYNCHRON, см. Раздел 6.9).

Не забудьте завершить OB 20 инструкцией BE (конец блока)!

После обработки OB 20 начинается обработка программы путем вызова OB 1 или FB 0.

Если OB 20 не запрограммирован, то процессор начинает после окончания нового пуска (после выполнения системной программы) циклическую обработку программы.

Если Вы в DX 0 ввели «Режим автоматический новый пуск при повторном включении напряжения», то при повторном включении напряжения будет также вызываться OB 20.

Ручной перезапуск: программирование организационного блока OB21

Если процессор выполняет ручной перезапуск, то вызывается OB 21. Вы можете занести туда программу на языке STEP5, которая будет выполняться один раз перед перезапуском циклической обработки программы.

Не забудьте завершить OB 21 инструкцией BE (конец блока)!

После обработки OB 21 продолжается циклическая обработка программы со следующей инструкции после прерывания. Это означает:

- Сигнал BASP (блокировка выходов) сохраняется на время обработки оставшейся части цикла и сбрасывается при начале следующего полного цикла.
- Область отображения выходов при обработке оставшейся части цикла сброшена и обновляется только по окончании следующего полного цикла.

Если OB 21 не запрограммирован, то процессор по окончании ручного перезапуска начинает обработку программы с места прерывания.

4.3.2. Автоматический перезапуск

Функция автоматического перезапуска идентична ручному перезапуску. Различие состоит только в способе разрешения.

Временная последов.	Автоматический перезапуск
| | Разрешение: | | V	Повторное включение сетевого напряжения после отключения
| | V Действия системной программы: | | | V | | | V | |	- Список блоков в DB 0 сохраняется - Маркеры, таймеры, счетчики сохраняются - Маркеры, связи сохраняются - Семафоры сохраняются
| V	- Вызывается интерфейс пользователя OB 22 (если имеется)
| V	- Запуск в многопроцессорном режиме синхронизирован

Автоматический перезапуск: программирование организационного блока OB22

При восстановлении напряжения питания процессор пробует продолжить прерванную программу.
При этом сначала вызывается OB 22, в котором Вы можете разместить программу на языке STEP5, которая будет выполняться один раз перед перезапуском циклической обработки программы.
Если Вы хотите избежать ситуации, когда процессор автоматически выполняет перезапуск, то запрограммируйте в OB 22 инструкцию «СТОП» и завершите его инструкцией «BE» (конец блока)!

```
OB 22 :  STP        (Стоп)
      :  BE         (Конец блока)
```

Результат: Процессор при восстановлении напряжения питания переходит в «СТОП».

После обработки OB 22 циклическая обработка программы продолжается со следующей инструкции на месте прерывания. После сбоя напряжения с последующим его восстановлением:

- Сигнал BASF (блокировка выходов) остается в течение обработки оставшейся части цикла в прежнем состоянии и сбрасывается в начале следующего полного цикла.
- Область отображения выходов остается сброшенной во время выполнения оставшейся части цикла и обновляется только по окончании следующего (полного) цикла.

4.3.3. Прерывание во время ЗАПУСКА

Программа запуска может быть прервана по следующим причинам

- Сбой напряжения
- Приведение в действие селектора STOP/RUN
- Программная и аппаратная ошибка (см. Раздел 6.6)

Если прерванный запуск необходимо продолжить с помощью одного из трех возможных видов запуска, то обратите внимание на следующее:

При **сбое напряжения в момент запуска** и последующем восстановлении напряжения нужно различать три случая:

1. Процессор переходит непосредственно в новый пуск (OB 20). После сбоя напряжения последующего восстановления на месте прерывания организационный блок OB 22 (автоматический перезапуск) вкладывается в OB 20.
2. Процессор переходит непосредственно к ручному перезапуску (OB 21). После сбоя напряжения и последующего его восстановления организационный блок OB 22 (автоматический перезапуск) вкладывается в OB 21 на месте прерывания.
3. Процессор выполняет автоматический перезапуск (OB 22). После сбоя напряжения и последующего его восстановления нельзя вложить следующий OB 22: прерванный OB 22 после восстановления напряжения питания не продолжается а, значит, прерывается, и затем обрабатывается заново вызванный OB 22.

«СТОП» с помощью селектора в режиме запуска и последующий ручной перезапуск

Если Вы прервали любой процесс запуска, нажав селектор «СТОП» (или PG-функцией ”AG-STOP”), и разрешили затем ручной перезапуск, то будет продолжен прерванный процесс запуска на месте прерывания. OB 21 не будет вкладываться!

“СТОП” выключателем в процессе запуска и последующий новый пуск

Если Вы прервали любой процесс запуска. Нажав селектор “СТОП” (или PG-функцией “AG-STOP”). И хотите завершить это новым пуском, то запуск прерывается, и выполняется новый пуск (если есть, - то с OB 20).

4.4. Режим работы ОБРАБОТКА

Состояние RUN характеризуется следующими особенностями:

- Программой пользователя обрабатывается циклически.
- Все запущенные в программе счетчики и таймеры «работают», область отображения процесса циклически актуализируется.
- Сигнал BASF (блокировка выходов) отсутствует. При этом все цифровые выходы разблокированы.
- Маркеры связи циклически обновляются.

Светодиоды на передней панели центрального модуля.

RUN: вкл.

STOP: выкл.

BASF: выкл.

ВАЖНО!
Если перед переходом в режим RUN был выполнен автоматический или ручной перезапуск, то в первый момент загорается светодиод BASF, пока обрабатывается остаток цикла и актуализируется область отображения процесса.

ВАЖНО!
Режим RUN достигается только после режима ЗАПУСК!

В режиме RUN имеется 12 основных уровней обработки программ:

1. Цикл:	Программа пользователя обрабатывается циклически.
2. 9 Прерываний по времени:	Обработка программы пользователя управляется по времени
3. Прерывание от РЕГУЛЯТОРА:	Дополнительно к программе пользователя обрабатывается заданное количество регуляторов
4. Прерывание от процесса:	Обработка программы пользователя управляется прерыванием от процесса.

Они имеют следующие различия:

a) Определяются различными условиями.

b) На каждом уровне обработки программ операционная система программа выполняет различные функции.

c) Для каждого уровня обработки программ существуют в качестве интерфейсов пользователя свои организационные или функциональные блоки.

4.4.1. ЦИКЛ: Циклическая обработка программы

В свободнопрограммируемых системах с памятью, в основном используется **циклическая обработка программы.**

Разрешение запуска:

Если процессор завершил программу запуска без ошибок, то он начинает циклическую обработку программы.

Действия операционной системы:

- К началу цикла устанавливается контрольное время цикла.
- Обновляется область отображения входов (RAE).
- Обновляются входные маркеры связи.
- Вызывается интерфейс пользователя.
- По окончании цикла обновляется область отображения выходов (PAA).
- Обновляются выходные маркеры связи.

Интерфейс пользователя: OB 1 или FB 0

При циклической обработке программ в качестве интерфейса пользователя регулярно вызывается организационный блок OB 1 или функциональный блок FB 0 с начала и последовательно с помощью вызовов различных блоков. После выполнения системных манипуляций процессор начинает новый цикл с первой инструкции на языке STEP5 в OB 1 (или FB 0).

В OB 1 программируются вызовы программных, функциональных и шаговых блоков, которые должны обрабатываться в циклической программе.

Если имеется короткая программа, критичная по времени, в которой Вы можете отказаться от структурированного программирования, то программируйте FB 0: т. к. он работает на всем наборе операций STEP5, то Вы можете сэкономить вызовы блоков и таким образом сократить время обработки программ.

ВАЖНО!
Вы должны запрограммировать либо OB 1 либо FB 0.
Если запрограммирован OB 1 и FB 0, то будет вызываться только OB 1. Если Вы используете в качестве интерфейса пользователя FB 0, этот последний не должен содержать параметров!

Точки прерывания

Циклическая обработка программы может быть кратковременно прервана на границе блоков для:

- обработки программы, управляемой по прерыванию от процесса;
- обработки регулятора;
- обработки программы, управляемой по времени;

(Путем параметрирования DX 0 Вы можете задать эти прерывания также на границе команд!)
Вы можете произвести прерывание на границе команд или вообще прервать обработку:

- при возникновении аппаратной или программной ошибки:
- с помощью устройств управления (PG-функции, селектора режимов работы).

Указание:
Регистры АКК 1, 2, 3 и 4 могут на границах цикла (с конца цикла программы и до конца следующего) использоваться как память данных.

4.4.2. ПРЕРЫВАНИЯ ПО ВРЕМЕНИ: Обработка программ, управляемых по времени

Обработка программы, управляемой по времени, выполняется, если от внутренних часов приходит сигнал (будильник) на процессор, который прерывает циклическую обработку программы и начинает выполнять специальную программу. После обработки этой программы процессор возвращается на место прерывания в циклической программе и продолжает дальнейшую обработку.
При этом определенные части программы автоматически сдвигаются в циклической обработке программы в заданном временном растре.

В CPU 928 можно обрабатывать до 9 различных программ, управляемых по времени, каждая из которых вызывается в своем временном растре.

Разрешение:
Прерывание по времени разрешается автоматически в соответствующем временном растре при условии, что запрограммирован необходимый ОВ.

Интерфейсы пользователя: ОВ 10… ОВ 18

В качестве интерфейса пользователя при возникновении определенного прерывания по времени, так называемого «будильника», на следующей границе блока (или границы команды) вызывается соответствующий организационный блок.

Соответствие	Организационный блок	Временной растр
	ОВ 10	Время обработки 10 мс
	ОВ 11	Время обработки 20 мс
	ОВ 12	Время обработки 50 мс
	ОВ 13	Время обработки 100 мс
	ОВ 14	Время обработки 200 мс
	ОВ 15	Время обработки 500 мс
	ОВ 16	Время обработки 1 с
	ОВ 17	Время обработки 2 с
	ОВ 18	Время обработки 5 с

Если запрограммировали, например, ОВ 13 какую-либо часть программы, то в циклической обработке программы все должно сдвинуться на 100 мс.

Вы можете запрограммировать все эти девять блоков ОВ, какую-либо их часть или не программировать вообще. Если не запрограммирован ни один из девяти организационных блоков, то обработка программ управляемых по времени выполняться не будет.

При каждом ОВ – будильников (ОВ 10 по ОВ 18) в АККу 1 заказывается сколько возникло временных растров с момента последнего вызова ОВ – будильников.

При этом:

АККу 1 := Число временных растров - 1

Если, например, при вызове ОВ 11 в АККу 1 заносится число «5», то это означает, что с последнего вызова ОВ 11 прошло 120 мс (6 временных растров). Пока не возникает ошибки будильника. В АККу 1 загружается «0».

Приоритетность прерываний по времени

Уровни обработки ПРЕРЫВАНИЯ ПО ВРЕМЕНИ расположены относительно других уровней следующим образом:

ЦИКЛ |
ПРЕРЫВАНИЕ ПО ВРЕМЕНИ |
ПРЕРЫВАНИЕ ОТ РЕГУЛЯТОРА |
ПРЕРЫВАНИЕ ОТ ПРОЦЕССОРА V

Приоритет отдельных прерываний по времени установлен так:

OB 18 (самый большой временной растр)
OB 17 |
OB 16 |
. |
. |
OB 11 |
OB 10 (самый маленький временной растр) V Увеличение приоритета

Важно!
OB с более коротким временным растром имеют более высокий приоритет и могут прерывать работу OB с более длинным временным растром.

Точки прерывания

Обработка программы, управляемой по времени, может быть прервана либо на границе блоков (стандартная предустановка), либо на границе команд (программирование DX 0) из-за:

- обработки другой программы прерывания, управляемой по времени;
- обработки регуляторов;
- обработки программы прерывания от процесса;

а на границе команды

- программной или аппаратной ошибки.

Пример:

Во время обработки OB 14 (вызов каждые 200 мс) был вызван OB 12 (вызов каждые 50 мс). OB 14 прерывается на следующей границе блока или команды и обрабатывается OB 12. После того как его обработка полностью завершается (возможны прерывания от регулятора, от процессора, от обработки ошибок или от OB 10 или OB 11), продолжается и завершается обработка программы в OB 14.

Ошибки «БУДИЛЬНИКА»

Важно!
Обработка программы, управляемой по времени, не может быть прервана новой обработкой программы, управляемой по времени!

Если какой-либо OB прерывания по времени еще не обработанный полностью, был вызван повторно так как истекло время, то возникает ошибка будильника. Ошибка будильника возникает также, если какой-либо OB вызывается во второй раз до того как был обработан первый вызов. Это возможно при возникновении “Прерывания по времени на границе блока”, особенно, если Ваша программа на языке STEP5 содержит долго выполняемые блоки.

При возникновении ошибки “БУДИЛЬНИКА” активирует уровень обработки программы WECKFE, на котором системная программа в качестве интерфейса пользователя вызовет ОВ 33. В ОВ 33 Вы можете запрограммировать необходимую Вам реакцию на эту ситуацию.
При вызове ОВ 33 ОПЕРАЦИОННАЯ СИСТЕМА ЗАНОСИТ В АККу 1 и 2 дополнительную информацию, которая поясняет первую возникшую ошибку:

АКК 1	АКК 2	
1001H	0016H	Ошибка «БУДИЛЬНИКА» в ОВ 10 (10 мс)
1001H	0014H	Ошибка «БУДИЛЬНИКА» в ОВ 11 (20 мс)
1001H	0012H	Ошибка «БУДИЛЬНИКА» в ОВ 12 (50 мс)
1001H	0006H	Ошибка «БУДИЛЬНИКА» в ОВ 13 (100 мс)
1001H	000EH	Ошибка «БУДИЛЬНИКА» в ОВ 14 (200 мс)
1001H	000CH	Ошибка «БУДИЛЬНИКА» в ОВ 15 (500 мс)
1001H	000AH	Ошибка «БУДИЛЬНИКА» в ОВ 16 (1 с)
1001H	0008H	Ошибка «БУДИЛЬНИКА» в ОВ 17 (2 с)
1001H	0006H	Ошибка «БУДИЛЬНИКА» в ОВ 18 (5 с)

Указание: Метка АКК 2 является меткой уровня прерывания по времени вызывающего ошибку.

Если ОВ 33 не запрограммирован, процессор перейдет в состояние “СТОП”. Затем с помощью программатора при “выводе USTACK” под управляющим битом “WECKFE” Вы увидите крестик, а в USTACK, в качестве метки уровня (УРОВЕНЬ), указано соответствующее прерывание по времени.

Если необходимо продолжить обработку программы при возникновении ошибки “БУДИЛЬНИКА”, то либо запрограммируйте в ОВ 33 инструкцию конца блока. Либо измените предустановку в DX 0 таким образом, чтобы при возникновении ошибки «БУДИЛЬНИКА» и незапрограммированном ОВ 33 обработка программы продолжалась. После обработки ОВ 33 обработка программы продолжится с места прерывания в ОВ 13.

Примечание:

- Обратите внимание при обработке программ управляемых по времени на новые спецфункции ОВ 120, ОВ 121, ОВ 122 и ОВ 123, с помощью которых Вы можете блокировать или задерживать обработку прерываний по времени на определенных участках программы. (Это возможно либо для всех запрограммированных прерываний по времени либо для некоторых из них).
- Чем быстрее уровень обработки программ управляемых по времени. Тем больше опасность ошибки «будильника». Прерывания по времени с короткими временными растрами (например 10мс и 20 мс прерывания по времени) требуют в основном, чтобы была выбрана установка прерывания на границе команды. Это возможно при условии, что прерывания от регулятора и от процессора тоже установлены на прерывание на границе команд (см. главу 7. DX 0 параметрирование).

4.4.3. ПРЕРЫВАНИЕ ОТ РЕГУЛЯТОРА: Обработка РЕГУЛЯТОРА

В CPU 928 наряду с циклической обработкой программы, обработкой программы прерывания по времени и от процесса возможна дополнительная обработка РЕГУЛЯТОРА. Во временном отрезке, установленном пользователем (= времени опроса), может быть прервана циклическая обработка программы или программа обработки прерывания по времени и начаться обработка соответствующего РЕГУЛЯТОРА. После этого процессор вернется на место прерывания в циклическую программу или в программу прерывания, управляемую по времени, и продолжит там свою обработку.

Разрешение:

Периодичность прерываний от РЕГУЛЯТОРА задается пользователем.

Действия операционной системы:

- Запускает интерфейс пользователя для обработки РЕГУЛЯТОРА.
- Обновляет область отображения РЕГУЛЯТОРА.

Интерфейс пользователя: Стандартный функциональный блок «Структура регулятора R64»

При обработке РЕГУЛЯТОРА в качестве интерфейса пользователя вызывается стандартный функциональный блок R64. Данный блок позволяет совместно с блоком параметрирования РЕГУЛЯТОРА DB 2 производить обработку от 1 до 64 РЕГУЛЯТОРОВ.
Для каждого РЕГУЛЯТОРА Вы должны запараметрировать определенный блок данных. В блоке данных DB 2 Вы располагаете так называемый «список РЕГУЛЯТОРОВ», который указывает, какой РЕГУЛЯТОР в какой момент времени должен быть обработан операционной системой. DB 2 резервируется для этой задачи!
(При параметрировании, эксплуатации и тестировании стандартного FB R64 Вы можете использовать для поддержки специальный пакет программ “COMREG”. Заказной номер для PG 685: 6ES5 395-3SA11).

Точки прерывания

Обработка РЕГУЛЯТОРА может быть прервана либо на границе блоков (стандартная предустановка) либо на границе команд (запрограммировано в DX 0) из-за:
- обработки программы, управляемой по прерыванию от процесса.

Вы можете также произвести прерывание на границе команд при
- программной или аппаратной ошибке.

4.4.4. ПРЕРЫВАНИЕ ОТ ПРОЦЕССОРА: Обработка программы управляемой по прерыванию от процесса

Обработка программы управляемой по прерыванию от процесса, производится в момент, когда на шине S5 появляется сигнал от модуля цифровых входов, установленного на прерывания (например, 6ES5 432-4UA11), или от модуля интеллигентной периферии. Обработка программы прерывается, и начинается выполнение специальной программы. По завершению обработки этой программы процессор возвращается на место прерывания и продолжает свою работу с этого места.

Разрешения:

Активное состояние сигнала прерывания на шине S5 вызывает прерывание от процесса. В зависимости от места установки каждому процессору соответствует одна из 7 линий прерывания (см. Руководство по эксплуатации CPU 928).

Интерфейс ПОЛЬЗОВАТЕЛЯ: OB 2

В качестве интерфейса пользователя при возникновении прерывания от процесса вызывается OB 2. В OB 2 Вы можете запрограммировать специальную программу, которая должна обрабатываться в случае прерывания процесса.

Если OB 2 не запрограммирован, обработка программы не прерывается, и обработка программы прерывания не происходит.

Точки прерывания

Обработка программы прерывания от процесса может быть прервана только при возникновении программной или аппаратной ошибки (на границе команд)

ВАЖНО!
Обработка программы, управляемой по прерыванию от процесса, не может быть прервана обработкой программы прерывания по времени или другой программой прерывания от процесса!

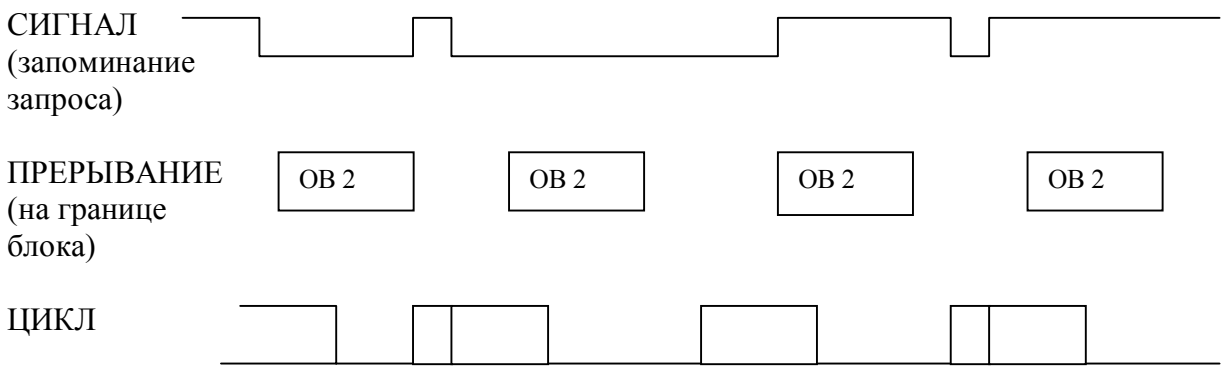
Если во время обработки программы прерывания от процесса возникает новое прерывание от процесса, то последнее игнорируется, пока не будет полностью обработан OB 2 (включая все вызываемые из OB 2 блоки).
Затем процессор возвращается на место прерывания и обрабатывает программу по следующей границе блока. Затем принимается новое прерывание от процесса, и заново вызывается OB 2. При этом возможно длительное прерывание циклической программы. Это не действительно для случая, когда в DX0 установлено «Прерывание от процесса на границе команды»).

Сигнал прерывания от процесса: установка по уровню

Обычно сигнал прерывания от процесса в CPU 928 устанавливается уровнем. Это означает, что активное состояние сигнала прерывания создает запрос, который на следующей границе блока или команды ведет к обработке OB 2. Этот запрос запоминается и сбрасывается после выполнения команды завершения блока (PE) в OB 2.

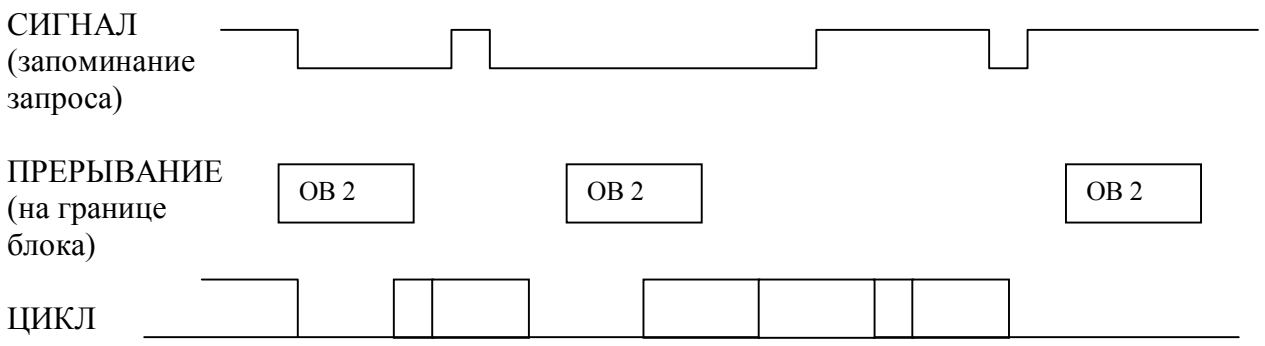
Поэтому в результате:
- Многократное прерывание не опознается
- Прерывание, которое возникло при обработке OB 2 и короче, чем OB 2, не опознается
- OB 2 вызывается также, если при достижении границы блока состояние сигнала прерывания уже снова пассивно (см. рисунок)

Вызванный OB 2 полностью обрабатывается. Если в конце OB 2 уровень всё еще установлен или установлен заново, то в циклической программе обрабатывается блок (команда), и затем снова вызывается OB 2. Если уровень больше не установлен, то при следующей установке сигнала (отрицательный фронт) OB 2 вызывается заново.



Сигнал прерывания от процесса: установка по фронту

Эта установка возможна если Вы запараметрируете ее в DX 0. После обработки 0B 2 новое прерывание от процесса может возникнуть только при смене состояния сигнала (из неактивного в активный).При прерываниях от процесса; устанавливаемых по фронту, запрос на обработку 0B 2 также держится до тех пор пока этот не будет полностью обработан.
Смена состояния сигнала, которая возникает во время обработки OB 2, игнорируется.



Блокировка обработки программ, управляемых по прерыванию от процесса

Программа, управляемая по прерыванию от процесса, вставляется в циклической программе на границе команды STEP5.

Это прерывание может сказаться отрицательно, если циклическая часть программы должна быть обработана за определенное время (Например, для того, чтобы было достигнуто определенное время реакции), или следующая команда не может быть прервана (например, при чтении или записи последовательного набора значений).

Если невозможно или нельзя прервать часть программы прерыванием от процесса, то возникают следующие возможности программирования:

- Запрограммируйте эту программу так, чтобы она не содержала смены блока и установите в DX 0 предустановку («Прерывание от процесса на границе блока»). Часть программы, которая не содержит смены блоков, не будет прерываться.

- Напишите свою программу в программе, управляемой по прерыванию. Здесь она не может быть прервана другим прерыванием.

- Запрограммируйте программу STEP 5 ‘AS’ (блокировка прерывания от процесса). С помощью команды ‘AF’ (деблокировка прерывания от процесса). Вы можете снова восстановить возможность прерывания. Между этими двумя командами прерывание невозможно, и стоящая между ними часть программы не может быть прервана от процесса.

  Использование ‘AS’ и ‘AF’ возможно только в функциональных блоках (расширенный набор операций)!

- Используйте новые спецфункции OB120 и OB122, с помощью которых Вы можете блокировать или задержать обработку возникающих прерываний от процесса на определенных участках программы.

Установка приоритетов при одновременном возникновении прерываний от процесса и по времени

Если во время обработки программы прерывания (граница блока или команды) прерывается, и обрабатывается прерывание от процесса. Затем завершается обработка прерывания по времени.

Если во время обработки программы прерывания от процесса возникает прерывание по времени, то сначала завершается обработка программы прерывания от процесса и только затем выполняется программа прерывания по времени.

Если одновременно возникает прерывание от процесса и по времени, то на следующем месте прерывания сначала обрабатывается прерывание от процесса. Затем, если это прерывание от процесса обработано, то обрабатывается имеющееся или установленное прерывание по времени.

Время реакции

Время реакции на сигнал прерывания по времени соответствует времени обработки блока или команды STEP 5 (в зависимости от выбранной предустановки). Однако, если в момент прерывания циклической обработки программы появилось еще прерывание от процесса, то программа прерывания по времени обрабатываться только после полной обработки всех возникших прерываний от процесса.

Максимальное время реакции между возникновением и обработкой прерывания по времени изменяется в этом случае в зависимости от времени обработки прерывания от процесса. Если вы хотите потом исключить ошибку будильника для определенных ОВ **xy** прерывания процесса, обратите внимание на следующие правила:

A + B < C, причем, A= сумма времен обработки всех высокоприоритетных уровней обработки программ (ОВ прерываний от процесса, от регулятора, по времени)
B = время обработки ОВ **xy** прерывания по времени
C = временной растр ОВ **xy** прерывания по времени

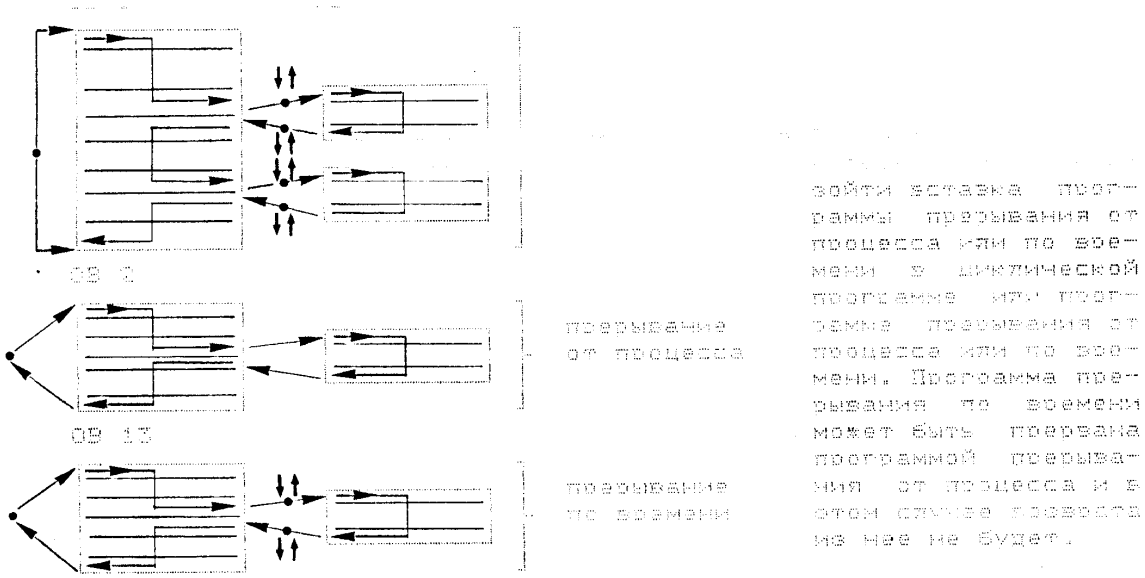


Рис. 4-2: Обработка программы прерывания на граница блока

ВАЖНО!
Если Вы свою программу обрабатываете не только циклически, а также по времени и/или по прерыванию, возникает опасность, что например, маркеры, которые используются в циклической программе в качестве промежуточных маркеров, при прерывании циклической обработки программы будут переписаны. Поэтому Вы должны перед началом обработки программы по времени или по прерыванию от процесса сохранить состояния сигналов маркеров в блоке данных и восстановить после окончания обработки прерывания.

Для этой цели в Вашем распоряжении имеется 4 организационных блока спецназначения: ОВ 190 и 0В 192 "Передача маркеров в блок данных" и ОВ 191 и ОВ 193 «Передача блока данных в маркерную область» (см. ниже).

5. Обработка прерываний и ошибок

Операционная система может обнаружить ошибку в работе процессора, ошибку в операционной системе или программе пользователя.

5.1. Наиболее распространенные ошибки в программе пользователя

Показанный далее список содержит указания на ошибки, которые наиболее часто встречаются в программах пользователя, но которые, однако. Легко избежать при создании программы.
На основании выше сказанного, обратите внимание на следующие моменты при программировании Ваших программ на языке STEP5:

- При вводе адресов байтов для входов и выходов необходимо установить соответствующие этим адресам модули в центральное устройство или устройство расширения.
- Помните. Что все операнды должны иметь корректные параметры.
- Выходы, маркеры, таймеры и счетчики не должны обрабатываться в разных местах программы с операциями противоположенного действия.
- Проследите, чтобы все открываемые в программе блоки данных имелись в наличии и были достаточной длины.
- Проверьте, действительно ли находятся в памяти все вызываемые из программы блоки.
- Будьте внимательны при последующих изменениях функциональных блоков. Контролируйте запрограммирован ли FB правильными операндами, и все ли фактические операнды заданы.
- Таймеры должны опрашиваться только один раз в цикл (например, U T1).

5.2. Обработка информации об ошибках

Если при запуске или во время циклической обработки программы пользователя возникает ошибка, то Вы получаете в свое распоряжение различные «источники информации» для того, чтобы определить ее источник.

a) Светодиоды на передней панели процессора

В случае незапланированного останова ориентируйтесь на светодиоды. По их свечению Вы можете выяснить причину ошибки:

- светодиод STOP горит постоянно
- светодиод STOP медленно мигает
- светодиод STOP быстро мигает

Различные состояния светодиод STOP указывают на определенные причины прерывания и ошибок.

Обратите внимание при этом на пояснения. Содержащиеся в Разделе 4.2 «Режим работы STOP».

Светодиод ошибки на передней панели горит постоянно при:

- ADF (ошибка адресации)
- QVZ (Задержка квитирования)
- ZYK (ошибка времени цикла)

b) On-line- функция «Вывод USTACK» (см. Раздел 5.3.)

С помощью On-line- функции “AG-INFO” и затем «вывод USTACK» Вы можете получить справку о состоянии управляющих битов и о содержании стека памяти прерываний (=USTACK).

Системная программа заносит в USTACK при переходе в состояние останова всю информацию, которая необходима для нового пуска или для повторного запуска. Эта информация при диагностике предоставляет значительную помощь. Полный USTACK выдается только в состоянии останова.

Перед выводом собственного USTACKа сначала на экран выводится состояние **управляющих битов**. Они маркируют текущей режим работы процессора и программы пользователя и выдают дополнительные разъяснения по причине ошибки.
On- line-функция «Вывод USTACK» запускается не только в состоянии «СТОП», но также и в состоянии «ЗАПУСК» и RUN («ОБРАБОТКА»). Однако, в этом случае Вы получите на экране только вызов управляющих битов.

c) Системные данные BS 3 и 4 (см. Раздел 5.5)

Если Ваш процессор уже при запуске обнаружил ошибку и вследствие ошибки перешел в состояние «СТОП», то причину ошибки можно точно определить по системным словам данных BS 3 и 4. В этом случае речь идет об ошибке, на которую «натыкается» операционная система при конфигурировании списка адресов в DB 0 или DB 1 или при обработке DB 2 или DX 0.

Системное слово BS 3: КН = EA03 (абсолютный адрес памяти)

Опознав ошибки в системном слове данных BS 3 Вы выясните, что привело к возникновению ошибки.

Опознав ошибки в системном слове данных BS 4 Вы выясните, где возникла ошибка.

Информация об ошибках хранится в формате КН.

Обработка системных слов данных BS 3 и BS 4 с помощью программатора:

- С помощью On-line-функции «Справка АДРЕС» (КН = EA03 или EA04) Вы можете непосредственно считать содержимое обоих системных слов данных и, таким образом, определить причину ошибки.

d) АКК 1 и АКК 2 (см. Раздел 5.6)

Если при обработке программы STEP5 в режиме «Запуск» или «Цикл» возникает ошибка, для которой в качестве интерфейса пользователя имеется организационный блок, то системная программа автоматически при вызове соответствующего организационного блока заносит в АКК 1 и АКК 2 дополнительную информацию, которая может прояснить причину ошибки.

Информация об ошибке в АКК 1 подскажет вам, что явилось причиной ошибки.

Информация в АКК 2 (если она имеется) подскажет Вам, где возникла ошибка.

Информация об ошибке выводится в формате КН.

Обработка АКК 1 и АКК 2 с помощью программатора:

- С помощью On-line-функции «Вывод USTACK» Вы можете содержание обоих аккумуляторов считать непосредственно из USTACKa и, таким образом, точно определить причину ошибки.

Обработка АКК 1 и АКК 2 с помощью STEP5:

- Т. к. информация об ошибке при вызове организационного блока ошибки автоматически заносится в АКК 1 и «. Вы можете использовать эти данные при программировании Ваших ОВ.

Вместе с тем, можно предусмотреть в организационном блоке различные реакции на различные ошибки в зависимости от занесенных в аккумуляторы данных об ошибках.

e) On-line-функция «Вывод USTACK»

С помощью On-line-функции «AG INFO» и затем «Вывод BSTACK» Вы можете получить при возникновении «СТОП» содержимое стека памяти блоков (=BSTACK) (см. Раздел 3.1.1)

В BSTACK заносятся все блоки. Начиная с OB 1 или FB 0. Которые вызывались последовательно до перехода в состояние «СТОП» и не обработаны до конца. Т.к. BSTACK заполняется снизу, то в верхней ячейке BSTACK расположен текущий блок, который обрабатывался последним и в котором возникла ошибка.

При обработке верхней строки Вы получите следующую информацию:

BAUST.-NR	Тип блока и номер блока. Который перед возникновением состояния «СТОП» обрабатывался.
BAUST.-ADR	Абсолютный адрес начала этого блока в памяти программы.
RUCKSPR.-ADR.	Абсолютный адрес следующей обрабатываемой команды в этом блоке. С помощью этой команды процессор может продолжить программу после выполнения ручного перезапуска.
REL.-ADR.	Относительный адрес (=RUCKSPR.-ADR- BAUST.-ADR) следующей обрабатываемой команды в этом блоке. (Относительный адрес может быть выведен на экран PG в режиме работы «блокировка ввода» (ключевой переключатель)).
DD-NR.	Номер последнего вызванного блока данных.
DB-ADR.	Абсолютный начальный адрес этого блока данных (адрес слова данных DW 0) в памяти программ.

Пример: обработка «Вызов BSTACK»

BAUST.-NR	BAUST.-ADR	RUCKSPR.-ADR	REL/-ADR	DB-NR	DB-ADR.
OB 23	0063	0064	0001	13	0078
FB 5	006A	0072	0008	13	0078
FB 6	008F	0091	0007	100	0098
OB 1	009D	009E	0001		

В приведенном примере состояние «СТОП» возникло в OB 23 при обработке инструкции STEP5, которая в памяти расположена по абсолютному адресу «0064 – 1 = 0063».

OB 23 (OB ошибки OVZ) был вызван в FB 5 по относительному адресу «0008 – 1 = 0007».

В FB 6 был вызван блок данных DB 100. При пере ходе процессора в состояние «СТОП» действительным является блок данных DB 13.
Блок данных DB 13 был вызван в FB 5.

Выводы

Постарайтесь воспользоваться при поиске причины ошибки всей доступной для Вас информацией.

Это могут быть:

1. Светодиоды на передней панели процессора:

   Картина свечения светодиодов указывает на причину ошибки или прерывания.

2. ON-line функция «Вывод USTACK»:

   Вы получите в каждом случае вывод в управляющих битах, а в состоянии «СТОП» также вывод USTACK.

3. Системные слова BS 3 и 4:

   При ошибках в режиме запуска Вы найдете в системных словах данных BS 3 и 4 указания на причину ошибок.

4. АКК 1 и АКК 2:

   При вызове организационного блока ошибки операционная система заносит в АКК 1 и 2 дополнительную информацию об ошибках.

5. ON-line функция «Вывод BSTACK»:

   В состоянии «СТОП» Вы можете по верхней строке USTACKа определить блок и адрес команды, при обработке которой возникла ошибка.

5.3. Управляющие биты и стек прерывания

С помощью ON-line функции «AG-INFO» (F7) и затем «Вывод USTACK» (F5) Вы можете проанализировать рабочее состояние процессора и программы пользователя с целью выяснения возможной причины ошибки прерывания.

ВАЖНО!
Вывод управляющих битов Вы можете получить в любом режиме работы, вывод USTACKа – только в состоянии «СТОП».

- Управляющие биты дадут Вам информацию о текущем или предыдущем рабочем состоянии и о причине сбоя. Если возникло несколько ошибок, то в управляющих битах будут представлены все появившиеся ошибки.
- В USTACKе представлено место прерывания (адрес) и содержимое аккумуляторов а также причина сбоя Если возникло несколько ошибок, то стек прерывания имеет несколько вложений:
  Глубина 01 = последняя причина сбоя.
  Глубина 02 = предпоследняя причина сбоя и т. д.

  При переполнении USTACKа происходит автоматический переход в состояние «СТОП». Для выхода из этого состояния требуется новый пуск.

Значение отдельных сокращений управляющих битов и стека прерываний пояснены ниже. Набор этих сокращений хранится в PG685.

Важно!

Текст на экране Вашего программатора может отличаться от приведенного здесь. В этом случае для Вас должно быть действительным описание конкретных позиций приведенное в данном руководстве по программированию!

БИТЫ УПРАВЛЕНИЯ

>>STR<<	STR-6	FE-STR	BARBEND	PG-STP	STP-SCH	STP-BEF	MP-STP
>>ANL<<	ANL-6	NEUSTX	MWA	AWA	ANL-2	NEU-ZULX	MWA-ZULX
>>RUN<<	RUN-6	EINPROZX	BARB	OB1GEL	FBOGELX	OBPROZA	OBWECKA
32KRAM	16KWRAM	SKWRAMX	EPROM	KM-AUS	KM-EIN	DIG-EINX	DIG-AUSX
URGELOE	URL-IA	STR-VER	ANL-ABB	UA-PG	UA-SYS	UA-PRFE	UA-SCH
DX0-FE	FE-22	MOD-FE	RAM-FE	DB0-FE	DB1-FE	DB2-FE	KOR-FE
NAU	PEU	BAU	STUE-FE	ZYK	QVZ	ADF	WECK-FE
BCF	FE-6	FE-5	FE-4	FE-3	LZF	REG-FE	DOPP-FE

При вводе на PG USTACK на первой странице экрана будут выведены состояния управляющих битов.
Показанные ниже управляющие биты маркируют текущее или предыдущее состояние процессора и выдают справку по определенным характеристикам процессора и программы пользователя STEP5.
Управляющие биты выдаются во всех режимах работы! Таким образом, можно в любой момент определить, загружен ли организованный блок OB 2 и тем самым возможна обработка программы прерываний или нет.

STP Процессор в состоянии «СТОП»: следующие биты указывают причины останова:

STP-6	Не используется
FE-STP	Ошибка «СТОП»: Состояние «СТОП», возникающее после ошибок NAU (сбой напряжения буферной батареи), STUEB (переполнение BSTACKa), STUEB (переполнение BSTACKa), DOPP (ошибка дублирования) или ошибки процессора
BARBEND	Конец контроля обработки: Состояние «СТОП» после функции PG «Конец контроля обработки». (Необходим новый пуск).
PG-STP	PG-стоп: Состояние «СТОП» по команде от PG/
STP-SCH	Селектор режимов в «СТОП»: состояние «СТОП» при переводе селектора режимов в состояние ОСТАНОВ
STP-BEF	Команда «СТОП»: а) Состояние «СТОП» при обработке операции «STP» б) Состояние «СТОП» по команде от операционной системы, если не запрограммирован организационный блок ошибки.
MP-STP	Многопроцессорный «СТОП»: а) Селектор выбора на координаторе стоит в положении STOP, либо б) Останов других процессоров при многопроцессорном режиме

ANL Процессор в режиме работы ЗАПУСК:

ANL-6	Не используется
NEUST	Новый пуск запрошен или был выполнен в качестве последнего запуска.
MWA	Ручной перезапуск запрошен, либо выполнялся в качестве последнего запуска.
AWA	Автоматический перезапуск после сбоя напряжения питания запрошен, либо выполняется в качестве последнего запуска.
ANL-2	Не используется
NEU-ZUL	Новый пуск допустим в качестве следующего запуска
MWA-ZUL	Ручной перезапуск допустим в качестве следующего запуска.

RUN Процессор в режиме работы RUN (активна циклическая обработка программы:

RUN-6	Не используется
EINPROZ	Однопроцессорный режим
BARB	On-line-функция «Контроль обработки» активен.
OB1GEL	Организационный блок OB 1 загружен в память пользователя. Циклическая обработка программы определяется OB 1
FBOGEL	Функциональный блок FB 0 загружен в память пользователя. Циклическая обработка программы определяется через FB 0, если не загружен OB 1. Если одновременно загружены FB 0 и OB 1, то для циклической обработки программы действителен OB 1.
OPBROZA	Загружен организационный блок прерываний от процесса OB 2, т.е. возможна обработка программы прерывания от процесса.
OBWECKA	Загружен организационный блок обработки прерывания по времени, т.е. возможна обработка программы прерывания по времени.

32KWRAM	Модуль памяти пользователя RAM 32 К слов
16KWRAM	Модуль памяти пользователя RAM 16 К слов
8KWRAM	Модуль памяти пользователя RAM 8 К слов
EPROM	Модуль памяти пользователя EPROM
KM-AUS	Список адресов для выходных маркеров связи расположен в DB 1.
KM-EIN	Список адресов для входных маркеров связи расположен в DB 1.
DIG-EIN	Список адресов для цифровых входов расположен в DB 1.
DIG-AUS	Список адресов для цифровых выходов расположен в DB 1.
URGELOS	Был выполнен общий сброс (необходим новый пуск).
URL-IA	Выполняется общий сброс.
STR-VER	Причина останова процессора – в центральном модуле.
ANL-ABB	Прерывание во время запуска (необходим новый пуск).
WA-PG	PG требует общего сброса.
UA-SYS	Операционная система требует общего сброса (запуск невозможен): должен быть выполнен общий сброс.
UA-PRFE	Требуется общий сброс из-за ошибки процессора.
UA-SCH	Требуется общий сброс от переключателя: выполняется общий сброс или, если общий сброс не нужен, выбирается режим запуска.

Показанные ниже управляющие биты маркируют ошибки, которые могут возникнуть в режимах работы ЗАПУСК (например, при первом новом пуске) и RUN (например, при обработке программы прерывания по времени)

Если возникло несколько ошибок, то в последних трех строках управляющих битов указываются все возникшие на данный момент (и еще не обработанные) причины прерываний. Обратите при этом внимание на системное слово BS 2: Оно содержит UAMK (общее слово индикации прерываний, 16 бит), в котором также хранятся все возникшие и еще не обработанные ошибки (Глава 8.2.4).

Ошибки при ЗАПУСКЕ:

DX0-FE	Ошибка параметрирования DX 0
FE-22	Не используется
MOD-FE	Содержимое модуля программы пользователя содержит ошибку (при RAM необходим общий сброс).
RAM-FE	Содержимое RAM операционной системы или DB- RAM содержит ошибку (необходим общий сброс).
DB0-FE	Неверная конфигурация блока списков адресов DB 0.
DB1-FE	Неверная конфигурация блока списков адресов DB 1 для обновления области отображения процесса: а) DB 1 при установленном координаторе или в многопроцессорном режиме не программируется; б) Указанные в DB 1 адреса байтов для входов, выходов или маркеров связи не подтверждаются при новом пуске соответствующими модулями.
DB2-FE	Ошибка при обработке блока данных параметрирования DB 2 структурой регулятора R64

Ошибка в режиме работы ЗАПУСК или RUN:

KOR-FE	Ошибка при обмене данными через координатор.
NAU	Сбой напряжения питания в центральном устройстве.
REU	Ошибка периферии = сбой напряжения в устройстве расширения.
BAU	Ошибка батареи = сбой напряжения буферной батареи (центральное устройство).
STUE-FE	Переполнен стек прерываний или блоков (глубина вложения слишком велика; необходим новый пуск).
ZYK	Превышено время цикла.

QVZ	Задержка квитирования при обмене данными с периферией.
ADF	Ошибка адресации входов или выходов. Ошибка возникает при обращении к области отображения периферии, когда были опрошены модули периферии, которые при последнем новом пуске не установлены, дефектны или не были указаны в DB 1
WECK-FE	Ошибка “будильника” (программы прерывания по времени): Если перед или во время обработки определенного OB прерывания по времени он вызывается повторно.
BCF	Ошибка кода команды: а) Ошибка замещения: обрабатываемая команда STEP5 не замещена; б) Ошибка кода операции: обрабатываемая команда STEP5 неверна; в) Ошибка параметра: неверный параметр, обрабатываемый командой STEP5 .
FE-6	Не используется
FE-5	Не используется
FE-4	Ошибка сбоя напряжения: Обработка предшествующего сбоя напряжения (NAU) операционной системой выполнена с ошибкой: поэтому перезапуск запрещен.
FE-3	Не используется
LZF	Ошибка времени цикла: а) Вызываемый блок не загружен б) Ошибка передачи блока данных в) Особые ошибки времени процесса
REG-FE	Ошибка при обработке структуры регулятора R64 в цикле.
DOPP-FE	Ошибка дублирования: Активный в данный момент уровень обработки программ (ADF, B активируется во второй раз (необходим новый пуск).

После вывода управляющих битов на экран PG и нажатия клавиши квитирования на следующей странице экрана появляется USTACK.
Операционная система заносит сюда при переходе в состояние «СТОП» всю информацию, которая необходима для нового пуска или перезапуска.

ВАЖНО!
Текст на экране Вашего программатора может отличаться от приведенного здесь. В этом случае действительным является описание конкретных позиций на экране в данном руководстве по программированию!

USTACK

TIEFE: 02

BEF-REG	C70A	BAZ:	00F3	DB-ADR:	0000	BA-ADR:	0000
BST-STP:	0000	FB-NR:	226	DB-NR:		-NR:	
		REL-ADR:	0006	DBL-REG:	0000		
EBENE:	004	UAMK:	0100	UALW:	0000		

AKKU1:0000 C464 AKKU2:0000 00FF AKKU3:0000 0000 AKKU4:0000 0000

KLAMMERN: KE1 111 KE2 100 KE3 111

БИТЫ РЕЗУЛЬТАТА:	ANZ1	ANZ0	OVFL	OVFLS	ODER	STATUS	VKE	ERAB
		X				X	X	
ПРИЧИНЫ ОШИБКИ:	NAU	PEU	BAU	HALT	ZYK	OVZ	ADF	STP
							X	X
	BCF	S-6	LZF	REG	STUEB	STUEU	WECK	DOPP

Приведенные ниже данные USTACK а содержат сведения. С помощью которых можно найти в программе пользователя инструкцию, при обработке которой процессор перешел в состояние «СТОП».

TIEFE	Уровень USTACK при вложении ошибок TIEFE 01 = последняя возникшая причина неисправности TIEFE 02 = предпоследняя возникшая причина неисправности ……..
BEF-REG	Регистр команд: Содержит машинные коды (первое слово) последней обработанной команды прерванного уровня обработки программ.
BST-STP	Указатель стека блока: Содержит число сохраняемых в данный момент элементов в стеке блоков (USTACK).

УРОВЕНЬ Z выдает уровни обработки программ, которые подразделяются на:

Z: 0002 = Новый пуск
0004 = Цикл
0006 = Прерывание по времени 5 сек (OB 18)
0008 = Прерывание по времени 2 сек (OB 17)
000A = Прерывание по времени 1 сек (OB 16)
000C = Прерывание по времени 500 мс (OB 15)
000E = Прерывание по времени 200 мс (OB 14)
0010 = Прерывание по времени 100 мс (OB 13)
0012 = Прерывание по времени 50 мс (OB 12)
0014 = Прерывание по времени 20 мс (OB 11)
0016 = Прерывание по времени 10 мс (OB 10)
0018 = Не используется
001A = Не используется
001C = Обработка регулятора
001E = Не используется
0020 = Не используется
0022 = Не используется
0024 = Прерывание от процесса
0026 = Не используется
0028 = Не используется
002A = Не используется
002C = Переход в состояние “СТОП” в многопроцессорном режиме, селектор режимов в “СТОП” или “СТОП” от PG
002E = Не используется
0030 = Ошибка прерывания по времени
0032 = Ошибка РЕГУЛЯТОРА
0034 = Ошибка цикла
0036 = Не используется
0038 = Ошибка кода команды
003A = Ошибка выполнения команды
003C = Ошибка адресации
003E = Задержка квитирования
0040 = Не используется
0042 = Не используется
0044 = Ручной перезапуск
0046 = Автоматический перезапуск

SAZ Счетчик STEP-адресов:

Содержит абсолютный адрес последней обработанной команды прерванного уровня обработки в памяти программ. При ошибке SAZ точно указывает на команду, являющуюся причиной ошибки!

Если ошибка расположена не на уровне STEP5, например, при обработке регулятора, в SAZ стоит 0, и содержимое BEF-REG не актуально.

…NR Тип и номер блока, обрабатывавшегося последним.

REL-SAZ Относительный счетчик адресов STEP:

Содержит относительный адрес (относительно начала блока) последней обработанной команды в последнем обработанном блоке.

(Относительный адрес можно посмотреть на экране PG в режиме работы “Блокировка ввода” (ключевой переключатель) или при выводе на печать)

UAMK Слово масок индикации прерываний:
В UAMK собраны все возникшие на данный момент и еще не обработанные причины прерываний (см. “Расположение системных данных” Раздел 8.2.4).

UALM Слово сброса индикации прерываний (см. «Расположение системных данных» Раздел 8.2.4).

DB-ADR Абсолютный начальный адрес последнего вызванного блока данных в памяти программ (DW 0)
(DB-ADR = 0000, если не был вызван DB).

OB-NR Номер последнего вызванного блока данных.

DBL-REG Длина последнего вызванного блока данных.

BA-ADR Абсолютный адрес в памяти программ для следующей обрабатываемой программы в последнем вызванном блоке.

…Nr Тип и номер блока вызванного последним.

AKKU1…4 Содержимое регистра вычислений перед переходом в состояние «СТОП».

При определенных ошибках системная программа при переходе в состояние «СТОП» заносит в АКК 1 и 2 данные об ошибках, которые могут пояснять прерывания.

СКОБКИ Число уровней скобок: «Кex abc»
x = уровень с 1 по 7

a = OR (или, см. биты состояний).
b = VKE (Результат логической операции, см. биты состояний).

c = 1: U(
c = 0: O(

БИТЫ
РЕЗУЛЬТАТА: см. Раздел 3.2

Следующие сокращения представляют важнейшие причины ошибок. Крестом помечаются только те причины прерываний, которые возникли непосредственно в индицируемом уровне обработки программ (смотри УРОВЕНЬ!)

При указании причины ошибок воспроизводятся слова индикации прерывания(UAMK. 16 бит: см. Раздел 8.2.4). Частично данная информация совпадает с управляющими битами.

NAU Сбой напряжения питания в центральном устройстве.

PEU Ошибка периферии = сбой напряжения в устройстве расширения.

BAU Ошибка батареи = сбой напряжения буферной батареи (центральное устройство)

MP – STP Многопроцессорный "СТОП":
а) селектор выбора режимов на координаторе переведен в положение "СТОП" или
б) "СТОП" других процессоров в многопроцессорном режиме

ZYK Превышено время цикла.

QVZ Задержка квитирования при обмене данными с периферией.

ADF Ошибка адресации входов и выходов

STP Останов установкой селектора режимов в STOP.
Останов по команде от PG.
Останов при обработке команд языка STEP5 ‘STP’
Останов от операционной системы, если не запрограммирован организационный блок обработки ошибки.

BCF Ошибка кода команды:
ошибка, которая может быть опознана во время декодирования команд

а) Ошибка замещения:	обрабатываемая команда языка STEP5 не замещаема
б) Ошибка кода операции:	обрабатываемая команда языка STEP5 неверна
с) Ошибка параметра:	неверный параметр, обрабатываемая командой STEP5

S – 6 Не занят

LZF Ошибка выполнения команды:
Ошибка, которая может быть опознана во время выполнения команды

a) вызываемый блок не загружен

b) ошибка передачи блока данных

c) Особые ошибки выполнения

REG Ошибка при обработке структуры регулятора R64 в цикле.

STUEB Переполнение стека блоков (глубина вложения слишком высока, необходим новый пуск)

STUEU Переполнение стека прерываний (глубина вложения слишком высока, необходим новый пуск)

WECK Ошибка "Будильника"
Если перед или во время обработки OB прерывания по времени происходит его повторный вызов.

DOPP Ошибка дублирования:

Активный в данный момент уровень обработки программ (ADF, BCE, LZF, Q\/Z, REG, ZYK) активируется во второй раз (необходим новый пуск).

Примеры обработки USTACK

На следующем изображении вы видите структуру USTACK в зависимости от возникающих прерываний.

1. Уровень обработки программы «ЦИКЛ» (OB1) прерван появлением прерывания по времени.
2 При этом активизируется уровень обработки программ «ПРЕРЫВАНИЕ ПО ВРЕМЕНИ» и обрабатывается OB13
3 Из-за возникновения прерывания от процесса происходит выход из уровня «ПРЕРЫВАНИЕ ОТ ПРОЦЕССА» и обрабатывается OB2
4 Неверная команда адресации ведет к тому, что активизируется уровень “ADF” и на нем обрабатываетсяOB25. В его программе обработки ошибок пользователь запрограммировал команду останова (STP): Процессор прервал обработку программы.

Перед завершающим переходом в состояние останова было прервано 4 различных уровня обработки программ. Если вы теперь вызовите на экран PG USTACK, то вы получите четырехуровневый USTACK с глубиной вложения 01, в котором содержатся метки уровня обработки программ прерванного последним (=ADF). Вы можете теперь «погружаться» в USTACK пока не достигните USTACK с глубиной вложения 04, который содержит уровень обработки программ ЦИКЛ, прерванный первым.

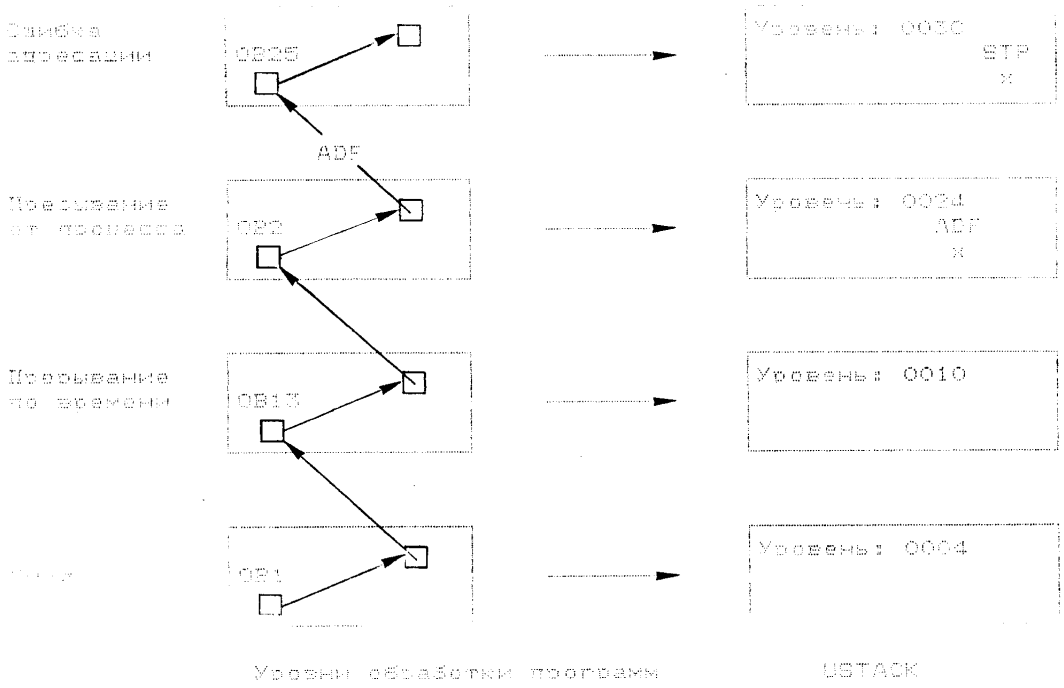
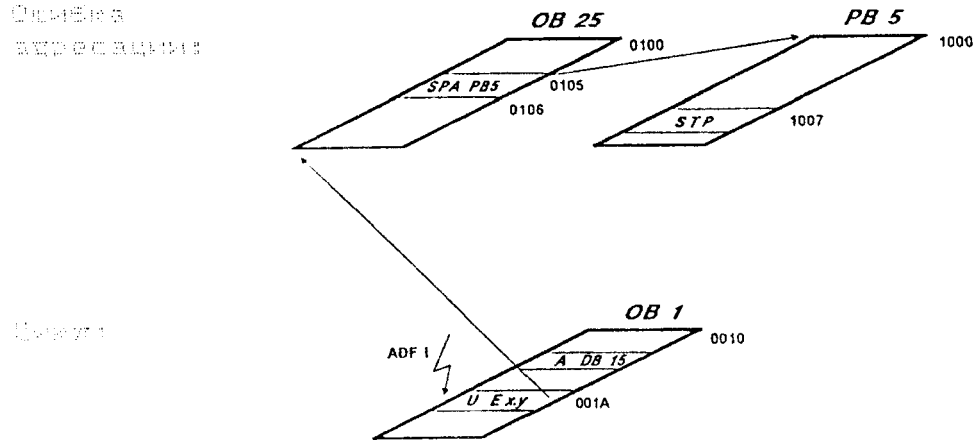


Рис. 5-1: <u>Пример организации USTACK</u>

В данном примере процессор при выполнении команды 'U E x. y" в OB1 распознает ошибку адресации. Это приводит к обработке 0B 25. При обработке команды STP в PB5 процессор переходит в СТОП.
Два прерванных уровня обработки программ создают двухуровневый USTACK:



USTACK							
TIEFE:	02						
BEF-REG	U S x.y	SAZ:	001A	DB-ADR:		BA-ADR:	0000
BST-STP:	1	FB-NR:	1	DB-NR:	16	-NR:	
		REL-ADR:	000A	DBL-REG:			
EBENE:	0004	UAMK:	0300	UALW:	0000		
AKKU 1:							
БИЫ РЕЗУЛЬТАТА:	.						
ПРИЧИНА ОШИБКИ: . . .							ADF
							X

USTACK							
TIEFE:	01						
BEF-REG:	STP	SAZ:	1007	DB-ADR:		BA-ADR:	0106
BST-STP:	3	FB-NR:	5	DB-NR:	16	-NR:	25
		REL-ADR:	0007	DBL-REG:			
EBENE:	0030	UAMK:	0300	UALW:	0000		
AKKU 1:							
БИТЫ РЕЗУЛЬТАТА:	. . . .						
ПРИЧИНА ОШИБКИ:							STP
							X

5.4. Обработка ошибок с помощью организационных блоков

Если операционная система опознает определенную ошибку, то она вызывает предусмотренный для этого случая организационный блок. С помощью соответствующей программы организационного блока Вы можете теперь установить дальнейшие действия процессора.

В зависимости от того, как запрограммирован организационный блок, ВЫ можете

- продолжить нормальную обработку программы.
- перевести процессор в "СТОП"

  или/и
- запустить на выполнение специальную программу обработки ошибки.

Для указанных причин ошибок предусмотрены следующие организационные блоки:

Причина ошибки	Вызов	Реакция на незапрограммированный ОВ
Вызов незагруженного блока (LZF)	ОВ 19	СТОП
Задержка квитирования в программе пользователя при обращении к модулям периферии (QVZ).	ОВ 23	Нет реакции
Задержка квитирования при обновлении области отображения процесса и передаче маркеров связи (QWZ)	ОВ 24	Нет реакции
Ошибка адресации (ADF)	ОВ 25	СТОП
Превышение времени цикла (ZYK)	ОВ 26	СТОП
Ошибка подстановки (BCF)	ОВ 27	СТОП
Селектор режима работ в состоянии СТОП, функция PG 'AG-СТОП', СТОП по шине S5 (в многопроцессорном режиме)	ОВ 28	СТОП
Ошибка кода операции (BCF)	ОВ 29	СТОП
Ошибка параметрирования (BCF)	ОВ 30	СТОП
Особые ошибки выполнения (LZF)	ОВ 31	СТОП
Ошибка передачи в блок данных (LZF)	ОВ 32	СТОП
Ошибка будильника (WECK-FE)	ОВ 33	СТОП
Ошибка при обработке структуры регулятора R64 (REG-FE)	ОВ 34	СТОП

Реакция при незапрограммированных организационных блоках зависит от ошибки:

а) Нет прерываний циклической обработки программы

Если возникает задержка квитирования и OB 23 или OB 24 не загружены, то циклическая обработка программы не прерывается.
Это означает, что реакция процессора отсутствует.

Если процессор при QVZ должен перейти в состояние «СТОП», то необходимо создать организационный блок, занести в него инструкцию «СТОП» и завершить его командой BE.

Программа для останова:
:
:
:STP
:BE

б) Состояние останова.

Во всех остальных случаях ошибки процессор немедленно переходит в «СТОП», если соответствующий организационный блок не был запрограммирован пользователем.

Если нужно в исключительных случаях (например, во время пуска-наладки) не учитывать одну или несколько ошибок во время циклической обработки программы, то достаточно в соответствующий организационный блок внести инструкцию конца блока.

Программа для работы без прерывания:

:
:
:BE

ВАЖНО!
Исключение составляет организационный блок OB 28: Здесь всегда происходит переход в состояние останова независимо от того, был ли запрограммирован OB 28 или нет.

Если Вы не хотите программировать соответствующий организационный блок, то имеете возможность, запрограммировав соответствующим образом блок данных DX 0, исключить переход процессора в состояние «СТОП».

Прерывание при обработке организационных блоков ошибки

После того, как операционная система вызвала соответствующий организационный блок. Имеющаяся в нем программа пользователя начинает обрабатываться.

Если во время обработки организационного блока заново появляется ошибка как и при циклической обработке программы, программа прерывается на следующей границе команды, и вызывается соответствующий организационный блок.

Организационные блоки обрабатываются в той последовательности, в которой они были вызваны. Сколько организационных блоков ошибки может быть вложено один в другой зависит от:

а) вида возникшей ошибки:
Не могут вызываться один из другого организационные блоки, которые принадлежат одному и тому же уровню обработки программ! (О принадлежности ОВ-ошибок к уровням обработки программ см. следующий Раздел)

При обработке ОВ 23 (уровень обработки программ BCF), можно, например, вызвать ОВ 32 (уровень обработки программ LZF), но нельзя вызвать ОВ 29 или ОВ 30 (потому что они тоже принадлежат к уровню обработки программ BCF).

При повторном вызов одного и того же уровня обработки программ процессор немедленно переходит в «СТОП».

б) количества в данный момент активизированных уровням обработки программ:

Для каждого активизированного уровням обработки программ необходимо, чтобы операционная система при прерывании отвела соответствующее место в памяти USTACK. Если этого места не достаточно, то возникает переполнение USTACK.

При переполнении USTACKа процессор немедленно переходит в «СТОП».

в) количество вызванных в данный момент времени блоков:

При переполнении USTACKа процессор немедленно переходит в «СТОП».

5.5. Ошибки при ЗАПУСКЕ

При инициализации и при запуске возникшие причины ошибок и сбоев могут привести к тому, что программа запуска будет прервана и процессор перейдет в СТОП.

В программе запуска (организационные блоки ОВ 20, ОВ21 и ОВ 22) возникшие причины прерывания обрабатываются в цикле.

Исключение: при останове в процессе запуска не вызывается организационный блок ОВ 28

Возможные причины прерываний и ошибок без соответствующих организационных блоков обработки ошибок)

STP:
Команда останова от операционной системы (при FE STP) или в программе пользователя

BAU:
Мало напряжение буферной батареи центрального устройства

NAU:
Сбой напряжения питания в центральном устройстве

PEU:
Сбой напряжения питания в устройстве расширения

STUEU:
Переполнение стеков прерывания (U-STECK)

STUEB:
Переполнение стека блока (B-STECK) при слишком большой глубине вложения.

DOPP:
Повторный вызов уровня обработки программы ошибок (об ошибке дублирования см. пример на странице 4-4)

RAM-FE:
Ошибка при инициализации: RAM операционной системы или DB—RAM содержит ошибку.

MOD-FE:
Ошибка при инициализации:: в содержимом модуля пользователя (модуль RAM или EPPOM) ошибка

DB0-FE:
Ошибка при создании списка адресов блоков (блок данных DB0)

DB1 -FE:
Ошибка при обработке DB 1 при создании списка адресов для обновления области отображения процесса.

DB2-FE:
Ошибка при обработке DB 1 при создании списка адресов для обновления области отображения процесса.

DX0-FE:
Ошибка при обработке блока данных DX 0 (DB0-FE. DB1-FE, DB 2-FE и DX0-FE: см. следующие страницы!).

5.5.1. DB0-FE: (ошибки DB0)

Ошибка при создании списка адресов блоков (блок данных DB0)

DB0 создается операционной системой после включения напряжения при ошибке DB0 вы найдете в системных словах данных BS 3 и BS 4 значения ошибки или метки ошибок, которые позволят определить возникшую ошибку.
Абсолютный адрес памяти: BS 3 KH = EA 03
BS 4 KH = EA 04
(указание по обработке информации об ошибках в словах системных данных BS 3 и BS4 вы найдете в разделе 5.2)

Метка BS 3	ошибки BS 4	Пояснение
8001H	yyyyH	Неверная длина блока yyyy = адрес блока с неверной длиной
8002H	yyyyH	Вычисленный конечный адрес блоков памяти не верен yyyy = адрес блока
8003H	yyyyH	Недействительная метка блока yyyy = адрес блока с неверной длиной
8004H	yyyyH	Слишком большой номер организационного блока (допустим OB1 по OB 39) yyyy = адрес блока с неверной длиной
8005H	yyyyH	Номер блока данных – 0 (допустимо: DB1…DB255) yyyy = адрес блока с неверной длиной

5.5.2. DB1 – FE: (ошибки DB1):

Ошибка в DB1 при создании списка адресов для обновления области отображения процесса

- ошибка DB1 в многопроцессорном режиме или
- неверный список адресов вDB1 при новом пуске

При ошибке DB1 вы также найдете в системных словах данных BS3 и BS4 метки ошибок, которые позволят определить возникшую ошибку.

Абсолютный адрес памяти: BS 3 KH = EA 03
BS 4 KH = EA 04

(Указания по обработке информации об ошибках и системных словах данных BS 3 и BS 4 Вы найдете в Разделе 5.2.)

Метки BS 3	ошибок BS 4	Пояснение
0410H	yyyyH	Недействительная метка 1.Метка заголовка отсутствует или не верна (корректно КС MASK01) 2.Недопустимая метка (допустимо КН DE00. DA00, CA00, BB00) 3. Метка окончания отсутствует или не верна (корректно КЕ EEEE) yyyyH = неверная метка
0411H	yyyyH	Неверный параметр в списке адресов «цифровых входов» (допустимо 0…128) yyyy = адрес неверно указанного байта входа
0412H	yyyyH	Неверный параметр в списке адресов «цифровых выходов» (допустимо 0…128) yyyy = адрес неверно указанного байта выхода
0413H	yyyyH	Неверный параметр в списке адресов «входных маркеров связи» (допустимо 0…256) yyyy = адрес неверно указанного маркерного байта
0414H	yyyyH	Неверный параметр в списке адресов «выходных маркеров связи» (допустимо 0…256) yyyy = адрес неверно указанного маркерного
0415H	yyyyH	Недействительное количество таймеров (допустимо 256) yyyy = неверное количество таймеров
0419H	yyyyH	Задержка квитирования от цифровых входов yyyy = адрес неквитируемого байта входов
041AH	yyyyH	Задержка квитирования от цифровых выходов yyyy = адрес неквитируемого байта выходов
041BH	yyyyH	Задержка квитирования от входных маркеров связи yyyy = адрес неквитируемого маркерного байта
041CH	yyyyH	Задержка квитирования от выходных маркеров связи yyyy = адрес неквитируемого маркерного байта

5.5.3. DB2-FE: (ошибки DB2):

Ошибка при обработке блока данных параметров DB2 структуры регулятора R64 (инициализация регулятора)

При ошибке DB2 Вы найдете в системных словах BS 3 и BS 4 метку ошибки, которая позволит определить возникшую ошибку.

Абсолютный адрес памяти: BS 3 КН = EA03
BS 4 КН = EA04

(Указание по обработке информации об ошибках в системных словах BS 3 и BS 4 Вы найдете в Разделе 5.2.)

Метки ошибок BS 3 BS 4		Пояснение
0421H	DByyH	Блок данных не загружен yy = номер незагруженного блока данных
0422H	FByyH	Функциональный блок не загружен yy = номер незагруженного функционального блока
0423H	FByyH	Функциональный блок не опознан yy = номер неопознанного функционального блока
0424H	FByyH	Функциональный блок загружен с неверным программным обеспечением PG yy = номер функционального блока
0425H	DByyH	Неверная длина блока данных регулятора yy = номер блока данных
0426H	-	Для переноса DB – регулятора из EPROMa пользователя в DB-RAM недостаточно места в памяти DB-RAM.

5.5.4. DX0-FE (ошибки DX0):

Ошибка при обработке блока данных DX0

При ошибке DX0 Вы найдете в системных словах BS 3 и BS 4 метку ошибки, которая позволит определить возникшую ошибку.

Абсолютный адрес памяти : BS 3 KH=EA03
BS 4 KH=EA04

(указание по обработке информации об ошибках в системных словах BS 3 и BS 4 Вы найдете в Разделе 3.5.2.)

Метки ошибок BS 3 BS 4	Пояснение
0431H yyyyH	Недопустимая метка блока 1. Метка заголовка отсутствует или не верна корректно КС MASKXO) 2. Недопустимая метка блока 3. Метка окончания отсутствует или не верна (корректно KH EEEE) yyyy = неверная метка
0432H yyyyH	Неизвестный параметр yyyy = неверный параметр
0433H yyyyH	Недопустимый параметр yyyy = неверный параметр
0434H yyyyH	Недопустимое количество таймеров (допустимо : 0… 256) yyyy = неверное число таймеров
0435H yyyyH	Недопустимое время цикла (допустимо: от 1 мс до 6000 мс) yyyy = неверное время

5.6. Ошибки в режимах ОБРАБОТКА и ЗАПУСК

В режиме работы ОБРАБОТКА при возникновении определенной причины ошибки может быть прервана на границе команды циклическая обработка программы, обработка прерывания по времени или обработка прерывания от процесса или обработка регулятора.

При инициализации и в режиме работы ЗАПУСК появление причин прерывания ведет к тому, что выполнение программы запуска прерывается и процессор переходит в стоп или вызывается организационный блок, предусмотренный для данного типа ошибок. В программе запуска возникающие причины прерываний обрабатываются как и в цикле.

Различают причины прерывания, которые непосредственно переводят процессор в «СТОП» и причины прерываний, при появлении которых операционная система перед переходом в состояние «СТОП» вызывает определенный организационный блок, который может быть запрограммирован пользователем.

Возможные причины прерываний и ошибок без соответствующих ОВ обработки ошибок.

STP:
Команда останова от операционной системы (при аппаратной ошибке) или в программе пользователя.

BAU:
Сбой напряжения буферной батареи в центральном устройстве.

NAU:
Сбой напряжения питания в центральном устройстве.

PEU:
Сбой напряжения питания в устройстве расширения.

STUEU:
Переполнение стека прерывания (U- СТЕК).

STUEB:
Переполнение стека блоков при слишком большой глубине вложения.

DOPP:
Повторный вызов того же уровня обработки программы (об ошибке дублирования см. Пример на стр. 4.4).

При всех вышеназванных ошибках происходит немедленный переход в состояние «СТОП» . При этом выдается U-СТЕК, в котором индицируется возникшая ошибка.

(Указания для обработки U-СТЕКа Вы найдете в Разделе 5.3)

Возможные причины прерываний и ошибок с соответствующими ОВ обработки ошибок в режимах ОБРАБОТКА и ЗАПУСК.

BCF:		
Ошибка кода команд	1. Ошибка замещения	OB 27
	2. Ошибка кода операции	OB 29
	3. Ошибка параметрирования	OB 30
LZF:		
Ошибка выполнения	1. Вызов незагруженного блока	OB 19
	2. Ошибка передачи в DB	OB 32
	3. Специальные ошибки выполнения	OB 31
ADF:		
Ошибка адресации		OB 25
QVZ:		
Задержка квитирования	1. В программе пользователя при обращении к модулям периферии	OB 23
	2. При обновлении области отображения процесса	OB 24

Возможные причины прерываний и ошибок с соответствующими ОВ обработки ошибок только в режиме ОБРАБОТКА.

ZYK-FE:	
Ошибка цикла.	OB 26
WECK-FE:	
Ошибка « будильника «.	OB 31
REG-FE:	
Ошибка регулятора.	OB 34
ABBR:	
Прерывание.	OB 28

В следующей главе все эти причины ошибок описываются более подробно.

5.6.1 BCF (ошибка кода команды)

Ошибка кодов команд возникает в том случае, если процессор не может интерпретировать команду языка STEP 5 в программе пользователя или ее выполнить. Все допустимые коды команд собраны в приложении.

Команда, которая явилась причиной соответствующей ошибки кодов команд, не выполняется. Если организованный блок BCF запрограммирован, то он вызывается, обрабатывается, а затем продолжается обработка прерванной программы пользователя со следующей команды. При незапрограммированном BCF OB процессор переходит в стоп.

Различают следующие ошибки кодов команд :

A) BCF = Ошибка замещения.

Если в функциональном блоке должна выполняться операция с формальным операндом, то процессор вставляет при обработке программы пользователя на место этого формального операнда фактический операнд, стоящий в вызове функционального блока.

Процессор опознает недопустимую подстановку. Операционная система при этом прерывает обработку программы пользователя и вызывает организованный блок OB 27.

В АКК 1 при этом находится дополнительная информация, которая поясняет возникшую ошибку.

Метка ошибки АКК 1 АКК 2	Пояснение
1801H -	Ошибка замещения по команде BBS
1802H -	Ошибка замещения по команде BDW. BMW
1803H -	Ошибка замещения по команде B=, B1=
1804H -	Ошибка замещения по команде L=. T=
1805H -	Ошибка замещения по командам U=, UN=, O=, ON=, = =, S= RB= .

b) Ошибка кодов операций

Недопустимый код операции возникает, если была запрограммирована команда, которая не входит в объем команд STEP 5 процессора (например: можно запрограммировать с помощью PG команды RU или SU, однако R- и S- процессорами и CPU 928 в AG 135U они не интерпретируются.)

При опознавании недопустимого кода операции на этом месте обработка программы пользователя прерывается, и вызывается организационный блок OB29.

При вызове OB29 в АКК 1 находится дополнительная информация, которая поясняет возникшую ошибку.

Ошибка кодов команд НЕ ДОЛЖНА квитироваться : процессор не опознает, является ли ошибочная команда однословной или многословной командой. После того, как процессор обрабатывает OB29 он отследит, чтобы программа продолжилась, начиная со следующего командного слова. При этом, если это окажется вторым словом многословной команды, то он либо опознает новую ошибку когда команды либо выполнит это слово как действительную команду.

Метка ошибки АКК 1	АКК 2	Пояснение
1811H	-	Команда с недопустимым кодом операции
1812H	-	Недопустимый код операции по команде, в которой старший байт первого слова команды содержит значение 68H.
1813H	-	Недопустимый код операции по команде, в которой старший байт первого слова команды содержит значение 78H.
1814H	-	Недопустимый код операции по команде, в которой старший байт первого слова команды содержит значение 70H.
1815H	-	Недопустимый код операции по команде, в которой старший байт первого слова команды содержит значение 60H.

c) BCF = Ошибка параметрирования

Недопустимый параметр появляется, если была запрограммирована команда с параметром, который не допустим для данного процессора (например, вызов зарезервированного блока данных) или если вызвана отсутствующая специальная функция.

Если процессор опознал недопустимый параметр, то операционная система прерывает обработку программы пользователя и вызывает организационный блок OB30.

Если OB30 не запрограммирован, то процессор переходит в «СТОП».

При вызове OB30 в АКК 1 находится дополнительная информация, поясняющая ошибку.

Метка ошибки АКК 1 АКК 2	Пояснение
	Недопустимый параметр при:
1821H -	A BD 0, 1, 2
182BH -	SPA(B) OB 0
182CH -	SPA(B) OB > 39: Отсутствует специальная функция
182DH -	AX DX0
182EH -	L MW/T MW/L PW/T PW/L QW/T QW/L DD/T DD/B MW 255
182FH -	L EW/T EW/L AW/T AW 127
1830H -	L MD/T MD 253, 254, 255
1831H -	L ED/T ED/L AD/T AD 125. 126. 127
1832H -	RLD/RRD/SVD/SLD 33-255
1833H -	SLW/SRW/LIR/TIR 16-255
1834H -	SES/SEF 32-255
1835H -	U=/UN=/0=/ON=/S=/RB=/==/RD=/FR=/ SI=/SE=/SVZ=/SSV=/SAR=/L=/LC=/LW=/T= 0.127-255
1836H -	B=/LD= 0. 126-255
1837H -	MBR с константой > 0FFFFH ($2^{16} – 2^{17}$ 0)

5.6.2. LZF (ошибка выполнения команды)

Ошибка выполнения возникает в случае если, процессор во время обработки команды языка STEP5 опознает ошибку.

Команда, которая явилась причиной соответствующей ошибки выполнения, не обрабатывается. Если программа в вызванном организационном блоке обработана, то выполнение программы пользователя продолжается с команды, следующей после прерывания. Если LZF-OB не запрограммирован, то процессор переходит в “СТОП”.

Различают следующие ошибки выполнения:

а) LZF = Вызов нагруженного блока

Если в программе пользователя вызван блок, который отсутствует, то операционная система опознает ошибку. Это действительно для всех видов блоков, а также для условных и безусловных вызовов.

Если опознан вызов незагруженного блока, то операционная система вызывает организационный блок OB19. В OB19 Вы можете определить дальнейшие действия процессора. Если OB19 запрограммирован, то он вызывается, а после окончания его обработки продолжается обработка прерванной программы STEP5 со следующей команды. Если OB19 не запрограммирован, то процессор при вызове незагруженного блока переходит в “СТОП”.

При вызове OB19 в АКК 1 находится дополнительная информация, поясняющая возникшую ошибку.

Метка ошибки АКК 1	АКК 2	Пояснение
1A01H	-	Нет загруженного блока данных при ADB
1A02H	-	Нет загруженного блока данных при AXDX
1A03H	-	Нет загруженного блока при SPA (B) FB, OB, PB, SB
1A04H	-	Нет загруженного блока данных при BA(B) FX
1A05H	-	Нет загруженного блока данных при OB 254 или 255

b)) LZF = Ошибка передачи

При передаче данных в блок данных (DB, DX) процессор сравнивает длину вызванного DB с параметром указанном в команде передачи. Если указанный параметр превышает длину блока данных, то инструкция передачи не выполняется, чтобы не переписывать недействительные данные в память.

Ошибка передачи определяется также в том случае, если должен быть выполнен опрос или изменение отдельного бита внутри несуществующего слова данных.

Ошибка передачи, кроме того, определяется, если должно состояться обращение к слову данных, прежде чем вызван сам блок данных (по команде A DBN или AX DXN).

При определении ошибки передачи операционная система вызывает организационный блок OB32. Команда, которая явилась причиной ошибки передачи, больше не обрабатывается. Если OB32 не запрограммирован, то процессор переходит в состояние «СТОП».

При вызове OB32 в АКК 1 находится дополнительная информация, которая поясняет возникшую ошибку.

Метка ошибки АКК 1	АКК 2	Пояснение
1A11H	-	Обращение к неопределенному слову данных с помощью команд U/UN D, O/ON D, S/R D, =D
1A12H	-	Ошибка передачи по команде TDR в неопределенное слово данных
1A13H	-	Ошибка передачи по команде TDL в неопределенное слово данных
1A14H	-	Ошибка передачи по команде TDW в неопределенное слово данных
1A15H	-	Ошибка передачи по команде TDD в неопределенное слово данных

c) Особые ошибки выполнения

К ним принадлежат все ошибки выполнения, которые нельзя отнести к выше описанным видам ошибок выполнения (ошибкам передачи или вызывам незагруженных блоков).

При определении таких ошибок выполнения операционная система вызывает организационный блок OB31. Команда, явившаяся причиной ошибки (например, специальные функции) дальше не обрабатывается. Если OB31 не запрограммирован, то процессор переходит в «СТОП». Если при возникновении данной ошибки нужна дальнейшая обработка, то в программу OB31 достаточно ввести инструкцию завершения блока BE.

При вызове OB1 в АКК 1 находится информация, которая поясняет возникшую ошибку:

Метка ошибки АКК 1	АКК 2	Пояснение
1A21H	-	Ошибка при EDB, EXDX: Блок данных уже существует
1A22H	-	Ошибка при EDB, EXDX: Недопустимая длина блока данных (> 4 x 2^{10} слов)
1A23H	-	Ошибка при EDB, EXDX: Недостаточно места в памяти RAM.
1A25H	-	Ошибка при B1 = : Недопустимый параметр в АКК 1 (< 1 или >255)
1A29H	-	Недостаток или избыток стека скобок по командам U(, O(,).
1A2AH	-	Ошибка при A DB или AX DX: длина блока в заголовке блока данных слишком мала (длина 5 слов)
1A2BH	-	Функциональный блок загружен с использованием неверного программного обеспечения PG.
1A2CH	-	Ошибка при ACR : Недопустимый номер буфера «Кафель»
1A31H	-	Ошибка специальной функции в OB254 или OB255 (копирование): Блок данных назначения уже имеется в DB RAM.
1A32H	-	Ошибка специальной функции в OB254 или OB255 (дублирование): Блок данных назначения уже имеется в DB RAM.
1A33H	-	Ошибка специальной функции в OB254 или OB255: Недостаточно места в памяти DB RAM.
1A3AH	-	Ошибка специальной функции в OB221: Недопустимое значение нового времени цикла (время цикла < 1 мс или > 6000 миллисекунд).

Метка ошибки АКК 1	АКК 2	Пояснение
1A3BH	-	Ошибка специальной функции в OB223: типы запусков разных процессоров в многопроцессорном режиме различны.
1A41H	-	Ошибка специальной функции в OB240, OB241 или OB242: Недопустимый номер регистра сдвига или блока данных (No < 192).
1A42H	-	Ошибка специальной функции при OB241: Регистр сдвига не инициализирован.
1A43H	-	Ошибка специальной функции при OB240 : Недостаточно места в DB памяти RAM.
1A44H	-	Ошибка специальной функции в OB240: Слово данных DW 0 не заполнено нулями.
1A45H	-	Ошибка специальной функции в OB240: Недопустимая длина регистра сдвига в DW 1 (не находится в пределах между 2 и 256).
1A46H	-	Ошибка специальной функции в OB240: Недопустимая позиция указателя или количества указателей.
1A47H	-	Ошибка специальной функции при OB120
1A48H	-	Ошибка специальной функции при OB122
1A49H	-	Ошибка специальной функции при OB110
1A50H	-	Ошибка при LRW, TRW: вычисленный адрес памяти <BS + константа> лежит вне 0 - EDFEH.
1A51H	-	Ошибка при LRD, TRD: вычисленный адрес памяти <BS + константа> лежит вне 0 - EDFEH.
1A52H	-	Ошибка при TSG, LB GB, LW GW, TB GB, TW GW: вычисленный адрес памяти <BS + константа> лежит вне 0 - EDFEH.
1A53H	-	Ошибка при LB GW, LW GD, TB GW, TW GD: вычисленный адрес памяти <BS + константа> лежит вне 0 - EDFEH.
1A54H	-	Ошибка при LB GD, TB GD: вычисленный адрес памяти <BS + константа> лежит вне 0 - EDFEH.
1A55H	-	Ошибка при TSC, LB CB, LW CW, TB CB, TW CW: вычисленный адрес памяти <BS + константа> лежит вне F400H - EDFEH.
1A56H	-	Ошибка при LB CW, LW CD, TB CW, TW CD: вычисленный адрес памяти <BS + константа> лежит вне F400H - EDFEH.

Метка ошибки АКК 1	АКК 2	Пояснение
1A57H	-	Ошибка при LB CD, TB CD: вычисленный адрес памяти <BS + константа> лежит вне F400H - EDFEH.
1A58H	-	Ошибка при TNW, TNB: блок источника не полностью находится в данной области: 0000 – 7FFF Память пользователя 8000 – DD7F RAM – блоков данных DD80 – EDFF Системная RAM (DB0, BA, BB, BS, BT, T, Z) TT00 – EFFF Маркер. Область отображения процесса F000 – FFFF Периферия
1A59H	-	Ошибка при TNW, TNB: блок приемника не полностью находится в данной области: 0000 – 7FFF Память пользователя 8000 – DD7F RAM – блоков данных DD80 – EDFF Системная RAM (DB0, BA, BB, BS, BT, T, Z) TT00 – EFFF Маркер. Область отображения процесса F000 – FFFF Периферия

5.6.3. ADF (ошибка адресации)

Ошибка адресации возникает. Если по команде языка STEP5 опрашиваются входы или выходы отображения процесса, которые на момент последнего нового пуска не были определены в модулях периферии (модуль не установлен, неисправен или не указан в блоке данных DB1 процессора).

Операционная система прерывает обработку программы пользователя и вызывает организационный блок OB 25. После обработки запрограммированной в OB 25 программы продолжается обработка программы пользователя со следующей команды после прерывания, т.е. инструкция STEP5, которая явилась причиной ошибки ADF не выполняется.

Если OB 25 не запрограммирован, процессор при возникновении ошибки адресации переходит в «СТОП», если только Вы не установите для этого случая в блоке данных DXO параметр продолжения обработки программы.

Контроль ошибки адресации может быть полностью подавлен путем соответствующего параметрирования блока DX 0.

При возникновении ошибки адресации в АКК 1 или АКК 2 метка ошибки не передается!

5.6.4. QVZ (задержка квитирования)

Задержка квитирования возникает, если от модуля входов или выходов. К которым производились обращения в течении определенного времени, не пришел сигнал RDY (готов). Причиной задержки квитирования может быть дефект модуля либо удаление модуля во время работы.

Следующие ошибки задержки квитирования прерывают обработку программы пользователя и вызывают соответствующий организационный блок:
1.Задержка квитирования в программе пользователя при непосредственном обращении по шине S5 к CP, IP, KOR или к модулям периферии (например, по командам загрузки передачи L/T P…или Q….):
Операционная система вызывает организационный блок OB23.

В АКК 1 и АКК 2 находится дополнительная информация, которая поясняет возникшую ошибку.

Метка ошибки АКК 1	АКК 2	Пояснение
1E23H	yyyyH	Задержка квитирования QVZ в программе пользователя при обращении к периферии yyyy = адрес QVZ

2.Задержка квитирования при обновлении области отображения процесса входов и выходов и при передаче маркеров связи:
Операционная система вызывает организационный блок OB 24.

В АКК 1и 2 находится дополнительная информация, которая поясняет возникшую ошибку.

Метка ошибки АКК 1	АКК 2	Пояснение
1E25H	yyyyH	Задержка квитирования при обновлении цифровых выходов yyyyH = адрес неквитированного байта выхода
1E26H	yyyyH	Задержка квитирования при обновлении цифровых входов yyyyH = адрес неквитированного байта входа
1E27H	yyyyH	Задержка квитирования при обновлении выходных маркеров связи yyyyH = адрес неквитированного маркерного байта
1E28H	yyyyH	Задержка квитирования при обновлении входных маркеров связи yyyyH = адрес неквитированного маркерного байта

Если вызванный организационный блок не запрограммирован, то обработка программы пользователя продолжается. Однако задержка квитирования удлиняет время выполнения команды языка STEP5, являющейся причиной данной ошибки.

При возникновении задержки квитирования процессор считывает в качестве замены значения «ООН» и работает в дальнейшем с этим значением, если QVZ подтвержден пользователем.

Если задержка квитирования должна переводить процессор в «СТОП», то необходимо в ОВ 23 или ОВ 24 запрограммировать команду останова STP.

С помощью соответствующего параметрирования DX0 Вы можете в случае QVZ также при незапрограммированных ОВ 23 и ОВ 24 вызвать останов системы.

5.6.5. ZYK – FE (Ошибка времени цикла)

Время цикла состоит из общего времени обработки циклических программ. Превышение установленного в процессоре времени цикла может возникнуть, например, при неправильном программировании из-за бесконечного цикла программы в функциональном блоке при неисправности тактового генератора или перегрузке процессора, как например, обновление области отображения процесса в совокупности с большой программой.

Если возникло превышение времени цикла, то операционная система прерывает обработку программы пользователя и вызывает организационный блок ОВ 26. Время цикла при этом запускается заново (тригируется). Если время цикла запускается заново, прежде чем ОВ 26 будет обработан до конца, то процессор переходит в состояние останова с ошибкой дублирования.

Время цикла может меняться (от 1 до 6000 мсек) и может сбрасываться (см. выше). Независимо от времени цикла через 150 мсек работы выдается сигнал BASP, если ОВ 26 к этому моменту еще не обработан до конца.

Если ОВ 26 не запрограммирован. то процессор переходит в состояние останова, если только Вы не изменили предварительную установку в DXO.

Пользователь может индивидуально задать время цикла либо путем параметрирования DXO, либо с помощью вызова организационного блока специальной функции ОА 221.

В циклической программе можно перезапустить контроль времени цикла с помощью вызова организационного блока специальной функции ОВ 222 перед тригированием.

При возникновении ошибки времени цикла в АККу 1 или АККу 2 метка ошибки не передается!

5.6.6. WECK-FE (ошибка будильника)

Если для определенного OB прерывания по времени возникает новый запрос, прежде чем был полностью обработан предыдущий запрос, то операционная система опознает ошибку «Будильника» и вызывает организационный блок OB 33. Обратите внимание в связи с этим на главу 4.4.2 «прерывание по времени»!

В АКК 1 и 2 операционная система заносит дополнительную информацию, поясняющую возникшую ошибку:

Метка АКК 1	ошибки АКК 2	Пояснения
1001H	0016H	Ошибка «будильника» – блок OB 10 (10мс)
1001H	0014H	Ошибка «будильника» – блок OB 11 (20мс)
1001H	0012H	Ошибка «будильника» – блок OB 12 (50мс)
1001H	0010H	Ошибка «будильника» – блок OB 13 (100мс)
1001H	000EH	Ошибка «будильника» – блок OB 14 (200мс)
1001H	000CH	Ошибка «будильника» – блок OB 15 (500мс)
1001H	000AH	Ошибка «будильника» – блок OB 16 (1с)
1001H	000BH	Ошибка «будильника» – блок OB 17 (2с)
1001H	0006H	Ошибка «будильника» – блок OB 18 (5с)

Указание: Метка в АКК 2 является меткой уровня прерывания по времени, явившегося причиной ошибки.

Если OB 33 не запрограммирован, то процессор переходит в «СТОП». При соответствующем параметрировании DX0 Вы можете определить дальнейшую обработку программы при возникшей ошибке будильника в незапрограммированном OB 33.

Обратите внимание, что новый вызов уже активного уровня обработки программы ошибки «Будильника» не приводит к ошибке дублирования (DOPP)!

5.6.7. REG-FE (ошибка регулятора).

Ошибка при обработке стандартного функционального блока структуры регулятора R 64, поддерживаемого операционной системой, опознается как ошибка регулятора.

Важно!

В то время как ошибка будильника операционной системой опознается всегда в том случае, если уровень прерывания по времени начался не внутри его временного растра (напр. В OB 13 внутри 100мс) и обрабатывается до конца (см. выше), неправильная обработка программы регулятора опознается и индуцируется в USTACKe при вызове уровня обработки программы регулятора.

При возникновении ошибки регулятора уровень обработки программ РЕГУЛИРОВКА покидается и вызывает уровень ОШИБКА РЕГУЛЯТОРА (уровень: 001CH) с помощью организационного блока OB 34.

Дальнейшая реакция процесса зависит от того, что запрограммировано в OB 34:

а) Если OB 34 не запрограммирован, то процессор переходит в «СТОП».

С помощью USTACKa можно определить причину ошибки.

б) Если запрограммирован OB 34, то обрабатывается содержащаяся в нем программа STEP 5 (например, обработка АКК 1 и 2, содержимое которых зависит от возникшей ошибки). После этого продолжается обработка регулятора с места прерывания.

Если ошибка регулятора должна быть игнорирована, то для этого достаточно в OB 34 завести команду завершения блока BE.

Если нужно продолжить обработку регулятора при возникшей ошибке регулятора и при незапрограммированном OB 34, то необходимо соответствующим образом изменить DX0.

При вызове OB 34 в АКК 1 и 2 находится дополнительная информация, которая поясняет возникшую ошибку:

Метка АКК1	Ошибки АКК2	Пояснение
0802H	DbyyH	Ошибка времени опроса yy= номер соответствующего блока данных регулятора
0802H	DbyyH	Блок данных регулятора не загружен yy =номер незагруженного блока данных
0803H	FbyyH	Функциональный блок регулятора не загружен yy = номер незагруженного функционального блока
0804H	FbyyH	Функциональный блок регулятора не опознан yy = номер неопознанного функционального блока
0805H	FbyyH	Функциональный блок регулятора загружен с неверным программным обеспечением PG yy = номер функционального блока
0806H	DbyyH	Ошибка длины блока данных регулятора yy = номер блока данных
0880H	yyyyH	Задержка квитирования (QWZ) во время обработки регулятора

Во всех 7 случаях ошибок на программаторе в управляющих битах крестом помечается метка ошибки REG-FE. Если Вы используете PG без операционной системы S5-DOS, то предпоследняя позиция в нижней строке маски управляющих битов не обозначена, однако помечена крестом. В маске USTACK на уровне РЕГУЛИРОВКА, как причина ошибки крестом помечено сокращение REG.

Ошибка времени опроса

После окончания заданного времени опроса циклическая программа прерывается на следующей границе блока и начинает обработку регулятора. Теперь возможно, что обработка «длинного» блока занимает слишком много времени, и происходит переход в обработку регулятора «исключительный шаг»; это определяет ошибку времени опроса.

Ошибка времени опроса может быть обработана как другие ошибки регулятора (как описано в пунктах а и б) или подавлена с помощью маски. В этом случае при возникновении ошибки времени опроса обработка программы не прерывается.

Обратитесь к описанию «Компактная регулировка в R-процессоре AG 135» C79000-B8500-C365-03!

Ошибки времени опроса можно, вероятно. Избежать, если Вы предварительные установки в DX0 «Обработка прерывания регулятора и от процесса на границе блока» измените на «обработка прерывания регулятора или от процесса на границе команды».

5.6.7. ABBR (прерывание)

Если в режиме работы «RUN» вызывается состояние останова с помощью

а) перевода селектора режимов работ на процессоре из RUN в STOP,

б) On-line–функции «СТОП»,

в) перевода селектора на координаторе в STOP (в многопроцессорном режиме),

то операционная система вызывает OB 28. После обработки OB 28 процессор переходит в состояние останова.

Важно!
Переход в состояние останова происходит независимо от того, запрограммирован OB 28 или нет.

Метки ошибок АКК 1 или АКК 2 не передаются!

6. Интегрированные специальные функции

Операционная система CPU928 предлагает Вам специальные функции, которые Вы можете использовать с помощью условного (BPB Овхх) или безусловного (SPA Овхх) вызова подпрограмм. Для этих функций зарезервированы организационные блоки ОА40…ОВ255.

Эти функции называются интегрированными специальными функциями, т.к. они являются обязательной составной частью операционной системы. Как пользователь, Вы можете только вызывать эти функции, но не можете читать или изменять.

Внимание!
Команды SPA OB>39 действуют не как «обычные» команды вызова блоков. Они не позволяют вложенности прерываний.

Специальные функции с границами псевдокоманд

Некоторые специальные функции, выполняемые сравнительно долго, содержат так называемые границы псевдокоманд, и выполнение функции производится в несколько шагов. Если во время выполнения шагов возникает ошибка (например, ZYK), или прерывания (например, сигнал прерывания по времени или от процесса), то в конце этого шага на псевдогранице команды вызывается соответствующий организационный блок.
Такие функции, содержащие границы псевдокоманд, показаны в нижеследующей таблице.

Обработка ошибок, возникающих в спецфункциях

В зависимости от реакции на ошибки различают две группы спецфункций.

Группа 1:

К группе 1 относятся все функции, для которых в случае возникновения ошибки вызывается определенная подпрограмма реакции на ошибку, в которой Вы можете определить дальнейшую работу процессора.

Если, например, процессор при обработке спецфункций натолкнется на неправильное параметрирование, он распознает ошибку превышения контрольного интервала времени и вызывает ОВ31. Или, например, если вызываемая спецфункция отсутствует. То процессор распознает ошибку кода команды и вызовет ОВ30. В этом случае, если подпрограмма реакции на ошибку ОВ30 или ОВ31 не запрограммирована, то процессор перейдет в состояние останова. В управляющий бит и USTACK (U-СТЕК) заносится «LZF» и “BCF” соответственно. В аккумуляторе содержится признак ошибки, который описывает ближайшую ошибку. В случае если ОВ30 или ОВ31 запрограммированы (и не содержат команду STP), после обработки ОВ30 или ОВ31 программа пользователя продолжит работу со следующей команды. В этом случае содержимое аккумулятора не изменится.

Группа 2:
Некоторые спецфункции используются для обработки ошибок специфических для спецфункций другим способом: они влияют на VKE или ANZ0/ANZ1.

Если при обработке спецфункции происходит ошибка, то в большинстве случаев VKE устанавливается в 1, и поэтому этот результат может быть опрошен с помощью команды перехода по условию BPB.

В этом случае Вы можете проанализировать VKE для выяснения, есть ошибка или нет, и, в зависимости от этого, предусмотреть специальную программу обработки ошибок.

Для некоторых спецфункций признаки результата ANZ0 и ANZ1 изменяются при выполнении спецфункции и, в таком случае, они также могут быть проанализированы.

Обзор: встроенных специальных функций в CPU 928

OB110	Обращение к байту индикации
OB111	Стирание аккумуляторов АКК 1, АКК 2, АКК 3 и АКК 4
OB112	Циклический сдвиг содержимого АКК вперед
OB113	Циклический сдвиг содержимого АКК назад
OB120	вкл. /выключение "блокировка всех прерывании"
OB121	вкл./выкл. «блокировка отдельных прерываний по времени"
OB122	вкл./выключение "Задержка всех прерываний"
OB123	вкл./выкл. "задержка отдельных прерываний по времени"
OB160 . . . OB163	Циклы счета
OB170	Считывание стека блоков BSTACK (В-СТЕК)
OB180	Управляемый доступ к блоку данных
OB181	Тестирование блока данных
OB190.192	Передача маркеров в блок данных
OB191.193	Передача блоков данных в область маркеров
OB200[1], 202[1]	
203,204[1] и 205	Функции многопроцессорной связи
OB2l6...218	Обращение к буферам связи
OB220	Расширение знака
OB221[1]	Установка времени цикла
OB222	Новый запуск времени цикла
OB223	Сравнение видов запуска в многопроцессорном режиме
OB224[2]	Передача маркеров межпроцессорной связи в блок в многопроцессорном режиме
OB226	Чтение слова из системной программы
OB227	Чтение контрольной суммы из системной программы
OB22S	Чтение информации о состоянии уровня обработки программ
OB230[1] - 237[1]	Функции для стандартных функциональных блоков
OB 240	Инициализация регистра сдвига
OB241	Обработка регистра сдвига
OB242	Стирание регистра сдвига
OB250[1]	Инициализация PID-регулятора
OB251[1]	Обработка PID-регулятора
OB254[1] 255[1]	Передача блоков данных (DB/DX) в память DB=RAM

[1] Специальные функции с псевдограницами команд (длительное выполнение)

[2] Специальные функции OB221 и OB224 из соображений совместимости берутся из S-процессора и не должны программироваться в CPU928. Вместо этого эти функции системы должны быть запараментированы в DX0 (смотри раздел 7)

Указание:

При задании параметров спецфункций используются следующие правила обозначений:

АКК 1	:	Аккумулятор 1	32 бита
АКК 1-L	:	Аккумулятор 1, младшее слово	16 бит
АКК 1—LL	:	Аккумулятор 1, младшее слово, младший байт	8 бит
АКК 1-LH	:	Аккумулятор 1, младшее слово, старший байт	8 бит



Указание:

Понятие параметр в дальнейшем будет обозначать все данные, необходимые процессору для правильного выполнения функции. Перед вызовом спецфункции в программе STEP5 эти данные должны быть помещены в аккумулятор или в заданные ячейки памяти.

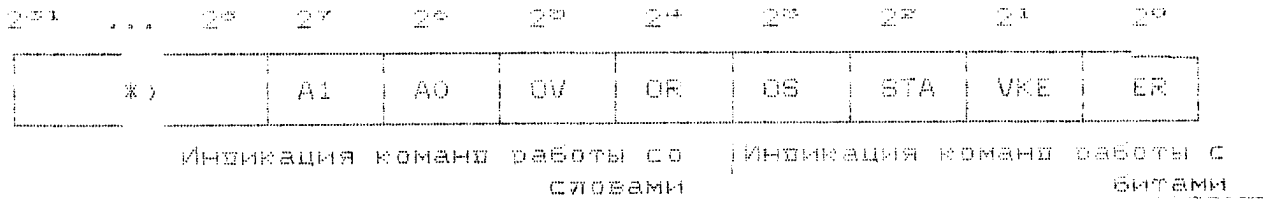
6.1 Работа с регистрами

6.1.1. Обращение к байту индикации (OB110)

Регистр индикации содержит информацию о результатах арифметических или логических операций и может обрабатываться специальными командами, зависящими от байта индикации.

С помощью организационного блока 0B 110 Вы можете загрузить регистр индикации в АККу 1 или записать в него содержимое АККу. Кроме того, Вы можете установить отдельные биты индикации в "0" или "1"

Содержание АККу 1 при обращении к регистру индикации:



*) Биты с 2^5 по 2^{31} резервированы и должны быть равны "0" при заполнении регистра индикации. При считывании содержимого регистра индикации они должны игнорироваться.

Параметры:

1. АККу 2 – L: Метка функции

   возможные значения: 1, 2 или 3

1. АККу 1: новый байт индикации или маска

АККу 2 – L	АККу до	АККу после	функция
1	новый байт индикации	новый байт индикации	Содержимое АККу загружается в регистр индикации
2	маска	новый байт индикации [1)]	Все биты установленные в маске АККу 1 в «1». В АККу 1 загружается новый байт индикации
3	маска	новый байт индикации [1)]	Все биты установленные в маске АККу 1 в «0». В АККу 1 загружается новый байт индикации

[1)] Ограничение:
Биты индикации OR. STA и ER не могут быть считаны, т.к. спец функция 0B 110 влияет на эти биты:
OFR = 0, STA = 1 ER = 0

Случаи ошибок:

-- Метки ошибок в АККу-L не равны 1, 2 или 3.
-- В АККу 1 установлены один из битов с 2^0 по 2^{31}.
В случае ошибки вызывается 0B 31 (особая ошибка выполнения/, а в АККу 1-L загружается метка ошибки 1A49H.

Пример

OB 110 используется как вспомогательное средство тестирования команд, которые обрабатывают и влияют на регистр индикации. Его применение однако не ограничиваются тестом команд. Следующий пример показывает другие случаи применения.
Распределитель вызовов:
В зависимости от содержимого байта маркеров MB0 должна вызываться одна из четырех частей программы. Четырем частях, программы соответствуют биты с M0.0 по M0.3. В любой момент времени можно установить только один из этих битов.

```
      : L  MB 0
      : SLW 4.          :M0.0 по M0.3 сдвигаются на четыре влево.
      : L  KB 1         :загрузка метки функции
      :TAK
      : SPA 0B 110
      : SPS = M000      :переход, если OS = 1
      : SPO = M001      :переход. если OV = 1
      : SPM = M002      :переход. если ANZ0 = 1
      : SPP = M003      :переход.  если ANZ1 = 1
      :  ...            :если ни один бит не установлен
      :…
      :BEA
      :
M000  :  ...            :если M0.0 = 1
      :
      :BEA
M001  :  ...            :если M0.1 = 1
      : …
      :BEA
M002  : …               :если M0.2 = 1
      : …
M003  :   ...           :если M0.3 = 1
      : …
      :BEA
```

6.1.2. Стирание аккумуляторов АКК 1. АКК 2, АКК 3 и АКК 4 (OB111)

Путем однократного вызова организационного блока спецфункции 0B 111 вы можете стереть содержимое АККу 1 по 4 : 0B 111 заполняет все четыре регистра "О".

Параметры: отсутствуют

Ошибки: Отсутствуют

6.1.3. Пролистывание аккумуляторов вперед (0В 112) и назад (0В 113)

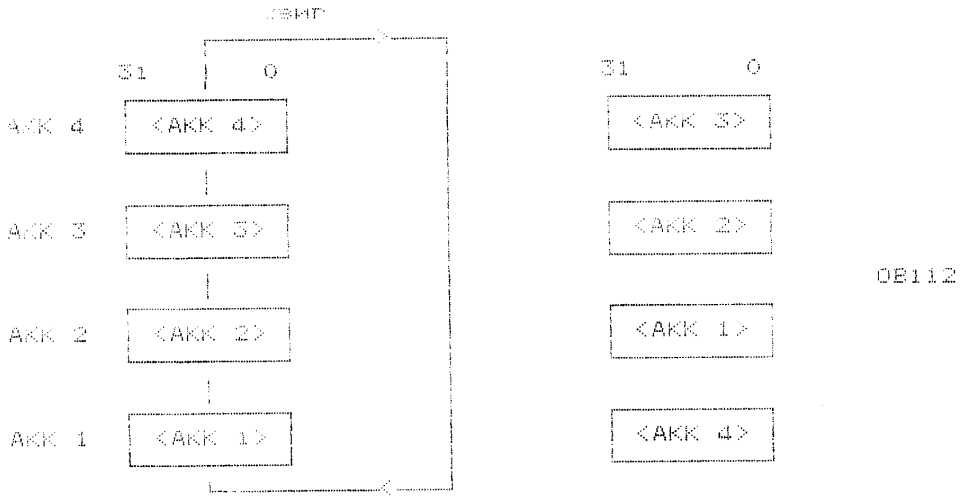
0В 112 и 0В 113 как бы "прокручивают» содержимое аккумуляторов в прямом и обратном направлении:

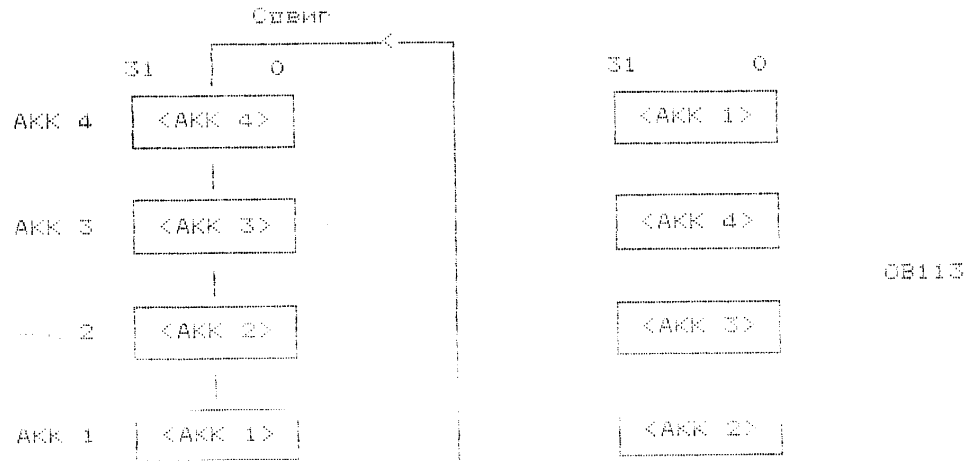
- ОВ 112 (прокручивание вперед) сдвигает содержимое АККу 1 в АККу 2, содержимое АКК 2 в АККу 3 и т. д.
- 0В 113 (прокручивание назад) сдвигает содержимое АККу 1 в АККу 4, содержимое АККу 4 в АККу 3 и т. д.

Параметры: отсутствуют

Ошибки: Отсутствуют

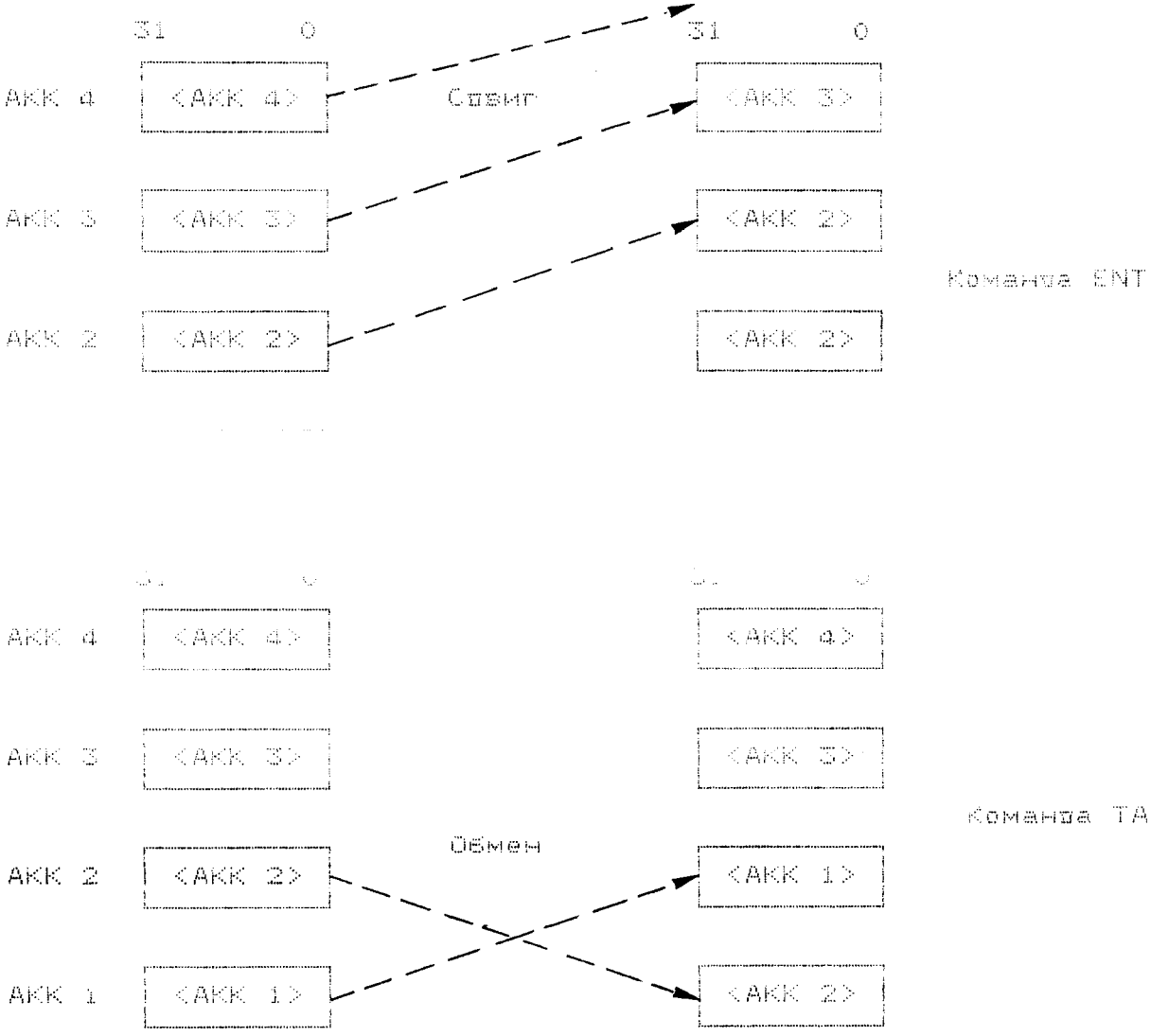
Следующие рисунки показывают содержимое аккумуляторов до и после вызова ОВ 112 и ОВ 113.





С помощью команд языка STEP 5 ENT (расширенный набор операций) и TAK (системная операция) можно тоже сдвигать содержимое аккумуляторов.

Обратите внимание на следующие различия:



6.2. Команды управления структурой

6.2.1. Счетчик времени цикла (OB160...0B163)

С помощью этой Функции можно реализовать программные циклы с необходимым временем выполнения.

Каждому из четырех организационных блоков специальных функций соответствует определенное слово системных данных:

BS60 → OB160

BS61 → OB161

BS62 → OB162

BS63 → OB163

Посредством этих системных слов можно задавать значение времени цикла. Если вызывается такая функция, то значение счетчика времени цикла уменьшается на значение, заданное в соответствующем слове системных санных. Цикл будет длиться до тех пор, пока значение в счетчике времени никла не будет равно 0. (если в счетчике времени цикла перед вызовом спецфункции было значение 0, то это значение все равно будет уменьшено. В результате длительность цикла станет равной 65535!).

Счетчик цикла в системном слове > 0 : VKE устанавливается (VKE = 1)

Счетчик цикла в системном слове = 0 : VKE сбрасывает (VKE = 0)

Остальные биты и слова индикации и признаков сброшены!

Содержимое аккумуляторов не изменяется и не анализируется. Таким образом, оно к началу следующего цикла сохраняется, и нет надобности в новой загрузке. Четыре спецфункции OB160..OB163 позволяют строить четырехуровневые циклы. Для этого необходимо занести в системные слова BS60…BS63 четыре разных значения счетчика циклов.

При необходимости эти функции можно использовать совместно с командой B BS (обработка системных данных).

Время выполнения функций OB160...0B163, если перед вызовом счетчик цикла не равен 1: 12…16 мкс.

Параметры:

1. Системное слово BS60...BS63: Счетчик цикла

Возможные значения: 0...65535 (FFFFH)

Ошибки: Отсутствуют

Пример:
В слове маркеров x устанавливается число циклов.

```
Инициализация цикла:        :L  KB 0
                            :L  MWx          счетчик циклов
                            :!=F
                            :SPB =M002
                            :T BS62          счетчик циклов
                            :                передается в системное
                            :                слово данных
                            :
                            :
                            :
                            :

Цикл:                 M001  :  …
                            :  …
                            :  …
                            :  …
Управление циклом:          :SPA OB 162      счетчик циклов
                            :SPB =M001       при VKE = 1 цикл
                            :                повторяется

Продолжение программы:M002  :  …
                            :  …
                            :  …
                            :  …
```

(Дополнительные примеры Вы найдете в главе 9.2 «TNW и TNB: передача блоков памяти».)

6.3. Считывание стека блоков (BSTACK) (OB 170)

В стеке блоков регистрируются все блоки (начиная с OB1 или FX0), вызываемые друг за другом, обработка которых еще не завершена.

С помощью спецфункции OB 170 можно считывать имеющиеся в BSTACK записи в блок данных. Таким образом можно выяснить число находящихся а BSTACK записей и, тем самым резерв, имеющейся для дальнейшей работы.

В каждой записи Вы получите адрес возврата (SAZ), абсолютный начальный адрес блока данных (DBA) этой подпрограммы и его длину (число слов = DBL).

Внимание!
Перед вызовом OB170 должен быть вызван блок данных (DB или DX) достаточной длины. Для каждой записи BSTACK необходимо резервировать четыре слова в блоке данных.

Параметры:

1. АКК 2-L: Номер слова данных (DWп). С которого должны размещаться данные, считываемые в DB (смещение).

2. АКК 1-L: Желаемое число элементов BSTACK

Возможные значения: 1-62

Например: если в АКК 1-L находится число ‘1’, то получается последняя запись, если ‘2’ – последняя запись и предпоследняя и т. д.

После успешного вызова OB170:

- В АКК 2-L находится указатель для блока данных.

- В АКК 1-L находится фактическое число считанных элементов BSTACK

Возможные значения: 0 – 62, если
фактическое число требуемого числа:

0 – ‘нет элементов BSTACK’ или ‘ошибка’

(Содержимое АКК 1-L, умноженное на 4, дает число переписанных слов в вызванный DB!).

— \VKE стирается

— можно анализировать признаки результата ANZ0 и ANZI (см. дальше),

— все остальные слова и биты результата стираются.

Ошибки:

— Не вызван блок данных

— Вызванный блок данных отсутствует или недостаточного размера

— Недопустимый параметр в АКК 1 или АКК 2

В случае ошибки, в VKE. ANZ0 и ANZ1 заносятся значения ‘1‘, остальные слова и биты стираются. Содержимое АКК 1 равно '0'.

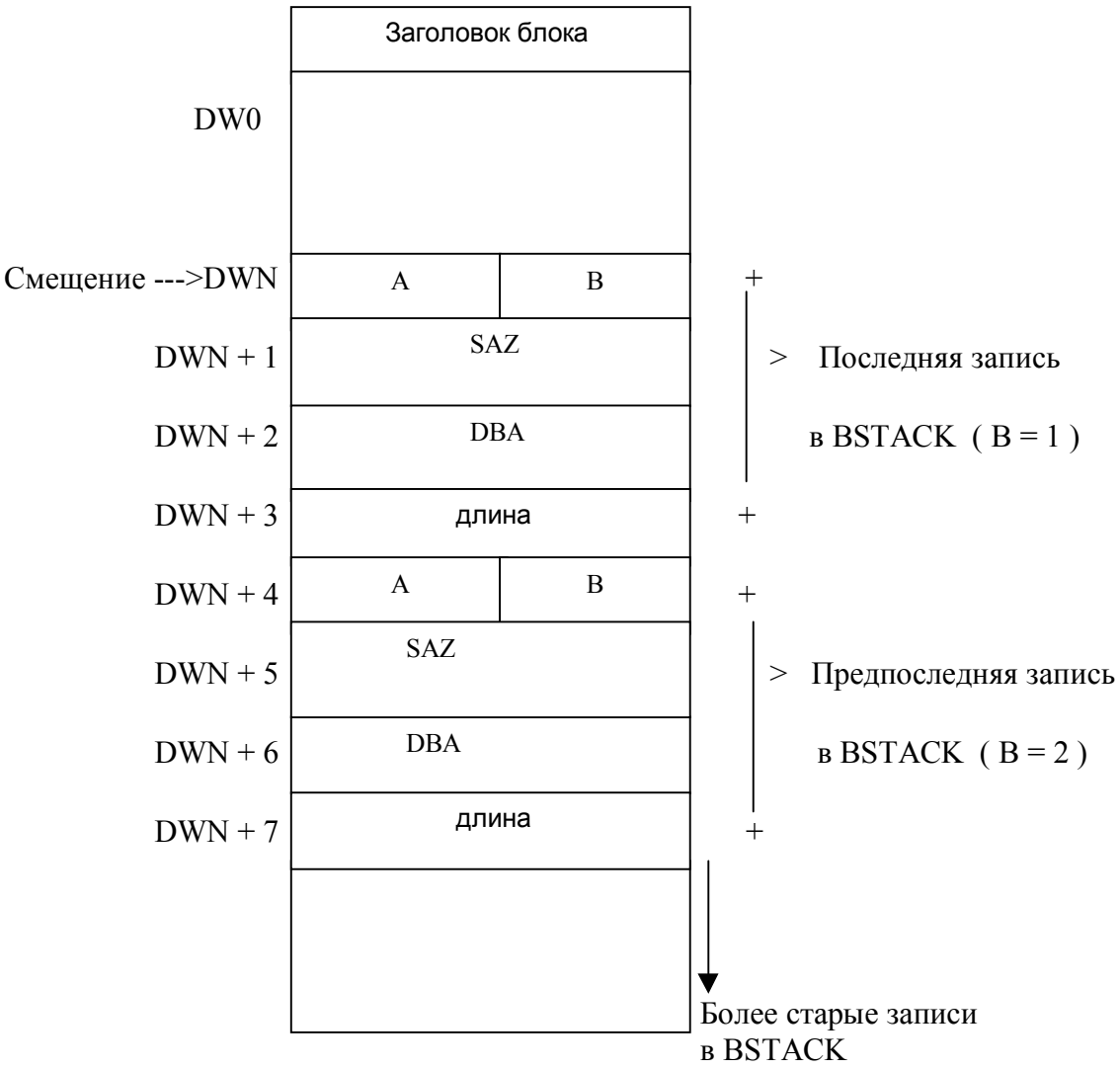
Установка признаков результата VKE, ANZ0, ANZI

VKE	ANZ0	ANZI	Опрос по команде	Значение:
0	1	0	SPM	Наличие элементов в BSTACK меньше требуемого количества
0	0	0	SPZ	Наличие элементов в BSTACK равно требуемому количеству
0	0	1	SPP	Наличие элементов в BSTACK больше требуемого количества
1	1	1	SPB	Ошибка

Пусть содержимое BSTACK при вызове OB 170 будет таким:

A = номер элемента BSTACK (62 – 1)
(Даже при чтении только последнего элемента BSTACK. необходимо предусмотреть резерв: A = 17 ---> резерв = A — 1 = 16).

B = глубина стека блоков (1 – 62)

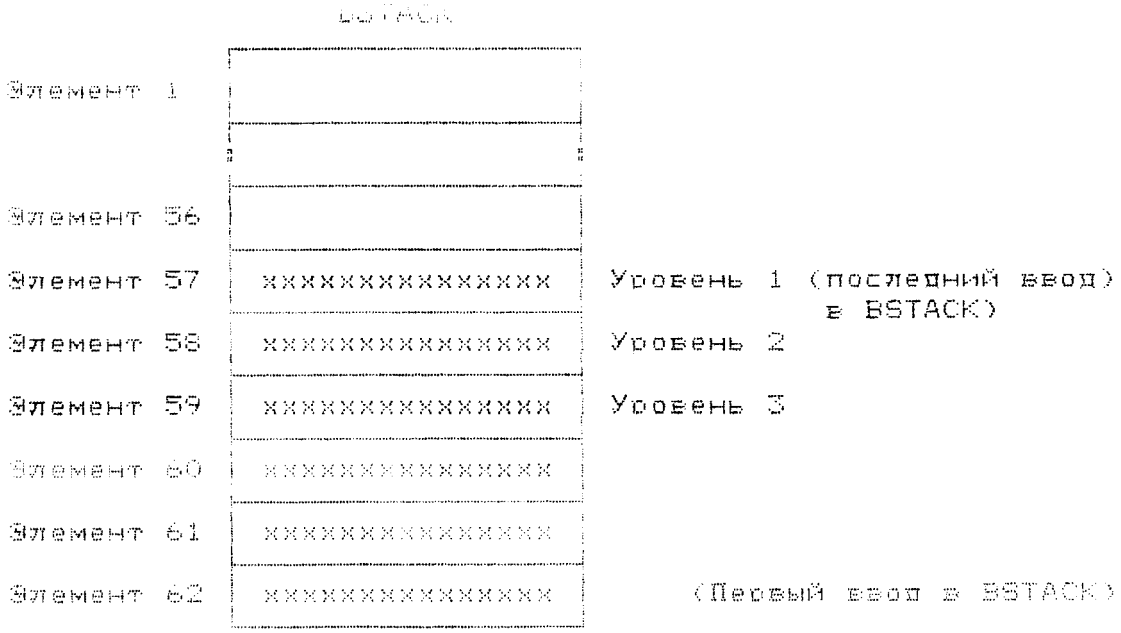


Пример:

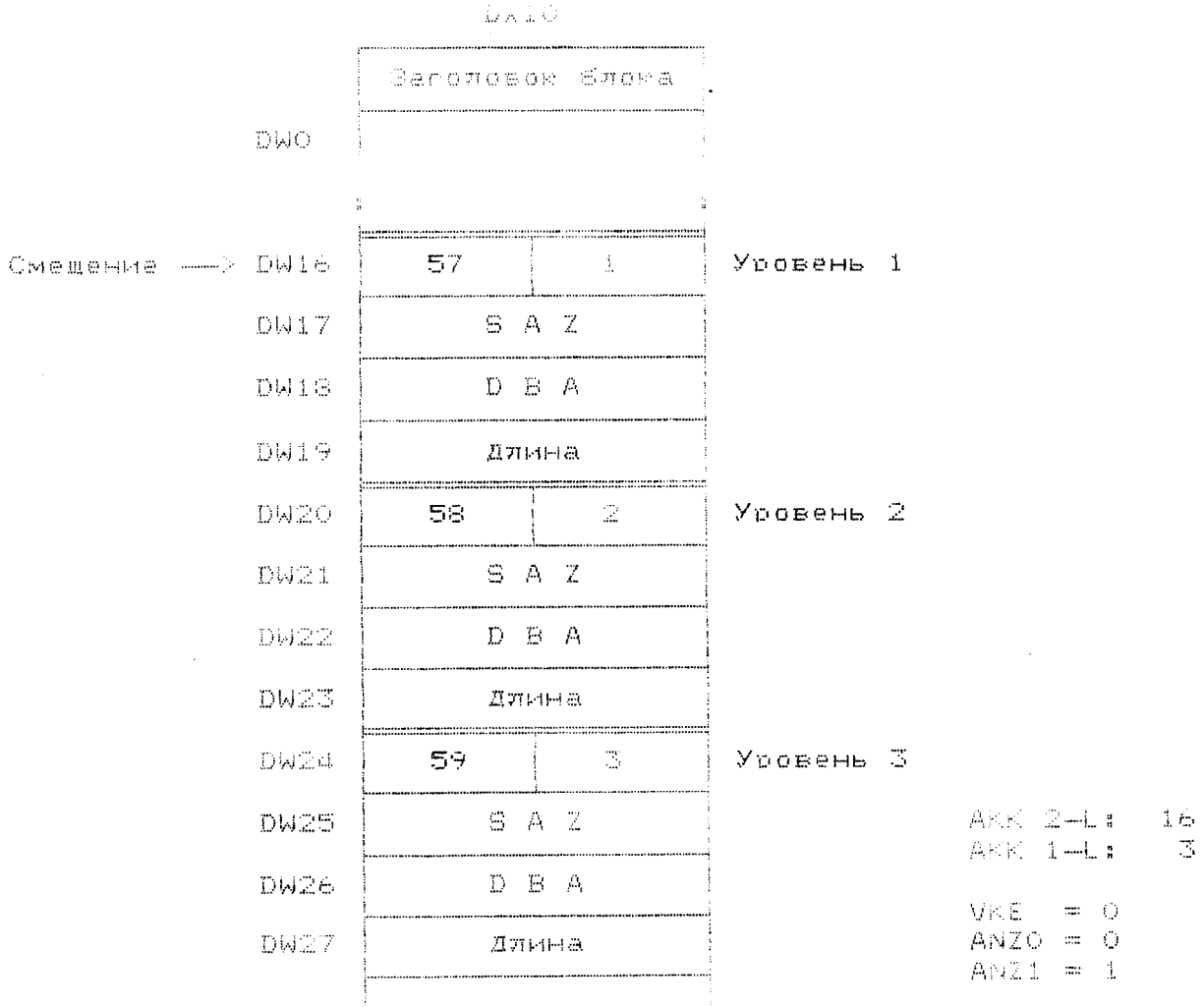
Вы хотите считать в DX10 три последних записи из BSTACK. Записи должны размещаться в DX 10, начиная со слова DW 16.

```
: AX   DX    10     : Вызов DX 10
: L    KY    0.16   : Записи должны размещаться с 16 слова
: L    KY    0.3    : Требуемое количество записей BSTACK
: SPA  OB    170    :
```

В BSTACK зарегистрировано 6 блоков:



После вызова спецфункции OB в блоке DX10 будет следующая информация:



6.4. Обработка блоков

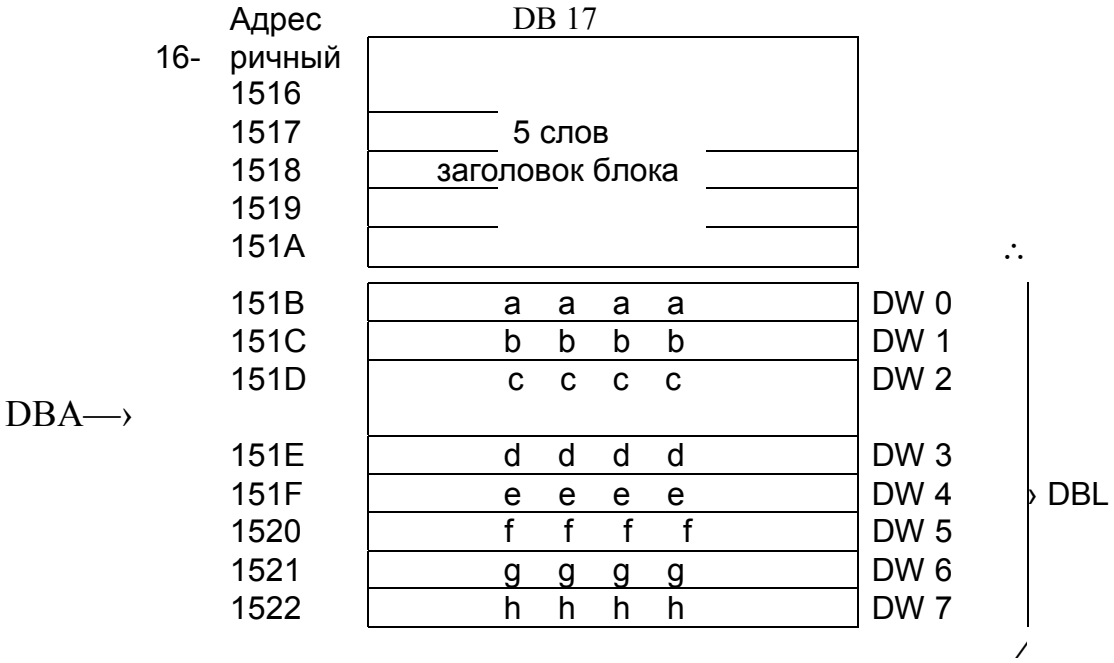
6.4.1. Управляемый доступ к блоку данных (0B 180)

При вызове блоков данных с помощью команд A DB и AX DX в регистр DBA (регистр адреса блока данных регистр № 6) загружается адрес слова DW0, которое находится в DB0. Пользователь имеет доступ к регистру DBA (16 бит) из программы STEP5 (команды LIR 6. TIR 6) доступ к блоку данных с помощью таких команд как L DF 60 или B DW 240 и т.д. осуществляется всегда относительно начального адреса блока данных.

Внимание!
Доступ к данным STEP—5 возможен только до слова DW255!
Обычно, наряду с регистром DBA пои каждом вызове блока паннах загружается и регистр DBL (регистр длины блока, регистр №8): он содержит длину (в словах) открываемого блока данных DB или DX без учета заголовка блока.
Пользователь имеет доступ к регистру DBL (16 бит) из программы STEP-5.

Внимание!
В регистр DBL можно записать, в зависимости от величины памяти используемого программатора, максимально 4091 слово!

Пример:



Регистр DBA содержит адрес слова памяти, в котором расположено DW 0, например: CBA = 151B (Шестнадцатеричный)

Регистр DBL содержит число слов данных, например: DBL=8 (DW 0 по DW 7)

Так как обращение к словам данных производится с помощью команд языка STEP 5 L DW, UD, B DW и т.д. всегда относительно DBA, то например, чтобы обратиться к DW 3, надо прибавить 3 к 151B. по адресу 151E расположено DW 3.

При обращении для записи с помощью DBL регистра проверяется, не возникнет ли ошибка передачи, например, T DW7 допустимо, а T DW8 уже ошибка.

OB спецфункции 180 увеличивает содержимое регистра DBA на заданное число слов данных. При этом вы сможете с помощью команд языка STEP 5 обращаться и к блокам данных, с длиной более 256 слов.

Путем соответствующего параметрирования и последующего вызова OB 180 можно сдвигать область обращения с помощью команд языка STEP5 размером в 256 слов в блоке данных в любом направлении.

ВАЖНО!

Перед вызовом OB180 необходимо вызвать достаточно длинный блок данных (DB или DX).

Параметры:

1.AKKu 1-L: Смещение (число слов данных, на которое должен быть сдвинут начальный адрес блока данных)

возможные значения: содержимое AKKu 1-L<DBL!

После соответствующего вызова OB180

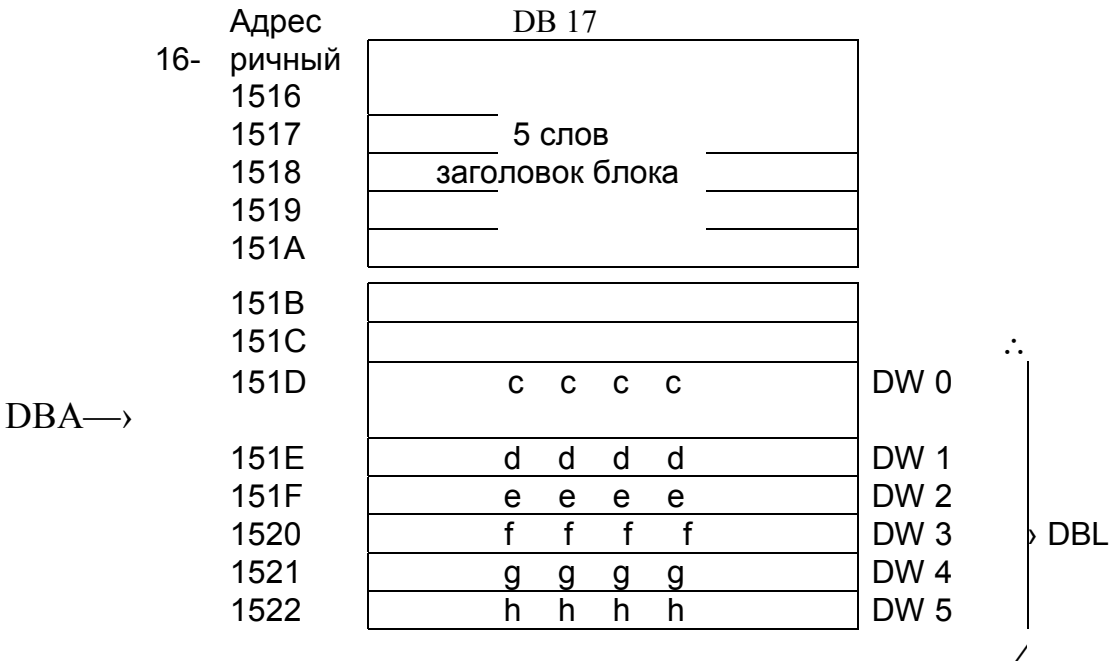
- значение регистра DBA(= адресу DW0) увеличивается на значение AKKu 1-L
- значение регистра DBL уменьшается на значение AKKu 1-L
- VKE сбрасывается (VKE = 0)
- все остальные биты индикации сбрасывается.

Ошибки:

- блок данных не открыт
- содержимое AKKu 1-L > DBL

В случае ошибки (содержимое Акku 1-L > DBL) регистры DBA и DBL не изменяются. VKE. устанавливается (VKE=1). Все остальные биты слова индикации сбрасываются.
Если регистр DBL содержит значение "0', то OB 180 опознает, что блок данных не открыт. VKE устанавливается (VKE=1) и сигнализирует об ошибке.
Сброс DBA и DBL в начальное значение.
Новый вызов блока данных командами A DB или AX DX возвращает все в исходное состояние.
Пример:
Начальный адрес блока данных (DBA=151B) DB 17 (DBL=3) должен быть сдвинут на два слова данных.

```
A     DB    17
L     KB    2
SPA   OB    180
```



DBA после вызова OB180 = 151DH
DBL после вызова OB180 = 6 (DW0...DW5)

После вызова OB180, например, содержимое слова FFFF с адресом 1520A нельзя прочитать как DW5, а только как DW3 и т. п.

Одновременно с передачей DBL производится контроль ошибок: команда T DW5 является правильной, а команда T DW6 – является уже ошибочной.

Новым вызовом OB180 можно еще больше увеличить DBA (при этом DBL будет еще меньше!): команда A DB17 восстановит значение регистров (DBA=151BH, DBL=8).

Если длина DB17, например. 258 слов, то с помощью команд STEP5 нельзя обратиться к словам DW256 и DW257. Однако, после смещения DBA на 2, слова DW256 и DW257 могут быть опрошены как слова DW254 и DW255 соответственно.

Возможное использование OB180

- Доступ к блокам данных с большой длиной, т.е. длиной больше 261 слово (5 слов заголовок, DW0…DW255), показан выше.
- Работа со структурами данных

  Если фрагмент данных, состоящий из нескольких слов, многократно повторяется внутри одного процесса, причем назначение отдельных слов данных всегда постоянно, то мы говорим о структурах данных.

  Например, состояние процесса (части процесса) можно описать 20 словами, причем первое слово содержит значение температуры, второе – давление и т. д. Если это описание состояния процесса многократно разместить внутри одного блока, например в случае существовании множества частных процессов и\или при необходимости запомнить промежуточное состояние процесса, то можно, используя OB180, иметь доступ к каждой структуре с одними и теми же параметрами (0…19) в командах L DD, S D, T DR и т.д.

  В отличие от других механизмов замещения (Замещение=явная параметризация) в данном случае мы получаем наиболее простые и быстрые подпрограммы.

(О регистре DBA смотри Раздел 9 «Доступ к памяти через абсолютные адреса», регистр 6).

6.4.2. Тестирование блоков данных DB/DX (OB181)

С помощью OB181 можно проверить:

a) имеется ли в наличии определенный блок данных DB или DX.

b) по какому адресу размещается первое слово блока данных.

c) сколько слов содержит этот блок.

d) тип памяти и области (память пользователя: RAM или EPROM, NDB-RAM).

Применять функцию «Тестирование DB/DX" имеет смысл перед командами TNB/TNW, E DB/DX и перед вызовом организационных блоков OB254 и OB255.

Так, например, перед пересылкой блоков данных OB181 вызывается, чтобы убедиться, что блок – приемник верен и его длина достаточна для того, чтобы в нем разместились все копируемые слова.

Параметры:

1. АКК 1-LL : Номер блока
возможные значения: 1…255

2. АКК 1-LH : Признак блока
возможные значения: 1 - DB
2 - DX

Если тестируемый блок в процессоре имеется, то

- в АКК 1-L содержится адрес первого слова блока (DW0).
- в АКК 2-L содержится длина блока в словах (без заголовка блока)
  Например. В АКК 2-L находится «7» –> блок состоит из DW0…DW6.
- VKE сброшен (VKE=0).
- ANZ0 и ANZ1 заполнены (смотри ниже).
- Слова и биты признаков сбрасываются.

Если тестируемого блока нет в наличии или его параметры не корректны то:

- устанавливается VKE (VKE=1).
- ANZ0 и ANZ1 заполнены (см. ниже).
- слова и биты признаков сбрасываются.
- состояние аккумуляторов изменяется.

Ошибки:

- Неправильный номер блока (недопустимо : 0)
- Неправильный признак блока (недопустимо : 0,3...255)
- Ошибка памяти

Обзор: значения VKE, ANZ0 и ANZ1:

VKE = 0: DB имеется
VKE = 1: DB не имеется

ANZ0 = 0: DD в модуле памяти пользователя
ANZ0 = 1: DB в DB-RAM

ANZ1 = 0: DB в памяти EAM
ANZ1 = 1: DB в памяти EPROM

VKE		ANZ1	Опрос по команде	Значение		
0	1	0	SPM	DB в памяти пользователя	DB в EPROM	>DB существует
0	0	0	SPZ			
0	0	1	SPP	DB в DB-RAM	> DB в RAM	
1	1	1	SPB	DB не существует/ ошибка памяти ошибка параметрирования		

Примеры смотрите:
Раздел 8.2.2 "Списки адресов блоков в DB—RAM"
Раздел 9.1 "LIR и TIR: Доступ к параметрам"
Раздел 9.2 "TNW и TNB: Передача блоков памяти"

6.4.3. Передача маркеров в блок данных (OB 190 и OB 192)

Организационные блоки OB 190 и OB 192 передают заданное пользователем число маркерных байтов в указанный блок данных.

Это удобно, например, перед вызовом блока, в организационных блоках ошибки или при прерывании циклической обработки программы программами управляемыми по времени или по прерыванию от процесса.

С помощью OB 191 и OB 193 можно выполнить обратную операцию – передать значения из блока данных назад в маркерные слова.

Указание: используйте OB 190 и OB191 для простого сохранения и восстановления маркерных байтов, т.к. при этом вы получите значительный выигрыш во времени выполнения.

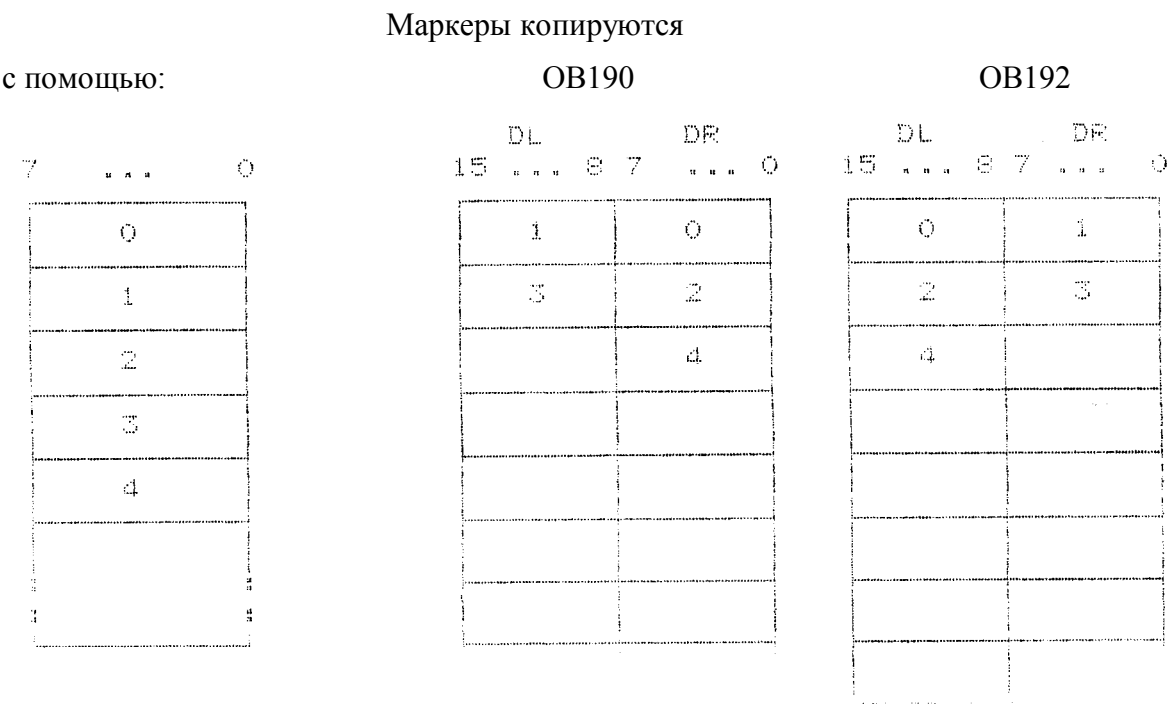
Внимание!
Перед вызовом спецфункции необходимо открыть блок данных (DB/DX)!

После вызова OB190/OB192 маркерные байты временно запоминаются блоке данных, начиная с заданного слова. Область сохраняемых таким образом маркерных байтов выбирается OB190/OB192 из АКК.

Спецфункции OB190/OB192 работают аналогично, за исключением передачи маркерных байтов:

- OB190 передает маркеры по байтам
- OB191 передает маркеры по словам

Это является существенным, если данные, передаваемые в блок данных, затем будут обрабатываться, и блок данных не является просто промежуточной памятью.
Следующий рисунок поясняет различия OB190 и OB 192:



Замечание: Если передается нечетное количество маркерных байтов, то последнее используемое слово блока данных используется только наполовину. При OB190 остается свободной левая часть, а при OB192 – правая часть слова.

Важно!
Обратите внимание на следующую таблицу (время в мкс):

(n – количество маркерных байтов)	OB190	OB192
Добавка к каждому байту маркеров, если первый байт лежит по четному адресу	25 + n.0.32	40 + n.0.57
Добавка к каждому байту маркеров, если первый байт лежит по нечетному адресу	25 + n.0.48	25 + n.1.8

Параметры:

Данные для источника:

1. АКК 2-LH: Первый передаваемый маркерный байт
   Возможные значения: 0…255
2. АКК 2-LL: Последний передаваемый маркерный байт
   Возможные значения: 0…255

(Последний маркерный байт первого маркерного байта!)

Данные для приемника:

3. АКК 1-L: Адрес первого переписываемого слова вызванного блока данных
   возможные значения:
   -ориентируйтесь на длину блока данных в памяти
   -при этом могут появиться номера > 255

Если функция OB190/192 выполнена правильно, то VKE сбрасывается (VKE=0). Аккумулятор остается без изменений.

В случае ошибки VKE=1, аккумулятор остается без изменений.

Ошибки:

- Не вызван блок данных DB или DX
- Неправильно задана область маркерных байтов (номер последнего байта меньше номера первого байта)
- Нет адреса слова в блоке данных
- Длина блока данных недостаточна

6.4.4. Передача из блока данных в область маркеров (OB191 и OB193)

С помощью OB191 и OB193 можно передать данные из блока данных в маркерную область. Так можно, например, предварительно сохраненные маркеры вернуть назад в маркерную область.

OB191/193 отличаются от OB190/192 тем, что источник и приемник меняются местами:

OB190/192: Область маркеров ⟶ маркер блок ⟶ данных

OB191/193: Область маркеров ⟵ данные ⟵ блок данных

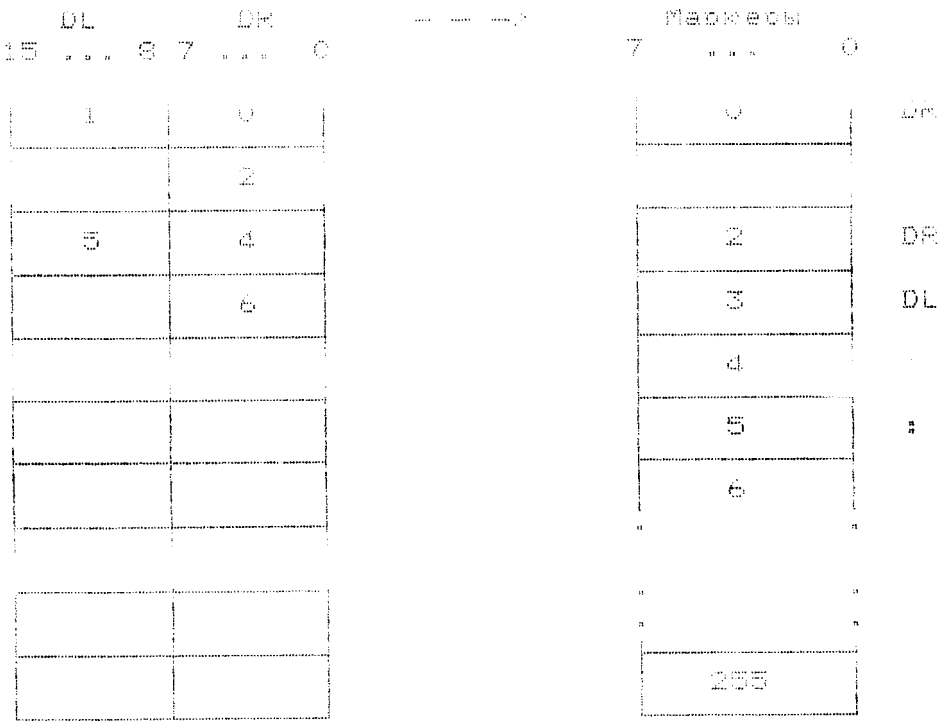
Внимание!
Перед вызовом функции должен быть открыт блок данных (DB/DX)!

Функции OB191 и OB193 идентичны за исключением способа передачи данных:

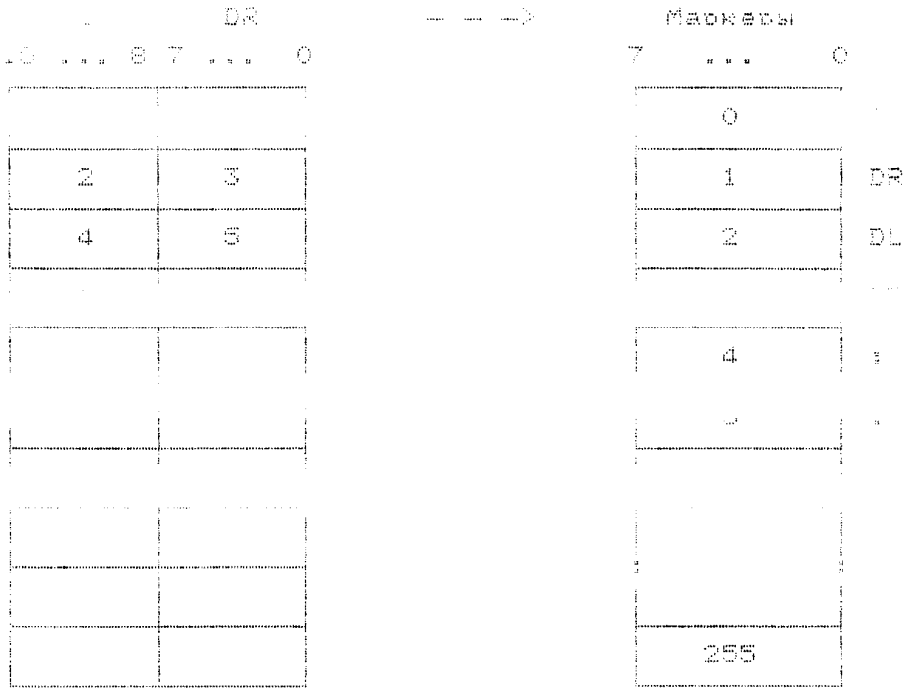
- OB191 передает слова данных побайтно
- OB193 передает слова данных пословно

Следующий рисунок поясняет это различие:

Данные передаются с помощью: **OB191:**



Данные передаются с помощью: OB193



Важно !
Обратите внимание на следующую таблицу (время в мкс):

(n – количество маркерных байтов)	OB191	OB193
Добавка к каждому байту маркеров, если первый байт лежит по четному адресу	25 + n.0.32	40 + n.0.57
Добавка к каждому байту маркеров, если первый байт лежит по нечетному адресу	25 + n.0.48	25 + n.1.8

Параметры:

Данные для источника:
1. АКК 2-L: Адрес первого передаваемого слова в открытом блоке данных

Данные для приемника информации:

2. АКК 1-LH: Первый переписываемый маркерный байт
возможные значения: 0…255
3. АКК 1-LL: Последний переписываемый маркерный байт
возможные значения: 0…255

(Последний маркерный байт > первого маркерного байта!)

Если функция OB191/193 выполнена правильно, то VKE сбрасывается (VKE=0).
Аккумулятор остается без изменений.

В случае ошибки VKE=1, аккумулятор остается без изменений.

Ошибки: Смотри ошибки для OB190/192

Пример:

Перед вызовом программного блока PB12 все маркеры (MB0…MB255) сохранить в блоке данных DX37 по адресу 100 а потом опять восстановить.

Сохранение:	:AX DX 37	Вызов блока данных
	:L KY0.255	Определение области маркеров MB0…MB255
	:L KB 100	Адрес первого слова в блоке данных
	:SPA OB190	Сохранение маркеров
Вызов программного блока	:SPA PB12	
Восстановление	:	Блок данных уже вызван
	:L KB 100	Адрес первого слова
	: L KY 0.255	Область маркеров MB0…MB255
	: SPA OB 191	Восстановление маркеров

Применение OB 190 / OB 191

Маркеры, которые используются в циклической программе пользователя, не могут кроме того использоваться в программах пользователя, управляемых по времени и по прерыванию от процесса. Каждому уровню обработки программ должна соответствовать определенная часть области маркеров.

Например:	циклическая программа пользователя:	MB0…MB99
	программа пользователя, управляемая по времени:	MB100…MB199
	программа пользователя, управляемая по прерыванию от процесса:	MB200…MB255

Однако, если в программе пользователя выполняемой циклически, уже используются все 256 байтов маркеров, а, например, для выполнения программы пользователя, управляемой по времени тоже необходимы все 256 байтов маркеров, то маркеры при смене уровня обработки должны меняться и буферизироваться.

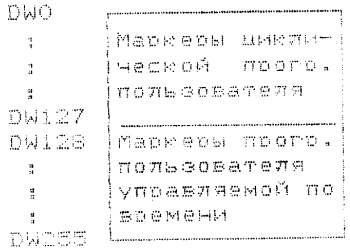
OB13

```
A DB100
MB 0...255    --> DW 0...127
DW 128...255 --> MB 0...255

программа пользователя
управляемая по времени

A DB100
MB 0...255    --> DW 028...255
DW 0...127    --> MB 0...255
BE
```

DB100



Самое быстрое сохранение и восстановление маркеров возможно с помощью спецфункций OB190 и OB191:

Программа STEP5 в OB 13

```
: A    DB100
: L    KY 0.255
: L    KB0
: SPA  OB190        → 352 мкс
: L    KB128
: L    KY 0.255
: SPA  OB191
: .
: .
: .
: A    DB100
: L    KY 0.255
: L    KB 128
: SPA  OB190        → 352 мкс
: L    KB0
: L    KY 0.255
: SPA  OBB191
: BE
```

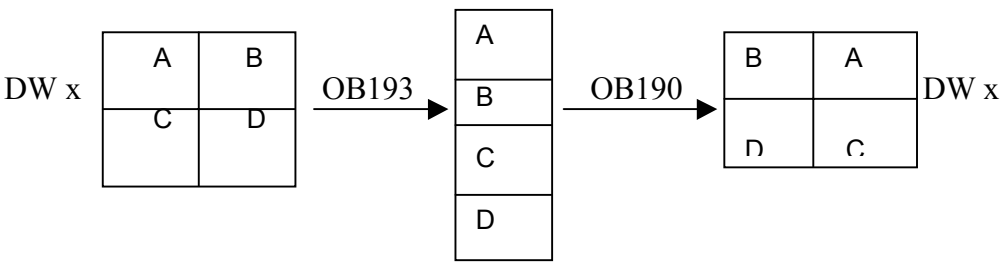
Для замены и буферизации маркера необходимо на вызов OB13 704 мкс.

Другие применения организационных блоков OB190…OB193

- В CPU928 команды обработки битов (U.O.ON.UN.S.R.=) выполняются значительно быстрее, если они работают с областью маркеров, чем при использовании блоков данных (сравните ‘UM’<->’UD’ ‘SSM’ <-> ‘SD’!)

  По этой причине можно значительно улучшить временные характеристики, если копировать данные из блоков данных в маркерную область, обработать их там, а после этого переслать обратно в блок данных.

- Без больших затрат можно переставить местами байты в словах, если переслать массив в маркерную область и вернуть затем обратно:



- Можно перемещать массивы данных внутри блока, если при обратной переписи из маркерной области укажете тот же номер DB, но другое начальное слово.
- Также можно просто передавать массивы данных (максимум 255 байтов) в другие блоки данных (перед этим целесообразно произвести контроль с помощью OB181 «Тестирование блока данных (DB/DX)»).

6.4.5. Передача блоков данных в область памяти DB-RAM (OB254, 255)

С помощью OB254и OB255 можно передать блок данных из области памяти пользователя в область данных процессора DB-RAM. Спецфункции OB254 и OB255 работают одинаково, но OB254 применяется к блокам данных DX, а OB255 к блока данных DB.

При передаче можно использовать перемещение блока или копирование:

- Перемещение блоков данных из области памяти пользователя в область DB-RAM.

  Блок данных из памяти пользователя при сохранении своего первоначального номера перемещается в DB-RAM. Новый начальный адрес блока данных заносится в список адресов в DBO. Старый адрес блока сбрасывается, т.е. блок данных в памяти пользователя становится недействительным.

Параметры:

1. АКК 1-L : Номер перемещаемого блока

2. АКК 1-H : 0

Ошибки:

- Перемещаемый блок отсутствует (OB19).
- Блок уже имеется в DB-RAM (OB31).
- Недостаточно места в памяти DB-RAM (OB31).

В случае появления ошибки функция не выполняется. Системная программа распознает ошибку выполнения во время цикла и вызывает OB19 или OB31. Дальнейшая реакция на ошибку зависит от того, как запрограммирован OB19 или OB31.
Если OB19 или OB31 не запрограммированы, то процессор переходит в состояние останова. В АКК 1 находится признак ошибки, который детализирует последнюю возникшую ошибку.

- Дублирование блоков данных в DB-RAM

  Блок данных из памяти пользователя или DB-RAM передается в DB-RAM, причем он получает другой номер. Начальный адрес нового блока заносится в список адресов в DB0. Начальный адрес старого блока остается в DB0 , т.е. первоначальный блок доступен для работы.

  Запись начального адреса в DB0 произойдет только в случае, если передача блока выполнена полностью и все признаки в заголовке блока сформированы правильно. Сдублированный блок после завершения передачи опознается операционной системой как действительный или имеющийся.

  Параметры:

  1. АКК 1-L: Номер дублируемого блока

  2. АКК 1-H: Номер нового блока данных

  Ошибки:

  - Нет дублируемого блока (OB19)
  - Новый блок уже имеется (OB31)
  - Недостаточный объем памяти DB-RAM (OB31).

В случае возникновения ошибки функция не выполняется. Системная программа распознает ошибку во время выполнения и вызывает OB19 или OB31. Дальнейшая реакция на ошибку зависит от того, запрограммирован ли OB19 или OB31 (смотри “Специальные ошибки выполнения”). Если OB19 или OB31 не запрограммированы. То процессор переходит в состояние останова. В АКК 1 находится признак ошибки, который детализирует последнюю возникшую ошибку.

6.5. Многопроцессорная связь (OB200…OB205)

Спецфункции OB200…OB205 позволяют в многопроцессорном режиме работы организовать передачу данных между процессорами через координатор KOR C.

OB200: Инициализация

Эта функция резервирует в KOR C память, в которую может быть временно записан передаваемый блок данных.

OB202: Передача

Эта функция передает блок в буферную память координатора KOR C и сообщает, сколько блоков данных еще можно передать.

OB203: Тест передачи

Функция OB203 сообщает о количестве свободных блоков памяти в координаторе.

OB204: Прием

Эта функция принимает блок данных из координатора и сообщает сколько блоков данных еще там осталось.

OB205: Тест приема

OB205 сообщает число занятых блоков памяти координатора KOR C.

Подробная информация по этим организационным блокам специальных функций находится в Разделе 8 Руководства.

6.6. Доступ к буферам связи

Организационные блоки OB216…OB218 дают возможность доступа к буферам связи.

Эти организационные блоки выполняют следующие функции:

OB216 записывает байт/слово/двойное слово в буфер связи

OB217 считывает байт/слово/двойное слово из буфера связи

OB218 закрепляет за процессором область буфера связи (служит для координации в многопроцессорной системе)

Эти функции, с одной стороны, служат для контроля, в то же время они могут быть использованы как элементарные средства программирования блоков управления и других аналогичных функций.
В многопроцессорных системах применение этих функций особенно полезно, когда информация записывается в буфер связи или читается из буфера связи несколькими процессорами.

Что такое буфер связи?

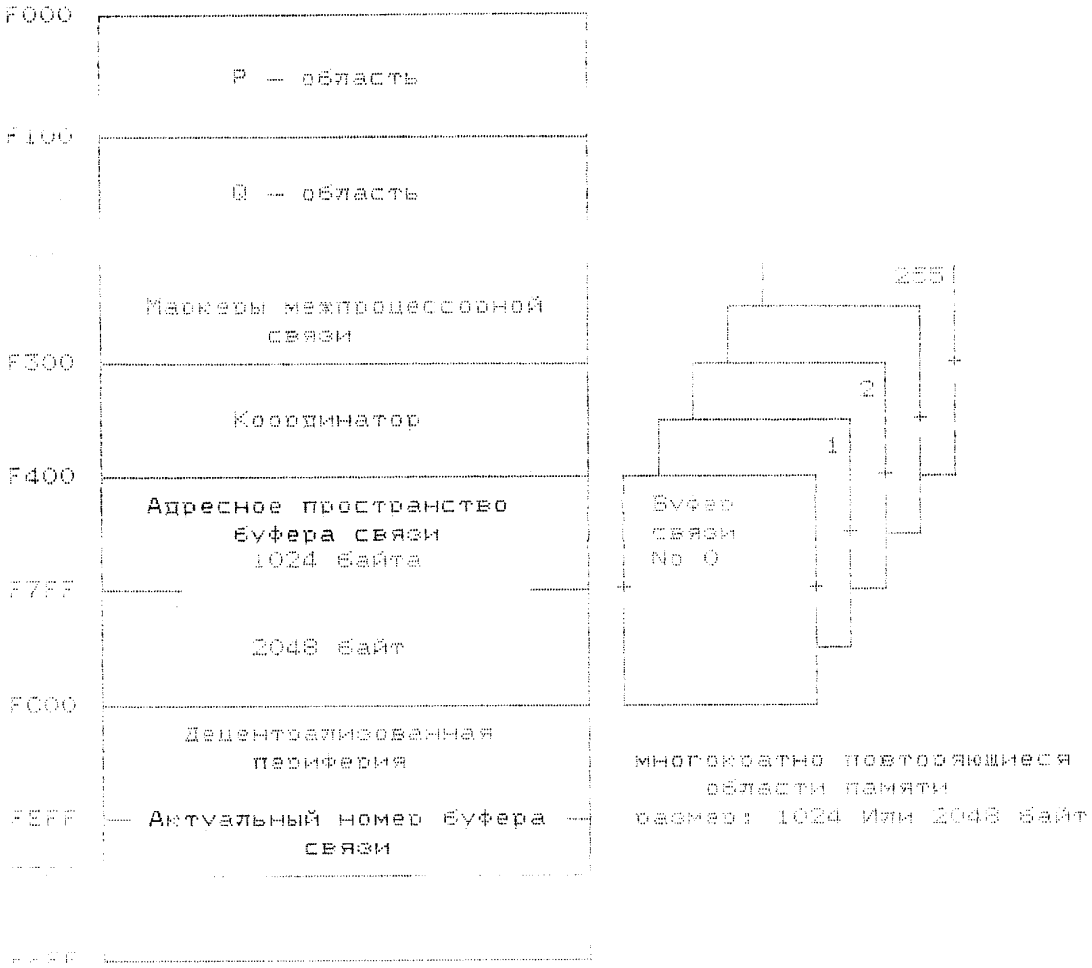
Буфер связи – это область памяти, которая имеется в коммуникационных процессорах, в ряде интеллигентных периферийных устройств и определенных координаторах для организации многопроцессорного режима работы.

Буфер связи имеет байтовую организацию (т.е. можно адресоваться к любому байту).

В контроллере может быть до 255 буферов связи.

Размер области буфера связи	Адресное пространство:
1024 байта	F400H – F7FFH
2048 байт	F400H – FBFFH

Адресное пространство периферийных устройств на шине S5.



При параметрировании спецфункций ОВ216...0В218 Вы задаете, какую совокупность из 255 буферов связи использовать. «Актуальный номер буфера связи» вырабатывается автоматически и заносится по адресу FEFFH (смотри Рисунок выше). После этого вся адресация в буфере связи ведется относительно этого адреса.

Внимание!
данные из ячейки с адресом FEFFH читать нельзя!

Указания по параметрированию.

При записи (OB 217) байта/ слова/ двойного слова используются следующие представления данных:



6.6.1. Запись в буфер связи (OB216)

Спецфункция передает байт/слово/двойное слово из АКК 1 в определенный буфер связи. Адресация буфера связи и передача полного набора данных (1/2/4 байта) образуют программную единицу, которая не может быть прервана.

Параметры:

1. АКК 2-L : Адрес цели в буфере связи
   возможные значения: 0…2048
2. АКК 3-LL: Актуальный номер буфера связи
   возможные значения: 0…255
3. АКК 3-LH: Признак передаваемых данных
   возможные значения: 0- байт
   1-слово
   2-двойное слово

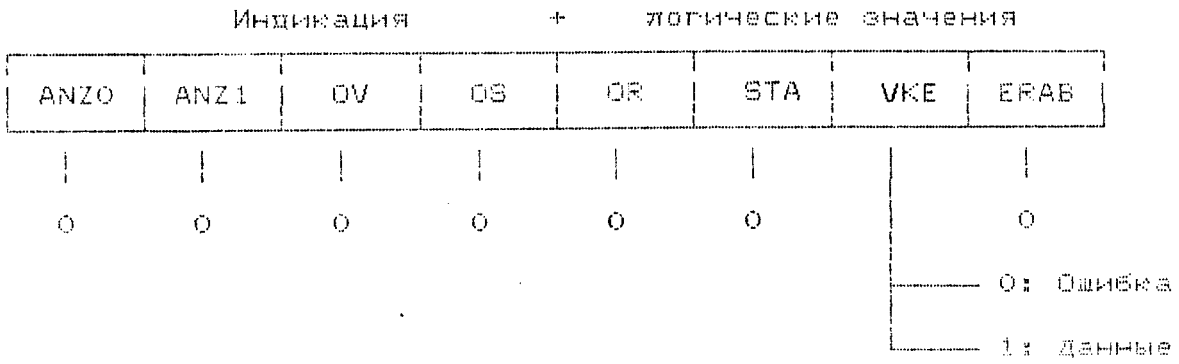
Если запись прошла успешно:

- содержимое АКК 1 и АКК 3 остается без изменений
- АКК 2-L содержит значение, увеличенное на 1/2/4 (на длину передаваемого набора данных)
- устанавливается VKE (VKE = 1)
- остальные признаки (биты и слова) сбрасываются (смотри индикацию результата)

Если запись в буфер связи невозможна, то:

- содержимое всех аккумуляторов остается неизменным
- VKE сбрасывается (VKE = 0)
- остальные признаки (биты и слова) сбрасываются

Индикация результата:

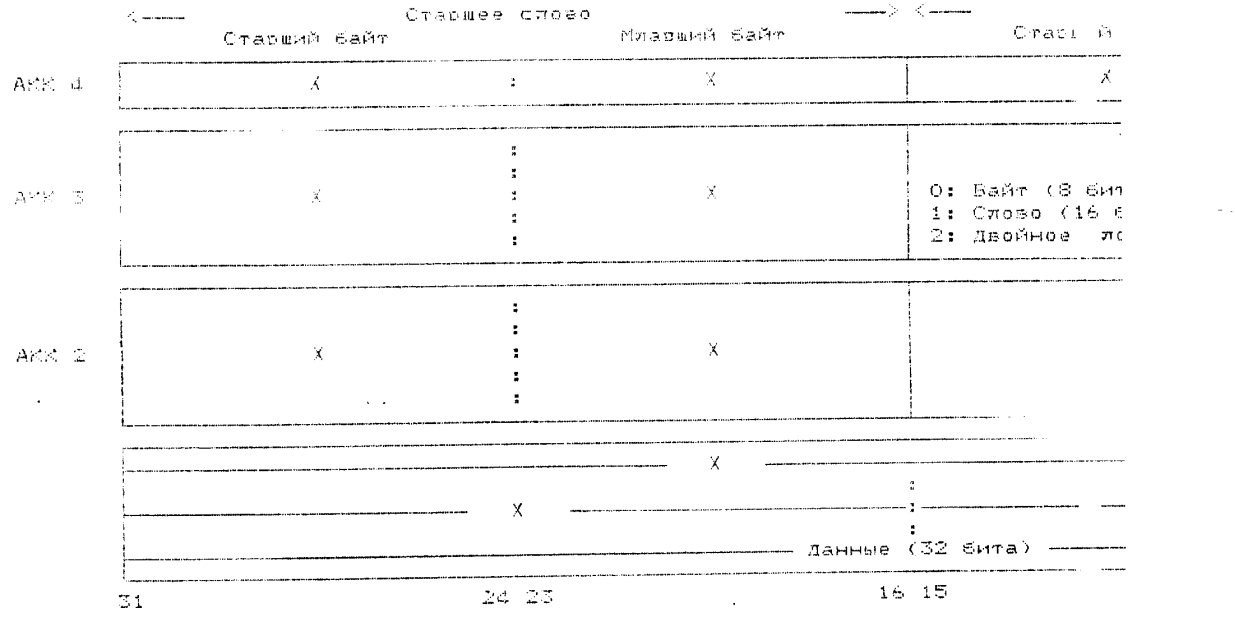


(смотри Раздел 3)

Ошибки:

- Неправильный признак в АКК 3-LH
- Адрес назначения в области буферной связи не верен или отсутствует
- Заданный номер области буферной связи отсутствует
- Области буферной связи нет вообще

Содержимое аккумуляторов перед записью



6.6.2. Чтение из буфера связи (OB217)

Спецфункция передает байт, слово или двойное слово из определенного буфера связи в аккумулятор АКК 1.
Адресация буфера связи и передача полного набора данных (1/2/4 байта) образует непрерываемый участок программы.

Параметры:

1. АКК 2-L: Адрес в области буферной связи
   Возможные значения: 0…2048
2. АКК 3-LL: Актуальный номер буфера связи
   Возможные значения: 0…255
3. АКК 3-LH: Признак передаваемых данных
   Возможные значения: 0- байт
   1-слово
   2-двойное слово

Если чтение прошло успешно:

- в АКК 1 записывается прочитанное значение (возможный остаток от 32 битов заполнятся нулями)
- содержимое АКК 3 не изменяется
- АКК 2-L содержит значение, увеличенное на 1/2/4 (на длину передаваемой порции данных)
- устанавливается VKE (VKE = 1)
- остальные признаки (биты и слова) сбрасываются

Если чтение из буфера связи невозможно, то:

- содержимое всех аккумуляторов остается неизменным
- VKE сбрасывается (VKE = 0)
- остальные признаки (биты и слова) сбрасываются

Ошибки:

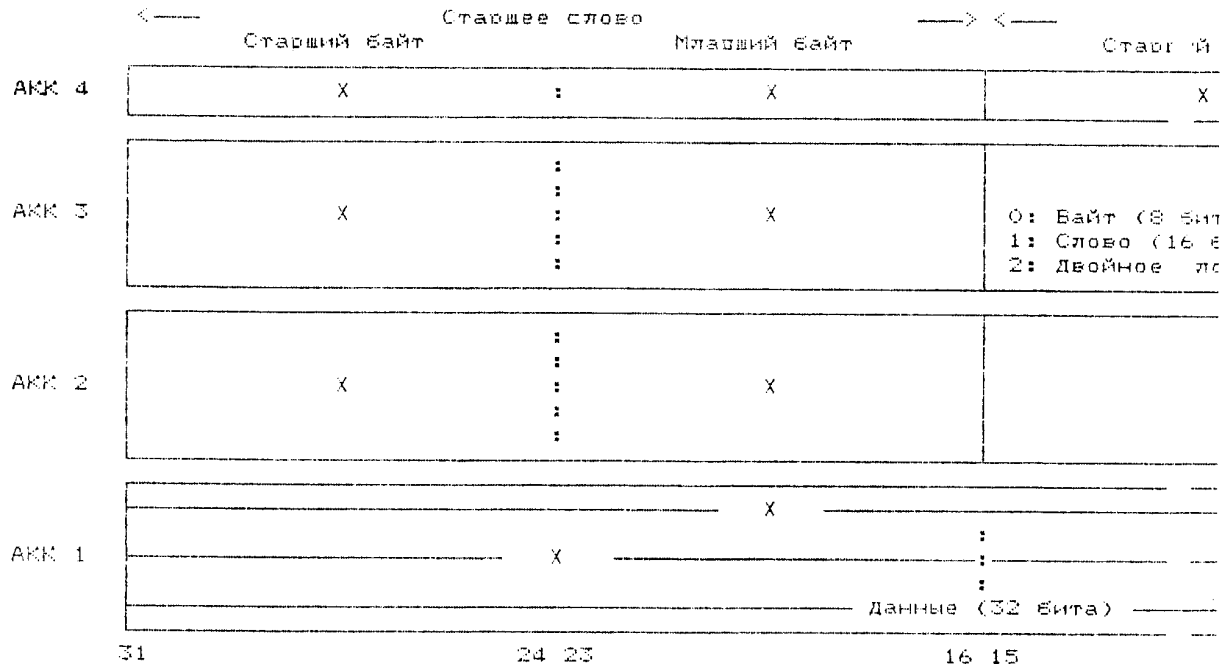
— Неправильный признак в АКК 3—LH

— Адрес в области буферной связи не верен или отсутствует

— Заданный номер области буферной связи отсутствует

— Области буферной связи нет вообще

Состояние аккумуляторов после процесса чтения



6.6.3 Захват буфера связи (OB218)

Организационный блок спецфункции передает метку установочного места «своего» процессора определенного буфера связи в случае, если адресуемая ячейка этой области буферной связи обнулена. До тех пор, пока указатель места остается в ячейке, этот буфер резервируется за определенным процессором и не может быть захвачен другим процессором.

Спецфункция OB218 служит для синхронизации передачи данных и поэтому особенно важна, когда должны срочно передаваться или приниматься большие связанные друг с другом блоки данных.

Адресация буфера связи. Чтение и запись указателя места процессора образуют непрерывающийся участок программы.

Параметры:

1. AKK 1-L: Адрес ячейки в буфере связи
   Возможные значения: 0…2047
2. AKK 2-LL: Актуальный номер области буферной связи
   Возможные значения: 0…255

(содержимое аккумуляторов АКК 3 и АКК 4 в этом случае не играет роли)

Если захват буфера связи прошел успешно (содержимое адресуемой ячейки равно 0), то

- содержимое аккумуляторов не изменяется
- устанавливается VKE (VKE = 1)
- остальные биты и слова признаков сбрасываются

Если захват невозможен (содержимое адресуемой ячейки не равно 0), то

- содержимое аккумуляторов не изменяется
- VKE сбрасывается (VKE = 0)
- остальные биты и слова признаков сбрасываются

Ошибки:

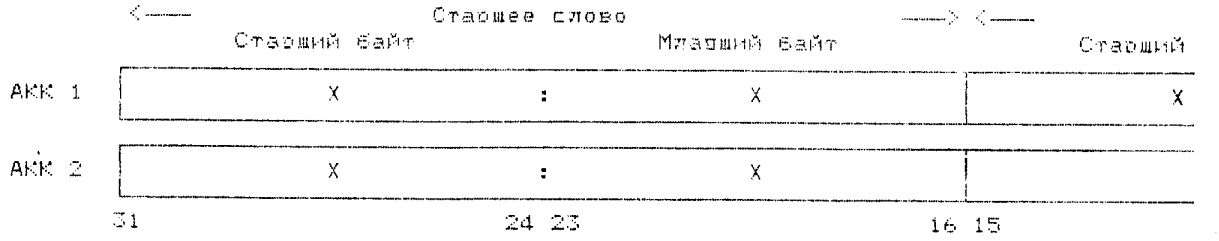
— Содержимое ячейки буфера связи не равно О

— Адрес в буферах связи неверный или отсутствует

— Заданный номер буфера связи отсутствует

— Вообще нет буфера связи

Аккумуляторы до/после захвата



Пример программы

Из DB45 одного CPU28 передать слова 4…11 через координатор KCR C в DX45 (слова 0..7) другого CPU28. Синхронизация между передатчиком и приемником (многопроцессорная система) осуществляется с помощью OB218.

Актуальный буфер связи в координаторе : No.255
Координирующая ячейка в буфере связи : Адр.53
Область для передачи данных в буфер
Связи (запись и чтение) : Адр.54…69

Передатчик:

```
        L  KB 255       Номер буфера связи
        L  KB 53        Адрес координирующей ячейки
        SPA OB 218      Передача метки места процессора в буфер
                        связи
        SPB =M001       Если VKE=1 (передача успешная), переход
                        к указанной метке.
        BEA             Иначе – конец процедуры
M001 :  A  DB 45        Открытие блока данных источника
        L  KY 2/255     2 = признак двойного слова для буфера
                        связи
        L  KB 54        Начальный адрес в буфере связи
        ENT             Запись из АКК 3
        L  DD 4         Слова 4 и 5 (= 4 байта)
        SPA OB 216      Передача первого двойного слова
                        Адрес увеличивается на 4 (АКК 2-L = 58)
        TAK             Сохранение адреса (цели)
        L  DD 6
        SPA OB 216      Передача второго двойного слова
        TAK
        L  DD 8
        SPA OB 216      Передача третьего двойного слова
        TAK
        L  DD 10
        SPA OB 216      Передача четвертого двойного слова
        TAK
        L  KY 0.255
        L  KB 53        Адрес метки места установки процессора
        ENT
        L  KB 0         АКК 1 = 0
        SPA OB 216      Метка места установки процессора
                        стирается → область для передачи данных
                        свободна
        BE
```

Приемник:

	L KB 255	Номер буфера связи
	L KB 53	Координирующая ячейка
	SPA OB 218	Захватка буфера связи через второй процессор
	SPB =M002	Если VKE = 1, переход на метку
	BEA	
M002222	AX DX 50	Блок данных = назначение
	L KY 2.255	
	L KB 54	
	ENT	Загрузка АКК 3
	L KB 0	Загрузка АКК 2
	SPA OB 217	Чтение первого двойного слова Адрес увеличивается на 4 (АКК 2-L = 56)
	T DD 0	Передача из АКК 1 слова данных 0 и 1
	SPA OB 217	Чтение второго двойного слова
	T DD 2	
	SPA OB 217	Чтение третьего двойного слова
	T DD 4	
	SPA OB 217	Чтение четвертого двойного слова
	T DD 6	
	L KY 0/255	
	L KB 53	Адрес с меткой места установки процессора
	ENT	
	L KB 0	АКК 1 = 0
	SPA OB 216	Стирание метки места установки процессора → освобождение области передачи данных
	BE	

6.7. Расширение знака (OB220)

Эта функция расширяет знак 16-битового числа с фиксированной запятой. Находящегося в АКК 1, на полуслово, находящееся в АКК 1-Н.

Если бит 2^{15}= 0 (положительное число), то в АКК 1-Н идет загрузка КН = 0000Н.

Если бит 2^{15} = 1 (отрицательное число), то в АКК 1-Н идет загрузка КН = FFFFH.

Это расширение знака необходимо для того, чтобы отрицательные 16-битовые числа с фиксированной запятой перед преобразованием в 32-битовые числа с плавающей запятой (команда FDG) представлялись в виде 32 разрядного числа с фиксированной запятой.

Время выполнения OB220: 16 Мкс.

Параметры:

1. АКК 1-L : 16-Битовое число с фиксированной запятой.

Ошибки: Отсутствуют.

6.8. Системные функции

6.6.1. OB 120: Включение /выключение блокировки всех прерываний и OB 122: Включение/выключение задержки всех прерываний

Программа STEP5 может быть прервана на границе блока или на границе команды программой с более высоким приоритетом. К таким более приоритетным уровням обработки программ принадлежат прерывание от процесса и прерывание по времени. Выполнение прерванной программы продолжается при этом после выполнения этой вложенной программы.

С помощью организационных блоков спецфункций OB 120 и OB 122 Вы можете блокировать прерывание от процесса и прерывание по времени на одной или нескольких, идущих друг за другом, границах блоков или команд.

OB 120: Включение /выключение блокировки всех прерываний

Этот организационный блок спецфункций имеет влияние на прием прерывания.

«Блокировка прерывания включена» означает, что начиная с этого момента прерывания больше не регистрируются и соответствующие прерывания уже были зарегистрированы (например, ожидают на границе блока) стираются. Только в случае OB 2 (прерывание от процесса) или OB 13 (прерывание по времени) если они уже начались, то эти прерывания обрабатываются полностью.

«Блокировка прерывания выключена» означает, что с этого момента уже возникающие прерывания регистрируются и будут вызваны и обработаны на следующей границе блока или команды.

OB 122: Включение/выключение задержки всех прерываний

Этот организационный блок спецфункций имеет влияние на обработку прерывания.

“Задержка прерывания включена” означает, что все возникающие в последствии прерывания остаются зарегистрированными. Обработка зарегистрированных прерываний, однако, не начинается сразу. *) Проход всех границ команд или блоков для обработки прерываний становится недействительным. Только если уже была начата обработка OB 2 (прерывание от процесса) или OB 13 (прерывание по времени), то они обрабатываются полностью.

«Задержка прерывания выключена» означает, что все зарегистрированные прерывания будут вызваны и обработаны на следующей границе блока или команды.

*) Если внутри фазы «Задержка прерываний» некоторый OB прерывания по времени вызван повторно, то возникает ошибка будильника.

OB 120 и OB 122 помечают блокированные или задержанные прерывания в управляющем слове со следующими значениями:
Бит 2^0: Прерывание по времени

Бит 2^2: Прерывание от процесса

Бит 2^1, 2^3 по 2^{31}: зарезервированы: эти биты должны быть равны 0!

Параметры:

1. AKKU 2-L: Метка функции
   Возможные значения : 1, 2 или 3
2. AKKU 1: Новое управляющее слово или маска

AKKU 2-L	AKKU 1 до	AKKU 1 после	Функция
1	управляющее слово	управляющее слово	**Содержимое аккумулятора 1 загружается в управляющее слово**
2	маска	новое управляющее слово	В маске аккумулятора 1 все биты установленные в «1», устанавливаются в «1» и в управляющем слове. Новое управляющее слово загружается в аккумулятор 1.
3	маска	новое управляющее слово	В маске аккумулятора 1 все биты установленные в «0», устанавливаются в «1» и в управляющем слове. Новое управляющее слово загружается в аккумулятор 1.

Случаи ошибок:
- Недопустимая метка функции в аккумуляторе AKKU 2-L

- В аккумуляторе 1 установлен в «1» один из битов от 2^0 до 2^{31}.

В случае ошибки вызывается OB 31 и в аккумулятор AKKU 1 передается метка ошибки:

1F47H	при OB 120
1A48H	при OB 122

Указания:

- Состояние управляющего слова опрашивается следующим способом:
  1. Загрузить в аккумулятор 2-L метку функции 2 или 3
  2. Загрузить в аккумулятор 1 значение «0»
  3. Вызвать спецфункцию OB 120, 122
  4. Считать аккумулятор 1
- Состояние обработки прерывания может быть определено путем считывания системных слов данных BS 131 и BS 132.

  BS 131 управляющее слово блокировки всех прерываний (OB 120)
  BS 132 управляющее слово задержки всех прерываний (OB 122)

- Для блокировки и деблокировки прерывания от процесса можно использовать также в случае OB 120 команды AS и AF:

AS	соответствует	:L KB2
		:L KM00000000 00000100
		:SPA OB120
AF	соответствует	:L KB3
		:L KM00000000 00000100
		:SPA OB120

6.8.2. OB 121: Включение /выключение блокировки отдельных прерываний по времени и OB 123: Включение/выключение задержки отдельных прерываний по времени

С помощью организационных блоков спецфункций OB 121 и OB 123 Вы можете блокировать отдельные прерывания по времени на одной или нескольких, идущих друг за другом, границах блоков или команд. Например, Вы можете для определенной части программы указать, что не выполнение не может быть прервано OB 10 (10мс) и OB 11 (20мс). Все остальные запрограммированные прерывания по времени действуют.

OB 121: Включение /выключение блокировки отдельных прерываний по времени

Этот организационный блок спецфункций имеет влияние на прием прерывания.

«Блокировка отдельных прерываний по времени включена» означает. Что начиная с этого момента указанные прерывания больше не регистрируются и соответствующие прерывания уже были зарегистрированы (например. Ожидают на границе блока) стираются, только если уже началась обработка OB прерывания по времени, то это прерывание обрабатывается полностью.

«Блокировка отдельных прерываний по времени выключена» означает, что с этого момента все возникающие прерывания по времени регистрируются и будут вызваны и обработаны на следующей границе блока или команды (в зависимости от предустановки в DX 0).

OB 123: Включение/выключение задержки отдельных прерываний по времени

Этот организационный блок спецфункций имеет влияние на обработку определенных прерываний по времени.

«Задержки отдельных прерываний по времени включена» означает, что все возникающие впоследствии прерывания регистрируются и уже имеющиеся прерывания остаются зарегистрированными. Обработка указанных в управляющем слове прерываний по времени, однако, не начинается сразу. Проход всех границ команд или блоков для обработки прерываний становится недействительным. Только если уже была начата обработка данного OB прерывания по времени, то он обрабатывается полностью.

«Задержки отдельных прерываний по времени выключена», означает, что все зарегистрированные прерывания будут вызваны и обработаны на следующей границе блока или команды (в зависимости от предустановки DX 0).

OB 121 и OB 123 помечают блокированные или задержанные прерывания по времени в управляющем слове со следующими значениями:

Бит	2^0	:	Должен быть равен «0»!
Бит	2^1	:	Должен быть равен «0»!
Бит	2^2	:	Должен быть равен «0»!
Бит	2^3	:	Прерывание по времени 10 мс (OB10)
Бит	2^4	:	Прерывание по времени 20 мс (OB11)
Бит	2^5	:	Прерывание по времени 50 мс (OB12)
Бит	2^6	:	Прерывание по времени 100 мс (OB13)
Бит	2^7	:	Прерывание по времени 200 мс (OB14)
Бит	2^8	:	Прерывание по времени 500 мс (OB15)
Бит	2^9	:	Прерывание по времени 1 с (OB16)
Бит	2^{10}		Прерывание по времени 2 с (OB17)
Бит	2^{11}	:	Прерывание по времени 5 с (OB18)
Бит	2^{12}	:	Должен быть равен «0»!
Бит	2^{13}	:	Должен быть равен «0»!
Бит	2	:	Должен быть равен «0»!
Бит	2	:	Должен быть равен «0»!

Пока бит установлен в “1”, соответствующее прерывание по времени блокировано или задержано.

<u>Параметры:</u>

1. AKKU 2-L: Метка функции возможные значения: 1, 2 или 3
2. AKKU 1: Новое управляющее слово или маска

АККU 2-L	АККU 1		Функция
	до	после	
1	Управляющее слово	Управляющее слово	Содержимое аккумулятора 1 загружается в управляющее слово
2	маска	Новое управляющее слово	В маске аккумулятора 1 все биты установленные в «1», устанавливаются в «1» и в управляющем слове. Новое управляющее слово загружается в аккумулятор 1.
3	маска	Новое управляющее слово	В маске аккумулятора 1 все биты установленные в «1», устанавливаются в «0» и в управляющем слове. Новое управляющее слово загружается в аккумулятор 1.

Случаи ошибок:

- Недопустимая метка функции в аккумуляторе АККU 2-L
- В аккумуляторе 1 установлен в «1» один из зарезервированных битов.

В случае ошибки вызывается ОВ 31 и в аккумулятор АККU 1 передается метка ошибки:

1A47H при ОВ 121
1A48H при ОВ 123

Указания:

- Состояние управляющего слова опрашивается следующим способом:
  1. Загрузить в аккумулятор 2-L метку функции 2 или 3
  2. Загрузить в аккумулятор 1 значение «0»
  3. Вызвать спецфункцию ОВ 120, 122
  4. Считать аккумулятор 1

- Состояние обработки прерывания по времени может быть определено путем считывания системных слов данных BS 135 и BS 137.

BS 135 управляющее слово блокировки отдельных прерываний по времени (ОВ 121)
BS 137 управляющее слово задержки отдельных прерываний по времени (ОВ 123)

6.8.3. Установка времени цикла (OB221)

С помощью этой спецфункции Вы можете изменить контролируемый интервал времени цикла (максимальное время цикла), устанавливаемого по умолчанию (стандартное соглашение) на 150 мс.
Одновременно с вызовом таймер контроля времени цикла запускается заново: актуальный цикл, т. е. цикл, в котором OB221 вызывается первый раз. Удлиняется на вновь устанавливаемое значение, отсчитанное от момента вызова спецфункции. Время контроля последующих циклов равно вновь установленному значению (= значению времени. Находящемуся в АКК 1).

Параметры:

1. АКК 1: Любое время цикла (в миллисекундах) возможные значения: 1…6000 мсек.

Ошибки:

- Заданное время цикла выходит за пределы 1 мсек…6000мсек.

Замечания:

OB221 работает с S-процессором и не должен программироваться в CPU928. Вместо этого необходимо параметрировать эту функцию системы в DX0 (смотри Раздел 7).

6.8.4. Новый запуск времени цикла (OB222)

Специальная функция OB222 сбрасывает триггер контроля времени цикла, т.е. вновь запускает таймер при помощи вызова этой функции. Максимально допустимое время цикла увеличивается от момента вызова функции на заданное значение – стандартное (150 мсек) или установленное в DX0.

Параметры: Отсутствуют

Ошибки: Отсутствуют

6.8.5. Проверка видов запуска процессоров (OB223)

Вызов OB223 – например, при запуске или в начале циклической обработки программы – позволяет проверить в многопроцессорном режиме работы контроллера, все ли процессоры имеют одинаковые установленные режимы запуска.

Если это не так. То тот процессор, в котором вызвана спецфункция OB223, опознает ошибку выполнения. Вызывается OB 31. В AKKU 1 находится метка ошибки 1A3B (16-рич.). Если OB 31 не запрограммирован, то процессор переходит в состояние останова с сообщением об ошибке LZF. Его светодиод СТОП медленно мигает. Остальные процессоры переходят в стоп с постоянным свечением светодиода СТОП.

Параметры: Отсутствуют

Ошибки: Отсутствуют

6.8.6. Передача маркера межпроцессорной связи в блок (OB224)

В многопроцессорной системе заданный маркер межпроцессорной связи передается в DB1 тогда, когда процессор получает сигнал о том, что он может обратиться к шине периферийных устройств.

Если нужно организовать одновременную работу нескольких процессоров, то координатор посылает по очереди каждому процессору сигнал о том, что шина свободна. При этом процессор может передать только один байт. В следствие такой квантованной передачи может получиться так, что взаимозависимая информация может быть при передаче разорвана и в результате будет обработана неверно.

Используя OB224, Вы добьетесь того, что все находящиеся в DB1 процессора маркеры связи будут переданы как единый блок: пока процессор занят передачей маркеров связи, он не может быть прерван другим процессором, т.к. другой процессор должен ждать своей очереди на передачу данных, время выполнения программ замедляется (время цикла!).

Организационный блок спецфункции OB224 гарантирует логическую неразрывность всей информации, заданной в маркерах связи. Он может быть вызван в программе запуска, а именно:
а) во всех отдельных процессорах при передаче маркеров связи.
б) при каждом запуске..

Параметры: Отсутствуют

Ошибки: Отсутствуют

Указание:

Спецфункция OB224 работает с S–процессором и не должна программироваться в CPU928. Вместо этого, необходимо параметрировать эту системную функцию в DX0 (смотри Раздел7).

6.8.7.Чтение слова из системной программы (OB226)

Системная программа процессора имеет длину до 64 Кслов и находится в памяти, к которой программа STEP5 доступа не имеет. Доступ к этой области памяти возможен только через OB226.

Параметры:
1. АКК 1-L: Адрес читаемой ячейки памяти

Ошибки: Отсутствуют

После вызова OB226:

- считанное слово находится в АКК 1, смещенное вправо,
- остальное содержимое в АКК 1 сбрасывается.
- предыдущее содержимое АКК 1 (т.е. адрес слова) переписывается в АКК 2.
- предыдущее содержимое АКК 2 теряется.

Перед использованием OB226 ознакомьтесь с описанием OB227 и примера программы.

6.8.8. Считывание контрольной суммы системной программы (OB227)

OB227 передает контрольную сумму из области системной программы в аккумулятор АКК 1.

Параметры: Отсутствуют

Ошибки: Отсутствуют

После вызова OB227

- в АКК 1 находится контрольная сумма, смещенная вправо.
- остальное содержимое АКК 1 сбрасывается.
- предыдущее содержимое АКК 1 переписывается в АКК 2.
- предыдущее содержимое АКК 2 теряется.

Применение:

Вы можете во время циклической обработки программы контролировать содержимое системной программы, а именно

- читать содержимое отдельных ячеек системной программы,
- суммировать эти данные с помощью команды +F (сложение чисел с фиксированной запятой),
- с помощью OB227 читать контрольную сумму и затем
- полученную с помощью суммирования контрольную сумму сравнить с прочитанной.

Пример программы

FB 111

```
Имя     :CHECKSUM
        :L  KH0000
        :T  MW254       Очистка маркера контрольной суммы
        :T  MW252       Очистка счетчика адресов
        :
M001    :L  MW252       Загрузка адреса читаемой ячейки
        :SPA OB226      Чтение слова
        :L  MW254       Загрузка маркера контрольной суммы
        :+F             Сложение
        :T  MW254       Запоминание маркера контрольной суммы
        :
        :L  MW252       Увеличение счетчика адресов
        :L  KF+1
        :+F
        :T  MW252
        :
        :L  KH000       Счетчик адресов не равен “0” ?
        :><  F
        :SPB =M001      Переход на метку M001
        :
        :SPA OB227      Если счетчик адресов=0, читаем контр. сумму
        :L  MW254       Загрузка маркера контрольной суммы
        :!=F            Если они не равны, то конец программы
        :BEB
        :
        :STP            Если не равны, то останов процессора
        :BE
```

6.8.9. Чтение информации о состоянии на уровне выполняемой программы

При наступлении определенных событий системная программа переводит вычислительный процесс на соответствующий этому событию уровень обработки программ. Тем самым данный уровень становится “активным”.

С помощью OB228 Вы можете только установить, является ли в данный момент определенный уровень активным или нет. В АКК 1 Вы заносите номер того уровня, статус которого хотите определить (номер соответствует номеру, занесенному в USTACK как “EBENE” (уровень)).
При вызове функции информация о статусе передается в АКК 1.

Параметры:
1. АКК 1-L: Номер уровня (смотри USTACK , EBENE)

возможные значения (шестнадцатеричные):

02	-	Новый пуск
04	-	Цикл
06	-	Прерывание по времени
08	-	Прерывание по времени
0A	-	Прерывание по времени
0C	-	Прерывание по времени
0E	-	Прерывание по времени
10	-	Прерывание по времени
12	-	Прерывание по времени
14	-	Прерывание по времени
16	-	Прерывание по времени
18	-	Не используется
1A	-	Не используется
1C	-	Регулятор
1E	-	Не используется
20	-	Не используется
22	-	Не используется
24	-	Прерывание от процесса
26	-	Не используется
28	-	Не используется
2A	-	Не используется
2C	-	Прерывание
2E	-	Не используется
30	-	Ошибка «будильника»
32	-	Ошибка регулятора
34	-	Ошибка цикла
36	-	Не используется
38	-	Ошибка кода команды
3A	-	Ошибка выполнения
3C	-	Ошибка адресации
3E	-	Задержка квитирования
40	-	Не используется
42	-	Не используется
44	-	Ручной перезапуск
46	-	Автоматический перезапуск

Ошибки: Отсутствуют

После вызова OB228

- **в АКК 1 информация о статусе:**
  содержимое АКК 1 = 0: уровень обработки программы не вызван
  содержимое АКК 1 не 0: программный уровень активирован
- в АКК 2 содержится предыдущее содержимое из АКК 1
- предыдущее содержимое АКК 2 теряется

Тем самым вы можете выполнять обработку в своей программе в зависимости от состояния другого уровня

Пример:

Ошибка квитирования QVZ должна указываться äî время запуска и обрабатываться на других уровнях.

В начале выполнения OB23 вызовите OB226, чтобы установить, произошла ли ошибка QVZ на уровне NEUSTART (новый пуск (номер 02)) или нет. Дальнейшую обработку программы выполняйте в зависимости от полученной информации

АКК 1 равен 0: NEUSTART — пасивен —> ошибка QVZ была не на уровне NEUSTART

—> нужно вызвать программу обработки ошибок

АКК 1 не равен 0: NEUSTART — активен —> ошибка QVZ была на уровне NEUSTART

—> QVZ игнорируется

OB228 позволит вам проводить дифференцированную обработку ошибок.

6.9. Функции для стандартных функциональных блоков (OB230…OB137)

Организационные блоки специальных функций OB230…OB237 зарезервированы для функций управления и могут вызываться только из стандартных функциональных блоков FB120…FB127

Если стандартные функциональные блоки, которые называют также «подпрограммами управления», управляют в одно- или многопроцессорной передачей данных через буфер связи: они применяются, если необходимо передать данные, параметры или управляющую информацию коммуникационному процессору или прочитать эту информацию.

Соответствие

Стандартные функциональные блоки	Организационные блоки специальных функций	Блоки управлений
FB 120	SF – OB 230	SEND
FB 121	SF – OB 231	RECEIVE
FB 122	SF – OB 232	FETCH
FB 123	SF – OB 233	CONTROL
FB 124	SF – OB 234	RESET
FB 125	SF – OB 235	SYNCHRON
FB 126	SF – OB 236	SEND ALL
FB 127	SF – OB 237	RECEIVE ALL

При использовании управляющих программ, которые поставляются как программный продукт на дискете, следует пользоваться подробным руководством с названием «Программируемый контроллер S5 – 135U блоки управления для R-процессора и CPU928» (AUTOMATISIERUGSGERAET S5135U HANTIERUNGSBAUSTEIN FUER R-PROCESSOR UND CPU928) (Каталоговый номер: C79000-B8500-C366-02).

6.10. Регистры сдвига

Программный регистр сдвига состоит из смежных восьмибитовых ячеек и может иметь длину от 2 до 256 ячеек.
Данные регистра сдвига находятся в RAM-блоках данных процессора. Каждый регистр сдвига имеет жестко закрепленный за ним блок данных: оба имеют один и тот же номер (допускается: 192 до 255). Если вы, например, установили регистр сдвига с номером 210, то принадлежащие ему данные находятся в блоке данных DB210.
Область DB-RAM занимает 23К слов (адреса KH 8000…KH DD7F). В этой области находятся блоки данных, копируемые с помощью OB254 и 255 (с адреса KH 8000), а также блоки данных, которые используются как регистры сдвига (с адреса KH DD7F). В случае, когда при копировании блока данных или при создании регистра сдвига памяти не хватает, процессор опознает ошибку выполнения и вызывает OB31.
Последующая за этим реакция зависит от того, запрограммирован ли OB31 (см. «Особые ошибки выполнения»)
Вы можете как записывать данные в регистр сдвига, так и читать их. Это определяется т.н. «указателем»: «указатель – это маркерный байт, в котором записано содержимое отдельных ячеек сдвигового регистра.

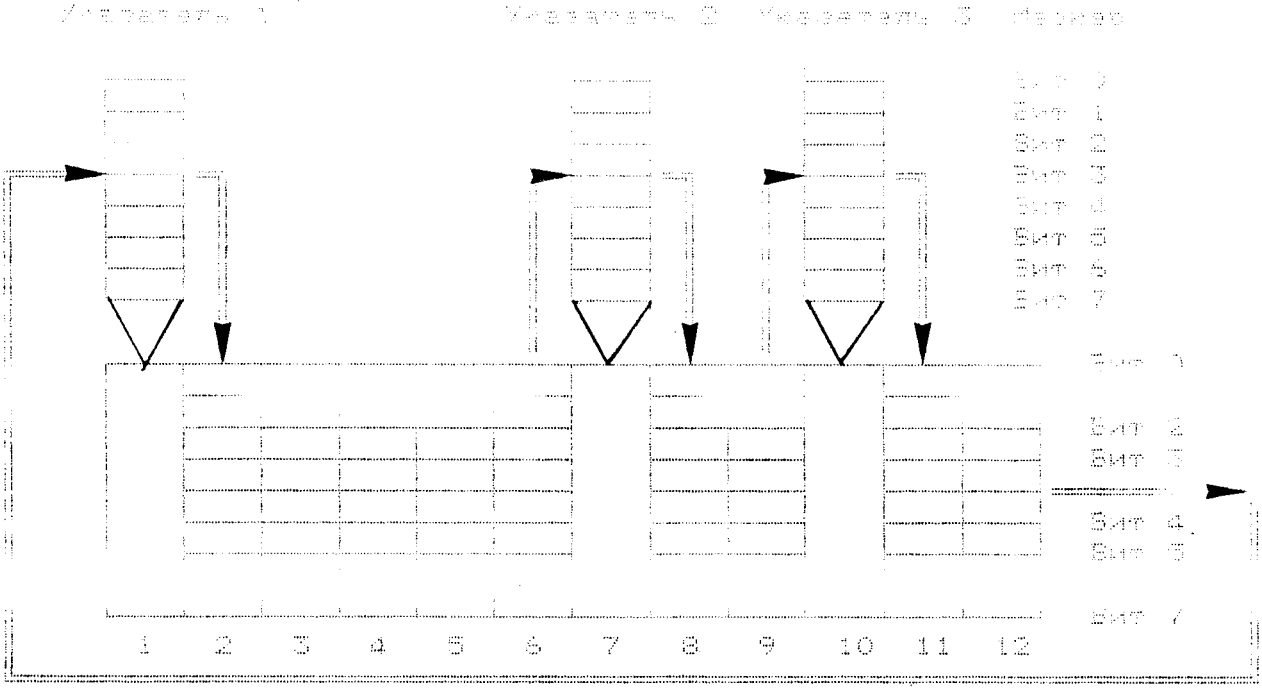


Рис. 6 – 1 **Принципиальная диаграмма сдвигового регистра с 3 указателями и 12 ячейками хранения.**

Первый указатель (базовый указатель) установлен постоянно в первой ячейке сдвигового регистра. Номер этого маркерного байта устанавливается пользователем. Все остальные указатели могут располагаться пользователем относительно базового указателя произвольно. Количество указателей выбирается пользователем от одного (базовый) до шести.

При работе регистра сдвига – как и для аппаратного регистра сдвига – информация побайтно передается из одной ячейки в другую (см. рисунок). Каждый вызов функции сдвигового регистра перемещает информацию точно на одну ячейку (= 1 такт). В указатель при этом попадает новая информация. Показанные на рисунке столбцы информации сдвигаются соответственно в одну сторону до последней ячейки, откуда они опять попадают в 1-ю ячейку (в приведенном примере это произойдет через 12 тактов).

Пример:

На следующем рисунке показано перемещение информации внутри регистра сдвига.

Перед вызовом регистра сдвига в маркерные биты указателей заносится:

Установить бит маркера 0 указателя 1	:S M 0.0
Установить бит маркера 3 указателя 2	:S M 1.3
Установить бит маркера 2 указателя 3	:S M 2.2

Потом вызывают функцию сдвига :SPA OB 24

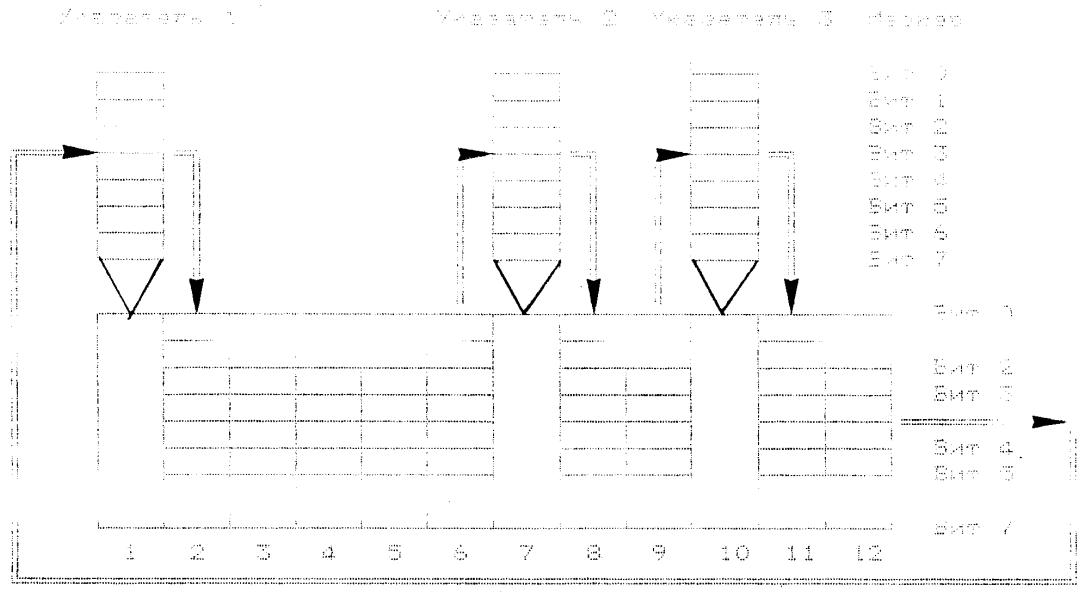


Рис. 6.2. Схематическая диаграмма сдвигового регистра с 3 указателями и 12 ячейками хранения перед первым тактовым импульсом.

После вызова спецфункции 8-битовая информация переместится на одну ячейку:

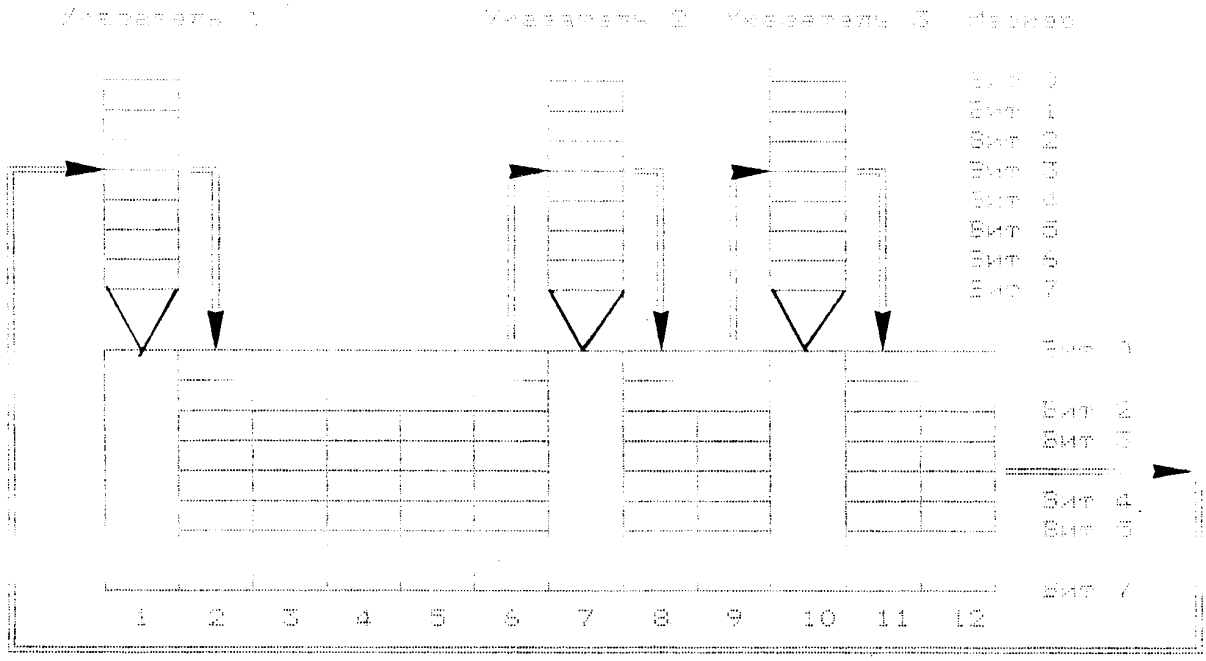


Рис. 6.3. Схематическая диаграмма сдвигового регистра с 3 указателями и 12 ячейками хранения после первого тактового импульса.

Информацию, которая находится теперь в указателях, Вы можете теперь проанализировать с помощью команды :L MB 0.

Маркерные биты 0, 3 и 2 можно спросить из базового регистра:
Всю информацию, которая заносится через любой из регистров, можно прочитать в базовом регистре (в примере это можно сделать через 12 тактов).

Если Вы применяете регистр сдвига, то пользуйтесь 3-мя спецфункциями:

OB 240: Инициализация регистра сдвига.

OB 241: Обработка регистра сдвига.

OB 242: Стирание регистра сдвига.

6.10.1. Инициализация регистра сдвига (ОВ 240)

Перед работой с регистром сдвига необходимо его инициализировать. Для этого используют специальную функцию ОВ 240.

Параметры, которые необходимы ОВ 240 для работы с регистром сдвига помещаются в блоке данных с номером инициализируемого с регистром сдвига. Допускается использование блоков данных ОВ 192 . . . 255.

Блок данных строится по определенной схеме, которой необходимо придерживаться. Длина его может быть не более 9 слов данных (DW0 . . .DW8).

В словах должна содержаться следующая информация:

0	DW 0
Длина регистра сдвига	DW 1
Номер первого байта маркеров	DW 2
Расстояние n_2	DW 3
Расстояние n_3	DW 4
Расстояние n_4	DW 5
Расстояние n_5	DW 6
Расстояние n_6	DW 7
0	DW 8

Рис. 6-4. Конфигурация блока данных для инициализации регистра сдвига.

Слово данных 0: Всегда содержит нули.

Слово данных 1: Длина регистра сдвига в виде числа (в байтах) ячеек памяти, отведенных под регистр сдвига. Оно может быть в пределах $2 < L < 256$.

Слово данных 2 : Содержит номер первого маркерного байта, устанавливающего базовый указатель регистра сдвига и, тем самым, блок маркеров, в котором упорядочены указатели. Если Вы задаете, например, два указателя, то это составит вместе с базовым три указателя. Тогда они указываются в блоке данных и резервируются два следующих друг за другом маркерных байта.
Следует предусмотреть, чтобы до конца маркерной области было достаточно места для всех задаваемых указателей.

Слова данных 3…7: Смещение (в байтах) указателей относительно базового:

n_2 = смещение указателя 2
n_3 = смещение указателя 3
n_4 = смещение указателя 4
и т. д. (максимум 5)

Слово данных последнего смещения: Слово данных после последнего смещения (в примере DW8) должно всегда содержать нули.

Если кроме базового требуется еще только 2 указателя, то в каждом следующем слове начиная с DW5, должны быть нули.

Все данные задаются в формате с фиксированной запятой.

Важно!
- **Число указателей (включая базовый) не должно превышать длины регистра.**
- **Смещение указателя относительно базового не должно быть больше длины регистра.**
- **DW0 и слово, следующее за последним смещением, должны быть равны 0.**
- **Блок данных перед вызовом OB 240 должен быть запрограммирован и вызван.**

На основании информации из этого блока данных резервируется и инициализируется определенная область памяти в конце памяти RAM блоков данных (DB-RAM).

Место в памяти:

Для каждого регистра сдвига требуется

$$n = L / 2 – 8 \text{ слов данных,}$$

т.е. длина DB-RAM сдвигается к младшим адресам.

Если регистр сдвига, который должен быть инициализирован, уже имеется, то, при одинаковой длине, новый и уже имеющийся регистры сдвига занятую область инициализируют заново. В противном случае уже занятая область объявляется недействительной. И открывается новая область.

Параметры:

1. Открываемый блок данных с номером от 192 до 255.

Ошибки:

- Недопустимый номер блока данных (<192, >255)
- Имеющейся памяти недостаточно
- Формальная ошибка в построении блока данных
- Ввод недопустимой длины регистра сдвига
- Ошибка параметрирования указателей

В случае ошибки процессор идентифицирует ее как ошибку выполнения и вызывает ОВ 31. Дальнейшая реакция зависит от того, запрограммирован ли ОВ 31 (смотри “Особые ошибки выполнения”).
Если ОВ 31 не запрограммирован, процессор переходит в “СТОП” В АКК 1 содержится признак последней ошибки.

6.10.2. Обработка регистра сдвига (OB 241)

OB 241 обрабатывает предварительно инициализированный OB 240 регистр сдвига.

В CPU928 можно вызвать максимум 64 регистра сдвига.

Параметры:

1. АКК 1-L: Номер регистра сдвига. Допустимые значения: 192 – 255

Перед вызовом OB 241 обычно в указателях устанавливается маркерные биты.

При каждом вызове OB 241 информация побайтно перемещается от ячейки к ячейке в сторону возрастания номеров ячеек. При этом в указатели попадает новая информация. Путем повторного вызова OB 241 можно переместить информацию через весь регистр по последней ячейке. Откуда она опять попадает в первую ячейку.

После вызова OB 241 указатели (максимум 6 на каждый регистр сдвига с любым позиционированием относительно базового) содержат информацию из предыдущих ячеек памяти. Эту информацию можно проанализировать.

Ошибки:

- Недопустимый номер регистра сдвига в АКК 1
- Регистр сдвига не инициализирован

В случае ошибки процессор идентифицирует ее как ошибку выполнения и вызывает OB 31. Дальнейшая реакция зависит от того, запрограммирован ли OB 31 (см. «Особые ошибки выполнения»).
Если OB 31 не запрограммирован, процессор переходит в «СТОП». В АКК 1 содержится признак последней ошибки.

6.10.3. Стирание регистра сдвига (OB 242)

С помощью этой функции регистр сдвига «очищается», т.е. стирается список адресов DB0. И соответствующий регистр сдвига в DB-RAM объявляется недействительным. (Внимание: даже стертый регистр сдвига занимает место в памяти).

Параметры:

1. АКК 1-L: Номер стираемого регистра сдвига, допустимые значения: 192…255

После вызова OB 242 регистр сдвига стирается, и его нельзя использовать: если в дальнейшем придется работать с регистром сдвига, то необходимо вновь его инициализировать.

Ошибки:

- Недопустимый номер регистра сдвига в АКК 1
- Регистр сдвига не инициализирован

В случае ошибки процессор идентифицирует ее как ошибку выполнения и вызывает OB 31. Дальнейшая реакция зависит от того, запрограммирован ли OB 31 (см. «Особые ошибки выполнения»).
Если OB 31 не запрограммирован, процессор переходит в «СТОП». В АКК 1 содержится признак последней ошибки.

6.11. Регулирование: PID-алгоритмы

Эта глава предназначена для пользователей, которые собираются работать с PID-алгоритмами!
Для других пользователей эта глава не обязательна!

В CPU 928 ПК 135U можно вызывать один или несколько PID-регуляторов.

Каждый регулятор должен быть инициализирован с помощью вызова соответствующей подпрограммы. Для передачи параметров используется блок данных.

Текущий алгоритм управления встроен в операционную систему и может быть вызван пользователем только через OB-подпрограмму. В качестве интерфейса данных между алгоритмом управления и программой пользователя используется блок данных.

Функциональное описание PID-регулятора

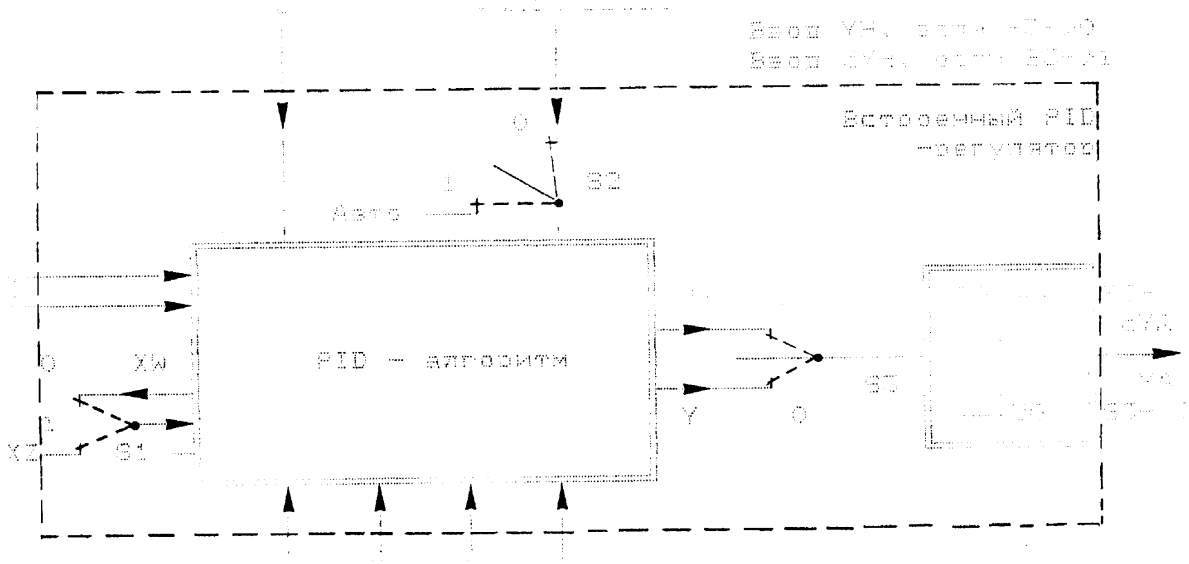


Рис. 6-5. Блок – схема PID-регулятора

Индекс k: k-опрос

Переключатель	Положение	Эффект
S1 STEU бит 1	 1	Дифференциатор выполняет дифференциацию регулировки XW_k. Через XZ может быть подан другой сигнал
S2 STEU бит 0	0 1	Ручное управление Автоматическое управление
S3 STEU бит 3	0 1	Алгоритм управления положением Алгоритм управления скоростью
S4 STEU бит 4	0 1	С прямым управлением Без прямого управления

Функции, относящиеся к положению переключателей на этой блок-схеме, устанавливаются с помощью параметрирования (установки) соответствующих битов управления в слове управления STEU PID-регулятора. Регулятор предназначен для управления высокоскоростными процессами регулирования, например, давления, температуры или потока.

Сам регулятор основывается на PID-алгоритме. Выходной сигнал может быть выдан в виде управляемой переменной (алгоритм управления положением) или в виде управляемой скорости изменения переменной (алгоритм управления скоростью).

Отдельные компоненты P, I и D могут быть отключены установкой соответствующих параметров R, TI и TD в нулевое значение. Это означает, что легко может быть получена требуемая структура системы регулирования – PI, PID, PD.

В блоке отклонений могут использоваться входной сигнал XW или XZ, поправка или инвертированное текущее значение.

Для компенсации влияния изменений осуществляется управление скоростью подачи без контроля времени. Сигнал помехи Z, связанный с процессом, подается для алгоритма управления. При ручном управлении в эту точку подается предварительно устанавливаемая переменная YN (см. таблицу значений управляющего слова STEU дальше в тексте).

Если требуется изменение направления, следует устанавливать отрицательное значение параметра K.

Если информация по управлению (dY или Y) на пределе, компонента I выключается автоматически для того, чтобы избежать ухудшения отклика регулирования.

Программа регулирования может содержать фиксированные значения или адаптивные (динамические) параметры (K, R, TI, TD). Они вводятся через установку в памяти отдельных параметров.

PID-алгоритм

PID-регулятор основан на алгоритме управления скоростью, согласно которому производится вычисление соответствующего управляющего приращения dYk в отдельной точке во время t=k TA по формуле:

$$dYk = K[(XW_k - XW_{k-1})R + \frac{TA}{2TN}(XW_k + XW_{k-1}) +$$

$$\frac{1}{2}\left\{\frac{TV}{TA}(XU_k - 2XU_{k-1} + XU_{k-z}) + dD_{k-1}\right\}]$$

$$= K\,(dPW_kR + dI_k + dD_k)$$

P-часть I-часть D-часть

DXXXk: Изменение величины XXX во время t.

U может быть W или Z, в зависимости от того, XW или XZ подключено к отклоняющему входу. Выполняется следующее:

При подключении XWk :	При подключении XZ:
PWk = Wk – Xk	
PWk = XWk – XWk-1	PZk = XZk – XZk-1
QWk = PWk – PWk-1	QZk = PZk – PZk-1
QWk = XWk – 2XWk-1 + XWk-2	QZk = XZk – 2XZk-1 + XZk-2

$$dPWk = [(XWk - XWk\text{-}1)R$$

$$dI = TI\ XWk \qquad TI = \frac{TA}{TN}$$

$$dD\ k = \tfrac{1}{2}(TD\ QU\ k + dD\ k\text{-}1) \qquad TD = \frac{TV}{TA}$$

Если управляемая переменная Yk используется в качестве управляющего выхода во время tk, то она формируется в соответствии с формулой:

$$Yk = \sum_{m=0}^{m=k} dYm$$

В большинстве проектируемых процессов управления можно допустить, что R=1, если требуется реакция P.
Величина R может быть использована для установки пропорциональной компоненты PID-регулятора.

Блок данных для PID-регулятора

Данные для регулятора вводятся с использованием специального блока данных (для инициализации и вызова PID-регулятора смотри Разделы 6.11.1 и 6.11.2).

Пользователь должен ввести следующие данные в передаваемый блок данных:

K, R, TI, TD, W, STEU, YH, BG0G, BGUG

Структура передаваемого блока, который состоит из 49 слов данных с номерами с 0 до 48, более детально описана ниже.

Структура передаваемого блока данных

Адрес в DB	Имя	Вх/вых	Формат числа	Формат PG	Примечание
DW 0	-	-	-	-	ЗАРЕЗЕРВИРОВАНО
DW 1	K	вх	GP	KG	Коэффициент пропорциональности K > 0: Положительное направление управления. т.е. изменение установки и изменяемой переменной в одинаковых направлениях. K < 0: Отрицательное направление управления: В пределах числа с плавающей запятой.
DW 3	R	вх	GP	KG	R параметр, обычно = 1 Для регуляторов с компонентой P: В пределах числа с плавающей запятой.
DW 5	TI	вх	GP	KG	TI = TA/TN: В пределах числа с плавающей запятой
DW 7	TD	вх	GP	KG	TD = TV/TA: В пределах числа с плавающей запятой
DW 9	W_1	вх	GP	KG	Здесь вход точки установки, Если STEU бит 6 = 1, в противном случае вход в слове №. 19 ($-1 < W_k < 1$)
DW 11	STEU	вх	BM	KM	Управляющее слово
DW 12	$YN_к$	вх	GP	KG	Здесь вводятся вручную данные для аппаратуры, если STEU бит 6 = 1; в противном случае вход в слове № 18 ($-1 < YH_k < 1$). При алгоритме управления скоростью здесь должен быть указан инкремент управляющей переменной.
	BGOG	вх	GP	KG	Значение верхнего предела $0 < BGOG < 1$ (YA_k макс.) ! ! BGUG < BGOG ! !
	BGUG	вх	GP	KG	Значение нижнего предела $-1 < BGUG < 0$ (YA_k мин.)
DW 16					

Адрес В DB	Имя	Вх/вых	Формат числа	Формат PG	Примечание
DW 18	Yhk	вх	LP	KF	Здесь ручной ввод, если STEU бит 6 = 0 ($-1<W_k<1$) При алгоритме управления скоростью здесь должен быть указан инкремент управляющей переменной.
DW 19	Wk	вх		KF	Здесь вход точки установки. Если STEU бит 6 = 0. ($-1<W_k<1$)
DW 20	MERK				Зарезервирован
DW 21	Xk	вх	LP	KF	Ввод текущего значения для STEU бит 6 = 0 ($-1<X_k<1$)
DW 22	Xk	вх	GP	KG	Ввод текущего значения для STEU бит 7 = 0 ($-1<X_k<1$)
DW 24	Zk	вх	LP	KF	Воздействующая величина ($-1 < Z_k < 1$)
DW 25	Zk	вх	GP	KG	Воздействующая величина вводится здесь, если STEU бит 7 = 1 ($-1 < Z_k <1$)
		вх	GP	KG	
DW 27	Zk-1				Предыдущее значение воздействующей величины
DW 29	XZk-1	вх	LP	KF	Значение, прикладываемое к блоку источника через вход XZ ($-1 < XZ_k <1$): вход здесь, если STEU бит 7 = 0
DW 30	XZk-1		GP	KG	XZ вводится здесь, если STEU бит 7 = 1 ($-1<XZ_k<1$)
DW 32	XZk-1	вх	GP	KG	Предыдущее ограничение XZ
DW 34	PZk-1		GP	KG	XZ_{k-1} $-XZ_{k-2}$
DW 36	DDk-1		GP	KG	Исходная компонента
DW 38	XWk-1		GP	KG	Предыдущее значение источника управления
DW 40	PWk-1			KG	XW_{k-1} XW_{k-2}

			GP		

Адрес в DB	Имя	Вх/вых	Формат числа	Формат PG	Примечание
DW 42	-	-	-	-	Зарезервировано
DD44	Yk-1		GP	KG	Предыдущее значение вычисленной управляемой величины Y_{k-1} или dY_{k-1} до ограничения
DD46	YAk	Вых	GP	KG	Выход величины
DW48	YAk	вых	LP	KF	Выход величины BGUG < YA , BGOG

Рекомендуемый формат
(KH и KM тоже допустимы)

GP = формат числа с плавающей запятой
LP = двоичная или десятичная дробь

Вх – вход, вых – выход

Значение слова управления STEU (слово 11 в передаваемом DB)

DW11 бит	Имя	Значение
11.0	AUTO	= 1 : Автоматический режим работы = 0 : Ручной режим работы
11.1	XZ_ON	= 1 : Дифференциальная величина, которая не должна быть XWk, подается на блок источника через вход XZ = 0 : XW_k подается на блок источника XZ игнорируется
11.2	REG_AUS	= 1 : При вызове регулятора (OB251) все значения в блоке данных регулятора (DW20…DW48) сразу стираются (кроме K, R, TI, TD, BGUG, BGOG< STEU, Yh_k и W_k). Регулятор выключается. Предыдущее значение входной величины обновляется = 0 : Регулятор включается
11.3	VELOC	= 1 : Алгоритм управления скоростью = 0 : Алгоритм управления положением
11.4 [1)]	MAN	= 1 : Для GESCHW = 0 (алгоритм управления положением) последняя управляемая переменная остается. Для GESCHW = 1 (алгоритм управления скоростью) устанавливается приращение позиционирования $dY_k = 0$. = 0 : Для GESCHW = 0, после переключения в ручной режим, значение управляемой переменной YA экспоненциально изменяется к значению, установленному вручную за 4 шага. После этого значение, устанавливаемое вручную, передается непосредственно на выход. Для GESCHW = 1, значение установленное вручную, передается непосредственно на выход регулятора. При ручном режиме работы действуют ограничители и выполняются следующие соотношения: 1) X_k, X_{k-1}, PW_{k-1} 2) XZ_k, XZ_{k-1}, PZ_{k-1}, если STEU бит 1 = 1 3) Z_k, Z_{k-1}, если STEU бит 5 = 0 Значение aD_{k-1} устанавливается в «0». Алгоритм не выполняется
11.5	NO_Z	= 1 : Без прямого управления = 0 : С прямым управлением
11.6	PGDG	= 1 : Wk -, Yhk- вводятся в виде чисел с плавающей запятой = 0 : Ввод в виде двоичной/десятичной дроби
11.7	VAR_GP	= 1 : Переменные Xk, XZk, Zk вводятся в виде дроби чисел с плавающей запятой = 0 : Ввод переменной в виде двоичной/десятичной дроби
11.8	STOS	= 1 : Нет безударного переключения ручной/автомат = 0 : Безударное переключение ручной/автоматический
11.9 . .	. .11.15	Не имеют значение

[1)] Имеет значение только при ручном режиме (AUTO = 0)

6.11.1. Инициализация PID-алгоритма (OB 250)

OB 250 инициализирует PID-алгоритм и вызывается в ходе выполнения организованных блоков запуска OB 20/21/22.

Требуемые для инициализации параметры заносятся в соответствующий блок данных DB x.

ВАЖНО!
DB x должен быть вызван перед вызовом OB 250.

Для каждого регулятора должен использоваться один единственный DB x. (x < 254). Системная программа автоматически создает путем копирования DB x в область DB-RAM другой блок данных DBx + 1, который регулятор использует в качестве поля данных во время выполнения циклической программы: поэтому соответствующий блок данных должен быть свободен. Блок данных DBx + 1 является интерфейсом данных между регулятором и пользователем или периферией.

OB 250 использует внутри себя OB 254 или 255 (дублирование блоков данных). В случае ошибки процессор идентифицирует ошибку выполнения и вызывает OB 31. Если он не запрограммирован, то процессор переходит в “СТОП”. После возвращения в OB 250 в АКК 1 содержится признак ошибки.

Внимание! Если DBx + 1 нельзя использовать во время инициализации, он будет использован без какой-либо индикации от операционной системы в качестве поля данных регулятора: подразумевается, что его длина такая же. Как и DB регулятора (48 слов данных), и слова DW20…DW48 стирается. В противном случае контроллер переходит в состояние «СТОП».

6.11.2. Обработка данных по PID-алгоритму (OB 251)

OB 251 вызывается во время выполнения циклической программы и обрабатывает данные по PID-алгоритму.

Через некоторое время нужно запустить регулятор.

При этом необходимо выполнить следующее:

- o вызвать DB x + 1
- o загрузить в него входные данные Xk, XZk, Zk, YHk или их комбинации
- o преобразовать входные данные в нужный формат и передать их в DBx+1
- o вызвать OB 251 (PID-регулятор)
- o загрузить выходные данные YAk DB x + 1
- o обработать данные и передать их входы/выходы

Формат входных и выходных данных регулятора

Алгоритм PID–регулятора использует формат чисел с плавающей запятой для внутреннего представления чисел и может параметрироваться значениями с плавающей запятой. Кроме того, могут использоваться двоичные и десятичные дроби (смотри бит 6 и 7 в слове управления STEU). В этом случае регулятор автоматически преобразует слова в формат с плавающей запятой.

В языке STEP-5 требуется меньше времени на преобразование слов входов и выходов, если они вводятся в двоичном или десятичном виде.

Входы

W, YH, X, Z и XZ могут быть введены как в виде чисел с плавающей запятой, так и в двоичном/десятичном виде. Блок передачи данных для каждого значения имеет различные позиции в памяти.

Ввод в виде двоично-десятичной дроби

(Более детальную информацию по этим числам смотри в Разделе 6.11)

Внимание! Для того, чтобы быть в номинальных пределах модуля аналоговых входов, важно помнить, что значение отдельных битов будет различно в зависимости от того, какие пределы входного сигнала используются. Это следует обязательно учитывать, особенно когда устанавливается прямое управление. В противном случае может возникнуть ситуация, когда прямое управление, введенное с помощью программатора, не может быть выполнено, хотя фактическое значение намного больше, чем требуемое значение.

Если аналого-цифровой преобразователь используется для отрицательных чисел в виде значения со знаком, то из этого значения сначала должно формироваться его двоичное дополнение, перед тем как оно передается в блок данных регулятора. Двоичное значение 15 бита должно быть установлено в «1».

Если при использовании аналого-цифрового преобразователя получается число «0», то оно представляется в виде:

1000000000000000

Затем формируется двоичное дополнение. Вместо этого числа в блок данных записывается +0:

0000000000000000

Выходы:

Выход регулятора YA формируется в DB в виде числа с фиксированной запятой, расположенной слева, формат двоичной/ десятичной дроби входа и выхода в зависимости от используемого модуля входов\выходов (аналого-цифровой преобразователь, цифро - аналоговый преобразователь) должен быть преобразован до и после вызова регулятора в программе пользователя STEP5, перед тем как значения передаются в или из блока данных регулятора.

Общие замечания

Если значение STOS (бит 8 STEU) = 0, происходит переключение из ручного в автоматический режим безусловно, т.е. подбираемый любой величины дифференциал регулятора управляется только через составляющую I. Однако, если был выбран TI = TA/TN = 0 (P- или PD-регулятор), дифференциал регулятора при переключении не вызывает изменение величины управляемой переменной.

Это можно исключить, установив STOS =1. Дифференциал регулятора при этом при переключении ручной\автоматической быстро перерегулируется, независимо от того TI = 0 или нет. Возникающие при этом изменения величины управляемой переменной соответствует величине дифференциала регулятора и, таким образом, не является произвольной в смысле ошибке работой регулятора.

Биты 0 и 1 MERK могут, если надо, использоваться для индикации, для того. чтобы показывать, что величина управляемой переменной (при алгоритме скорости приращения положения) на верхней или нижней границе. Так как эти биты обрабатываются алгоритмом для отключения I – составляющей то она не может быть переписана.

Блок данных регулятора DBx+1 не может быть перезагружен во время циклической работы.

Если каскадируется управление с двумя или более регуляторами. следует учитывать следующее:

- Если каскад должен быть разделен на части, то или все регуляторы должны быть переведены в ручное управление одновременно, так, чтобы не один из регуляторов не изменял компоненту I, или, по крайней мере, регулятор внешней петли должен быть в ручном режиме работы, так, чтобы последняя управляемая переменная, которая относится к точке установки внутренней петли, была установлена или могла бы быть установлена в безопасное значение.

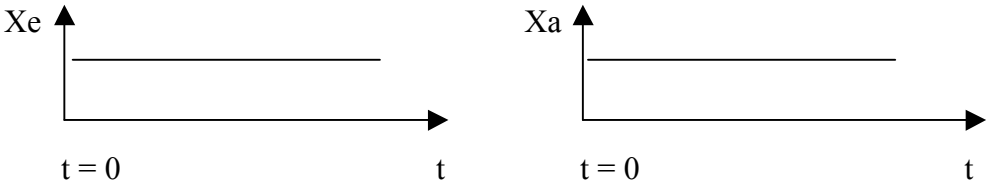
- Если каскад должен быть замкнут то обе петли должны находиться в автоматическом режиме управления или, по крайней мере, внутренняя петля, так, чтобы управляемая переменная внешней петли могла бы быть принята за точку установки.

Если во время переключения в ручной режим работы управляемая система отделяется от регулятора и регулируется непосредственно конечный элемент управления, то результирующая управляемая переменная должна быть подключена к ручному входу регулятора. Этим обеспечивается выдача набора управляемых переменных, соответствующих работе в ручном режиме при переключении из ручного режима работы в автоматический. При использовании алгоритма управления скоростью – это управляемая переменная отклонения.

характеристики регулятора

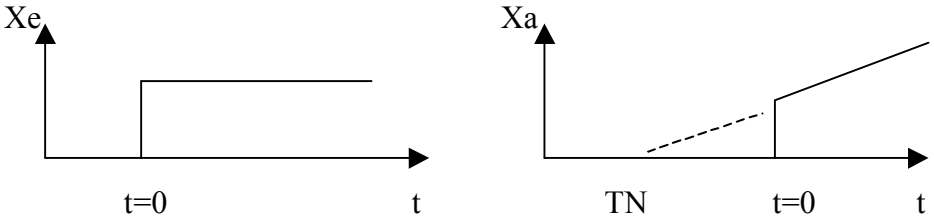
P-регулятор

Количественная характеристика для P-регулятора – это K. это отношение входной и выходной величин: K = Xa/Xв



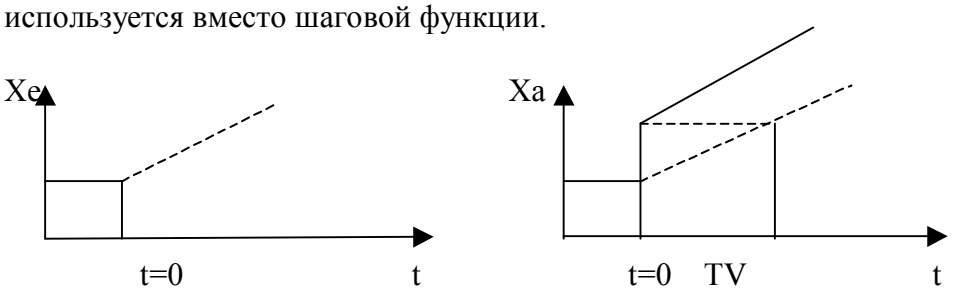
PI – регулятор

Коэффициент пропорциональности K и интегральная константа времени TN являются количественными характеристиками для PI – регулятора. Коэффициент пропорциональности K – отношение входной и выходной величин и определяет реакцию P.
Время сброса TN – это время, требуемое для реакции и достижения того же значения регулируемой величины с помощью воздействия I, как и для компоненты P.



PD – регулятор

Коэффициент пропорциональности K и интегральная (см. выше) и константа времени отклонения TV являются количественными характеристиками PD –регулятора. Время прямого управления – это время, которое необходимо PD – регулятору, чтобы произошло такое же изменение на выходе, какое PD – регулятор выполняет в результате наличия компоненты D. Для определения времени подачи линейного изменения входное значение используется вместо шаговой функции.



PID – регулятор

Количественные характеристики PID- регулятора – это коэффициент пропорциональности K, время сброса TN и время подачи TV. Они определяют P, I и D реакцию.

Изменение параметров

P – составляющая управляемой переменной вычисляется по следующей формуле:

P- составляющая = KP ($XW_k - XW_{k-1}$)

Если KP или R изменяются в автоматическом режиме, то это повлияет только на последующие изменения дифференциальной составляющей регулятора XWk. Мгновенное значение управляемой переменной при изменении параметров не меняется. Это позволяет осуществлять безусловное изменение параметров.
Однако, если это не желательно, то этого можно избежать путем следующих вычислений, которые производятся один раз при каждом изменении параметров(пример для изменения KP):

$Y_{k-1} = Y_{k-1} + XW_{k-1}(KP_{nau} - KP_{alt})$

Если при изменении параметров используется следующая программа, то регулятор ведет себя как аналоговый:

```
: L     KP нов  KPнов
: L     Kpстар  KPстар
: -G
: L     DD38            XW k-1
: xG
: L     DD44            Y k-1
: +G
: T     DD44            = Y k-1
```

Сокращения для PID- регулятора

DY_k	Вычисляемое приращение позиционирования
DZ_k	Приращение воздействия
GP	Представление в виде числа с плавающей запятой
K	к-тый опрос
K	Коэффициент пропорциональности
LP	Представление в виде числа с запятой слева (0.123)
UL	Верхний предел
R	R-параметр
TA	Время опроса
TD	TV/TA
TI	TA/TN
T	Точка опроса (время) = k TA
TN	Время сброса
TV	Время подачи вперед
LL	Нижний предел
W_k	Точка установки
X_k	актуальное значение
XW_k	Отклонение управляющего воздействия
Y_k	Вычисленная управляемая переменная
YA_k	Значение управляемой переменной (приращение положения или управляемой переменной)
Z_k	Воздействующая величина

Двоичные и десятичные дроби

Для представления двоичных и десятичных дробей в блоке данных требуется одно слово. Связь между десятичным числом, двоичным числом и представлением в виде числа в формате с фиксированной запятой в программаторе показана на следующем примере:

Дробь		Число с фиксированной запятой
десятичная	двоичная	
-0,999. . .	100000000000000001	-32767
-0,75	101000000000000000	-24576
-0,5	110000000000000000	-16384
-0,25	111000000000000000	-8192
0	000000000000000000	0
+0,25	001000000000000000	+8192
+0,5	010000000000000000	+16384
+0,75	011000000000000000	+24576
+0,999. . .	011111111111111111	+32767

В двоичном представлении отрицательные числа отличаются от положительных тем, что они образованы двоичным дополнением.

Двоичные/десятичные дроби (LP) могут быть преобразованы в значения, отображаемые программатором в формате KF, по формуле:
LP x 32767 = KF

Если –1 < LP < 1 и –32767 < KF < +32767

7 Предустановки системных функций в блоке данных DX 0

Как пользователь, вы можете изменить некоторые функции операционной системы контроллера в соответствии со своими требованиями с помощью записи в блок данных DX0 измененных предварительных установок (в следующей таблице стандартные предустановки помечены символом “V”).

При каждом новом пуске автоматически устанавливаются стандартные предустановки операционной системы (V). После этого начинает обрабатываться DX 0. Если DX 0 не запрограммирован, то действуют стандартные предустановки, в противном случае действуют предустановки пользователя.

Важно!
Ввод или изменение DX 0 становится действительным только после нового пуска контроллера.

Структура DX 0

DX 0 состоит из трех следующих частей:
1. метка начала блока DW0 (DW0 1 и 2)
2. множество блоков различной длины (в зависимости от числа параметров)
3. метка окончания EEEE.

Числовые значения вводятся в 16-ричном формате.

Максимальная структура:

№ бита	15	87	0	
DW 0	4	D	4	1	Символ MA
1	5	3	4	B	ASCIE SK
2	5	8	3	0	X0
3	Метка блока 1		Длина блока 1		
.	параметр				
.	параметр				Блок 1
.	параметр				
.	Метка блока 2		Длина блока 2		
.	параметр				Блок 2
.	Метка блока n		Длина блока n		
.	параметр				
.	параметр				Блок n
.	параметр				
m	E	E	E	E	Метка окончена

Рис. 7-1: <u>Структура блока DX 0</u>

Пример ввода DX 0

Метка начала	DW0: KH = 4D41
	DW1: KH = 534D
	DW2: KH = 5830
Метка блока/длина	DW3: KH = 0203
Параметр (заполняется 1DW)	DW4: KH = 3001 Блок 1
Параметр (заполняется 2DW)	DW5: KH = DD00
	DW6: KH = 0000
Метка блока/длина	DW7: KH = 0402
Параметр (заполняется 2DW)	DW8: KH = 1000 Блок 2
	DW9: KF = 4000
Метка окончания	DW10: KH = EEEE

Блок в DX0 состоит из от 1 до N слов данных

Он содержит
- метку блока
- длину блока
- параметр блока

Метка блока показывает, какое значение имеют нижеследующие параметры. Каждый блок соответствует определенной части операционной системы или операционной системной функции (например, метка блока 04 соответствует циклической обработке программы)

Длина блока показывает, сколько слов данных занимают последующие параметры.

Возможные параметры описываются на следующих страницах.

При параметрировании DX0 обратите внимание на следующие особенности:

Последовательность ввода блоков произвольная

- Ненужные блоки вводить не следует
- Если определенный блок повторяется многократно, действительным является блок, введенный последним
- Последовательность ввода параметров произвольная
- Ненужные параметры вводить не следует
- Если определенный параметр повторяется многократно, действительным является параметр, введенный последним

Важно!
После ввода последнего блока DX0 должен быть обязательно завершен меткой окончания EEEE!

Возможные вводы в DX 0

Метка/длина блока	Параметр	Значение [1])
Режимы ЗАПУСК И обработка (RUN)		
02xx [2])	1000	V Автоматический перезапуск после включения напряжения
	1001	Автоматический новый пуск после включения напряжения
	2000	V Синхронизация запуска в многопроцессорном режиме работы
	2001	Без синхронизации запуска в многопроцессорном режиме работы
	3000	V Контроль ошибки адресации
	3001	Без контроля ошибки адресации
	6000	V Арифметика с плавающей запятой с 16-битовой мантиссой (оптимизация по скорости)
	6001	Арифметика с плавающей запятой с 24-битовой мантиссой (оптимизация по точности)
	BB00 yyyy	Количество активных таймеров [3])
		Предустановка: yyyy = 256 таймеров допустимо: от 0 до 255

1) V = предустановка при новом пуске

2) xx = длина блока (количество слов данных, занятых под параметры)

3) для активизации таймеров:

- Стандартно обрабатываются таймеры с T0 по T255
- Если вы введете в DX0 число «0», то обработки таймеров не будет
- При числе от «1» до «128» обрабатываются таймеры от T0 до T127. Если введенное число лежит между «129» и «256», то обрабатываются все таймеры от T0 до T255.

Метка/длина блока	Параметр	Значение [1])
Обработка циклической программы:		
04xx[2])	1000yyyy	Длина времени цикла в миллисекундах: Предустановка: yyyy = 150мс Допустимо: 1< yyyy < 1770 (16-ричный) 1 мс . . . 6000мс (десятич)
	4000	V Обновление области отображения маркеров межпроцессорной связи без использования семафоров
	4001	Обновление области отображения маркеров межпроцессорной связи с защитой с помощью семафоров (в блоке)
Обработка программы прерывания		
04xx[2])	1006	Обработка прерываний по времени, от регулятора и от процесса на границе команд STEP5 [3])
	1008	Обработка прерываний от регулятора и от процесса на границе команд STEP5, прерывание по времени на границе блока
	100A	Обработка прерываний от процесса на границе команд STEP5, прерывание от регулятора и по времени на границе блока
	100C	V Обработка прерываний по времени, от регулятора и от процесса на границе блока

Приведенные выше параметры действительны для версии CPU928, которая знает только 100мс прерывание по времени. В новой версии CPU928 можно программировать до 9 различных OB прерывания по времени. Для этого в DX 0 имеются новые метки (см. следующую страницу)

1) V = стандартная предустановка при отсутствии или некорректном DX 0

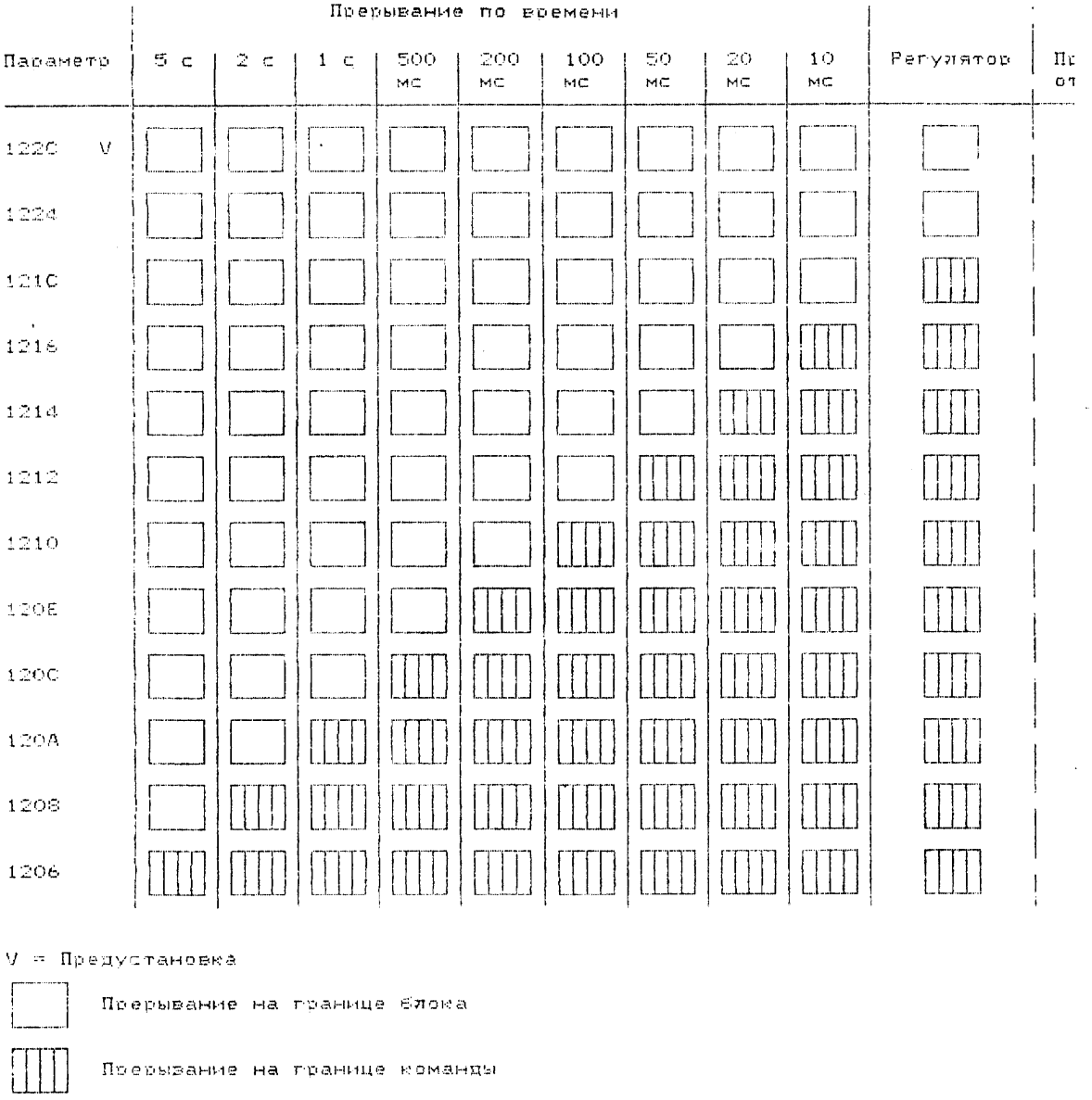
2) Xx = длина блока (количество слов данных, занятых под параметры)

3) Указание:
Если вы в DX 0 предусмотрели обработку прерывания на границе команды, то следует помнить, что прерывание команд TBN и TNW будет не полностью обработано, так как они содержат псевдограницы команд. Это действительно и для некоторых небольших организационных блоков специальных функций, стандартных функциональных блоков и функциональных блоков регуляторов.

Прерывание от процесса = обработка программы прерывания по сигналу, поступающему от процесса.

Прерывание по времени = обработка программы прерывания, управляемой по времени.

Путем программирования новых параметров в DX 0 создаются следующие возможности установок для обработки прерываний:



Метка длины блока	параметр	Значение [1])
06xx[2])	2000	V Сигнал прерывания от процесса, устанавливаемый по фронту
	2001	Сигнал прерывания от процесса, устанавливаемый по фронту

Обработка ошибок

		Обработка ошибок по времени
10xx[2])	1000	V Останов системы, если появилось условие и OB33 не загружен
	1001	Нет останова системы, если появилось условие и OB33 не загружен
		Обработка ошибок регулятора
	1200	V Останов системы, если появилось условие и OB34 не загружен
	1201	Нет останова системы, если появилось условие и OB34 не загружен
		Обработка ошибок цикла
	1400	V Останов системы, если появилось условие и OB26 не загружен
	1401	Нет останова системы, если появилось условие и OB26 не загружен
		Обработка ошибок кодов и команд
	1800	V Останов системы, если появилось условие и OB27/OB29/OB30 не загружены
	1801	Нет останова системы, если появилось условие и OB27/OB29/OB30 не загружены

[1]) V = стандартная предустановка при отсутствии или некорректном DX 0
[2]) xx = длина блока (количество слов данных, занятых под параметры)

Ошибка кодов команд = ошибка замещения, ошибка кодов операции, ошибка параметров

Ошибка выполнения = вызов незагруженного блока, ошибка передачи или особые ошибки выполнения.

Метка /длина блока	Параметр	Значение [1])
10xx [2])		Обработка ошибки выполнения
	1A00	V Останов системы, если появилось условие и OB19/OB31/OB32 не загружены
	1A01	Нет останова системы, если появилось условие и OB19/OB31/OB32 не загружены
		Обработка ошибки адресации
	1000	V Останов системы, если появилось условие и OB25 не загружен
	1001	Нет останова системы, если появилось условие и OB25 не загружен
		Обработка ошибки задержки квитирования
	1800	V Останов системы, если появилось условие и OB23/OB24 не загружены
	1801	Нет останова системы, если появилось условие и OB19/OB31/OB32 не загружены
EEEE		Метка окончания

[1]) V = стандартная предустановка при отсутствии или некорректном DX 0

[2]) xx = длина блока (количество слов данных, занятых под параметры)

Пример A для параметрирования DX 0:

Вы хотите в многопроцессорном режиме установить три процессора:
процессоры A, B, C. Процессоры A и B работают совместно, часто обмениваются данными и обрабатывают большие программы запуска. Процессор C обрабатывает независимо от них короткую критичную по времени программу.

Обычно в многопроцессорном режиме все процессоры одновременно начинают обработку циклической программы, т. е. все процессоры ждут друг друга до тех пор, пока все не выполнят программу запуска и после этого все вместе переходят к выполнению циклической программы запуска.

Процессор C выполняет свою программу независимо от других процессоров и обрабатывает очень короткую программу запуска, при которой не требуется синхронизации запуска. При параметрировании DX0 Вы можете потребовать, чтобы процессор C сразу после выполнения программы запуска переходил к выполнению циклической программы без ожидания процессоров A и B.

Программирование DX 0:

DX0	Метка начала	DW0: KH=4D41
		DW1: KH=534B
		DW2: KH=5830
	Метка первого блока – длина	DW3: KH=0201
	Параметр 1	DW4: KH=2001
	Метка окончания	DW5: KH=EEEE

Если Вы загрузите DX0 в память программы, то он станет действительным после выполнения следующего нового пуска. Процессор C выполняет очень короткую программу запуска и не ожидая процессоры A и B немедленно зажигают зеленый светодиод RUN. Однако сигнал BASP (блокировка выходных команд) снимается только после выполнения всеми процессорами программ запуска. Это означает, что процессор C не может обращаться к цифровой периферии.

Пример B для параметрирования DX 0:

Параметрируя DX0 следующим образом, можно:

а) отключить контроль ошибки адресации
б) отключить актуализацию таймеров
в) установить время цикла на 4 сек.

DX0	Метка начала	DW0: KH= 4D41
		DW1: KH= 534B
		DW2: KH= 5830
	Метка первого блока\длина	DW3: KH= 0203
	параметр	DW4: KH= 3001
	параметр *)	DW5: KH= BB00
		DW6: KH= 0000
	Метка второго блока\длина	DW7: KH= 0402
	параметр *)	DW8: KH= 1000
		DW9: KH= 4000
	Метка окончания	DW5: KH= EEEE

*) Параметр, который занимает два слова данных при вводе длины блока, должен определяться цифрой 2.

Это параметрирование DX0 имеет следующее влияние на выполнение программы:

- Некоторая часть области отображения процессора, которая не соответствует модулю периферии, может быть использована в качестве дополнительной области памяти маркеров.
- Выполнение системной программы сокращается, так как не нужно актуализировать таймеры.
- Ошибка цикла опознается в том случае, если выполнение циклической программы пользователя и системной программы вместе превысят 4 сек.

8. Заполнение и организация памяти

Общая область памяти в CPU928 делится на следующие основные области:

		Ширина
1.	Память пользователя для OB, FB, FX, PB, SB, DB, DX	(16 бит)
2.	DB-RAM для блоков данных и регистров сдвига	(16 бит)
3.	-Область интерфейса: BA, BB -системная область: BS, BT -счетчики: Z -таймеры: T	(16 бит) (16 бит) (16 бит) (16 бит)
4.	–Маркеры: M -отображение процесса на входах и выходах PAE, PAA	(8 бит) (8 бит)
	5. – область периферии: -P-периферия -Q-периферия -маркеры межпроцессорной связи -координатор KOR -буфер связи -децентрализованная периферия	(8 бит)

Точные адреса этих областей Вы можете найти в схеме распределения памяти на следующей странице.

Важно!
Обращение STEP5 к ячейкам памяти внутри области операндов, например, маркерам, должно производиться не непосредственно по абсолютным адресам этих ячеек памяти, а относительно базового адреса соответствующей области операндов. Базовые адреса всех областей операндов занесены в области системных данных (BS-область) (см. «распределение системных данных»).

8.1. Распределение адресного пространства в CPU928

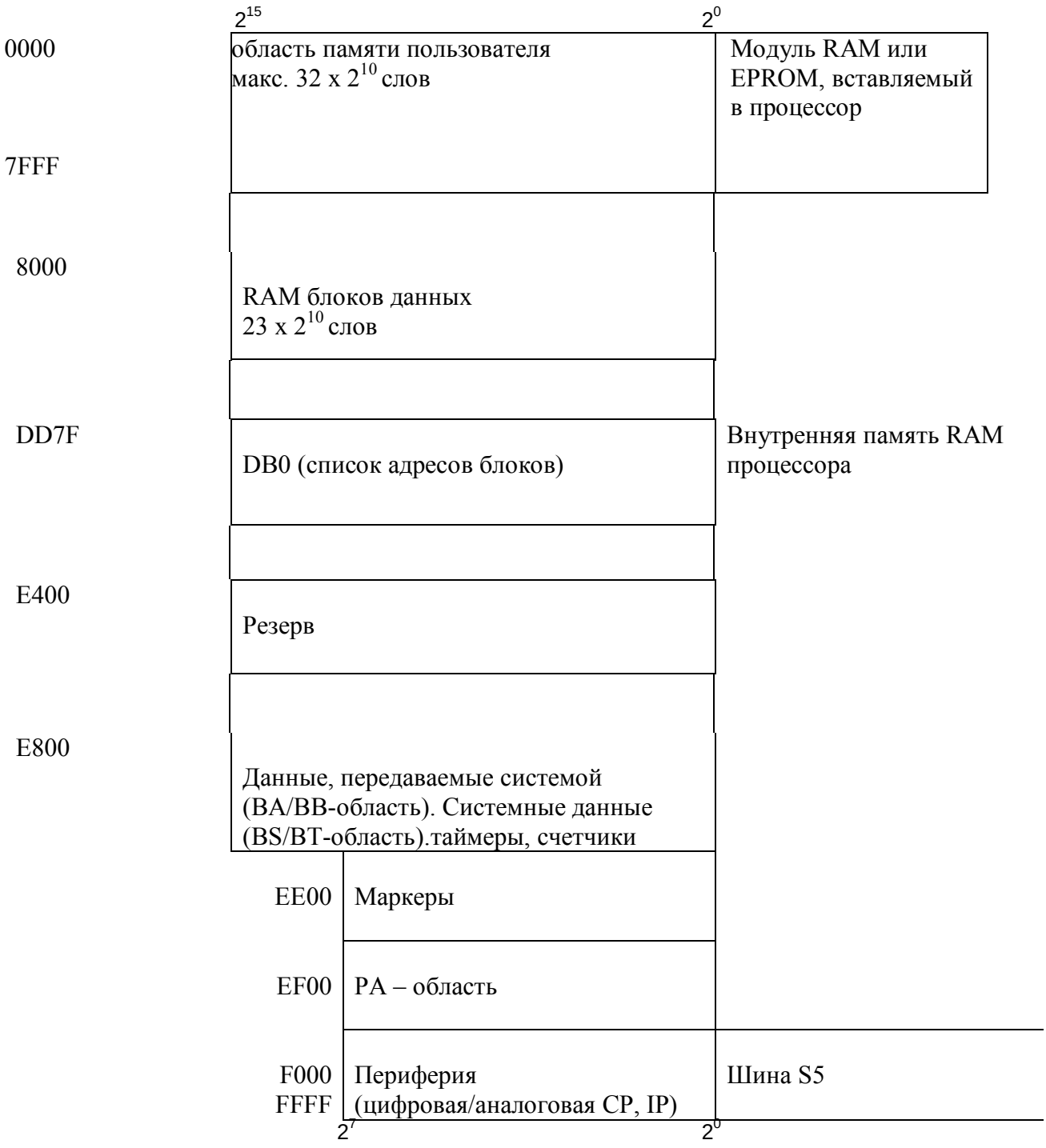


Рис. 8.1: Распределение адресного пространства FFFF в CPU928

8.1.1. Распределение адресного пространства системной области RAM.



Рис. 8-2; Распределение адресного пространства системой памяти RAM (16 бит)

8.1.2. Распределение адресного пространства периферии.

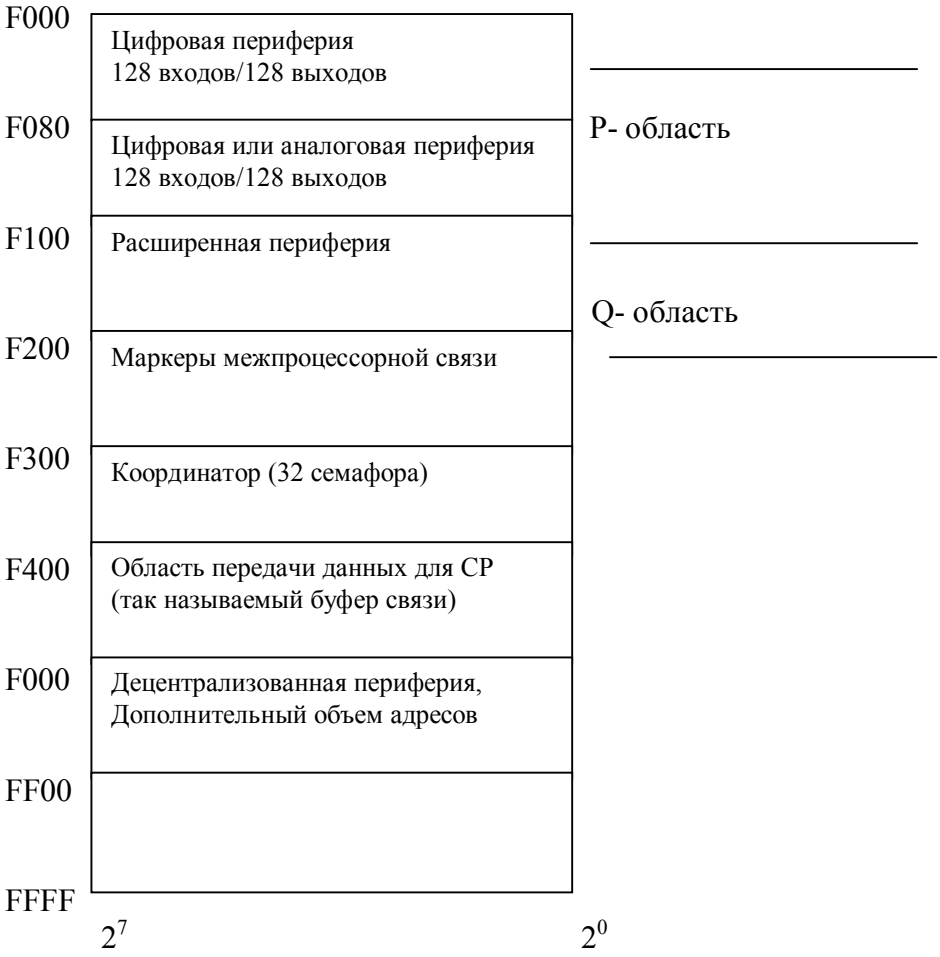


Рис. 8-3: Распределение адресного пространства периферии (8 бит).

Область адресов для периферии

Область (абсолютный адрес)		Адресуется с помощью	Параметр
EF00 EF7F	PAE Отображение входов	L EB/T EB L EW/T EW L ED/T ED L E/UN E/O E/ON E S E/R E/=E	0. . .127 0. . . 126 0. . . 124 0.0. . . 127.7
EF80 EFFF	PAA отображение выходов	L AB/T AB L AW/T AW L AD/T AD L A/UN A/O A/ON A	0. . . 127 0. . . 126 0. . . 124 0.0. . .127.7
F000 F07F	Цифровые входы/выходы	L PY/T PY L PW/T PW	0. . . 126 0. . . 127
P- периферия с отображением процесса			
F080 F0FF	Цифровые или аналоговые входы/выходы	T PY/T PY T PW/T PW	128. . .255 128. . . 254
P-периферия без отображения процесса			
F100 F1FF	Область расширенной периферии	L QB/T QB L QW/T QW	0. . . 255 0. . . 254
Q-периферия			

Рис. 8-4: Распределение адресного пространства для периферии/программирования.

Доступ к периферии

С помощью команд языка STEP5 мы можем либо непосредственно, либо через область отображения процесса обращаться к периферии. При этом следует помнить, что область отображения процесса существует только для байтов входов и выходов P- периферии с адресами байтов от 0 до 127!

Прямой доступ к периферии: L/T PY, L/T PW, L/T QB, L/T QW/

Входы и выходы считываются или устанавливаются в момент обработки программы. Для цифровой периферии (от 0 до 127)переустанавливаются выходы в области отображения процесса.

Обращение к периферии через область отображения процесса:

L/T EB, L/T EW, L/T QB, L/T QW
L/T AB, L/T AW, L/T AD
U/UN/O/ON/ E = A и т.д.

В момент обработки команды изменяется только область отображения процесса. Сразу после окончания цикла на периферию переносится новое состояние области отображения процесса.

8.2. Организация памяти в CPU 928

Память пользователя занимает область памяти от 0 до 7FFFH. При загрузке отдельных блоков программы пользователя они располагаются в памяти в любой последовательности (по возрастанию адресов). Если память пользователя заполнена, то блоки данных располагаются во внутренней области памяти DB-RAM (от 8000H до DD7FH).
С помощью on-line-функции программатора “SPAUS” (структура памяти) вы можете получить в 16-ричном виде адрес ячейки памяти, которая содержит команду окончания последнего имеющегося в памяти блока.
При корректировке блоков старые блоки объявляются в памяти недействительными (т.е. метка начала переписывается), и в память заносится новый блок. Таким образом, при стирании блоков в действительности блоки не стираются, а только объявляются недействительными. Стирание и корректировка блоков занимает, таким образом, постепенно все место в памяти.
С помощью on-line-функции программатора «Сжатие памяти» можно удалить все недействительные блоки из памяти и сместить все действительные блоки один к другому.

8.2.1. Заголовок блока памяти пользователя и DB-RAM

Каждый блок в памяти начинается с заголовка блока длиной 5 слов.
Слово: Метка начала блока: 7070H
Слово: Старший байт = тип блока



Младший байт = номер блока
Номер блока (от 0 до 255) располагается в младшем байте второго слова заголовка и представляет собой числа в двоичном коде от 00 до FFH.

3. Слово: В старшем байте третьего слова находится метка программатора, в младшем байте – часть библиотечного номера.
4. Слово: Четвертое слово содержит остаток библиотечного номера
5. Слово: В пятом слове в старшем и младшем байте хранится информация о длине блока (включая заголовок). Длина указывается в словах.

8.2.2. Список адресов блоков в блоке данных DB0

Блок данных DB0 содержит список начальных адресов всех блоков, которые находятся в памяти пользователя или в DB-RAM процессора. Этот список адресов создается операционной системой после подачи напряжения питания и автоматически обновляются при вводе новых или изменении уже имеющихся блоков с помощью программатора.
Для каждого типа блоков BDB0 зарезервировано по 256 слов адресного списка. Незагруженные блоки имеют начальный адрес «0».
Кроме того, начальные адреса отдельных списков адресов блоков хранятся в словах системных данных с BS32 по BS38.

BS32: Начальный адрес списка адресов блоков DX
BS33: Начальный адрес списка адресов блоков FX
BS34: Начальный адрес списка адресов блоков DB
BS35: Начальный адрес списка адресов блоков SB
BS36: Начальный адрес списка адресов блоков PB
BS37: Начальный адрес списка адресов блоков FB
BS38: Начальный адрес списка адресов блоков OB (только 48 слов)

Начальные адреса указывают всегда на первое слово данных после заголовка блока:
для блоков данных это слово данных DW 0!
Для кодовых блоков это первая инструкция STEP5! (для FB это команда ‘SPA’)

Расположение адресов блоков в DB 0:

n = начальный адрес списка адресов в PS (= содержимое слова BS36)

	DB0	
n	Адрес PB0	
n+1	Адрес PB1	
n+2	Адрес PB2	
		Если в качестве адреса указано значение «0», то блок не загружен.
n+178	Адрес PB178	
n+179	Адрес PB179	

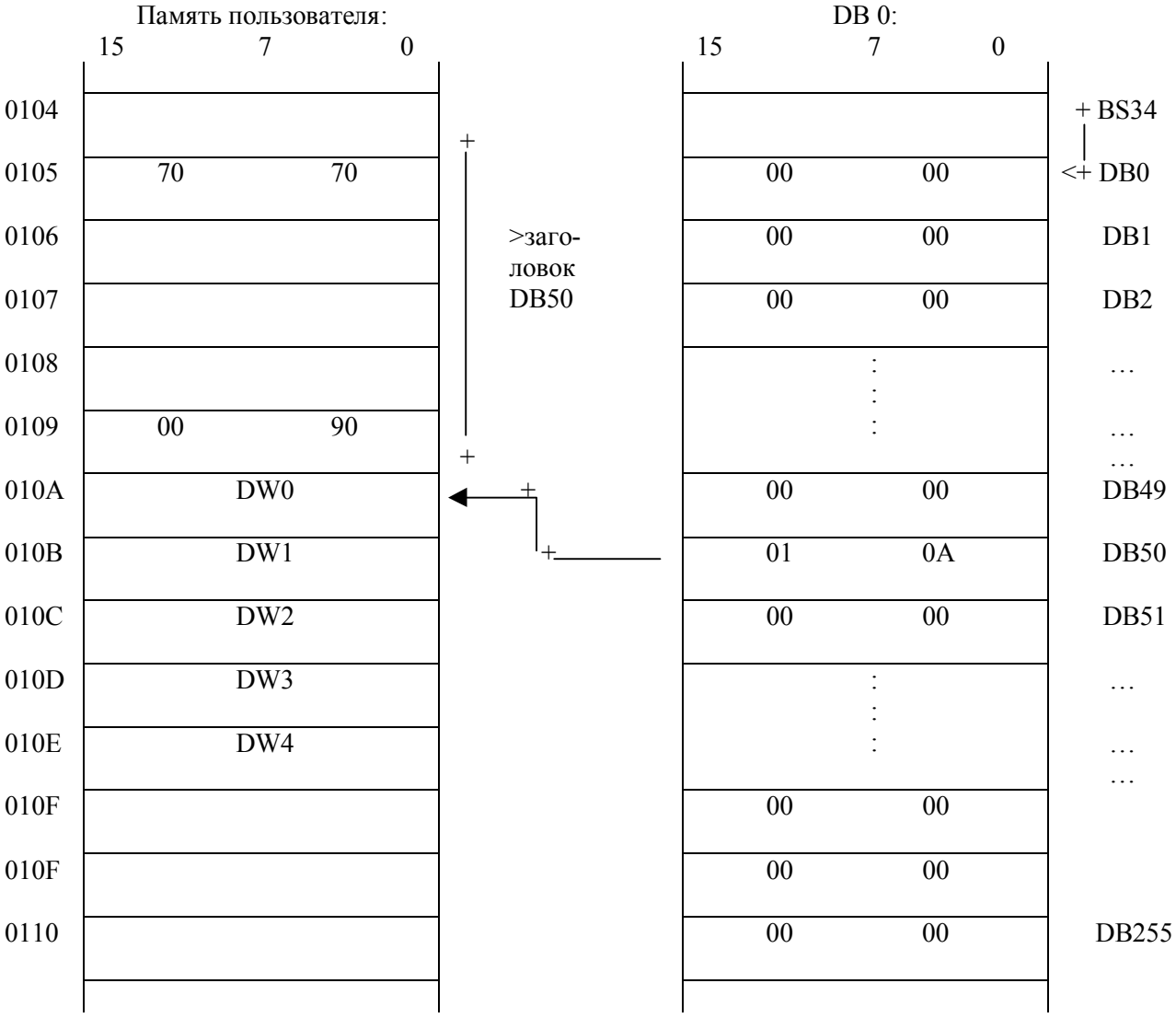
Таким образом, вы можете определить адрес любого блока в памяти:

Пример: Начальный адрес блока FB 40

```
: L     BS 37       Базовый адрес списка адресов FB
: L     KB 40       + номер FB
: +F                = адрес ячейки памяти, содержащей
:                   Начальный адрес FB40
:
: LIR 1             Начальный адрес FB40 загружается в
:                   аккумулятор АКК 1 (если блока нет, то
:                   начальный адрес равен 0)
:
```

Пример: Начальный адрес и длина блока DB 50

а) через косвенное обращение к памяти




```
: L BS 34       Загрузка базового адреса списка адресов DB
: L KB 50       Вычисление входного адреса DB50
: +F            и загрузка в аккумулятор АКК 1
: LIR 1
: L KB 0        При отсутствии блока – переход на метку
: ! = F         NIVO
: SPB = NIVO
: ENT           Начальный адрес DB50 загружен в АКК 3 и АКК 1
: TAK
: L KF –1       Начальный адрес уменьшается на 1 и
: +F            длина блока загружается в АКК 1
: LIR 1
…
…
NIVO: …….
```

Результат: АКК 1-L: длина DB50
АКК 2-L: начальный адрес DB50

б) с помощью специальной функции OB181 «Тестирование блока данных (DB\DX)” .

OB181 содержит ту же самую функцию, которая была описана в пункте а). Однако здесь дополнительно проверяется, находится ли блок данных в памяти пользователя (RAM или EPROM) или в DB-RAM.

```
                : L  KY 1.50          Блок данных DB50
                : SPA OB181           “Тестирование блока данных DB/DX”
                : SPB =NIVO
                : SPM =PROM
                : SPZ =ANWE
                : SPP =DBRA

        NIVO    :                     Блок не найден
                :
                BEA

        PROM    :                     Блок находится в памяти пользователя
                :                     ( модуль EPROM)
                : BEA

        ANWE    :                     Блок находится в памяти пользователя
                :                     (Модуль RAM)
                : BEA

        DBRA    :                     Блок находится в DB-RAM
                :
                : BE
```

Результат АКК 1-L : Начальный адрес DB50
АКК 2-L : Длина DB50
VKE = 1 : Если DB50 отсутствует

8.2.3. Область BA\BB

Область BA представляет собой область памяти длиной 256 слов во внутренней системной памяти RAM процессора. Она занимает адреса E800H до E8FFH.

Область BB представляет собой область памяти длиной 256 слов во внутренней системной памяти RAM процессора. Она занимает адреса с E900H до E9FFH.

Вся область BA (от BA0 до BA255) и вся область BB (от BB0 до BB255) может быть использована пользователем для своих целей.

Область BA\BB инициализируется только при полном стирании памяти!

8.2.4. Область BS/BT .

Область BS представляет собой область памяти длиной 256 слов во внутренней системной памяти RAM процессора. Она занимает адреса с EA00H до EAFFH.

Важно!
Можно изменять только системные слова данных BS1, BS60…BS63 и BS133:

- BS60…BS63 могут быть использованы для целей пользователя.
- BS1 и BS133 имеют жестко заданное значение, и на них влияет обработка программы. Вы можете записывать в них только определенные метки!

Важно!
Все остальные системные данные можно только считывать :

- Они содержат часть информации для системного программирования, часть системных переменных, значение которых закрыто.
- Изменение этих системных данных может повлиять на функционирование программируемого контроллера, а также на подключенный программатор!

Область BT представляет собой область памяти длинной 256 слов во внутренней системной памяти RAM процессора. Она занимает адреса с EB00H до EBFFH. Всю область BT (с BT0 до BT255) вы можете использовать для собственных целей.

Область BS\BT инициализируется только при полном стирании памяти!

Размещение систем данных в области BS

3S	Описание		Адрес
0	Слово битов индикации прерывания (STOER. USS)		EA00
1	Слово стирания индикации битов прерывания (UALW)		EA01
2	Слово общих битов индикации прерывания (UAMK)		EA02
3	Биты индикации меток- ошибок- запуска		EA03
4			EA04
5	Метка останова	Метка запуска	EA05
6	Метка цикла	Метка модуля	EA06
7	Метка полного стирания	Метка ошибок (H)	EA07
8	Метка ошибок (M)	Метка ошибок (L)	EA08
9	XXXXXXXXXXXXXXXXXXXXXXXX	Текущий идентиф. номер	EA09
10	Вазовый адрес формирователя входных сигналов		EA0A
11	Вазовый адрес формирователя выходных сигналов		EA0B
12	Вазовый адрес области отображения процесса входов		EA0C
13	Вазовый адрес области отображения процесса выходов		EA0D
14	Вазовый адрес области маркеров		EA0E
15	Вазовый адрес области таймеров		EA0F
16	Вазовый адрес области счетчиков		EA10
17	Вазовый адрес области интерфейсных модулей		EA11
18	Метка статуса	Версия SOFTWARE AG	EA12
19	Конечный адрес модуля памяти пользователя		EA13
20	Вазовый адрес системной области		EA14
21	Длина списка адресов DB		EA15
22	Длина списка адресов SB		EA16
23	Длина списка адресов PB		EA17
24	Длина списка адресов FB		EA18
25	Длина списка адресов OB		EA19
26	Длина списка адресов FX		EA1A
27	Длина списка адресов DX		EA1B
28	Длина блока данных списка адресов (DB0)		EA1C
29	Метка места установки	Метка CPU 2 (тип)	EA1D
30	Длина информационной части заголовка блока		EA1E
31	Метка CPU 1	Версия SOFTW Интерф. PG	EA1F
32	Вазовый адрес списка адресов DX		EA20
33	Вазовый адрес списка адресов FX		EA21
34	Вазовый адрес списка адресов DB		EA22
35	Вазовый адрес списка адресов SB		EA23
36	Вазовый адрес списка адресов PS		EA24
37	Вазовый адрес списка адресов FB		EA25

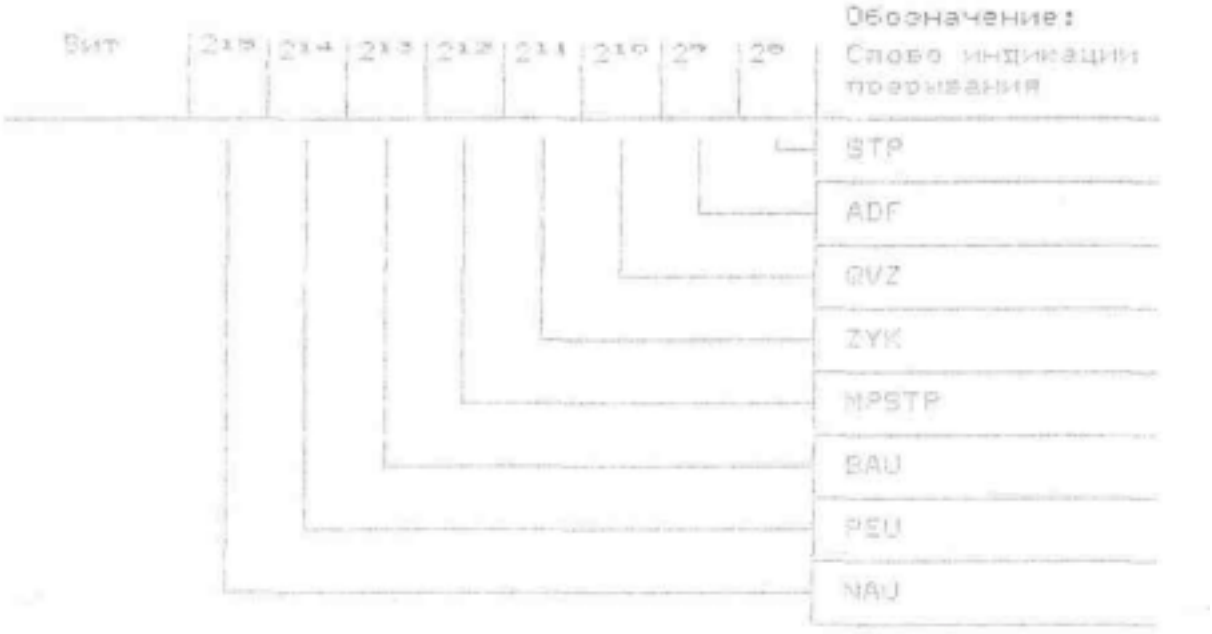
38	Вазовый адрес списка адресов OB	EA26
39 54	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	EA27 EA36
55	Счетчик на 1 час (до 3599 сек. 16-ричный)	EA37
56 59	Зарезервировано для блоков управления	EA38 EA3B
60 63	Зарезервировано для целей пользователя	EA3C EA3F
64 127	Зарезервировано для системных программ	EA40 EA7F
128	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	EA80
129	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	EA81
130	Метка «регулятор»	EA82
131	Метка «прерывание по времени»	EA83
132	Метка «прерывание от процесса»	EA84
133	Метка «обновление области отображения процесса»	EA85
134	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXX	EA86
135	Сл. индикации «Блокировка отдельн. прер-й по времени	EA87
136	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXX	EA88
137	Сл. индикации «Задержка отдельн. прер-й по времени	EA89
138 255	XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX	EA8A EAFF

XXXXX- закрытые области

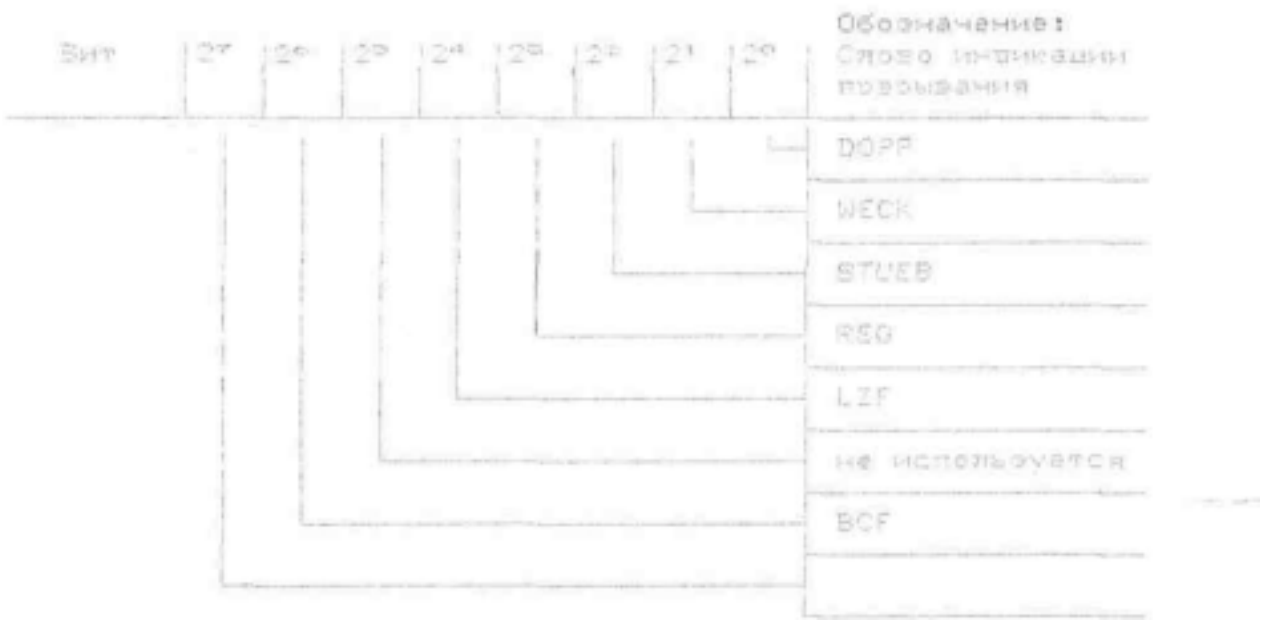
Информацию некоторых системных данных (о внутреннем строении процессора, версии матобеспечения, метки CPU и т.д.) Вы можете получить только с помощью on-lline- функции программатора “Системные параметры”.

Дополнением к вышеприведенному списку является показанная ниже конфигурация битов некоторых системных данных, которые пользователь может обработать с помощью команд языка STEP5 или с помощью программатора (сокращения- смотри Раздел 5.3)

Системное слово: BS 0 Адрес: EA00 (старший байт)

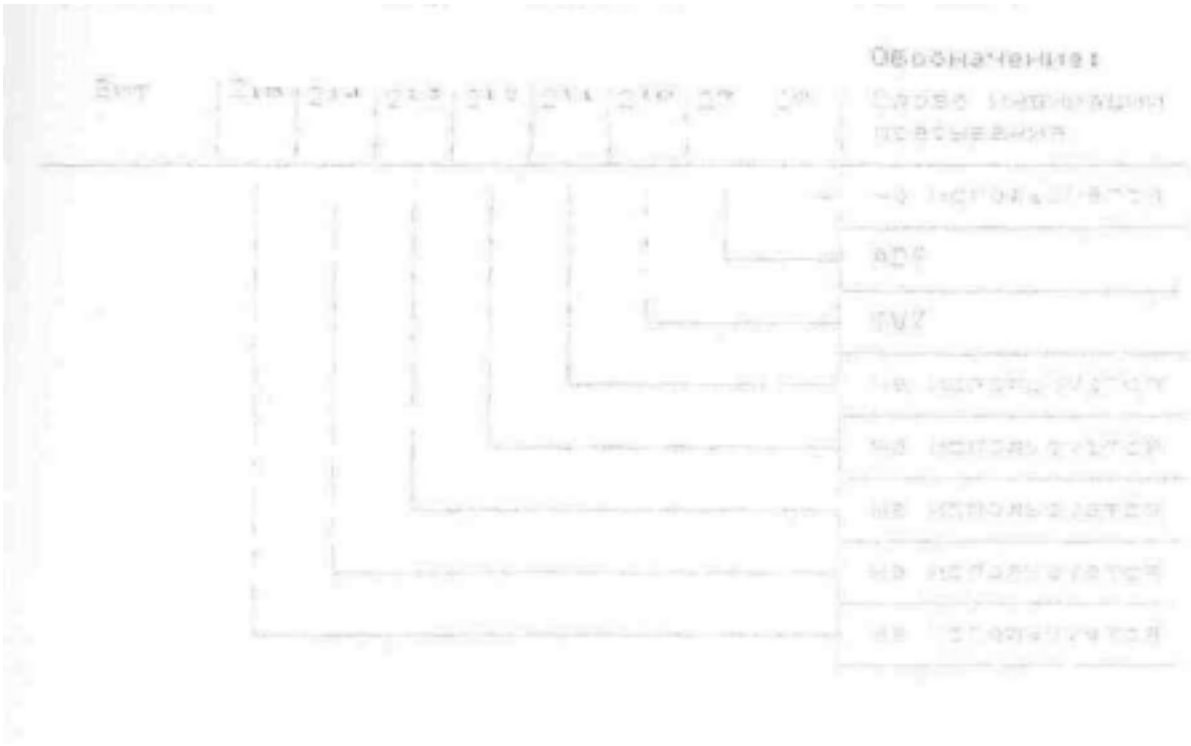


Системное слово: BS 0 Адрес: EA00 (младший байт)

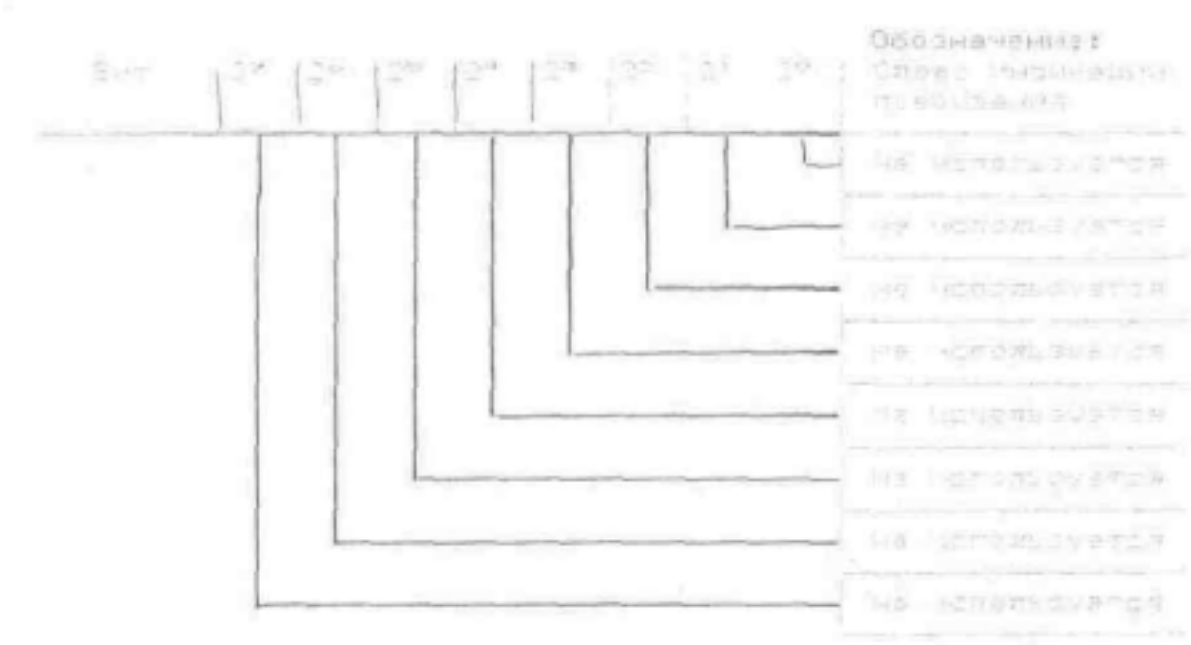


Слово системных данных BS0 указывает на причину прерывания при выполнении функции программатора “U-CTEK”. В случае возникновения при выполнении программы, например, ошибки выполнения, устанавливается бит 2^5. Если уровень программы LZF полностью обработан, бит 2^5 сбрасывается.

Системное слово: BS 1 Адрес: EA01 (старший байт)



Системное слово: BS 1 Адрес: EA01 (младший байт)



BS1: Активный интерфейс, разрешенный для пользователя (смотри дальше) !

Слово стирания битов индикации прерывания (слово системных данных BS1)

Установив бит 9 или 10, Вы можете вычислить UALW, которое игнорирует последующее QVZ или ADF и таким образом не изменяет текущую обработку программы. После возникновения QVZ или ADF операционная система сбрасывает указанный бит.

Каждый уровень обработки программ имеет собственное UALW!

В следующем примере проверяется, есть ли ответ модуля периферии. Если модуль отсутствует или неисправен, то с помощью UALW исключается задержка квитирования и обрабатывается предусмотренная для этого случая программа. Кроме того, тестируется, введены ли определенные адреса периферии в DB1. Если нет, то с помощью UALW игнорируется ошибка адресации, и выполняется специальная программа.

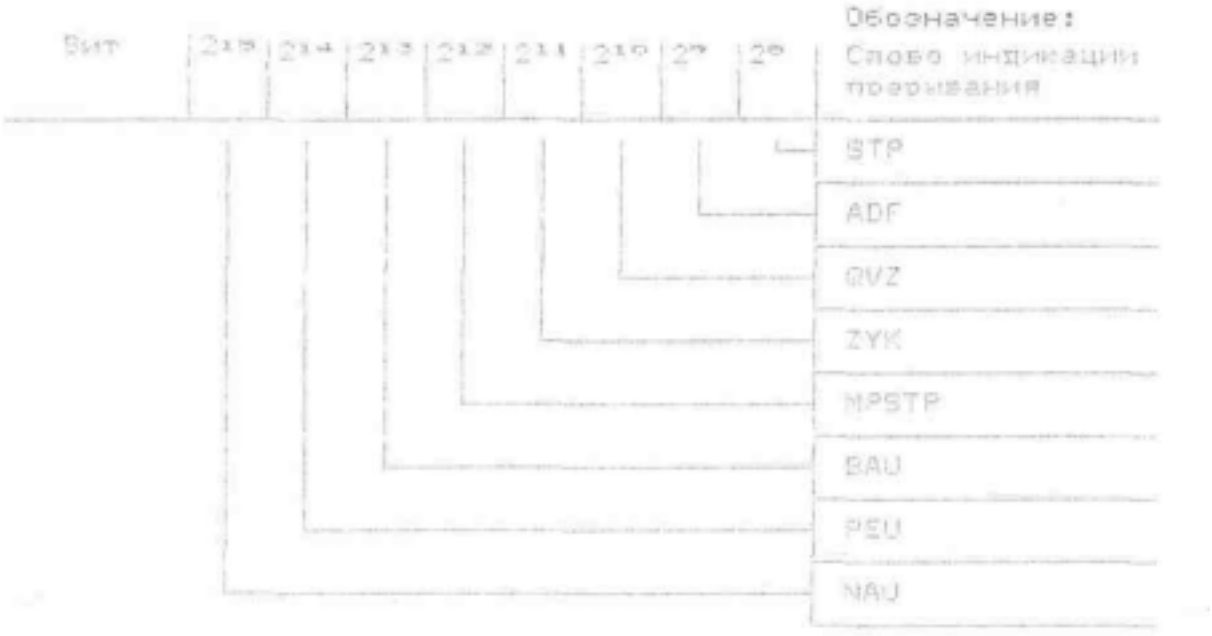
```
FB 10
Имя:          PERITEST
Метка:        PADR        E/A/D/B/T/Z:  E     BI/BY/W/D:    BY
Метка:        MASK        E/A/D/B/T/Z:  D     KM/KH/KY/KC/KF/KT/KZ/KG:     KM

              :L    BS1           UALW загружен
              :T    BS60          и сохранен
:L     =MASK      Бит QVZ или ADF установлен
              :OW
              :T    BS1           UALW перезаписан
              :L    =PADR Отдельное обращение к периферии или к
              :L    BS1           области отображения процесса
              :LW   =MASK         Вит QVZ или ADF маскирован
              :UW
              :T    BS1           Старое UALW перезаписано, для того, чтобы
              :                   опознать следующую ошибку QVZ или ADF
              :BE

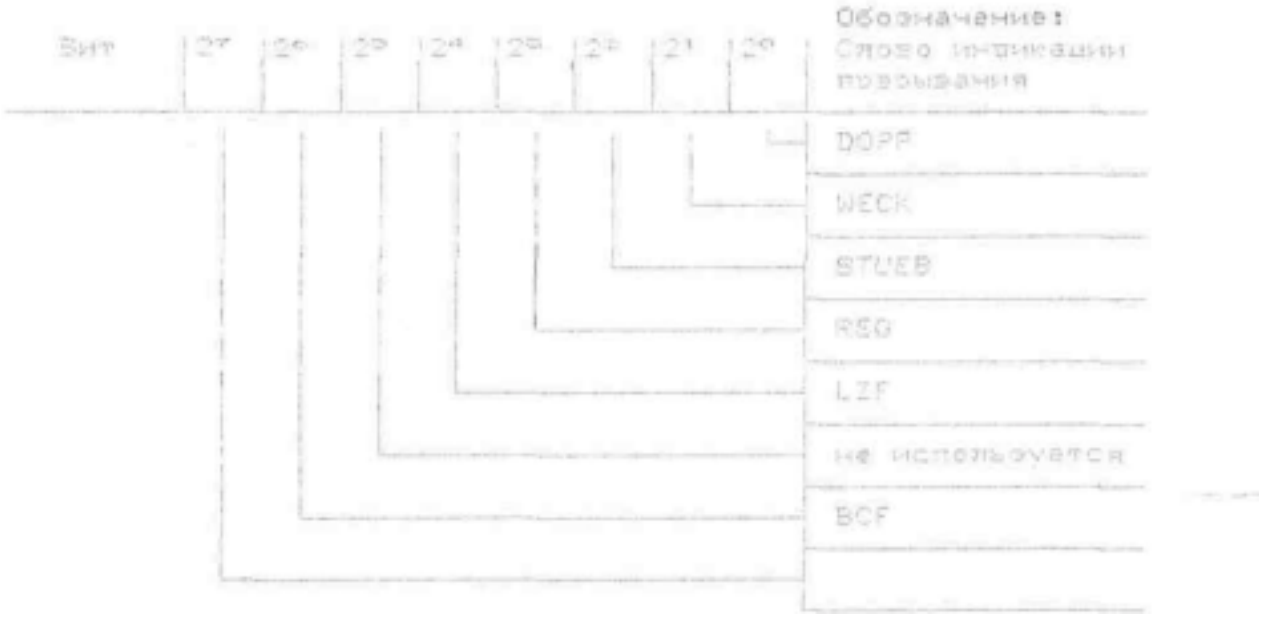
FB0
Имя:          :L

              :SPA  FB10
              :PERITEST                        Проверка
PADR          :     PB128                      отвечает ли модуль с адресом
MASK          :     KM00000100  00000000       периферии 128
              :SPN  =M001
              : . . .                          Эта часть программы выполняется,
              : . . .                          если модуль не отвечает
              : . . .
M001          :
              :SPA  FB10
имя           :PERITEST                        Проверка,
PADR          :     AB4                        введен ли в DB1 модуль с
MASK          :     KM00000010  00000000       адресом периферии 4
              :SPN  =M002
              : . . .                          Эта часть программы выполняется,
              : . . .                          если адрес периферии не введен
              : . . .
M002          :
              :BE
```

Системное слово: BS 2 Адрес: EA02 (старший байт)



Системное слово: BS 2 Адрес: EA02 (младший байт)



Общее слово индикации прерывания (UAMK в U-CTEK, смотри дальше) можно только считывать!

Общее слово битов индикации прерывания (слово системных данных BS2):

16 битов общего слова битов индикации прерывания соответствует причинам ошибок, которые можно вывести в U- CTEKe под названием «причине ошибок».

При возникновении неисправности устанавливается соответствующий бит.

Пример:

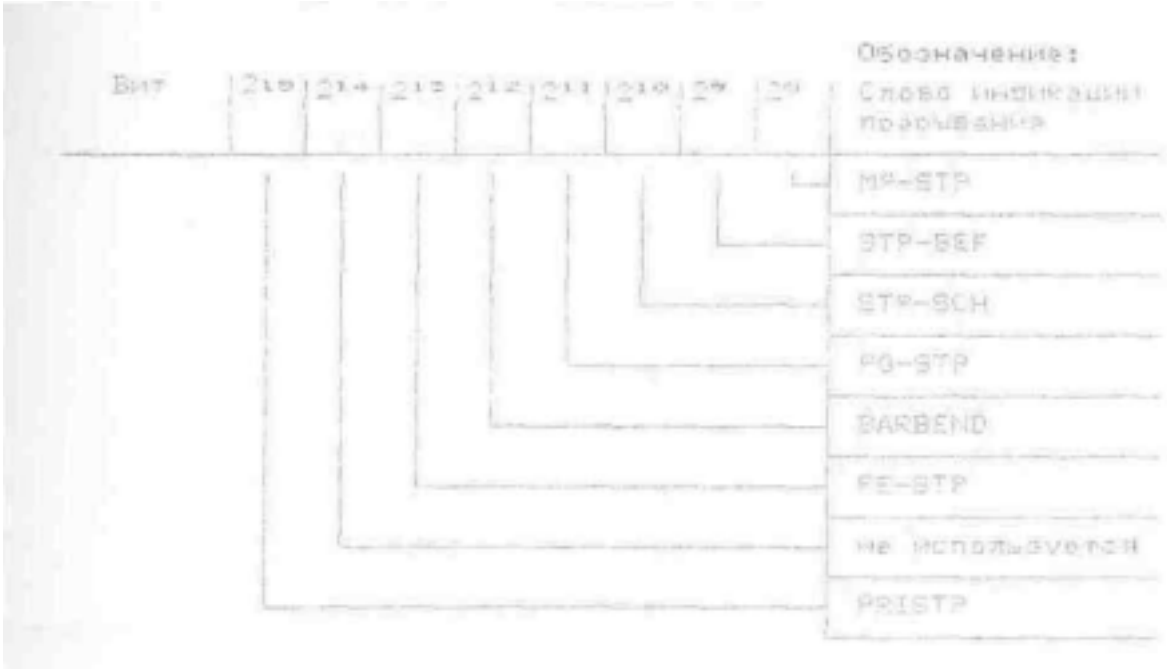
Если процессор переходит в случае ошибки (ADF) в состояние “СТОП”, то в UAMK устанавливается бит 9. Если теперь при обработке ADF возникнет ошибка кода команды (BCF), то кроме того в UAMK устанавливается бит 7.

Содержимое UAMK (двоичный вид):	00000010	10000000
Представление (16-ричн.) в U- CTEKe:	0280	

В это время в U- CTEKe под надписью “причина неисправности”, в месте, соответствующем этой ошибке, становится “X”. В UAMK все до этого возникшие ошибки суммируются (глубина U- CTEK 05: в UAMK установлены 5 битов). Путем преобразования 16-ричного кода в двоичный можно обратить содержимое UAMK. В этом случае Вы можете выяснить, какие ошибки явились причиной остановки

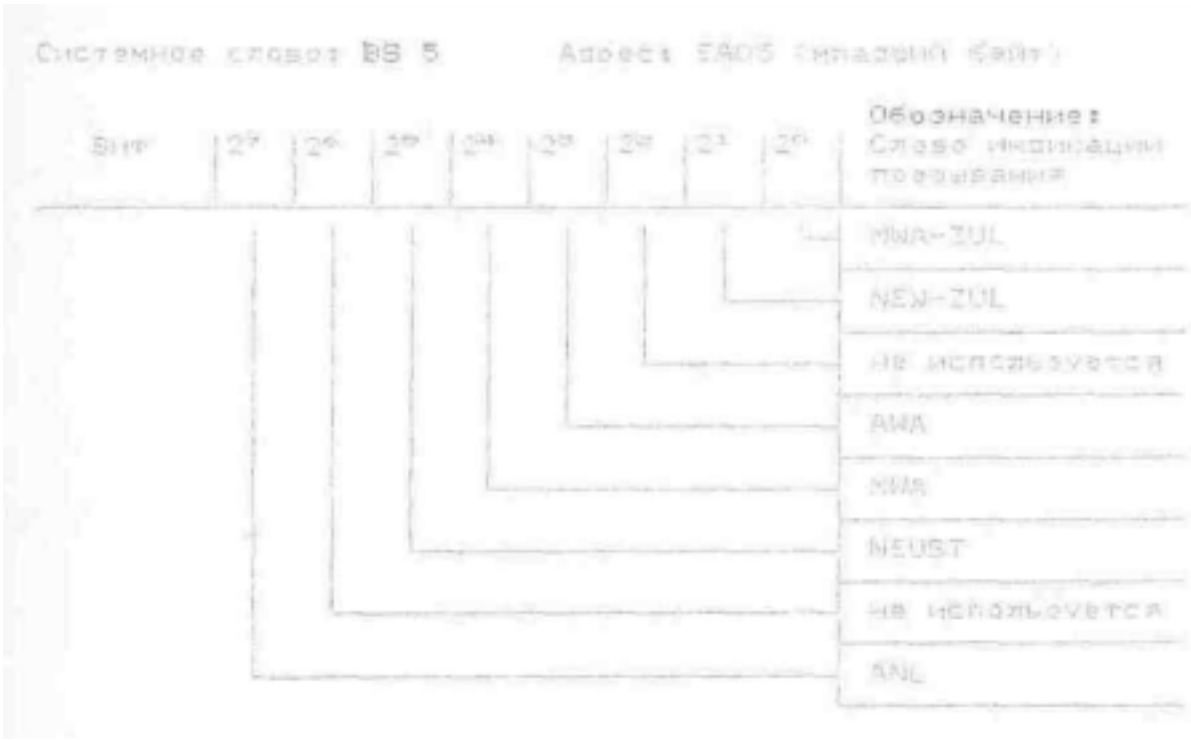
Биты ошибок сбрасываются, как только соответствующие данному уровню программы ошибки полностью обрабатываются, и программа переходит на другой уровень.

Системное слово: BS 5 Адрес: EA05 (старший байт)



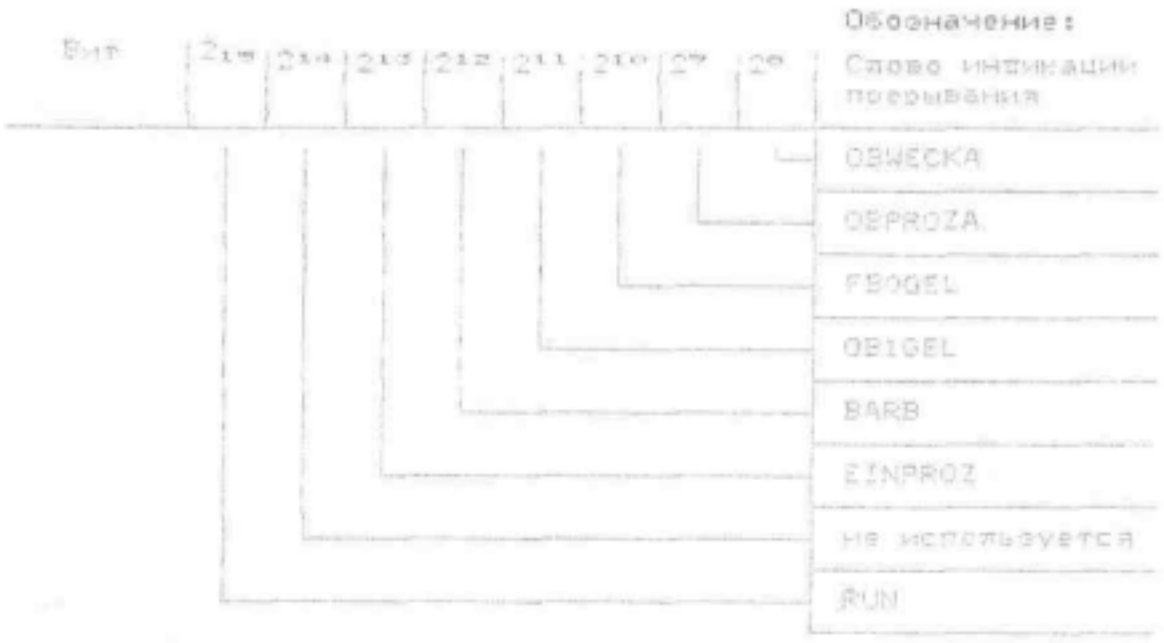
—> 1.Первая строка битов управления

Системное слово: BS 5 Адрес: EA05 (младший байт)



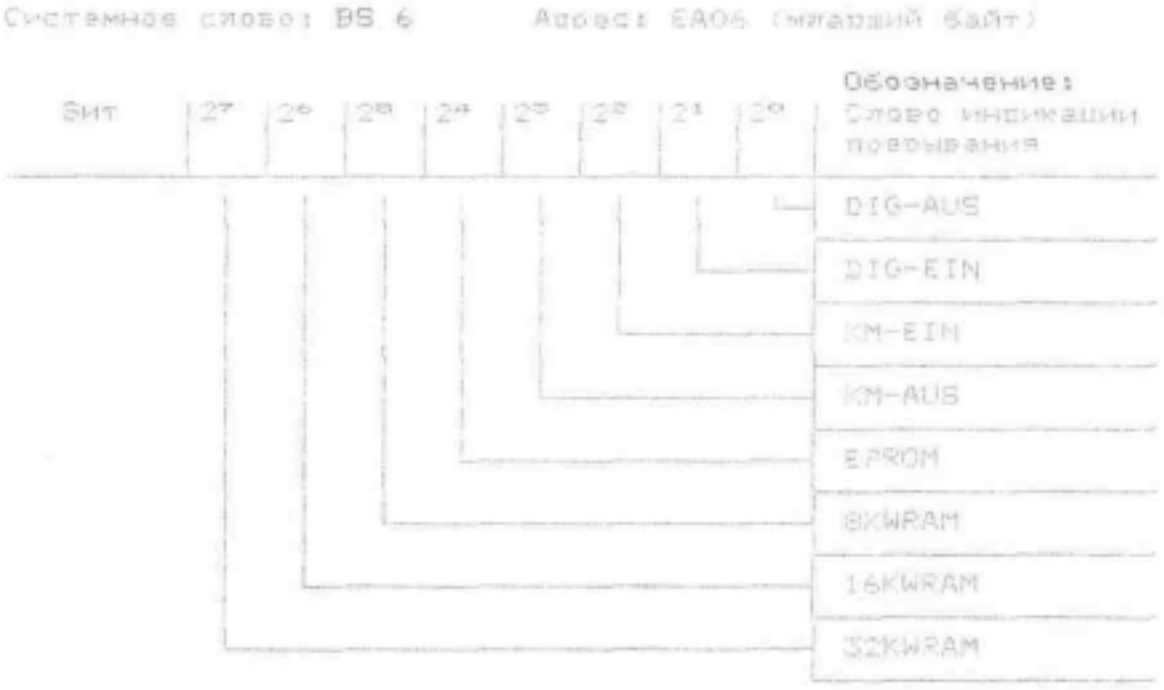
—> 2.Вторая строка битов управления

Системное слово: BS 6 Адрес: EA06 (старший байт)



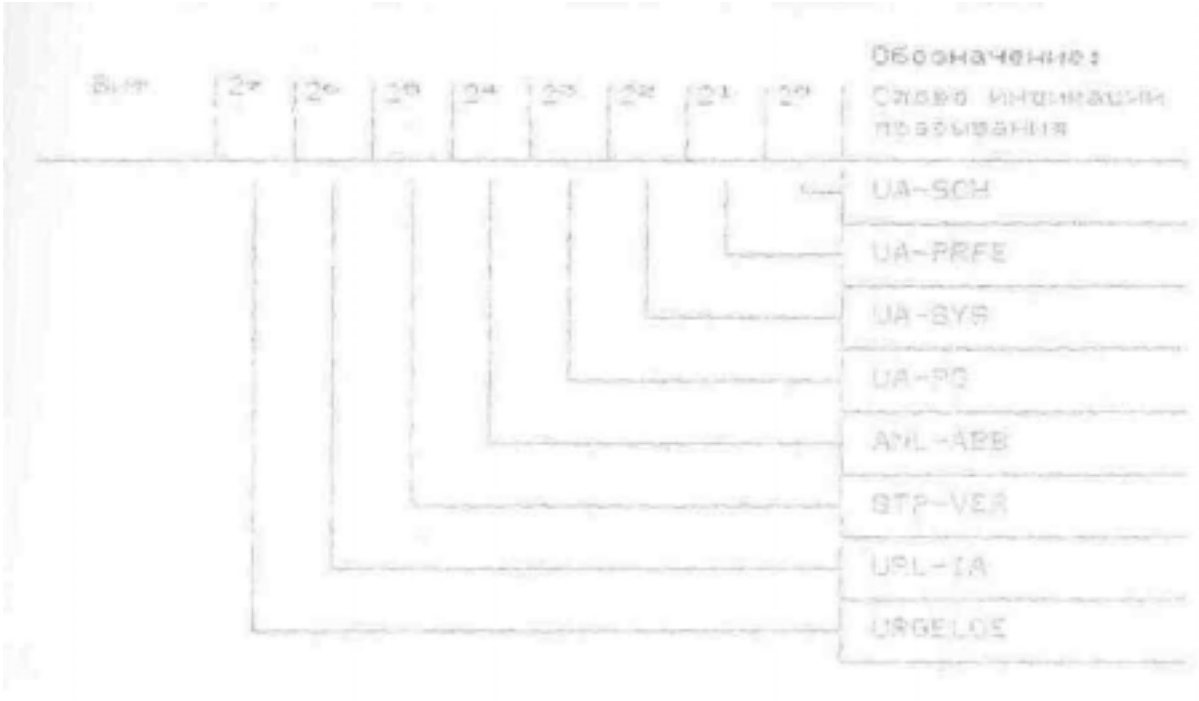
—> 3. Третья строка битов управления

Системное слово: BS 6 Адрес: EA06 (младший байт)



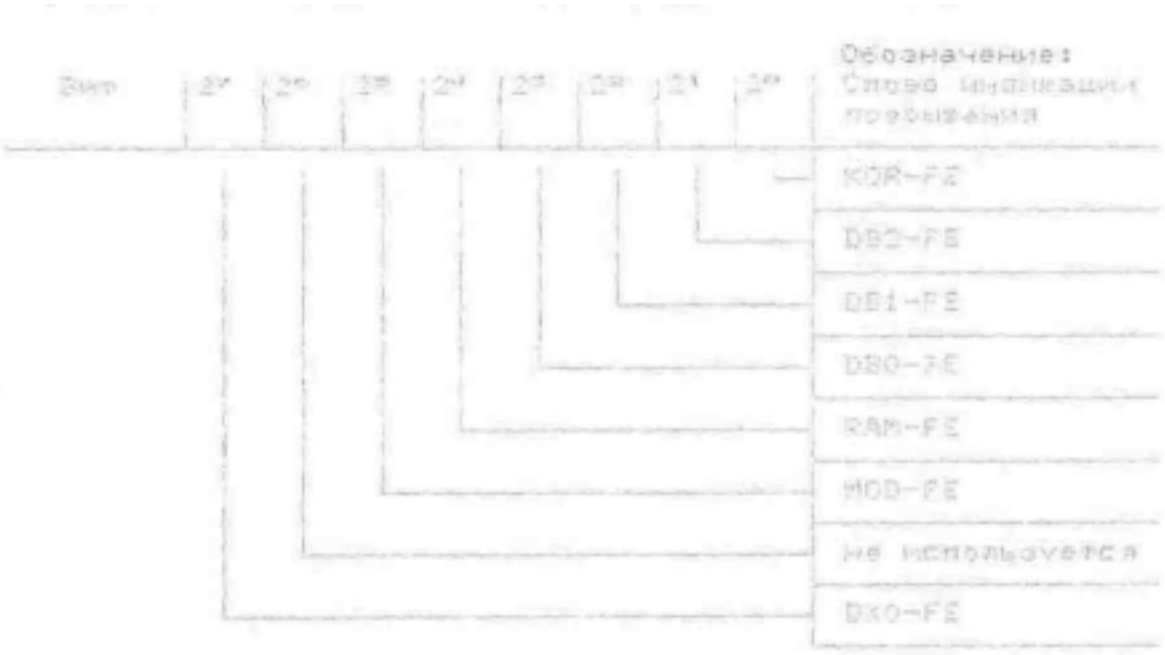
—> 4. Четвертая строка битов управления

Системное слово: BS 7 Адрес: EA07 (старший байт)



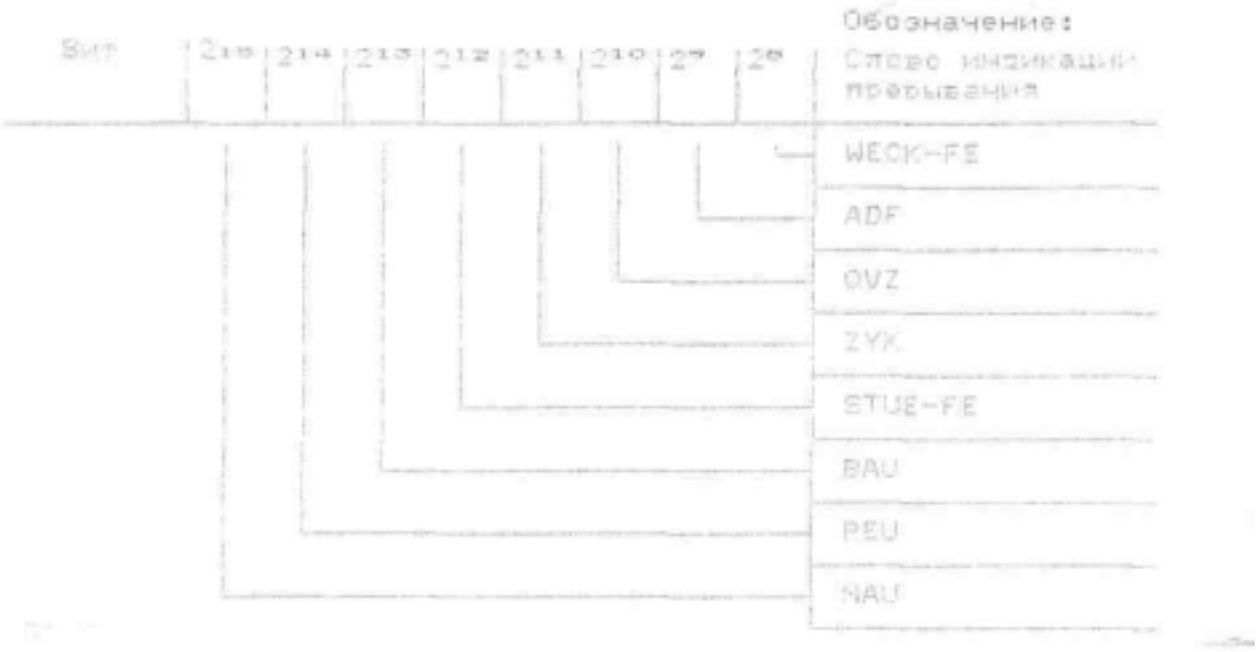
—> 5. Пятая строка битов управления

Системное слово: BS 7 Адрес: EA07 (младший байт)



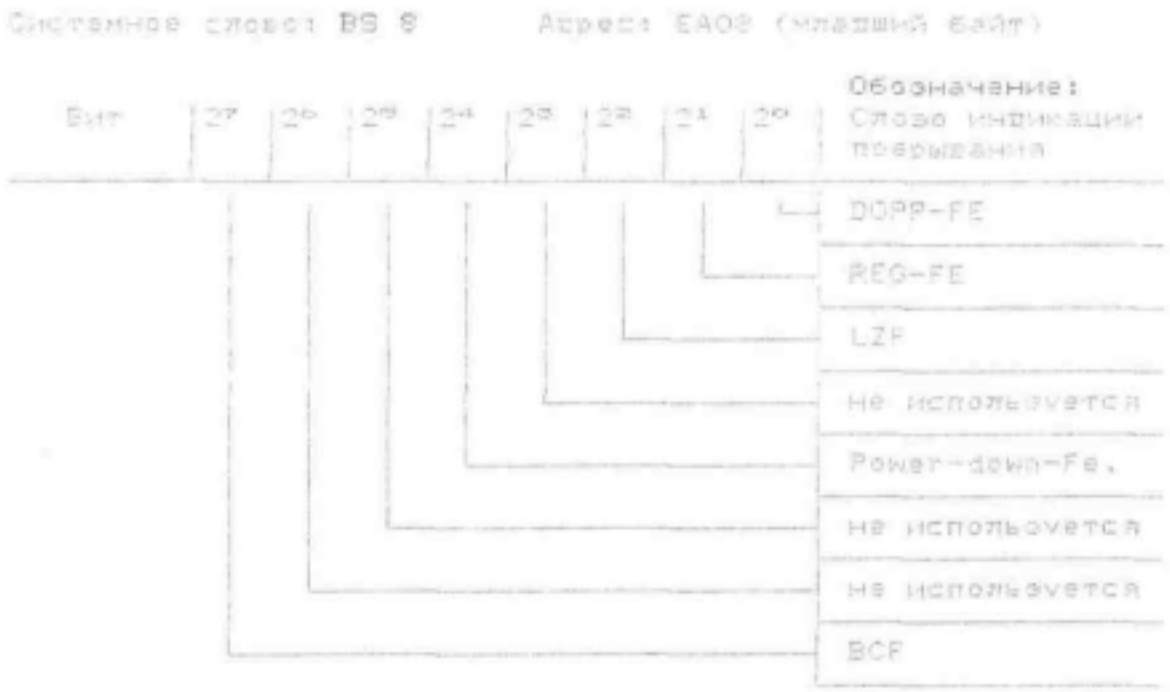
—> 6. Шестая строка битов управления

Системное слово: BS 8 Адрес: EA08 (старший байт)



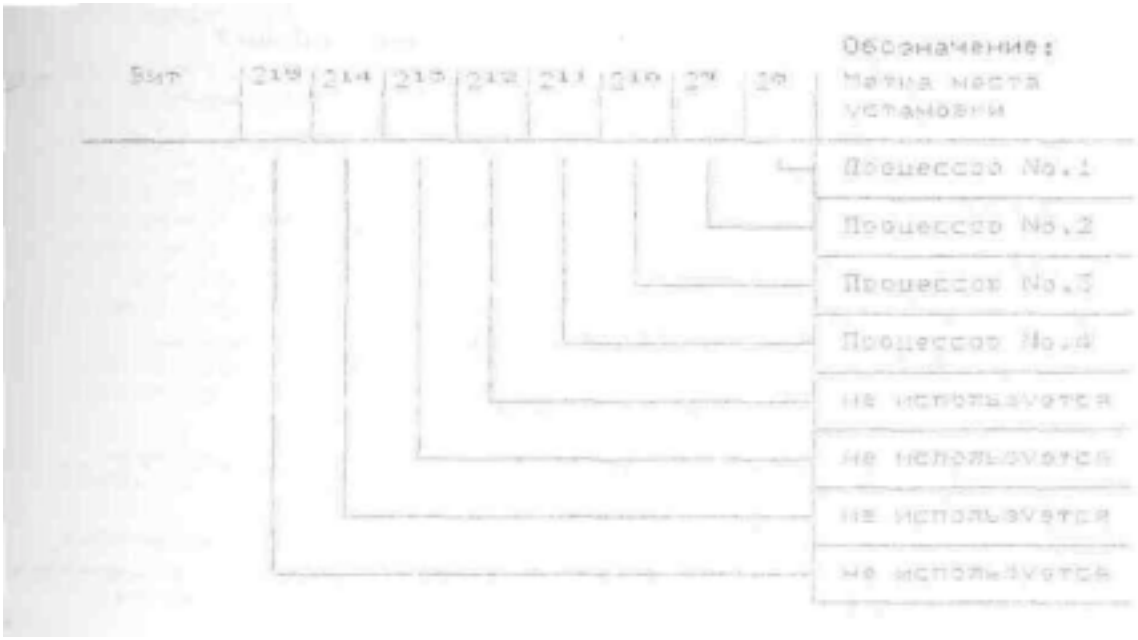
—> 7. Седьмая строка битов управления

Системное слово: BS 8 Адрес: EA08 (младший байт)



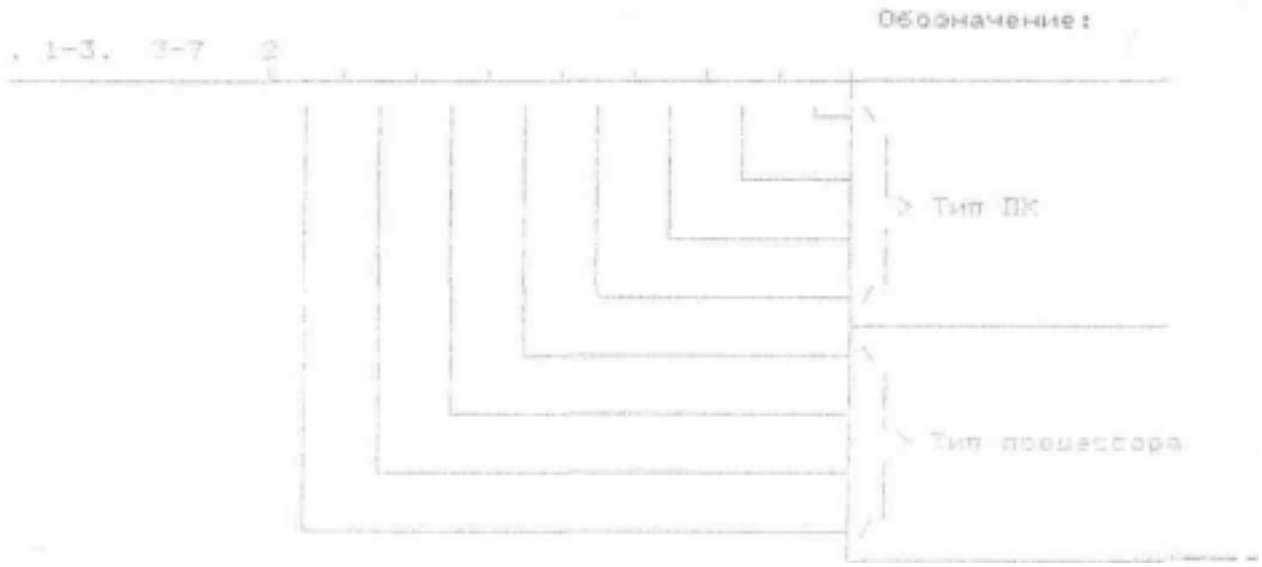
—> 8. Восьмая строка битов управления

Системное слово: BS 29 Адрес: EA1D (старший байт)



BS29 (старший байт): Активный интерфейс, который используется блоками управления и при многопроцессорной коммуникации, а также в OB 216 и командах SES и SEF.

Системное слово: BS 29 Адрес: EA1D (младший байт)



Тип ПК:
0 1 1 1 AG 135U

Тип процессора: 0 0 1 1 CPU 928

Слово системных данных: BS 130 Адрес: EA82 (младший байт)

Системные данные BS 130 выполняют информационную Функцию:

Вит 0=0: Активизация уровня обработки программы "регулирование"

Вит 0=1: Подавление уровня обработки программы " регулирование "

Перед вызовом организационного блока запуска (OB 20, 21 или 22) операционная система обрабатывает блок данных DB2 (если он имеется). В зависимости от результата обработки операционной системой устанавливается или сбрасывается BS130. Затем вызывается блок запуска.

Если BS 130 (младший) сброшен, то выполняется обработка регулятора в циклическом режиме согласно списку регулятора в DB2.

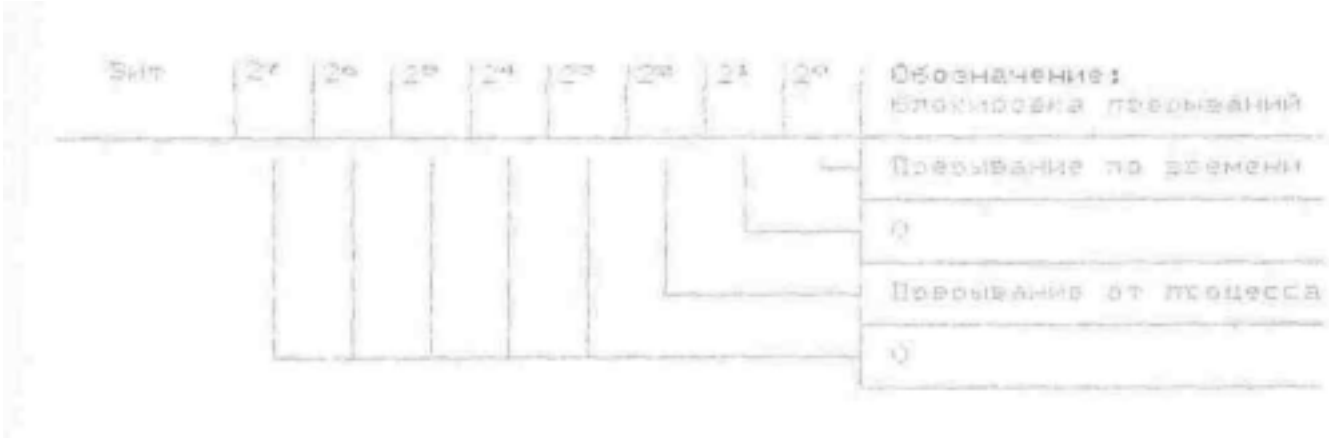
Слово системных данных: BS 131 Адрес: EA83

Системные данные BS 131 выполняют чисто информационную Функцию;

Слово индикации "Блокировка всех прерываний";

EA83 (старший байт)=0 см. Раздел 6.8.1 (OB 120)

EA83 (младший байт):



Бит =1 обозначает: Это прерывание блокировано!

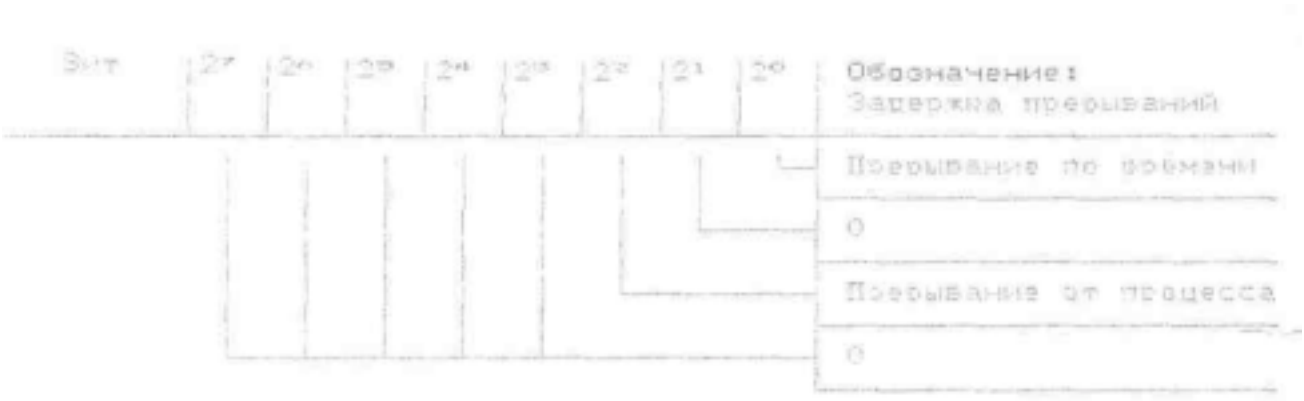
Слово системных данных: BS 132 Адрес EA84

Системные данные BS 132 выполняют чисто информационную функцию.

Слово индикации "Задержка всех прерываний":

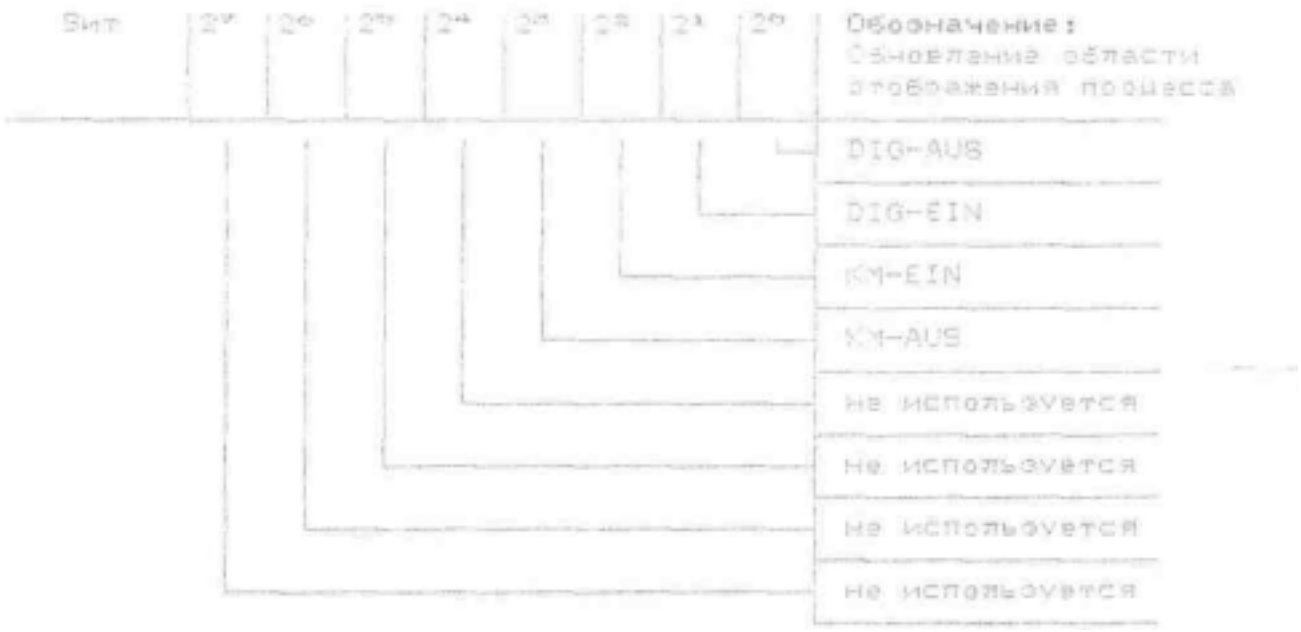
EA84 (старший байт) =0

EA84 (младший байт) :



Бит = 1 обозначает: Это прерывание задержано:

Слово системных данных: BS 133 Адрес EA85 (младший байт)



Бит 2^0 = 0: Следующее отображение .цифровых выходов выдается

Бит 2^0 = 1: Следующее отображение .цифровых выходов подавляется

Бит 2^1 = 0: Следующее отображение .цифровых выходов выдается

Бит 2^1 = 1 : Следующее отображение цифровых выходов подавляется

Бит 2^2 = 0 : Следующее отображение цифровых входных маркеров связи выдается
Бит 2^2 = 0 : Следующее отображение цифровых входных маркеров связи подавляется

Бит 2^3 = 0 : Следующее отображение цифровых входных маркеров связи выдается
Бит 2^3 = 0 : Следующее отображение цифровых входных маркеров связи подавляется

Указание: Каждый бит, если он установлен, однократно отменяет обновление области отображения процесса, а затем снова сбрасывается операционной системой в "0".

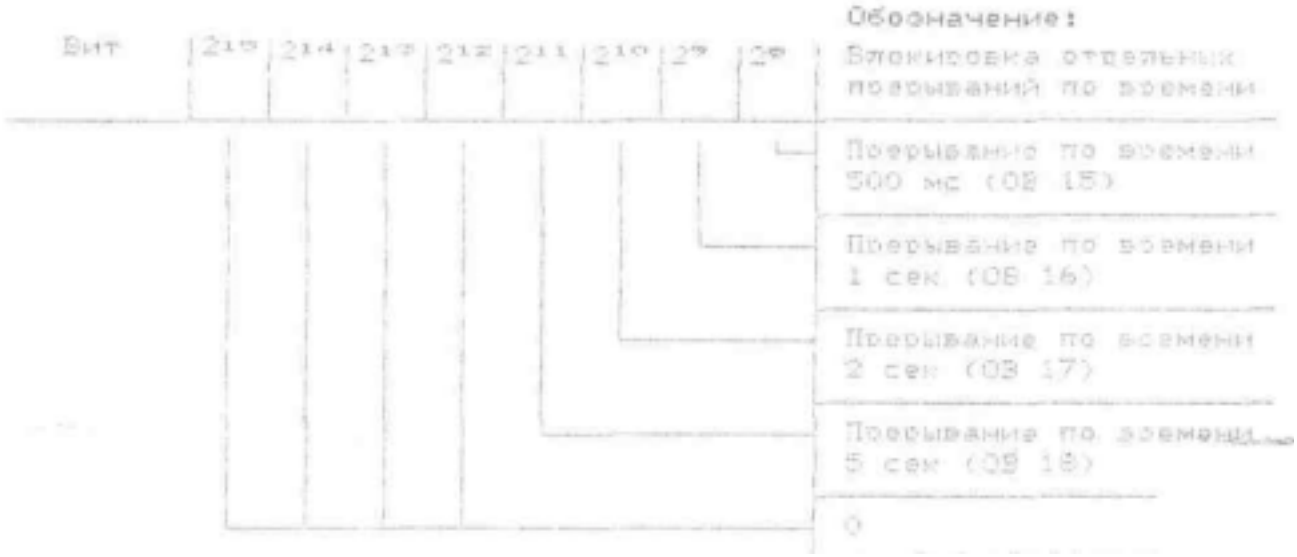
Системное слово: BS 135 адрес: EA87

Системные данные BS135 выполняют чисто информационную Функцию.

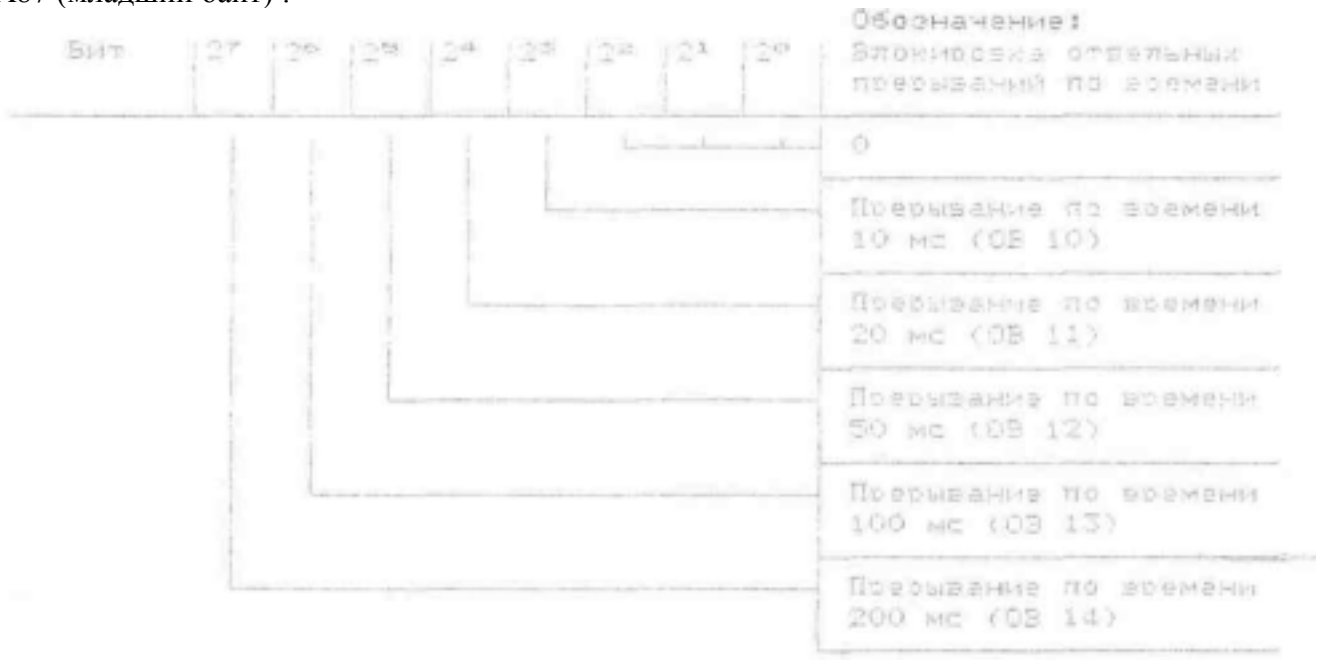
Слово индикации "Блокировка отдельных прерываний по времени" :

См. раздел 6.8.2 (OB 121)

EA87 (старший байт):



EA87 (младший байт) :



Бит = 1 обозначает: Это прерывание по времени блокировано!

Системное слово: BS 137 адрес EA87

Системные данные BS 137 выполняют чисто информационную функцию.

Слово индикации "Задержка отдельных прерываний по времени":

См. раздел 6.8.2 (OB 121)

Слово системных данных BS 137 аналогично BS 135. Если бит установлен в «1», то обработка соответствующего прерывания по времени задерживается. Если бит сброшен в «0», то соответствующее прерывание по времени обрабатывается.

9. Обращение к памяти по абсолютным адресам

Язык программирования STEP '5 содержит операции, позволяющие выполнять обращения ко всему адресному пространству.

ВАЖНО!

При неправильном применении эти команды могут несанкционированно переписать блоки на языке STEP 5 и системные данные. Это может привести к непредсказуемым последствиям. Поэтому операции, которые работают с абсолютными адресами, должны применяться только пользователями с очень хорошими знаниями системы.

Локальная память

Локальной памятью называется область памяти, которая имеется в каждом процессоре (модуль памяти пользователя, DB—RAM, области BA,. BB, BS, BT, счетчики, таймеры, маркеры, область отображения процесса).

Глобальная память

Глобальная память одна для всех процессоров и адресуется она по шине S5.

Организация памяти

Области памяти могут быть организованы побайтово и пословно.
Побайтовая организация: Каждый адрес адресует байт.
Пословная организация: Каждый адрес адресует слово.
Организация локальных областей памяти жестко задана см. раздел 8 "Заполнение и организация памяти"). Организация глобальной памяти зависит от типа установленных модулей.

внутренняя, локальная	внешняя, глобальная,
повторяется много раз	с доступом по шине S5
(для каждого процессора)	(одна на весь программируемый контроллер)

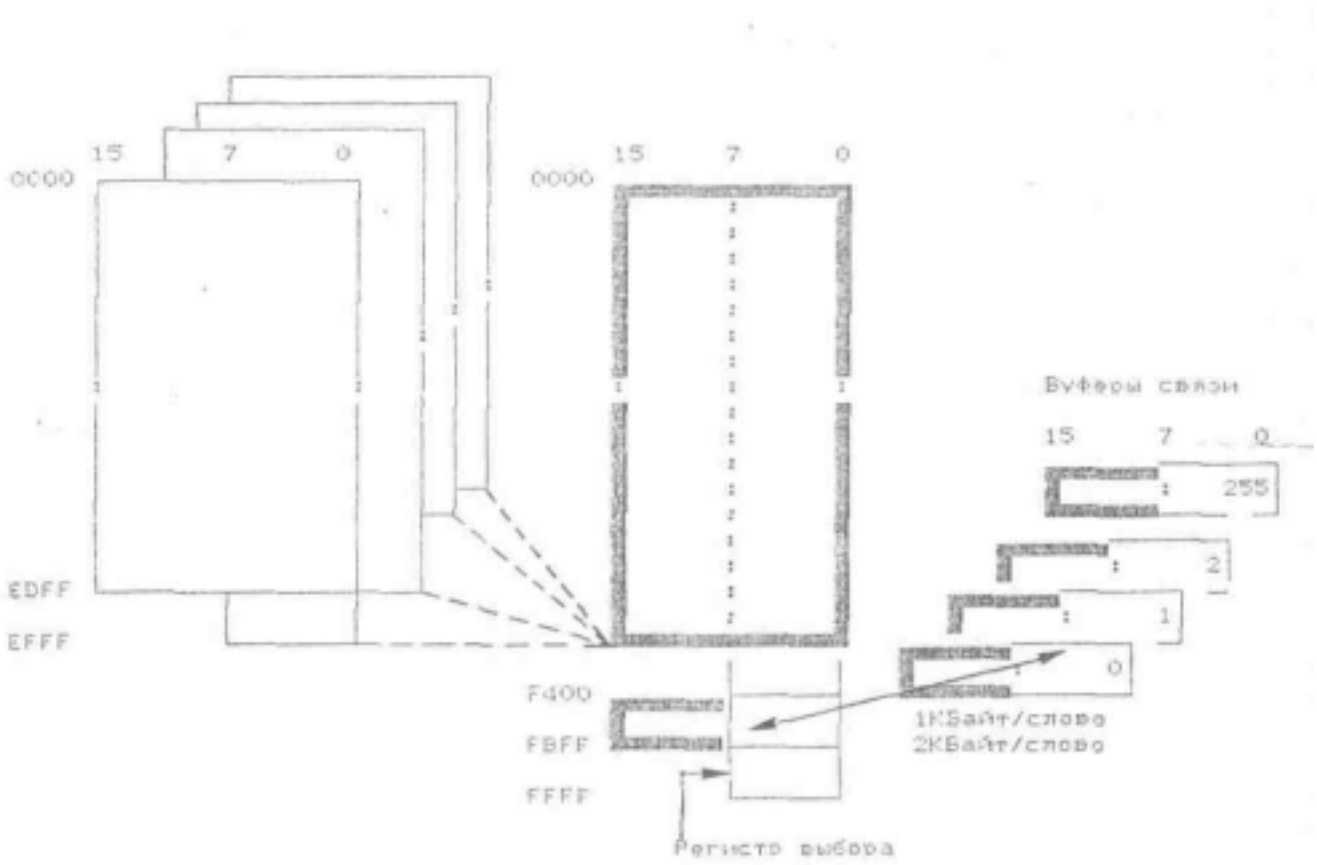


Рис. 9-1: Глобальная и локальная память

Следующие команды позволяют обращаться к локальной или глобальной области памяти по абсолютным адресам.

a) локальной области памяти (0000 по EFFF) и побайтово организованной части глобальной области (F0000 по F3FF, FC00 по FFFF):

TNB, TNW, LIR, TIR

b) к пословно организованной части локальной области (0000 по EDFF):

LRW, TRM, LRD, TRD

c) к побайтово организованной части глобальной области (0000 по EFFF):

LB GB, LB GW, LB GD, TB GB, TB GW, TB GD, TSG

d) к пословно организованной части глобальной области (0000 по EFFF):

LW GW, LW GD, TW GW, TW GD, TSG

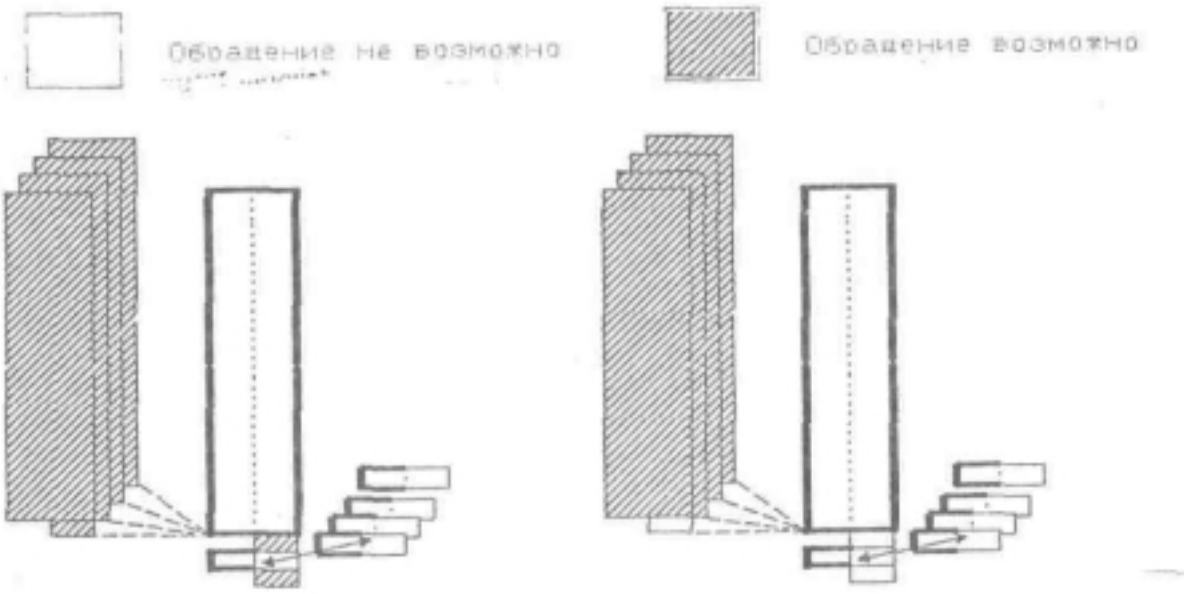
e) к побайтово организованной части глобальной области (F400 по FBFF,= область буферов связи):

LB CB, LB CW, LB CD, TB CB, TB CW, TB CD, TSC

t') к пословно организованной части глобальной области (F400 по FBFF,= область буферов связи):

LW CW, LW CD, TW CW, TW CD, TSC

Обращение к локальным или глобальной области памяти по абсолютным адресам



a) LIR, TIR, TNB, TNW

b) LRW, TRW, LRD, TRD

c) LB GB, LB GW, LB GD
TB GB, TB GW, TB GD, TSG

d) LW GW, LW GD
TW GW, TW GD, TSG

c) LB CB, LB CW, LB CD
TB CB, TB CW, TB CD, TSC

d) LW CW, LW CD
TW CW, TW CD, TSC

9.1. Обращение к регистрам и памяти по адресу в аккумуляторе 1

Регистром называется особая ячейка памяти, которая используется процессором при обработке программы на языке STEP5. Каждый регистр имеет ширину 16 бит. С помощь системных функций LIR (косвенная загрузка регистра) и TIR (косвенная передача из регистра) Вы можете обращаться к содержимому регистра.

LIR 0 . . . 15 Загружает в регистр 0 . . . 13 содержимое ячейки памяти, адресованной через аккумулятор 1-L.

TIR 0 . . . 15 Передает в ячейку памяти, адресованную через аккумулятор 1-L, содержимое регистра 0 . . . 15.

Ячейка памяти находится либо в локальной области (0000 по EFFF) либо в побайтово организованной части глобальной области (F000 по F3FF, FC00 по FFFF).

LIR и TIR в области буферов связи

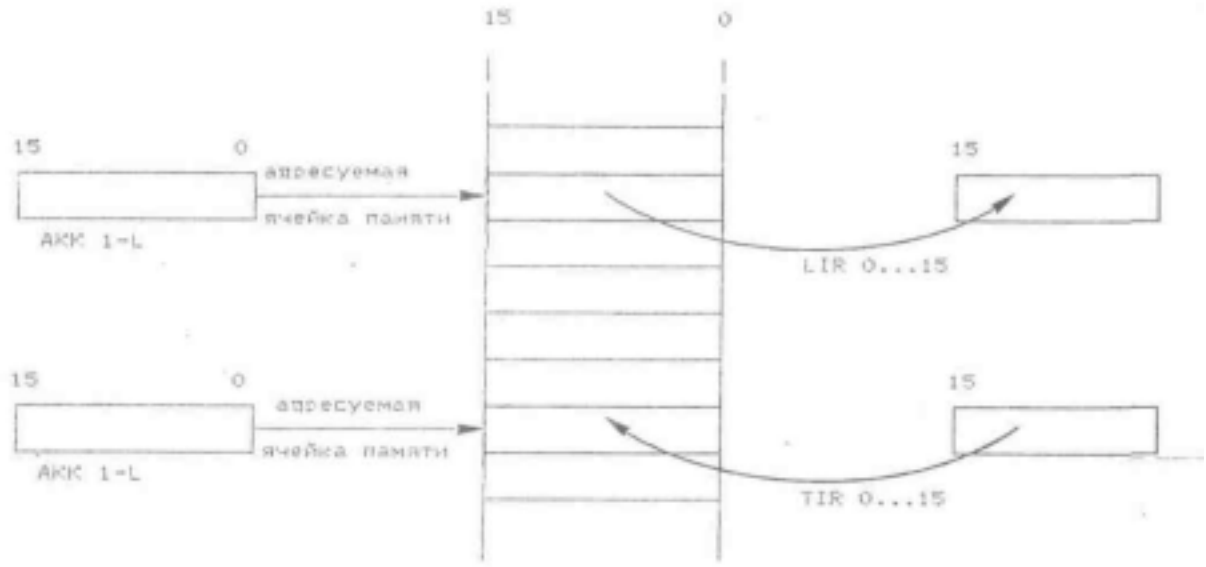
Команды LIF и TIR в многопроцессорном режиме AG 135U не могут быть использованы для обращения к области буферов связи (F4000- FBFF). В этом случае пользуйтесь командами из главы 9.3.5 "0бращение к буферам связи" или спецфункциями из главы 6.6 "Доступ к буферам связи".

Следующая таблица показывает структуру регистров для CPU 928. Они отличаются от регистров S или R процессоров.

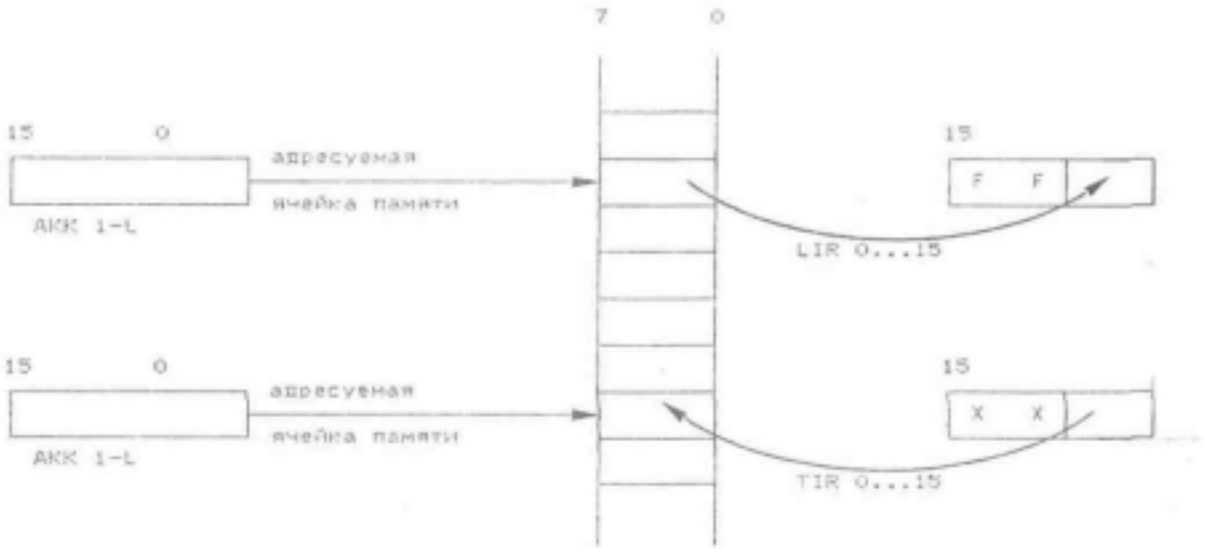
Номер регистра	Структура регистра
Регистр 0:	Аккумулятор 1-H (левое слово аккумулятора 1, бит 16-31)
Регистр 1:	Аккумулятор 1-L (правое слово аккумулятора 1, бит 0-15)
Регистр 2:	Аккумулятор 2-H
Регистр 3:	Аккумулятор 2-L
Регистр 6:	DBA (начальный адрес блока данных)
Регистр 8:	DBL (длина блока данных)
Регистр 9:	Аккумулятор 3-H
Регистр 10:	Аккумулятор 3-L
Регистр 11:	Аккумулятор 4-H
Регистр 12:	Аккумулятор 4-L
Регистр 15:	SAZ (счетчик адресов STEP)

Регистры 4, 5, 7, 13 и 14 не используется или отсутствуют. Если команды LIR/TlR обращаются к этим номерам регистров, то они будут обработаны как нулевые операнды (NOP).

LIR и TIR на 16-битовой области памяти



LIR и TIR на 8-битовой области памяти



Если Вы обращаетесь с помощью команд LIR/TIR к области памяти шириной 8 бит (адреса памяти >= EF00H), то помните, что

- по команде TIR передается только младший байт регистра (старший байт регистра теряется), и
- по команде LIR в старший байт регистра записывается FFH.

o Регистры с 0 по 3 и с 9 по 12: АКК 1, 2, 3 и 4

Аккумуляторы используются процессором при обработке программы как промежуточная память или как буфер. С помощью команд LIR и TIR Вы можете передавать содержимое аккумуляторов в абсолютно адресованную ячейку памяти или содержимое абсолютно адресованных ячеек памяти загружать в АКК. Абсолютный адрес ячейки памяти находится в АКК 1 .

Пример:

Содержимое ячейки памяти с адресом A000 необходимо загрузить в слово маркеров MW100.

```
:L     KHA000     Адрес A000 ячейки памяти загружен в АКК1
:LIR   1          Содержимое ячейки памяти, адресованной через АКК1 загружается в
:                 регистр 1 = АКК1
:
T      MW100      Содержимое адреса A000 заносится в маркерное слово
:                 MW 100
:BE
```

Пример:

Содержимое маркерного слова 200 необходимо передать в ячейку памяти с адресом A000.

```
:L     MW200      Маркерное слово MW200 загружено в АКК1
:L     KHA000     Адрес A000 по которому должна производиться передача, загружен в
:                 АКК1 (маркерное слово MW 200 сдвигается в АКК2)
:
TIR    3          Содержимое регистра 3 = содержимому АКК2 передается в адресную с
:                 помощью АКК1 ячейку памяти
:
:BE
```

o Регистр 6: DBA (начальный адрес блока данных)

При вызове блока данных с помощью команд A DB и AX DX регистр 6 загружается адресом DW0, т.е. нулевого слова вызванного блока данных. Этот адрес содержится в списке адресов блоков в DB 0.

Регистр DBA перед каждым вызовом 0B 1 или FB 0 сбрасывается в ‘0’

DBA —регистр остается неизменным, если

—по команде перехода (SPA/SPB) обработка программы продолжается в другом блоке

—включается другой уровень обработки программы,.

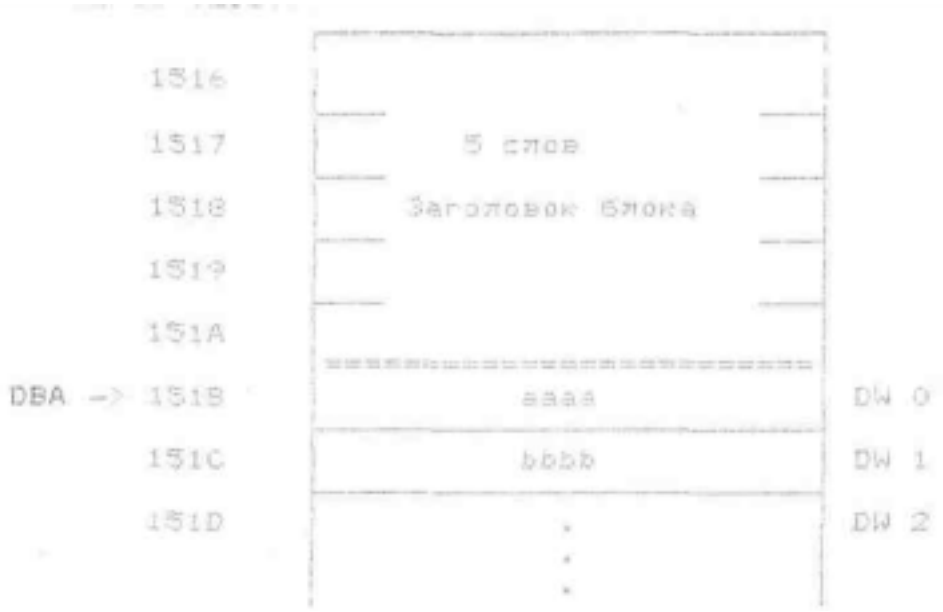
Он изменяется, если

—открывается другой блок данных или

—происходиТ возврат в вызывающий блок после того как в вызванном блоке был открыт новый блок данных (см. главу 2.4.3. Область действия блоков данных)..

Пример: AX DX17

Адрес (шест.) DX17



При вызове DX 17 в DBA- регистр будет занесен в адрес слова памяти, в котором находится DW 0, в примере: DBA = 151B (в .шестнадцатеричном виде).

Указание: В USTACKe под меткой 'DB-ADR' указан адрес, хранящийся в DBA регистре.

Обращение к словам данных происходит обычным образом с помощью команд STEP 5 L/T DW, L/T DR, L/T DL, L/T DD. U/O/ON/ON/=/S/R/DX.Y. Однако это допустимо до слова данных DW 255. манипулируя регистром DBA, Вы можете обращаться с помощью этих команд к словам данных > 255.

Пример:

Изменяя регистр 6, загрузим слово данных DB 300 блока данных DB100.

FB 7

NAME: LIR/TIR6

:L	BS34	Начальный адрес списка адресов DB плюс 100
:ADD	BF+100	задает адрес DB 100 в списке адресов
:LIR	1	Начальный адрес DB 100 (DWO) в АКК1
:ARR	KF+200	Адрес DB 200 в DB 100 расположен в слове
:T	BS 62	системных данных BS 62
:L	BS 20	Загружен базисный адрес системных данных
:ADD	KF+62	Адрес BS 62 загружен *в* аккумулятор 1
:LIR	6	Регистр DBA загружается содержимым с адресом
:		из BS 62. т.е.
:		начало блока данных установлено на DW 200
:L	DW100	DW (200+100)=DW 300 загружен
:T	MW100	DW 300 занесен в маркерное слово MW 100
:BE		

Внимание:

Если Вы установите регистр DBA как показано выше, то регистр DBL. (см. ниже) не изменится. Вместе с тем. контроль ошибки передачи больше не работает!

Используя спецфункцию 0B 130 "Переменное обращение к блоку данных", Вы можете сдвигать DBA—регистр на заданное количество слов данных. Так как OB 180 изменяется одновременно с DBL-регистром, то ошибка передачи контролируется в дальнейшем.

Пример:

FB 7

```
NAME:OB180
      :A    DB100     Регистры DBA и DBL загружены значениями
      :L    KF200     DB 100 н с помощью OB180 DBA регистр
      :SPA  OB180     Увеличивается на 200, а DBL-регистр
      :               на 200 уменьшается
      :SPB  = FEHL    Переход на обработку ошибки, если DB 100
      :               содержит меньше 200 слов данных
      :L    DW100     DW 300 загружается и заносится
      :T    MW100     в MW100
      :BEA
FEHL  :               Часть программы для обработки ошибок
      :
      :BE
```

o Регистр 8 : DBL = длина блока данных

Дополнительно к регистру DBA при каждом вызове блока данных загружается регистр DBL. Он содержит длину вызванного слова данных без заголовка.

Регистр DBL перед каждым вызовом OB 1 или FB 0 сбрасывается в'0'.

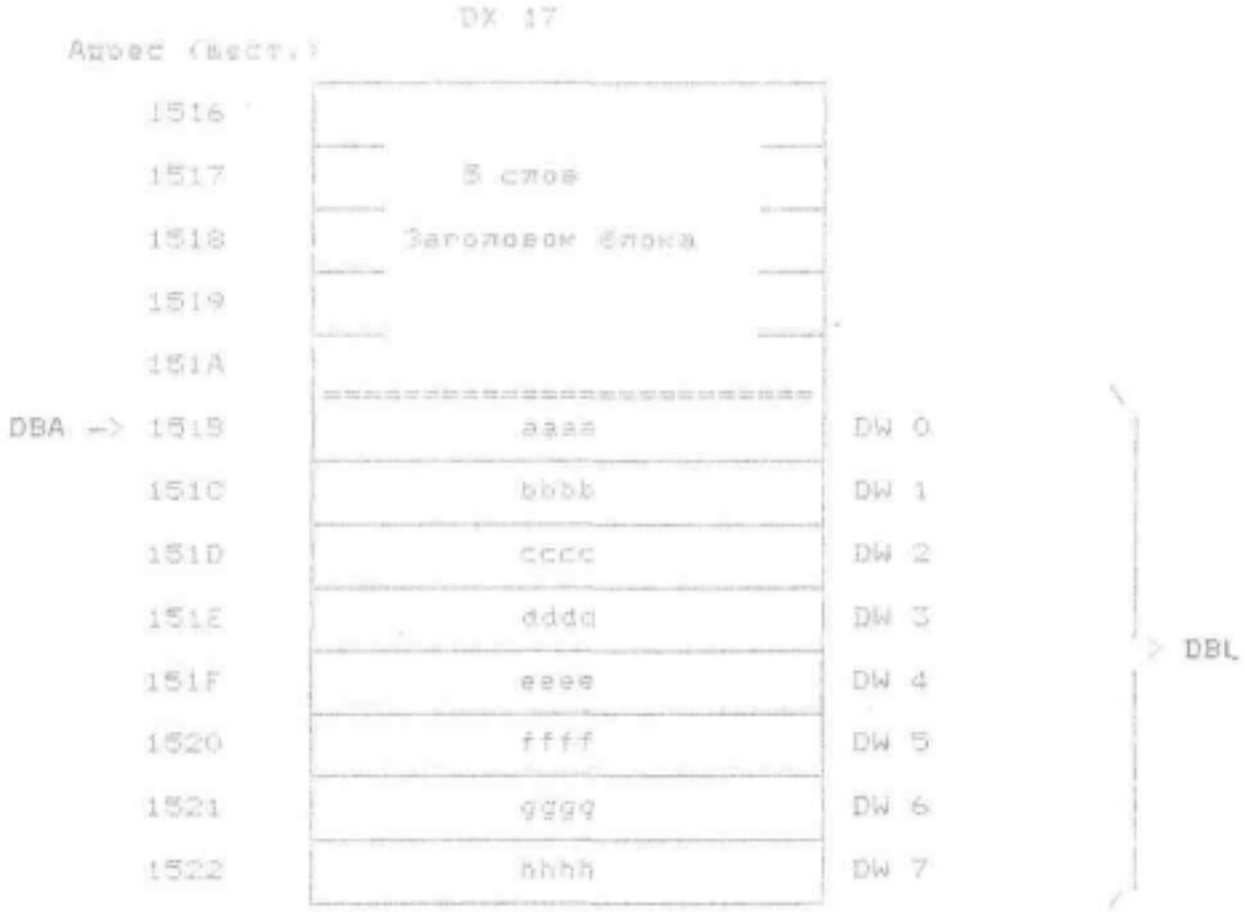
Регистр DBL не меняется, если

- по команде перехода (SPA/SPB) обработка программы продолжается в другом блоке
- включается другой уровень обработки программы.

Он изменяется, если

- открывается другой блок данных или
- происходит возврат в вызывающий блок после того как в вызванном блоке был открыт новый блок данных (см. главу 2.4.3. "Область действия блоков данных).

Пример: AX DX17



При вызове DX17 в DBL— регистр вводится имеющееся количество слов данных, в примере: DBL = 8 (DW 0 до DW 7).

Указание: В USTACKe под меткой 'DWL-REG' указано число, хранящееся в DB—регистре.

o Регистр 15: SAZ = счетчик адресов STEP

В регистр 15 во время обработки программы STEP5 заносится абсолютный адрес той команды в памяти программы, которая будет обрабатываться следующей.

Пример: Все слова данных блока данных заменяются константами.

Представленная ниже программа заменяет все слова данных DB 50 на константы КН A5A5. После изменения выделенные команды STEP5 могут также использоваться для перезаписи других блоков данных (DB или DX) на любые значения. Отсутствующие блоки опознаются; и это приводит к переходу на метку NIVO.

Начальный адрес (DBA) и длина (DBL) блока данных определяется с помощь специальной функции ОВ 181 'Проверка БЛОКОВ данных (DB/DX)[1].

Программа использует все четыре аккумулятора. На рисунке Вы видите заполнение аккумуляторов во время обработки программы по метки SCHL. Внутри цикла содержимое аккумулятора не изменится. АКК 1 содержит сначала адрес последнего слова данных (DBA + DBL-1) и уменьшается при каждом выполнении цикла на единицу.

АКК2 содержит адрес первого слова данных (DBA). Цикл завершается как только содержимое АКК 1 становится меньше содержимого АКК 2. Для перезаписи слов данных используется команда TIR 10, которая запоминает содержимое аккумулятора 3-L (константа) по хранящемуся в АКК 1-L адресу.

```
                .
                .
                .
        :L      KHA5A5      Константы, на которые должны быть переписаны все слова
        :                   данных
        :L      KY1, 50     Тип и номер блока данных
        :ENT
        :SPA    OB181       Специальная функция ОВ  'Тестирование блоков данных'
        :
        :SPB    =NIVO       Прерывание, если DB 50 не обнаружен
        :TAK
        :ENT
        :+F
        :                   АКК 1 := адрес последнего слова данных +1
        :                   АКК 2 := адрес первого слова данных
        :                   АКК 3 := константа
        :!=F                Прерывание, если DB 50 содержит ноль
        :SPB    =NIVO       слов данных
        :
SCHL    :ADD    BF-l        Все слова данных, начиная с последнего
        :TIR 10             слова данных заменяются на хранящуюся
        :<F                 в АКК 3-L константу
        :SPB    =SCHL
        :                   Продолжение программы . . .
WEIT    :                   ...после того как все слова данных будут переписаны
        :
        :BEA
NIVO    :                   …если DB50 отсутствует или содержит
        :                   ноль слов данных
```

:BE

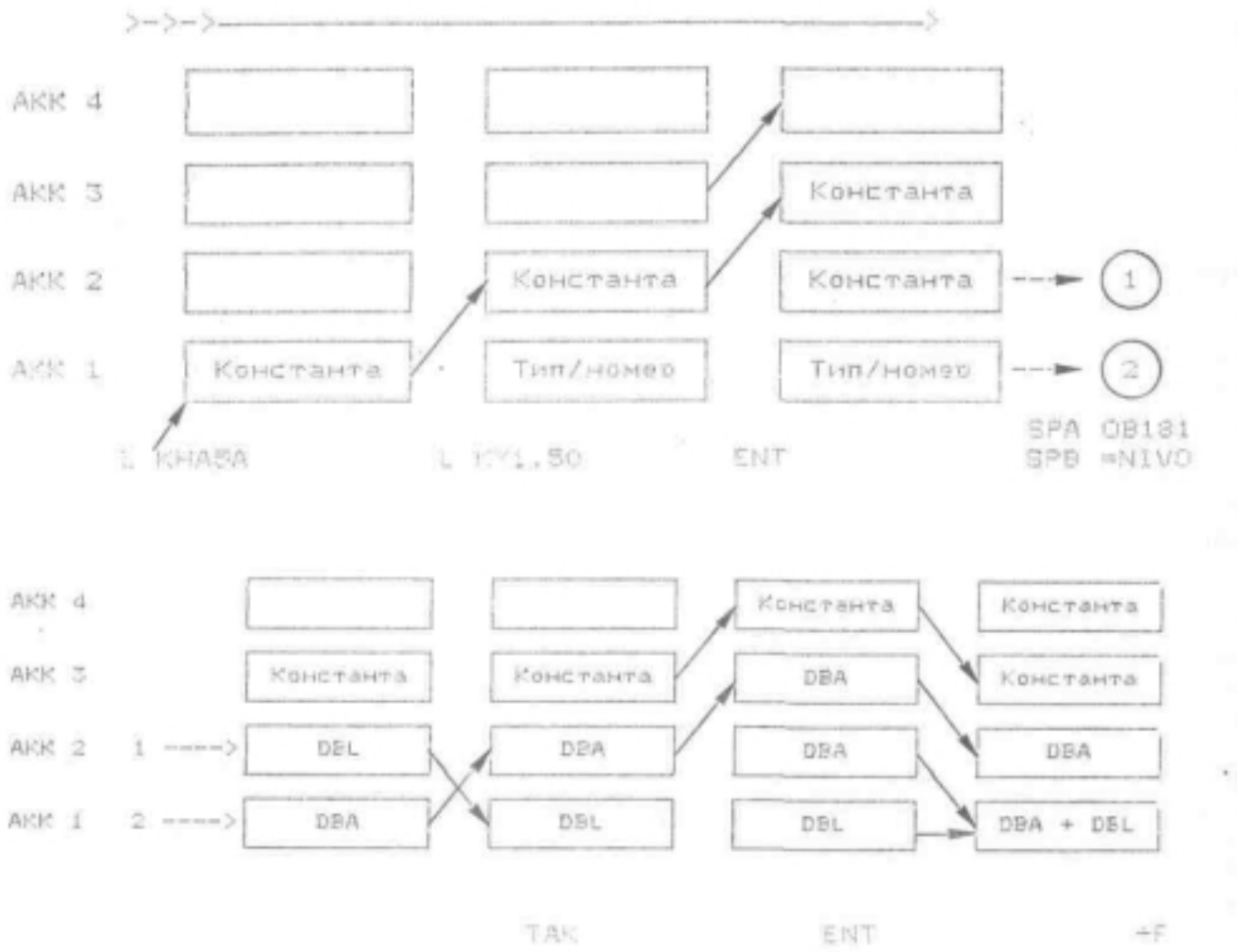


Рис. 9-2: <u>Заполнение аккумуляторов во время выполнения программы</u>

Указание:
Часть программы, начинающаяся с метки SCHL. может использовать для перезаписи любую область памяти (например: маркеры, счетчики, таймеры) с константами.

Пример 9-3: <u>Стирание всех маркерных байтов (MB0 по MB255)</u>

```
        :L      KB0         Константы, с помощью которых должны быть
        :                   переписаны все маркерные байты
        :L      BS14        Базовый адрес области маркеров =(адрес
        :                   первого маркерного байта MBO)
        :ENT
        :L      KF256       + длина области маркеров = (адрес послед-
        :ENT                него маркерного байта MB255) + 1
        :+F
        :
SCHL    :ADD    BF-1        Все 256 маркерных байтов, начиная с мар-
        :TIR    10          керного байта MB255, переписываются
        :<F                 на константы, хранящиеся в АКК 3—LL
```

```
:SPB        =SCHL
   .
   .        .
```

9.2. Передача блоков памяти

С помощью системных операций TNB и TNW Вы можете передавать все блоки памяти (максимум 255 байт с TNB. 255 слов с TNW).

Операция	Параметр	Описание
TNB	0 . . . 255	Передача блока данных (1 . . . 255 байт)
TNW	0 . . . 255	Передача блока данных (1 . . . 255 байт)

С помощью команд TNB и TNW Вы можете обращаться как к локальной области памяти, так и к побайтово организованной части глобальной области памяти (F0000 по F3FF, FC00 по FFFF).

TNB и TNW на области буфера связи

Команды TNB и TNW в многопроцессорном режиме AG S5 135U не могут быть использованы для обращения к области буферов связи (F4000 –FBFF). В этом случае пользуйтесь командами из главы 9.3.5 "Обращение к буферам связи" или спецфункциями из главы 6.6 " Доступ к буферам связи".

Параметр в TNW/TNB указывает на длину (количество слов/количество байт) передаваемой области. Конечный адрес исходной области должен быть предварительно загружен в АКК2, а коночный адрес области приема- в АКК1. Поэтому указываются влияние (наибольшие) адреса исходной и приемной областей. Передача сама выполняет в CPU 928 [1]декрементирование[1], т.е. она начинает свою пересылку с наивысшего адреса исходной области и завершает ее на наинизшем.

Исходная и приемная области должны полностью находиться в области памяти и не могут пересекаться. При этом различаются следующие области памяти с границами областей:

Адреса (шестнадцатеричные)

1. 0000 . . . 1FFF — Модуль пользователя (16 бит) 8К-слов
   0000 . . . 3FFF — Модуль пользователя (16 бит) 16К-слов
   0000 . . . 7FFF — Модуль пользователя (16 бит) 32К-слов
2. 8000 . . . DD7F — DS-RAM (16 бит)
3. DD80 . . . EDFF — Системная RAM (16 бит: DB0, BA/BS, таймеры,, счетчики и т.д.)
4. EE00 . . . EFFF — RAM (8 бит: маркеры, область отображения процесса)
5. F000 . . . FFFF — Периферия (8 бит)

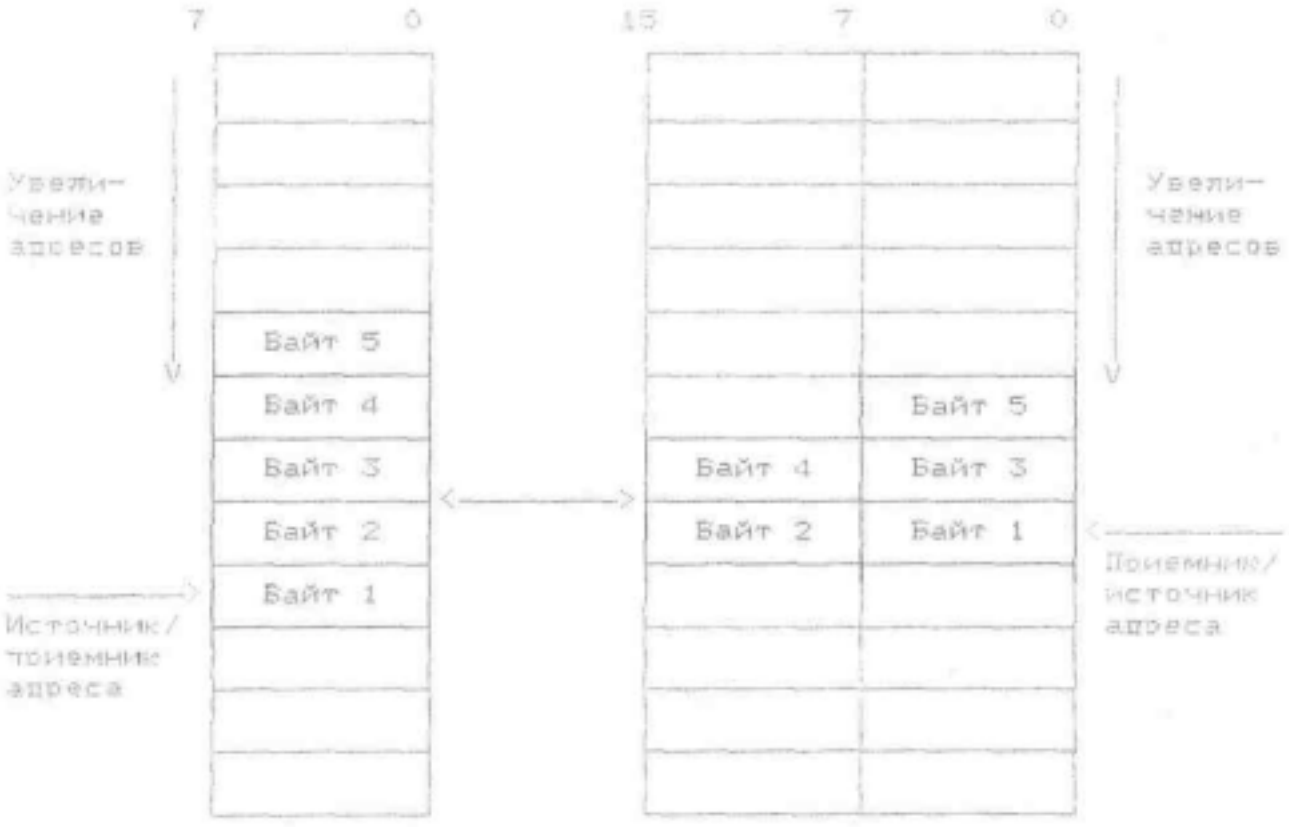
(см. Главу 8 " Распределение памяти")

Псевдограницы команд TNB и TNW

Команды TNB и TNW- это медленно выполняемые команды STEP5, они содержат так называемые 'псевдограницы команд': т.е. передача данных выполняется от источника к приемнику в виде частей блоков различной величины. Если теперь во время передачи части блока возникнет ошибка (например ZYK) или прерывание (например прерывание по времени или от процесса), то в конце этой части блока на псевдогранице команды будет вложен соответствующий организационный блок. Для вызова ОВ обработки прерывания от процесса или 0В обработки прерывания по времени необходимо предусмотреть, чтобы в DX О был установлен режим "Прерывание на границе команды".

Исключение: Если во время передачи возникает задержка квитирования и/или ошибка адресации, то сначала передается весь блок данных, а этом на границе команды однократно вызывается организационный блок, предусмотренный для этого (при QVZ и ADF одновременно только QVZ-OB). В качестве адреса ошибки QVZ всегда указывается младший адрес исходной области. Независимо от этого можно осуществлять вложение 0В 2, 0В 10 по ОВ 18 или организационного блока обработки ошибки на псевдограницах команд.

TNB и TNW между областями памяти 8 и 16. бит



Передача байтов с 1 по 5

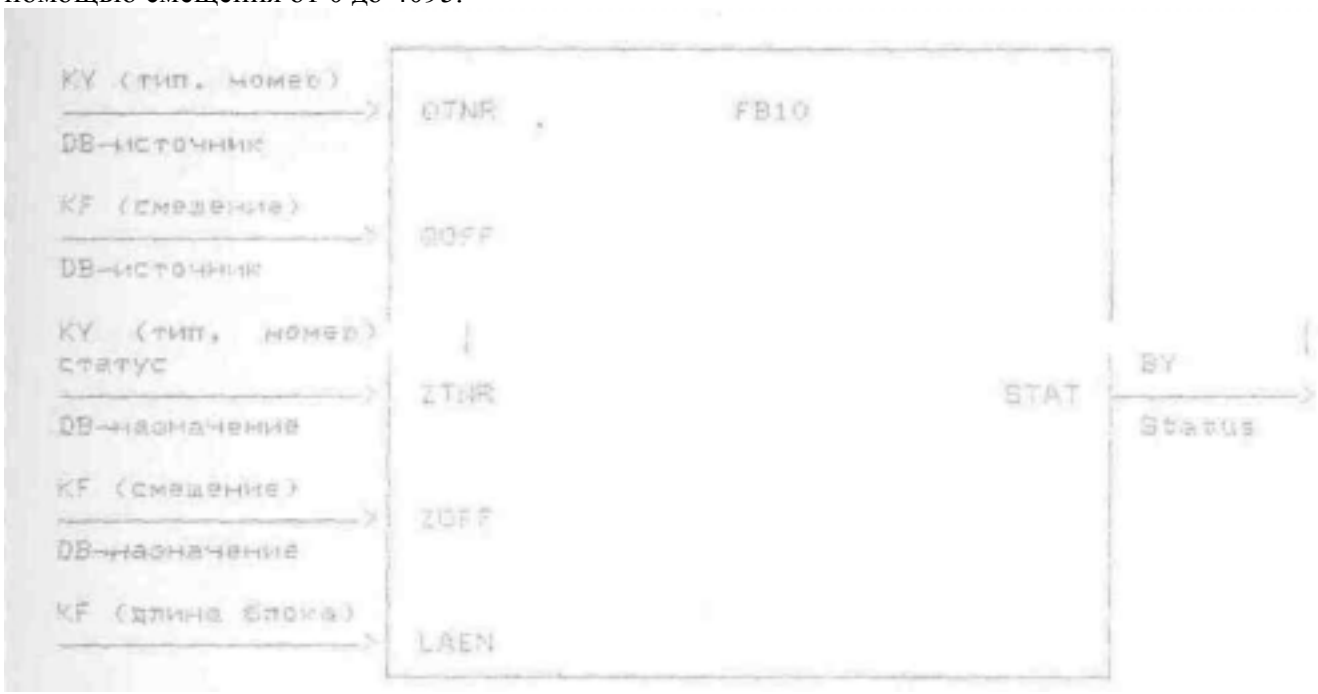
```
L     <исходный адрес>
L     <адрес приема>
TNB 5
```

Передача байтов с 1 по 4

```
L     <исходный адрес>
L     <адрес приема>
TNB 2
```

Пример:

Копируется блок максимум в 4095 слов данных из DB- или DX-блока данных в другой DB- или DX- блок данных. Начало блока устанавливается внутри блока источника и приемника с помощью смещения от 0 до 4095.



Перед копированием проверяются входные параметры. В случае ошибки в выходном параметре устанавливается STAT бит 27 = 1, и в битах с 20 по 22 выдается тип ошибки:



0 = нет ошибки
1 = ошибка

Вид ошибки:

1. = DB- источник = DB- назначение
2. = Смещение или длина > 4095
3. = DB—источник отсутствует и или недействителен
4. = DB—источник слишком короток
5. = DB—приемник отсутствует или недопустим
6. = DB—приемник в памяти только для чтения (EPROM—модуль)
7. = DB—приемник слишком короток

FB 10 делится на пять шагов программы со следующей конфигурацией;

1. Входные параметры

— Контролируется, не имеют ли блок-источик и блок-приемник один и тот же номер.
— Проверяется, являются ли параметры 'смещение источника', 'смешение приемника' и [1]длина блока' меньше 4096.

2. Блок данных- источник

- Проверяется, имеется ли исходный блок данных и достаточна ли его длина.
- Вычисляется абсолютный адрес последнего слова данных в исходном блоке.

3. Блок данных- приемник

- Проверяется, имеется ли блок—приемник достаточна ли его длина и лежит ли он в памяти записи—чтения *(*RAM-модуль).
- Зачисляется абсолютный адрес последнего слова данных в блоке—приемник

4. Передача

- Процесс копирования выполняется с помощью команды TNW

  Блок длиной более 255 слов передается частями по 128 слов (команда TNW 128).

  Можно имеющийся еще остаток передать с помощью дополнительной команды TNW

5. Биты индикации

- Выходной параметр ′Статус' обеспечивается результатом предпринятого контроля.

Заполнение ячеек памяти;

MW 242 Конечный адрес данных назначения

MW 244 Конечный адрес исходных данных

MW 246 Длина блока

MW 248 Смещение в блоке данных назначения

MW 250 Тип и номер блока данных назначения

MW 252 Смещение в исходном блоке данных

MW 254 Тип и номер исходного блока данных

BS 60 Счетчик частей блока

Программирование функционального блока FB 10

Указание: Необходимо, начиная со слова данных DW 0, произвести копирование, исключая выделенную курсивом часть программы. Значение смещения не вводится.

FB10

Сегмент 1

```
Имя   :DB-DB-TR          Блок  данных-блок данных-передача
BEZ   :QTNR      E/A/D/B/T/Z  D      KM/KH/KY/KC/KF/KT/KZ/KG:    KY
BEZ   :QOFF      E/A/D/B/T/Z  D      KM/KH/KY/KC/KF/KT/KZ/KC:    KF
BEZ   :ZTNR      E/A/D/B/T/Z  D      KM/KH/KY/KC/KF/KT/KZ/KG:    KY
BEZ   :ZOFF      F/A/D/B/T/Z  D      K11/KH/KY/KC/KF/KT/KZ/KG:   KF
BEZ   :LAEN      E/A/D/B/T/Z  D      KM/KH/KY/KC/KF/KT/KZ/KG:    KF
BEZ   :STAT      E/A/D/B/T/Z  A      BI/BY/W/D:   BY
                  :                  Начало входных параметров
                  :LW  = QTNRR       Тип (DB/DX) и номер
                  :T    MW  254      исходного блока данных
                  :LW  =ZTNR         Тип (DB/DX) и номер блока
                  :T  MW250          данных назначения
                  :! = F             Исходный DB = DB назначения ?
                  :SPB  =F001        Переход, если ДА
                  :
                  :LW  =QOFF         Смещение в исходном
                  :T  MW252          блоке данных
                  :LW  =ZOFF         Смещение в блоке данных
                  :T  MW248          назначение
                  :OW
                  :LW  =LAEN         Длина (количества слов данных)
                  :T  MW246          передаваемого блока
                  :OW                (длина блока)
                  :L  KHF000         Смещение источника, смещение приемника
                  :UW                или длина >= 4096 ?
                  :SPP  =F002        Переход, если Да
                  :                  Конец входных параметров
                  :
                  :                  Начало исходного блока данных
                  :L  MW254          тип и номер исходного блока данных
                  :SPA OB181         Тестирование блока данных
                  :SPB  =F003        Переход, если отрицателен тест блока
                  :TAK               A1: Число DW, A2 адрес
                  :ENT               A3: Адрес
                  :L  MW252          Смещение в исходном блоке данных
                  :ENT               A3: Число DW. A4: Адрес
                  :L  MW246          длина блока
                  :+F                смещение + длина блока
                  :<F                число DW < смещение + длина блока?
                  :SPB  =F004        Переход если Да
                  :L  KB1            A2: Смещение + Длина блока A3: Адрес
                  :-F                Смещение + Длина блока – 1
                  :+F                Смещение + Длина блока – 1 – адрес
                  :T  MW244          Конечный адрес исходных данных
                  :                  Конец исходного блока данных
                  :
                  :                  Начало блока данных назначения
                  :L  MW250          Тип и номер блока данных назначения
                  :SPA  OB181        Тестирование блока данных
                  :SPB  =F005        Переход если тест блока отрицателен
                  :SPM = F006        Переход если блок в EPROM
```

```
          :TAK          A1: число DW, A2: Адрес
          :ENT          A3: Адрес
          :L  MW248     Смещение в блоке данных назначения
          :ENT          A3: Число DW, A4: Адрес
          :L  MW246     Длина блока
          :+F           Смещение + длина блока
          :<F           Число DW< Смещение + Длина блока?
          :SPB  =F007   Переход если Да
          :L  KB1       A2: Смещение + длина блока,  A3: Адрес
          :-F           Смещение + Длина блока-1
          :+F           Смещение + Длина блока-1 +Адрес
          :T  MW242     Конечный адрес блока данных назначения
          :             Конец блока данных назначения
          :             Начало передачи
          :L  KB0       Значение сравнения
          :L  MW246     Длина блока, старший байт
          :!=F          Длина блока >= 256 слов ?
          :SLW  1       Умножение на 2, число частичных
          :T  BS60      блоков по 128 слов
          :L  MW244     Конечный адрес исходных данных
          :L  MW242     Конечный адрес данных назначения
          :SPB =REST    Переход, если длина блока < 256 слов
SCHL      :TNW  128     Передача одного частичного блока
          :ADD  KF-128  Уменьшение конечного адреса источника
          :TAK          на длину частичного блока
          :ADD  KF-128  Уменьшение конечного адреса приемника
          :TAK          на длину частичного блока
          :SPA  OB160   Цикл отсчета
          :SPB  = SCHL  переход, если не все частичные блоки
          :             переданы
REST      :B  MW246     длина блока . младший байт
          :TNW  0       Передача последнего блока
          :             Конец передачи
          :
          :             Начало индикации
          :L  KB0       Метка 00 (шестнадцатирич.): Нет ошибки
ENDE      :T  =STAT     Выходной параметр статус\ошибка
          :BEA
 F001     :L  KB129     Метка ошибки 81 (шестнадцатерич.):
          :SPA  =ENDE   Исходный DB =DB назначение
 F002     :L  KB130     Метка ошибки 82 (шестнадцатерич.):
          :SPA  =ENDE   Смещение или длина >= 4096
 F003     :L  KB131     Метка ошибки 83 (шестнадцатерич.):
          :SPA  =ENDE   Недопустимый исходный DB
 F004     :L  KB132     Метка ошибки 84 (шестнадцатерич.):
          :SPA  =ENDE   Исходный DB слишком мал
 F005     :L  KB133     Метка ошибки 85 (шестнадцатерич.):
          :SPA  =ENDE   DB назнач. не допустим
 F006     :L  KB134     Метка ошибки 86 (шестнадцатерич.):
          :SPA  =ENDE   DB назнач. В памяти только для чтения
 F007     :L  KB135     Метка ошибки 87 (шестнадцатерич.):
          :SPA  =ENDE   DB назначения слишком мал
          :             Конец индикации
          :BE
```

9.3. Операции над регистром BR

Регистр BR (регистр базового адреса, 32 бита) используются описанными ниже командами загрузки/передачи (глава 9.3.3) для адресации памяти. Обращение происходит к ячейке памяти, абсолютный адрес которой вычисляется как сумма содержимого регистра BR и константы:

Абсолютный адрес = Содержимое регистра BR + константа

Регистр BR на меняется, если

— по команде перевода (SPA/SPB) обработка программы продолжается в другом блоке

— включается другой уровень обработки программы

Регистр BR перед вызовом уровня обработки программ сбрасывается в 'O'.

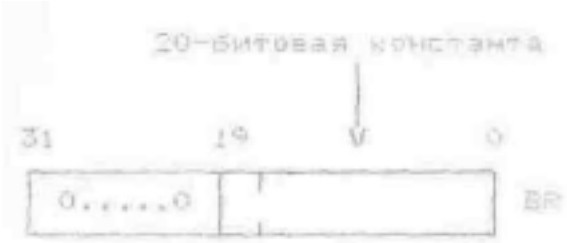
9.3.1. Загрузка BR- регистров

С помощью следующих команд можно перезагружать или содержимое регистра BR:

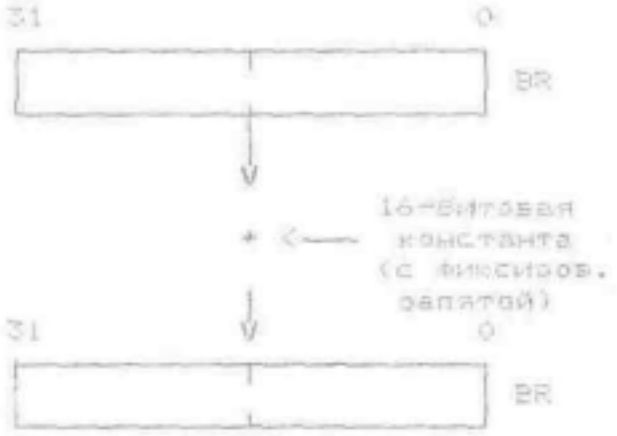
Операция	Параметр	Описание
MBR	0…FFFFF	Загрузка в регистр BR 20-битовой константы 1)
ABR	-32768… …+32767	Прибавление 16- битовой константы к содержимому регистра BR

1) Биты 2^{20} по 2^{31} регистра BR сбрасывается в 'O'.

MBR 0 … FFFFF



ABR -32767 … +32767



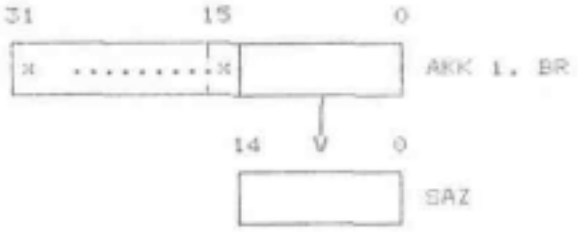
9.3.2. Сдвиг содержимого регистров

Для того чтобы можно было более гибко использовать регистры введены новые команды, с помощью которых можно изменять содержимое отдельных регистров или передавать его в другие регистры.

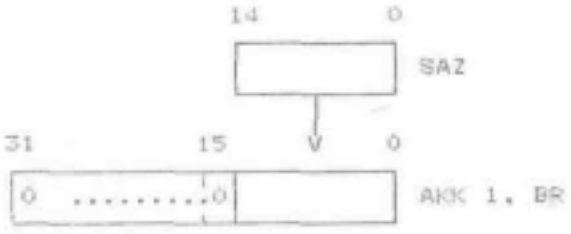
Операция	Параметр	Описание
MAS	-	Передача содержимого аккумулятора 1 (биты с 2^0 по 2^{14}) в регистр SAZ (счетчик адресов команд)
MBS	-	Передача содержимого регистра BR (биты с 2^0 по 2^{14} регистр базового адреса) в регистр SAZ (счетчик адресов команд)
MSA	-	Передача содержимого регистра SAZ (счетчик адресов команд) в аккумулятор 1 1)
MSB	-	Передача содержимого регистра SAZ (счетчик адресов команд) в регистр BR (регистр базового адреса) 1)
MBA	-	Передача содержимого аккумулятора 1 (биты с 2^0 по 2^{31}) в регистр BR (регистр базового адреса)
		Передача содержимого регистра (регистр базового адреса) в аккумулятор 1

1) Биты с 2^{15} по $2^{3:1}$ сбрасываются в '0'.

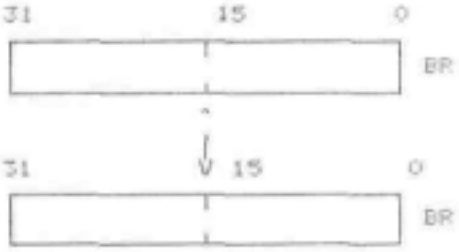
MAS, MBS



MSA. MSB



MAB, MBA



9.3.3. Обращение к локальной памяти

Следующие команды позволяют обращаться к локальной, пословно организованной области памяти по абсолютным адресам. Абсолютный адрес является суммой содержимого регистра BR н 16-битавой константы, содержащейся в команде (-32768 по -32767).

Операция	Параметр	Описание
LRW	-32768 … … +32767	Загрузка слова, адресуемого через регистр BR+ константа в аккумулятор 1-L.
LRD	-32768 … … +32767	Загрузка двойного слова, адресуемого через регистр BR+ константа в аккумулятор 1
TRW	-32768 … … +32767	Передача содержимого аккумулятора 1-L в слово, адресуемое через регистр BR+ константа
TRD	-32768 … … +32767	Передача содержимого аккумулятора 1 в двойное слово, адресуемое через регистр BR+ константа

Абсолютный адрес должен лежать в диапазоне между '0' н EDF'H (при LRW. TRW) или О и EDFEH (при LRD. TRD). Если это не так,, то профессор опознает ошибку выполнения и вызывает 0В 31. В аккумулятор 1 загружается метка ошибки, которая поясняет возникшую ошибку (см. главу 3.6.2 "Особые ошибки выполнения").

9.3.4. Обращение к глобальной памяти

Следующие команды позволяют обращаться к глобальной побайтово или пословно организованной области памяти по Абсолютным адресам. Абсолютный адрес является суммой содержимого регистра BR и 16-битовой константы, содержащейся в команде (-3276S по +32767).

Тестирование и установка контрольной ячейки в глобальной памяти

Обращением отдельного процессора к сообща используемой области памяти можно управлять с помощью контрольной ячейки. Каждой сообща используемой области памяти ставится в соответствие контрольная ячейка, которая должна тестироваться всеми процессорами участвующими в работе перед каждым обращением. Контрольная ячейка содержит либо значение '0' либо метку установочного места процессора, который использует в данный момент эту область памяти и после завершения работы освободит ее, записав в контрольную ячейку значение '0'.

Команда TSG поддерживает тестирование и установку капитальной ячейки

Операция	Параметр	Описание
TSG	-32768 … … +32767	Тестирование и установка контрольной ячейки адресуемой через регистр BR+ константа

Абсолютный адрес должен располагаться в диапазоне между 0 и EFFFH. Если это не так, то процессор опознает ошибку выполнения и вызывает 0B 31. В аккумулятор 1 загружается метка ошибки, которая поясняет возникшую ошибку (см. главу 5.6.2 «Особые ошибки выполнения»)

В качестве контрольной ячейки используется младший байт слова адресуемого через регистр BR+ константа. Если содержимое младшего байта равно "0" то команда TSG заносит в контрольную ячейку метку установочного места (из BS 29)

Тестирование (=чтение)' и возможная установка (= запись) представляют собой фрагменты программы, работу которых прерывать нельзя.

Результат тестирования можно определить по битам индикации ANZ0 и ANZ1:

ANZ 0	ANZ 1	Значение
0	0	Содержимое контрольной ячейки равно 'O'; процессор заносит метку своего установленного места.
0	1	Собственная метка установочного места уже занесена в контрольную ячейку.
1	0	Ячейка содержит чужую метку установочного места.

Важно!

Команда TSG должна использоваться всеми процессорами, которые должны синхронизировать доступ к общей глобальной области памяти.

Обратите внимание на пояснения к командам SES и SEF (Установка/ сброс семафора, глава 3.2.2) и к организационному блоку спецфункции OB 2I8 (Захват буфера связи, глава 6.6.3).

Операции загрузки и передачи для побайтово организованной глобальной памяти

Операция	Параметр	Описание
LB GB	-32768 … … +32767	Загрузка байта, адресуемого через регистр BR+ константа в аккумулятор 1-LL.
LB GW	-32768 … … +32767	Загрузка слова, адресуемого через регистр BR+ константа в аккумулятор 1-L
LB GD	-32768 … … +32767	Загрузка двойного слова, адресуемого через регистр BR+ константа в аккумулятор 1
TB GB	-32768 … … +32767	Передача содержимого аккумулятора 1-LL в байт, адресуемое через регистр BR+ константа
TB GW	-32768 … … +32767	Передача содержимого аккумулятора 1-L в слово, адресуемое через регистр BR+ константа
TB GD	-32768 … … +32767	Передача содержимого аккумулятора 1 в двойное слово, адресуемое через регистр BR+ константа

Абсолютный адрес должен лежать в диапазоне

- между 0 и EFFFH (при LB GB, TB GB),
- между 0 и EFFEH (при LB GW, TB GW),
- между 0 и EFFCH (при LB GD, TB GD).

Если это не так, то процессор опознает ошибку выполнения и вызывает OB 31. В аккумулятор 1 загружается матка ошибки, которая поясняет возникшую ошибку (см. главу 5.6.2 «Особые ошибки выполнения»)

Операции загрузки и передачи для пословно организованной глобальной памяти

Операция	Параметр	Описание
LW GW	-32768 … … +32767	Загрузка слова, адресуемого через регистр BR+ константа в аккумулятор 1-L.
LW GD	-32768 … … +32767	Загрузка двойного слова, адресуемого через регистр BR+ константа в аккумулятор 1
TW GW	-32768 … … +32767	Передача содержимого аккумулятора 1-L в слово, адресуемое через регистр BR+ константа
TW GD	-32768 …	Передача содержимого аккумулятора 1 в двойное слово, адресуемое через регистр BR+ константа

Абсолютный адрес должен лежать в диапазоне между 0 и EFFFH (при LW GW, TW GW), или между 0 и EFFEH (при LW GD, TW GD). Если это не так, то процессор опознает ошибку выполнения и вызывает 0B 31. В аккумулятор загружается метка ошибки, которая поясняет возникшую ошибку (см. главу 5.6.2 "Особые ошибки выполнения".

9.3.5. Обращение к буферам связи

Глобальная область содержит между адресами F400H до FBFFH 'окно' для высвечивания одной из максимум 256 областей памяти (= буферов связи). Буфер связи занимает максимум 2К адресов и может быть организован побайтово или пословно. Перед каждым обращением к области буферов связи не об код им о выбрать один из 256 буферов связи путем занесения номера буфера связи в регистр выбора. Занесение номера в регистр выбора и последующее обращение к области буферов связи не должны прерываться.

С помощью следующих команд Вы можете обращаться к буферам связи организованным побайтово или пословно по абсолютным адресам в памяти. Абсолютный адрес складывается из содержимого регистра BR. и константой содержащейся в команда (-32768 до +32767).

перед каждым обращением (загрузкой/передачей) к области буферов связи не обходимо открыть один из 256 буферов связи. Для этого загрузите номер открываемого буфера связи в аккумулятор 1—L; этот номер вводится с помощью команды ACR в регистре буферов связи. Все последующие операции с буферами связи перед обращением к ним записывают содержимое регистра буферов связи в регистр выбора.

Регистр связи не меняется, если

— по команде перехода (SPA/SPB) обработка программы продолжается в другом блоке

— включается другой уровень обработки программы.

Регистр связи перед вызовам уровня обработки программ сбрасывается в «0»

Открытие буфера связи

Операция	Параметр	Описание
ACR		Открытие того буфера, чей номер находится в аккумуляторе 1-L

допустимые значения: 0 до 255

Номер буфера связи должен лежать в диапазоне между О и 255. Если это не так, то процессор опознает ошибку выполнения и вызывает 0В 31. В аккумулятор 1 загружается метка ошибки, которая поясняет возникшую ошибку (см. главу 5.6.2 "Особые ошибки выполнения").

Тестирование и установка контрольной ячейки в области буферов связи

Обращением отдельного процессора к сообща используемой области памяти можно управлять с помощью контрольной ячейки. Каждой сообща используемой области памяти ставится в соответствие контрольная ячейка, которая должна тестироваться всеми процессорами, участвующими в работе, перед каждым обращением, контрольная ячейка содержит либо значение “0' либо метку установочного места процессора, который использует в данный момент эту область памяти и после завершения работы освободит ее, записав в контрольную ячейку значение "0”.

Команда TSC поддерживает тестирование и установку контрольной ячейки.

Операция	Параметр	Описание
TSC	-32768… …+32767	Тестирование и установка контрольной ячейки, адресуемой через регистр BR + константа

Абсолютный адрес должен располагаться в диапазоне между F400H и FBFFH. Если это не так, то процессор опознает ошибку выполнения и вызывает 0B 31. В аккумулятор 1 загружается метка ошибки, которая поясняет возникшую ошибку (см. главу 5.6.2 "Особые ошибки выполнения").

В качестве контрольной ячейки используется младший байт слова адресуемого через регистр BR + константа. Если содержимое младшего байта равно "0", то команда TSC заносит в контрольную ячейку метку установочного места (из FS 29).

Тестирование (= чтение) и возможная установка (= запись) представляют собой фрагменты программы, работу которых прерывать нельзя.

Результат тестирования можно определить по битам индикации ANZ0 и ANZ1:

ANZ 0	ANZ 1	Значение
0	0	Содержимое контрольной ячейки равно “0”; процессор наносит метку своего установочного места.
0	1	Собственная метка установочного места уже занесена в контрольную ячейку.
1	0	Ячейка содержит чужую метку установочного места

Важно!

Команда TSC должна использоваться всеми процессорами, которые должны синхронизировать доступ к общей глобальной области памяти.

Обратите внимание на пояснения к командам SES и SEF (Установка/ сброс семафора, глава 3.2.2) и к организационному блоку спецфункции 0B 218 ("Захват буфера связи, глава 6.6.3).

Операции загрузки и передачи для побайтово организованных буферов связи

Операция	Параметр	Описание
LB CB	-32768… …+32767	Загрузка байта, адресуемого через регистр BR + константа в аккумулятор 1-LL
LB CW	-32768… …+32767	Загрузка слова, адресуемого через регистр BR + константа, в аккумулятор 1-L
LB CD	-32768… ...+32767	Загрузка двойного слова, адресуемое через регистр BR + константа, в аккумулятор 1
TB CB	-32768… ...+32767	Передача содержимого аккумулятора 1-LL в байт, адресуемый через регистр BR + константа
TB CW	-32768… ...+32767	Передача содержимого аккумулятора 1-L в слово, адресуемое через регистр BR + константа
TB CD	-32768… ...+32767	Передача содержимого аккумулятора 1в двойное слово, адресуемое через регистр BR + константа

Абсолютный адрес должен лежать в диапазоне

- между F400H и FBFFH (при LB CB, TB CB),
- между F400H и FBFEH (при LB CW, TB CW)
- между F400H и FBFCH (при LB CD, TB CD),

Если это не так, то процессор опознает ошибку выполнения и вызывает 0B 31. В аккумулятор 1 загружается метка ошибки, которая поясняет возникшую ошибку (см. главу 5.6.2 «Особые ошибки выполнения»)

Операции загрузки и передачи для условно организованных буферов связи

Операция	Параметр	Описание
LW CW	-32768… …+32767	Загрузка слова, адресуемого через регистр BR + константа, в аккумулятор 1-L
LW CD	-32768… …+32767	Загрузка двойного слова, адресуемого через регистр BR + константа в аккумулятор
TW CW	-32768... …+32767	Передача содержимого аккумулятора 1-L в слово, адресуемое через регистр BR + константа
TW CD	-32767... ...+32767	Передача содержимого аккумулятора 1 в двойное слово, адресуемое через регистр BR + константа.

Абсолютный адрес должен лежать в диапазоне между F400H и EFFFH (при LW CW, TW CW'), или между F400H и FBFEH (при LW CD, TW CD), если это не так, то процессор опознает ошибку выполнения и вызывает 0B 31. В аккумулятор 1 загружается метка ошибки, которая поясняет возникшую ошибку (см. главу 5.6.2 "0собые ошибки выполнения").

10. Многопроцессорный режим работы

10.1. Общие замечания

Программируемый контроллер 135U больше всего подходит для комплексной обработки данных, так как может включать до четырех одновременно работающих процессоров (CPU). Каждый процессор при этом обрабатывает свою программу независимо от других процессоров. В многопроцессорной системе можно использовать в различных комбинациях следующие процессоры:

- S- процессор, применяется для управления (обработка двоичных сигналов), для коммуникации, контроля и сигнализации: занимает одно место.
- M- процессор, применяется для программирования на языках высокого уровня; одно место.
- CPU928, универсального применения. целесообразно использовать для управления, расчетов, регулирования (обычное и цифровое); занимает два места.

Применение многопроцессорного режима работы – когда и как ?

- Если пользовательской программе не хватает места в памяти или процессоре, то ее можно разбить на части, выполняемые на нескольких процессорах,
- Если какая-либо часть установки должна работать особенно быстро, то можно соответствующую программу выделить из общего комплекса программ и обрабатывать ее на отдельном, « быстром « , процессоре.
- Если установка состоит из нескольких частей, которые четко разграничены и, следовательно, относительно независимо управляются или регулируются, то можно для первой части использовать процессор 1, для второй – процессор 2 и т. д.

По вопросам многопроцессорного режима смотрите описание «Многопроцессорный режим в AG 135U (C79000-B8500-C500-01). Оно находится в регистре 7 данного руководства и поясняет по шагам процесс эксплуатации вашего многопроцессорного устройства, дает указание по обслуживанию, описывает типичные картины ошибок и указывает на возможные их причины.

Важно!
Если вы используете многопроцессорный режим, то обязательно должен быть установлен координатор (KOR) независимо от того, сколько вы используете процессоров 1 или несколько!

Координатор предназначен для координации обмена данными между отдельными процессорами. При этом он должен знать сколько мест обслуживать, т. е. между несколькими процессорами он должен делить время использования шины S5.

Важно!
Поэтому вы должны установить на координаторе число занятых установочных мест (см. руководство по эксплуатации координатора)! При этом обратите внимание используется ли процессор нормальной (напр. R-процессор) или двойной (напр. CPU 928) ширины.

Если, например, вы установите на координаторе число 3, он будет делить шину между тремя процессорами, т. е. обслуживать первые три установочных места справа от координатора (№ 17, 19 и 27, см. руководство CPU928). Четвертое место (№ 35) в этом случае координатором не обслуживается. Установленный туда процессор к шине не подключается.

Важно!
Координатор и процессоры должны устанавливаться в контроллер без пропусков установочных мест.

Когда вы работаете в многопроцессорном режиме, необходимо кроме того, запрограммировать для каждого процессора блок данных DB1. Он содержит список цифровых входов и выходов, а также входных и выходных маркеров связи, которые соответствуют каждому процессору (см. раздел 10.3.1)!

10.2.Обмен данными между процессорами

Для циклического обмена данными между процессорами или между процессором и коммуникационными процессорами служат «маркеры межпроцессорной связи» (см. раздел 10.2.1).

При обмене большим количеством данных (напр., при передаче целого блока данных) между R-, M- процессорами и CPU928 используйте «Специальные функции для межпроцессорной связи» OB200…OB205 (см. разд. 10.2.2).

Для связи с группой «интеллигентных» периферийных модулей (IP) и с коммуникационными процессорами (CP) используются управляющие блоки (см. разд. 6.9).

Если вы хотите передавать большие блоки данных и обеспечить при этом правильную работу другого процессора, то нужно пользоваться «семафорами» (разд. 3.2.2).

10.2.1. Маркеры связи

Для циклического обмена двоичными данными служат «маркеры связи». В первую очередь они обеспечивают по байтовую передачу информации.

Это может быть передача:

процессор (ы) < --------> процессор (ы)
процессор (ы) < --------> коммуникационный процессор (ы)

Операционная система передает маркеры связи один раз в цикл. При обмене данными между процессорами маркер связи физически запоминается в координаторе.

Маркеры связи представляют собой байты маркеров. они описываются в каждом процессоре как входные или выходные маркеры межпроцессорной связи.
Если вы опишите, напр., маркерный байт 50 в процессоре 1 как выходной байт маркеров связи, то его состояние будет циклически передаваться от координатора к процессору, в котором маркерный байт 50 описан как входной байт маркеров связи.

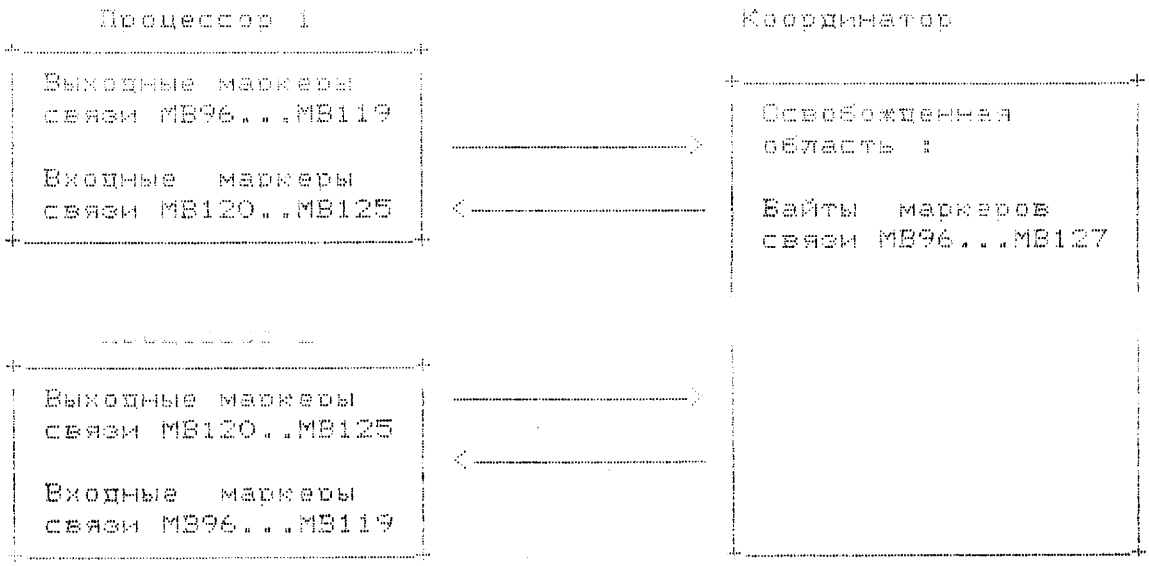
Область памяти для маркера связи в координаторе и коммуникационном процессоре занимает адреса F200H…F2FFH. Для каждого процессора можно иметь до 256 маркеров связи.

Обмен данными между процессорами

Обмен данными между отдельными процессорами (CPU928, R-, S- и M-процессоры) осуществляется через координатор. Процессор читает из координатора в DB1 маркерные байты, описанные как выходные маркеры связи, или пишет в координатор свои маркеры связи, описанные в качестве выходных.

В координаторе резервируется определенное количество маркеров связи: максимум 256 маркеров, объединенных в блоки по 32 байта (8 блоков). Перед работой внимательно изучите Руководство по координатору.

Пример:



Внимание !

— В качестве маркеров связи должны указываться только те маркерные байты, которые в координаторе свободны.

— Если в одном процессоре маркерные байты описаны как входные, то в другом процессоре они должны быть описаны как выходные. И: любой маркерный байт может быть выходным только для одного процессора, однако в качестве входного можно использовать его в трех других процессорах.

— Маркерный байт, не описанный как маркер межпроцессорной связи, может быть использован в качестве "нормального" байта маркеров.

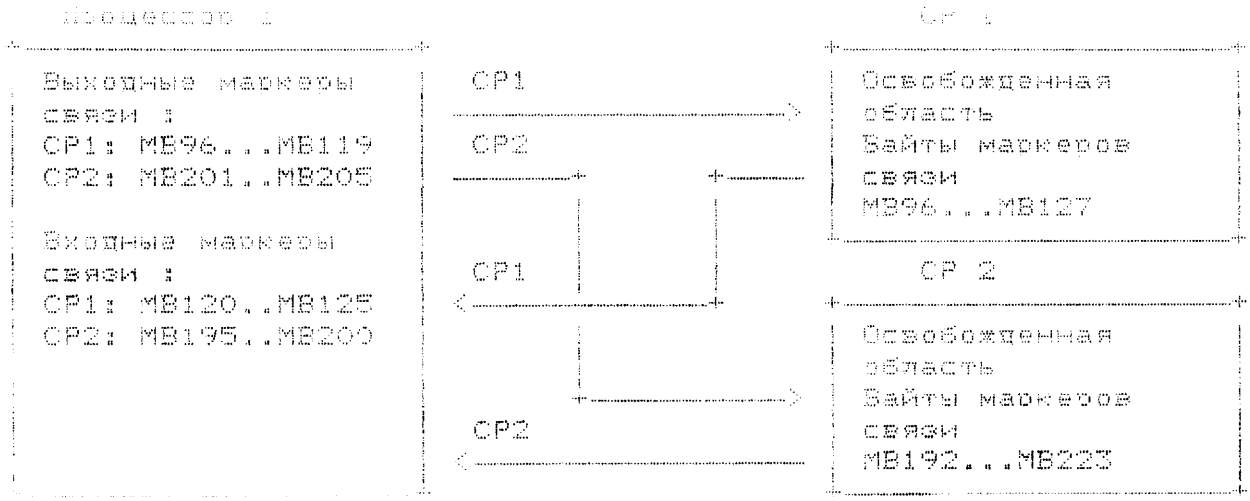
— Задавайте только нужное Вам количество маркеров связи; чем меньше число маркеров связи, тем меньше время передачи данных.

Обмен данными между центральными и коммуникационными процессорами

Если необходимо передать данные от процессора к коммуникационному процессору, то надо освободить необходимое количество маркеров связи на коммуникационном процессоре (CP). Там имеется 256 байтов, сгруппированных в области по 32 байта.

Если необходимо передать данные от процессора к нескольким коммуникационным процессорам, то надо, чтобы освобождаемые области на коммуникационных процессорах не пересекались, чтобы адреса не публиковались.

Пример:



Если Вы хотите одновременно использовать маркеры связи на координаторе и одном или нескольких коммуникационных процессорах, то должны избегать двойной адресации: Распределите маркеры связи на KOR и CP на области по 32 байта в каждой: те маркерные байты связи, которые используются в коммуникационном процессоре, выключаются на координаторе удалением перемычек (смотри Руководство по эксплуатации координатора).

Существенно, что байты маркеров могут быть определены в качестве выходных маркеров связи только на одном процессоре. Однако, как входной маркер связи, байт маркеров можно пометить на нескольких процессорах.

Передача маркеров связи в многопроцессорной системе

Маркеры межпроцессорной связи, находящиеся в DB1, передаются только тогда, когда процессор получит от координатора сигнал, что он может обращаться к шине.

Если нужно организовать одновременную работу нескольких процессоров, то координатор по очереди посылает каждому процессору сигнал о том, что шина свободна. При этом процессор может передавать только один байт. Из-за такой квантованной передачи может получиться так, что взаимозависимая информация при передачи может быть разорвана и в результате будет обработана неверно.

Если требуется передавать информацию объемом более одного байта, надо использовать OB224. Используя OB224, вы добьетесь того, что все маркеры связи, находящиеся в DB1, передаются как единый блок. Пока процессор занят этой передачей, он не может быть прерван другим. Т. к. другой процессор вынужден ожидать освобождения шины, то время выполнения программы в нем увеличивается. Обратите внимание на то, что использование OB224 удлиняет время цикла обработки программы! (см. разд. 6.8.6).

10.2.2. Межпроцессорные связи

Организационные блоки спецфункций OB200…OB205 позволяют производить в многопроцессорных системах обмен большими массивами данных между R-, M-процессорами и CPU928, причем данные временно записываются в координаторе KOR C.

OB200 захватывает в координаторе KOR C память, в которой временно запоминается блок данных.

OB202 передает блок данных в память KOR C и сообщает о том, сколько блоков данных еще можно записать в KOR C.

OB203 сообщает о числе свободных блоков в памяти KOR C.

OB204 выбирает блок данных из памяти координатора и сообщает о том, сколько блоков данных еще имеется в координаторе.

OB205 сообщает о числе блоков данных в памяти координатора.

Подробное описание этих спец. функций приведено в разделе 8 «SIMATIC S5 программируемый контроллер 135U, многопроцессорная коммуникация» данного руководства.

10.2.3.Передача взаимозависимых блоков данных

Если необходимо передать большой блок данных и обеспечить при этом корректную работу другого процессора, то используйте «семафор» (см. разд. 3.2.2).

10.3. Разделение доступа к шине периферийных устройств

Область для периферийных устройств в каждом процессоре занимает адреса F000H…FFFFH. В этой области адресуется группа периферийных устройств. Эта область может быть доступна для чтения и записи всем процессорам. Координатор имеет данные, которые позволяют ему координировать доступ отдельных процессоров к периферийным устройствам.

10.3.1. Блок данных DB1

В многопроцессорной системе для каждого процессора нужно запрограммировать блок данных DB1. При этом надо задать столько входов и выходов, сколько входных и выходных маркеров связи будет у процессора во время работы.

Важно!
При обновлении области отображения процесса принимаются во внимание только входы и выходы, определённые в DB1.

Ввод/корректировка данных в DB1.

1. С помощью программатора в режиме ON-LINE (программатор подключен к контроллеру), если контроллер находится в состоянии останова и в нём имеется память пользователя RAM.
2. С помощью ввода программы в модуль памяти ERROM пользователя.

Внимание!
Введённые в DB1 данные передаются процессоры только при запуске в режиме «Новый пуск»!

Так программируется маска DB1 с помощью функциональных клавиш:

DB1:

ЦИФРОВЫЕ ВЫХОДЫ	, 0,	1,	2,	3,	7,	10,	,	,	,	,
ЦИФРОВЫЕ ВЫХОДЫ	, 0,	2,	4,	12,	,	,	,	,	,	,
ВЫХОДНЫЕ МАРКЕРЫ СВЯЗИ	, 50,	51,	60,	,	,	,	,	,	,	,
ВЫХОДНЫЕ МАРКЕРЫ СВЯЗИ	, 70,	72,	100,	,	,	,	,	,	,	,
ДЛИНА ТАЙМЕРНОГО НАБОРА	, 128,	,	,	,	,	,	,	,	,	,

(Помните, что если маска DB1 заполнена с помощью S5 DOS, то при определённых обстоятельствах Вы не сможете считать её без ошибок с помощью пакета программ Pg “Studos”)

DB1 вручную записывается так:

- В словах данных 0, 1 и 2 должна находиться метка заголовка DB1. Поэтому Вы должны жёстко указать

  DW0 : KH = 4D41
  DW1 : KH = 534B
  DW2 : KH = 3031
- Начиная со слова 3 задаются области операндов.

  Для каждой области операндов задаётся определённая метка:
  Возможные метки:

Метка для цифровых входов	:	KH = DE00
Метка для цифровых выходов	:	KH = DA00
Метка для входных маркеров связи	:	KH = CE00
Метка для выходных маркеров связи	:	KH = CA00
Метка для длины таймерного набора	:	KH = BB00

После метки перечисляются номера используемых входов и выходов и требуемая длина таймерного блока в формате с фиксированной запятой.

- В качестве признака окончания блока DB1 используется KH = EEEE.

Важно!

- **Последовательность данных может быть любой. Следите за тем, чтобы входы и выходы сигналов от процессоров точно соответствовали описаниям в DB1.**
- **Возможно многократное использование одних и тех же байтов, например, для тестирования. В этом случае следите за тем, чтобы процессы многократно активировали заданные в DB1 байты.**
- **После последнего слова с информацией в DB1 должен находиться признак конца блока KH = EEEE.**

Пример программирования DB1 приведён на следующей странице.

Пример DB1

0:	KH = 4D41	DW 0-2:
1:	KH = 534B	Начальная метка
2:	KH = 3031	для DB1
3:	KH = DE00	Метка цифровых входов
4:	KH = +00000	Входной байт 0
5:	KH = +00001	Входной байт 1
6:	KH= +00002	Входной байт 2
7:	KH = +00003	Входной байт 3
8:	KH = +00007	.
9:	KH = +00010	.
10:	KH = DA00	Метка цифровых выходов
11:	KH = +00000	Выходной байт 0
12:	KH = +00002	Выходной байт 2
13:	KH = +00004	.
14:	KH = +00012	.
15:	KH = CE00	Метка входных маркеров связи
16:	KH = +00050	Маркерный байт 50
17:	KH = +00051	.
18:	KH = +00060	.
19:	KH = CA00	Метка выходных маркеров связи
20:	KH = +00070	Маркерный байт 70
21:	KH = +00072	.
22:	KH = +00100	.
23:	KH =BB00	Метка блока таймеров *
24:	KH = +00127	Таймер 0….Таймер 127
25:	KH = EEEE	Метка окончания блока

* Введя длину блока таймеров в DB1, Вы можете указать сколько таймеров должно быть обновлено циклически операционной системой. Эта функция системы должна быть запараметрирована в DX 0, смотри раздел «Дополнительный блок данных DX 0».

При новом пуске DB1 передаётся операционной системе. Она проверяет, есть ли подтверждение от соответствующих групп входов и выходов, а также маркеров связи. Если нет, то процессор переходит в состояние останова и на передней панели начинает медленно мигать красный светодиод STOP. Программа пользователя не запускается.

Когда Вы программируете DB1 для того, чтобы передать его при «новом пуске» процессору, то придерживайтесь следующих правил:

- доступ к группе периферийных устройств через область отображения процесса возможен только для заданных в DB1 входов и выходов (команды L/T EB, EW ED, AB, AW, AD и операции со входами и выходами).
- Непосредственная загрузка периферийного байта, минуя область отображения процесса, возможна с помощью команд L PB/PY, PW, QB, QW для всех квитируемых входов, независимо от распределения в DB1
- Непосредственная передача (T PB/PY, PW, QB, QW) возможна только для входов, заданных в DB1, поэтому здесь существенно описание области отображения процесса. Область отображения процесса имеется только для P-периферии с относительными адресами 0…127.

10.4. Запуск в многопроцессорном режиме

Координатор в многопроцессорном режиме запускается следующим образом:

- Селекторы режимов работ всех процессоров контроллера - положении “RUN”, селектор режимов работ координатора – в положении “STOP”.
- Переведите селектор режимов работ из положения “STOP” в положение “RUN”

(Запуск контроллера в многопроцессорном режиме возможно только через запуск координатора, если он находился перед этим в состоянии “СТОП”.

или:

- Селекторы режимов работ всех процессоров и координатора – в положении “RUN”.
- Используя функцию программатора “AG-START”, запустите процессор, который находился в состоянии “СТОП”, в нужном режиме запуска.

Режим запуска отдельного процессора зависит от того, обслуживается ли он в состоянии “СТОП” общим таймером. Может статься, что одни процессоры запускаются в режиме ручного перезапуска, а другие – в режиме нового пуска.

Если у процессоров нет общего таймера, то пуск производится в режиме ручного перезапуска (CPU 928 и R-процессор) или ручного нового пуска со стиранием (S-процессор).

Важно!
При различных видах запуска, если перед этим AG работал, возможна ложная передача сигналов через маркеры связи от одного процессора к другому. Чтобы избежать этого, необходимо запрограммировать соответствующие ОВ функции запуска ОВ 20, 21 и 22.

После сбоев питания координатор запускается автоматически. Все S-процессоры запускаются автоматическим новым пуском ч запоминанием, все CPU 928 и R-процессоры – автоматическим перезапуском или автоматическим новым пуском, в зависимости от того, как запрограммирован DX 0.

Запуск отдельных процессоров в многопроцессорном режиме синхронизирован, т.е. каждый процессор ждёт, пока не запустятся все другие, после чего все они одновременно переходят в циклическую работу. Для CPU 928 и R-процессора Вы можете выбрать синхронный запуск путём соответствующего программирования DX 0.

10.5. Тестовый режим

Запуск тестового режима производится следующим образом:

- Функция «Тестовый режим» в координаторе должна быть разблокирована.
- Переведите селектор режимов работ координатора из положения “STOP” в положение “TEST”. Должен засветиться светодиод BASP.
- Выберите вид запуска в том процессоре, который собираетесь переводить в режим “RUN”.

Особенности тестового режима

В тестовом режиме можно использовать как один процессор, так и любую их комбинацию. Процессоры, которые находятся в состоянии останова, можно при этом не отключать от контроллера.

Запуск отдельных процессоров в тестовом режиме не синхронизируется. Процессоры входят в цикл в разные моменты времени в зависимости от продолжительности вызова OB 20, 21 или 22.

Если в тестовом режиме в каком-то процессоре возникает ошибка, то в состояние останова переходит только этот процессор. Другие процессоры не реагируют на эту ошибку.

Важно!
В тестовом режиме выдача сигнала BASP у всех процессоров подавляется. Если возникает ошибка, то выходы периферии не блокируются, т.е. не отключаются (исключения смотри дальше).

Важно!
По завершении тестового режима отключите его переключателем на координаторе! Это позволит Вам избежать аварийной ситуации, которая может возникнуть в определённых обстоятельствах при ошибке!

Выводы: Порядок запуска многопроцессорного режима

- На координаторе устанавливается количество процессоров. В координаторе освобождаются маркеры связи.
- Процессоры вставляются в центральное устройство без пропусков установочных мест.
- Подключается напряжение питания.
- Селектор режимов работ координатора установить в положение “STOP”.
- На всех процессорах, установленных в центральном устройстве, произвести стирание памяти.
- Загрузить в процессор программу пользователя.
- Селектор режимов работ координатора перевести в положение “RUN” или “TEST”.

11. Вспомогательные средства тестирования: On-line – функции

Важнейшими вспомогательными средствами для тестирования программы пользователя являются On-line – функции. Подробное описание программатора и использования этих функций Вы найдёте в Руководстве по программатору. В предлагаемом Разделе описаны некоторые особенности On-line – функций применительно к CPU 928.

В программируемом контроллере On-line – функции выполняются в определённых точках. При этом задаются точки в операционной системе
(= точки системного контроля) и точки в виде команд в программе пользователя (= точки контроля пользователя).

При использовании контрольных точек обработка программы дополнительно увеличивается максимум на 5 мс.

Точки системного контроля

В рабочем состоянии “СТОП” существует точка системного контроля СТОП, которая регулярно опрашивается.

В рабочем состоянии RUN (“РАБОТА”) вызываются точки системного контроля ЦИКЛ в конце уровня обработки цикл перед обновлением области отображения процесса.

Если процессор находится в состоянии ОЖИДАНИЯ, то регулярно вызывается точка системного контроля состояния ОЖИДАНИЯ.

Дополнительно имеется зависящая от времени точка системного контроля TIMEOUT. Она вызывается только в том случае, если не были вызваны все другие точки системного контроля в течение 250 мс. Эта точка системного контроля может быть также включена во время выполнения программы. Такая ситуация может возникнуть, например, при бесконечном цикле в программе пользователя или при цикле, длительность которого превышает 250 мс.

Обработка On-line – функциями точек системного контроля прерывается максимум через 5 мс и продолжается со следующей точки системного (см. также таблицу 11.11)

Точки контроля пользователя

С помощью тестовых функций “СТАТУС” и “КОНТРОЛЬ ОБРАБОТКИ” можно использовать точки контроля прерывателя. Последние вызываются, если выполняются команды, помеченные с помощью PG.

Рабочее состояние ожидание

До сих пор Вы знали рабочие состояния СТОП, ЗАПУСК и РАБОТА. С помощью функции PG “КОНТРОЛЬ ОБРАБОТКИ” контроллер может перейти ещё в одно состояние работы – ОЖИДАНИЕ. Если процессор находится в состоянии ОЖИДАНИЯ, то можно вызвать некоторые функции PG.

Включение состояния ожидания

- В состоянии ОЖИДАНИЕ программа пользователя не обрабатывается.
- Светодиоды на передней панели: RUN : Выключен
  STOP: Выключен
  BASP: Включен
- Все таймеры не работают (значения таймеров не изменяются). Кроме того, остаются неизменными все системные таймеры (например, для регулировки и для обработки программы, управляемой по времени).

  При выходе из состояния ожидания, таймеры продолжают свою работу.

- Причины прерывания, такие как, например, PEU, MP-STP или STP-SCH регистрируются в состоянии ожидания, однако реакция на них отсутствует.

  Если в состоянии ожидания зарегистрирована причина прерывания, то соответствующий уровень обработки программы вызывается непосредственно после выхода из состояния ожидания.

  Если возникла ошибка NAU, происходит выход из состояния ожидания и прерывание On-line – функции “КОНТРОЛЬ ОБРАБОТКИ”. После подачи напряжения в управляющих битах помечается крестиком сокращение “BARBEND”. Выход из состояния останова возможен только с помощью нового запуска.

11.1. On-line – функция “СТАТУС ПЕРЕМЕННЫХ”

С помощью On-line – функции “СТАТУС ПЕРЕМЕННЫХ” Вы можете вывести текущее состояние сигналов определённых операндов (переменных процесса). Функция активирует точки системного контроля в ЦИКЛЕ, в СТОПЕ и в ОЖИДАНИИ. Если достигнута точка системного контроля, то в этот момент выводится текущее состояние требуемых сигналов от процесса. Вы можете вывести все переменные процесса. В области отображения процесса ADF не разрешена.

Выполнение функции во время обработки программы:

Функция выполняется в рабочем состоянии ЗАПУСК или РАБОТА, если обработка программы продолжается столь долго, что будет достигнута точка системного контроля ЦИКЛ. При этом, в конце цикла опрашивается и выдаётся на выход состояние сигналов операндов. Выходы считываются из области отображения процесса. Пока функция не будет прервана, состояния сигналов будут продолжать обновляться при текущей обработке программы. Состояния сигналов при этом опрашиваются не на каждой точке системного контроля.

Если точка системного контроля “цикл” не достигнута, состояния сигналов не выводятся (например, при бесконечном цикле программы пользователя)!

Выполнение функции в состоянии СТОП:

Если функция “СТАТУС ПЕРЕМЕНЫХ” выполняется в СТОПе, то выдаётся состояние операндов, которые опрошены на точке системного контроля “состояние останова”. При этом важно, что информация о состоянии входов принимается непосредственно с периферийных модулей. В связи с этим возможна, например, проверка подачи на процессор сигнала с периферийного входа. Кроме того, Вы можете опросить в многопроцессорном режиме все входы независимо от их распределения в DB1. информация о состоянии выходов считывается из области отображения.

Выполнение функции в состоянии ОЖИДАНИЯ:

Функция “СТАТУС ПЕРЕМЕНЫХ” также может быть вызвана, если процессор находится в состоянии ожидания по функции “КОНТРОЛЬ ОБРАБОТКИ”. В точке системного контроля “состояние ожидания” опрашивается и выдаётся состояние сигналов операндов. Как и в цикле, при этом входы считываются не непосредственно, а из **области отображения процесса.**

Если процессор переходит из одного состояния в другое (например, RUN → STOP → ручной перезапуск), функция в дальнейшем остаётся вызванной.
Завершение функции “СТАТУС ПЕРЕМЕНЫХ” выполняется нажатием клавиши прерывания на программаторе.

Указание: Переменные выводятся не в каждом следующем цикле.

11.2. On-line – функция “СТАТУС”

С помощью On-line – функции “СТАТУС” Вы можете определить в любом месте программы пользователя работающие совместно в блоках последовательности команд.
Для каждой выполняемой команды в блоке на программатор выдаётся текущее состояние сигнала операнда, содержимое аккумулятора, VKE и т.д. Параметрирование функциональных блоков также может быть проверено с помощью этой функции: при этом индицируется состояние сигналов текущих операндов.

Вызов функции и указание точек останова

Если Вы с помощью PG вызвали функцию “СТАТУС” и ввели тип блока и номер блока (возможно с последовательностью вложений и с определением поиска), Вы также должны указать для тестируемого блока так называемую точку останова.

При вызове функции во время режимов ЗАПУСК или РАБОТА обработка программы продолжается долго, пока не будет достигнута команда, помеченная как точка останова в правильной последовательности вложения. Затем эта команда выполняется до границы команды, и результат обработки команды выдаётся на PG.

Функция “СТАТУС” вызывается также и в состоянии останова. Затем возможны как новый пуск, так и ручной перезапуск. Процессор обрабатывает при этом программу пользователя до указанной в программе точки останова. Затем данные желаемой последовательности команд выводятся. Вместе с тем, использование функции “СТАТУС” возможно в том случае, если программа пользователя проверяется при запуске или при первом цикле.

Указание: Результат обработки команд выдаётся не в каждом следующем цикле.

Вложение и прерывание

Последовательностью команд, помеченная заданной точкой останова, выполняется полностью, даже если в промежутке на границах команд вклиниваются другие уровни обработки программ (например, OB ошибки, прерывание от процесса или по времени). Данные, которые были изменены вложенными обработками программ, могут быть опознаны.

Если во вложенном уровне обработки программ возникает причина прерывания, переводящая процессор в СТОП, то в состоянии СТОП выдаются данные до соответствующей команды, которая была выполнена последней перед вложением. Данные остальных команд заполняется нулями (также SAZ = 0).

Если процессор из одного рабочего состояния переходит в другое (например, RUN → STOP → ручной перезапуск), то функция остаётся вызванной.
Завершить функцию “СТАТУС” можно, нажав клавишу прерывания на программаторе.

11.3. On-line-функция «КОНТРОЛЬ ОБРАБОТКИ»

С помощью On-line-функции «КОНТРОЛЬ ОБРАБОТКИ» можно проверить отдельные шаги программы пользователя в любом месте программы. При этом Вы останавливаете обработку программы, и затем процессор начинает обработку команд одну за другой. Для каждой выполненной команды текущее состояние сигналов операндов, содержимое аккумуляторов, VKE и т. д. выводятся на программатор.

Вызов функции и указание первой точки останова

Для вызова функции «КОНТРОЛЬ ОБРАБОТКИ» необходимо указать тип и номер (возможно с последовательностью вложений) тестируемого блока и пометить с помощью PG первую команду, данные которой должны выводиться, Затем укажите первую точку останова. В битах управления будет помечен крестиком бит «BARB». Вывод команд блокирован (светодиод BASP= вкл).

Указание: Если Вы установили на координаторе тестовый режим, то выводы не блокируются (светодиод BASP=выкл.). Если теперь выполнить команду, изменяющую состояние цифровой периферии, или если процессор выполняет обновление области отображения процесса, модули выходов выдают соответствующие сигналы.

Если Вы укажите первую точку останова при обработке программы в режиме ЗАПУСК или ОБРАБОТКА , процессор будет продолжать обработку столь долго, пока не будет достигнута команда, помеченная точкой останова. Команды выполняются до границы команды. (Команды BMW и BDW обрабатываются исключительно как замещение команды). Затем процессор перейдет в состояние ожидания. В этом режиме выводятся данные обработанных команд.

Вызов функции в режиме СТОП:

Функцию «КОНТРОЛЬ ОБРАБОТКИ» можно вызвать также в состоянии останова и указать первую точку останова. Процессор остается при этом в состоянии СТОП. Теперь можно выполнить новый пуск или ручной перезапуск. Процессор продолжит обработку программы до помеченной команды и дальше все произойдет. как описано выше.

Продолжение функции и задание дальнейших точек останова

Исходная точка: Процессор находится в состоянии ожидания.

Для того, чтобы продолжить функцию «КОНТРОЛЬ ОБРАБОТКИ» мы имеем две возможности.

1. Вы задаете следующие точки останова:

   Указанная точка останова смещается на одну команду. Процессор покидает состояние ожидания и продолжает обработку программы на этой команде. Если команда обработана до границы команды, процессор снова переходит в состояние ожидания и выдает данные. Однако, если достигнута следующая команда во вложенном уровне обработки программы, то процессор продолжает обработку программы. Последовательность точек останова остается при этом заданной.

   Важно: В состояние останова нельзя задавать последующие точки останова!

2. Вы можете ввести новые точки останова:

   Вы можете с помощью PG определить любые другие команды в том же или в других блоках. Процессор продолжит обработку программы, пока не будет достигнута новая точка останова. Команда обрабатывается до границы команды. Затем процессор переходит в состояние ожидания, и выводятся данные.

   Вы можете с помощью контроля обработки запустить процессор на выполнение полного цикла (циклическое тестирование). Для этого установите в состоянии ожидания точку останова на той же команде, что и перед этим. Правда команда может быть не найдена в цикле программы. В этом случае цикл выполняется один раз, обработка программы не переходит через границу цикла.

Указание: В состоянии ожидания можно вызвать другие функции, такие как, например, «ВЫЗОВ БИБЛИОТЕКИ» или «СТАТУС ПЕРЕМЕННЫХ».
Как только обработка программы будет продолжена, таймеры и системные часы продолжат работать, пока не будет достигнута точка останова.

Сброс точек останова:

Если указанная точка останова еще не достигнута, то имеется возможность сбросить ее, нажав на PG-клавишу прерывания. Процессор при этом войдет в состояние ожидания. Затем можно задать новую точку останова или вызвать функцию «КОНЕЦ КОНТРОЛЯ ОБРАБОТКИ».

Прерывание Функции

Вызвав Функцию «КОНЕЦ КОНТРОЛЯ ОБРАБОТКИ», можно во время обработки программы, в состоянии ожидания и в состоянии СТОП прервать выполнение функции. Процессор перейдет в СТОП (или останется в СТОПе). Светодиод STOP медленно мерцает. Управляющий бит 'BAREBEND" помечается крестиком. После этого необходим новый пуск. Если во время Функции 'КОНТРОЛЬ ОБРАБОТКИ' возникла ошибка интерфейса (обрыв кабеля PG), то функция прерывается, как было описано выше.

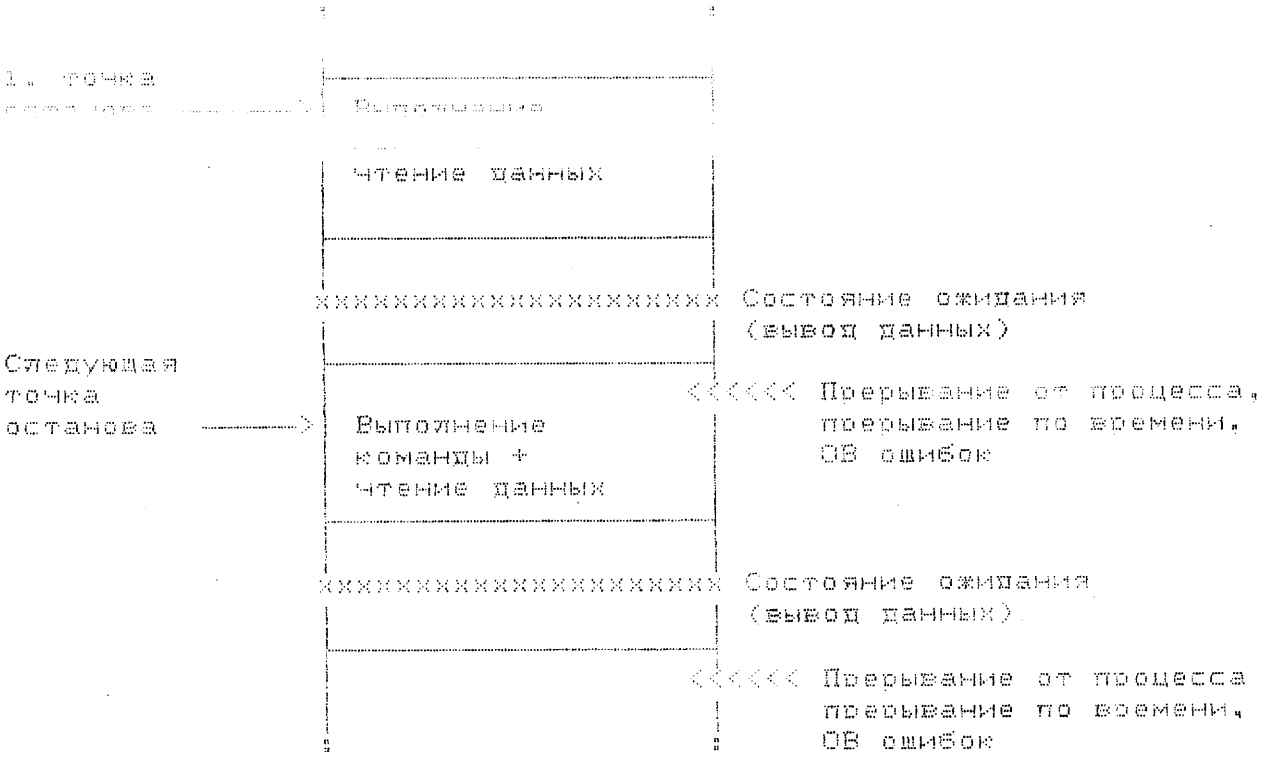
Вложения

При вызове функции «КОНТРОЛЬ ОБРАБОТКИ» перед переходом процессора в состояние ожидания может произойти вложение или, при переходе из состояния ожидания, - вызов других уровней обработки программы.

Если обработана команда точки останова и на этом месте вызван другой уровень обработки программы (ОВ ошибки, прерывание от процесса или по времени), то он непосредственно вкладывается и полностью обрабатывается, при этом мы считаем; что он может быть прерван на границе команд. Сразу после этого процессор переходит в состояние ожидания и выдает данные.

Важно!

Данные считываются на границе команды, однако, выдаются сразу, если все соответствующие вложения обработаны до конца.



Если в состоянии ожидания возникают запросы PEU, MP-STP, STP-SCH и т. д., то они только регистрируются. Сразу после выхода из состояния ожидания они могут стать действительными: Вкладывается уровень обработки программы или выполняется прерывание и происходит переход в состояние СТОП.
Действует последовательность событий. Одновременно приоретезируются запросы.

Указание: Если процессор находится в состоянии ожидания и требуется вложение, то Вы имеете возможность установить точку останова на команде вложения. Таким образом, Вы, например, можете сразу после команды, которая вызывает QVZ, проследить выполнение 0B ошибки QVZ.

Прерывания

- Обработка программы (ЗАПУСК/РАБОТА)—>состояние останова

  Если во время обработки программы возникает причина прерывания (например, MP—STP. PEU, STP-"SCH, не запрограммирован OB ошибки и т.д.) и указанная точка останова еще не достигнута, процессор немедленно переходит в состояние останова.
  Если теперь будет выполнен новый запуск (новый пуск или ручной перезапуск), Функция «КОНТРОЛЬ ОБРАБОТКИ» остается вызванной и точки остаются заданными.

- Обработка команды в точке останова (ЗАПУСК/РАБОТА) —>состояние останова

  Если во время обработки команды в точке останова или в следующей точке останова возникает условие останова (STP-SCH, команда STEP5 ‘STP’, не запрограммирован OB ошибки), процессор немедленно переходит в СТОП и после этого начинает выдавать данные.
  Если состояние СТСП не было задано новой точкой останова, процессор после запуска переходит в состояния ожидания. Функция "КОНТРОЛЬ ОБРАБОТКИ' остается вызванной,

- Состояние ожидания —> состояние останова

  Причина прерывания, которая возникла в режиме ожидания (например MP-STP, PEU или переключатель режимов в СТОП) регистрируется, однако, процессор остается в режиме ожидания. Если в с режиме ожидания была задана новая точка останова и процессор выходит из состояния ожидания, то имеющиеся причины прерываний переводят его немедленно в состояние останова. Указанная точка останова не постигается.
  Если после этого выполняется запуск (новый пуск или ручной перезапуск), точка останова остается заданной.

Важно!
Если в состоянии ожидания перевести переключатель в СТОП, то сразу после выхода из состояния ожидания процессор перейдет в состояние СТОП.

Важно!
Появление ошибки во время выполнения Функции "КОНТРОЛЬ ОБРАБОТКИ' переводит процессор в состояние СТОП, если это происходит, то после последующего запуска Функция "КОНТРОЛЬ ОБРАБОТКИ” (и возможные точки останова) остаются активными!

11.4 On-line-функция «УПРАВЛЕНИЕ»

С помощью функции «УПРАВЛЕНИЕ» выходные байты программируемого контроллера можно установить вручную в требуемое состояние. Вы имеете возможность непосредственно управлять и контролировать выходы исполнительных устройств контроллера (двигатели, вентили).

Важно!
Функция «УПРАВЛЕНИЕ» допустима только в состоянии останова!

Вызов функции

При вызове функции в состоянии СТОП снимается блокировка выхода команд (BASP = выкл.). Вся цифровая периферия (F000H по F07FH) стирается, и в каждый адрес записывается значение 0. Во время стирания периферии функция не может быть прервана.

Указание периферийными выходами осуществляется по байтам, непосредственно и без влияния на область отображения выходов!
В многопроцессорном режиме Вы можете управлять всеми периферийными выходами (независимо от распределения периферии в DB1).

Если функция актива (сообщение «Управление готово»), Вы можете выполнить новый пуск иди ручной перезапуск. После нового перехода в СТОП Вы можете продолжать управление, формирователь выходных сигналов в этом случае не стирается.

Прерывание функции

Функцию можно завершить, нажав клавишу прерывания на PG. Блокировка выдачи команд снова включается (BASP = вкл.).

11.5. On-line-функция «УПРАВЛЕНИЕ ПЕРЕМЕННЫМИ»

С помощью On-line-функции «УПРАВЛЕНИЕ ПЕРЕМЕННЫМИ» Вы можете однократно изменить значения операндов (переменных процесса). Это допустимо в любом состоянии процессора. Вы можете изменить все переменные процесса. В области отображения процесса ADF не действует.

Изменение действует в точках системного контроля. Помните, что управляемые значения могут быть потом переписаны (например, программой пользователя или при обновлении области отображения процесса)!

Указание: PG управляет переменными процесса E, A, M по байтам, а DW, T, Z по словам.

Важно!
Если Вы управляете несколькими операндами, то изменяемые байты (при DW, T, Z – слова) распределяются друг за другом по нескольким точкам системного контроля в памяти и изменяются.

11.6 On-line-функция «СЖАТИЕ ПАМЯТИ»

С помощью этой функции можно все действительные блоки программы пользователя уплотнить, поставить в определенном порядке один за другим. Эта функция выполняется раздельно в модуле RAM и в DB-RAM. Промежутки, которые возникают в процессе стирания и коррекции блоков, удаляются. При этом даже один полный блок сдвигается к началу области памяти. Это может быть выполнено в точках системного контроля «цикл» и «стоп».

Функция может быть прервана сообщением об ошибке, если на точке системного контроля «стоп» BSTACK не пуст. Этот случай возникает, когда во время обработки программы прерывание перевело процессор в состояние СТОП. В этом случае дальнейшее сжатие может происходить только в цикле.

Если во время сжатия произошел сбой напряжения, то остальные блоки не сдвинутся. При новом вызове «СЖАТИЯ ПАМЯТИ» происходит дальнейшее уплотнение.

Важно!
Функция «СЖАТИЕ ПАМЯТИ» опознает следующие ошибки в памяти блоков:
- **неверная длина блока**
- **неверное значение «7070» в заголовке блока**
- **недействительный тип блока (недействительный номер блока OB)**

Функция в этом случае может быть прервана. На PG выдается сообщение. Затем должно быть выполнено полное стирание памяти. Функция может быть снова вызвана сразу после полного стирания.

Указание: «СЖАТИЕ ПАМЯТИ» недопустимо пока активен «КОНТРОЛЬ ОБРАБОТКИ»!

11.7. Функции PG «СТАРТ» / «СТОП»

Применение PG соответствует обслуживанию вручную.
С помощью функции «СТОП» Вы можете перевести программируемый контроллер в СТОП. На том процессоре, который подключен к PG, Вы увидите следующую картину:

Светодиод STOP: Вкл.
Светодиод BASP: Вкл.

В управляющих битах метка «MP-STP» помечается крестиком. В многопроцессорном режиме устанавливается управляющий бит «MP-STP».

Вы можете запустить процессор с помощью нового пуска или ручного перезапуска. В однопроцессорном режиме сначала регистрируются виды запуска (бит управления «NEUST» или «M W A» установлены), но процессор остается в СТОПе. По следующей команде «запуск системы» Вы можете запустить программируемый контроллер. Это эквивалентно включению координатора в RUN.

Функция PG «СТАРТ» позволяет в многопроцессорном режиме последовательно на всех процессорах выбрать вид запуска, и сразу после последнего процессора AG запустится.

11.8. 0n-line"'функция ‘ПОЛНОЕ СТИРАНИЕ AG'

С помощью PG можно полностью обнулить память процессора. При этом полное стирание выполняется, безусловно (см. Раздел 4.2)

При вызове функции «ПОЛНОЕ СТИРАНИЕ AG» в состоянии ЗАПУСК или РАБОТА сначала выполняется переход в состояние останова. При этом вызывается организационный блок OB28, если он загружен.

Указание: ‘ПОЛНОЕ СТИРАНИЕ AG' недопустимо пока активен ‘КОНТРОЛЬ ОБРАБОТКИ’!

11.9. On-line-функция 'ВЫВОД АДРЕСА'

Функции 'ВЫВОД АДРЕСА' позволяют вывести на PG содержимое адресов памяти и периферии в дословном виде. Вы можете спросить все адреса. В области отображения процесса ADF не действует и не возникает QVZ.

В побайтно адресованных областях (маркеры, области отображения процесса) старшие байты представляются в виде ‘FF’.

В области периферии при квитируемых адресах старшие байты выдаются как «00». Если адреса периферии не квитируются, старшие байты индицируются как ‘FF’.

11.10. On-line—Функция 'ЗАПОЛНЕНИЕ ПАМЯТИ'

Функция 'ЗАПОЛНЕНИЕ ПАМЯТИ' показывает на PG наивысший используемый адрес модуля RAM (при EPROM индицируется ‘0’) и последний занятый блоками программы пользователя адрес модуля памяти.

11.11. Таблица: Действия в точках контроля

Действия при On-line-функциях	Точки системного контроля			
	СТОП	ЦИКЛ	СОСТОЯНИЕ ОЖИДАНИЯ	TIMEOU
Ввод адреса: Запись данных 1)	*	*	*	*
Ввод стека: Блок объявляется действительным	*	*	*	
Стирание памяти	*	*	*	*
Сжатие памяти: Блоки сдвигаются 2)	*	* 3)		
Старт/стоп	*	*	*	
Полное стирание	*	*		*
SRATUS VAR: Данные считываются и выводятся	*	*	*	
STATUS: данные считываются и выводятся				
Контроль обработки: Задаются точки останова	*	*	*	*
Данные считываются				
Данные выводятся	*		*	
Управление формирователем сигналов 1)	*			
Управление переменными 1)	*	*	*	*

1) Действия, которые можно разделить на несколько точек системного контроля

2) На каждую точку системного контроля максимум один блок

3) Только если нет данных в BSTACKe

Таблица: Действия, которые выполняются в точках системного контроля и контроля пользователя.

Приложение A: Технические данные ПК S5-135U

	S-процессор	R-процессор
Среднее время выполнения команды для команд обработки битов с		
M, E, A	1,1мкс	22мкс
D	100мкс	37мкс
Формальными операндами	94мкс	416мкс
Среднее время выполнения команды для команд обработки слов		
-Операции загрузки		
L MB (байт)	31мкс	15мкс
L MW (слово)	33мкс	15мкс
L MD (двойное слово)	70мкс	20мкс
-Арифметика с фиксированной и с плавающей запятой	< 900мкс	< 86мкс
Циклическая обработка программы (однопроцессорный режим)		
- Основное время при вызове OB1/FB0	221/263мкс	107/11'9мкс
- Добавка на обновление области отображения процесса в зависимости от числа байтов вх/вых (n), где 0 < n <= 256	23мкс + n x 1,63мкс	33мкс + n x 6мкс
- Добавка на передачу маркеров связи в зависимости от числа маркеров связи (n), где 0 < n<= 256	25мкс + n x 1,63мкс	35мкс + n x 6мкс
- Добавка на обработку таймеров в зависимости от длины таймерного блока (ZBL)	все 10мс	все 2.5мс
ZBL = 0	20мкс	50мкс
ZBL # 0	50мкс + ZBL + 9,3мкс	60мкс + ZBL + 1,06
n= Число действующих таймеров (база времени: 10мс)	+ n x 17,9мкс	+ n x 1,24мкс
- Обработка программ управляемых по прерыванию от процесса Увеличение времени цикла за счет вложенности пустого 0B 2 (без команд STEP 5) на границе блока	439мкс	367мкс
- Время реакции	425мкс	300мкс

	S-процессор	R-процессор
- Обработка программ управляемых по времени Увеличение времени цикла за счет вложения пустого 0B 13 (без команд STEP 5) на границе команды	327мкс	373мкс
Такт времени для вызова программ управляемых по времени	100мкс	100мкс
Контроль времени цикла предустановка диапазон установок запоминание	 100мс - да	 150мс 1...4000мс да
Размер области памяти (в К Словах)	< 32	< 32
Размер памяти для блоков данных (DB-RAM, в К Словах)	около 3,7	около 11.1
Число таймеров и счетчиков	до 123	до 256
Число маркерных байтов	до 256	до 256

Пояснения понятий:

Основное время: Под основным временем понимается время выполнения соответствующей, циклически выполняемой, части операционной системы без обновления области отображения процесса и без передачи маркеров связи.

Время реакции: Под временем реакции понимается время от активизации уровня обработки программ ПРЕРЫВАНИЕ ОТ ПРОЦЕССА до обработки первой команды в 0B 2. предполагая, что 0B 2 вызывается сразу после определения прерывания от процесса. Если же нужно ждать до следующей границы команды или блока, то время реакции увеличивается.

Приложение B: Обзор меток ошибок

Слова системных данных 3 и 4

SD3 SD4	
Создание спи	ска адресов блоков:
800114 yyyyH	Неверная длина блока, yyyy—адрес блок
8002H yyyyH	Вычисленный конечный адрес блока в памяти неверен yyyy—адрес блок
8003H yyyyH	Недействительная метка блока yyyy—адрес блок
8004H yyyyH	Слишком большой номер организационного блока (допустимо 1...39) yy
8005H yyyyH	Номер блока данных = 0 (допустимо: DB1...255) yyyy—адрес блок
Создание спи	ска адресов для области отображения процесса:
0410H vyyyH	Недействительная метка yyyy—неверно
0411H yyyyH	Неверный параметр в списке адресов цифровых входов yyyy—адрес не
0412H vyyyH	Неверный параметр в списке адресов цифровых выходов yyyy—адрес не
0413H vyyyH	Неверный параметр в списке адресов входных маркеров связи yvyy—ад. маркеров
0414H yyyyH	Неверный параметр в списке адресов выходных маркеров связи yyyy— выходных маркеров
0415H yyyyH	Неправильное количество таймеров (допустимо 256) yyyy—неверно
0419H vyyyH	Задержка квитирования от цифровых входов yyyy—адрес неквитируем
041AH yyyyH	Задержка квитирования от цифровых выходов yyyy—адрес неквитируем
041BH vyyyH	Задержка квитирования от входных маркеров связи yyyy—адрес некви межпроцессорной связи
041CH vyyyH	Задержка квитирования от выходных маркеров связи yyyy—адрес некв межпроцессорной связи
ОбработкаDB	2:
04214 DBvvH	Блок данных не загружен yy—номер незагруженного DB
0422H FBvyH	Функциональный блок не загружен yy—номер незагруженного F5
0423H FByyH	Функциональный блок не опознан yy—номер неопознанного FB
0424H FByyH	Загружен Функциональный блок с неверным программным обеспечен. F
0425H DByyH	Неверная длина блока данных регулятора yy=номер блока данных
0426H -	Не достаточно места в памяти DB-RAM
ОбработкаDX	0:
0431H yyyyH	Неверная метка yyyy—неверная метка
0432H yyyyH	Неизвестный параметр yyyy—неверный параметр
0433H yyyyH	Недопустимое число таймеров (допустимо 256) yyyy—неверное
0434H yyyyH	Недопустимое время цикла (допустимо 1мсек...4сек) yyyy-неверная E

АКК 1 иАКК2

АКК 1 АКК 2	
Обработка р	егулятора:
0801H DByyH	Ошибка времени опроса yy—номер соответствующего DB
0802H DByyH	Не загружен DB регулятора yy-номер незагруженного DB
0803H FByyH	Не загружен FB регулятора yy—номер незагруженного FB
0804H FByyH	Не опознан FB регулятора yy—номер FB
0805H FByyH	FB регулятора загружен с неверным SOFTWARE PG yy—номер
0880H yyyyH	FB Задержка квитирования QVZ во время обработки регулятора

1801H	—	Ошибка замещения по команде BBS
1802H	—	Ошибка замещения по команде BDW/BMW
1803H	—	Ошибка замещения по команде B=/BI=
1804H	—	Ошибка замещения по команде L=/T=
1805H	—	Ошибка замещения по команде U=/UN=/O=/ON=/==/S= и RB=
1811H	—	Команда с недопустимым кодом операции
1812H	—	Команда с недопустимым кодом операции (старший байт первого слова команды равен 68H)
1813H	—	Команда с недопустимым кодом операции (старший байт первого слова команды равен 78H)
1814H	—	Команда с недопустимым кодом операции (старший байт первого слова команды равен 70H)
1815H	—	Команда с недопустимым кодом операции (старший байт первог слова команды равен 60H)
1821H	—	Недопустимый параметр при A DBO, A DB1 и A DB2
1822H	—	Недопустимый параметр при SPA(B) OBO
1823H	—	Недопустимый параметр при SPA(B) OB>39. Спец.функция отсутствует
1824H	—	Недопустимый параметр при AX DXO
1825H	—	Недопустимый параметр при LMW/TMW/LPW/TPW/LOW/TOW/LDD/TDD/BMW255
1826H	—	Недопустимый параметр при LEW/TEW/LAW/TAW 127
1830H	—	Недопустимый параметр при LMD/TMD 253, 254, 255
1831H	—	Недопустимый параметр при LED/TED/LAD/TAD 125,126,127
1832H	—	Недопустимый параметр при RLD/RRD/SVD/SLD 33–255
1833H	—	Недопустимый параметр при SLW/SRW/LIR/TIR 16–255
1834H	—	Недопустимый параметр при SES/SEF 32–255
1835H	—	Недопустимый параметр при U=/UN=/O=/ON=/S=/RB=/==/RD=/FR=/SI=/ SE=/SVZ=/SSV=/SAR=/L=/LC=/LW=/T= 0, 127–255
1836H	—	недопустимый параметр при B=/LD= 0, 127–255
Ошибки выполнения команд STEP5:		
1A01H	—	Незагруженный блок данных при A DB
1A02H	—	Незагруженный блок данных при AX DX
1A03H	—	Незагруженный блок при SPA(B) FB,OB,PB и SB
1A04H	—	Незагруженный блок при BA(B) FX
1A05H	—	Незагруженный блок при OB254 или OB255
1A11H	—	Обращение к неопределенному слову данных
1A12H	—	Ошибка передачи в неопределенное слово данных (T DR)
1A13H	—	Ошибка передачи в неопределенное слово данных (T DL)
1A14H	—	Ошибка передачи в неопределенное слово данных (T DW)
1A15H	—	Ошибка передачи в неопределенное слово данных (T DD)
1A21H	—	Ошибка при EDB,EXDX: DB уже существует
1A22H	—	Ошибка при EDB,EXDX: Недопустимая длина DB
1A23H	—	Ошибка при EDB,EXDX: Недостаточно места в памяти RAM
1A25H	—	Ошибка при BI=: Недопустимый параметр в AKKU1
1A29H	—	Недостато или превышение количества скобок U(,O(,)
1A2AH	—	Ошибка при ADB или AXDX:Длина заголовка слишком мала (<5слов)
1A2BH	—	
1A2CH	—	
1A31H	—	Ошибка SOFTWARE в OB254 или OB255: DB уже имеется в RAM
1A32H	—	Ошибка SOFTWARE в OB254 или OB255: Новый DB уже имеется
1A33H	—	Ошибка SOFTWARE в OB254 или OB255: Недостаточно места в RAM
1A3AH	—	Ошибка SOFTWARE в OB221 : Недопустимое новое время цикла
1A3BH	—	
1A41H	—	Ошибка SOFTWARE в OB240,241,242: Недопустимый номер регистра сдвига или DB
1A42H	—	Ошибка SOFTWARE в OB241: Регисто сдвига не инициализирован
1A43H	—	Ошибка SOFTWARE в OB240: Недостаточно места в памяти RAM
1A44H	—	Ошибка SOFTWARE в OB240: Содержимое DW0 блока данных не равно
1A45H	—	Ошибка SOFTWARE в OB240: Недопустимая длина регистра в DW1
1A46H	—	Ошибка SOFTWARE в OB240: Недопустимая позиция указателя или число указателей

1A47H	—	Ошибка SOFTWARE в OB120: Недопустимое значение в AKKU 1 или AK
1A48H	—	Ошибка SOFTWARE в OB122: Недопустимое значение в AKKU 1
1A49H	—	Ошибка SOFTWARE в OB110: Недопустимое значение в AKKU 1
1A4AH	—	Ошибка SOFTWARE в OB121: Недопустимое значение в AKKU 1
1A4BH	—	Ошибка SOFTWARE в OB123: Недопустимое значение в AKKU 1
1A50H	—	Ошибка при LRW, TRW: Недопустимый адрес
1A51H	—	Ошибка при LRD, TRD: Недопустимый адрес
1A52H	—	Ошибка при TSG, LBGB, LWGW, TBGB, TWGW: Недопустимый адрес
1A53H	—	Ошибка при LBGW, LWGD, TBGW, TWGD: Недопустимый адрес
1A54H	—	Ошибка при LBGD, TBGD: Недопустимый адрес
1A55H	—	Ошибка при TSC, LBCB, LWCD, TBCW, TWCD: Недопустимый адрес
1A56H	—	Ошибка при LBCW, LWCD, TBCW, TWCD: Недопустимый адрес
1A57H	—	Ошибка при LBCD, TBCD: Недопустимый адрес
1A58H	—	Ошибка при TNW/TNB: Исходный блок не полностью лежит в обл ти
1A59H	—	Ошибка при TNW/TNB: Приемный блок не полностью лежит в области

Задержка квитирования:

1E23H yyyyH	Задержка квитирования (QVZ) в программе пользователя yyyy — адрес QVZ
1E25H yyyyH	Задержка квитирования (QVZ) в области отображения цифровых выходов yyyy — адрес неквитируемого выходного байта
1E26H yyyyH	Задержка квитирования (QVZ) в области отображения цифовых входов yyyy — адрес неквитируемого входного байта
1E27H yyyyH	Задержка квитирования (QVZ) в области отображения выходных маркеров связи yyyy — адрес неквитируемого байта маркеров
1E28H yyyyH	Задержка квитирования (QVZ) в области отображения входных маркеров связи yyyy — адрес неквитируемого байта маркеров

Приложение C: Обзор операций языка STEP-5

Основные операции

Операция	Параметр
О Двоичные логические операции	
U E	0(
U A	O
U M	0.0. . .127,7
U D	0.0. . .127,7
U T	0.0. . .255,7
U Z	0.0. . .255,15
UN E	0. . .255
UN A	0. . .255
UN M	0.0. . .127,7
UN D	0.0. . .127,7
UN T	0.0. . .255,7
UN Z	0.0. . .255,15
0 E	0. . .255
0 A	0. . .255
0 M	0.0. . .127,7
0 D	0.0. . .127,7
0 T	0.0. . . 255,7
0 Z	0.0. . . 255,15
ON E	0. . . 255
ON A	0. . . 255
ON M	0.0. . .127,7
ON D	0.0. . .127,7
ON T	0.0. . . 255,7
ON Z	0.0. . . 255,15
)	0. . .255
U(	0. . .255
О Функция сравнения	
!=F	
><F	
>F	
>=F	
<F	
<=F	
!=D	
><D	
>D	
>=D	
<D	
<=D	
!=G	
><G	
>G	
>=G	
<G	
<=G	

Операция	Параметр
о Операции с памятью	
S E	0.0. . .127,7
S A	0.0. . .127,7
S M	0.0. . .255,7
S D	0.0. . .255,15
R E	0.0. . .127,7
R A	0.0. . .127,7
R M	0.0. . .255,7
R D	0.0. . .255,15
= E	0.0. . .127,7
= A	0.0. . .127,7
= M	0.0. . . 255,7
= D	0.0. . . 255,15
О Функция загрузки	
L EB	0 . . . 127
L EW	0 . . . 126
L ED	0 . . .124
L AB	0 . . .127
L AW	0 . . .126
L AD	0 . . .124
L MB	0 . . .255
L MW	0 . . .254
L MD	0 . . .252
L DL	0 . . .255
L DR	0 . . .255
L DW	0 . . .255
L DD	0 . . .254
L T	0 . . .255
L Z	0 . . .255
L PY	0 . . . 127 128. . . 255
L PW	0 . . . 126 128 . . . 254
L QB	0 . . . 255
L QW	0 . . . 254
LC T	0 . . . 255
LC Z	0 . . . 255
L KB	0 . . . 255
L KC	2 алфав.-цифр. Символа
L KM	Набор 16 бит
L KH	0 . . . FFFF
L KF	-32768 . . . . . .+32767
L KY	0 . . . 255 для каждого байта
L KT	0.0. . .999,3
L KZ	0. . . 999
L KG	[1])

[1]) 0.1469368 x 10^{-38}
0.1701412 x 10^{39}

Операция	Параметр
с таймерами и счетчиками:	
	255
SV T	0 ... 255
SE T	0 ... 255

Операция	Параметр
LC	
LF	

о Операции передачи:	
T CV	0 ... 127
	0 ... 126
T AB	0 ... 127
T AW	0 ... 126
T AD	0 ... 124
T DL	0 ... 255
T DW	0 ... 255
T FY	0 ... 127
	128 ... 255
T PW	0 ... 126
	128 ... 254
T OB	0 ... 255

STP	
BLD	0 ... 255

Дополнительные операции

DW	
AUW	

о Операции с таймерами и счетчиками:	
SI =	Форм.операнд
SE =	Форм.операнд
SVZ =	Форм.операнд
SSV =	Форм.операнд
SAZ =	Форм.операнд

SPA PB	0 ... 255
SPA PB	0 ... 255
SA FX	0 ... 255
SPA PB	0 ... 255
SPA OB [1])	40 ... 255
SPB PB	0 ... 255
SPB PB	0 ... 255
SAB FX	0 ... 255
SPB S3	0 ... 255
SPB OB	1 ... 39
SPB OB [1])	40 ... 255
A DE	3 ... 255
	... 26
SEA	

о Операции загрузки :	
L =	Форм.операнд
LC =	Форм.операнд
LW =	Форм.операнд
LD =	Форм.операнд
L BS	0 ... 255
LIR [*])	0 ... 15

[1]) Вызов спецфункции

[*]) Системная операция

Операция	Параметр
o Операции загрузки (продолжение):	
LD	-32768...+32767
LS GD	-32768...+32767
LW GW	-32768...+32767
LW GD	-32768...+32767
LB GB	-32768...+32767
LB GW	-32768...+32767
o Операции передачи:	
I =	Форм. операнд
T BA	0 ... 255
TBP B)	0 ... 15
TBW	-32768...+32767
TBD	-32768...+32767
TBQ	-32768...+32767
TB GB	-32768...+32767
TB GW	-32768...+32767
TBC	-32768...+32767
TB GD	-32768...+32767
TW GW	-32768...+32767
TW GD	-32768...+32767
o Двоичные логические операции:	
O =	Форм.операнд
ON =	Форм.операнд
o Операции преобразования	
XEW	
XEW	
XEQ	
DEF	
DUF	

Операция	Параметр
o Операции преобразования (продолжение):	
DUD	
ETR	
o Функции сдвига	
SLW	0 ... 15
SRW	0 ... 15
RRD	0 ... 32
SVW	0 ... 15
o Функции перехода	
SPA =	Символич.адрес
SPP =	Символич.адрес
SPM =	Символич.адрес
SPS =	Символич.адрес
SPP	-32768...+32767
o Операции с регистром статуса:	
MAB	
MBA	
MBS	
o Операции установки:	
S =	Форм.операнд
o Другие функции	
ACR	
AF	
AS	
ENT	
O	0 ... 255
I	0 ... 255
B =	Форм.операнд

операция	Параметр
o Другие функции (продолжение):	
B DW B MW BI 2) B BS TAK BLD E DB EX DX SES SEF	0. . . 255 0. . . 255 0. . . 255 0. . . 255 0. . . 255 0. . . 255 0. . . 31 0. . . 31
o Арифметические операции	
ADD BF ADD KF ADD DF 2) +D 2) -D 2)	-127. . . +127 -32768 . . . +32767 -2147483648 . . . . . . +2147483647

2) Системные операции

Приложение D: Обзор команд STEP5 (в алфавитном порядке).
Команды, помеченные (*), принадлежат к дополнительному набору команд и действительны только в функциональных блоках (FB/FX)

Команда STEP5		Группа команд
<=D		Операция сравнения
<=F		"
<=G		"
<>		Двоичная логич.опер.
-D	*	Системная операция!
-F		Арифметич.операция
-G		"
-D	*	Системная операция!
-F		Арифметич.операция
-G		"
<F		"
<G		"
<F		"
<G		"
<=D		Операция сравнения
<=F		"
<=G		"
<D		"
<F		"
<G		"
==	*	Операция установки
=A		Операция запоминания
=D		"
=E		"
=M		"
><D		Операция сравнения
><F		"
><G		"
>=D		"
>=F		"
>=G		"
>D		"
>F		"
>G		"
ABR	*	Системная операция!
ACR	*	Системная операция!
A DB		Вызов блока данных
ADD BF	*	Арифметич.операция
ADD DF	*	"
ADD KF	*	"
AF	*	Специальная операция
AS	*	"
AX DX		Вызов блока данных
BA FX		"
BAB FX		"
B=	*	Специальная операция
B BS	*	Системная операция!
B DW	*	Специальная операция

Команда STEP5		Группа команд
BE		Конец блока
BEA		"
BEB		"
BI	*	Системная операция!
BLD		Специальная операция
B MW	*	"
D	*	Декрементирование
DED	*	Операция преобразов.
DEF	*	"
DUD	*	"
DUF	*	"
E DB		Создание DB
ENT	*	
EX DX		Специальная операция
FDG	*	Создание DX
FR=	*	Операция преобразов.
FR T	*	Операция таймера
FR Z	*	Операция счетчика
GFD	*	Операция преобразов.
I	*	Инкрементирование
KEW	*	Операция преобразов.
KID	*	"
KZW	*	"
L AB		Операция загрузки
L AW		"
L BA	*	"
L BB	*	"
L BS	*	"
L BT	*	"
LB CB	*	Системная операция!
LB CD	*	"
LB CW	*	"
LB GB	*	"
LB GD	*	"
LB GW	*	"
LC=		Операция загрузки
LC T		"
LC Z		"
LD=	*	"
L DD		"
L DL		"
L DR		"
L DW		"
L EB		"
L ED		"
L EW		"
LIR	*	Системная операция !
L KB		Операция загрузки

Команда STEP5		Группа команд
L KF		"
L KG		"
L KM		"
L KM		"
L KT		"
L KY		"
L KZ		"
L MB		"
L MD		"
L MW		"
L PY		"
L PW		"
L QB		"
L QW		"
LFD	*	Системная операция!
LRW	*	Системная операция!
L T		Операция загрузки
LW=	*	"
LW CD	*	Системная операция!
LW CW	*	"
LW GD	*	"
LW GW	*	"
L Z		Операция загрузки
L=	*	"
MAB	*	Системная операция!
MAS	*	"
MBA	*	"
MBR	*	"
MBS	*	"
MSA	*	"
MSB	*	"
NOP 0		Нулевая операция
NOP 1		"
O		Двоичная логич.опер.
O(		"
O=	*	"
O A		"
O D		"
O E		"
O M		"
ON=	*	"
ON A		"
ON D		"
ON E		"
ON M		"
ON T		"
ON Z		"
O T		"
OW	*	Цифровая логич.опер.
OZ		Двоичная логич.опер.
RA		Операция запоминания
RB=	*	Операция/установки
RD		Операция запоминания
RD=	*	Таймер/счетчик опер.
RE		Операция запоминания

Команда STEP5		Группа команд
RLD	*	Функция сдвига
R M		Операция запоминания
RRD	*	Функция сдвига
RT		Операция таймера
RZ		Операция счетчика
S=	*	Операция установки
S A		Операция запоминания
SAR=	*	Таймер/счетчик опер.
SA T		Операция таймера
S D		Операция запоминания
S E		"
SEF	*	Специальная операция
SES	*	"
SS =	*	Операция таймера
SE T		"
SI =	*	"
SI T		"
SLD	*	Функция сдвига
SLW	*	"
S M		Операция запоминания
SPA=	*	Операция перехода
SPA FB		Вызов блока
SPA CB		Вызов блока+переход
SPA PB		Вызов блока
SPA SS		"
SPB =	*	Операция перехода
SPB FB		Вызов блока
SPB CB		Вызов блока+переход
SPB PB		Вызов блока
SPB SS		"
SPM =	*	Операция перехода
SPN =	*	"
SPO =	*	"
SPP =	*	"
SPR =	*	Системная операция !
SPS =	*	Операция перехода
SPZ =	*	"
SRW	*	Функция сдвига
SS T		Операция таймера
SSV =	*	Таймер/счетчик опер.
STP		Команда останова
SVD	*	Функция сдвига
SV T		Операция таймера
SVW	*	Функция сдвига
SVZ =	*	Таймер/счетчик опер.
S Z		Операция счетчика
T =	*	Операция передачи
T AB		"
T AD		"
TAK	*	Специальная операция
T AW		Операция передачи
T BA		"
T BB		"
TB CB	*	Системная операция!
TB CD	*	"

Команда STEP5	Группа команд
TB CW *	“
TB GB *	“
TB GD *	“
TB GW *	“
T BS *	Операция передачи
T BT *	“
T DD *	“
T DL	“
T DR	“
T DW	“
T ED	“
T EW	“
TIR *	системная операция
T MB	операция передачи
T MD	“
T MW	“
TNB *	системная операция
TNW *	системная операция
T PB/PY	операция передачи
T PW	“
T QB	“
T QW	“
TRD *	системная операция
TRW *	“
TSC *	“
TSG *	“
U (	двоичная логич. Операция!
U= *	“
U A	“
U D	“
U E	“
U M	“
UN= *	“
UN A	“
UN D	“
UN E	“
UN M	“
UN T	“
UN Z	“
U T	двоичная логич. Операция!
U W *	Цифровая логич. операция
U Z	двоичная логич. Операция!
XOW *	Цифровая логич. Операция
Z RZ	Операция счетчика
Z VZ	“

Приложение E: Команды STEP5 CPU928, отсортированные по кодам команд.

Пояснения:

- Столбец «код команды» :

  Код команды состоит максимально из тех слов (максимум 48 бит), которые представлены либо в виде шестнадцатеричных кодов, либо – для небольшого числа команд – в виде набора битов. Каждый бит соответственной из следующих областей:
  «код операции» (определяет вид команды)
  «параметр» (определяет с чем работает команда)
  «не влияет» (не был декодирован)

  Позиции битов, которые содержат параметр, заменены буквой «P» (первый параметр, например адрес байта) и «Q» (второй параметр, например адрес бита).

  Позиции бита, которые не кодируются, помечаются буквами «x».

- Колонка «область параметров»:

  Содержит допустимую область значений, которые могут указываться на месте букв «P» или «Q». Все значения указаны в десятичном виде.

- Столбец «STEP5»:
  Содержит предусмотренные для программирования в виде AWL мнемонические выражения команд (мнемоника STEP5).

- Колонка «примечание»:
  Изменения по сравнению с R- процессором:

  P - увеличенная на 128 область счетчиков \таймеров
  N - новая команда

Код команды			Область параметров	STEPS	Примечание
Слово					
---1	---2	---3			
00xx				NOP 0	
0100				KEW	
02pp			0-255	LT	P
03pp			0-255	TNB	
04pp			0-255	FRT	P
0500				BEB	
06pp			1-126	FR=	
07pp			1-126	U=	
0800				AS	
0880				AF	
0900				KZW	
0App			0-255	LMB	
0Bpp			0-255	TMB	
0Cpp			0-255	LCT	
0Dpp			-128, +127	SPO=	
0Epp			1-126	LC=	
0Fpp			1-126	0=	
10xx				BLD	
11pp			0-255	I	
12pp			0-254	LMW	
13pp			0-254	TMW	
14pp			0-255	SAT	P
15pp			-128, +127	SPP=	
16pp			1-126	SAR=	
17pp			1-126	S=	
18pp			0-255	BBS	
19pp			0-255	D	
1App			0-255	LMD	
1Bpp			0-252	TMD	
1Cpp			0-255	SVT	P
1Dpp			0-255	SPB FB	
1Epp			1-126	SVZ=	P
1Fpp			1-126	= =	
20pp			3-255	ADB	
2120				>F	
2140				<F	
2160				><F	
2180				! = F	
21A0				> = F	
21C0				< = F	
22pp			0-255	LDL	
23pp			0-255	TDL	
24pp			0-255	SET	P

Код команды			Область параметров	STEPS	Примечание
Слово					
---1	---2	---3			
25pp			-128, +127	SPM=	
26pp			1-126	SE=	
27pp			1-126	UN=	
28pp			0-255	LKB	
29pp			0-32	SLD	
2App			0-255	LDR	
2Bpp			0-255	TDR	
2Cpp			0-255	SST	P
2Dpp			-128, +127	SPA=	
2Epp			1-126	SSV=	
2Fpp			1-126	0N=	
3001	pppp		0-65535	LKZ	
3002	pppp		0-65535	LKT	
3004	pppp		0-65535	LKF	
3010	pppp		0-65535	LKC	
3020	pppp		0-65535	LKY	
3040	pppp		0-65535	LKH	
3080	pppp		0-65535	LKM	
3120				> G	
3140				< G	
3160				>< G	
3180				! = G	
31A0				> = G	
31C0				< = G	
32pp			0-255	LDW	
33pp			0-255	TDW	
34pp			0-255	SIT	P
35pp			-128, +127	SPN=	
36pp			1-126	SI=	
37pp			1-126	RB=	
3800	pppp	pppp	0-4294967295	LKG	
3920				> D	
3940				< D	
3960				>< D	
3980				! = D	
39A0				> = D	
39C0				< = D	
3App			0-254	LDD	
3Bpp			0-254	TDD	
3Cpp			0-255	RT	P
3Dpp			0-255	SPA FB	
3Epp			1-126	RD=	

Код команды			Область параметров	STEPS	Примечание
---1	---2	---3			
3Fpp			1-126	LW=	
40pp			0-15	LIR	
41pp				UW	
42pp			0-255	LZ	P
43pp			0-255	TNW	
44pp			0-255	FRZ	P
4500			-128, +127	SPZ=	
46pp			1-126	L=	
47pp			0-255	LBB	N
48pp			0-15	TIR	
4900				QW	

Битовое представление (слово 1)

5432	1098	7654	3210		
0100	1010	0ppp	pppp	0-127	LEB
0100	1010	1ppp	pppp	0-127	LAB
0100	1011	0ppp	pppp	0-127	TEB
0100	1011	1ppp	pppp	0-127	TAB

Слово

---1	---2	---3			
4Cpp			0-255	LCZ	P
4Dpp			1-255	SPB OB	
4Epp			0-255	BMW	
4Fpp			0-255	LBT	N
50pp			-128, +127	ADD BF	
5100				XOW	

Битовое представление (слово 1)

5432	1098	7654	3210		
0101	1010	0ppp	pppp	0-126	LEW
0101	1010	1ppp	pppp	0-126	LAW
0101	1011	0ppp	pppp	0-126	TEW
0101	1011	1ppp	pppp	0-126	TAW

Слово

---1	---2	---3	
54pp			0-255
55pp			0-255
56pp			1-125
57pp			0-254
5800	pppp		-32768, +32767
5900			

Код команды				Область параметров	STEPS	Примечание
Битовое представление (слово 1)						
5432	1098	7654	3210			
0101	1010	0ppp	pppp	0-124	LED	
0101	1010	1ppp	pppp	0-124	LAD	
0101	1011	0ppp	pppp	0-124	TED	
0101	1011	1ppp	pppp	0-124	TAD	
Слово						
---1	---2	---3				
5Cpp				0-255	SZ	P
5Dpp				0-255	SPB SB	
5Fpp				0-255	LQB	
6000					:F	
6003					:G	
6004					xF	
6005	pppp	pppp		-2147483648, +2147483647	ADD DF	N
6007					xG	
6008					ENT	
6009					- D	N
600B					- G	
600C	xxpp			-128, +127	SPS=	
600D					+ D	N
600F					+ G	
61pp				0-15	SLW	
62pp				0-255	LBS	
63pp				0-255	TBS	
64pp				0-32	RLD	
65pp					BE	
6501					BEA	
66pp				1-126	T=	
67pp				0-255	TBB	N
6800	pppp			-32768, +32767	LRW	N
68p1				0-15	SVW	
6802					GFD	
6803	pppp			-32768, +32767	TRW	N
6804	pppp			-32768, +32767	LRD	N
6805	pppp			-32768, +32767	TRW	N
6806					FDQ	
6807					KZD	
6808					DUF	
680A					DUD	
680C					DEF	
6819					MAS	N

Код команды			Область параметров	STEPS	Примечание
Слово					
---1	---2	---3			
6829				MAB	N
6849				MSA	N
6869				MSB	N
6889				MBA	N
6899				MBS	N
69pp			0-15	SRW	
69pp			0-255	LBA	
6App			0-255	TBA	
6Cpp			0-255	ZVZ	P
6Dpp			0-255	SPA OB	
6Epp			0-255	BDW	
6Fpp			0-255	TBT	N
7002				TAK	
7003				STP	
700B	pppp		-32768, +32767	SPR	
71pp			0-32	SVD	
72pp			0-255	LPY	
73pp			0-255	TPY	
74pp			0-32	RRD	
75pp			0-255	SPA PB	
76pp			1-125	B=	
77pp			0-254	TOW	
7801	xxpp		0-255	BA FX	
7802	xxpp		0-255	BAB FX	
7803	xxpp		0-255	AX DX	
7804	xxpp		0-255	EX DX	
7805	xxpp		0-255	E DB	
7806	xxpp		0-31	SES	
7807	xxpp		0-31	SEF	
78p9	pppp		-32768, +32767	MBR	N
780A	pppp		-32768, +32767	ABR	N
780D	pppp		-32768, +32767	LBCB	N
780E	pppp		-32768, +32767	LBGB	N
781D	pppp		-32768, +32767	LBCW	N
781E	pppp		-32768, +32767	LBGW	N
782D	pppp		-32768, +32767	LBCD	N
782E	pppp		-32768, +32767	LBGD	N
783D				ACR	N
783F	Qqpp		0.0 – 255.15	UD	
783F	1qpp		0.0 – 255.15	OD	
783F	2qpp		0.0 – 255.15	UND	
783F	3qpp		0.0 – 255.15	OND	

Код команды			Область параметров	STEPS	Примечание
Слово					
---1	---2	---3			
783F	4qpp		0.0 – 255.15	SD	
783F	5qpp		0.0 – 255.15	RD	
783F	6qpp		0.0 – 255.15	= D	
785D	pppp		-32768, +32767	LWCW	N
785E	pppp		-32768, +32767	LWGW	N
786D	pppp		-32768, +32767	LWCD	N
786E	pppp		-32768, +32767	LWGD	N
788D	pppp		-32768, +32767	TBCB	N
788E	pppp		-32768, +32767	TBGB	N
789D	pppp		-32768, +32767	TBCW	N
789E	pppp		-32768, +32767	TBGW	N
78AD	pppp		-32768, +32767	TBCD	N
78AE	pppp		-32768, +32767	TBGD	N
78CD	pppp		-32768, +32767	TSC	N
78CE	pppp		-32768, +32767	TSC	N
78DD	pppp		-32768, +32767	TWCW	N
78DE	pppp		-32768, +32767	TWGW	N
78ED	pppp		-32768, +32767	TWCD	N
78EE	pppp		-32768, +32767	TWGD	N
7900				+ F	
7App			0-254	LPW	
7Bpp			0-254	TPW	
7Cpp			0-254	RZ	
7Dpp			0-254	SPA SB	
7Epp				BI	
7Fpp			0-255	TQB	

Битовое представление (слово 1)

5432	1098	7654	3210		
1000	0qqq	pppp	pppp	0.0 – 255.7	UM
1000	1qqq	pppp	pppp	0.0 – 255.7	OM
1001	0qqq	pppp	pppp	0.0 – 255.7	SM
1001	1qqq	pppp	pppp	0.0 – 255.7	= M
1010	0qqq	pppp	pppp	0.0 – 255.7	UNM
1010	1qqq	pppp	pppp	0.0 – 255.7	ONM
1011	0qqq	pppp	pppp	0.0 – 255.7	RM

Слово					
---1	---2	---3			
B8pp			0-255	UZ	P
B9pp			0-255	OZ	P
Bapp				U (	

Код команды			Область параметров	STEPS	Примечание
Слово					
---1	---2	---3			
BBpp				O (	
BCpp			0-255	UNZ	P
BDpp				ONZ	P
BEpp				BAS 1)	
BFpp				)	

Битовое представление (слово 1)

5432	1098	7654	3210		
1100	0qqq	0ppp	pppp	0.0 – 127.7	UE
1100	0qqq	1ppp	pppp	0.0 – 127.7	UA
1100	1qqq	0ppp	pppp	0.0 – 127.7	OE
1100	1qqq	1ppp	pppp	0.0 – 127.7	OA
1101	0qqq	0ppp	pppp	0.0 – 127.7	SE
1101	0qqq	1ppp	pppp	0.0 – 127.7	SA
1101	1qqq	0ppp	pppp	0.0 – 127.7	= E
1101	1qqq	1ppp	pppp	0.0 – 127.7	= A

Код команды			Область параметров	STEPS	Примечание
Битовое представление (слово 1)					
5432	1098	7654	3210	0.0 – 127.7	
1100	0qqq	0ppp	pppp	0.0 – 127.7	UNE
1100	0qqq	1ppp	pppp	0.0 – 127.7	UNA
1100	1qqq	0ppp	pppp	0.0 – 127.7	ONE
1100	1qqq	1ppp	pppp	0.0 – 127.7	ONA
1111	0qqq	0ppp	pppp	0.0 – 127.7	RE
1111	0qqq	1ppp	pppp	0.0 – 127.7	RA

Слово					
---1	---2	---3			
F8pp			0-255	UT	P
F9pp			0-255	OT	P
FApp			-128,+127	SPB=	
FBxx				O	
FCpp			0-255	UNT	P
FDpp			0-255	ONT	P
FFxx				NOP 1	

Приложение F: Команды STEP-5, которых нет в CPU928.

Обратите внимание на то, что следующие команды STEP-5 контроллера S5-150U не выполняются в CPU928:

BAS		Блокировка выдачи команд
BAF		Деблокировка выдачи команд
P	E,A,M,Y,Z,D,BA,BB,BS,BT	Проверка бита на «1»
PN	E,A,M,Y,Z,D,BA,BB,BS,BT	Проверка бита на «0»
SU	E,A,M,Y,Z,D,BA,BB,BS,BT	Безусловная установка бита
RU	E,A,M,Y,Z,D,BA,BB,BS,BT	Безусловный сброс бита
LIM		Загрузка маски прерываний
SIM		Установка маски прерываний
UBE		Конец блока прерываний
STM		Команда останова обработки прерываний по времени
AFS		Блокировка прерываний из-за ошибки адресации
AFF		Деблокировка прерываний из-за ошибки адресации
AAF		Деблокировка обработки запроса на прерывание
AAS		Блокировка обработки запроса на прерывание

Приложение G: Обзор меток уровней обработки программ

Метки соответствуют меткам (шестнадцатеричным) указанным в USNFCK после пометки УРОВЕНЬ

0002H	Новый пуск
0004H	Цикл
0006H	Прерывание по времени 5 сек
0008H	Прерывание по времени 2 сек
000AH	Прерывание по времени 1 сек
000CH	Прерывание по времени 500 мс
000EH	Прерывание по времени 200 мс
0010H	Прерывание по времени 100 мс
0012H	Прерывание по времени 50 мс
0014H	Прерывание по времени 20 мс
0016H	Прерывание по времени 10 мс
0018H	Не используется
001AH	Не используется
001CH	Регулятор
001EH	Не используется
0020H	Не используется
0022H	Не используется
0024H	Аппаратное прерывание
0026H	Не используется
0028H	Не используется
002AH	Не используется
002CH	Прерывание
002EH	Не используется
0030H	Ошибка «будильника»
0032H	Ошибка регулятора
0034H	Ошибка цикла
0036H	Не используется
0038H	Ошибка кода команды
003AH	Ошибка выполнения
003CH	Ошибка адресации
003EH	Ошибка квитирования
0040H	Не используется
0042H	Не используется
0044H	Ручной перезапуск
0046H	Автоматический перезапуск

Приложение H: Пример использования стека прерываний

Данный (сильно упрощенный) пример покажет Вам возможный образ действий при пользовании USTACKом.
Дополнительно к вышесказанному смотрите также раздел 5.3 «Управляющие биты и стек прерываний»!

Исходная точка: Процессор прервал циклическую обработку программы и перешел в состояние останова.
Для того чтобы определить причину, вызовите на экран программатора On-line- функцию «Вызов USTACK».

Сначала на PG появятся управляющие биты:

>>STR<<	STR-6	FE-STR	BARBEND	PG-STP	STP-SCH	STP-BEF	MP-STP
X						X	
>>ANL<<	ANL-6	NEUST	MWA	AWA	ANL-2	NEU-ZUL	MWA-ZUL
						X	X
>>RUN<<	RUN-6	EINPROZ	BARB	OB1GEL	FBOGEL	OBPROZA	OBWECKA
X		X		X			
32KRAM	16KWRAM	SKWRAM	EPROM	KM-AUS	KM-EIN	DIG-EIN	DIG-AUS
	X					X	X
URGELOE	URL-IA	STR-VER	ANL-ABB	UA-PG	UA-SYS	UA-PRFE	UA-SCH
DX0-FE	FE-22	MOD-FE	RAM-FE	DB0-FE	DB1-FE	DB2-FE	KOR-FE
NAU	PEU	BAU	STUE-FE	ZYK	QVZ	ADF	WECK-FE
BCF	FE-6	FE-5	FE-4	FE-3	LZF	REG-FE	DOPP-FE
					X		

В Управляющих битах описывается текущий режим работы процессора (>>STR<<), и отмечает определенные свойства процессора (OB 1 загружен, однопроцессорный режим, память пользователя 16 К Слов и т.д.). В верхней строке в качестве причины останова помечено крестиком сокращение “STP-BEF". Так как мы в нашем случае можем исключить команду STP в программе пользователя на языке STEP5, то остается под вопросом только команда останова от операционной системы вследствие отсутствия OB ошибки. В нижней строке крестиком помечено сокращение “LZF”. Может быть операционная система при возникновении ошибки выполнения установила, что соответствующий организационный блок ошибки не запрограммирован. Однако, так как существуют различные ошибки выполнения и мы не знаем какая из них явилась причиной, информации содержащейся в управляющих битах не достаточно.

Мы вызываем USTACK:

СТЕК ПРЕРЫВАНИЙ

ГЛУБИНА: 01

BEF-REG:	0000	SAZ:	0000	DB-ADR:	0000	BA-ADR:	0000
BST-STR:	0001	-NR:		DB-NR:		-NR:	
		REL-SAZ:		DBL-REG:	0000		
УРОВЕНЬ:	003A	UAMK:	0120	UALW:	0000		

AKKU1:0000 1A01 AKKU2: 0000 0000 AKKU3: 0000 0000 AKKU4: 0000 0000

БИТЫ РЕЗУЛЬТАТА:	ANZ1	ANZ0	OVEL	OVFLS	ODER	STATUS	VKE	ERTAB
ПРИЧИНА ОШИБКИ:	NAU	PEU	BAU	HALT	ZYK	QVZ	ADF	STR X
	BCF	S-6	LZF	REG	STUEB	STUEU	WECK	DOPP

USTACK глубины 01 представляет уровень обработки программ, который был активен непосредственно перед переходом в СТОП. По метке «3A»H (после пометки УРОВЕНЬ) мы видим, что это USTACK уровня обработки программ ОШИБКА ВЫПОЛНЕНИЯ. В аккумуляторе 1 находится метка ошибки «1A01»H. Из нее мы узнаем, что ошибка выполнения возникла из=за незагруженного блока данных по команде «A DB». Так как соответствующий OB 19 ошибки в нашей программе пользователя отсутствует, то обработку программы прервала операционная система (STP). В слове маски индикации прерываний UAMK также хранится информация о причине прерывания: метка «0120»H соответствует набору битов «0000 0001 0010 0000». Бит 2? (LZF) и бит 2? (STP) установлены.

Теперь действительно ясно, в каком блоке и из-за какой команды возникла ошибка выполнения.

При повторном квитировании на PG появляется Глубина 02 USTACKa:

ГЛУБИНА: 02

BEF-REG:	2006	SAZ:	0037	DB-ADR:	0000	BA-ADR:	0000
BST-STR:	0001	OB-NR:	1	DB-NR:		-NR:	
		REL-SAZ:	0004	DBL-REG:	0000		
УРОВЕНЬ:	0004	UAMK:	0020	UALW:	0000		

AKKU1:0000 1A01 AKKU2: 0000 0000 AKKU3: 0000 0000 AKKU4: 0000 0000

БИТЫ РЕЗУЛЬТАТА:	ANZ1	ANZ0	OVEL	OVFLS	ODER	STATUS	VKE	ERTAB
ПРИЧИНА ОШИБКИ:	NAU	PEU	BAU	HALT	ZYK	QVZ	ADF	STR
	BCF	S-6	LZF X	REG	STUEB	STUEU	WECK	DOPP

По метке «04»H (после пометки УРОВЕНЬ) мы видим, что это USTACK уровня обработки программ ЦИКЛ. Счетчик адресов команд (SAZ) содержит адрес “37”H. По этому абсолютному адресу в памяти пользователя расположена команда, явившаяся причиной ошибки. Прерывание возникло в организационном блоке OB 1. Внутри OB 1 команда вызвавшая ошибку расположена по относительному адресу «04»H (REL-SAZ). Эта команда привела, как мы установили выше, к ошибке выполнения (см. UAMK, бит 2 , и ПРИЧИНУ ОШИБКИ).

Вызовем теперь на экран программатора неверную команду с помощью On-line-функции «ПОИСК». Мы указываем соответствующий блок (OB1) и относительный адрес команды.

F1	F2	F3	F4	F5	F6	F7
SYMB.ANZ.	BIB.NR.	SUCHLAUF				

ВЫВОД НА ПРИБОР: AG БЛОК: OB1 ПОИСК: 4H

На PG мы видим, после успешного поиска, команду “A DB 6», которая несет ответственность за прерывание, так как блок данных с номером 6 в памяти пользователя отсутствует.

```
OB 1
NETZWERK 1        0000
0004              :A      DB  6
0005              :
0006              :
0007              :
0008              :BE
```

Перечень рисунков и надписей

Рисунок: Типовая структура устройства AG S5-135U
Рисунок: Виды представления языка программирования STEP5
Рисунок: Расположение блоков в памяти программ
Рисунок: Структура организационных, программных и шаговых блоков
Рисунок: Вызов блоков
Обзор: Организационные блоки специальных функций
Рисунок: Структура функционального блока (FB/FX)
Пример: Программирование функционального блока
Пример: Вызов и параметрирование функционального блока
Рисунок: Вызов и параметрирование функционального блока
Рисунок: Структура блоков данных
Пример: Передача слов данных
Рисунок: Вызов блока данных
Пример: Область действия блоков данных
Рисунок: Область действия вызванного блока данных
Пример: Организация программы пользователя по структуре устройств
Пример: Организация программы пользователя по структуре устройств
Пример: Глубина вложения блоков и стек блоков
Рисунок: Размещение блоков в памяти программ
Рисунок: Циклическая обработка программ
Обзор: Рабочее состояние и уровни обработки программ
Рисунок: Обработка программы, управляемой по прерываниям
Обзор: Встроенные специальные функции в CPU928
Рисунок: Эскиз принципа работы регистра сдвига с 3 указателями
Рисунок: Эскиз принципа работы регистра сдвига с 3 указателями
Рисунок: Эскиз принципа работы регистра сдвига с 3 указателями
Рисунок: Структура блока данных PID – регулятора
Рисунок: Блок – схема PID – регулятора
Рисунок: Структура DX0
Пример: Параметрирование DX0
Пример: Параметрирование DX0
Рисунок: Распределение адресного пространства в CPU928
Рисунок: Распределение адресного пространства: Системная RAM (16 бит)
Рисунок: Распределение адресного пространства: Периферия (8 бит)
Рисунок: Распределение адресного пространства для периферии / программирования
Пример: Слова данных DB/DX описанные с помощью констант
Пример: Стирание всех байтов маркеров (MB0…MB255)

Таблица: Действие точек контроля системы и пользователя
Обзор: Сообщений об ошибках в аккумуляторах АКК 1 и АКК 2
Обзор: Сообщений об ошибках в BS3 и BS4