МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ имени академика С.П. КОРОЛЕВА
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ» (СГАУ)

ОРГАНИЗАЦИЯ АРИФМЕТИКО-ЛОГИЧЕСКИХ УСТРОЙСТВ ЭВМ

Рекомендовано редакционно-издательским советом федерального государственного бюджетного образовательного учреждения высшего профессионального образования «Самарский государственный аэрокосмический университет имени академика С.П. Королева (национальный исследовательский университет)» в качестве методических указаний

2-е издание, переработанное

С а м а р а
Издательство СГАУ
2014

УДК 004 (075)
ББК 32.97я7

Составители: ***О. А. Заякин, В. П. Павлов***

Рецензент д-р техн. наук проф. П. К. Л а н г е

Организация арифметико-логических устройств ЭВМ : метод. указания к лабораторным работам по дисциплине Б3.Б.2 «ЭВМ и периферийные устройства» направления 230100.62 «Информатика и вычислительная техника» / сост. *О. А. Заякин, В. П. Павлов.* – 2-е изд., перераб. – Самара: Изд-во Самар. гос. аэрокос. ун-та, 2014. – 60 с.

В методических указаниях изучается структурная организация основной составляющей процессора ЭВМ – его арифметико-логического устройства.

Для изучения использованы графический редактор и эмулятор из системы автоматизированного проектирования MAX+PLUS II фирмы «Altera», разработанной ею для программирования собственных ПЛИС.

Предназначены для студентов, обучающихся дисциплине Б3.Б.2 «ЭВМ и периферийные устройства» по направлению 230100.62 «Информатика и вычислительная техника», а также по специальности «Автоматизированные системы обработки данных и управления». Могут быть полезны студентам, обучающимся по другим специальностям, связанным с информационными технологиями.

Работа выполнена на кафедре информационных систем и технологий Самарского государственного аэрокосмического университета.

УДК 004 (075)
ББК 32.97я7

СВЕДЕНИЯ О СОСТАВИТЕЛЯХ

Заякин Олег Александрович – кандидат технических наук, научный сотрудник лаборатории моделирования и автоматизации лазерных систем Самарского филиала Физического института РАН (СФ ФИАН), доцент кафедры информационных систем и технологий (ИСТ) Самарского государственного аэрокосмического университета (СГАУ). Область научных интересов: лазерные измерения геометрических величин, когерентная оптика, жидкокристаллическая адаптивная оптика.

Результаты его работы нашли отражение в 40 научных работах, в том числе в патенте на изобретение, выступлениях более чем на 10 международных и региональных конференциях и симпозиумах.

Является руководителем более 40 дипломных проектов и работ.

Павлов Владимир Павлович – кандидат технических наук, доцент СГАУ. Область научных интересов: организация структур и вычислительных процессов в ЭВМ; научное направление: оценка производительности вычислительных систем; учебные курсы: организация ЭВМ, периферийные устройства систем автоматизации, устройства хранения информации.

Количество публикаций – более 60, из них одно авторское свидетельство. Участник многих региональных, международных, всесоюзных, а затем всероссийских конференций и симпозиумов.

Его школу прошли многие студенты и аспиранты.

ОГЛАВЛЕНИЕ

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

АЛО – арифметико-логическая микрооперация;

АЛУ – арифметико-логическое устройство;

АМК – адрес микрокоманды;

БИС – большая интегральная микросхема;

БП – безусловный переход;

ВМ – вычислительная машина;

Г – закодированный граф микропрограмм;

ГТИ – генератор тактовых импульсов;

ЕП – естественный переход (то есть по адресу следующей по порядку микрокоманды);

ЖЛ – жесткая логика, названа в противоположность программируемой, «мягкой» (soft) логике;

ЗУ – запоминающее устройство;

ЛУ – логическое условие;

ИСТ – кафедра информационных систем и технологий СГАУ;

КС – комбинационный сумматор;

КСЧ – комбинационный счетчик;

МК – микрокоманда;

МО – микрооперация;

МП – микропрограмма;

МУ – методические указания;

НЛУ – номер логического условия;

ОА – операционный автомат;

ОЗУ – оперативное запоминающее устройство;

ОПУ – комплекс операционных устройств;

ОС – операционная система;

ОУ – операционное устройство;

ПЗУ – постоянное запоминающее устройство (то есть без возможности перезаписи);

ПЛ – программируемая логика;

ПЛИС – программируемая логическая интегральная схема;

ПМП – память микропрограммы;

ПНА – преобразователь начального адреса;

ПО – программное обеспечение;

РА – регистр адреса микрокоманды;

РАН – Российская академия наук;

РМК – регистр микрокоманды;

РОН – регистр общего назначения;

САПР – система автоматизированного проектирования;

СГАУ – федеральное государственное бюджетное образовательное учреждение высшего профессионального образования «Самарский государствен-

ный аэрокосмический университет имени академика С. П. Королёва (национальный исследовательский университет)»;

СЛУ – список (множество) логических условий;

СМК – счетчик микрокоманд;

СМО – список (множество) микроопераций;

СФАМ – схема формирования адреса следующей микрокоманды;

СФ ФИАН – Самарский филиал федерального государственного бюджетного учреждения науки Физического института им. П. Н. Лебедева Российской академии наук;

ТИ – тактовый импульс;

УА – управляющий автомат;

УА с ЖЛ – управляющий автомат с жесткой логикой управления;

УА с ПЛ – управляющий автомат с программируемой логикой;

УКС – универсальный комбинационный сумматор;

УМКД – учебно-методический комплекс дисциплины;

УП – условный переход;

УС – указатель стека;

ФК – формирователь кодов;

ФУС – формирователь управляющих сигналов;

ЦУУ – центральное устройство управления;

ЭВМ – электронная вычислительная машина;

ALO – арифметико-логическая операция;

BR – буферный регистр;

CISC – Complex Instruction Set Computer = архитектура с полным набором команд;

C – «Carry» − значение бита переноса из старшего разряда сумматора при выполнении результатов сложения или вычитания;

CF – флаг переноса;

FLU – формирователь логических условий;

DMC – управляющий демультиплексор;

I-автомат – ОА с максимальными характеристиками (с максимальной производительностью и максимальными затратами оборудования);

IBM PC – International Business Machines Personal Computer;

IM-автомат – ОА с промежуточными (между минимальными и максимальными) значениями характеристик между I- и M-автоматами;

M-автомат – ОА с минимальной производительностью (одна МО за один такт);

MAX+PLUS II − Multiple Array MatriX Programmable Logic User System = пользовательская система программирования логики упорядоченных структур;

MS – мультиплексор;

MS-триггер – триггер типа «Master-Slave»;

MSA – мультиплексор А;

MSA – мультиплексор адреса;

MSB – мультиплексор B;

MSC – управляющий мультиплексор;

MSF – мультиплексор флагов;

MSL – мультиплексор левого сдвига;

MSR – мультиплексор правого сдвига;

NOP – «no operation» − пустой такт;

OVR – «Overflow» − признак (флаг, бит) переполнения, значение равно единице, если результат арифметической МО переполняет разрядную сетку, и нулю в противном случае;

OF – флаг переполнения, устанавливается в единицу, если в результате сдвига изменился знаковый (старший) разряд результата;

P – поле микрокоманды, указывающее тип этой микрокоманды;

RF – регистр флагов;

RISC – Reduced Instruction Set Computer = архитектура с сокращенным набором команд;

RON – регистр общего назначения;

S – «Storage» − регистр (обозначение на схеме) или его содержимое;

S – «Sign» − значение старшего разряда результата, при обработке чисел со знаком трактуется как знак результата;

SH – «Shift» − сдвигатель, то есть регистр сдвига;

SL – вход сдвигателя, используемый при левом сдвиге;

SM – сумматор;

SR – вход сумматора, используемый при правом сдвиге;

T – продолжительность такта работы цифрового устройства;

TC – счетный триггер;

TT – счетный триггер;

UKS – универсальный комбинационный сумматор;

W – «Write» − сигнал записи;

Z – «Zero» − признак (флаг, бит) нулевого результата.

ПРЕДИСЛОВИЕ

Данные методические указания (МУ) составлены в соответствии с учебным планом СГАУ бакалаврской подготовки 230100.4.62-2014-О-П-4г00м для изучения дисциплины Б3.Б.2 «ЭВМ и периферийные устройства», относящейся к направлению 230100.62 «Информатика и вычислительная техника» (ИВТ). Они также могут быть полезны студентам, обучающимся по специальностям, связанным с информационными технологиями.

В пособии описаны только две лабораторные работы из всего курса, рассчитанного на один семестр. Студентам отводится на них 8 академических часов практических занятий и 6 часов самостоятельной работы.

МУ к лабораторным работам посвящены изучению структурной организации электронно-вычислительных машин (ЭВМ). Изучение этих вопросов необходимо для понимания работы вычислительных систем.

МУ дают студентам знания принципов устройства ядра современных ЭВМ, а также дают им типичные примеры практической реализации этих принципов на примере арифметико-логического устройства (АЛУ) – основного узла микропроцессора.

Рассмотрен двухтактный конвейерный АЛУ, работающий на основе принципа микропрограммного управления, применяющегося сейчас практически во всех цифровых вычислительных устройствах. Этот АЛУ имеет двухуровневое управление процессом вычислений, также широко применяемое в современных ЭВМ и контроллерах.

Обсуждаются различные способы кодирования микрокоманд, синхронизация комбинационных схем с обычно более медленными оперативными запоминающими устройствами (ОЗУ).

Для изучения использованы виртуальные макеты двух составных структурных блоков АЛУ – операционного и управляющего автоматов. Для их создания и для обучения на них студентов использован графический редактор и эмулятор из одной из систем автоматизированного проектирования (САПР), разработанной для программирования ПЛИС. Это позволяет не только изучить проект и даже самостоятельно его создать, но и увидеть «изнутри» его работу, изучить реакцию системы во времени на множество факторов, которые заранее трудно учесть.

В лабораторных работах использованы виртуальные макеты (так называемые «проекты», по терминологии фирмы-разработчика данной САПР), разработанные студентами и преподавателями СГАУ. Они прошли успешную апробацию в образовательном процессе СГАУ на факультете «Информатика» в курсовом и дипломном проектировании, а также в практических занятиях и лабораторных работах студентов специальности «Автоматизированные системы обработки информации и управления» (АСОИУ). Все эти проекты можно найти в учебно-методическом комплексе дисциплины (УМКД) «Аппаратные средства вычислительной техники» по специальности 090105 «Комплексное обеспечение информационной безопасности автоматизированных систем». Этот УМКД относится ко второму поколению таких комплексов. Он составлен на кафедре ИСТ СГАУ.

Изучая описанные здесь лабораторные работы, студенты решают следующие основные задачи:

– ознакомление с принципами организации АЛУ;

– изучение основ работы с современными средствами проектирования цифровых вычислительных систем;

– изучение работы АЛУ.

Перед выполнением лабораторных работ по данным МУ студентам следует ознакомиться с используемой в них САПР. Студентам предлагается это сделать на примере самостоятельного проектирования относительно несложного устройства – четырехразрядного двоичного вычитателя.

В данных лабораторных работах студенты проходят фактически лишь первоначальное знакомство с АЛУ. Поэтому для относительной простоты изложения в него не включены вопросы программной и аппаратной реализации арифметических операций для чисел с плавающей запятой.

Изучение предмета в данных лабораторных работах основано на булевой логике, теории автоматов и цифровой схемотехнике.

В подготовке лабораторных работ – в качестве ядра автоматизированной системы использовались компьютеры, совместимые с IBM PC. Использованы компьютеры на основе микропроцессоров типа Pentium.

Программное обеспечение (ПО) компьютеров, использованное в лабораторных работах, следующее. Это операционные системы (ОС) Windows XP, либо Windows Vista, либо Windows 7. Прикладное программное обеспечение включает в себя типичный набор прикладных программ для офиса, обычно устанавливаемых на компьютере вместе с ОС.

Для выполнения лабораторных работ потребуется также и специализированное приложение – САПР MAX+PLUS II («Altera», США) в версии не ниже десятой. В данных работах в последнее время использовалась версия 10.2.

По описанию данной САПР и работе с ней издано немало пособий. Для успешного выполнения лабораторных работ мы рекомендуем студентам ознакомиться с трудами украинских специалистов В. Поречного [1] и В. Стешенко [2]. Первое из них подходит для первоначального знакомства с предметом, а второе – для более углубленного изучения и для справок.

В МУ достаточно подробно излагаются теоретические вопросы.

Успешное изучение предмета на данных лабораторных работах требует знания основ цифровой схемотехники. Кроме этого студентам также потребуются базовые физико-математические знания в объеме технического вуза.

Знания, полученные из данного практикума, будут полезными в курсах «Микропроцессорные средства систем автоматизации», «Операционные системы», «Интерфейсы АСОИУ». Эти знания также являются основой для курсового и дипломного проектирования.

Лабораторные работы из данных методических указаний следует дополнить ознакомительной работой с САПР. Кроме того, в практическую часть дисциплины «ЭВМ и периферийные устройства» входит лабораторная работа, посвященная функциональной организации ЭВМ (на примере системы пре-

рываний компьютеров IBM PC), а также лабораторная работа, посвященная системе ввода/вывода компьютеров этого семейства.

Отметим, что вместе с другими названными здесь лабораторными работами практическая часть занимает большой объем. Так, согласно рабочей программе по дисциплине «ЭВМ и периферийные устройства» и учебному плану 230100.4.62-2014-О-П-4г00м, на очные занятия отводятся 54 академических часа, и столько же − на самостоятельную работу.

При сокращенном объеме обучения по данным МУ можно совместить ознакомление с САПР и первую из работ.

Все использованные в данных работах материалы взяты из открытых источников и использованы (а также предназначены для использования в дальнейшем) только в учебных и образовательных целях.

Этот труд впервые выходит в тираж, хотя в практике работы со студентами на кафедре ИСТ СГАУ к настоящему времени использованы уже две его редакции. В ходе этой апробации был замечен и устранен ряд недостатков как самих МУ, так и используемых в практике проектов. Однако, как в любой сложной открытой системе, отдельные недостатки все же не могут не присутствовать. Это следует учесть тем читателям, кто решится использовать данный материал «в железе». Авторы и редакция не могут нести ответственность за возможные последствия в этих случаях. Тем не менее, все замечания и конструктивные предложения будут рассмотрены и учтены.

Информация для контактов: адрес электронной почты: oleg_zayakin@inbox.ru; адрес обычной почты: СФ ФИАН, ул. Ново-Садовая, 221, г. Самара, 443011, Россия; номер служебного телефона: +7 846 335 95 83; номер служебного факса: +7 846 335 56 00; служебная веб-страница (строго модерируется!): http://www.fian.smr.ru/personal_page.php?id=48&lang=rus .

Хочется выразить признательность доценту кафедры ИСТ А. С. Овсянникову за любезное предоставление проектов для использования их в MAX+PLUS II.

ВВЕДЕНИЕ

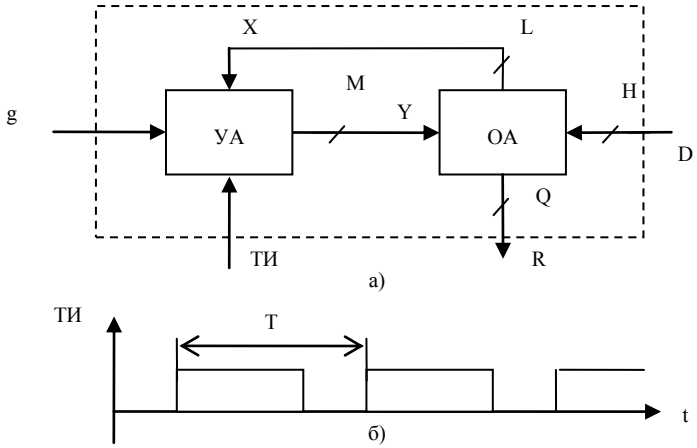
Назначением АЛУ является реализация арифметических, логических и других операций из списка операций. В состав современных процессоров обычно включаются АЛУ двух типов – целочисленные (с фиксированной запятой) и с плавающей запятой. Функционирование АЛУ осуществляется под управлением центрального устройства управления (ЦУУ) процессора. Для управления работой АЛУ обычно применяется один из двух вариантов: централизованное или децентрализованное (автономное) управление.

Централизованное управление используется в компьютерах класса RISC. При этом работой АЛУ управляет ЦУУ процессора путём выработки управляющих сигналов, обеспечивающих выборку операндов, выполнение операции, сохранение результата.

Децентрализованное управление используется в компьютерах класса CISC. При этом ЦУУ осуществляет выборку операндов из памяти в регистры АЛУ. Выполнением операции управляет не ЦУУ, а встроенное в АЛУ автономное устройство управления, которое, приняв от ЦУУ код «g» операции $f_g \in F$, формирует сигналы, под управлением которых в АЛУ выполняется операция f_g. Запись результата операции f_g в память осуществляется опять под управлением сигналов ЦУУ.

Известно, что АЛУ с автономным устройством управления, ЦУУ процессора, контроллеры различных периферийных устройств относятся к классу операционных устройств (ОУ) [3]. Назначением ОУ является выполнение операций из заданного списка операций $F = (f_1,\ldots,f_G)$ над словами-операндами из списка $D = (d_1,\ldots,d_H)$ с целью получения слов-результатов $R = (r_1,\ldots,r_Q)$, причём в каждый момент времени в ОУ может выполняться одна операция $R = f_g(D^*)$ (рис. 1 *а*).

Организация ОУ базируется на принципе микропрограммного управления [3], в соответствии с которым любая операция $f_g \in F$ рассматривается как сложное действие и разделяется на совокупность элементарных действий – микроопераций (МО). Совокупность МО задаётся списком: $СМО=(МО_1, \ldots, МО_M)$. Порядок выполнения МО задаётся алгоритмом A_g операции $f_g \in F$. Элементарность микрооперации $МО_m \in СМО$ означает, что для её выполнения достаточно операнды подать на входы специальной (уникальной) комбинационной схемы (КС) и после завершения переходных процессов в ней (после паузы) сохранить (принять) результат с выхода $КС_m$. Пример: двоичный комбинационный сумматор обеспечивает выполнение МО сложения кодов (операндов A и B), подаваемых на входы A, B сумматора: $C_{out}.C: = A + B + C_{in}$, где C – сумма, C_{in} – значение на входе переноса в младший разряд сумматора, C_{out} – значение на выходе переноса из старшего разряда сумматора. Для сохранения (приема) результатов микроопераций с выходов комбинационных схем обычно используются регистры.



Р и с . 1. Операционное устройство АЛУ:
а – функциональная схема; б – временная диаграмма работы

Порядок выполнения МО в процессе выполнения операции $f_g \in F$ задаётся алгоритмом A_g и зависит от значений логических условий (ЛУ). Логическое условие – булева переменная, которая принимает значение истина или ложь в зависимости от значений операндов и результатов МО, формируемых в процессе выполнения алгоритма A_g. Пример ЛУ: переменная Z (Zero) принимает значение 1 (истина), если результат МО равен 0, и значение 0 (ложь) – в противном случае. Логические условия используются в процессе выполнения алгоритма A_g в качестве условий (обычно) альтернативных переходов.

Алгоритм A_g, представленный в терминах МО и ЛУ, называется микропрограммой $МП_g$. Микропрограмма $МП_g$ определяет (задаёт) порядок выполнения микроопераций и проверки логических условий во времени. Совокупность микропрограмм $МП_1,\ldots,МП_G$ определяет (задаёт) функцию ОУ. Так, функцию АЛУ задают МП всех операций из списка операций $F_{АЛУ}$, в который включаются обычно арифметические операции сложения, вычитания, умножения, деления, логические операции и другие.

Итак, для выполнения микроопераций $МО_1,\ldots,МО_M$ используются соответствующие комбинационные схемы $КС_1,\ldots,КС_M$. Для сохранения результатов МО используются регистры. Для управления подачей операндов на входы КС (из различных регистров) используются специальные КС – мультиплексоры. Для управления записью результатов МО (с выходов различных КС) в регистры – демультиплексоры.

Часть ОУ, содержащая перечисленные выше схемы, образует исполнительную часть ОУ, которую принято называть операционным автоматом (ОА) устройства, в частности ОА АЛУ, (рис. 1 *а*). *Назначение ОА* – хранение слов

информации, выполнение микроопераций из списка СМО = ($МО_1$, …$МО_M$) под воздействием управляющих сигналов Y = (y_1, …, y_M) из УА, формирование значений логических условий (осведомительных сигналов) X = (x_1, …, x_L) для УА.

Выполнение операции $f_g \in F$, задаваемой кодом операции «g» на управляющем входе ОУ, осуществляется в последовательности, заданной алгоритмом A_g, представленным в форме $МП_g$ (другими словами, в ОУ по коду g запускается процесс выполнения микропрограммы $МП_g$). Управление процессом выполнения $МП_g$ осуществляется частью ОУ, которая относится к классу конечных автоматов. Ее принято называть управляющим автоматом (УА) АЛУ (рис. 1 *а*). *Назначение УА* – управление работой ОА. Управление осуществляется путём выработки управляющих сигналов $y_m \in Y$ в той последовательности, которая задаётся микропрограммой $МП_g$ операции $f_g \in F$ и значениями осведомительных сигналов X = (x_1, …, x_L) из ОА. УА является управляющей (активной) частью ОУ.

Самый простой способ организации работы ОУ во времени – *синхронный*, при котором ОУ работает тактами. Такт – это фиксированный отрезок времени Т определённой длительности, задаваемый обычно как интервал времени между двумя соседними фронтами тактовых импульсов ТИ, вырабатываемых генератором тактовых импульсов с периодом Т (с частотой F = 1/T) (рис. 1 *б*). Этот отрезок времени (такт Т) отводится на выполнение (одной или нескольких) МО в ОУ и состоит из следующей последовательности этапов:

1) этап выработки управляющих сигналов $y_m \in Y$ в УА,

2) этап выборки операндов и выполнения $МО_m$ в $КС_m$ в ОА,

3) этап формирования ЛУ $x_L \in X$ по результатам $МО_m$,

4) этап занесения результатов $МО_m$ в регистры.

Далее эта последовательность этапов периодически повторяется с периодом Т до завершения процесса выполнения операции $f_g \in F$, т. е. до конца $МП_g$.

Такая организация работы ОУ называется синхронной. Положительный фронт тактового импульса ТИ (рис. 1 б) используется обычно как начало этапа 2, а отрицательный фронт (срез) ТИ – как начало этапа 4. То есть по фронту запускается процесс выбора и подачи операндов на входы КС ОА, выполнения МО в КС и формирования значений ЛУ по результатам МО. А по срезу – процесс сохранения результатов МО и ЛУ в регистрах ОА с последующей выработкой управляющих сигналов в УА.

Продолжительность такта Т при синхронной организации работы ОУ определяется суммой продолжительностей этапов 1, 2, 3, 4 – $\tau_1 + \tau_2 + \tau_3 + \tau_4$, причём для наихудшего случая (максимальных по длительности МО и ЛУ):

$$T = \tau_{yA} + \max(\tau_1, \ldots, \tau_m) + \max(\tau_{X1}, \ldots, \tau_{XL}) + \tau_{сохр.} \qquad (1)$$

Здесь: $\tau_1 = \tau_{yA} = \text{const}$ – время формирования управляющих сигналов $\{y_m\}$ в УА обычно постоянно, не зависит от того, какие управляющие сигналы $y_m \in Y$ вырабатываются в данном такте; $\tau_4 = \tau_{сохр} = \text{const}$ – время занесения результатов в регистры ОА. Время выполнения $МО_1$, …$МО_M$ (по сигналам

$y_1, \ldots, y_M$), а также время формирования ЛУ $x_1, \ldots, x_L$ в общем случае разное (поскольку задержка сигналов в соответствующих КС разная), поэтому продолжительность этапов τ_2, τ_3 определяется как максимум от всех возможных значений.

Основные *технические характеристики* АЛУ – быстродействие и затраты оборудования [4]. Быстродействие определяется количеством операций $f_g \in F$, выполняемых АЛУ в единицу времени: ($V_g = 1/t_g$) и зависит от времени выполнения операции $t_g = n_g \cdot T$, где n_g – количество тактов на выполнение операции $f_g \in F$. Продолжительность такта T = const, количество тактов n_g – величина переменная, зависит от сложности микропрограммы $МП_g$ операции f_g и для сложных операций (умножение, деление) значений операндов.

1 ЛАБОРАТОРНАЯ РАБОТА №1. ОРГАНИЗАЦИЯ ОПЕРАЦИОННОГО АВТОМАТА АРИФМЕТИКО-ЛОГИЧЕСКОГО УСТРОЙСТВА

Цель лабораторной работы – изучение принципов построения операционных автоматов арифметико-логических устройств (ОА АЛУ).

1.1 Теоретические основы лабораторной работы

Перед данной лабораторной работой ознакомьтесь с принципами построения АЛУ [4], а также с его назначением, функциями и структурой «на верхнем уровне». На этом уровне описания АЛУ состоит из операционного автомата (ОА) и управляющего автомата (УА).

1.1.1 Принципы построения операционного автомата арифметико-логического устройства

При построении ОА АЛУ можно использовать различные способы организации, отличающиеся по техническим характеристикам – производительности, быстродействию и затратам оборудования [4].

Под *производительностью* ОА понимается количество МО, выполняемых за один такт. В одном такте (одновременно) могут выполняться только совместимые МО, которые указываются в одном операторе микропрограммы МПg. Их число может меняться от такта к такту (в зависимости от микропрограммы МПg и значений операндов). Следовательно, производительность – это случайная величина, которую можно оценивать либо максимальным значением (равным максимальному числу МО за один такт), либо средним значением. Средняя производительность зависит от числа совместимых МО в операторах МПg, частоты использования операторов и структуры ОА, которая может накладывать ограничения на функциональную совместимость МО.

Быстродействие ОА характеризуется длительностью такта ОА – T_{OA}. Такт ОА – это отрезок времени, необходимый для выполнения микроопераций и формирования значений логических условий, то есть промежуток времени от момента поступления управляющих сигналов на вход ОА до момента выработки значений осведомительных сигналов, соответствующих состоянию ОА. Такт ОА зависит от внутренней структуры комбинационных схем и характеристик логических и запоминающих элементов, используемых в комбинационных схемах и памяти (регистрах) ОА.

Затраты оборудования определяются суммарным числом элементов в памяти ОА, комбинационных схемах, реализующих МО и формирующих значения ЛУ, и зависят от сложности и количества КС и регистров.

Важные характеристики структуры – *регулярность и универсальность*. Регулярной называется структура, состоящая из однотипных частей, одинаковым образом связанных между собой. *Регулярность* структуры проявляется в использовании одинаковых элементов или схем для обработки значений в каждом разряде слова. Структура ОА является максимально регулярной,

если все слова обрабатываются одинаковым образом (одним набором МО) и одновременно с этим все разряды слова обрабатываются одинаково. Чем более регулярна структура, тем проще процесс её производства, что в итоге приводит к уменьшению стоимости изделия.

Универсальность (многофункциональность) структуры – возможность выполнения этой структурой широкого класса функций. Если структура ОА универсальна, то реализация любого алгоритма сводится к перестройке структуры внешними средствами путём микропрограммирования. Определение степени универсальности вызывает трудности, однако несложно выполнить сравнительную оценку универсальности двух структур, то есть определить, какая структура из двух является более универсальной. Чем более универсальна структура, чем шире область её применения. Увеличение степени универсальности структур позволяет сохранить номенклатуру выпускаемых изделий и увеличить объём их выпуска, что приводит к снижению стоимости производства. Эти характеристики структур особенно важны для БИС, так как их проектирование требует значительных затрат, которые окупаются при большом объёме производства. Следовательно, увеличение универсальности структуры приводит к снижению стоимости даже в тех случаях, когда универсальность достигается за счёт введения дополнительного оборудования.

Опыт проектирования показывает, что регулярность и универсальность – взаимосвязанные свойства, поэтому увеличение степени универсальности можно достичь за счёт увеличения степени регулярности структуры.

В зависимости от производительности и затрат оборудования различают ОА с максимальными характеристиками (с максимальной производительностью и максимальными затратами оборудования) – это *I-автоматы*, ОА с минимальной производительностью (одна МО за один такт) – *M-автоматы*, ОА с промежуточными значениями характеристик – *IM-автоматы*.

Структура ОА предопределяется его функцией. *Функция* ОА задаётся в виде трёх множеств:

- множество (список) СМО = ($МО_1$, …, $МО_M$) или Y = (y_1,…,y_M),
- множество (список) ЛУ = СЛУ = ($ЛУ_1$, …, $ЛУ_L$) или X = (x_1,…,x_L),
- множество слов S_1, …, S_K (операндов, результатов, внутренних слов).

Микрооперация описывается *оператором присваивания*:

$$S_i := \varphi_m(S_j,S_n), \qquad (2)$$

где φ_m – некоторая вычислимая функция (например, сложение);

S_j, S_n – её аргументы (слова-операнды);

S_i – значение функции φ_m, вычисленное при заданных значениях $S_j = S_j^*$, $S_n = S_n^*$ и присвоенное слову S_i (S_i – слово-результат).

Выполнение $МО_m \in$ СМО в ОА осуществляется под управлением сигнала $y_m \in Y$, поступающего из УА (рис. 1). Поэтому микрооперация $МО_m$ обычно отождествляется с управляющим сигналом y_m, который возбуждает её выполнение в ОА – y_m:$S_i := \varphi_m(S_j,S_n)$. Поэтому набор Y = ($y_1$,…,$y_M$) рассматривается и как список микроопераций СМО, и как список сигналов Y.

Логическое условие описывается выражением:

$$x_l := \psi_l(S_i), \qquad (3)$$

где ψ_l – булева функция;

S_i – ее аргумент;

x_l – значение функции ψ_l, вычисленное при $S_i = S_i^*$.

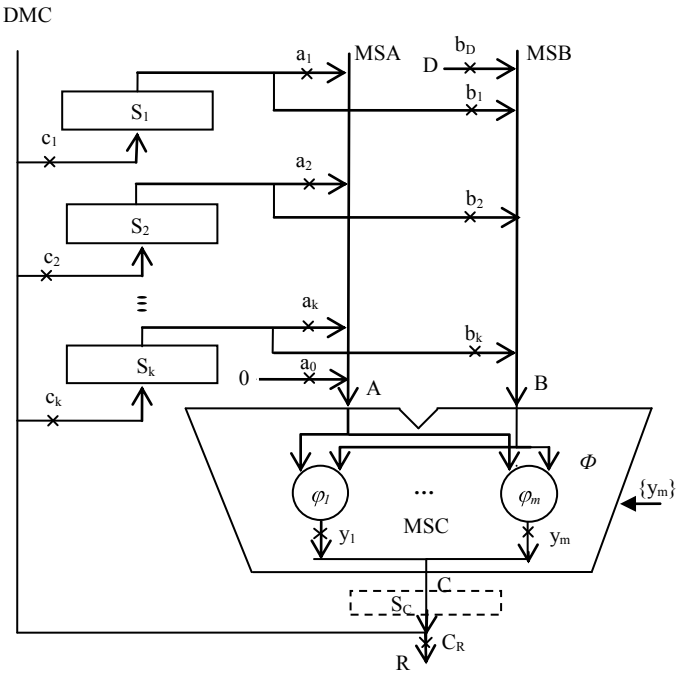
Например, сигнал $x_l = 1$, если значение слова $S_i^* = 0$, или $x_l = 0$, если $S_i^* \neq 0$. Набор значений $X = (x_1,\ldots,x_L)$ отображает состояние ОА.

Для хранения значений слов S_1, …, S_K используются одноименные регистры ОА, количество которых равно К. В этом случае МО вида (1) будет выполняться за один такт. Однако в целях экономии оборудования в ОА используют всего один регистр-аккумулятор (как в ЭВМ с аккумуляторной архитектурой). В этом случае для хранения остальных слов используются ячейки памяти, поэтому в алгоритме операции $f_g \in F$ появляются дополнительные действия: загрузить (операнд в регистр), записать (результат в ячейку), увеличивающие время выполнения операции $f_g \in F$ (количество тактов).

Максимальная производительность ОА первого типа (I-автомата) обеспечивается тем, что его структура организуется в виде, который позволяет в одном такте выполнять все функционально совместимые МО. Для I-автомата характерно, что каждый из регистров S_1, …, S_K обслуживается своей комбинационной схемой Φ_1, …, Φ_K, средствами которых реализуются $МО_!$, …, $МО_K$, вычисляющие значения соответствующих слов. Отсюда максимальная производительность, которая при наличии К регистров и К комбинационных схем Φ_1, …, Φ_K в принципе может достигать К микроопераций за такт.

В структуре I-автомата могут содержаться эквивалентные по функциям КС, используемые для обслуживания различных регистров. Следовательно, затраты оборудования в комбинационной части ОА можно уменьшить, если использовать одну схему для выполнения всех эквивалентных МО. ОА, построенные на основе *принципа обобщения* комбинационных схем, используемых для выполнения МО, называются М-автоматами. В них для вычисления любого двоичного выражения $\varphi_m(S_{\alpha 1}, \ldots, S_{\alpha k})$ используется одна (универсальная) комбинационная схема Ф, равнодоступная по отношению к регистрам S_1, …, S_K ОА. Структура М-автомата представлена на рис. 2.

Операнды, участвующие в $МО_m$, подаются на входы А и В универсальной КС Ф по шинам А, В с помощью мультиплексоров MSA, MSB. Для выборки слов на шину А (для управления схемой MSA) используются сигналы a_1, …, a_K, а для выборки слов на шину В (для управления MSB) – b_1, …, b_K. Сигнал a_i инициирует микрооперацию передачи А: = S_i, а сигнал b_j – микрооперацию В := S_j. Схема Ф на выполнение микрооперации С: =φ_m (A,B) настраивается сигналом y_m, m = 1, … , M, управляющим мультиплексором MSC. Результат $МО_m$ с выхода С схемы Ф заносится в регистр S_n сигналом c_n, управляющим демультиплексором DMC.



Р и с . 2 . Структурная организация М-автомата

Чтобы выполнить микрооперацию $S_n := \varphi_m(S_i,S_j)$, необходимо подать набор управляющих сигналов (a_i, b_j, y_m, c_n), под воздействием которых на входы КС φ_m подаются слова S_i, S_j. Над ними выполняется преобразование φ_m, и результат загружается в регистр S_n. Для выполнения унарной МО (например, микрооперации счета $S_n := S_n+1$) ни один из сигналов b_j не вырабатывается (операнд B = 0) и схема Ф реализует МО счета (увеличение операнда S_n на 1).

Загрузка операндов в регистры ОА извне (с шины D) осуществляется путем их подачи на вход B схемы Ф под управлением сигнала b_D B: = D. Одновременно на вход A подается константа ноль (по сигналу a_0), а в схеме Ф возбуждается МО «ИЛИ» C: = AvB (0vD), результат которой по сигналу c_n записывается в регистр S_n. Выдача результата операции $f_g \in F$ на выходную шину R осуществляется по сигналу c_R.

Как видно из рис. 2, в каждом такте М-автомат может выполнить только одну МО вида $S_n := \varphi_m(S_i,S_j)$, т. е. производительность М-автомата минимальная. По быстродействию М-автомат и I-автомат отличаются незначительно, так как длительность такта М-автомата увеличивается не более чем на 2τ за счет введения в схему мультиплексоров MSA, MSB. Затраты оборудования в М-автомате минимальны, поскольку каждая схема $\varphi_i \in \Phi$ используется для выполнения всех эквивалентных МО.

Для обеспечения устойчивости (*устранения «гонок»*) в М-автомате регистры S_1, … , S_K выполняют на основе двойных триггеров типа MS (Master Slave). Количество оборудования в каждом регистре при этом удваивается.

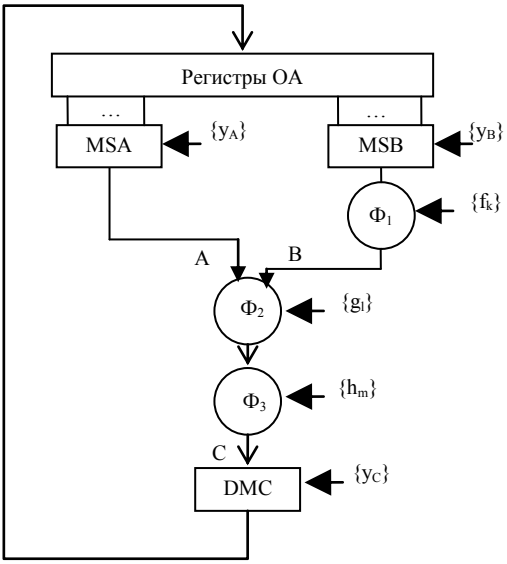
Более экономична схема, в которой в выходную шину С (на входе демультиплексора DMC) устанавливается вспомогательный регистр S_c (одинарный, типа D). Запись в него осуществляется в середине такта (по инверсному сигналу ТИ). Регистры $S_1,\ldots, S_K$ при этом также организуются на более простых одинарных триггерах типа D. Запись информации (обновление) в них осуществляется в начале такта (по прямому сигналу ТИ).

Итак, структурная организация I-автоматов базируется на принципе закрепления комбинационных схем, используемых для выполнения МО, за каждым из регистров $S_1,\ldots, S_K$. За счёт этого все функционально совместимые МО могут выполняться параллельно в одном такте, и их результаты записываются в разные регистры. Структурная организация М-автоматов базируется на обобщении КС по отношению ко всем регистрам, за счёт чего уменьшаются затраты оборудования.

Эти два класса автоматов обладают противоположными свойствами. Для I-автоматов характерны максимальная производительность и наибольшие затраты оборудования, а для М-автоматов – наоборот. Между этими двумя классами структур ОА лежат варианты структур ОА (IM-автоматы), обладающие промежуточными свойствами. ОА, структурная организация которых вносит ограничения на совместимость МО и одновременно с этим обеспечивает выполнение более чем одной МО за такт, называют IM-автоматами. Организация IM-автоматов также базируется на принципе обобщения эквивалентных КС, однако степень обобщения не столь высока, как в М-автоматах, и позволяет в одном такте выполнять более сложные действия, чем $S_i := \varphi_m(S_j)$, $S_i := \varphi_m(S_j,S_n)$, типичные для М-автоматов.

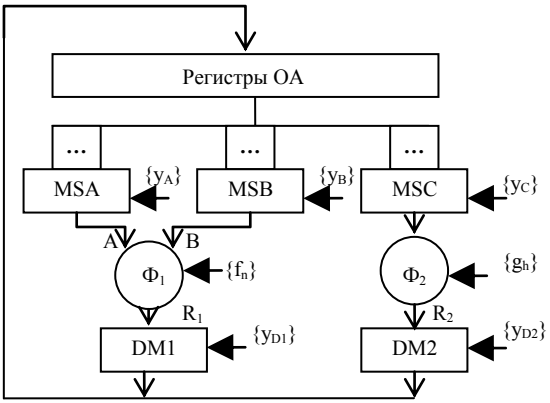
Структурная организация IM-автоматов базируется на использовании (для выполнения МО) последовательного и параллельного соединения многофункциональных КС Φ_1, Φ_2, … . Первое приводит к структурам IM-автоматов с *последовательной комбинационной частью* (рис. 3), а второе – к IM-автоматам с *параллельной комбинационной частью* (рис. 4).

Принцип последовательной организации приводит к структурам, в которых комбинационная часть состоит из обычно из трёх схем Φ_1, Φ_2, Φ_3, реализующих микрооперации из трёх множеств $\{f_k\}$, $\{g_l\}$, $\{h_m\}$ соответственно (рис. 3). Схема Φ_1 обычно служит для формирования констант, полей, инверсий кодов, возбуждаемых одним из сигналов множества $\{f_k\}$, и называется формирователем кодов (для схемы Φ_2). Схема Φ_2 обычно служит для выполнения бинарных операций (например, сложение), возбуждаемых одним из сигналов множества $\{g_l\}$. Схема Φ_3 используется для выполнения унарных МО сдвига, возбуждаемых одним из сигналов множества $\{h_m\}$, и называется сдвигателем. В одном такте такой ОА может выполнить преобразование $S_n := h_m(g_l(S_i,f_k(S_j)))$, эквивалентное трем последовательно выполняемым МО f_k, g_l, h_m. Выбор операндов из регистров S_i, S_j и запись результата в регистр S_n обеспечивается мультиплексорами MSA, MSB и демультиплексором DMC под управлением сигналов их наборов $\{y_A\}$, $\{y_B\}$, $\{y_C\}$. В результате производительность IM-автомата в три раза превышает производительность М-автомата. Однако продолжительность такта больше вследствие последовательного соединения схем Φ_1, Φ_2, Φ_3.



Р и с . 3. Структура IM-автомата
с последовательной комбинационной частью

Принцип параллельной организации приводит к структурам, в которых комбинационная часть состоит из нескольких (например, двух) схем Φ_1, Φ_2, реализующих микрооперации из $\{f_n\}$, $\{g_h\}$ (рис. 4). Схема Φ_1 обычно служит для выполнения бинарных операций, возбуждаемых одним из сигналов множества $\{f_n\}$. Схема Φ_2 используется для выполнения унарных МО, возбуждаемых одним из сигналов множества $\{g_h\}$. В одном такте такой ОА может выполнить две МО – $S_n := f_n(S_i,S_j)$, $S_r := g_h(S_k)$. Выбор операндов из регистров S_i, S_j и запись результата в регистр S_n обеспечивается мультиплексорами MSA, MSB, MSC и демультиплексорами DM1, DM2, управляемыми сигналами из наборов $\{y_A\}$, $\{y_B\}$, $\{y_C\}$, $\{y_{D1}\}$, $\{y_{D2}\}$. В результате производительность такого IM-автомата в два раза превышает производительность M-автомата. IM-автомат с параллельной организацией можно считать композицией из нескольких (двух и более) M-автоматов, имеющих общую память $S_1, \ldots, S_K$.



Р и с . 4. Структура IM-автомата с параллельной комбинационной частью

1.1.2 Схема лабораторного макета операционного автомата арифметико-логического устройства

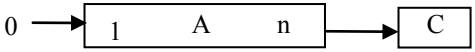
В качестве примера рассмотрим IM-автомат с последовательной комбинационной частью, функции которого заданы списками МО, ЛУ, количеством слов К и их разрядностью n в памяти ОА.

Список арифметико-логических микроопераций $Y_1 = \{y_m\}$, $m \in (1, \ldots, 8)$, A(1:n), B(1:n) − слова-операнды, F(1:n) − слово-результат:

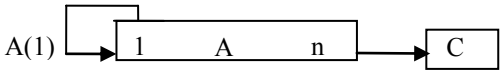
1) F:=A+B+C_{in} – сложение, C_{in} – вход переноса в младший разряд сумматора;
2) F:= A-B-1+C_{in} – вычитание (сложение с дополнительным кодом B);
3) F:= B-A-1+C_{in} – вычитание;
4) F:= A v B – логическое сложение (дизъюнкция);
5) F:= A&B – логическое умножение (конъюнкция);
6) F:= $\overline{A} \vee B$ – инверсия, если B = 0;
7) F:= $A \oplus B$ – исключающее ИЛИ;
8) F:= A $\oplus$B – сравнение на равенство.

Список МО сдвига $Y_2 = \{y_k\}$, $k \in (9, \ldots, 16)$, (A(1:n) − операнд, B(1:n) − результат, C – бит переноса, сформированный ранее и занесенный в триггер C регистра признаков результата):

1) C.B: = C.A – нет сдвига;
2) B.C: = R1(0.A) – сдвиг вправо на 1 разряд с обнулением старшего разряда и с сохранением младшего разряда в C (логический сдвиг вправо);

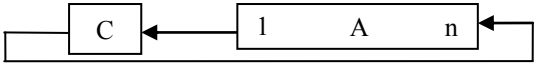


3) B.C: = R1(A(1).A) – сдвиг вправо на 1 разряд с сохранением старшего разряда (знака) A(1) и младшего разряда в C (арифметический сдвиг вправо);

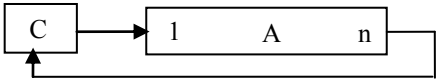


4) C.B: = L1(A.0) – сдвиг влево на 1 разряд с сохранением старшего разряда в C и обнулением младшего разряда;

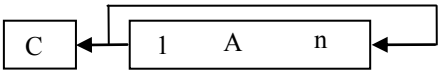
5) C.B: = L1(A.C) – циклический сдвиг влево на 1 разряд через C;



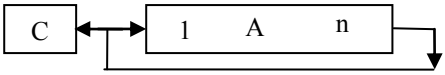
6) B.C: = R1(A.C) – циклический сдвиг вправо на 1 разряд через C;



7) C.B: = L1(A.A(1)) − циклический сдвиг влево на 1 разряд с сохранением старшего разряда A(1) в C;



8) B.C: = R1(A(0).A) – циклический сдвиг вправо на 1 разряд с сохранением младшего разряда A(n) в C;



Список (типичных) логических условий:

1) Z(zero) − признак (флаг) нулевого результата. Значение Z: = 1, если результат F арифметико-логической МО равен 0, иначе (т. е. если F≠0) Z: = 0;

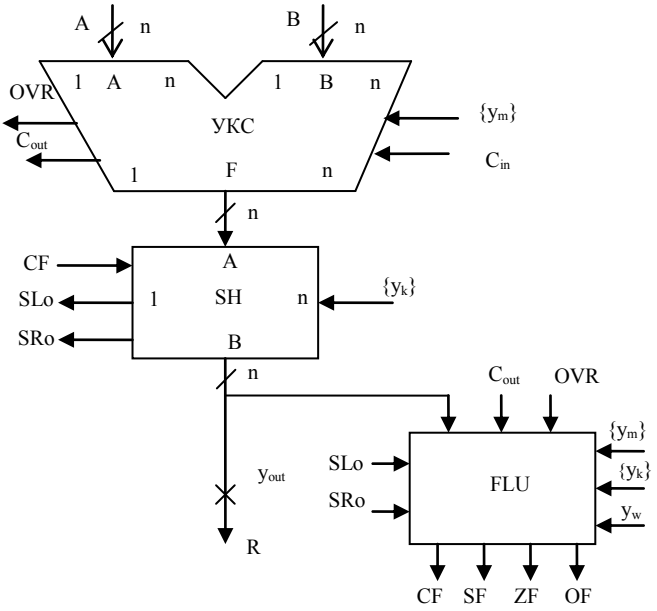
2) S: = F(1) – значение старшего разряда результата F (при обработке чисел со знаками трактуется как знак (sign) результата);

3) OVR – признак переполнения. Значение OVR: = 1, если результат F арифметической МО переполняет n-разрядную сетку;

4) C: = C_{OUT} – значение бита переноса из старшего разряда сумматора при выполнении операций сложения и вычитания (бит Carry).

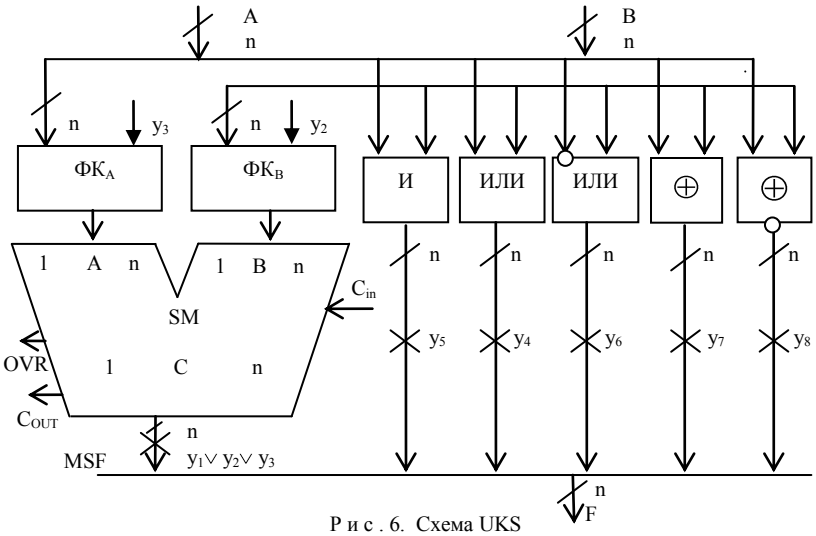
При выполнении арифметических операций формируются все четыре ЛУ, при выполнении логических операций – только два – Z, S, сигналы C, OVR не изменяются. При выполнении операций сдвига могут измениться все четыре ЛУ.

Типичная структура IM-автомата с последовательной комбинационной частью представлена на рис. 3. Память S аналогична памяти M-автомата (на рис. 2). Комбинационная часть ОА АЛУ лабораторного макета (рис. 5) состоит из универсальной комбинационной схемы UKS и сдвигателя SH, соединенных последовательно, а также формирователя логических условий FLU.



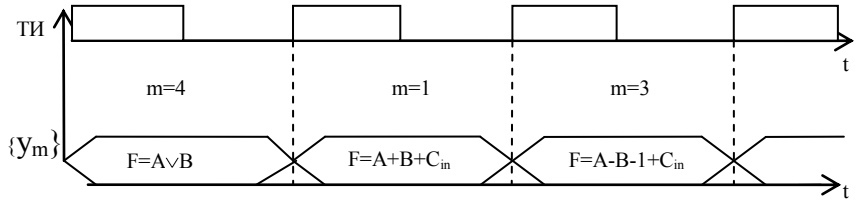
Р и с . 5. Схема комбинационной части ОА

Схема UKS (рис. 6) обеспечивает выполнение МО из списка арифметико-логических операций под воздействием управляющих сигналов $y_1, \ldots, y_8$, поступающих из УА АЛУ на входы $\{y_m\}$.



Р и с . 6. Схема UKS

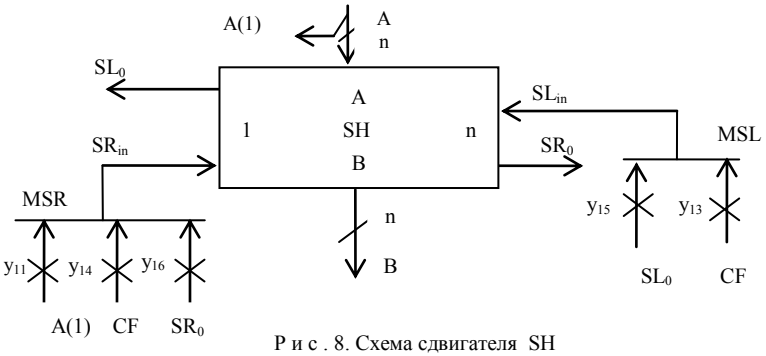
Сигнал y_1 возбуждает МО сложения (первую МО в списке) y_1:F: = A + B + + C_{in}, сигнал y_2 – вторую и т. д. по списку $Y_1 = (y_1, \ldots, y_8)$. Сигналы $y_1,\ldots,y_8$ – это импульсы, продолжительность которых равна длительности такта ОА (остается неизменной, пока идет выборка операндов, выполнение МО, сохранение результата – см. временную диаграмму, рис. 7).



Р и с . 7. Временная диаграмма работы UKS

Схемы $ФК_A$ и $ФК_B$ (формирователи кодов на входах A и B сумматора SM) обеспечивают инвертирование операнда при выполнении операции вычитания по сигналам y_2, y_3 соответственно. Выполнены они на основе двухвходовых элементов «исключающее или» в количестве n штук, на вторые входы которых подаётся сигнал y_2 (или y_3). Крестик на шинах, возле которого указан символ y_m (например, y_5), означает управление шины этим сигналом.

Схема сдвигателя SH (рис. 8) обеспечивает выполнение МО сдвига из списка Y2 под воздействием сигналов $y_9,\ldots,y_{16}$, поступающих из УА АЛУ на входы $\{y_k\}$ сдвигателя. Сигнал y_9 возбуждает первую МО в этом списке (нет сдвига), сигнал y_{10} – вторую и т. д. по списку. SL_{in}, SR_{in} – входы сдвигателя, используемые при левом (SL_{in}) и правом (SR_{in}) сдвигах. Выходы SL_0, SR_0 используются соответственно при левом и правом сдвигах.



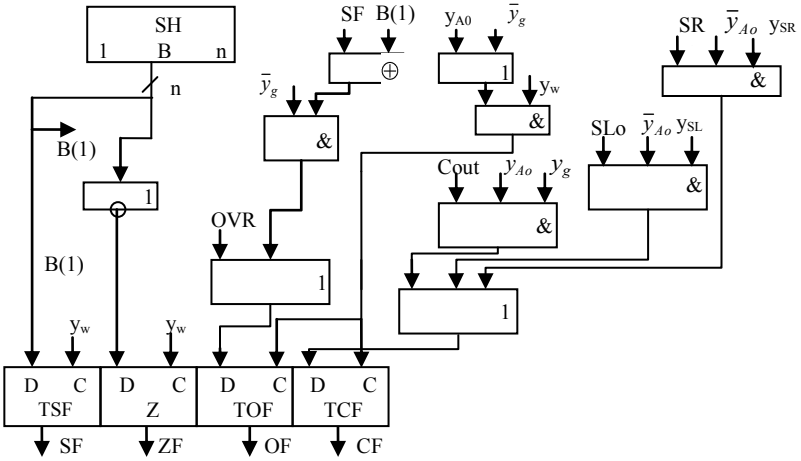
Р и с . 8. Схема сдвигателя SH

Нужная МО сдвига обеспечивается мультиплексорами правого (MSR), левого (MSL) сдвига и мультиплексором MSC, установленным на входе триггера TC, в котором сохраняется значение бита C. Микрооперации, выполняемые мультиплексором (например, y_{11}, y_{14}, y_{16} для MSR), являются несовместимыми и призваны обеспечивать коммутацию одного из входов на единственный выход. Например, сигнал y_{11}=1 подает старший бит A(1) сдвигаемого слова A на вход SR_{in} при выполнении МО y_{11}:B.C:=R1(A(1).A)) – сдвиг правый арифметический.

Формирователь логических условий FLU (рис. 9) обеспечивает формирование сигналов Z, C, S, OVR и сохранение их в регистре флагов RF.

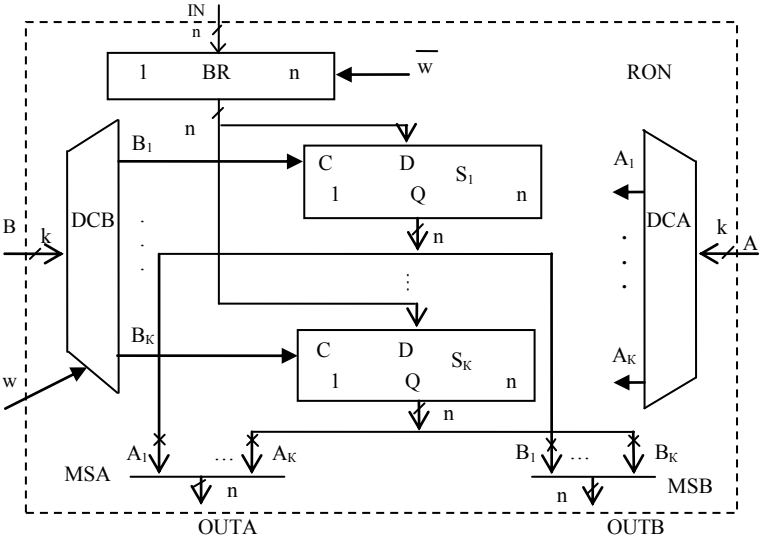
Формирование ЛУ осуществляется по следующим правилам. Если выполняется только АЛО микрооперация (без последующего сдвига ее результата), то формируются и заносятся в RF все четыре ЛУ. Если выполняется логическая микрооперация (без последующего сдвига ее результата), то формируются и заносятся в RF только Z, S (флаги CF, OF не изменяются). Если выполнение МО в УКС совмещается с микрооперацией сдвига, то формирование ЛУ осуществляется на основе результата МО сдвига на выходе сдвигателя. Флаг OF устанавливается в единицу, если в результате сдвига изменился знаковый (старший) разряд результата.

При выполнении арифметических МО формируется сигнал $y_{AO} = y_1 \vee y_2 \vee y_3$. Сигналы $y_{SL} = y_{12} \vee y_{13} \vee y_{15}$, $y_{SR} = y_{10} \vee y_{11} \vee y_{14} \vee y_{16}$ формируются при выполнении соответствующих МО левого и правого сдвига. Для формирования сигнала Z используется схема ИЛИ-НЕ. Сигнал S определяется значением B(1), т. е. S = B(1). Сигналы OVR и C_{OUT} формируются в сумматоре SM.



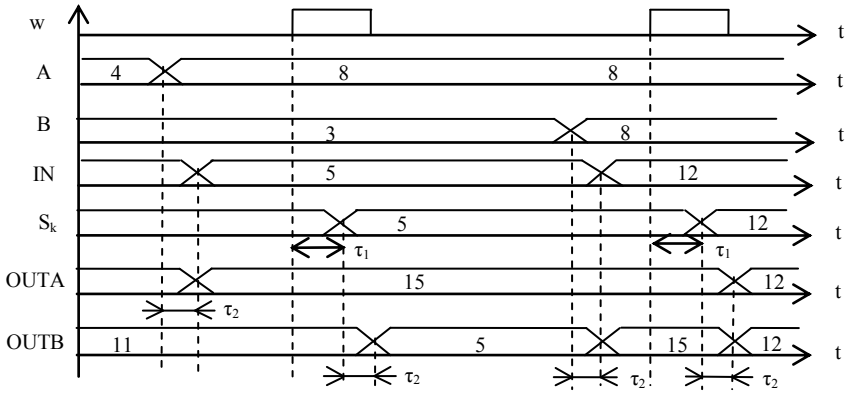
Р и с . 9. Схема FLU

Память S ОА (регистры $S_1,\ldots,S_K$), см. рис. 10, выполнена в виде блока RON (регистров общего назначения – РОН). Доступ к ячейкам (регистрам) осуществляется по адресам (номерам) регистров 1, ..., К, подаваемым на адресные входы A, B. Такая организация эквивалентна схеме на рис. 2. Но в ней для экономии элементов регистры $S_1,\ldots,S_K$ выполнены на основе одинарных (синхронизируемых) D-триггеров, а для обеспечения устойчивости ОА (на входе IN блока РОН) поставлен буферный регистр BR, также построенный из одинарных D-триггеров.



Р и с . 1 0. Структура блока РОН ОА АЛУ

Запись слова со входа IN (по адресу на входе B) осуществляется по сигналу записи W демультиплексором, который возбуждается сигналами B_1, …, B_K с выходов дешифратора DCB, подключенными к входам синхронизации C регистров S_1, …, S_K соответственно. Сигнал записи W передаётся на вход C адресуемого регистра в виде импульса B_k с соответствующего выхода DCB, то есть сигнал записи W возбуждает МО записи: $W:S_k$: = IN (или W:[B]: = IN).



Р и с . 1 1. Временная диаграмма работы РОН ОА АЛУ

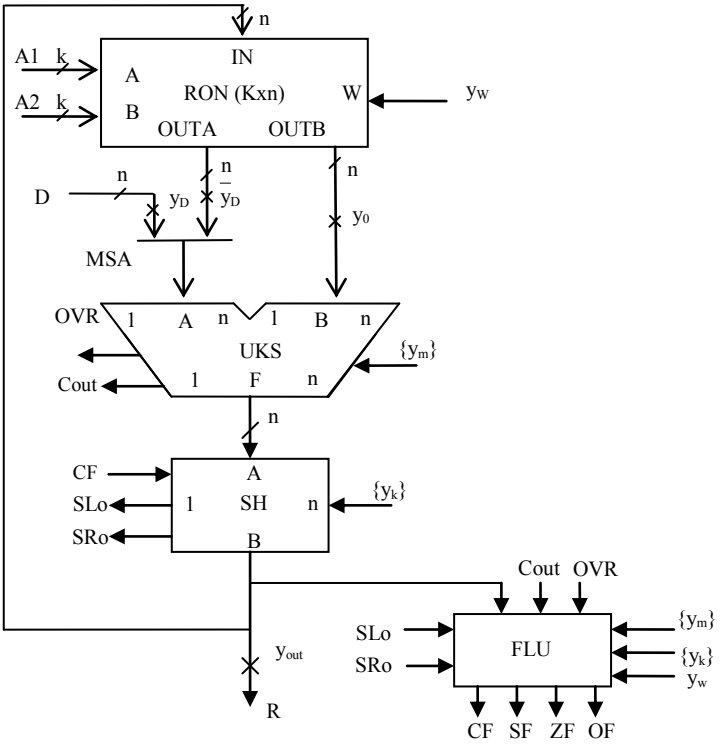
Таким образом, список МО записи результатов в регистры (S_1,…, S_K) $c_1:S_1$: = IN, …, $c_K:S_K$: = IN, возбуждаемых сигналами c_1,…,c_K в структуре ОА, представленной на рис. 2, в блоке РОН заменяется одной МО записи вида «W:[B]: = IN», возбуждаемой сигналом W. Время выполнения МО записи $\tau_{зап} = \tau_{DC}+\tau_{TP}=\tau+3\tau = 4\tau$. Чтение слов из регистров (выдача их на выходы OUTA, OUTB) осуществляется с помощью мультиплексоров MSA, MSB, управляемых сигналами A_1, …, A_K, B_1, …, B_K с выходов дешифраторов DCA, DCB, на входы которых подаются адреса A, B регистров.

Микрооперации передачи слов на шины A, B a_1:A: = S_1, …, a_K:A: = S_K, b_1:B:=S_1, …, b_K:B:=S_K, возбуждаемые сигналами a_1, …,a_K, b_1, …,b_K в структуре ОА (рис. 2), в блоке РОН заменяются двумя МО вида – OUTA:=[A], OUTB:=[B]. Изменение адресов на входах A, B изменяет значения слов на выходах OUTA, OUTB с задержкой $\tau_1+\tau_2$ – на выходе OUTB (относительно сигнала W), и с задержкой τ_1 – по отношению к моменту изменения адреса на адресном входе (рис. 11). Это значит, что на выход OUTA (OUTB) выдается значение слова S_k, k=A, (k=B) до тех пор, пока не изменится адрес на входе A (входе B) либо пока не изменится содержимое регистра S_k в момент записи по адресу B. $\tau_2=3\tau$ – это задержка в дешифраторе адреса и мультиплексоре MSA (MSB), $\tau_1=\tau_{зап}=4\tau$ – это время переключения триггеров регистра при записи, где τ – время переключения логического элемента.

Укрупненная схема лабораторного макета ОА АЛУ показана на рис. 12.

ОА АЛУ выполняет МО двух типов. Микрооперация первого типа эквивалентна $S_n = \varphi_m(S_i,S_j)$, возбуждаемой (рис. 12) сигналами a_i, b_j, y_m, c_n. Она возбуждается сигналами y_0, y_D, обеспечивающими подачу операндов A, B на входы A, B схемы UKS, сигналом $y_m \in Y_1$, по которому выполняется АЛО МО, сигналом $y_k \in Y_2$, обеспечивающим МО сдвига, и сигналом $y_W = 1$, по которому результат МО записывается в регистр A2. При $y_W = 0$ результат МО (второго типа) не записывается в регистр (NOP). Таким образом, запись результата в регистр осуществляется в каждом такте работы ОА, за исключением тех тактов, в которых, например, выполняются микрокоманды перехода.

Регистр RF выполнен на основе двойных триггеров типа MS. Запись признаков результатов S, Z, OVR, C в регистр флагов RF осуществляется (как в РОН) по сигналу y_W (за исключением пустых тактов – NOP в ОА, когда $y_W = 0$). Отметим, что запись флагов Z, OVR осуществляется только при выполнении арифметических МО. Запись флага S осуществляется при выполнении арифметических и логических МО, а запись флага C – при выполнении арифметических МО и соответствующих МО сдвига.



Р и с . 1 2. Схема лабораторного макета ОА АЛУ

Регистр RF выполнен на основе двойных триггеров типа MS. Запись признаков результатов S, Z, OVR, C в регистр флагов RF выполняется (как в РОН) по сигналу y_W (за исключением пустых тактов). Следует отметить, что запись флагов Z, OVR осуществляется только при выполнении арифметических МО. Запись флага S осуществляется при выполнении арифметических и логических МО, а запись флага C – при выполнении арифметических МО и соответствующих МО сдвига.

По завершении операции $f_g \in F$ в АЛУ ее результат выдается на выходную шину R под управлением сигнала y_{OUT}: R: = B (B – выход схемы SH).

Итак, для настройки ОА (рис. 12) на нужные действия на его управляющие входы необходимо подавать сигналы из УА АЛУ. Полный набор сигналов (формат микрокоманды для ОА) представлен на рис. 13.

C_{in}	y_0	y_D	1 ALO 3	1 SH 3	y_W	y_{OUT}	1 A1 k	1 A2 k

Р и с . 1 3. Формат микрокоманды ОА АЛУ

Поля (биты) y_0 ,y_D обеспечивают подачу операндов на входы A, B UKS в четырех комбинациях (табл. 1).

Поле ALO задает АЛО МО $y_m \in Y_1$, список которых приведен в табл. 2. Поле SH задает МО сдвига $y_k \in Y_2$, список которых приведен в табл. 3. В поле C_{in} размещается значение бита переноса (на входе C_{in} сумматора), которое используется при выполнении операций сложения и вычитания.

В поле y_W помещается 1, если результат нужно записать в РОН по адресу A2, указанному в поле A2. Поля A1, A2 используются для указания местоположения операндов, извлекаемых из памяти ОА. Поле A2, кроме того, используется и для записи результата (формат МК – двухадресный). Разрядность k полей A1, A2 зависит от количества K регистров в памяти ОА, остальные поля (слева от A, B) имеют фиксированную длину.

Поля (биты) y_W ,y_{OUT} обеспечивают выдачу результатов (с выхода схемы SH) на выход R в четырех комбинациях (табл. 4).

Т а б л и ц а 1

Источники операндов

y_0	y_D	A	B
0	0	OUT A	0
0	1	D	0
1	0	OUT A	OUT B
1	1	D	OUT B

Максимальная производительность ОА – две МО (АЛО и сдвига) за 1 такт, продолжительность такта ОА:

$$T_{OA}=\tau_{BO}+\tau_{UKS}+\tau_{SH}+\tau_{POH}, \qquad (4)$$

Т а б л и ц а 2

АЛО микрооперации

Код ALO	Микрооперация
000	$F = A + B + C_{in}$
001	$F = A – B – 1 + C_{in}$
010	$F = B – A – 1 + C_{in}$
011	$F = A \vee B$
100	F = A&B
101	$F = \neg A + B$
110	$F = A \oplus B$
111	$F = A + \neg B$

Таблица 3

МО сдвига

Код SH	Микрооперация
000	C.B = C.A (NOP)
001	B.C = R1(0.A)
010	B.C = R1(A(1).A)
011	C.B = L1(A.0)
100	C.B = L1(A.C)
101	B.C = R1(A.C)
110	C.B = L1(A.A(1))

Таблица 4

Приемники результата

y_W	y_{OUT}	Микрооперация
0	0	NOP
0	1	R = B
1	0	[A2] = B
1	1	[A2] = B, R = B

где $\tau_{BO} = \tau_{OUT}+\tau_{MS}$ – время выборки (чтения) операндов из РОН и подачи их на входы UKS;

τ_{UKS} – максимальная задержка в схеме UKS;

τ_{SH} – максимальная задержка в схеме SH;

τ_{POH} – время записи результата в РОН.

1.2 Задание на самостоятельную работу

1. Изучить схему и операционные возможности ОА АЛУ.

2. Разработать микропрограмму в соответствии с вариантом задания, указанным преподавателем. Варианты заданий представлены в табл. 5.

Выполнить микропрограмму в шаговом режиме. Результаты выполнения представить в виде временной диаграммы и трассы в виде таблицы.

Таблица 5

Микропрограммы для вариантов задания

Номер варианта	Задание	Источники операндов	Знаки и разрядность операндов			Особенности результата
			N1	N2	n	
1	*2*	*3*	*4*	*5*	*6*	*7*
1	R = N1 - 2N2	OUT A, OUT B	-	+	4	Переполнение
2	R = 2N1 + N2	OUT B, D	+	+	8	Переполнение
3	R = 2N1 - N2/2	OUT A, OUT B	+	+	12	Больше 0
4	R = 2N1 + N2/2	OUT B, D	-	+	16	Меньше 0
5	R = 4N1 - N2	OUT A, OUT B	+	+	4	Равен 0
6	R = N1 - N2/2	OUT A, OUT B	+	-	8	Больше 0
7	R = N1/2-N2/4	OUT B, D	-	-	12	Меньше 0
8	R = 2N1 - N2/4	OUT B, D	+	-	16	Больше 0
9	R = 2N1 - 4N2	OUT A, OUT B	-	+	4	Переполнение
10	R = 2N1⊕N2/2	OUT A, OUT B			8	R=11111111
11	R = 4N1&N2	OUT B, D			12	Не равен 0
12	R = N1/20v4N2	OUT B, D			16	Не равен 0
13	R = 3N1 - N2/2	OUT A, OUT B	-	+	4	Переполнение
14	R = (5N1)&N2	OUT A, OUT B			8	Не равен 0
15	R = 2N1 - N2/2	OUT B, D	+	-	12	Переполнение
16	R = 2N1⊕N2/4	OUT B, D			16	Равен 0
17	R = 3N1 + N2/2	OUT B, D	-	+	4	Меньше 0

Окончание табл. 5

18	R = 5N1 - N2	OUT A, D	-	-	8	Больше 0
19	R = 3N1 - 5N2	OUT A, OUT B	+	+	12	Больше 0
20	R = 2N1 - 3N2	OUT A, OUT B	-	+	16	Переполнение
21	R = N1/2 - 3N2	OUT B, D	+	+	4	Равен 0
22	R = 8N1⊕N2	OUT B, D			8	R=11111111
23	R = 2N1 + 3N2	OUT A, OUT B	-	-	12	Переполнение
24	R = 5N2 - N1/2	OUT A, OUT B	+	-	16	Переполнение

3. Определить задержки сигналов от входов до выходов ОА и продолжительность такта T (по максимальной задержке).

4. Написать отчет, который должен содержать задание, микропрограмму, временную диаграмму и трассу выполнения микропрограммы.

Проект по лабораторной работе №1 находится в папке «Лаб работы `Организация АЛУ` - Примеры выполнения программ для лаб работы\2011\oa_full.dir». В нем для компиляции рекомедуется использовать файл ..\oa_full.gdf. Перед компиляцией рекомендуется выбрать опцию Assign | Device | FLEX10KA. Микропрограмму рекомендуется задавать в «полуавтоматическом режиме» в редакторе волновых фронтов, выбрав предварительно файл ..\oa_full.scf.

1.3 Содержание отчета

Отчет должен содержать цель работы, постановку задачи, блок-схему проектируемой схемы, блок-схему проекта, созданного в ходе выполнения лабораторной работы, а также временные диаграммы с заданными параметрами сигналов. Следует записать длительность периода тактовых импульсов. Также отчет должен содержать значение максимальной рабочей частоты проекта. Следует сделать выводы по результатам работы.

1.4 Контрольные вопросы

1. Какой принцип лежит в основе построения операционных устройств (в частности, АЛУ)? Поясните основные положения этого принципа и организацию ОУ.
2. Назначение и основные характеристики АЛУ.
3. Организация работы ОУ во времени. Поясните понятия: такт работы, синхронная работа с постоянной длительностью такта.
4. От чего зависит и как определяется продолжительность такта?
5. Назначение и основные характеристики ОА.
6. Поясните понятия МО и ЛУ. Описание и реализация МО и ЛУ.
7. Функция и структура операционного автомата. Способы построения ОА.
8. Краткая характеристика I-автоматов, M-автоматов, IM-автоматов.
9. Поясните организацию и порядок работы лабораторного макета ОА АЛУ.
10. Поясните организацию блока РОН ОА АЛУ. Почему у него два информационных выхода, а вход – один?
11. Какие сигналы (входы) предназначены для управления работой блока РОН?

12. Объясните смысл и назначение сигналов S, Z, OVR, C схемы ОА.
13. Какие сигналы (входы) предназначены для настройки комбинационной части ОА на выполнение требуемых действий?
14. Какие сигналы (входы) предназначены для настройки ОА АЛУ на выполнение требуемых действий?
15. Поясните формат и назначение полей микрокоманды, предназначенной для управления ОА АЛУ.
16. Как оценить продолжительность такта T ОА АЛУ?
17. Поясните порядок выполнения работы на примере первого варианта задания.

2 ЛАБОРАТОРНАЯ РАБОТА №2. ОРГАНИЗАЦИЯ УПРАВЛЯЮЩЕГО АВТОМАТА АРИФМЕТИКО-ЛОГИЧЕСКОГО УСТРОЙСТВА

Цель лабораторной работы – изучение принципов построения управляющего автомата арифметико-логического устройства (УА АЛУ).

2.1 Теоретические основы лабораторной работы

2.1.1 Принципы построения управляющих автоматов

Функция УА определяется совокупностью закодированных графов микропрограмм $\Gamma_1, \ldots, \Gamma_G$. *Закодированный граф* Γ_g получается из содержательного графа $МП_g$ путем замены микроопераций $МО_1, \ldots, МО_M$, указанных в операторных вершинах $МП_g$, символами $y_1, \ldots, y_M$ и замены логических условий $ЛУ_1, \ldots, ЛУ_L$, содержащихся в условных вершинах графа, символами $x_1, \ldots, x_L$ (кодированием микропрограммы $МП_g$). На основе графов $\Gamma_1, \ldots, \Gamma_G$ можно синтезировать соответственно G управляющих автоматов $УА_1, \ldots, УА_G$, которые обеспечат управление ОА. Однако такое решение получается неэффективным. Графы различных МП обычно содержат некоторое число одинаковых операторных и условных вершин, которые сольются, если построить граф Г, объединяющий в себе графы $\Gamma_1, \ldots, \Gamma_G$. Уменьшение числа вершин в МП влечет уменьшение затрат оборудования в УА. Поэтому функцию УА принято представлять объединенным графом Г.

Метод объединения закодированных графов $\Gamma_1, \ldots, \Gamma_G$ в единый граф Г, содержащий минимальное число вершин, изложен в работе [5].

В объединенной микропрограмме пути развития процессов выполнения операций $f_1, \ldots, f_G$ задаются кодом $g = g_1 \ldots g_k$ операции f_g (где $g \in (1, \ldots, G)$ – номер операции f_g), длина которого $k \geq \log_2 G$. Переменные $g_1, \ldots, g_k$ кода g в объединенной МП играют роль логических условий (как и условия $x_1, \ldots, x_L$).

Для формирования управляющих сигналов Y, вырабатываемых в зависимости от значений $g_1, \ldots, g_k$ и $x_1, \ldots, x_L$, можно использовать последовательностную логическую схему – (конечный) автомат с памятью. При этом множество входных сигналов X автомата определяется множеством осведомительных сигналов, поступающих из ОА, и битов $g_1, \ldots, g_k$ кода операции f_g: $X = x_1, \ldots, x_L, g_1, \ldots, g_k = (x_1, \ldots, x_l, x_{l+1}, \ldots, x_L)$, $L = l+k$. Множество выходных сигналов определяется множеством управляющих сигналов $Y = y_1, \ldots, y_M$, возбуждающих микрооперации $МО_1, \ldots, МО_M$ в ОА.

Закон функционирования конечного автомата задается объединенным графом Г и определяет порядок преобразования входной последовательности X(0), X(1), …, X(t) в выходную Y(0), Y(1), …, Y(t), где t – время. Реализация УА на принципе интерпретации микропрограммы автоматом с памятью приводит к построению *УА с жесткой логикой* управления (УА с ЖЛ) [4].

УА можно построить и иначе – на основе *принципа управления по хранимой* (в памяти) *микропрограмме* (принципа Уилкса). Такой способ приводит к *УА с программируемой логикой* управления [4]. В соответствии с этим прин-

ципом для формирования управляющих сигналов используется *микрокоманда* управляющее слово (разделенное на поля определенного назначения и фиксированной длины), которое определяет порядок функционирования (операционного) устройства в течение одного такта.

Микрокоманда (МК) содержит информацию о микрооперациях, которые должны выполняться в данном такте, а также информацию об адресе следующей МК. Совокупность МК, описывающих объединенную МП, образует массив микрокоманд МК[1:P], хранимый в памяти (в ПЗУ).

Простая структура управляющего слова, достаточная для представления (описания) МК, имеет вид:

l Y m	l X h	l A1 p	l A2 p

Поле Y определяет номера микроопераций, возбуждаемых микрокомандой в ОА (в текущем такте). Если поле Y = 0 (пустое), то УА не возбуждает ни одной МО. Номера МО в списке $y_0, y_1, \ldots, y_M, y_{M+1}$ можно кодировать m-разрядным позиционным кодом, $m \geq \log_2(M+2)$, где y_0 – пустая МО, y_{M+1} – сигнал об окончании операции f_g (конец микропрограммы МПg).

Для определения адреса следующей МК можно использовать принудительную адресацию МК (основанную на принципе связного списка). В этом случае в адресной части МК (в полях A1, A2) указываются адреса следующих МК. Адрес следующей МК можно задавать *безусловно* (независимо от значений ЛУ) или *условно* (в зависимости от условия, определяемого значениями осведомительных сигналов). В первом случае адрес следующей МК указывается в поле A1. Во втором случае адрес следующей МК выбирается из поля A1 или из поля A2 адресной части МК, в зависимости от заданного условия.

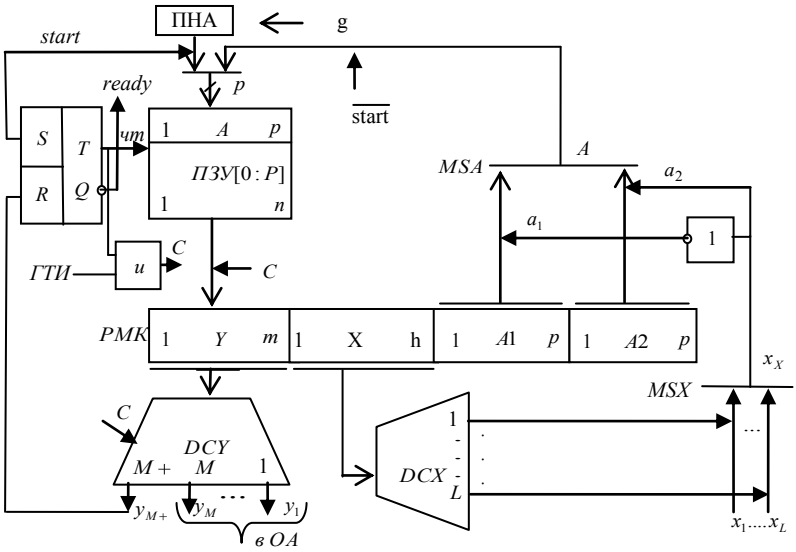
Примем, что в каждой условной МК можно проверять значения только одного ЛУ из множества X. В этом случае адресная часть МК состоит из полей X, A1, A2. В поле X указывается номер $l \in (1, \ldots, L)$ осведомительного сигнала x_l, значение которого анализируется МК условного перехода. Если поле $X = l \neq 0$, то адрес следующей МК определяется в зависимости от значения x_l. Если значение $x_l = 0$, то адрес следующей МК A_{MK} определяется полем A1, если $x_l = 1$ – полем A2. Если поле X = 0 (пустое), то адрес МК равен A1.

В итоге получаем:

$$A_{MK} := \begin{cases} A1, \text{ если } X=0, \\ A1, \text{ если } X \neq 0 \text{ и } x_X = 0 \ (X = l), \\ A2, \text{ если } X \neq 0 \text{ и } x_X = 1. \end{cases} \quad (4)$$

Длина p полей A1, A2 зависит от P – количества МК (емкости ПЗУ): $p \geq \log_2 P$. Длина поля X: $h \geq \log_2(L+1)$.

УА, построенный на основе принципа Уилкса, называется *УА с принудительной адресацией МК* (рис. 14). Для хранения микрокоманд используется ПЗУ емкостью P ячеек, в каждой из них размещается МК длиной n = m + h + +2p разрядов. Работа УА показана на рис. 15. Перед началом работы ЦУУ посылает код операции g в УА АЛУ (на вход g преобразователя начального адреса ПНА микропрограммы МПg).



Р и с . 1 4. Структура УА с принудительной адресацией

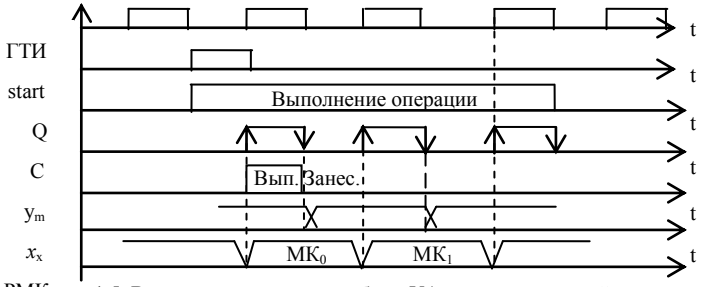
Запуск автомата на выполнение операции f_g производится сигналом «start» (из ЦУУ процессора). По нему триггер Т устанавливается в 1, тем самым разрешается выборка слова из ПЗУ по адресу, сформированному на выходе ПНА, и открывается путь для тактовых сигналов с выхода ГТИ в УА (на выход С). ПНА обеспечивает формирование адреса первой МК микропрограммы МПg. По нему она извлекается из ПЗУ и по первому сигналу С загружается в регистр микрокоманд РМК.

Адрес очередной МК формируется в соответствии с выражением (4). В последнем такте микропрограммы МПg триггер Т сбрасывается сигналом y_{M+1} – конец операции f_g.

Микрокоманда, выбранная из ПЗУ и загруженная в РМК, выполняется следующим образом. Поле Y расшифровывается (дешифратором DCY), и выходной сигнал y_m с выхода DCY (импульс, сформированный по С, по $\overline{C}$ или не стробируемый совсем) поступает в ОА и возбуждает в нем соответствующую МО. Поскольку в каждом такте вырабатывается один сигнал y_m, то в ОА будет выполняться одна МО.

Переход к следующей МК осуществляется в зависимости от значения поля Х и значений ЛУ x_1, …, x_L в соответствии с (4). Если выполняется (в РМК загружена) МК условного перехода, то адрес МК выбирается (мультиплексором адреса MSA) из полей А1 или А2 в зависимости от значения условия x_X одним из сигналов a_1 (А: = А1) или a_2 (А: = А2). Здесь считается, что $x_0 \equiv 0$. Тогда адрес МК берется из поля А1. Выбор одного из условий x_1, …, x_L осуществляется мультиплексором MSX, которым управляет дешифратор DCX. Таким образом, к середине такта на адресном входе А ПЗУ будет сформиро-

ван адрес следующей МК, а к концу такта микрокоманда появится на выходе ПЗУ и по очередному сигналу С будет загружена в РМК.



. 1 5. Временная диаграмма работы УА с принудительной адресацией

Кодирование МО. Микрооперации из списка Y = y_1, …, y_M возбуждаются МК, в которой указываются наименования (коды) микроопераций, выполняемых совместно в ОА. Простой (очевидный) способ кодирования МО состоит в следующем: набору сигналов y_1, y_2, ..., y_M ставится в соответствие слово Y = = y_1, y_2, ..., y_M, в котором каждой МО, если она должна выполняться в данном такте, ставится в соответствие единица, если не должна – то ноль, т.е. производится **унитарное кодирование МО**. Пример: M = 8, такт i+1, y_2 = = y_5 = y_6 =1, остальные биты равны нулю. Это слово используется для управления (элементами И): 1 – элемент И открыт для прохождения импульса с выхода ГТИ, 0 – закрыт (рис. 16). Чтобы в следующем такте можно было вырабатывать другие сигналы, достаточно сменить слово Y на входе этой схемы.

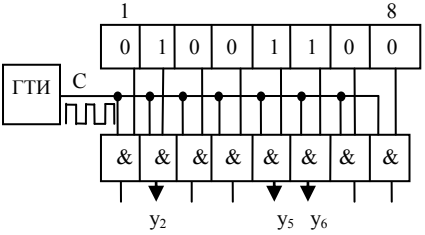
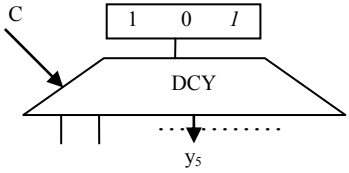


Рис. 16. Унитарное кодирование микроопераций

Недостаток унитарного кодирования – большая длина МК и, следовательно, большая ёмкость ПЗУ для хранения МП. В связи с этим встаёт естественная задача сокращения длины МК.

Простой способ сокращения длины МК – **позиционное кодирование** МО: каждой МО $y_m \in Y$, которая должна выполняться в данном такте, ставится в соответствие m-разрядный позиционный код ($m \geq \log_2 M$). Пример: M = 8, m = 3, сигнал y_5 = 1, остальные сигналы равны нулю, y_5 кодируется кодом 5 Dec = 101 Bin. Сигнал y_5 = 1 на основе этого кода вырабатывается схемой, представленной на рис. 17. Дешифратор DCY стробируется сигналами С с выхода ГТИ, один из которых появляется на выходе y_5. Недостаток позиционного кодирования – в каждом такте можно вырабатывать только один сигнал. Это нежелательно, так как вносит ограничения на совместимость МО и снижает производительность ОА.

Чтобы не вносить ограничений на совместимость МО и сократить длину МК (по сравнению с унитарным кодированием), используется **смешанное кодирование** МО. Для кодирования совместно выполняемых МО в МК выделяются поля Y_1, Y_2, …, Y_I, количество которых определяет предельное количество совместно выполняемых МО. В общем случае поле Y_i может возбуждать некоторое подмножество (несовместимых) МО $Y_i = (y_\alpha, y_\beta, \ldots, y_\omega)$ множества Y, для кодирования которых используется $m_i \geq l\log_2(M_i + 1)$ разрядов, причем значение $Y_i = 0$ является признаком пустого поля (сигнал МО в этом такте не вырабатывается). Сумма $M_O = m_1 + m_2 + \ldots + m_I$ определяет длину операционной части МК. Она влияет на затраты оборудования в УА и на быстродействие ОУ. В частности, уменьшение длины M_O уменьшает разрядность ячеек ПЗУ (т.е. сокращает затраты оборудования в нем). С целью экономии оборудования стремятся уменьшить длину операционной части МК. Существуют различные методы, которые позволяют это сделать.



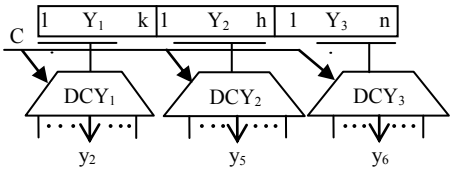
Р и с . 1 7. Позиционное кодирование

Пример смешанного кодирования МО (рис. 18): три группы несовместимых МО – первая состоит из семи МО, вторая – из 9, третья – из 18. Разрядность полей: k = 3 ($\log_2 7$), h = 4 ($\log_2 9$), n = 5 ($\log_2 18$). Всего: 3 + 4 + + 5 = 12 разрядов вместо M = 36 (7 + 9 + 18) при унитарном кодировании.

Количество операционных полей. Если ОА строится на основе принципа обобщения МО, то количество операционных полей равно числу групп несовместимых МО. Так, для управления М-автоматом (рис. 2) нужна МК с четырьмя операционными полями A, B, Y, C. В них указываются двоичные номера МО, принадлежащих наборам $\{a_i\}$, $\{b_j\}$, $\{y_m\}$, $\{c_k\}$ и возбуждаемых в одном такте (совместно выполняемых).

Для управления IM-автоматом с последовательной комбинационной частью (рис. 3) необходима МК с шестью операционными полями A, B, F, G, H, C. В них указываются номера из наборов $\{a_i\},\{b_j\},\{f_k\},\{g_l\},\{h_m\},\{c_n\}$.

Для управления I-автоматом (построенным на основе принципа закрепления МО), который допускает совместное выполнение до H микроопераций, число полей в МК можно выбирать равным K = 1, 2, …, H. В случае K = 1 операционная часть МК имеет минимальную длину, но в каждом такте реализуется только одна МО. В результате чего функциональный оператор микропрограммы, состоящий из K совместимых МО, будет выполняться за K тактов. При K = H любой функциональный оператор (состоящий из K МО) будет выполняться за 1 такт, но операционное поле МК будет иметь суммарную длину, что увеличивает емкость ПЗУ. Таким образом,



Р и с . 1 8. Смешанное кодирование

уменьшение количества операционных полей позволяет экономить оборудование в УА, но одновременно уменьшает быстродействие ОУ (увеличивает количество тактов).

Адресация микрокоманд. Для адресации МК используются два основных способа – *принудительная* и *естественная* адресация.

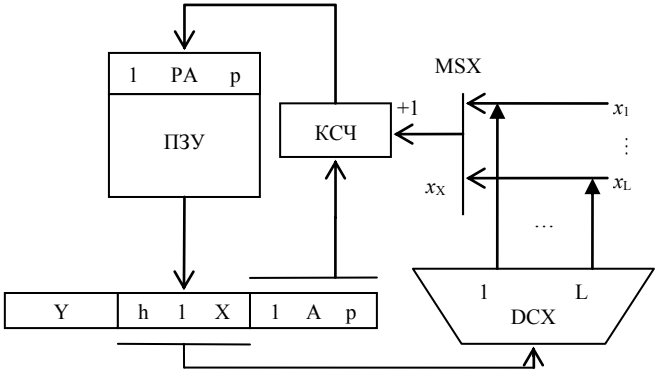
Принудительная адресация МК реализована в УА, структура которого приведена на рис. 14. Адрес следующей МК определяется либо полем А1, либо полем А2 в зависимости от кода в поле Х и значения условия x_X.

Недостаток принудительной адресации – длинная адресная часть МК (и большие затраты памяти). Сократить длину адресной части МК можно, если оставить одно адресное поле А. Сделать это можно следующим образом. Если поле Х = 0, то значение поля А (безусловно) определяет адрес следующей МК – $A_{МК}$: = А. Если же поле Х ≠ 0, то $A_{МК}$: = А + x_X, где x_X – значение ЛУ с номером Х, т.е. в этом случае реализуется условный переход: если x_X = 0, то $A_{МК}$: = А, если x_X = 1, то $A_{МК}$: = А + 1 – на единицу больше А. Для увеличения адреса на 1 можно использовать комбинационный счетчик КСЧ (рис. 19).

Однако введение счетчика КСЧ в схему УА снижает быстродействие автомата, поскольку длительность его такта в этом случае увеличивается на время выполнения операции счета в р-разрядном счетчике.

Естественная адресация МК. При естественной адресации адрес следующей МК формируется путем увеличения на 1 адреса предыдущей МК. В этом случае отпадает необходимость вводить адресное поле А в каждую МК. Если микрокоманды следуют в естественном порядке, то для формирования адреса МК можно использовать (накопительный) счетчик микрокоманд – СМК, состояние которого увеличивается на 1 после чтения очередной МК и дешифрации поля Х:

$$\text{СМК}:= \begin{cases} \text{СМК+1, если } X\neq 0,\ x_X=0 \quad (\text{ЕП}); \\ \text{А, если } X=0 \qquad\qquad (\text{БП}); \\ \text{А, если } X\neq 0,\ x_X=1 \qquad (\text{УП}). \end{cases} \tag{5}$$



Р и с . 1 9. Комбинационный счетчик

Как видно из (5), микрокоманды естественного перехода ЕП (по СМК+1) не используют поле А для формирования адреса МК и поэтому могут содержать только операционную часть, представленную полями $Y_1,\ldots,Y_H$.

При выполнении МК переходов (безусловного – БП, условного – УП) используются поля X, A. Однако их выполнение трудно (не всегда удается) совместить с выполнением МО в ОА (так как по результатам выполненных в текущем такте МО сформируются сигналы $x_1,\ldots,x_L$, которые используются в качестве ЛУ, обычно, в следующем такте). Поэтому операционная часть в этих микрокомандах переходов не используется (поля Y пустые, в ОА нет МО – NOP). В результате при выполнении последовательности из двух функциональных операторов не используется адресная часть по крайней мере в одной МК (т.е., проще говоря, не встречается последовательность из двух идущих подряд команд перехода), чтобы по ее результатам сформировались сигналы $x_1,\ldots,x_L$ (вторую МК можно совместить, например, с БП). При выполнении последовательности операторов перехода не используется (пустует) операционная часть большинства МК.

В этих условиях рационально использовать МК двух типов – *операционные* и *управляющие*.

Операционная МК содержит только поля $Y_1,\ldots,Y_H$ и неявно полагает адрес следующей МК равным СМК+1.

Управляющая МК используется только для изменения естественного порядка выполнения МК (путем выполнения МК безусловного и условного переходов), поэтому содержит только поля X, A.

Для определения (указания) типа МК вводится поле P типа МК:

P	1	Y_1	m_1	…	1	Y_H	m_H

P	1	X	h	1	A	p

Например, если P = 0, то МК является операционной, если P = 1, то – управляющей (или наоборот).

При использовании МК двух типов сокращается длина МК (экономится память), однако увеличивается количество тактов на реализацию операций (микропрограмм), поскольку такты отводятся либо только на выполнение МО в ОА, либо только на выполнение переходов в микропрограмме (пустые для ОА такты). Для уменьшения количества тактов можно не делить МК на два типа, а использовать один формат (как на рис. 19). Экономии памяти в этом случае не будет.

Итак, стремление повысить производительность УА приводит к неэффективному использованию емкости ПЗУ, а стремление к экономии оборудования – к увеличению количества тактов и снижению производительности.

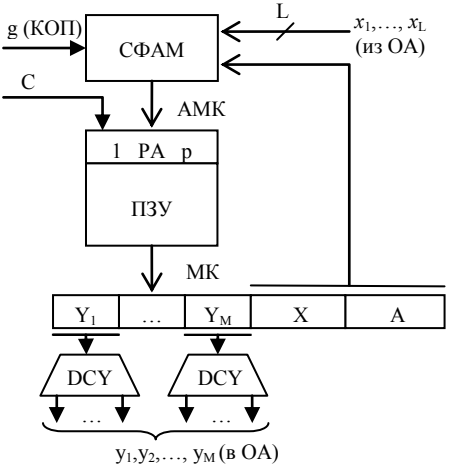
Быстродействие УА. Быстродействие автомата характеризуется продолжительностью такта УА, т.е. временем, затрачиваемым на формирование одного набора управляющих сигналов. Оно складывается из 3 составляющих: 1) времени формирования адреса МК; 2) времени обращения к ПЗУ; 3) времени дешифрации МК.

Основная доля времени приходится на чтение из ПЗУ, поэтому ощутимого увеличения быстродействия УА с программируемой логикой можно достичь путем уменьшения времени обращения к ПЗУ (применяя ПЗУ с более высоким на данный период времени быстродействием) или за счет сокращения количества обращений к ПЗУ.

При фиксированном быстродействии ПЗУ быстродействие УА можно повысить за счет параллельной выборки (нескольких) МК. В этом случае за одно обращение из ПЗУ выбирается K микрокоманд и время обращения к ПЗУ, приходящееся на одну МК, в пределе сокращается в K раз. Ясно, что не все из K выбранных МК будут исполнены (ввиду возможных переходов). Таким образом, из K выбранных МК в среднем реализуется N микрокоманд (где $1 < N < K$), т.е. эффективное быстродействие увеличивается в N раз ($N < K$).

Укрупненная структура УА с ПЛ (рис. 20) состоит из трех частей: схемы формирования адреса следующей МК СФАМ (формирующая адрес, например, в соответствии с выражением (5)), ПЗУ и схемы формирования управляющих сигналов $y_1,\ldots, y_M$, состоящей из дешифраторов $DCY_1,\ldots, DCY_M$.

Временная диаграмма работу УА с ПЛ приведена на рис. 21.



Р и с . 2 0. Схема УА с ПЛ

Продолжительность такта $T_{OY} = T_{YA} + T_{OA}$ работы ОУ определяется суммарным временем T_{YA}, затрачиваемым в УА на формирование адреса – $\tau_{ФА}$, на выборку МК из ПЗУ – $\tau_{ПЗУ}$ и на дешифрацию МК – $\tau_{ДШ}$: $T_{YA} = \tau_{ФА} + \tau_{ПЗУ} + \tau_{ДШ}$, и суммарным временем T_{OA}, затрачиваемым в ОА на выполнение МО – τ_{OA}, на формирование логических условий $\tau_{ЛУ}$ и на запись результатов в регистры ОА – $\tau_{РЕГ}$: $T_{OA} = \tau_{OA} + \tau_{ЛУ} + \tau_{РЕГ}$. При последовательной работе УА и ОА продолжительность такта T_{OY} оказывается большой.

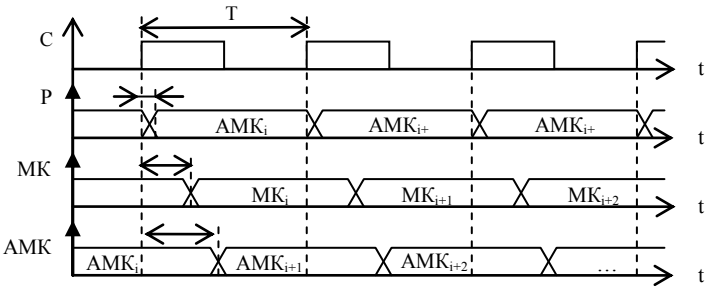
Для уменьшения длительности такта T_{OY} можно использовать *конвейерную организацию* работы ОУ, которая предполагает совмещение во времени работы УА и ОА. С целью совмещения работа ОУ разделяется на этапы, например: 1) выборка МК из ПЗУ; 2) ее реализация. В этом случае УА, выбрав i-ю МК, может (не дожидаясь завершения процесса ее выполнения в ОА) начать выборку следующей (i+1)-й МК.

Чтобы обеспечить возможность *опережающей выборки* МК, очевидно, в состав УА необходимо ввести специальный регистр МК (РМК), в который

заносится выбранная МК. Надобность в РА (как на рис. 20) при конвейерной обработке отпадает. К моменту времени, когда выбранная МК будет реализована, в РМК заносится (i+1)-я МК, выбранная с опережением (рис. 22). Как видно из временной диаграммы на рис. 22, такт $T_{ОУ}$ равен такту работы конвейера $T_К$, продолжительность которого определяется:

$$T_К = \max(T_Р, T_В), \qquad (6)$$

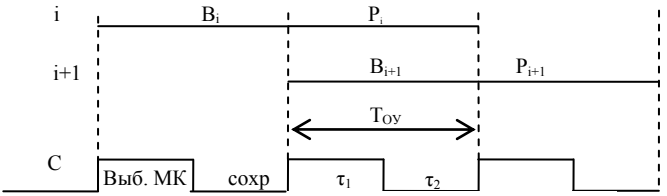
где $T_Р$ – время выполнения i-й МК, $T_В$ – время выборки (i+1)-й МК.



Р и с . 2 1. Временная диаграмма работы УА с ПЛ

Для реализации конвейерного режима работы процессы, протекающие в УА и в ОА, следует разделить на два этапа. В УА к первому этапу относятся: формирование адреса МК и обращение по этому адресу к ПЗУ (выборка МК), а ко второму – занесение МК в РМК и ее дешифрация. В ОА к первому этапу относятся: выборка операндов из памяти ОА и выполнение МО – $\tau_{МО}$, формирование осведомительных сигналов $\tau_{ЛУ}$; ко второму – запись результатов в регистры ОА. Отсюда следует, что продолжительность такта конвейера $T_К$ определяется:

$$T_К = \max[\underbrace{(\tau_{ФА} + \tau_{ПЗУ})}_{УА}, \underbrace{(\tau_{МО} + \tau_{ЛУ})}_{ОА}] + \max[\underbrace{(\tau_{ЗАП} + \tau_{ДЕШ})}_{УА}, \underbrace{(\tau_{РЕГ})}_{ОА}]. \qquad (7)$$



Р и с . 2 2. Опережающая выборка микрокоманд

Первое слагаемое определяет продолжительность первого полутакта τ_1, а второе – второго полутакта τ_2: $T_К = \tau_1 + \tau_2$, задаваемого сигналами С со выхода ГТИ. В первом полутакте УА формирует адрес МК и выбирает ее из ПЗУ, а ОА в это время выбирает операнды, выполняет МО и формирует ЛУ. Во втором полутакте УА заносит выбранную МК в РМК и декодирует ее, а ОА – заносит результаты МО в регистры.

Итак, по фронту сигнала С запускаются первые этапы в УА и ОА, а по срезу – вторые этапы, а сигнал С заводится на РМК.

Следует отметить, что конвейерная организация увеличивает быстродействие ОУ почти вдвое при условии, что продолжительности этапов в ОА и УА приблизительно одинаковые.

Время выполнения операций в ОУ, которое определяется величиной $t_{опер} = nT_{ОУ}$, где n – среднее количество тактов, при конвейерной организации зависит от количества условных переходов в микропрограмме. Точнее, от количества «промахов» при выборке МК, условия перехода для которой еще не «созрели» в ОА. Если при условном переходе выбрана не та МК, то результаты ее выполнения аннулируются (пустой такт в ОА) и потребуется дополнительный такт для выборки МК по альтернативному адресу.

Уменьшить количество «промахов» при выполнении МК условных переходов можно путем предсказания переходов. Простейший способ предсказания – привязать выбор (одного из двух) адреса к типу МК перехода, например, если это обычный альтернативный переход, то вероятность перехода – пятьдесят на пятьдесят. В этом случае можно выбирать следующую МК по условию «нет». Если условный переход используется для организации цикла (МК стоит в конце цикла), то можно его выделить в специальный тип перехода – «конец цикла». В этом случае вероятность перехода в начало цикла выше, чем завершения цикла, поэтому опережающую выборку МК следует выполнять по адресу в начало цикла.

Следует отметить, что микропрограммы различных операций ОУ могут содержать общие по функциям участки, которые целесообразно оформлять как подпрограммы, вызываемые по (микро) команде Call, возврат из которых обеспечивается МК возврата – Ret. В этом случае в СФАМ встраивается регистр, в который по команде Call заносится адрес возврата из подпрограммы (т.е. i + 1, где i – адрес МК Call), а по команде Ret адрес i + 1 извлекается из регистра и выдается на выход СФАМ в качестве $A_{МК}$. Если вызов подпрограммы осуществляется из цикла, при организации которого используется команда типа «конец цикла», и если условие истинно, то в этом случае одного регистра для сохранения адреса возврата недостаточно (нужно как минимум два). Их (регистры) в этом случае рационально организовать как стековое ЗУ.

2.1.2 Схема лабораторного макета управляющего автомата арифметико-логического устройства

При разработке схемы макета выбран ориентированный на конвейерную работу вариант построения УА с естественной адресацией и единым форматом МК, состоящим из полей X, A, $Y_1 \ldots Y_H$. Структура УА представлена на рис. 23. Для реализации конвейерной работы в УА включен регистр МК РМК. Поле X используется для указания типа МК. Всего УА реализует 13 МК, список которых представлен ниже (табл. 6).

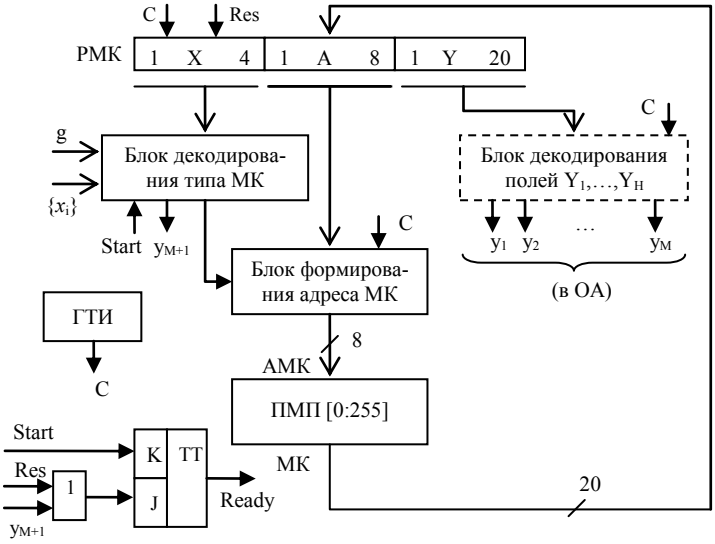
Таблица 6

Список МК

Тип МК	Описание МК
1. cnt	Переход по СМК (АМК = СМК)
2. Jmp	БП по адресу А из РМК (АМК = А)
3. Резерв	Не используется
4. Call	БП на подпрограмму (АМК = А, стек: = СМК)
5. Ret	Возврат из подпрограммы (АМК = стек)
6. Push	Запись в стек (СМК → стек) и переход по СМК
7. Endmp	Конец микропрограммы, формирование Ready
8. Резерв	Не используется
9. JZ	УП по нулю (по Z = 1 СМК: = А)
10. JS	УП по знаку (по S = 1 СМК: = А)
11. JC	УП по переносу (по С = 1 СМК: = А)
12. JOVR	УП по переполнению (по OVR = 1 СМК: = А)
13. EndZ	Конец цикла по нулю (по Z = 1 – выход из цикла и выталкивание стека, по Z = 0 АМК = стек)
14. EndS	Конец цикла по знаку (по S = 1 – выход из цикла)
15. EndC	Конец цикла по переносу (по С = 1 – выход из цикла)
16. Резерв	Не используется

Адрес МК формируется на выходе СМК. Счетчик микрокоманд выполнен на основе трех блоков: СМК, КСЧ, СМК′. Регистр микрокоманд РМК выполнен на триггерах типа MS, т. е. состоит из РМК′ (1-й уровень) и РМК (2-й уровень). В РМК′ микрокоманда заносится по срезу сигнала С, т. е. по $\overline{C}$, а в РМК она переписывается по фронту С.

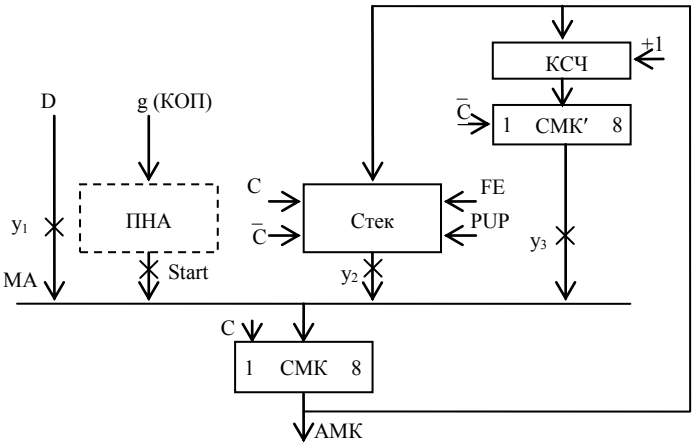
Блок декодирования полей $Y_1 \ldots Y_H$ (пунктирные линии на рис. 23) в лабораторном макете не реализован (так как ОА отсутствует).



Р и с . 2 3. Схема лабораторного макета УА

Блок (схема) формирования адреса МК (рис. 24) содержит накопительный СМК, стек, преобразователь начального адреса ПНА на основе трех блоков: СМК, КСЧ, СМК′. Комбинационный счетчик КСЧ реализует микрооперацию счета ВЫХ: = ВХ + 1, т.е. СМК′: = СМК + 1. Блоки СМК′, СМК выполнены как регистры на синхронных D-триггерах. Адрес МК на вход СМК мультиплексор адреса МА может выдавать из 4 мест:

– со входа D (по сигналу y_1),
– с выхода ПНА (по сигналу Start),
– из (вершины) стека (по сигналу y_2),
– из СМК′ (по сигналу y_3).

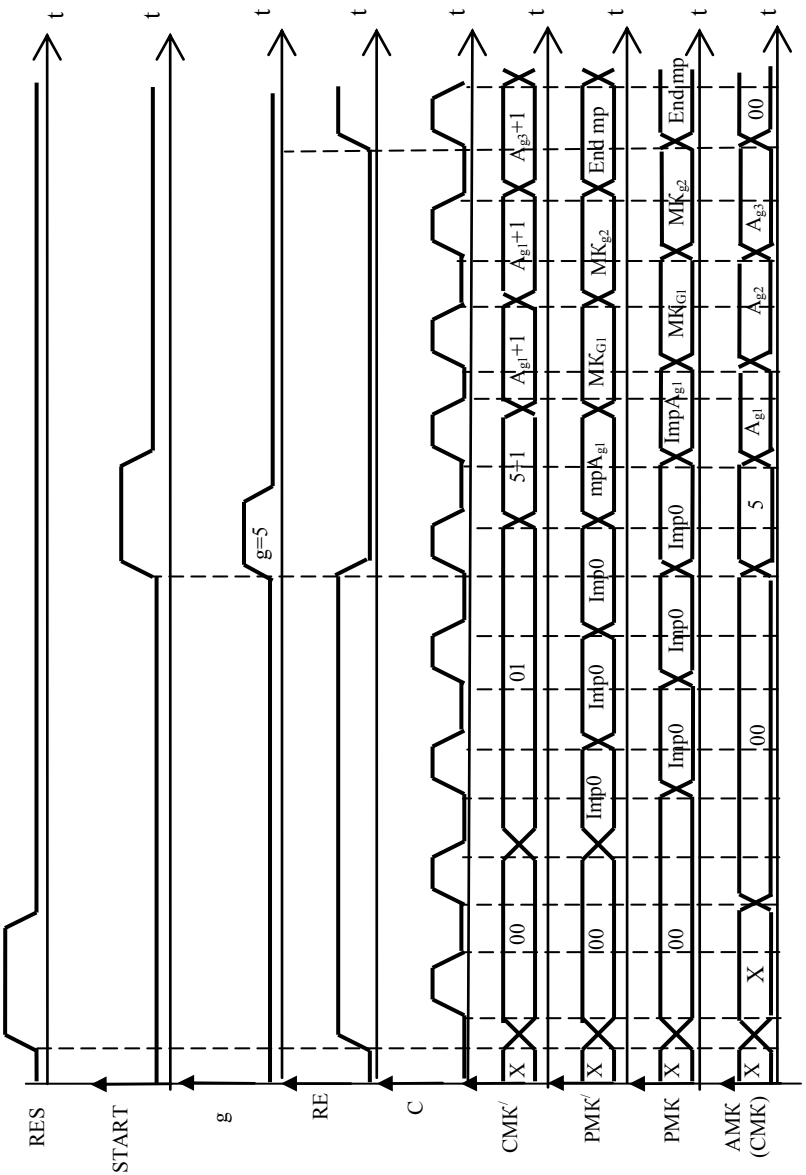


Р и с. 2 4. Схема блока формирования адреса МК

На вход D адрес А подается из поля А РМК. Сигнал y_2 вырабатывается при выполнении МК условного и безусловного переходов. Стек используется для обслуживания МК Call, Ret, Push, EndZ, EndC, EndS (табл. 6). При их выполнении может вырабатываться сигнал y_2 (см. табл. 7). Сигнал y_3 вырабатывается при выполнении МК естественного перехода по СМК.

ПНА обычно реализуется на основе ПЗУ, в ячейках (с адресами $g \in (1, 2, 3, \ldots, G)$) которого размещены пусковые адреса микропрограмм МПg. Однако введение дополнительного ПЗУ в состав схемы ФАМ увеличивает продолжительность такта $T_{УА}$. Чтобы этого не произошло, можно код g использовать как пусковой адрес. Для этого в ячейке ПМП с номером $g \in (1, 2, 3, \ldots, G)$ следует поместить МК безусловного перехода, в адресной части которой указан адрес первой микрокоманды $МКg_1$ микропрограммы МПg. Поэтому в схеме лабораторного макета ПНА отсутствует.

Порядок работы УА представлен временной диаграммой (рис. 25). По сигналу сброса Res УА приводится в исходное состояние (готовности выполнять операции) – сбрасывается РМК, устанавливается сигнал (флаг) готовности Ready (на выходе триггера ТТ). Сброшенный РМК содержит МК «Jmp 0».



Р и с. 2 5. Временная диаграмма работы УА

По первому (после RES) тактовому сигналу С в регистр СМК заносится адрес МК, равный нулю (поскольку РМК сброшен и на входе D адрес А = 0).

По нулевому адресу в ПМП записана МК Jmp с адресом А = 0 − переход на нулевую ячейку. Она извлекается и загружается в РМК. Тем самым начинается (бесконечный) цикл передачи управления на себя. Он прерывается по сигналу Start, и запускается процесс выполнения микропрограммы МПg.

По сигналу Start и коду операции g (поступающим из ЦУУ процессора) начинается процесс выполнения операции f_g (микропрограммы МПg). По сигналу Start сбрасывается триггер ТТ, т.е. сигнал готовности READY = 0, и формируется начальный (пусковой) адрес МПg, по которому из памяти микропрограммы выбирается первая МК микропрограммы МПg и заносится в РМК. По сигналу Start адрес первой микрокоманды $МК_{g1}$ МПg заносится в СМК (по фронту сигнала С) и появляется на выходе АМК. По нему из ПМП выбирается первая $МК_{g1}$ и заносится в РМК′ (по $\overline{C}$), а затем − в РМК (по С). Кроме того, по сигналу $\overline{C}$ в регистр СМК′ заносится АМК (из СМК), увеличенный на 1 в КСЧ.

Во втором такте адрес МК формируется в зависимости от типа первой МК, записанной в РМК. Если эта МК генерирует переход, то в СМК заноситься адрес либо со входа D (из поля А РМК), либо из (вершины) стека. Если это МК естественного перехода, то в СМК′ заносится СМК + 1 с выхода КСЧ. В конце МПg ставится МК Endmp (конец МП), по которой вырабатываются сигналы y_{M+1} (конец алгоритма). Он устанавливает триггер Т и тем самым формирует сигнал READY=1, обнаружив который, ЦУУ сохраняет результат операции f_g в памяти (в соответствии с адресной частью команды).

В макете реализован стек емкостью две ячейки (регистры R0, R1, на основе триггеров типа MS). Двух ячеек достаточно для обслуживания команд Call, Ret, EndZ и др., но недостаточно для написания вложенных подпрограмм, организованных с использованием команд конец цикла. Схема стека и временная диаграмма представлены на рис. 26 и 27 соответственно.

Стек работает следующим образом. После сброса указателя стека УС (по сигналу Res) стек пуст и готов выполнять действия в соответствии с типом МК, находящейся в РМК. Блок дешифрации типа МК (поля Х) вырабатывает сигналы, управляющие схемой ФАМ: y_1, y_2, y_3, FE, PUP. Для управления работой стека предназначены сигналы FE, PUP. Сигнал FE = 1 разрешает работу стека (FE = 0 – запрещает). Если FE = 1, то по сигналу PUP = 1 выполняется запись в стек, а по сигналу PUP = 0 – чтение стека. Запись в стек осуществляется в два этапа:

1) сначала модификация УС: =УС + 1 (в начале такта – по С). Увеличение УС на 1 означает проталкивание стека;

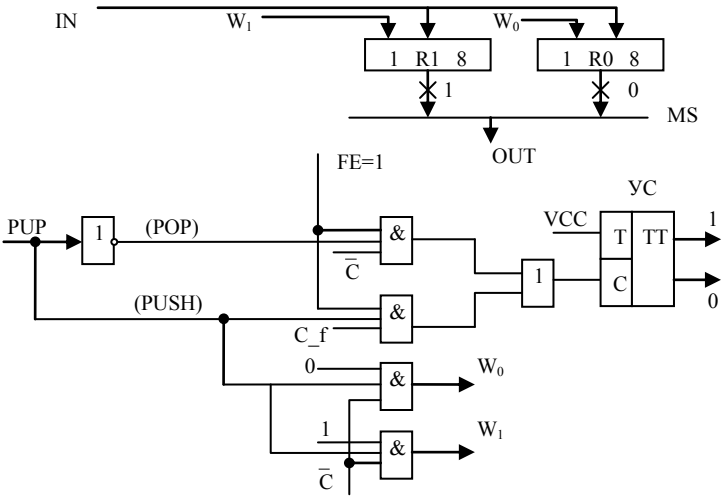
2) затем запись в вершину стека [УС]: = IN (сигналами W_0, W_1 в середине такта – по $\overline{C}$).

Чтение (выталкивание) стека:

1) чтение из вершины стека OUT: = [УС] (постоянно);

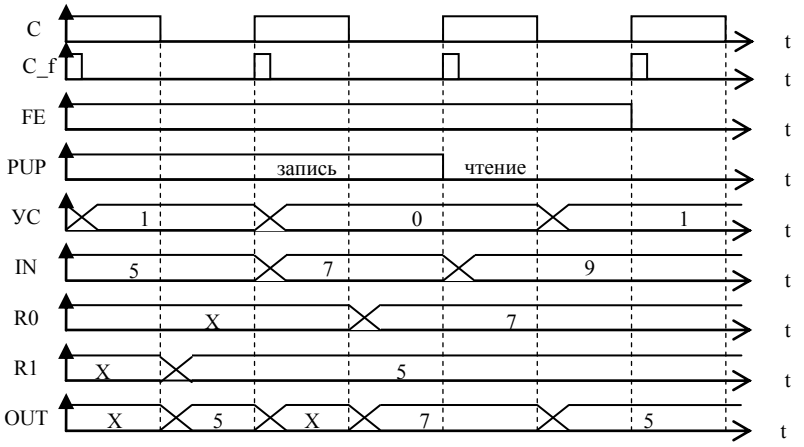
2) модификация УС: = УС – 1 (по сигналу $\overline{C}$) – выталкивание стека.

Чтение из стека (выдача на выход OUT) осуществляется с помощью мультиплексора MS, управляемого сигналами «0», «1» с выхода УС, выполненного на основе триггера со счетным входом (типа ТТ).

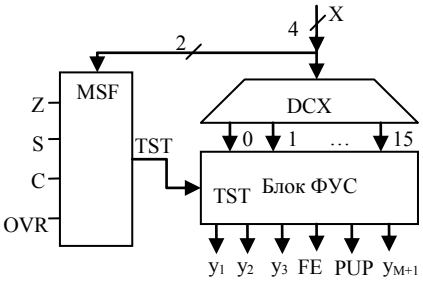


Р и с . 2 6. Схема стека

Блок декодирования типа МК (поля X) (рис. 28) обеспечивает управление схемой ФАМ: коду из поля X и логическим условиям Z, S, C, OVR из ОА ставит в соответствие набор управляющих сигналов y_1, y_2, y_3, FE, PUP, y_{M+1}.



Р и с . 2 7. Временная диаграмма работы стека



Р и с. 2 8. Схема блока декодирования поля X

Формирование управляющих сигналов производится в зависимости от типа МК (кода в поле X) и признаков (флагов) Z, S, C, OVR, вырабатываемых в ОА. Мультиплексор флагов MSF выбирает одно из условий и подает его значение на вход TST блока формирования управляющих сигналов (ФУС). Номер логического условия НЛУ представлен младшими битами поля X. Блок ФУС вырабатывает сигналы в соответствии с табл. 7.

Т а б л и ц а 7

Формирование управляющих сигналов

Тип перехода	Код X		TST	Выходные сигналы (активные)					
	ст. биты	НЛУ		y_1	y_2	y_3	FE	PUP	y_{M+1}
Jmp	00	00	–	1					
Cnt	00	01	–			1			
Резерв	00	10	–						
Call	00	11	–	1			1	1	
Ret	01	00	–		1		1	0	
Push	01	01	–			1	1	1	
Endmp	01	10	–						1
Резерв	01	11	–						
JZ	10	00 (Z)	0			1			
			1 (да)	1					
JS	10	01 (S)	0			1			
			1	1					
JC	10	10 (C)	0			1			
			1	1					
JOVR	10	11 (OVR)	0			1			
			1	1					
EndZ	11	00	0		1				
			1			1	1	0	
EndS	11	01	0		1				
			1			1	1	0	
EndC	11	10	0		1				
			1			1	1	0	
Резерв	11	11	–						

2.2 Задание на самостоятельную работу

1. Изучить схему и операционные возможности УА АЛУ.

2. В соответствии с указанным преподавателем вариантом задания и пусковым адресом микропрограммы разработать микропрограмму, обязательно содержащую (использующую) указанные в задании типы микрокоманд переходов. Варианты заданий представлены в табл. 8. Выполнить микропрограмму в шаговом режиме. Перед выполнением загрузить (инициализировать) память микропрограмм. Для этого после этапа компилирования проекта запустить симулятор, войти в меню инициализации, выбрать режим инициализации памяти. Результаты выполнения представить в виде временой диаграммы и трассы в виде таблицы.

3. Определить задержки сигналов от входов до выходов УА и продолжительность такта Т УА.

4. Написать отчет, содержание отчета см. в подразделе 2.3.

Проект для лабораторной работы №2 находится в папке «Лаб работы `Организация АЛУ` - Примеры выполнения программ для лаб работы\2011\alu_full6310s.dir» (рекомедуется использовать этот проект). В нем для компиляции рекомедуется использовать файл ..\ua.gdf. Микропрограмму следует сначала записать в память. Для этого перед компиляцией рекомендуется выбрать опцию «Assign | Device | FLEX10KA». Далее, после компиляции рекомендуется выбрать файл ..\\ua.scf. После выбора опции «Simulator», но до нажатия его клавиши «Start» следует инициализировать память путем выбора опции «Initialise | Initialise memory». Рекомендуется выбрать опцию «Memory name | pzu:37:lpm_rom:LPM_ROM_component|altrom:srom|content» и в открывшемся окне выбрать «Import file | ua(1)й.mif».

Таблица 8

Варианты для самостоятельных заданий

Номер варианта	Условн. переход	Безусловн. переход	Цикл
1	jz	jmp	endz
2	jz	jmp	ends
3	js	jmp	endc
4	jovr	jmp	endz
5	jc	call, jmp	ends
6	jz	jmp	endc
7	jz	call, jmp	endz
8	js	jmp	ends
9	jovr	jmp	endc
10	jc	jmp	endz
11	jz	jmp	ends
12	jz	call, jmp	endc
13	js	jmp	endz
14	jovr	jmp	ends
15	jc	jmp	endc
16	jz	jmp	endz
17	jz	jmp	ends
18	js	jmp	endc

Окончание табл. 8

19	jovr	call, jmp	endz
20	jc	jmp	ends
21	jz	jmp	endc
22	jz	call, jmp	endz
23	js	jmp	ends
24	jovr	jmp	endc
25	jc	call,jmp	endz

2.3 Содержание отчета

Отчет должен содержать цель работы, задание, микропрограмму, схему алгоритма программы, а также временные диаграммы с заданными параметрами сигналов и трассу выполнения микропрограммы. Следует записать длительность периода тактовых импульсов. Также отчет должен содержать значение максимальной рабочей частоты проекта.

Следует сделать выводы по результатам работы.

2.4 Контрольные вопросы

1. Назначение и организация работы УА (рис. 23).
2. Назначение и организация работы блока ФАМ (рис. 24).
3. Какие сигналы определяют адрес на выходе БФАМ?
4. Назначение стека и порядок его работы (рис. 26).
5. Назначение СМК и порядок его работы.
6. Назначение полей X и A микрокоманды.
7. Назначение и организация работы блока декодирования типа МК (поля X).
8. Назначение сигналов start, ready, res, y_{M+1}.
9. Поясните порядок выполнения работы на примере 1 варианта задания.
10. Поясните микрокоманды условного перехода.
11. Поясните микрокоманды безусловного перехода.
12. Поясните микрокоманды push, endmp.

ЗАКЛЮЧЕНИЕ

В предлагаемых методических указаниях к лабораторным работам рассмотрены вопросы:

1) принципы действия и структурная организация составных блоков АЛУ – операционного и управляющего автоматов;

2) основы работы с современными средствами проектирования цифровых вычислительных систем;

3) процесс функционирования АЛУ во времени (в режиме компьютерного симулятора).

Рассмотрен двухтактный конвейерный АЛУ на основе принципа микропрограммного управления, применяющегося сейчас практически во всех цифровых вычислительных устройствах. Для изучения использован макет АЛУ, характерной особенностью которого является 2-уровневое управление процессом вычислений, также широко применяемое в современных ЭВМ и контроллерах.

В лабораторных работах использованы компьютерные модели, так называемые «проекты», разработанные в среде Max+Plus II студентами и преподавателями СГАУ. Данный термин введен фирмой «Altera», разработавшей указанную САПР. В графическом диалоговом интерфейсе этой САПР проект представлен как структурно-функциональная блок-схема, содержащая условные обозначения, близкие к стандартным. Аналогичный подход применен и при компьютерном моделировании работы проекта. На экране дисплея при этом изображаются временные диаграммы, которые может изменять пользователь. Пользовательский интерфейс дает возможность заглянуть в любую точку проекта, то есть как бы добраться щупом виртуального осциллографа туда, куда обычно, при интегральном исполнении устройств, это сделать невозможно. Это способствует глубокому изучению предмета.

В данных методических указаниях студенты вначале знакомятся с основами проектирования в MAX+PLUS II. Полученные знания они затем используют при изучении АЛУ. Студенты изучают АЛУ, используя проекты его отдельных компонентов, таких как ОА и УА.

Описанные здесь лабораторные работы входят в практическую часть курсов «Организация ЭВМ и вычислительных систем» и «ЭВМ и периферийные устройства» по специальности «АСОИУ». Для этого они используются совместно с четырьмя другими лабораторными работами, методические указания к которым издаются отдельно. Это лабораторная работа по ознакомлению с MAX+PLUS II, лабораторная работа по изучению структурной организации АЛУ в целом (то есть, с ОА и УА), лабораторная работа, посвященная функциональной организации ЭВМ, на примере системы прерываний компьютеров IBM PC, а также лабораторная работа, посвященная системе ввода/вывода компьютеров этого семейства.

Данные методические указания к лабораторным работам предоставляют возможность для обучения студентов принципам структурной организации АЛУ, а также дают основы и практические примеры разработки вычислительных систем с помощью современной САПР.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Поречный, В. Использование САПР «MAX+PLUS II" для разработки цифровых устройств на ПЛИС фирмы «Altera» [Электронный ресурс] / В. Поречный. − URL: http://www.epos.ua/view.php/pubs_3?subaction=showfull&id=983311200&archive=&start_from=&ucat=3& – (Дата обращения 30.05.2013).

2. Стешенко, В. Б. ПЛИС фирмы «Altera»: элементая база, система проектирования и языки описания аппаратуры [Электронный ресурс] / В. Б. Стешенко. − 3-е изд., стер.- М.: Издательский дом «Додэка-XXI», 2007. – 576 с. − ISBN 978-594120-112-9. – 1 электрон. опт. диск (CD-ROM).

3. Орлов, С. А. Организация ЭВМ и систем [Текст]: учебник для вузов / С. А. Орлов, Б. Я. Цилькер. – 2-е изд. – СПб.: Питер, 2011. – 688 с. – ISBN 978-5-498097-862-5.

4. Павлов, В. П. Организация ЭВМ и систем [Текст]: учеб. пособие для студентов заоч. формы обучения / В. П. Павлов; Самар. гос. аэрокосм. ун-т им. С. П. Королева. – Самара: Самар. аэрокосм. ун-т, 2000. – 181, (1) с. – ISBN 5-7883-0122-X.

5. Баранов, С. И. Синтез микропрограммных автоматов [Текст] / С. И. Баранов. – 2-е изд., перераб. и доп. – Л.: Энергия, 1979. – 231 с.

ПРИЛОЖЕНИЕ А.

ПРИМЕРЫ ОТЧЕТОВ ПО ЛАБОРАТОРНЫМ РАБОТАМ

Организация ОА АЛУ

Цель лабораторной работы – изучение принципов построения и архитектуры АЛУ.

Задание на самостоятельную работу: выполнить вариант №1 задания: $Y = N_1 - 2N_2$ (для n = 4). Коды микропрограммы приведены в табл. А.1.

Таблица А.1

Микропрограмма

(цифры в столбцах начиная со второго – в бинарном коде)

Описание МО	Источники операндов	Операция	Приемники результата	Адреса		Вход перен.	Тип сдвига
	y_0, y_D	ALO	y_w, y_{out}	A1	A2	C_{in}	SH
1. РОН0: = D∨0	01	011	10	-	0000	-	000
2. РОН1: = 2(D∨0)	01	011	10	-	0001	-	011
3. РОН0: = РОН0 - РОН1	10	010	11	0001	0000	1	000

Первая микрооперация обеспечивает загрузку (ввод) в регистр РОН0 значения первого операнда N_1, подаваемого извне на вход D.

Вторая микрооперация обеспечивает загрузку (ввод) в РОН1 значения второго операнда N_2 со сдвигом влево на один разряд.

Третья микрооперация обеспечивает вычитание $N_1 - 2N_2$ и запись результата в РОН0.

Кодирование микроопераций выполнено в соответствии с табл. А.1 (источники операндов), табл. А.2 (реализуемая УКС операция), табл. А.3 (тип сдвига) и табл. А.4 (приемники результата) [1], список источников см. в конце примера.

Таблица А.2

Трасса микропрограммы

(цифры в столбцах начиная со второго – в бинарном коде)

Микрооперация	Входы		Результаты (выходы)								
	D	C_{in}	F	B	РОН0	РОН1	SL_0	CF	ZF	OF	SF
1. РОН0:= D∨0	1101	0	1101	1101	1101	-	-	-	0	-	1
2. РОН1:= 2(D∨0)	0011	0	0011	0110	1101	0110	0	-	0	-	0
3. РОН0:= РОН0-РОН1	-	1	0111	0111	0111	0110	-	1	0	1	0

Результаты выполнения (трасса) микропрограммы для N_1 = -3 (в дополнительном коде это 1101) и N_2 = +3 = 0011 представлены ниже в табл. А.2. Значения операндов выбраны таким образом, чтобы результат переполнил разрядную сетку.

Временные диаграммы работа проекта ОА представлены на рис. А.1.

Микрооперации, выполняемые ОА:

1. D$\vee$0 - 1101$\vee$0000 = 1101 $\Rightarrow$ РОН0), флаг ZF = 0, так как результат не нулевой, SF = 1 – старший разряд результата равен 1,

2. 2(D$\vee$0) $\Rightarrow$ SHL1(0011$\vee$0000) = 0110$\Rightarrow$РОН1, SL_0 = 0, ZF = 0, SF = 1, флаги CF, OF – не формируются,

3. -3-L1(+3) = -3-6 = -9, РОН0-РОН1 $\Rightarrow$ 1101 – 0110 = 0111$\Rightarrow$ РОН0, флаги ZF = 0, SF = 0, CF = 1, OF = 1, так как результат переполняет разрядную сетку. Операция вычитания производится путем формирования дополнительного кода вычитаемого (0110$\Rightarrow$1001+1=1010) и сложения его с первым операндом (1010 + 1101 = 0111 = +7). В процессе сложения формируется сигнал переноса из старшего разряда сумматора C_{OUT}, равный 1. Старший (знаковый) разряд суммы при этом становится равным нулю (т.е. портится), что свидетельствует о переполнении разрядной сетки.

Таблица А.3

Обозначения входов и выходов ОА АЛУ в проекте и в МУ

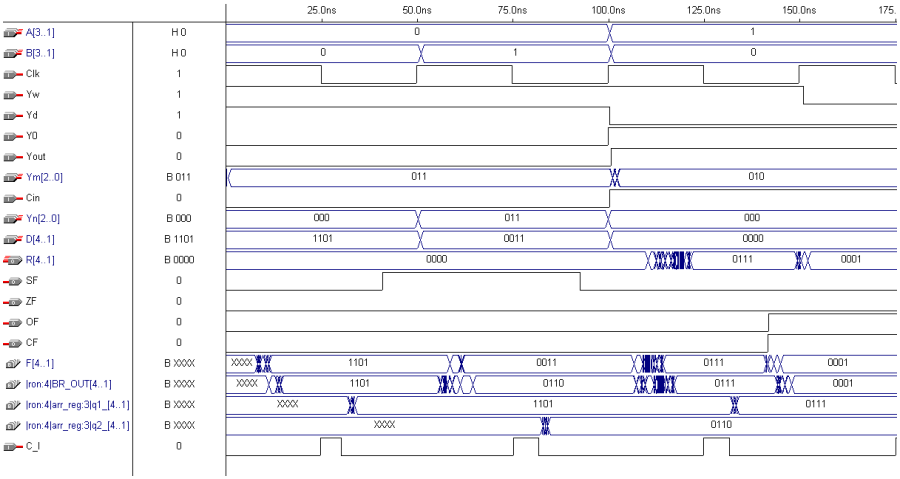
Метод. указания	Проект
D	D
C_0	Cin
Y	Y[3..0]
PQ	PQ[3..0]
РОН	Lq0_[3..0]
PR3	PRhi
C_4	Cout
Z	Z
OVR	OVR
F_3	Fhigh

В табл. А.3 приведены обозначения входов и выходов в проекте и методических указаниях. Проект находится в папке «\oa_full.dir» [2]. В проекте для работы микросхемы памяти (там находятся РОНы) надо на входе P/D_USE (в методич. указаниях он не отмечен) задать единицу. Перед запуском команды Max+II| Simulator выбрано значение параметра Option| Grid Size=5 нсек.

Продолжительность такта – 50 нс. Откуда тактовая частота – 20 МГц. Время задержки определено между передним фронтом ГТИ (вход Clc) и выходом R[4..1], так как на нем позднее всего появляются результаты расчетов.

Передний фронт ГТИ, откуда начинаем отсчет, первый после фронта C_I, запускающего ОА. Время задержки составило 20 нс. Оно немного меньше, чем половина периода ГТИ, поскольку расчеты должны быть завершены в течение первой половины такта, во второй половине такта результаты записываются в память ОА. По внутреннему выводу |ron_4|arr_reg3|q2_[4..1] определяем быстродействие ОА, так как на этом выводе результаты появились позже всего. Итак, время задержки выполнения обеих операций в ОА составило 30 нс. Тактовая частота проекта выбрана максимальной, из расчета длительности первой из операций. Длительность этой операции определяется временем чтения и записи данных в ПЗУ. Итак, максимальная тактовая часто-

та работы ОА АЛУ составила 20 МГц. Это много меньше, чем тактовая частота микропроцессора у современных компьютеров, но характерно для современных микроконтроллеров.



Р и с . А . 1. Временные диаграммы работы ОА

Выводы:

– изучены принципы организации ОА АЛУ;

– выполнен вариант задания в «полуавтоматическом» режиме, то есть без УА;

– успешно проведена верификация проекта на тестовой микропрограмме;

– быстродействие ОА ограничивают микросхемы ОЗУ.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Организация арифметико-логических устройств ЭВМ [Электрон. ресурс]: метод. указания к лаб. раб. по курсу Б3.Б.2 «ЭВМ и периферийные устройства» направления 230100.62 «Информатика и выч. техника» / сост. О. А. Заякин, В. П. Павлов. – Самара: Изд-во СГАУ, 2014.

2. Учебно-методический комплекс дисциплины «Аппаратные средства выч. техники» [Электрон. ресурс] / Самарский гос. аэрокосм. ун-т; специальность 090105 «Комплексное обеспечение информационной безопасности автоматизированных систем»; учебный план 0901105.65-10-О-П; фак-т «Информатика», каф. информационных систем и технологий. – Самара, 2010. – 1 CD-R. – Доступен на каф. ИСТ.

Организация управляющих автоматов АЛУ

Цель лабораторной работы – изучение принципов построения операционных автоматов арифметико-логических устройств (ОА АЛУ).

Задание на самостоятельную работу: выполнить вариант №1 задания, коды микропрограммы приведены в таблице А.4.

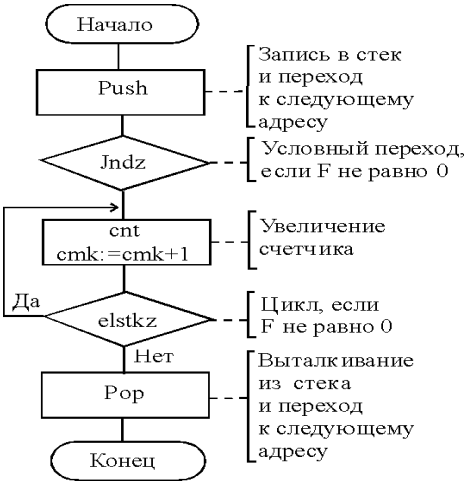
Т а б л и ц а А. 4

Микропрограмма (для g = 1 и пускового адреса a = 0Ah)

Микропрограмма		Код микрокоманды, HEX	
Адрес МК	Мнемоника МК	Тип микрокоманды X	Адрес микрокоманды A
a + 0 (0Ah)	Cnt (начало)	0001	-
a + 1 (0Bh)	Jz 11h	1000	11h
a + 2 (0Ch)	Push	0101	-
a + 3 (0Dh)	Cnt	0001	-
a + 4 (0Eh)	Endz	1100	-
a + 5 (0Fh)	Cnt	0001	-
a + 6 (10h)	Endmp (конец)	0110	-
a + 7 (11h)	Cnt	0001	-
a + 8 (12h)	Jmp 0Fh	0000	0Fh

В работе рассматривается АЛУ с программируемой логикой управления [1]. Список источников см. в конце примера.

На рис. А.2 приведена схема алгоритма микропрограммы.



Р и с . А . 2. Алгоритм микропрограммы ОА АЛУ

Результаты ее выполнения (трасса и временная диаграмма) представлены в таблице А.5 и рис. А.3.

В работе использован проект, который находился в папке «Лаб работы `Организация АЛУ` - Примеры выполнения программ для лаб работы\2011\alu_full6310s.dir» [2]. В нем для компиляции использован файл ..\ua.gdf. Микропрограмма сначала была записана в память. Для этого перед компиляцией была выбрана опция Assign | Device | FLEX10KA. Далее, после компиляции, при исполнении проекта на симуляторе MAX+PLUS II использован файл ..\\ua.scf. После выбора опции «Simulator», но до нажатия его клавиши «Start» была инициализирована память путем выбора опции «Initialise | Initialise memory». При этом была выбрана опция:

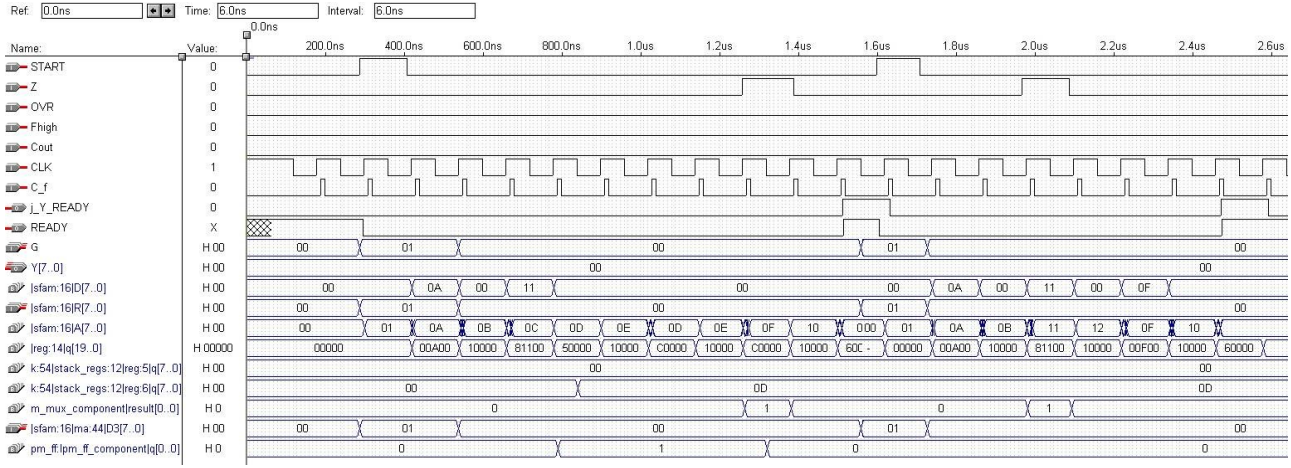
«Memory name | pzu:37:lpm_rom:LPM_ROM_component|altrom:srom|content», и затем в открывшемся диалоговом окне была выбран файл «Import file | ua(1)й.mif».

Т а б л и ц а А. 5

Трасса микропрограммы

Микрокоманда		Входы (h)				Выходы (h)					Стек (h)	
Адрес МК (h)	Мнемоника МК	RES	Start	g	Z	Усл. TST	Адрес сл. МК	Ready	Регистр МК		R0	R1
									X	A		
01	Jmp 0Ah	1	1	1	0	0	0A	1	0000	0A	➔	
0A	Cnt	0	0	0	0	0	0B	0	0001	00	➔	
0B	Jz 11h	0	0	0	0	0	0C	0	1000	11h	➔	
0C	Push	0	0	0	0	0	0D	0	0101	00		➔0Dh
0D	Cnt	0	0	0	0	0	0E	0	0001	00		➔0Dh
0E	Endz	0	0	0	0	0	0D	0	1100	00		➔0Dh
0D	Cnt	0	0	0	0	0	0E	0	0001	00		➔0Dh
0E	Endz	0	0	0	1	1	0F	0	1100	00	➔	0Dh
0F	Cnt	0	0	0	0	0	10	0	0001	00	➔	0Dh
10	end mp	0	0	0	0	0	00	1	1100	00	➔	0Dh
01	Jmp 0Ah	0	1	1	0	0	0A	1	0000	0A	➔	0Dh
0A	Cnt	0	0	0	0	0	0B	0	0001	00	➔	0Dh
0B	Jz 11h	0	0	0	1	1	11	0	1000	11h	➔	0Dh
11	Cnt	0	0	0	0	0	12	0	0001	00	➔	0Dh
12	Jmp 0Fh	0	0	0	0	0	0F	0	1000	11h	➔	0Dh
0F	Cnt	0	0	0	0	0	10	0	0001	00	➔	0Dh
10	end mp	0	0	0	0	0	00	1	1100	00	➔	0Dh
00	Jmp 00h	0	0	0	0	0	00	1	0000	00	➔	0Dh

П р и м е ч а н и е - знак “➔” показывает положение указателя стека УС.



Р и с . А . 3. Временные диаграммы работы проекта УА АЛУ

Результаты исследований

Продолжительность такта – 130 нс.

Задержка определена как время от фронта ГТИ, первого после фронта сигнала START и до установления уровня после перехода из 0 в 1 (окончания длительности переднего фронта) вывода памяти ПЗУ: |sfam:16|A[7..0]. Этот переход был выбран как последний на временной диаграмме из всех выводов, по которым мы определяли реакцию макета УА.

Таким образом, тактовая частота в макете была практически максимальной. При этом частота ГТИ составила 7,7 МГц. Если сравнить с ОА, то получается, что УА работает медленнее. Очевидно, быстродействие ограничивают блоки ОЗУ.

Выводы:

– изучены принципы организации УА АЛУ;
– выполнен проект устройства без УА;
– успешно проведена верификация проекта на тестовой микропрограмме;
– оказалось, что УА работает медленнее, чем ОА, очевидно, это связано с быстродействием ОЗУ;
– быстродействие УА ограничивают микросхемы ОЗУ.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

(см. в конце предыдущего примера)

Учебное издание

ОРГАНИЗАЦИЯ АРИФМЕТИКО-ЛОГИЧЕСКИХ УСТРОЙСТВ ЭВМ

Методические указания

Составители: ***Заякин Олег Александрович,***
Павлов Владимир Павлович

Редактор: Ю.Н. Литвинова
Доверстка: Т.Е. Половнева

Подписано в печать 25.08.2014. Формат 60×84 1/16.
Бумага офсетная. Печать офсетная. Печ. л. 3,75.
Тираж 100 экз. Заказ . Арт. - 55 /2014.

Самарский государственный аэрокосмический университет.
443086 Самара, Московское шоссе, 34.

Изд-во Самарского государственного аэрокосмического университета.
443086 Самара, Московское шоссе, 34.