_____

Московский Государственный Технический Университет Гражданской Авиации

Кафедра вычислительных машин, комплексов, систем и сетей

Курсовой проект
защищен с оценкой

_____

_____
(подпись руководителя, дата)

Курсовая работа по дисциплине “Схемотехника”

Тема: “ Специализированный процессор ”

Выполнил студент
группы ЭВМ 3-1
Иванов В.И.
(ф.и.о.)
руководитель:
Шапкин Ю.А.

Москва 2008г.

ЗАДАНИЕ НА КУРСОВОЕ ПРОЕКТИРОВАНИЕ:

Задание № 1-16

студенту Нарушевичу В.И. группы ЭВМ 3-1 на выполнение курсового проекта по дисциплине “Схемотехника ЭВМ”.

1.Тема курсового проекта “Специализированный процессор”.
2.Исходные данные к проекту:
2.1 Команды МП 181.

Команды:
Команда останова (F4)
Команда вычитания (FE)
Команда условного перехода (71)

2.2Технические характеристики процессора:

-оперативная память ёмкостью 512 кбайт, шириной выборки 4 байта, с циклом времени обращения 0,1 мкс;

-АЛУ с закреплёнными микрооперациями;

-УУ с жёсткой логикой.

2.3. Максимально допустимое время выполнения коротких операций - 8 мкс, длинных – 40 мкс.

3.Перечень подлежащих разработке вопросов:

3.1Расчётная (логическая) часть:

-синтез операционного и управляющего автоматов процессора;

-расчёт рабочего такта и времени выполнения операций.

3.2Графическая часть:

-функциональная электрическая схема процессора – 1 л.;

-принципиальная электрическая схема блока управления операциями - 0,5 ÷ 1л.

Содержание.

АННОТАЦИЯ

Цель и задачи проектирования:

Целью курсового проектирования является освоение методики проектирования и разработка операционных и управляющих устройств ЭВМ и выпуска технической документации на эти устройства.

Основным содержанием проекта является синтез структуры и проектирование операционных и управляющих устройств процессора. Процессор проектируется для четырёх команд из списка команд микро-ЭВМ, реализованных на базе микропроцессора типа К1810ВМ86. Разработке подлежит процессор в целом, на который составляются схемы структурных алгоритмов операций и функциональная схема. Синтез структуры процессора производится по объединенной ГСА.

По объединённой ГСА производится расчет количества и допустимой длительности тактов для каждой операции. Синтезируются логические схемы операционного и управляющего автоматов. Составляются принципиальные схемы операционного и управляющего автоматов или их отдельных блоков. Рассчитываются время выполнения операций в спроектированном процессоре.

СТРУКТУРА МИКРОПРОЦЕССОРА К1810ВМ86

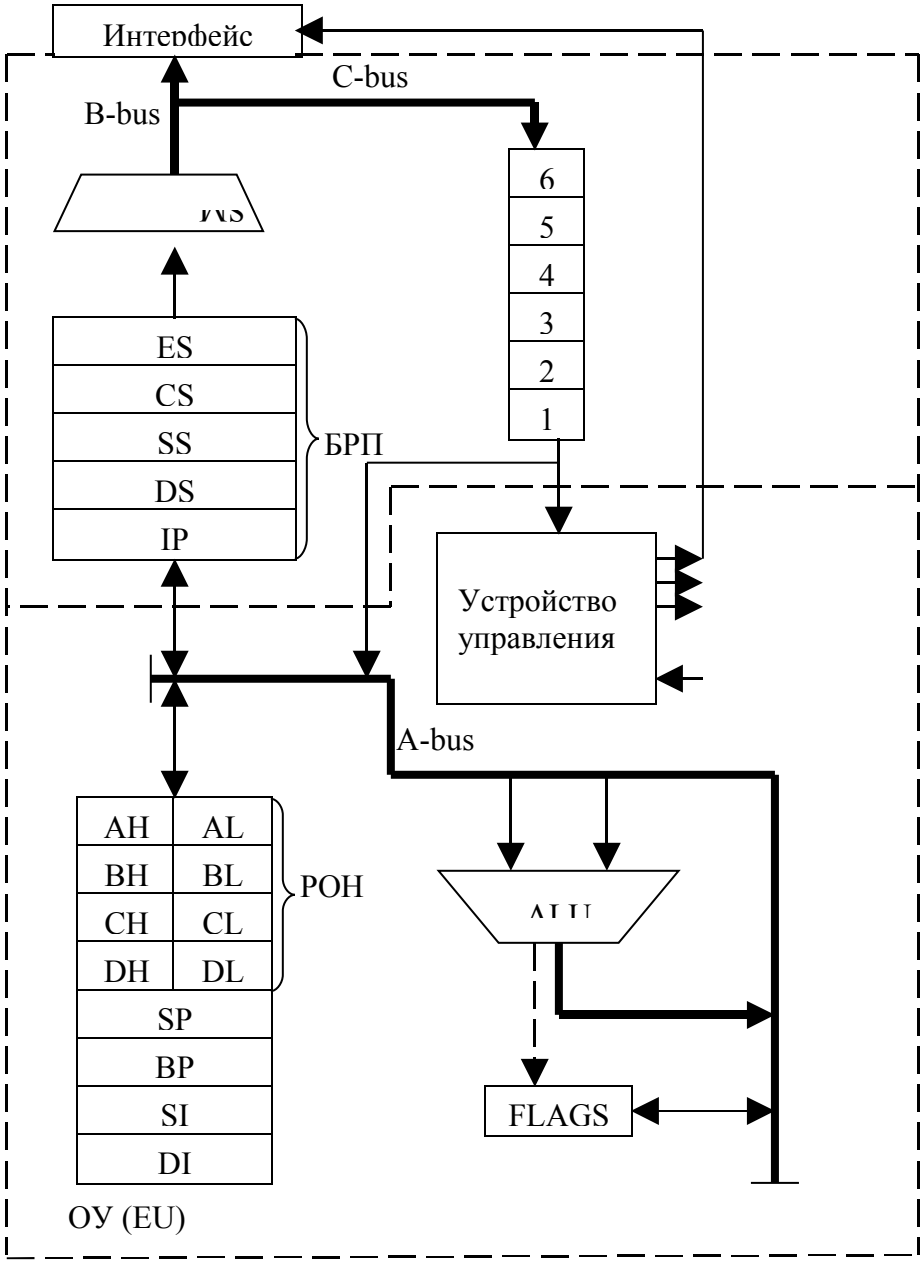


Рис. 1

Микропроцессор третьего поколения к1810 – однокристальный 16-ти разрядный центральный процессорный элемент, выполненный на кристалле размером 5,5 х 5,5 мм^2. Микропроцессор реализован по варианту схемотехнологии n-МОП (high quality MOS). Эта технология характеризуется малыми размерами элементов и самосовмещеными поликремниевыми затворами транзисторов. Микропроцессор содержит около 29 000 транзисторов и однофазную систему тактовых импульсов частоты 5 МГц (для выборочной версии 8 МГц). По напряжению питания (один источник) 5В микропроцессор совместим с элементами ТТЛ, он размещён в корпусе с 40 выводами. Для достижения номинальной производительности необходима память с циклом 500-800 нс. и временем обращения при считывании 290-460 нс. Все регистры и двунаправленная мультиплексная шина AD адреса и данных 16-ти битные. Старшие 4 бита адреса мультиплексируются с сигналами состояния. Следовательно, длина физического адреса памяти составляет 20 бит, что обеспечивает адресное

пространство 1 Мбайт. Микропроцессор прямо адресует 256 портов ввода и 256 портов вывода и косвенно – 64 К 8-битных портов ввода-вывода.

В архитектуре микропроцессора видна тенденция сохранения программной совместимости с 8-ми битным микропроцессором КР580.

Объём непосредственно адресуемой памяти составляет 2^{20} байт, т. е. 1 Мбайт, что обеспечивает применение 20-ти разрядной шины. Отдельное адресное пространство для УВВ обеспечивает адресацию 64 К внешних устройств.

Микропроцессор имеет два режима работы - минимальную и максимальную конфигурацию. В первом режиме сам микропроцессор вырабатывает сигналы управления шинами, необходимые для модулей памяти и УВВ. Во втором режиме в систему вводится специальный модуль 8288, который берет на себя управление модулями, подключенными к системным шинам, а микропроцессор вырабатывает лишь сигналы, необходимый для управления модулем 8288. микропроцессор К1810 по производительности в среднем в десять раз превосходит своего предшественника – микропроцессор КР580ИК80А (I8080), находясь на уровне средней мини-ЭВМ.

Для оценки качества микропроцессора иногда пользуются критерием:

$$A = \frac{\text{длина слова x число команд}}{\text{площадь кристалла x мощность x цикл команды}}.$$

В этот критерий наряду с характеристиками, непосредственно влияющие на характеристики системы, введена и площадь кристалла, играющая роль коэффициента нормализации. По этому критерию микропроцессор К1810 превосходит микропроцессор КР580 в 50 раз.

Микропроцессор делится на 2 части – шинный интерфейс ШИ (BIU, bus interface unit) и операционное устройство ОУ (EU, execution unit). Блок шинного интерфейса содержит шину С (C-bus), устройство очереди команд УОК, блок сегментных регистров (регистров переадресации, БРП), указатель команд IP (instruction pointer) и сумматор адресов SM.

В операционное устройство входит блок регистров общего назначения РОН (GRF, general register file), арифметико-логическое устройство АЛУ (ALU), регистр признаков (флажков) и устройство управления УУ. В АЛУ имеются 2 внутренних регистра хранения операндов.

Блок сегментных регистров совместно с сумматором реализуют сегментацию памяти. При этом адресное пространство делится на сегменты, начальные адреса которых кратны 16-ти, т. е. Имеют нули в 4-х младших разрядах. Других ограничений на начальный адрес сегмента не накладывается. Нулевые разряды начального адреса могут быть подразумеваемыми и их можно не передавать по шине. Тогда начальный адрес можно задать 16-ти разрядным словом и число сегментов составит 2^{16} = 64 К. если номер ячейки памяти внутри сегмента (адрес смещения) так4же указывать 16-ти разрядным словом, то объём сегмента составит 64Кбайт (при байтовой организации памяти). Из сказанного ясно, что сегменты могут перекрываться (при отсутствии перекрытия в памяти ёмкостью 1 Мбайт разместилось только 16 сегментов). При сегментации памяти физический (исполнительный) адрес ячейки, имеющий 20 разрядов, “конструируются” из 2-х 16-ти разрядных слов (см. рис. 2).

Формирование физических адресов ячеек памяти в микропроцессоре К1810

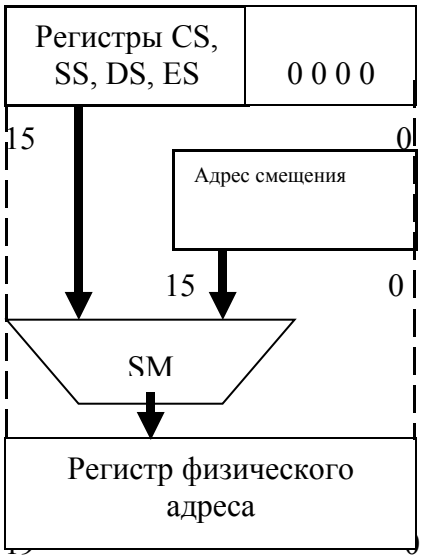


Рис. 2

Регистр флагов представляет собой набор триггеров, содержащий признаки результатов операций, необходимые для контроля вычислительного процесса и управления им. Таких триггеров 16, хотя для флажков используется лишь 9.

Операционное устройство в целом выполняет операции, определяемые командами, и формирует эффективные адреса.

Устройство управления микропроцессора обменивается сигналами с другими блоками микропроцессорной системы. В зависимости от числа и сложности внешних устройств может быть применена минимальная или максимальная конфигурация системы.

Исходное состояние центрального процессора перед запуском программы.

ТПО – должен находится в 0 состоянии.
РС – программный счётчик – содержит эффективный начальный адрес программы.
CS – начальный адрес сегмента, в котором располагается программа.
SP – эффективный текущий адрес вершины стека.
SS – начальный адрес сегмента, в котором располагается стек.
DS – начальный адрес сегмента, в котором располагаются данные.
ОП – программа, данные и стек.
РП – данные, адреса.

В остальных регистрах может находиться любая информация.

Передача информации из одного регистра в другой осуществляется за три такта, разрядность цифровых магистралей определяется разрядностью самого длинного регистра.

Формулирование содержания операций.

1.Команда управления микропроцессором: Команда останова(F4)

Команды главной группы обеспечивают программное управление различными функциями МП. Команда HLT предназначена для синхронизации МП с внешними событиями не влияющие на состояние флагов.

Команда HLT останова заставляет МП перейти в состояние останова. Из этого состояния МП может быть выведен или сигналом сброса CLR, или аппаратным прерыванием на входе NMI, или запросом прерывания на входе INT, если эти прерывания разрешены. Данную команду можно использовать вместо бесконечного программного цикла, когда программа должна ожидать внешнего прерывания.

По этой команде блок устройства процессора устанавливается в нулевое состояние.

Команда по варианту задания – Команда останова(F4)

Мнемоника	КОП	Длинна, байт	Функция
HLT	**F4**	1	Остановить микропроцессор

111	010

Работа «Триггера Пуска Останова»

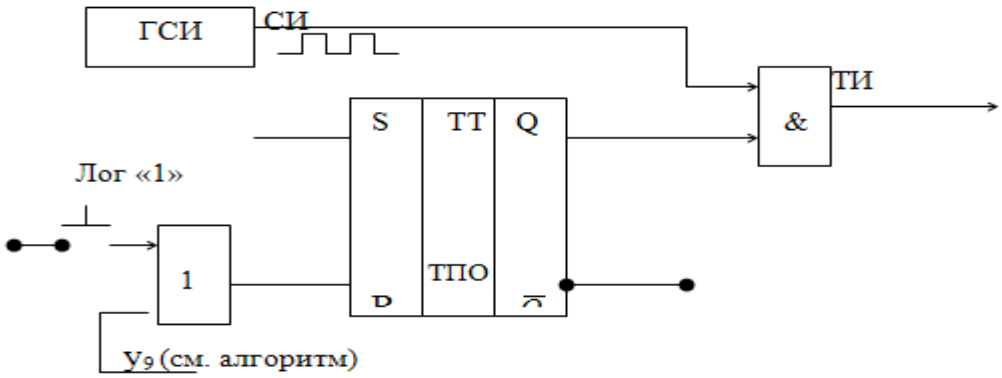


Рис. 3

Временная диаграмма работы «Триггера Пуска Останова»

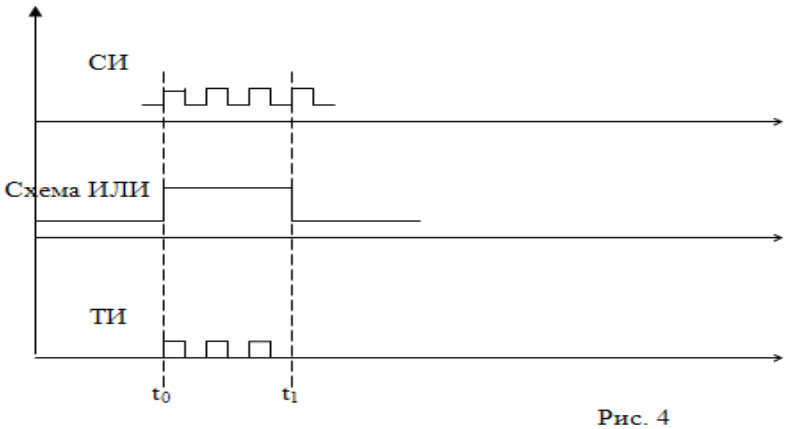


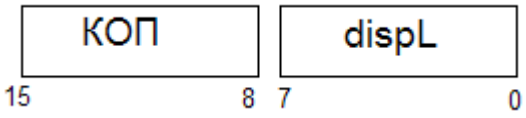
Рис. 4

Пока на схеме “или” логическая 1 (момент времени t_0) – процессор находится в рабочем состоянии, как только на схеме “или” логический 0, на выходе (схема “и”) – сигнала не будет – процессор будет остановлен (момент времени t_1).

2.Команда управления микропроцессором: Команда условного перехода (71)

В системе команд МП К1810 есть 19 двухбайтовых команд условных переходов, называемых также разветвлениями. Все они имеют единый формат. При выполнении этих команд анализируется некоторое условие, закодированное текущими состояниями флажков и в зависимости от удовлетворения условия переход осуществляется или нет.
Если условие истинно т.е. OF=0 (RF[11]=0), управление передается по адресу перехода путем прибавления к содержимому PC однобайтового знакового смещения (с расширением знака до16 бит), а если условие ложно т.е. OF=1 (RF[11]=1), выполняется следующая по порядку команда. Таким образом, все условные переходы являются короткими.

Формат команды:



Эффективный адрес очередной команды :
EA=(PC)+dispL
Где PC - адрес текущей команды,
Условие: OF=0 (RF[11]=0)

2.Команды арифметических операций:
Команда вычитания (FE) -

Микропроцессор К1810 имеет широкий набор команд, реализующих
арифметические операции, что позволяет применять его в сложных системах обработки данных. Арифметические операции выполняются над целыми числами 4-х типов: беззнаковыми двоичными, знаковыми двоичными, упакованными и не упакованными десятичными числами.

Длинна беззнаковых чисел составляет 8 или 16 бит, и все они считаются значащими, т.е. Учитываются при определении значения числа. Диапазон значений 8-битных чисел от 0 ÷ 256, а 16-битных чисел 0 ÷ 65535. Имеются команды сложения, вычитания, умножения и деления чисел, представленных в таких форматах.

Знаковые двоичные числа также м.б. 8 и 16-битными. Старший (левый) бит определяет знак числа: 0 – число положительное, 1 – число отрицательное. Числа представляются в стандартном дополнительном коде. Диапазон значений составляет от –128 ÷+127 для 8-битных чисел и от 32768 ÷ +32767 для 16-битных чисел. Число нуль содержит во всех разрядах нули и считается положительным. Для знаковых двоичных чисел имеются специальные команды умножения и деления, а сложение и вычитание благодаря применению дополнительного кода реализуются теми же командами, что и для беззнаковых чисел.

При выполнении арифметических операций особенности (или признаки) полученного результата фиксируются в 6 битах регистра флажков. Состояния большинства флажков проверяются командами условных переходов.

Операция вычитание в сумматоре производится следующем образом: Вычитаемое число сначала переводится в обратный код, затем прибавлением единицы к младшему разряду – в дополнительный код. Затем оба операнда складываются.

Пример – вычесть из 8-ки 2-ку:
1000
+
0010-->1101□
+1
1110
Затем 8-ка складывается с числом 2 в дополнительном коде.
1000
+
1110

0110 - Получаем число 6 в двоичном представлении. Проверка 8 – 2 = 6.

Команда по варианту задания - команда вычитания (FE)

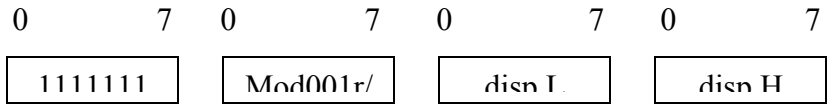
Команда декремента с общим представлением -

DEC dst dst:=(dst) - 1

Первый операнд находится в ОП, второй операнд – вычитаемая единица.

Предназначена для уменьшения на единицу содержимого регистра или ячейки памяти. Она не влияет на состояние флажка переноса CF.

DEC mem/reg



Вариант по заданию - Байт адресации: (BP)+(SI)+D16.

mod = 10; байт режима адресации -001; r/m = 010. Эффективный адрес – 8A.

1000	1010	- 8A

При выполнении данной команды будут рассмотрены поведение следующих флажков (по варианту задания):

Флаг ZF – фиксирует получение нулевого результата;

= 0 - ZF = 1
≠ 0 - ZF = 0

Флаг SF - соответствует значению старшего бита (бит 7 или бит 15) результата операции. В знаковой арифметике отражает знак результата; в беззнаковой – интерпретируется как цифра, а не как знак.

“ + ” - SF = 0
“ - ” - SF = 1

Флаг OF – в операциях со знаковыми числами фиксирует арифметическое переполнение, т. е. вывод результата на диапазон представляемых значений (при использовании дополнительного кода переполнение определяется сложением по модулю два значений переносов из знакового бита и старшего значащего бита).

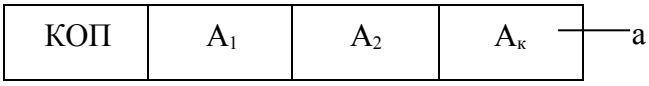
есть переполнение - OF = 1
3.нет переполнения - OF = 0

2. Выбор способов, методов и алгоритмов выполнения команд.

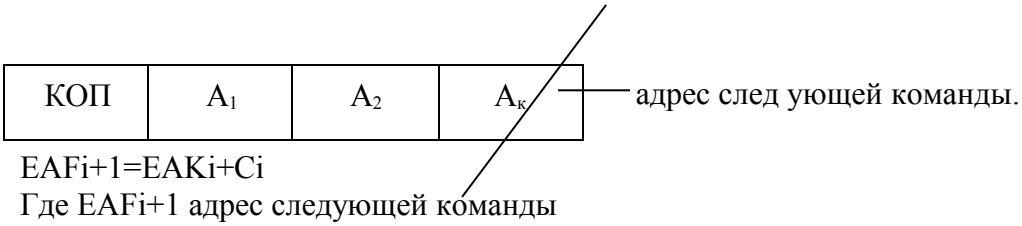
1)Выбираем естественный порядок выборки команд и их использование ,т.е. как команды записаны в программе в том же порядке они и выбираются. Этот способ применяется в МП1810 экономит емкость в памяти в битах за счет более коротких команд и существенно не снижает быстродействие.

Покажем пример

-принудительный порядок



-естественный порядок



EAFi+1=EAKi+Ci

Где EAFi+1 адрес следующей команды

EAKi эффективный адрес текущей команды

Ci количество байт в формате текущей команды.

A физический адрес Ki+1=(CS) +EAKi

В сегментном регистре CS находятся начальный адрес, в котором расположена текущая программа.

2)Выбираем последовательный порядок выполнения этапов программы. Он прост в реализации: проще схемы, меньше затрат оборудования и более хорошее быстродействие.

3)Принимаем полу синхронный способ управления. Он проще в реализации: экономичнее, но имеет хуже быстродействие ,т.к. у нас среднее быстродействие ,то данные характеристики являются нормальными.

4)Выбираем ОП и РП готовыми устройствами. Это более экономично и значительно упрощает процесс проектирования . Такт обращения к ОП берем продленным. Он определяется сигналом конца обращения к ОП, который вырабатывается в ОП. Для различных видов памяти он будет различным.

5)Для выполнения операции арифметики и переходов принимаем дополнительный код ,т.к. он сейчас широко распространен и используется в Мп1810 все операнды в ОП и РП хранятся в дополнительном коде.

6)Для выявления случаев переполнения задаем результат операции в модифицированном дополнительном коде (имеются два знаковых разряда)

7)Сложение выполняем в комбинационном параллельном сумматоре ,т.к. он применяется повсеместно и дает высокое быстродействие. Операцию сложения и формирования адресов производим в одном и том же сумматоре, что позволяет экономить оборудование хотя и снижает быстродействие.

8)В целях унификации и экономии оборудования первый этап выполняется оператором выбора команды и расшифр. И последний этап формирование прерывания делается общим для всех операций.

3 этап. Словесное описание процесса выполнения команд.

1.Расшифровка и выборка очередной команды.

- Адрес текущей команды посылается в ОП и по нему выбирается очередная команда и пересылается в регистр команд.
- Выбранная команда расшифровывается, т.е. определяется вид команды.
- Если в КОП выбранной команды не соответствует ни одной команде в списке команд для данного процессора ,то вырабатывается особый вид прерывания. В этом случае осуществляется переход к конечному этапу выполнения команды - организация прерывания.

2.Выборка операндов.

- Формируется адрес первого операнда.
- По этому адресу из ОП выбирается первый операнд и засылается на входной регистр АЛУ R1.
- Второй операнд из регистра команд засылается на входной регистр R2.

3.Выполнение сложения.

- В сумматоре производится сложение, и результат фиксируется в регистре суммы.
- Вырабатываются признаки результата и заносятся в RF.
- Если возник особый случай прерывания переполнения, то осуществляется переход к последнему этапу организации прерывания.

4.Организация прерываний.

Для стека отводится один сегмент памяти. В стеке хранятся адреса прерываний программы

- Формируется эффективный и физические адреса вершины стека
- По этому физическому адресу в стек записывается место прерываний команды: эффективный адрес текущей команды (содержимое регистра PC) и начальный адрес сегмента SS (хранится в сегментном регистре).

На основе таблиц векторов прерываний определяется адрес прерывающей программы данного типа прерывания, по которому из ОП выбирается начальный адрес прерывания программы. На этом выполнение текущей команды заканчивается. Если в течении выполнения команды выработан ни один тип прерываний, то четвертый этап отсутствует.

4.Выбор блоков памяти и регистров.

1.Выбираем обобщенный операционный элемент –ОП.
По заданию емкость ОП составляет 512кбайт.
RAОП[18/0] $n_{raop} = \log_2 512 = 19$разрядов.
RИОП[31/0] $n_{RИОП} = S = 4\ байта$, где S-ширина выборки.

2.Выбираем сверхоперативный регистр память.
РП
RАРП[2/0] $\log_2 8 = 3$
RИРП[15/0] разрядность регистров ОП.

3. $R_k[31/0] = 32$ разряда. Старшие разряды команды КОП)выбирается в старшие разряды регистра команд.

4. PC[15/0]-для хранения и формирования эффективного адреса текущей команды.

5. RF[15/0]-для хранения флагов, признаков результата, признаков прерывания.

6. R1[18/0],R2[18/0]-входные регистры сумматора,разрядность Которых определяется разрядностью RAОП., т.к. в сумматоре производится формирование Аф.

7. RS[18/0]-регистр результата триггер ‘Пуска-Останова’ – триггер управления блока ‘Пуска-Останова’.

8. SP[15/0], CS[15/0], SS[15/0], DS[15/0]-указатель стека ЕА вершины стека.

5. Выбор операционных элементов.

1.Для выполнения микроопераций суммирования используем комбинационный параллельный сумматор. Для приема информации и ее выдачи используем линейки схем ‘И’, на одни входы подается поразрядно информация, а на другие входы –управляющий сигнал. Если на вход регистра поступает информация со многих регистров. То лучше использовать мультиплексор.

2.Выбираем операционные элементы для выработки признаков результата, прерывания и других логических условий. Выбираем различные комбинационные схемы в зависимости от флага.

6.Разработка объединенного ГСА для выполнения команд.

1. ГСА составляется объединенной для всех команд в виде ориентированного графа, вершинами которого являются операционные условные ждущие вершины. Вершины начала и конца ГСА.
2. Микрооперации и логические условия записываются в элементах языка ОСС-2-языка моделирования процессов вычислений на уровне регистров.
3. Продленный такт реализуется с помощью ждущей вершины.
4. Пуск процессора также осуществляется в ждущем режиме
5. Этапы выборки и расшифрования команды и организация прерывания делаются одними и теми же для всех команд.
6. микрооперации указываем через операционные вершины
7. Условие перехода в ГСА задаем через условные вершины
8. Продленный такт и начальный пуск реализуется через ждущие вершины
9. Начало и конец алгоритма реализуется через начальные и конечные вершины
10. в одной операционной вершине может быть записано несколько микроопераций, если они выполняются одновременно
11. ГСА должен быть универсальной для любых значений и комбинаций операндов в заданном интервале изменения величин и в пределах заданных форматов данных
12. Любая ветвь алгоритма должна начинаться с условной вершины и входить(т.е. заканчиваться) в операционную, ждущую или конечную вершины.

Общий (универсальный) алгоритм:

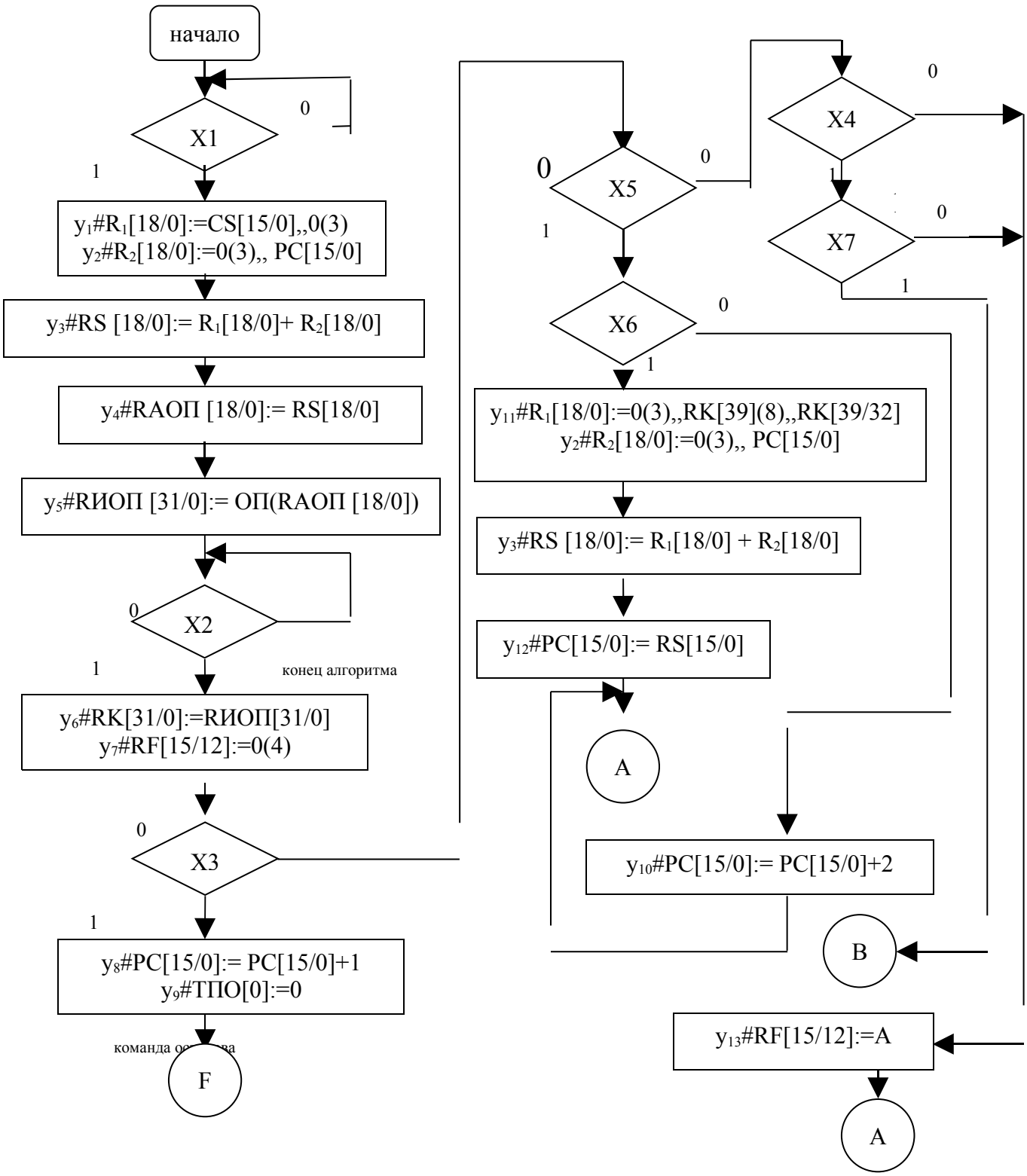


Рис. 5

Алгоритм операции вычитания:

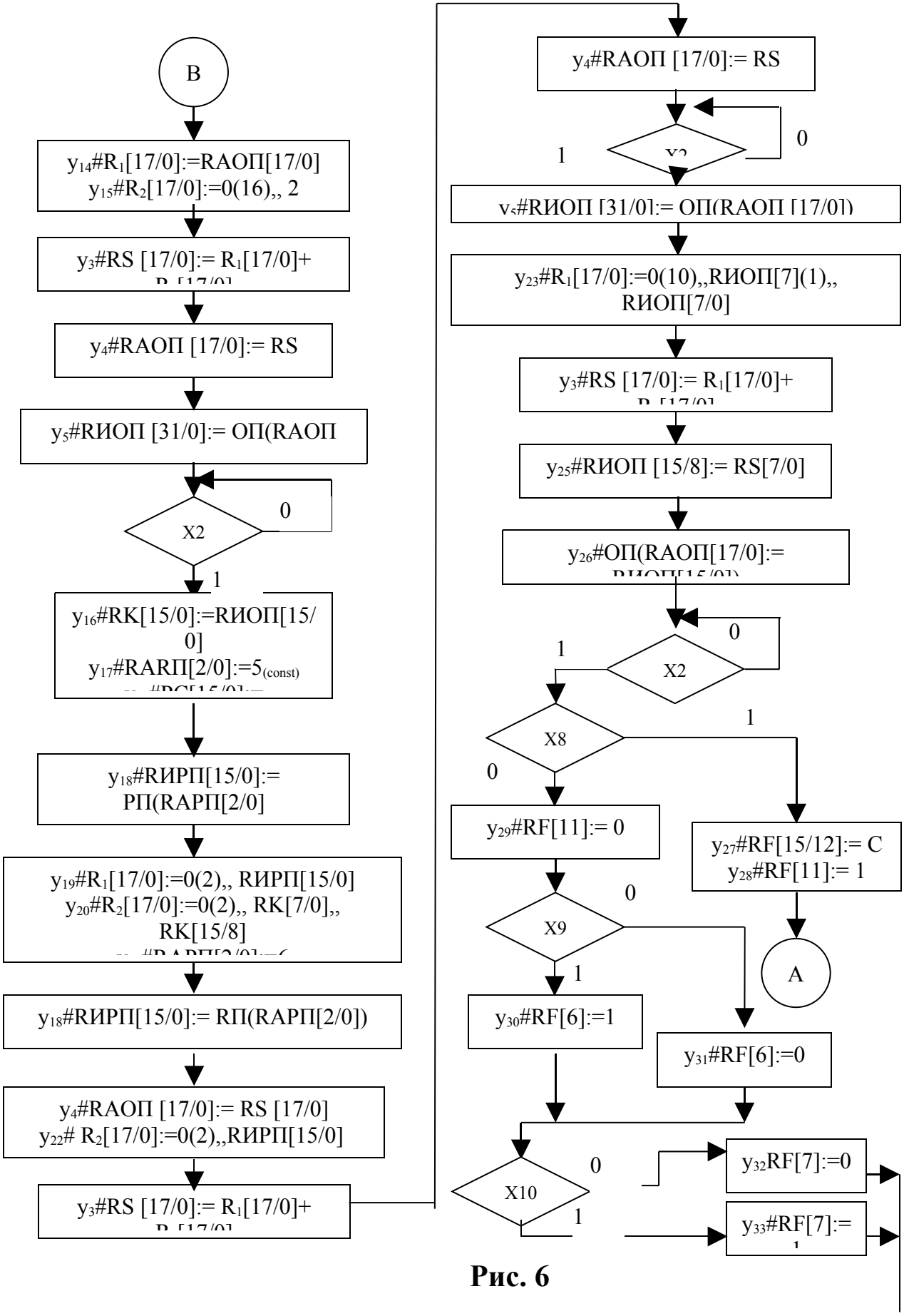
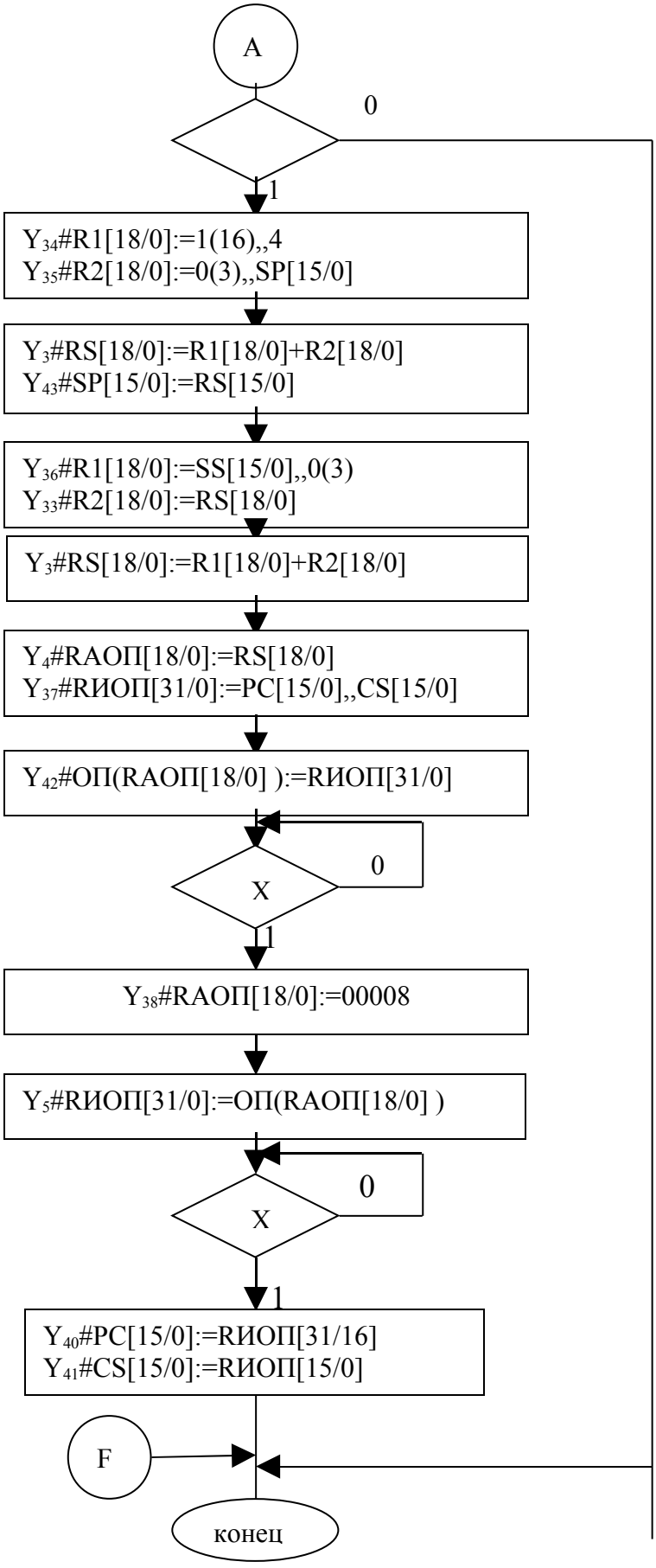


Рис. 6


A
0
1
Y34#R1[18/0]:=1(16),,4
Y35#R2[18/0]:=0(3),,SP[15/0]
Y3#RS[18/0]:=R1[18/0]+R2[18/0]
Y43#SP[15/0]:=RS[15/0]
Y36#R1[18/0]:=SS[15/0],,0(3)
Y33#R2[18/0]:=RS[18/0]
Y3#RS[18/0]:=R1[18/0]+R2[18/0]
Y4#RAОП[18/0]:=RS[18/0]
Y37#RИОП[31/0]:=PC[15/0],,CS[15/0]
Y42#ОП(RAОП[18/0]):=RИОП[31/0]
X
0
1
Y38#RAОП[18/0]:=00008
Y5#RИОП[31/0]:=ОП(RAОП[18/0])
X
0
1
Y40#PC[15/0]:=RИОП[31/16]
Y41#CS[15/0]:=RИОП[15/0]
F
конец

7-й Этап

Составление таблицы регистров, микроопераций и логических условий

№	Регистры и блоки управления	МКО	Логические условия
1	R1	Y1#R1[18/0]:=CS[15/0],,0(3)	X1=1
		Y11#R1[18/0]:=0(3),,RK[23](8),,RK[23/16]	X6=1
		Y17#R1[18/0]:=0(3),,РИРП[15/0]	1
		Y20#R1[18/0]:=RS[18/0]	1
		Y22#R1[18/0]:=0(3),,РИОП[31/16]	X2=1
		Y34#R1[18/0]:=1(16),,4	X10=1
		Y36#R1[18/0]:=SS[15/0],,0(3)	1
2	R2	Y2#R2[18/0]:=0(3),,PC[15/0]	X1=1
		Y16#R2[18/0]:= 0(3),,RK[7/0],,RK[15/8]	1
		Y21#R2[18/0]:= DS[15/0],, 0(3)	1
		Y23#R2[18/0]:=0(3),, РИРП[15/0]	X2=1
		Y35#R2[18/0]:=0(3),,SP[15/0]	X10=1
		Y33#R2[18/0]:=RS[18/0]	1
3	RS	Y3#RS[18/0]:=R1[18/0]+R2[18/0]	1
		Y25#RS[18/0]:=R1[18/0]^R2[18/0]	1
4	РАОП	Y4#РАОП[18/0]:=RS[18/0]	1
		Y38#РАОП[18/0]:=00008	1
5	РИОП	Y5#РИОП[31/0]:=ОП(РАОП[18/0])	1
		Y37#РИОП[31/0]:=PC[15/0],,CS[15/0]	1
6	РАРП	Y14#РАРП[2/0]:=7	X7=1
7	РИРП	Y18#РИРП[15/0]:=РП(РАРП[2/0])	1
8	RK	Y6#RK[31/0]:=РИОП[31/0]	X2=1
9	RF	Y7#RF[15/12]:=0(4)	X2=1
		Y13#RF[15/12]:=A	X4=0||X7=0
		Y28#RF[6]:=1	X8=1
		Y29#RF[6]:=0	X8=0
		Y30#RF[7]:=1	X9=1
		Y31#RF[7]:=0	X9=0
10	CS	Y41#CS[15/0]:=РИОП[15/0]	X2=1
11	PC	Y8#PC[15/0]:=PC[15/0] + 1	X3=1
		Y12#PC[15/0]:=RS[15/0]	1
		Y10#PC[15/0]:=PC[15/0]+2	X6=0
		Y40#PC[15/0]:=РИОП[31/16]	X2=1
12	ТПО	Y9#ТПО[0]:=0	X3=1

УСЛОВНЫЕ ПЕРЕХОДЫ.

0, *если* $ТПО = 0$,
1, *если* $ТПО = 1$.

¿
$X1 = ¿[¿¿¿$
¿

0, *если обращение к ОП не закончено*,
1, *если обращение к ОП закончено*.

¿
$X2 = ¿[¿¿¿$
¿

1, *если* $КОП = F4$,
0, *если* $КОП \neq F4$.

¿
$X3 = ¿[¿¿¿$
¿

1, *если* $КОП = 85$,
0, *если* $КОП \neq 85$.

¿
$X4 = ¿[¿¿¿$
¿

1, *если* $КОП = 71 = RF[11]$,
0, *если* $КОП \neq 71 \neq RF[11]$.

¿
$X5 = ¿[¿¿¿$
¿

0, *если* $SF = 0$,
1, *если* $SF \neq 0$.

¿
$X6 = ¿[¿¿¿$
¿

1 *если* $БА = 81$,
0, *если* $БА \neq 81$.

¿
$X7 = ¿[¿¿¿$
¿

$$1\ ,\ RS[15/0]=0$$
$$0\ ,\ RS[15/0]\neq 0.$$

¿
$X8=$¿[¿¿¿
¿

$$0,\ \textit{если}\ RS[15]=1,$$
$$1,\ \textit{если}\ RS[15]=0.$$

¿
$X9=$¿[¿¿¿
¿

$$1,\ \textit{если}\ RF[15/12]=A,$$
$$0,\ \textit{если}\ RF[15/12]\neq A.$$

¿
$X\,10=$¿[¿¿¿
¿

Описание последней части алгоритма.

1.Расшифровка и выборка очередной команды.

a. Адрес текущей команды посылается в ОП и по нему выбирается очередная команда и пересылается в регистр команд.
b. Выбранная команда расшифровывается, т.е. определяется вид команды.
c. Если код выбранной команды не соответствует ни одной команде в системе команд для данного процессора, то вырабатывается особый случай прерывания “не существующий КОП”.
d. В этом случае осуществляется переход к конечному этапу выполнения команды организации прерывания.

2.Выборка операндов.

a.Формируется адрес 1-го операнда.
b.По этому адресу из ОП выбирается 1-ый операнд и засылается на выходной регистр АЛУ R1.
c.2-ой операнд из регистра команд засылается на входной регистр R2.

3.Выполнение вычитания.

4.В сумматоре производится сложение (операндов) – (вычитаемое – в дополнительном коде) и результат фиксируется в регистре суммы.

5.Вырабатывается признак результата и заносится в регистр RF.

6.Если возник особые случай прерывания переполнения, то осуществляется переход к последнему этапу организации прерывания.

4.Организация прерываний.

Для стека отводится один сегмент памяти, в стеке хранятся адреса прерванных программ.

a.Формируются ЕА и Аф вершины стека.
b.По этому адресу (Аф) в стек записывается место прерванной команды: ЕА текущей команды (содержимое PC) и начальный адрес сегмента SS, который хранится в сегментном регистре SS.
c.На основе таблицы векторов прерывания определяется адрес прерывающей программы данного типа прерывания, по которому из ОП выбирается начальный адрес прерывающей программы.

(PC) } начальный адрес
(CS) } прерывающей команды.

d.Эти адреса записываются в соответствующие регистры:

$PC \leftarrow (PC)$
$CS \leftarrow (CS)$

На этом выполнения текущей команды заканчивается.

Если в течении выполнения команды не вырабатывается ни один случай прерывания 4-ый этап отсутствует.

Проектирование АЛУ с закрепленными микрооперациями.

Структура такого АЛУ обычно определяется алгоритмом выполнения операций ГСА.

Преимущества:

Высокое быстродействие, т.к. информация из одного регистра в другой передается без всяких промежуточных регистров.

Недостатки:

Высокая нерегулярность схем (увеличение брака, стоимости).

Для выработки логических условий X3, X4, X5, X7 – определение КОП разрабатываются отдельные схемы. Их синтез приведен в приложении 1.

8. Минимизация ГСА.

В процессе моделирования часто приходится возвращаться на более ранние этапы проектирования. Чтобы удовлетворить заданным требованиям. В нашем случае это быстродействие и экономичность оборудования.

1.для удовлетворения заданных требований по быстродействию необходимо:

- использовать более быстрод. алгоритм;
- стремиться к сокращению операционных вершин;
- как можно более заключать м/о в одну вершину, если они могут выполняться совместно за 1 такт.

2. для уменьшения затрат оборудования нужно:

- использовать одни и те же м/о для выполнения однотипных действий;
- составлять логические условия так, чтобы схема их выработки получилась более экономичной;
- сокращать общее количество м/о за счет многократного использования одной м/о.

9. Выбор функциональных элементов.

1.В первую очередь необходимо, если есть возможность выбирать функциональные элементы из имеющегося набора интегральных схем.

2.если нет таких функциональных элементов, то их необходимо синтезировать и построить из более мелких элементов серии заданной или другой, у которой элементы с ней совместимы.

10. Построение функциональной электрической схемы процессора.

1.Все функциональные элементы изображаются на входе регистров;

2.На схеме изображаются все регистры и блоки памяти, а на их входах схемы функциональных элементов.

3.Наносятся все связи между регистрами.

- Информационные
- Управляющие
- Адресные
- Связи логических условий

№	Регистры и блоки управления	МКО	Логические условия
1	SP	Y43#SP[15/0]:=RS15/0]	1
2	ОП(РАОП)	Y42# ОП(РАОП[18/0]):=РИОП[31/0]	1

Синтез логических переходов:

X1 = { 0, если ТПО = 0
       1, если ТПО = 1

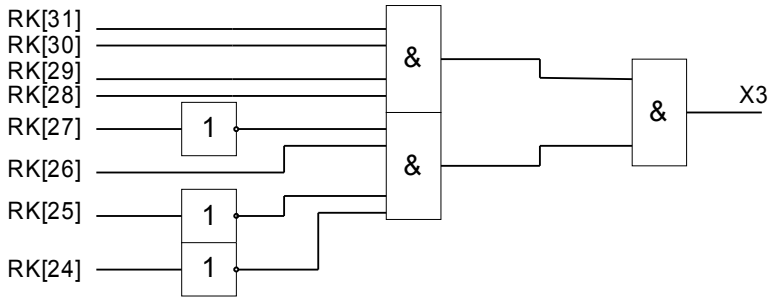
ТПО[0] ——————————— X1

X2 = { 0, если обращение к ОП не закончено
       1, если обращение к ОП закончено

X3 = { 1, если КОП = F4
       0, если КОП ≠ F4

F4 = 11110100, A_i = RK[i]

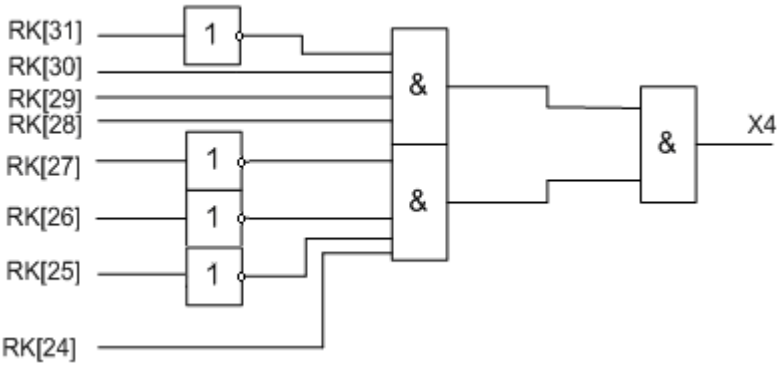
$$X3 = A_{31} \cdot A_{30} \cdot A_{29} \cdot A_{28} \cdot \bar{A}_{27} \cdot A_{26} \cdot \bar{A}_{25} \cdot \bar{A}_{24}$$



X4 = { 1, если КОП = 71
       0, если КОП ≠ 71

73 = 01110001, A_i = RK[i]

$$X4 = \bar{A}_{31} \cdot A_{30} \cdot A_{29} \cdot A_{28} \cdot \bar{A}_{27} \cdot \bar{A}_{26} \cdot \bar{A}_{25} \cdot A_{24}$$

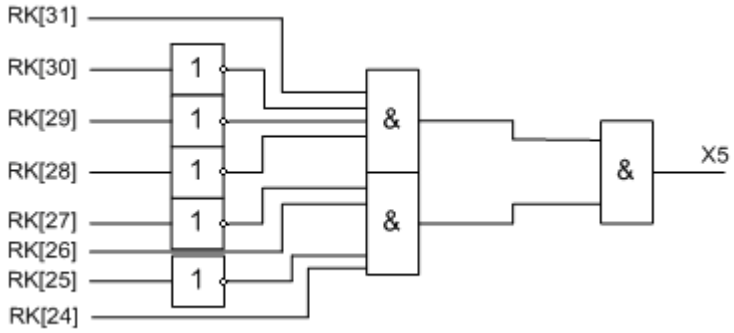


1, *если КОП* = 85,
0, *если КОП* ≠ 85.

¿
X5 = ¿{¿¿¿
¿

05 = 00000101, A_i = RK[i]

$$X5 = A_{31} \cdot \bar{A}_{30} \cdot \bar{A}_{29} \cdot \bar{A}_{28} \cdot \bar{A}_{27} \cdot A_{26} \cdot \bar{A}_{25} \cdot A_{24}$$



0, *если SF* = 0,
1, *еслиSF* ≠ 0.

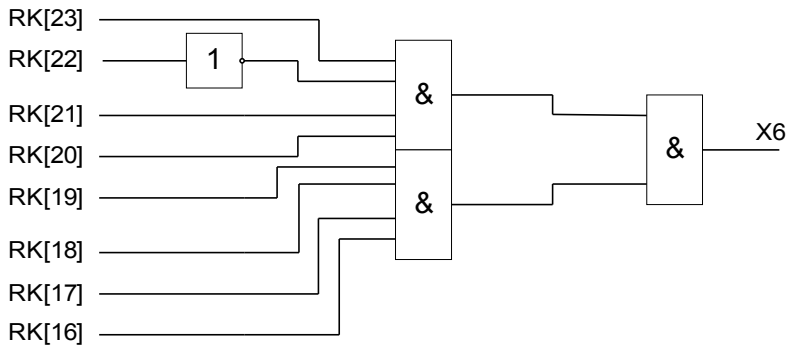
¿
X6 = ¿{¿¿¿
¿

SF ——————————— X6

X7 = { 1, если БА = BF
0, если БА ≠ BF

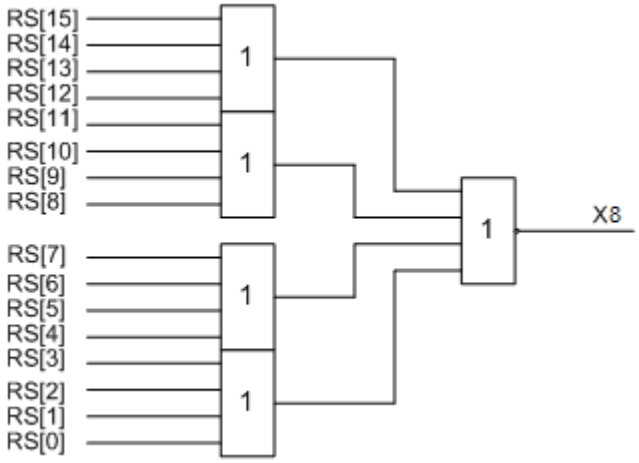
BF = 1011 1111, A_i = RK[i]

$$X6 = A_{23} \cdot \bar{A}_{22} \cdot A_{21} \cdot A_{20} \cdot A_{19} \cdot A_{18} \cdot A_{17} \cdot A_{16}$$



X8 = $\begin{cases} 1, \text{если RS[15/0]} = 0(16) \\ 0, \text{если RS[15/0]} \neq 0(16) \end{cases}$

$$X8 = \frac{Ai = RS[i]}{A_0 + A_1 + A_2 + A_3 + A_4 + A_5 + A_6 + A_7 + A_8 + A_9 + A_{10} + A_{11} + A_{12} + A_{13} + A_{14} + A_{15}}$$



X9 = $\begin{cases} 1, \text{если RS[15]} = 1 \\ 0, \text{если RS[15]} \neq 1 \end{cases}$

X10 = $\begin{cases} 1, \text{если RF[15/12]} = A \\ 0, \text{если RF[15/12]} \neq A \end{cases}$

Выводы.

В данном курсовом проекте был разработан специализированный процессор для двух команд (АЛУ, выполняющее операции сложения и вычитания с фиксированной запятой).

В работе составлено словесное описание каждой выполняемой операции.

При выборе регистров и блоков памяти была выбрана готовая ОП, а в качестве сверхоперативной памяти используется регистровая.

Синтез структуры процессора был проведен по объединенной граф-схеме алгоритма (ГСМ), что позволило снизить объем оборудования по сравнению с реализацией по раздельным ГСА.

Принципиальная схема операционного блока реализована на ИС типа ТТЛ серий К155 и К555, которые широко распространены и дешевле подобных ИС других серий, хотя и обладают сравнительно низким быстродействием.

При выборе методов, способов, алгоритмов для выполнения операций акцент делался на уменьшение потери быстродействия устройства без ущерба в работе.

Используемая литература:

1.Шапкин. Курс лекций. МГТУГА 2001-2002 г.
2.В.Л. Григорьев. Программирование однокристальных процессоров. Энергоатомиздат. 1987 г.
3.Е.П. Угрюмов. Проектирование элементов и узлов ЭВМ. Высшая школа 1987 г.
4.В.Г. Майоров, А.И. Гаврилов. Практический курс программирования микропроцессорных систем. Машиностроение 1989 г.
5.С.А. Майоров, Г.И. Новиков. Структура ЭВМ. Машиностроение 1989 г.
6.Е.П. Балашов, В.Л. Григорьев, Г.А. Петров. Микро и мини – ЭВМ. Энергоиздат 1984 г.