МИНИСТЕРСТВО ОБОРОНЫ РОССИЙСКОЙ ФЕДЕРАЦИИ

Ставропольское высшее военное авиационное инженерное училище
(военный институт) имени маршала авиации В.А. Судца

А.В. Баженов

ОСНОВЫ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

Учебное пособие

Рекомендовано государственным образовательным учреждением высшего профессионального образования «Военно-воздушной академией имени Ю.А. Гагарина» в качестве учебного пособия для курсантов Ставропольского высшего военного авиационного инженерного училища (военного института) имени маршала авиации В.А. Судца, обучающихся по специальности «Техническая эксплуатация транспортного радиооборудования»

Ставрополь
Издание училища
2008

ББК 32.97

Баженов, А.В. ОСНОВЫ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН/ Учебное пособие с грифом УМО. [Текст]/ А.В. Баженов- 2-е изд., перераб. – Ставрополь: СВВАИУ (ВИ) , 2008. – 338 с.

Учебное пособие написано в соответствии с учебным планом дисциплин «Основы бортовых вычислительных машин» специальностей 160905.65 «Техническая эксплуатация транспортного радиооборудования».

В нем изложены математические основы построения и функционирования бортовых вычислительных машин. Приведены примеры реализации теоретических положений в современных и перспективных бортовых ЦВМ летательных аппаратов. Второе издание дополнено контрольными вопросами и заданиями для самостоятельной работы обучаемых. Кроме этого добавлена глава «Технические средства цифровой обработки сигналов», в которой изложены основы теории цифровой обработки информации и их реализация в бортовом РЭО на современной элементной базе: микроконтроллерах и цифровых сигнальных процессорах.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ……. 8
Раздел 1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ЦИФРОВОЙ ОБРАБОТКИ ИНФОРМАЦИИ…… 10
Глава 1. МАТЕМАТИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН …….. 10
1.1 Позиционные системы счисления и их основные свойства 10
1.1.1 Основные сведения о системах счисления ……..... 10
1.1.2 Двоичная система счисления……. 13
1.1.3 Двоично-десятичная система счисления……..... 14
1.1.4 Системы счисления по основанию 2^n …… 15
1.2 Арифметические действия в позиционных системах счисления и перевод чисел из одной системы счисления в другую …… 16
1.2.1 Арифметические действия в двоичной системе счисления …….. 16
1.2.2 Сложение и умножение чисел по модулю …… 19
1.2.3 Перевод чисел из одной системы счисления в другую 19
1.2.4 Понятие о системах счисления в остаточных классах 22
1.3 Машинные двоичные коды и действия с ними ……. 26
1.3.1 Формы представления чисел ……. 26
1.3.2 Машинные двоичные коды …… 27
1.3.3 Арифметические действия в кодах ……..... 29
1.4 Общие сведения о функциях алгебры логики …….. 37
1.4.1 Функции и операции …… 37
1.4.2 Основные законы и правила алгебры логики ……. 42
1.4.3 Алгебры Жегалкина, Шеффера и Пирса ……. 44
1.4.4 Элементарные конъюнкции и дизъюнкции …… 46
1.5 Переключательные функции и их минимизация ……. 48
1.5.1 Канонические формы функций …… 48
1.5.2 Общие понятия о минимизации переключательных функций …… 51
1.5.3 Минимизация переключательных функций посредством диаграмм Вейча …… 54
1.5.4 Минимизация неполностью определенных переключательных функций и представленных в форме СКНФ … 57
Глава 2. СХЕМОТЕХНИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН ……. 60
2.1 Синтез цифровых автоматов …….. 60
2.1.1 Основные понятия из теории автоматов ……. 62
2.1.2 Этапы синтеза схем автоматов ……. 66

2.1.3 Функционально полные системы логических функций .. 68
2.1.4 Техническая реализация переключательных функций .. 72
2.2 Особенности синтеза цифровых автоматов с памятью …… 74
2.2.1 Общие сведения о триггере как элементе памяти 74
2.2.2 Триггеры с одним информационным входом и законы их функционирования ……. 76
2.2.3 Триггеры с двумя информационными входами …..... 79
2.3 Переходные процессы в цифровых автоматах ……..... 83
2.3.1 Анализ переходных процессов в комбинационных схемах …… 83
2.3.2 Синтез логических схем с учетом риска сбоя ……. 85
2.3.3 Проблемы гонок в цифровых автоматах с памятью…… 86
Глава 3. ЭЛЕМЕНТНАЯ БАЗА ЦИФРОВЫХ УСТРОЙСТВ …. 90
3.1 Дешифраторы, шифраторы, схемы сравнения …… 90
3.1.1 Дешифраторы одноступенчатого и многоступенчатого типов …….. 90
3.1.2 Преобразователи кодов, шифраторы …… 94
3.1.3 Многоразрядные схемы сравнения ……..... 97
3.2 Цифровые коммутаторы ……. 99
3.2.1 Мультиплексоры …… 99
3.2.2 Демультиплексоры …… 101
3.2.3 Синтез комбинационных схем на мультиплексорах 103
3.3 Регистры …… 107
3.3.1 Общие сведения о регистрах ……. 107
3.3.2 Регистры параллельного действия …… 109
3.3.3 Регистры последовательного действия …… 111
3.4 Счетчики …… 113
3.4.1 Общие сведения о счетчиках ……. 113
3.4.2 Суммирующий счетчик ……. 114
3.4.3 Вычитающий и реверсивный счетчики …… 117
3.4.4 Счетчик с заданным коэффициентом пересчета …… 119
3.5 Комбинационные сумматоры ……..... 121
3.5.1 Одноразрядный комбинационный сумматор ……..... 121
3.5.2 Многоразрядный комбинационный сумматор последовательного действия …… 124
3.5.3 Многоразрядный параллельный сумматор ……... 125
3.5.4 Способы ускорения переноса в сумматорах ……. 127
Раздел 2. ПРИНЦИПЫ ПОСТРОЕНИЯ И ФУНКЦИОНИРОВАНИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН …… 129

Глава 4. ОБЩИЕ СВЕДЕНИЯ О НАЗНАЧЕНИИ И ПРИНЦИПАХ ФУНКЦИОНИРОВАНИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН …… 129
4.1 Назначение, характеристики и классификация БВМ …… 129
4.1.1 Роль и место БВМ в составе авиационного радиоэлектронного комплекса …… 129
4.1.2 Классификация БВМ ……. 133
4.2 Структура и принцип работы аналоговых БВМ ……... 133
4.3 Состав и принципы работы цифровой БВМ ……. 136
4.3.1 Принцип программного управления БЦВМ…… 136
4.3.2 Структура и порядок функционирования цифровых БВМ …… 138
4.3.3 Основные характеристики БВМ ……..... 141
4.3.4 Форматы представления команд и чисел в БЦВМ …….. 143
4.4 Особенности программирования в кодах команд БВМ …… 146
4.5 Общие сведения о системе команд БВМ …… 151
Глава 5. ФУНКЦИОНАЛЬНЫЕ УСТРОЙСТВА БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН …… 154
5.1 Устройства памяти БЦВМ ……. 154
5.1.1 Назначение, основные характеристики и классификация запоминающих устройств …… 154
5.1.2 Оперативные запоминающие устройства …… 157
5.1.3 Постоянные запоминающие устройства ……..... 161
5.1.4 Внешние запоминающие устройства …….. 164
5.1.5 Способы организации памяти ……. 165
5.1.6 Виды адресации и способы их реализации в БЦВМ…… 171
5.2 Процессоры БВМ ……. 178
5.2.1 Основные сведения о процессорах БВМ …….. 178
5.2.2 Классификация и основные характеристики арифметико-логических устройств …… 180
5.2.3 Выполнение арифметических и логических операций в арифметико-логическом устройстве …….. 181
5.2.4 Устройства управления со схемной организацией ……. 184
5.2.5 Микропрограммный принцип управления …….. 189
5.2.6 Микропроцессоры и их программирование ……..... 193
5.3 Интерфейсные устройства БВМ ……. 198
5.3.1 Назначение, классификация и основные характеристики интерфейсов ……. 198
5.3.2 Режимы работы БВМ при взаимодействии с внешними устройствами ……... 203

5.3.3 Особенности организации информационного обмена по прерываниям ……... 204
5.3.4 Аналого-цифровые и цифроаналоговые преобразователи сигналов …….. 212
5.3.5 Микропроцессорные средства информационного обмена ……... 225
Глава 6. ОСОБЕННОСТИ ЭКСПЛУАТАЦИИ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН ……. 229
6.1 Основное содержание и этапы эксплуатации БВМ …….. 229
6.2 Программные методы контроля работоспособности БВМ .. 229
6.3 Помехоустойчивое кодирование сигналов и его применение для схемного контроля БВМ ……. 236
6.3.1 Общие сведения об избыточном кодировании сигналов 236
6.3.2 Коды, контролирующие ошибки ……..... 240
6.3.3 Коды, исправляющие ошибки ……. 242
6.3.4 Каскадные коды ……. ….. 246
Глава 7. ОБЩИЕ СВЕДЕНИЯ О БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ ……. 249
7.1 Краткая характеристика бортовых вычислительных систем и их классификация …….. 249
7.2 Особенности функционирования БЦВС авиационных радиоэлектронных комплексов ……..... 257
7.2.1 Программное обеспечение БЦВС АРЭК ……. 257
7.2.2 Выполнение программ, жестко связанных с временной диаграммой …… 260
7.2.3 Выполнение программ по асинхронным заявкам 262
7.3 Надежность бортовых вычислительных систем …….. 267
7.4 Повышение живучести системы за счет автоматической реконфигурации …… 270
Глава 8. ТЕХНИЧЕСКИЕ СРЕДСТВА ЦИФРОВОЙ ОБРАБОТКИ СИГНАЛОВ ……. 275
8.1 Общие принципы технической реализации алгоритмов цифровой обработки……..... 275
8.2 Применение однокристальных микроконтроллеров для цифровой обработки сигналов……. 279
8.2.1 RISC – архитектура современных микроконтроллеров .. 284
8.3 Программируемые процессоры цифровой обработки сигналов ……..... 291
8.3.1 Основные операции обработки сигналов в цифровых системах и их влияние на структуру сигнального процессора 291
8.3.2. Обобщенная архитектура DSP…… 292

8.4 Процессоры семейства DSP56000…….. 295
8.4.1. Обзор архитектуры и шинной организации DSP56000 295
8.4.2 Порты ввода/вывода…… 319
8.5 Специализированная цифровая вычислительная машина «Багет-55» …… 324
ЗАКЛЮЧЕНИЕ. ПЕРСПЕКТИВЫ РАЗВИТИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН И СИСТЕМ …….. 333
ЛИТЕРАТУРА 337

ВВЕДЕНИЕ

Любая форма человеческой деятельности, функционирование любой из организованных систем (как искусственных, так и естественных) немыслимы без обработки информации. Понятие информация, в этом случае, означает совокупность данных, сведений подлежащих хранению, обработке и передаче.

Для управления современным самолетом и его вооружением используется разнообразная и в большинстве случаев быстроизменяющаяся информация. Возможности летчика-оператора по ее правильному восприятию и переработке ограничены, особенно в условиях дефицита времени, присущего боевым режимам.

Все возрастающие объемы информации при ограниченном времени ее переработки обуславливают широкое внедрение средств автоматизированной обработки, основным элементом которых являются бортовые вычислительные машины (БВМ).

В зависимости от формы представления информации БВМ делятся на два класса: аналоговые и цифровые (дискретные). В настоящее время все более широкое применение в авиационном радиоэлектронном оборудовании (РЭО) (в системах связи, радиолокации, при обработке речевых сигналов, изображений и др.) находят цифровые ЭВМ, имеющие по сравнению с аналоговыми ряд преимуществ, таких как: более высокая надежность; стабильность параметров при воздействии дестабилизирующих факторов; высокая точность обработки информации; значительное сокращение трудоемкости и упрощение операций регулировки и настройки; возможность создания микросхем с очень высокой степенью интеграции.

Успехи в области разработки быстродействующих элементов цифровой техники позволили создать бортовые ЦВМ, выполняющие десятки миллионов арифметических операций в секунду.

Принципиально новые возможности открывает применение цифровых интегральных схем в радиосвязи. Так, использование цифровых синтезаторов частоты позволило существенно снизить аппаратурные затраты и повысить фазовую стабильность генерируемых сигналов. Обработка сигналов цифровыми методами позволяет обеспечить высокую точность, стабильность параметров и получить характеристики, не достижимые аналоговыми методами. В частности, обеспечение очень высокой прямоугольности амплитудно-частотной характеристики (АЧХ) фильтра на аналоговых элементах практически

невозможно. Цифровые же фильтры позволяют реализовать произвольную форму АЧХ, в том числе и сколь угодно близкую к прямоугольной.

Цифровая схемотехника интенсивно внедряется и в радиоприемную аппаратуру, главным образом в системы управления. В частности, беспоисковая настройка приемника на любую из принимаемых частот осуществляется набором на пульте управления соответствующего кода, предварительно занесенного в устройство памяти. Используя электронные таймеры, также выполняемые на цифровых интегральных микросхемах, можно обеспечить последовательную перестройку радиостанции по любой заранее заданной программе. При этом указанная программа может долговременно храниться в памяти и при отключенном питании. Возможно также управление режимом работы приемника (коэффициентом усиления, полосой пропускания, диаграммой направленности антенны и т.п.).

Является перспективным внедрение цифровой техники и в оптико-локационные и телевизионные системы. Цифровое телевидение позволяет повысить качество передачи сигналов благодаря существенному уменьшению накопления искажений в цифровых линиях связи по сравнению с аналоговыми, а также за счет применения специальных способов кодирования, обнаруживающих и исправляющих ошибки передачи информации. Сигналы представленные в цифровой форме, практически не подвержены амплитудным и фазовым искажениям, что позволяет передавать телевизионную информацию на большие расстояния с сохранением ее высокого качества. В результате использования методов и устройств цифровой техники становится возможным длительный бесподстроечный режим работы.

С помощью цифровых устройств обеспечивается кодирование сигналов командных радиолиний управления, что обеспечивает высокую помехозащищенность и скрытность передаваемых сообщений.

Широкое применение средств цифровой вычислительной техники в бортовых радиоэлектронных комплексах позволяет значительно повысить боевую эффективность современных и перспективных самолетов, а также безопасность полетов. Однако это предъявляет высокие требования к уровню подготовки авиационных радиоинженеров в области цифровой вычислительной техники. Дать основы такой подготовки ставит своей целью дисциплина «Основы бортовых вычислительных машин».

Раздел 1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ЦИФРОВОЙ ОБРАБОТКИ ИНФОРМАЦИИ

Глава 1. МАТЕМАТИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

1.1 Позиционные системы счисления и их основные свойства

1.1.1 Основные сведения о системах счисления

Цифровые сигналы, с которыми работают цифровые вычислительные машины, представляют собой числа или цифры некоторой системы счисления. Происхождение чисел и способов счета связано с практическими потребностями людей в измерении различных величин. Числом выражают отношение между измеряемой величиной и величиной, принятой за единицу измерения. Число является отвлеченным, абстрактным выражением количества. Для пользования числами их необходимо свести в определенную систему, условиться о названиях чисел и способах их обозначения.

Система счисления (СС) - это совокупность приемов и правил, устанавливающих взаимно однозначное соответствие между числом и его записью.

Каждому числу A или каждой цифре d в записи числа A можно сопоставить некоторый количественный эквивалент, выражаемый числом.

Систему счисления называют позиционной (ПСС), если значение цифр определяется не только особенностями их начертания, но и зависит от места (позиции), занимаемого ими в числе. В противном случае систему счисления называют непозиционной (НПСС). Примерами ПСС и НПСС являются, соответственно, десятичная и римская системы счисления. Действительно, пусть $A_{дес} = 55 = 5'5$ и $B_{рим}$= XX – X'X, тогда (5') = 50; (5) = 5, то есть (5') не равно (5), но $(X') = (X) = 10$.

Количество p различных символов, используемых в ПСС для записи цифр, называют основанием системы счисления. Цифрами обозначают первые p целых чисел от 0 до p - 1. В современных ЭВМ применяют десятичную, шестнадцатеричную, восьмеричную и двоичную системы счисления.

Для этих систем с основанием p (p = 2,8,10,16) применяют следующие наборы символов - цифр d (d= 0,1,..., p- 1):

для p = 2; d = 0,1;

для p = 8; d = 0,1,2,3,4,5,6,7;

для p = 10; d = 0,1,2,3,4,5,6,7,8,9;

для p = 16; d = 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F, где буквы A,B,C,D,E,F означают цифры 10,11,12,13,14,15.

Запись числа A (A ≥ 0) в p - ичной ПСС имеет вид:

$$A_p = d_{n-1}...d_1d_0,\ d_{-1}...d_{-m},$$

где через $d_{n-1}...d_1d_0$ обозначены n цифр целой, а через $d_{-1}...d_{-m}$ m цифр дробной части числа A.

Условно место (позицию) цифры в записи числа можно охарактеризовать весом этой позиции. На примере десятичной системы счисления вес первых четырех цифр целой и дробной части числа будет равен:

1000 100 10 1; 0,1 0,01 0,001 0,0001,

что соответствует:

10^3 10^2 10^1 10^0; 10^{-1} 10^{-2} 10^{-3} 10^{-4} .

Тогда число A может быть представлено в виде полинома:

$$A = d_{n-1}p^{n-1} + d_0p^0 + d_1p^{-1} + d_{-m}p^{-m}$$

или

$$A = \sum_{i=-m}^{n-1} d_i \cdot p^i, \qquad (1.1)$$

где p - основание ПСС; i - номер позиции или разряда; d_i - цифра i-го разряда; p^i - вес i -го разряда; $d_i\,p^i$ - количественный эквивалент цифры d , расположенной на i-ой позиции.

Приведем пример использования формулы (1.1). Пусть необходимо определить значение восьмеричного числа A = $173{,}2_{(8)}$ в десятичной системе счисления:

$$173{,}2_{(8)} = 1\cdot 8^2 + 7\cdot 8^1 + 3\cdot 8^0 + 2\cdot 8^{-1} = \sum_{i=-1}^{3-1} d_i \cdot 8^i = 123{,}25.$$

Совокупность $n + m = r$ разрядов, используемое для записи числа, с указанием положения запятой, образует разрядную сетку (рисунок 1.1).

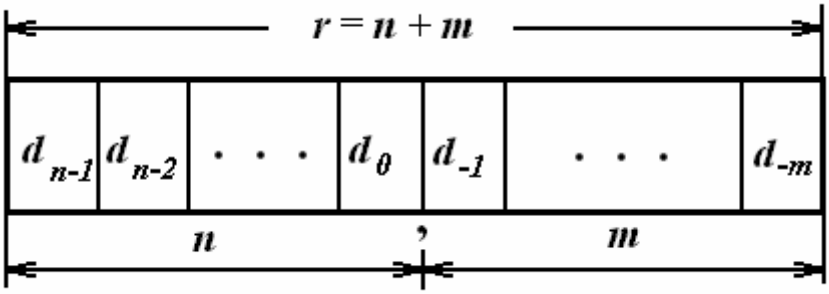


Рисунок 1.1

Позиционные системы счисления имеют следующие свойства.

1. Наибольшее число, которое можно записать в r - разрядной сетке, где $r = n + m$, получается, когда цифры всех разрядов имеют наибольшую величину: $d_i = p - 1$. При этом,

$$A_{наиб} = \sum_{i=-m}^{n-1}(p-1)p^i = \sum_{i=-m}^{n-1}(p^{i+1} - p^i) = \sum_{i=-m}^{n-1} p^{i+1} - \sum_{i=-m}^{n-1} p^i =$$

$$= \left(p^{-(m-1)} + ... + p^{n-1} + p^n\right) - \left(p^{-m} + p^{-(m-1)} + ... + p^{n-1}\right) = p^n - p^{-m}.$$

В частности для целого числа ($m = 0$) $A_{наиб} = p^n - 1$, а для дробного числа ($n = 0$) $A_{наиб} = 1 - p^{-m}$.

2. Наименьшее число, не равное нулю, получается, когда цифра младшего разряда равна 1, а во всех остальных разрядах цифры $d_i = 0$.

Для числа, записываемого в r - разрядной сетке ($r = n + m$), $A_{наим} = p^{-m}$. Для целого числа ($m = 0$) $A_{наим} = 1$.

3. Шагом (дискретностью) чисел h называется интервал между двумя ближайшими числами, которые могут быть записаны в заданной разрядной сетке. Величина шага равна весу младшего разряда:

$$h = p^{-m}.$$

Все числа от $A_{наим}$ до $A_{наиб}$ распределены с шагом h.

4. Количество различных чисел, которые можно записать в r-разрядной сетке (где $r = n + m$) равно:

$$N = \frac{A_{наиб}}{A_{наим}} + 1 = \frac{p^n - p^{-m}}{p^{-m}} + 1 = p^{n+m} = p^r.$$

5. Количество разрядов, нужное для записи целого числа A в системе с основанием p, находится из условия $A \le A_{наиб} = p^n - 1$, откуда

$$n \ge \log_p(A+1) \quad \text{или} \quad n = \text{int}\left[\log_p(A+1)\right],$$

где функция *int* осуществляет округление аргумента до ближайшего большего целого числа.

1.1.2 Двоичная система счисления

Основанием двоичной системы счисления является число 2, то есть в ней используются только две цифры: 0 и 1. Любое число в двоичной системе счисления записывается как комбинация цифр 0 и 1.

Двоичная система счисления нашла широкое применение в цифровых ЭВМ. Это обусловлено следующими основными её достоинствами.

1. Существует большое число устройств, имеющих два устойчивых состояния (закрытый или открытый транзистор, намагниченный или размагниченный сердечник, наличие или отсутствие импульса) которые могут быть приняты в качестве физического эквивалента символов 0 и 1. Это значительно упрощает автоматическую реализацию процессов записи, хранения, считывания и обработки двоичных чисел.

2. Арифметические и особенно логические операции над двоичными числами выполняются наиболее просто.

3. Для представления каждой цифры любого алфавита отводится определенный поддиапазон некоторой физической переменной. За счет помех и погрешностей измерения возникают ошибки представления информации этой переменной. Эти ошибки минимальны при двоичном алфавите.

В соответствии с формулой (1.1) некоторое число A в двоичной системе счисления (p = 2) можно записать в виде

$$A = d_{n-1}...d_1 d_0 d_{-1}...d_{-m} = \sum_{i=-m}^{n-1} d_i 2^i, \qquad (1.2)$$

где цифры d_i равны 0 или 1.

Для известных значений цифр d_i можно найти значение числа в десятичной системе счисления.

Например,

A = 11011,11 = $1\cdot2^4 + 1\cdot2^3 + 0\cdot2^2 + 1\cdot2^1 + 1\cdot2^0 + 1\cdot2^{-1} + 1\cdot2^{-2}$ = 27,75

Одними из основных недостатков двоичной СС является: трудоемкость перевода в нее чисел из десятичной СС, которой пользуется человек, и громоздкая форма записи.

Так для записи трехзначного числа десятичной СС требуется не менее 10 разрядов двоичной СС. Для более компактной записи чисел двоичной СС используют вспомогательные системы счисления с основанием 2^n - восьмеричную (2^3) или шестнадцатеричную (2^4) СС и двоично-десятичную СС.

1.1.3 Двоично-десятичная система счисления

Двоично-десятичная система счисления является гибридной, в этой системе цифры десятичной системы счисления представлены в виде двоичного кода. Для этой цели под каждый разряд десятичного числа отводится четыре разряда двоичного числа.

Запись и чтение чисел в двоично-десятичной системе счисления поясним на примерах.

Пример 1. Записать число 1983 в двоично-десятичной системе счисления.

Для этого каждый знак числа в десятичной системе счисления представляем в виде четырехразрядного двоичного кода (рисунок 1.2) и записываем полученное выражение в ряд:

1983 → 0001100110000011

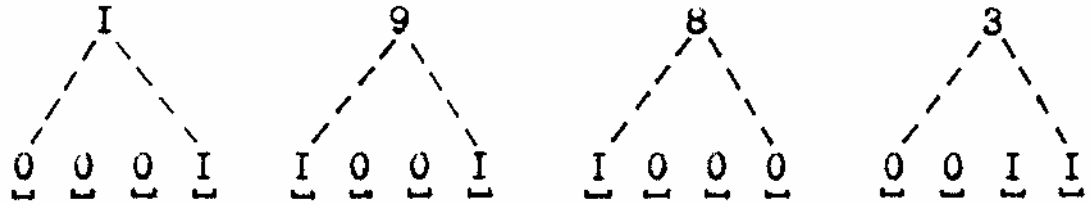


Рисунок 1.2 - Переход от десятичной к двоично-десятичной записи числа

Пример 2. Прочитать число, записанное в двоично-десятичной системе счисления 1000010010010101.

Для прочтения числа необходимо разделить его на декады, начиная с младшего (самого правого) разряда, по четыре разряда двоичного числа в каждой. Далее прочитать результат для каждой декады в виде цифры десятичной системы счисления (рисунок 1.3).

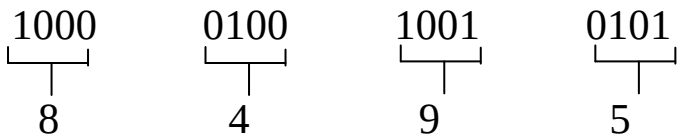


Рисунок 1.3 – Результат преобразования: 8495

Двоично-десятичная система широко используется в контрольно-проверочной аппаратуре самолетных ответчиков СО-69, СО-73 и других устройствах, в которых информация о десятичном числе записана в виде двоичного кода.

1.1.4 Системы счисления по основанию 2^n

На практике часто возникает проблема компактной записи двоичных чисел. При этом перевод записанного числа в двоичную систему должен быть простым и не требовать громоздких вычислений. Этим требованиям удовлетворяют системы счисления с основанием 2^n: четверичная (2^2), восьмеричная (2^3), шестнадцатеричная (2^4), тридцатидвухричная (2^5) и т.д. Для перевода из двоичной системы счисления в СС с основанием 2^n число разбивается на части по n разрядов и каждая часть записывается цифрой новой системы счисления. Для p =4 $d \in$ {0, 1, 2, 3}, для p =8 $d \in$ {0, 1, 2, 3, 4, 5, 6, 7}, для $p = 2^n$ $d \in$ {0, 1, ...p- 1}.

При использовании $p = 4$ ($n = 2$) исходное слово двоичной СС разбивается на части по 2 разряда, поэтому результирующее число имеет всего лишь в два раза меньше цифр. Поэтому использование четверичной СС малоэффективно. Системы счисления с $n > 4$ также малоэффективны так как в этом случае необходимо помнить много символов, обозначающих цифры этих СС. Например, для $n = 5$ число цифр равно 32.

Наибольшее распространение получили восьмеричная ($n = 3$) и шестнадцатеричная ($n = 4$) СС.

Для перевода двоичного числа в восьмеричную (шестнадцатеричную) СС необходимо разбить двоичное число вправо и влево от запятой на триады (тетрады) и заменить каждую триаду (тетраду) равной ей восьмеричной (шестнадцатеричной) цифрой. Если при разбиении крайние триады (тетрады) окажутся неполными, то их следует дополнить нулями.

Например,

1110001101,0111 = 001 110 001 101, 011 100 =

= 1 6 1 5, 3 $4_{(8)}$.

111011110001,0111101 = 1110 1111 0001, 0111 1010 =

= E F 1, 7 $A_{(16)}$.

В таблице 1.1 приведены значения некоторых чисел в различных СС.

Таблица 1.1 Запись чисел в различных ПСС

Основание ПСС				
p = 10	p = 16	p = 8	p = 2	p = "2 - 10"
0	0	0	000	0000
1	1	1	001	0001
2	2	2	010	0010
3	3	3	011	0011
4	4	4	100	0100
5	5	5	101	0101
6	6	6	110	0110
7	7	7	111	0111
8	8	10	1000	1000
9	9	11	1001	1001
10	A	12	1010	00010000
11	B	13	1011	00010001
12	C	14	1100	00010010
13	D	15	1101	00010011
14	E	16	1110	00010100
15	F	17	1111	00010101
16	10	20	10000	00010110
-	-	-	-	-
59	3B	73	111011	01011001

1.2 Арифметические действия в позиционных системах счисления и перевод чисел из одной системы счисления в другую

1.2.1 Арифметические действия в двоичной системе счисления

Арифметические действия в любой ПСС выполняют с использованием таблиц умножения, сложения и вычитания по правилам, аналогичным правилам десятичной арифметики.

Наиболее простыми из упомянутых выше таблиц являются таблицы для двоичной системы:

таблица умножения	таблица сложения	таблица вычитания
$0 \cdot 0 = 0$	$0 + 0 = 0$	$0 - 0 = 0$
$0 \cdot 1 = 0$	$0 + 1 = 1$	$1 - 1 = 0$
$1 \cdot 0 = 0$	$1 + 0 = 1$	$1 - 0 = 1$
$1 \cdot 1 = 1$	$1 + 1 = 10$	$10 - 1 = 1$

Старшие цифры двухразрядной суммы и уменьшаемого называют соответственно переносом и займом.

Сложение многоразрядных чисел $A = a_{n-1} \ldots a_0 \ldots a_{-m}$ и $B = b_{n-1} \ldots b_0 \ldots b_{-m}$ выполняется поразрядно, начиная с младшего разряда, с учетом правил таблицы сложения. Перед сложением количество разрядов целой и дробной части слагаемых уравнивается путем добавления незначащих нулей - в дробной части справа, а в целой части слева от разрядов чисел. В каждом i - ом разряде производится сложение трех цифр $a_i + b_i + p_i$, где p_i - цифра переноса из предыдущего (i - 1) разряда.

Пример:

$$\begin{array}{r} P_i = 110111\ 0 \\ A = \ 11011{,}11 \\ + \\ B = \ 01001{,}10 \\ \hline A + B = 100101{,}01 \end{array}$$

Вычитание многоразрядных чисел производится также поразрядно, причем, если цифра вычитаемого больше цифры уменьшаемого, то занимается 1 из следующего по старшинству разряда. Занятая единица приравнивается к двум единицам данного разряда.

Пример:

$$\begin{array}{r} \overset{1+1\ \ 1+1\qquad 1+1}{\nearrow\ \ \nearrow\qquad\quad \nearrow} \\ A = 1\,1\,0\,1\,1\,,\,1\,0 \\ - \\ B = 0\,1\,1\,0\,1\,,\,0\,1 \\ \hline A - B = 0\,1\,1\,1\,0\,,\,0\,1 \end{array}$$

Умножение многоразрядных двоичных чисел производится путем образования частичных произведений и их суммирования. Пусть даны числа $A = a_{n-1} \ldots a_0 \ldots a_{-m}$ и $B = b_{n1-1} \ldots b_0 \ldots b_{-m1}$.

Тогда $\Pi = A \cdot B = A \sum_{i=-m_1}^{n_1-1} b_i 2^i = \sum_{i=-m_1}^{n_1-1} A b_i 2^i = \sum_{i=-m_1}^{n_1-1} \Pi_i$,

где $\Pi_i = A b_i 2^i$ - частичное произведение, образуемое путем умноже-

ния множимого A на цифру множителя b_i и ее весовой коэффициент 2^i.

Умножению числа A на 2 соответствует сдвиг всех цифр этого числа на один разряд влево, т.е. в сторону старшего разряда:

$$2A = \left(a_{n-1} \cdot 2^{n-1} + a_{n-2} \cdot 2^{n-2} + ... + a_0 2^0 + ... + a_{-m} \cdot 2^{-m}\right) \cdot 2 =$$

$$= a_{n-1} \cdot 2^n + a_{n-2} \cdot 2^{n-1} + ... + a_0 2^1 + a_{-1} 2^0 + ... + a_{-m} \cdot 2^{-(m+1)}.$$

С учетом этого для получения частичных произведений Π_i достаточно сдвинуть код множимого на i разрядов влево (при i >0) или вправо (i < 0) и умножить его на 0 или 1 в зависимости от значения b_i.

Пример,

	A =	101,10
	B = ×	11,01
$\Pi_{-2} = A \cdot 1 \cdot 2^{-2}$ =		1,0110
$\Pi_{-1} = A \cdot 0 \cdot 2^{-1}$ =		00,000
$\Pi_{-0} = A \cdot 1 \cdot 2^{0}$ =		101,10
$\Pi_{-1} = A \cdot 1 \cdot 2^{1}$ =		1011,0
$\boldsymbol{\Pi} = A \cdot B$ =		10001,1110

При выполнении умножения на бумаге можно не указывать положений запятой в частичных произведениях, следует лишь правильно записать коды один под другим и произвести их сложение. Положение запятой в произведении определяется числом разрядов в его дробной части, которое равно сумме чисел разрядов в дробных частях множимого и множителя.

Деление многоразрядных чисел производится по таким же правилам, как и в десятичной системе счисления. Перед делением в делимом и делителе уравнивается количество цифр после запятой путем добавления нулей, затем запятые отбрасываются. Процедура нахождения частного иллюстрируется на примере деления числа A = 1100,011 на число B = 10,01:

Делимое	Делитель
1100,011	10, 010
- 1001 0	101, 1
0011 011	*Частное*
10 010	
01 0010	
1 0010	
0 0000	

1.2.2 Сложение и умножение чисел по модулю

При аппаратурной реализации устройства цифровой обработки информации обычно существуют ограничения на длину разрядной сетки, т.е. количество разрядов целой и дробной части. В этом случае, вне зависимости от реального значения, результат арифметических операций будет также ограничен разрядной сеткой. Пусть цифровое устройство работает только с целыми числами, для записи которых используется n, разрядов. Произведем сложение двоичных чисел A и B. Если сумма $C = A + B < 2^n$, то она может быть записана в n разрядах. Но может оказаться, что $C \geq 2^n$, в этом случае для регистрации суммы потребовался бы еще один дополнительный разряд слева с весом 2^n. Так как в разрядной сетке этого разряда нет, то результат будет соответствовать остатку от деления суммы C на величину $M = 2^n$, (деление на 2^n соответствует сдвигу числа вправо на n разрядов). Этот остаток и называется суммой чисел A и B по модулю 2^n. Для обозначения суммы по модулю применяется запись

$$[C] \bmod 2^n = [A + B] \bmod 2^n.$$

Операции сложения и умножения чисел по модулю могут производиться в любой системе счисления при любой величине модуля M.

Например, при сложении двоичных чисел $A = 10110_{(2)} = 22_{(10)}$ и $B = 10011_{(2)} = 19_{(10)}$ получаем $A + B = 101001_{(2)} = 41_{(10)}$. В рамках 5 - разрядной сетки старший разряд этой суммы не регистрируется ($2^5 = 32$). В результате будет зарегистрировано число, равное значению указанной суммы по модулю $M = 2^5$:

$$[A + B] \bmod 2^5 = [101001] \bmod 2^5 = 01001_{(2)} = 9_{(10)}.$$

Произведение двоичных чисел $A = 01011_{(2)}$ и $B = 00111_{(2)}$ равно $A \cdot B = 1001101_{(2)} = 77_{(10)}$ в рамках пятиразрядной сетки будет зарегистрировано число, равное значению найденного произведения по модулю 2^5 , т.е.:

$$[A \cdot B] \bmod 2^5 = [1001101] \bmod 2^5 = 01101_{(2)} = 13_{(10)}.$$

1.2.3 Перевод чисел из одной системы счисления в другую

Для перевода чисел из системы счисления с основанием q в систему счисления с основанием p могут быть использованы следующие

методы:

метод непосредственного замещения;

метод последовательного деления на основание новой системы счисления;

метод последовательного умножения на основание новой системы счисления.

Метод непосредственного замещения. Метод применяется для перевода всех видов чисел (целых, дробных и смешанных). Перевод числа из СС с основанием q, в СС с основанием p производится в следующем порядке.

1. Заданное число в СС с основанием q представляется в виде взвешенной суммы (1.1):

$$A = \sum_{i=-m}^{n-1} a_i q^i .$$

2. Все цифры a_i и основание q записываются (замещаются) цифрами системы счисления с новым основанием p .

3. Выполняются все арифметические операции в соответствии с формулой (1.1) в СС с основанием p .

Пример 1: $A_{(10)} = 35{,}25 \rightarrow A_{(2)}$

$35{,}25_{(10)} = 3\cdot10^1 + 5\cdot10^0 + 2\cdot10^{-1} + 5\cdot10^{-2}$;

$$A_{(2)} = 11\cdot1010 + 101\cdot1 + 10\cdot\frac{1}{1010} + 101\cdot\frac{1}{1010\cdot1010};$$

$A_{(2)} \approx 100011{,}01$.

Пример 2 : $A_{(2)} = 1001{,}011 \rightarrow A_{(10)}$

$1001{,}011_{(2)} = 1\cdot10^3 + 0\cdot10^2 + 0\cdot10^1 + 1\cdot10^0 + 0\cdot10^{-1} + 0\cdot10^{-2} + 1\cdot10^{-3}$;

$A_{(10)} = 1\cdot2^3 + 0\cdot2^2 + 0\cdot2^1 + 1\cdot2^0 + 0\cdot2^{-1} + 1\cdot2^{-2} + 1\cdot2^{-3}$;

$A_{(10)} = 9{,}375$.

Метод непосредственного замещения удобен при переводе чисел из любой СС в ту СС, в которой наиболее просто выполняются арифметические операции. При ручном счете этим методом удобно переводить числа в десятичную СС, а при автоматическом переводе с помощью ЭВМ - в двоичную СС.

Метод последовательного деления на основание новой системы счисления. Этот метод используется только для перевода целых чисел. Для перевода числа из СС с основанием q, в СС с основанием p необходимо :

1. Представить основание новой СС p в старой СС с основанием q.

2. Последовательно делить число и промежуточные частные деле-

ния этого числа на основание новой системы счисления, представленные в старой СС. Полученные остатки и последнее частное будут искомыми изображениями цифр p - ичного числа, выраженные в старой СС.

3. Первый остаток записать в младший разряд, все последующие остатки в более старшие, а последнее частное - в старший разряд числа.

4. Перевести найденные цифры p - ичного числа в СС с основанием p .

Пример: $A_{(10)} = 559 \rightarrow A_{(16)}$

1. $q = 16$

2.

$$\begin{array}{r|l} 559 & 16 \\ \hline 48 & 34 \\ \hline 79 & 32 \\ \hline 64 & 2 \\ \hline 15 & \end{array} \qquad \begin{array}{r|l} 16 & \\ \hline 16 & \\ \hline 2 & 16 \\ \hline 0 & \\ \hline 2 & \end{array}$$

$$a_0(q) = 15 \quad a_1(q) = 2 \quad a_2(q) = 2$$

3. $a_{0(10)} = 15 \rightarrow a_{0(16)} = F$; $a_{1(10)} = 2 \rightarrow a_{1(16)} = 2$;
$a_{2(10)} = 2 \rightarrow a_{2(16)} = 2$.

4. $A_{(16)} = 22F$

Ответ: $559_{(10)} = 22F$.

Метод последовательного умножения на основание новой системы счисления. Этот метод используется для перевода из одной СС в другую только правильных дробей. При переводе числа из одной СС с основанием q, в другую с основанием p этим методом необходимо:

1. Представить основание новой СС p в СС с основанием q.

2. Последовательно умножать это число и промежуточные произведения на основание новой СС p, отбрасывая каждый раз целые части произведений. Эти целые части промежуточных произведений являются изображением цифр искомого p - ичного числа, записанные в старой СС.

3. Записать целую часть первого произведения в старший (a_{-1}) разряд p - ичного числа, а целые части i - тых произведений в последующие a_{-i} разряды.

4. Записать найденные цифры p - ичного числа, выраженные в СС с основанием q, в СС с основанием p .

Пример: $A_{(10)} = 0{,}675 \rightarrow A_{(2)}$

1. $p_{(10)} = 2$;
2. 0,675 x 2 = 1,350 $a_{-1} = 1$;
   0,350 x 2 = 0,7 $a_{-2} = 0$;
   0,7 x 2 = 1,4 $a_{-3} = 1$;
   0,4 x 2 = 0,8 $a_{-4} = 0$;
   0,8 x 2 = 1,6 $a_{-5} = 1$;
   0,6 x 2 = 1,2 $a_{-6} = 1$

и т.д.

3. Так как $1_{(10)} = 1_{(2)}$, $0_{(10)} = 0_{(2)}$, то $A_{(2)} = 0{,}101011$ и т.д.

Дробная часть (частичное произведение), полученная при последовательном умножении на основание, часто не принимает значения нуля. В этом случае процесс умножения продолжается до получения количества разрядов, определяемых разрядной сеткой числа или до обеспечения заданной точности представления числа.

Для перевода смешанных чисел (содержащих целую и дробную часть) целую часть получают методом деления на основание новой СС, а дробную - методом умножения на основание новой СС. Если основание одной СС является целой степенью основания другого (например $8 = 2^3$, $16 = 2^4$), то перевод чисел из одной СС в другую существенно упрощается, для этого каждую восьмеричную (шестнадцатеричную) цифру заменяют равным ей трехзначным (четырехзначным) двоичным числом, т.е. двоичной триадой (тетрадой).

1.2.4 Понятие о системах счисления в остаточных классах

Недостатком позиционных систем счисления является наличие связи между разрядами при выполнении арифметических действий, что проявляется в возникновении и учете распространения переносов и займов от младших разрядов к старшим. Это усложняет аппаратуру и уменьшает быстродействие цифровой ЭВМ.

Система счисления в остаточных классах (СОК) является непозиционной системой и позволяет выполнять арифметические действия, кроме деления, во всех разрядах одновременно и независимо (поразрядно).

Каждый разряд в СОК имеет свое основание. В качестве оснований системы выбирают ряд целых положительных взаимно простых

чисел $p_1, p_2, ..., p_n$. Основания системы определяют диапазон представления целых чисел от 0 до P:

$$P = p_1 \cdot p_2 \cdot ... \cdot p_n.$$

Любое целое число A, удовлетворяющее условию $0 \le A < P$, представляется в СОК набором остатков от деления A на основания системы, такое представление является единственным:

$$A_{сок} = a_1 \cdot a_2 \cdot ... \cdot a_n,$$

где a_i - наименьший неотрицательный остаток от деления числа A на основание p_i $(a_i = ост(A / p_i))$.

Пример. Записать число A = 17 в СОК, если p_1= 3, p_2= 5, p_3= 7.

Решение. Проверим выполнение условия $A < P$ представимости заданного числа в СОК с основаниями $\{3,5,7\}$:

$$P = 3 \cdot 5 \cdot 7 = 105, \quad 17 < 105.$$

Условие представимости числа выполняется (17<105), остатки по заданным основаниям будут следующими:

$$a_1 = ост(A / p_1) = ост(17/3) = 2;$$

$$a_2 = ост(A / p_2) = ост(17/5) = 2;$$

$$a_3 = ост(A / p_3) = ост(17/7) = 3.$$

Следовательно, $A_{сок}$= $2 \cdot 2 \cdot 3$.

Рассмотрим правила выполнения арифметических операций.

Правило сложения. При выполнении условия представимости, как слагаемых, так и суммы (A < P, B < P, A + B < P), для получения в СОК суммы $C_{сок} = (A + B)_{сок} = C_1 \cdot C_2 \cdot ... \cdot C_n$ слагаемых $A_{сок}$ и $B_{сок}$:

$$A_{сок} = a_1 \cdot a_2 \cdot ... \cdot a_n, \quad B_{сок} = в_1 \cdot в_2 \cdot ... \cdot в_n,$$

достаточно сложить остатки одноименных слагаемых (сложение следует производить по модулю соответствующего основания):

$$C_i = ост\left[\left(a_i + в_i \right) / p_i \right].$$

Пример. Сложить в СОК числа A = 8, B = 4 с набором оснований $\{3,5,7\}$.

Решение. Условия представимости выполняются, так как P= 3 · 5 · 7 = 105; A=8 < 105, B = 4 < 105, $A+B$= 8 + 4 = 12 < 105; $(B_{сок} = 1 \cdot 4 \cdot 4,\ A_{сок} = 2 \cdot 3 \cdot 1)$.

Согласно правилу сложения находим

+	$A_{СОК}$	=	2.	3.	1
	$B_{СОК}$	=	1.	4.	4
			3.	7.	5
	$(A+B)_{СОК}$	=	0.	2.	5

Действительно, $(12)_{сок} = 0 \cdot 2 \cdot 5.$

Правило умножения. Обозначим «*» символ операции умножения. При выполнении условия представимости $(A < P,\ B < P,\ A * B < P)$ для получения произведения $C_{сок} = (A * B)_{сок} = C_1 \cdot C_2 \cdot ... C_n$ двух сомножителей $A_{сок}$ и $B_{сок}$: $A = a_1 \cdot a_2 \cdot ... a_n,\ B_{сок} = в_1 \cdot в_2 \cdot ... в_n$ достаточно перемножить по соответствующим модулям одноименные остатки сомножителей:

$$C_i = ост\left[\left(a_i * в_i\right)\right] / p_i .$$

Пример. Перемножить в СОК числа с набором оснований $\{3,5,7\}$ $A = 8,\ B = 4.$

Решение. Так как $P = 3 \cdot 5 \cdot 7 = 105,\quad A = 8 < 105,\quad D = 4 < 105,$ $A * B = 32 < 105$, то

*	$A_{СОК}$	=	2.	3.	1
	$B_{СОК}$	=	1.	4.	4
			2.	12	4
	$(A*B)_{СОК}$	=	2.	2.	4

Действительно, $(32)_{сок} = 2 \cdot 2 \cdot 4.$

Главное преимущество СОК состоит в том, что основные арифметические операции, кроме деления, в СОК выполняют поразрядно, вследствие чего могут быть достигнуты значительно более высокие скорости выполнения операций, чем в позиционных системах.

К недостаткам СОК можно отнести:

1. Сложность выполнения операций деления и сравнения чисел.

2. Сложность определения факта переполнения разрядной сетки, т.е. выхода результата за диапазон представления чисел.

3. Сложность обратного перевода чисел из СОК в позиционную СС.

Указанные недостатки являются причиной того, что СОК находят пока ограниченное применение в специализированных ЭВМ, при решении задач, в которых преобладают операции сложения, вычитания и умножения.

Задания для самостоятельной работы

1. Перевести десятичные числа: 19,5; 32,125; 17,25; 44,5; 27,25; 52,6 в двоичную систему счисления.

2. Перевести двоичные числа: 11011, 101; 1101,011; 10101,11; 11001,101; 100001,1 в десятичную систему счисления.

3. Перевести десятичные числа: 20; 34; 17; 42; 87; 95 в восьмеричную систему счисления.

4. Перевести десятичные числа 125; 1022; 511; 62; 128; 100; 120 в двоичную систему счисления (в восьмеричную СС; в шестнадцатеричную СС).

5. Перевести десятичные дроби 0,5; 0,25; 0,375; 0,1; 0,6; 0,1275 в двоичную систему счисления с ошибкой, не превышающей значения 2^{-8}.

6. Записать в двоичной системе счисления число $A_{(8)} = 373,25$.

7. Записать в восьмеричной системе счисления число $A_{(2)} = 0100111010$.

8. Записать в восьмеричной системе счисления число $A_{(10)} = 389,79$.

9. Записать в двоичной системе счисления число $A_{(16)} = 3ДF,CA$.

10. Записать в восьмеричной системе счисления число $A_{(16)} = AF,ДВ2$.

11. Записать в десятичной системе счисления число $A_{(16)} = A3FE,2Д$.

12. Записать в двоичной системе счисления число $A_{(10)} = 67, 125$.

13. Записать в двоичной системе счисления число $A_{(10)} = 1,678$.

14. Записать в восьмеричной системе счисления число $A_{(10)} = 29,375$.

15. Записать в шестнадцатеричной системе счисления число $A_{(10)} = 738,275$.

1.3 Машинные двоичные коды и действия с ними

1.3.1 Формы представления чисел

В ЭВМ применяются две формы представления чисел: с фиксированной запятой и с плавающей запятой. Прежде чем рассмотреть каждую из форм представления чисел запишем некоторое произвольное число A в виде полинома

$$A = a_{n-1} \cdot q^{n-1} + a_{n-2} \cdot q^{n-2} + ... + a_0 \cdot q^0 + a_{-1} \cdot q^{-1} + ... + a_{-m} \cdot q^{-m}, \qquad (1.3)$$

где a_i - цифра из алфавита цифр выбранной позиционной системы счисления (ПСС);

q^i - основание ПСС;

i – коэффициент, учитывающий позицию (место расположения) цифры в составе числа.

Затем преобразуем формулу (1.3) к виду:

$$A = q^p \cdot \sum_{i=-m}^{n-1} a_i \cdot q^{i-p} = q^p \cdot M_A.$$

Здесь M_A - цифровая часть числа A, называемая его мантиссой, p - целое число, которое называют порядком. Множитель q^p называют масштабом.

Для чисел с фиксированной запятой (ФЗ) величина p постоянна и обычно равна n . При $p = n$ мантисса числа будет правильной дробью, причем

$$M_A \le 1 - q^{-r}, \qquad (1.4)$$

где r - количестве разрядов, используемых для записи мантиссы числа. Изображается M_A записью вида

$$M_A = 0, \quad a_{-1}a_{-2}...a_{-r}.$$

В машинах с ФЗ любое число A представляется только своей мантиссой, а масштаб q^p в машине никак не отображается. Поэтому масштабы чисел необходимо учитывать в алгоритмах решаемых задач. Это делается путем масштабирования всех исходных данных и результатов всех операций в математическом списании задачи так, чтобы не допустить нарушения условия (1.4). В противном случае происходит грубое искажение результата из-за потери в ЭВМ старших разрядов мантиссы.

Достоинством представления чисел в форме с ФЗ является малый расход оборудования для представления числовой информации и

высокая производительность арифметико-логического устройства (АЛУ), обусловленные простотой алгоритмов выполнения арифметических операций. Недостаток - необходимость осуществления масштабирования исходных данных на этапе подготовки вычислений.

Представление чисел в форме с плавающей запятой (ПЗ) требует отображения не только мантиссы M_A каждого числа, но и его масштаба q^p. Для представления последнего достаточно записать только его порядок p. Основное достоинство такого представления в том, что отпадает необходимость в масштабировании переменных. Недостатком является сложность алгоритмов реализации арифметических операций. Необходимость выполнения действий и над мантиссами чисел, и над их порядками ведет к увеличению объема оборудования и к снижению производительности машин.

К бортовым ЦВМ предъявляются жесткие ограничения на массогабаритные характеристики и время выполнения арифметических операций. Одним из путей достижения этих ограничений является использование представления чисел с фиксированной запятой. А сами числа должны быть представлены в виде правильных дробей, целая часть которых равна нулю. В связи с этим между числом и его представлением в ЦВМ существуют отличия. Число, преобразованное для размещения внутри ЦВМ, называется машинным кодом.

1.3.2 Машинные двоичные коды

В цифровых БВМ применяют прямой, обратный и дополнительный коды чисел. Многообразие применяемых кодов определяется их достоинствами и недостатками, а также особенностями используемых операционных устройств.

Прямой код удобен для хранения и передачи чисел, обратный и дополнительный коды - для выполнения арифметических операций. Рассмотрим эти коды на примере кодирования двоичных чисел с фиксированной запятой вида $A_{(2)} = \pm 0.a_{-1}...a_{-m} \left(|A| < 1\right)$. Индекс 2 для обозначения двоичных чисел будем в дальнейшем опускать.

Прямой код обозначают через $[\,A\,]_{\text{пк}}$:

$$[A]_{\text{пк}} = \begin{cases} 0.a_{-1}...a_{-m}, & \textit{если} \quad A \geq 0, \\ 1.a_{-1}...a_{-m}, & \textit{если} \quad A \leq 0. \end{cases} \qquad (1.5)$$

Нуль в прямом коде имеет два представления:

$$[+0]_{\text{пк}} = 0.0...0 \text{ и } [-0]_{\text{пк}} = 1.0...0.$$

Пример. A = + 0,101; B = - 0,101. Найти $[A]_{\text{пк}}$ и $[B]_{\text{пк}}$.

Решение. Согласно (1.5) находим: $[A]_{\text{пк}} = 0.101$; $[B]_{\text{пк}} = 1.101$.

Обратный код обозначается следующим образом: $[A]_{\text{ок}}$. Обратный код положительных чисел совпадает с прямым кодом. Для отрицательных чисел правило формирования обратного кода заключается в записи 1 в знаковый разряд и инвертировании (замена нуля единицей и наоборот) всех остальных разрядов числа.

$$[A]_{\text{ок}} = \begin{cases} 0.a_{-1}...a_{-2}...a_{-m}, & \textit{если} \quad A \geq 0, \\ 1.\overline{a}_{-1}\overline{a}_{-2}...\overline{a}_{-m}, & \textit{если} \quad A \leq 0. \end{cases}$$

Нуль имеет следующие два представления:

$$[+0]_{\text{ок}} = 0.0...0 \text{ и } [-0]_{\text{ок}} = 1.1...1.$$

Пример. A = + 0,101; B = - 0,101

Решение. Так как A положительное число, то $[A]_{ОК} = 0.101$. Для отрицательного B получим $[B]_{ОК} = 1.010$.

Дополнительный код $[A]_{\text{дк}}$ положительного числа равен его прямому коду. Отрицательное число A в процессе кодирования заменяется дополнением его прямого кода до двух т.е. величины $10.0_{(2)}$:

$$[A]_{\text{дк}} = 10. - [A]_{\text{пк}}, \text{ если } A < 0,$$

$$[A]_{\text{дк}} = \quad [A]_{\text{пк}}, \text{ если } A \geq 0.$$

Пусть A = - 0,1011. Найти $[A]_{\text{дк}}$.

В соответствии с правилом кодирования

$$\begin{array}{rl} & 1\ 1111{+}1 \\ 10_{(2)} = & 10.0000 \\ [A]_{\text{пк}} = & \underline{\ 1.1011} \\ [A]_{\text{дк}} = & \ 1.0101 \end{array} = [A]_{\text{ок}} + 1 \text{ мл. разряда}$$

Как видно, все цифры результата оказались инвертированными ($\overline{a}_i = 1$, если $a_i = 0$ и $\overline{a}_i = 0$, если $a_i = 1$), за исключением последней значащей единицы и знакового разряда.

Таким образом, можно сформулировать два правила перевода:

1. Дополнительный код отрицательного числа получается из прямого кода этого числа путем инвертирования всех разрядов, кроме знакового, и прибавлением единицы младшего разряда.

2. Для получения дополнительного кода отрицательного числа следует в прямом коде этого числа инвертировать все цифры кроме знаковой, последней значащей единицы и следующих за ней в более младших разрядах нулей.

Примеры:

$[A]_{пк}$ = 1.1010111; $[A]_{дк}$ = 1.0101000 + 0.0000001 = 1.0101001;

$[B]_{пк}$ = 1.1010000; $[B]_{дк}$ = 1.01011111 + 0.0000001 = 1.0110000.

В дополнительном коде нуль имеет единственное значение $[0]_{дк}$ =0.000...00.

Результат выполнения операций сложения или вычитания для дробных чисел может оказаться по абсолютной величине большим или равным единице, поэтому, чтобы не потерять значащую цифру результата в разряде единиц в обратных и дополнительных кодах, необходимо, в общем случае, предусмотреть представление цифра a_0 разряда единиц, при этом знак числа $a_{зн}$ будет уже представлен в разряде для двоек: $[A] = a_{зн} a_0 . a_{-1} ... a_{-m}$. Коды таких чисел называют модифицированными.

Модифицированный обратный код (МОК) числа A имеет вид:

$$[A]_{мок} = \begin{cases} 0 a_0 . a_{-1} ... a_{-m}, & если \quad A \geq 0, \\ 1 \overline{a}_0 . \overline{a}_{-1} ... \overline{a}_{-m}, & если \quad A \leq 0. \end{cases}$$

Модифицированный дополнительный код (МДК) обозначается через

$$[A]_{мдк} = \begin{cases} [A]_{мок}, & если\ A \geq 0, \\ [A]_{мок} + 1\ мл.разряда, & если\ A \leq 0. \end{cases}$$

1.3.3 Арифметические действия в кодах

Выполнение операций сложения и вычитания кодов. Правила сложения многоразрядных чисел (двух положительных или двух отрицательных), записанных в прямом коде, отличаются от правил сложения многоразрядных двоичных чисел только в определении знака. Знаковый разряд при суммировании кодов чисел с одинаковыми знаками равен знаковым разрядам слагаемых.

Пример:

$$
\begin{array}{r}
A=0{,}01011\\
\underline{B=0{,}00111}\\
A+B=0{,}10010
\end{array}
\qquad
\begin{array}{r}
+\begin{array}{r}[A]_{пк}=0{,}01011\\ [B]_{пк}=0{,}00111\end{array}\\
\hline
[A]_{пк}+\ [B]_{пк}=1{,}10010
\end{array}
$$

$$
\begin{array}{r}
A=-0{,}01011\\
\underline{B=-0{,}00111}\\
A+B=-0{,}10010
\end{array}
\qquad
\begin{array}{r}
+\begin{array}{r}[A]_{пк}=1{,}01011\\ [B]_{пк}=1{,}00111\end{array}\\
\hline
[A]_{пк}+\ [B]_{пк}=1{,}10010
\end{array}
$$

Сложение прямых кодов чисел, имеющих разные знаки осуществляется по правилу вычитания мантисс многоразрядных двоичных чисел. Вычитание всегда осуществляется из большего по абсолютной величине числа меньшего. Результату присваивается знак большего числа.

Пример:

$$
\begin{array}{r}
A=\ 0{,}01011\\
\underline{B=-0{,}00111}\\
A+(-B)=A-B=\ 0{,}00100
\end{array}
\qquad
\begin{array}{r}
-\begin{array}{r}[A]_{пк}=0{,}01011\\ [B]_{пк}=1{,}00111\end{array}\\
\hline
[A]_{пк}-[B]_{пк}=0{,}00100
\end{array}
$$

$$
\begin{array}{r}
A=-0{,}01011\\
\underline{B=\ 0{,}00111}\\
-A+B=-(A-B)=-0{,}00100
\end{array}
\qquad
\begin{array}{r}
-\begin{array}{r}[A]_{пк}=1{,}01011\\ [B]_{пк}=0{,}00111\end{array}\\
\hline
[A]_{пк}-[B]_{пк}=1{,}00100
\end{array}
$$

Во всех случаях при возникновении переноса из старшего разряда мантиссы в знаковый разряд фиксируется сбой нормальной работы машины, вызванный переполнением разрядной сетки.

При сложении кодов чисел, имеющих разный знак удобнее пользоваться обратными и дополнительными кодами. В этом случае знаковый разряд участвует в сложении наравне с разрядами мантиссы.

Пример: Обратные коды

$$
\begin{array}{r}
+\begin{array}{r}A=-0{,}01011\\ B=\ 0{,}00111\end{array}\\
\hline
-A+B\quad -0{,}00100
\end{array}
\qquad
\begin{array}{r}
+\begin{array}{r}[A]_{ок}=1{,}10100\\ [B]_{ок}=0{,}00111\end{array}\\
\hline
[A]_{ок}+[B]_{ок}=1{,}11011
\end{array}
\Rightarrow -0{,}00100
$$

$$
\begin{array}{r}
+\begin{array}{r}A=\ 0{,}01011\\ B=-0{,}00111\end{array}\\
\hline
A+(-B)=\ 0{,}00100
\end{array}
\qquad
\begin{array}{r}
+\begin{array}{r}[A]_{ок}=0{,}01011\\ [B]_{ок}=1{,}11000\end{array}\\
\hline
10{,}00011
\end{array}
\Rightarrow 0{,}00100
$$

При сложении обратных кодов двоичного числа единица переноса из знаковых разрядов прибавляется к младшему разряду результата.

Дополнительные коды

$A = -0,01011 \qquad [A]_{дк} = 1,10101$

$B = 0,00111 \qquad [B]_{дк} = 0,00111$

$$\begin{array}{rl} & [A]_{дк} = 1,10101 \\ + & [B]_{дк} = 0,00111 \\ \hline [A]_{дк} + [B]_{дк} = & 1,11100 \Rightarrow 1,11011_{ок} \Rightarrow -0,00100 \end{array}$$

$A = 0,01011 \qquad [A]_{дк} = 0,01011$

$B = -0,00111 \qquad [B]_{дк} = 1,11001$

$$\begin{array}{rl} & [A]_{дк} = 0,01011 \\ + & [B]_{дк} = 1,11001 \\ \hline & 10,00100 \Rightarrow 0,00100 \\ \hline \end{array}$$

В случае дополнительных кодов перенос из знаковых разрядов отбрасывается.

Выполнение операций умножения в кодах. При представлении двоичных чисел в виде кода для реализации операций умножения и деления проще всего использовать прямые коды двоичных чисел. В этом случае умножение и деление кодов осуществляется по правилам умножение и деления многоразрядных двоичных чисел. Знак результата определяется суммированием знаковых разрядов сомножителей по модулю 2, т.е. $0 \oplus 0=0$; $0 \oplus 1=1$;$1 \oplus 0=1$; $1 \oplus 1=0$.

В большинстве бортовых ЦВМ для представления двоичных чисел используются дополнительные коды. Особенность выполнения умножения двоичных чисел, представленных в дополнительном коде, заключается в том, что если хоть один из сомножителей является отрицательным, то результат будет неправильным. Для получения правильного результата необходимо осуществить коррекцию произведения.

Например, число X = - 0,1101 в дополнительном коде равно $[X]_{дк}$ = 1,0011, что может быть получено следующим образом

$$2 + X = 10,0000 + (-0, 1101)= 1,0011.$$

Используя равенство $[X]_{дк} = 2 + X$, определим, как должен выглядеть правильный результат умножения в различных случаях:

а) $X > 0, Y > 0$

$Z = X – Y$; $[Z]_{дк} = Z$;

$[X \cdot Y]_{дк} = X \cdot Y$;

б) $X > 0, Y < 0$

$Z = XY < 0$; $[Z]_{дк} = 2 + Z$;

$[X \cdot Y]_{дк} = 2 + X \cdot Y$;

в) $X < 0, Y > 0$

$Z = X \cdot Y < 0; [Z]_{дк} = 2 + Z;$

$[X \cdot Y]_{дк} = 2 + X \cdot Y;$

г) $X < 0, Y < 0$

$Z = X \cdot Y; [Z]_{дк} = Z;$

$[X \cdot Y]_{дк} = X \cdot Y.$

Для этих же вариантов, если просто перемножать дополнительные коды, получаются следующие результаты:

а) $X > 0, Y < 0$

$[X]_{дк} = X; [Y]_{дк} = Y,$

$[X]_{дк} \cdot [Y]_{дк} = X \cdot Y = [X \cdot Y]_{дк}.$

Результат правильный

б) $X > 0, Y < 0$

$[X]_{дк} = X, [Y]_{дк} = 2 + Y,$

$[X]_{дк} \cdot [Y]_{дк} = X \cdot (2 + Y) = 2X + X \cdot Y.$

Правильный результат равен $2 + X \cdot Y$.

в) $X < 0, Y > 0$

$[X]_{дк} = 2 + X, [Y]_{дк} = Y,$

$[X]_{дк} \cdot [Y]_{дк} = (2 + X) \cdot Y = 2Y + X \cdot Y.$

При правильном результате $2 + X \cdot Y$.

г) $X < 0, Y < 0$

$[X]_{дк} = 2 + X; [Y]_{дк} = 2 + Y;$

$[X]_{дк} \cdot [Y]_{дк} = (2 + X) \cdot (2 + Y) = 4 + 2X + 2Y - X \cdot Y;$

а нужно получить $X \cdot Y$.

Для получения правильного результата существует ряд способов выполнения операции умножения кодов с коррекцией. В бортовых ЦВМ Орбита 10, Орбита 20, Ц-100, А-15А умножение осуществляется по следующему правилу.

1. Умножаются мантиссы дополнительных кодов чисел, т.е. дополнительные коды без знакового разряда.

2. К полученному произведению добавляется корректирующее слагаемое.

Корректирующее слагаемое, в свою очередь, формируется по следующему правилу:

а) Если множитель отрицательный, то формируется дополнительный код отрицательной мантиссы множимого, такое число (мантисса) обозначается буквой М.

б) Если множимое отрицательное, то формируется дополнительный код от отрицательной мантиссы множителя.

в) Если оба сомножителя отрицательные, то формируется и первое и второе корректирующие слагаемые.

Пример:

I. $X > 0, Y > 0$

В БЦВМ $[X]_{дк} = X; [Y]_{дк} = Y$

Перемножение осуществляется без учета знаковых разрядов. Мантиссы кодов совпадают с кодами $X У = [X]_{дк} [Y]_{дк}$.

II. $X > 0, Y < 0$

$[X]_{дк} = X; [Y]_{дк} = 2 + Y$

$[X]_{дк} = 0,$ мантисса X

……

$[Y]_{дк} = \underbrace{1,\overbrace{\ldots\ldots\ldots}^{1+Y}}_{2+Y}.$ Например: $1{,}1011 \sim 2 + Y$

$0{,}1011 \sim 1 + Y$

В БЦВМ перемножаются мантиссы кодов и формируется произведение:

$$X (1 + Y) = X + XУ$$

Результат пока все равно неверный, поэтому к нему прибавляется корректирующее слагаемое К1:

$$X + XY + К1 = 2 + XУ$$

Отсюда $К1 = 2 - X$.

К1 получается путем формирования дополнительного кода отрицательной мантиссы множимого (X).

$$X > 0 \text{ т.е. } M_x = X ,$$

где М - мантисса.

$$К1 = [-M_x]_{дк} = [-X]_{дк} = 2 + (-X) = 2 – X.$$

Таким образом, сформировано первое корректирующее слагаемое.

III. $X < 0, Y > 0$

$[X]_{дк} = 2 + X, M_x = 1 + X$

$[Y]_{дк} = Y$

Результат перемножения $(1 + X) \cdot Y = Y + XY$ требует введения корректирующего слагаемого

$Y + X \cdot Y + К2 = 2 + XY$, отсюда $К2 = 2 - Y$.

Второе слагаемое формируется по правилу:

$$К2 = [-Y]_{дк} = 2 + (-Y) = 2 – Y$$

IV. $X < 0, Y < 0$

$[X]_{дк} = 2 + X, M_x = 1 + X$

$[Y]_{дк} = 2 + Y; M_y = 1 + Y$

$(1 + X) \cdot (1 + Y) = 1 + X + Y + XY \neq XY.$

В машине формируются два корректирующих слагаемых.

$K1 = [-M_x]_{дк} = [-(1 + X)] = 2 + (-(1 + X)) = 1 – X.$

$K2 = [-Y]_{дк} = -Y$, так как $Y < 0$, то $– Y > 0$.

Прибавление корректирующих слагаемых дает:

$1 + X + Y + XY + K1 + K2 = 1 + X + Y + X \cdot Y + 1 – X – Y = 2 + X \cdot Y$

Сумма $2 + X \cdot Y$ составляет 10,..... в двоичной системе счисления.

Так как старшая единица в целой части выпадает из разрядной сетки, результат будет равен 0,...., что соответствует правильному произведению ХУ.

Пример:

$$X = -0{,}1011$$
$$Y = -0{,}1010$$

Получим результат перемножения чисел по обычным правилам двоичной арифметики:

```
          1 0 1 1
          1 0 1 0
          0 0 0 0
        1 0 1 1
      0 0 0 0
    1 0 1 1
+0 , 0 1 1 0 1 1 1 0
```

В БЦВМ:

$[X]_{дк} = 1{,}0101, \quad M_x = 0{,}0101 = 1 + X$

$[Y]_{дк} = 1{,}0110, \quad M_y = 0{,}0110 = 1 + Y$

```
            0 1 0 1
            0 1 1 0
            0 0 0 0
          0 1 0 1
        0 1 0 1
      0 0 0 0
1 + X + Y + X Y = 0 , 0 0 0 1 1 1 1 0
```

Корректирующие слагаемые равны

$$K1 = [-M_x]_{дк} = 1{,}1011$$
$$K2 = [-Y]_{дк} = 0{,}1010$$

Прибавляя корректирующие слагаемые к произведению, получим

$$0{,}00011110 + 1{,}1011 + 0{,}1010 = 0{,}01101110$$

Что соответствует правильному результату.

Операция деления двоичных чисел представленных дополнительными кодами сводится к операциям сложения (вычитания) и сдвига.

В БЦВМ операция деления является самой длительной и осуществляется за несколько циклов, в каждом цикле определяется одна цифра частного. В первом цикле выполняется операция вычитания делимого и делителя, если знаки одинаковые или операция сложения, если знаки разные. При разных знаках частное формируется в обратном коде, а после окончания деления к его младшему разряду прибавляется единица для перевода в дополнительный код. В каждом последующем цикле выполняется либо операция сложения, либо операция вычитания, при этом предыдущий остаток удваивается. Если частное в данном разряде получилось равным нулю, то в следующем цикле выполняется операция вычитания, если единице, то сложение. По знаковым разрядам очередного остатка в зависимости от знака делителя формируется соответствующая цифра частного.

Пример:

$$X = -0,011011$$
$$Y = 0,1001$$
$$\frac{X}{Y} = -0,11$$

11011	100100
100100	0,11
0100100	
100100	
0	

$[X]_{дк} = 1,100101$

$[Y]_{дк} = 0,1001$

I цикл $[X]_{дк} + [Y]_{дк} \begin{matrix} \rightarrow \le 0 \rightarrow \text{переполнение} \\ \rightarrow > 0 \rightarrow Z = -0,...; [Z]_0 = 1,... \end{matrix}$

$1,100101 + 0,1001 = 1\ \ 0,001001; \quad [Z]_{ок} = 1,\ldots$

II цикл $[X]_{дк} + [Y]_{дк} - \frac{Y}{2} \begin{matrix} \rightarrow < 0 & [Z]_{ок} = 0 \\ \rightarrow > 0 & [Z]_{ок} = 1 \end{matrix}$

III цикл $[X]_{дк} + [Y]_{дк} - \frac{Y}{2} + \frac{Y}{4} \begin{matrix} \rightarrow \le 0 & [Z]_{ок} = 0 \\ \rightarrow > 0 & [Z]_{ок} = 1 \end{matrix}$

-0,001001 + 0,001001 = 0; $[Z]_{ок} = 1{,}00$

$[Z]_{дк} = [Z]_{ок} + 0{,}01 = 1{,}01$

Z = - 0,11

Задания для самостоятельной работы

1. A = 0,101011, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
2. A = -0,100001 , $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
3. A = -0,000000, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
4. A = 0,000000, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
5. A = -0,100010, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
6. A = -0,0000100, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
7. A = -0,101001, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
8. A = 0,100001, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
9. A = 0,000001, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
10. A = 0,0010101, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
11. A = -0,101111, $[A]_{ДК}, [A]_{ОК}, [A]_{ПК}$ - ?
12. Выполнить сложение чисел в прямом, обратном и дополнительном кодах: A = 0,010111 и B = -0,00101.
13. Выполнить сложение чисел в прямом, обратном и дополнительном кодах: A = -0,00111 и B = 0,10001.
14. Выполнить сложение чисел в прямом, обратном и дополнительном кодах: A = 0,0101 и B = -0,000101.
15. Выполнить сложение чисел в прямом, обратном и дополнительном кодах: A = 0,001111 и B = -0,10010.
16. Выполнить сложение чисел в прямом, обратном и дополнительном кодах: A = -0,000010 и B = 0,100001.
17. Определить произведение дополнительных кодов чисел A = -0.1001 и B = 0.01.
18. Определить произведение дополнительных кодов чисел A = 0.0111 и B = -0.1001
19. Определить произведение дополнительных кодов чисел A = -0.11 и B = -0.101
20. Выполнить деление дополнительных кодов до четырех цифр после запятой. Делимое x = - 0.01011 и делитель y = 0.01101.

1.4 Общие сведения о функциях алгебры логики

Наука рассуждать (т.е. логика) была создана в IV веке до н.э. древнегреческим мыслителем Аристотелем. Он рассмотрел какие законы, приемы, формы присущи человеческому мышлению. Отсюда и название логики – формальная логика.

Основоположником математической логики считается немецкий математик Лейбниц. Он в XVII веке попытался сформулировать первые логические исчисления.

На фундаменте, заложенном Лейбницем, другой математик, Джордж Буль, отец писательницы Войнич, автора романа «Овод», ввел для логического построения особую алгебру. В отличие от обычной, в ней символами обозначают не числа, а высказывания. Все высказывания имеют всего только два значения – ложь или истина. Так как в двоичной системе счисления только две цифры 0 и 1, то правила алгебры логики приемлемы для двоичной СС.

1.4.1 Функции и операции

При синтезе и анализе схем цифровых устройств используется аппарат математической логики, объектом исследования которого являются двоичные функции f двоичных переменных $x_0,...,x_{n-1}$, называемые булевыми функциями или функциями алгебры логики:

$$f = f(x_0,...,x_{n-1}).$$

Аппарат булевой алгебры имеет дело с переменными, отражающими истинность или ложность высказываний: если высказывание истинно – ему ставится в соответствие значение $x = 1$, если ложно - $x = 0$.

Функция f, также как и ее аргументы, принимает значение 0 или 1.

Совокупность значений $d_0,...,d_{n-1}$ аргументов $x_0,...,x_{n-1}$ называется набором длины n и обозначается через $(d_0,...,d_{n-1})$. Число D, соответствующее набору, называется номером набора (таблица 1.2).

Множество наборов $W = \{(0...0),...,(1...1)\}$ характеризует область определения функции. Значение функции f на наборе с номером D обозначается через f_D:

$$f_D = f(d_0,...,d_{n-1}).$$

Из множества наборов принято выделять нулевой и единичный – это соответственно наборы (0…0) (номер набора равен нулю) и (1…1) (номер набора равен $2^n - 1$). Два набора $(d_0,...,d_{n-1})$ и $(\bar{d}_0,...,\bar{d}_{n-1})$ называются противоположными, если значения одноименных переменных в наборах взаимно противоположны $(\bar{d}_i = 1 - d_i)$. Два набора $(d_0...0...d_{n-1})$ и $(d_0...1...d_{n-1})$, различающиеся значениями только одной переменной, называют соседними.

Таблица 1.2 Таблица наборов длины 3

Номер набора D	Значение аргументов (набор)		
	d_2	d_1	d_0
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Поскольку область определения и область значений функции конечны, то она может быть задана таблицей, в которой каждому набору с номером D сопоставляют значение функции f_D. Такие таблицы называют таблицами истинности (таблица 1.3).

Таблица 1.3 Пример задания различных функций таблицей истинности

Номер набора	Набор		Значения функции	Пример функций		
	d_1	d_0		$f(d_1d_0)$	$f^*(d_1d_0)$	$\varphi(d_1d_0)$
0	0	0	f_0	0	0	1
1	0	1	f_1	1	1	0
2	1	0	f_2	1	*	1
3	1	1	f_3	0	*	0

Значениям функций $f_0 ... f_{2^n-1}$ можно сопоставить двоичный код $F_{(2)}$, такой что

$$F_{(2)} = f_0 f_1 ... f_{2^n-1}.$$

Множество W наборов аргументов области определения функции можно разбить на два подмножества W_1 и W_0:

$$W = W_1 \bigcup W_0,$$

где W_1 и W_0 - подмножества наборов, на которых функция принимает значения 1 и 0 соответственно.

В частности, для $f(d_1, d_0)$ из таблицы 1.3 находим:

$$W_1 = \{(01), (10)\}, \quad W_0 = \{(00), (11)\}.$$

На части наборов функция может быть не определена, такие функции называют частичными и помечают звездочками. Наборы, на которых функция может принимать одновременно противоположные значения, называют запрещенными, значения функции на таких наборах также отмечают звездочками. Например, функция $f^*(d_1, d_0)$ не определена на наборах с номерами 2 и 3 (см. таблицу 1.3).

На практике частичные функции на запрещенных наборах доопределяют для того, чтобы полученная функция имела более простую формулу, либо допускала бы более простую схемную реализацию.

Количество N_n различных наборов длины n в точности равно количеству различных 2^n - разрядных двоичных чисел, а количество N_f различных функций – количеству различных наборов N_n:

$$N_n = 2^n, \quad N_f = 2^{N_n} = 2^{2^n}.$$

Для $n = 1$ количество различных значений $N_n = 2$, а $N_f = 2^2 = 4$ (таблица 1.4). Как следует из таблицы 1.4, значения функций F_0, F_3 - постоянны и не зависят от изменения входной переменной, а функция F_1 совпадает с d_0. Единственной нетривиальной функцией является функция $F_2(d_0) = \bar{d}_0$ - инверсия переменной d_0.

Таблица 1.4 - Функции одной переменной

D	d_0	F_0	F_1	F_2	F_3
0	0	0	0	1	1
1	1	0	1	0	1

Для $n = 2$ количество различных наборов $N_n = 4$, а количество различных функций $N_f = 2^4 = 16$ (таблица 1.5).

Таблица 1.5- Логические функции двух переменных

D	*Набор*		*F0*	*F1*	*F2*	*F3*	*F4*	*F5*	*F6*	*F7*	*F8*
	d_1	d_0	0	$\overline{d_0 \vee d_1}$					$d_0 \oplus d_1$	$\overline{d_0 \,\&\, d_1}$	$d_0 \,\&\, d_1$
0	0	0	0	1	0	1	0	1	0	1	0
1	0	1	0	0	1	1	0	0	1	1	0
2	1	0	0	0	0	0	1	1	1	1	0
3	1	1	0	0	0	0	0	0	0	0	1

Продолжение таблицы 1.5

D	*Набор*		*F9*	*F10*	*F11*	*F12*	*F13*	*F14*	*F15*
	d_1	d_0	$\overline{d_0 \oplus d_1}$		$d_0 \to d_1$			$d_0 \vee d_1$	1
0	0	0	1	0	1	0	1	0	1
1	0	1	0	1	1	0	0	1	1
2	1	0	0	0	0	1	1	1	1
3	1	1	1	1	1	1	1	1	1

Количество различных функций с увеличением числа аргументов растет чрезвычайно быстро, достаточно быстро растет и число различных наборов. Так для $n = 2$, $N_n = 4$, $N_f = 16$, но для $n = 5$ уже $N_n = 32$, $N_f > 4000000000$, поэтому табличное задание функций становится громоздким, и функции предпочитают задавать алгебраически формулами через простейшие (элементарные) функции – операции.

В качестве операций обычно выбирают следующие функции одного и двух аргументов (см. таблицы 1.4, 1.5): инверсия (операция НЕ: $y = \bar{x}$), конъюнкция (операция И: $y = x_1 \,\&\, x_0$), дизъюнкция (операция ИЛИ: $y = x_1 \vee x_0$), штрих Шеффера (операция И-НЕ: $y = \overline{x_1 \,\&\, x_0}$), стрелка Пирса (операция ИЛИ-НЕ: $y = \overline{x_1 \vee x_0}$), сумма по модулю два (операция М2: $y = x_1 \oplus x_0$).

Инверсия, отрицание или операция НЕ реализуется инвертором или логическим элементом (ЛЭ) НЕ (рисунок 1.3,а).

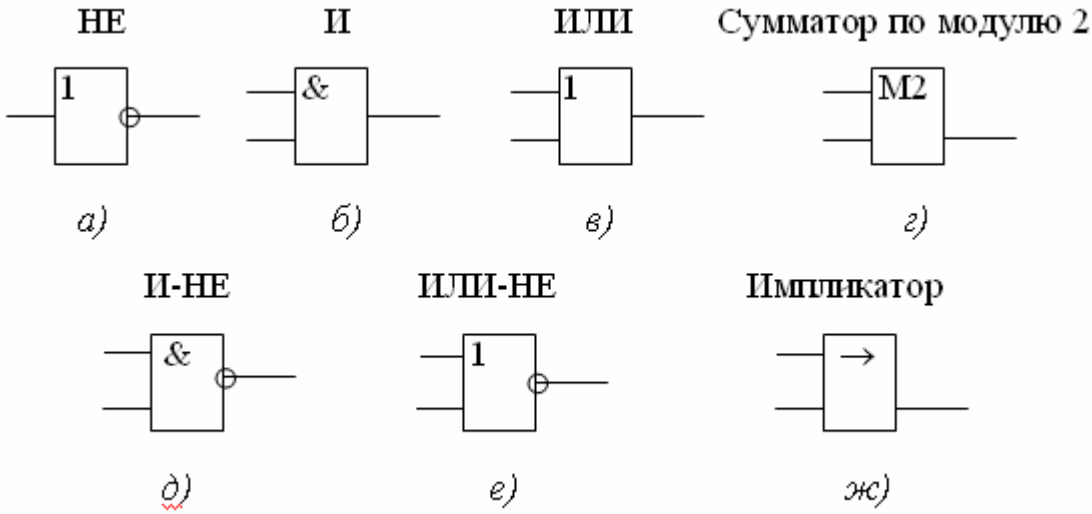


Рисунок 1.3- Логические элементы

Конъюнкция, логическое произведение (умножение) или операция И – это функция $x \,\&\, y$, совпадающая с арифметическим произведением двоичных переменных. Конъюнкцию обозначают также через xy, $x \cdot y$, $x \wedge y$. Аргументы функции часто называют сомножителями.

Элемент, реализующий конъюнкцию, называют конъюнктором или логическим элементом И (см. рисунок 1.3,б). Говорят (см. таблицу 1.5), что элемент И всегда пропускает нули со входов на выход (действительно, если хоть один из сомножителей равен нулю, то и произведение равно нулю).

Дизъюнкция, логическая сумма (сложение) иначе операция ИЛИ – это функция $x \vee y$, которая равна нулю тогда и только тогда, когда все ее аргументы равны нулю. Аргументы дизъюнкции часто называют слагаемыми. Элемент, реализующий дизъюнкцию, называют дизъюнктором или логическим элементом ИЛИ (см. рисунок 1.3,*в*). Говорят, что элемент ИЛИ всегда пропускает единицы со входов на выход.

Сумма или операция сложения по модулю два – это функция $x \oplus y$, которая равна единице тогда и только тогда, когда сумма значений аргументов есть число нечетное. Элемент, реализующий эту функцию, называют сумматором по модулю два и отмечают символом М2 (см. рисунок 1.3,*г*).

В математической логике широко используют функцию «импликация». Импликация – это функция $x \to y \quad (x \to y = \bar{x} \vee y)$, которая равна единице, если выполняется неравенство $x \le y$ (см. таблицу 1.5). Элемент, реализующий импликацию, называют импликатором (см. рисунок 1.3,*ж*).

Функцию, противоположную конъюнкции, называют функцией штрих Шеффера или операцией И-НЕ и обозначают через $x / y \quad (x / y = \overline{x \& y})$. Логический элемент, реализующий эту функцию, называют элементом Шеффера или элементом И-НЕ (см. рисунок 1.3,*д*).

Функцию, противоположную дизъюнкции, называют функцией стрелка Пирса или операцией ИЛИ-НЕ и обозначают через $x \uparrow y \quad (x \uparrow y = \overline{x \vee y})$. Элемент, реализующий эту функцию, называют элементом Пирса или элементом ИЛИ-НЕ (см. рисунок 1.3,*е*). Все перечисленные функции за исключением функций инверсии и импликации справедливы для любого числа ($n > 2$) аргументов.

1.4.2 Основные законы и правила алгебры логики

Основные тождества булевой алгебры, используемые для преобразования формул функций, получили название законов и правил. После определения операций алгебры эти тождества являются следствиями этих определений и могут быть доказаны.

Законы коммутативности (переместительные) для дизъюнкции и конъюнкции

$$x \vee y = y \vee x; \quad x \cdot y = y \cdot x.$$

Законы ассоциативности (сочетательные) для дизъюнкции и конъюнкции

$$x \vee (y \vee z) = (x \vee y) \vee z; \quad x \cdot (y \cdot z) = (x \cdot y) \cdot z.$$

Первый и второй законы дистрибутивности (распределительные)

$$x \cdot (y \vee z) = x \cdot y \vee x \cdot z; \quad x \vee (y \cdot z) = (x \vee y) \cdot (x \vee z).$$

Законы идемпотентности (повторения) для дизъюнкции и конъюнкции

$$x \vee x \vee ... \vee x = x; \quad x \cdot x \cdot ... \cdot x = x.$$

Законы отрицания (инверсии)

$$x \vee \bar{x} = 1; \quad x \cdot \bar{x} = 0; \quad \bar{\bar{x}} = x.$$

Законы двойственности или «правило де Моргана»

$$\overline{x \vee y} = \bar{x} \cdot \bar{y}; \quad \overline{x \cdot y} = \bar{x} \cdot \bar{y}.$$

Правило свертки

$$x \vee \bar{x} \cdot y = x \vee y.$$

Правила поглощения

$$x \vee x \cdot y = x; \quad x \cdot (x \vee y) = x.$$

Правила полного склеивания

$$x \cdot y \vee x \cdot \bar{y} = x; \quad (x \vee y) \cdot (x \vee \bar{y}) = x.$$

Правило неполного склеивания

$$x \cdot y \vee x \cdot \bar{y} = x \vee x \cdot y \vee x \cdot \bar{y}.$$

Правило Порецкого

$$x \cdot y \vee \bar{y} \cdot z \vee x \cdot z = x \cdot y \vee \bar{y} \cdot z.$$

Правила операций с константами

$0 \cdot 0 = 0$	$0 \cdot 1 = 0$	$1 \cdot 0 = 0$	$1 \cdot 1 = 1$
$0 \vee 0 = 0$	$0 \vee 1 = 1$	$1 \vee 0 = 1$	$1 \vee 1 = 1$
$x \cdot 0 = 0$	$x \cdot 1 = x$	$x \vee 0 = x$	$x \vee 1 = 1$

Доказательство большинства законов и правил алгебры логики очевидны.

Например,

$$x \vee x \cdot y = x \cdot (1 \vee y) = x \cdot 1 = x;$$

$$x \cdot y \vee x \cdot \bar{y} = x \cdot (y \vee \bar{y}) = x \cdot 1 = x;$$

$$(x \vee y) \cdot (x \vee \bar{y}) = x \cdot x \vee x \cdot \bar{y} \vee x \cdot y \vee y \cdot \bar{y} = x \vee x \cdot \bar{y} \vee 0 =$$

$$= x \vee x \cdot \bar{y} = x \cdot (1 \vee \bar{y}) = x.$$

В других случаях доказательство тождества может быть выполнено сравнением значений правой и левой части тождества на всех возможных наборах переменных.

Например, необходимо доказать справедливость тождества:

$$\overline{x \vee y} = \bar{x} \cdot \bar{y}.$$

Зададим таблицей (таблица 1.6) значения выражений $\overline{x \vee y}$ и $\bar{x} \cdot \bar{y}$.

Значения обоих выражений (см. таблицу 1.6) на всех наборах значений аргументов совпадают, следовательно, они задают одну и ту же функцию. Тождество доказано.

Таблица 1.6

Номер набора	Набор (xy)	$\overline{x \vee y}$	$\bar{x} \cdot \bar{y}$
0	0 0	$\overline{0 \vee 0} = \bar{0} = 1$	$\bar{0} \cdot \bar{0} = 1 \cdot 1 = 1$
1	0 1	$\overline{0 \vee 1} = \bar{1} = 0$	$\bar{0} \cdot \bar{1} = 1 \cdot 0 = 0$
2	1 0	$\overline{1 \vee 0} = \bar{1} = 0$	$\bar{1} \cdot \bar{0} = 0 \cdot 1 = 0$
3	1 1	$\overline{1 \vee 1} = \bar{1} = 0$	$\bar{1} \cdot \bar{1} = 0 \cdot 0 = 0$

Логические выражения могут быть использованы и в обычной жизни.

Например. Три девушки – Аня (А), Валя (В), Клава (К) ходили на танцы в военное училище. Одна из них была в красном платье, другая – в синем, третья в белом. Курсант Иванов был в наряде и на танцы не попал, а его друзья (сослуживцы) Новиков, Петров и Сидоров после танцев сообщили:

Аня надела красное платье (Ак)

Валя – не красное ($\overline{\text{В}}$к)

Клава – не синее ($\overline{\text{К}}$с)

Стало известно, что только одно из сообщений истинно. Как определить в каком платье была каждая из девушек.

Решение: Из трех вариантов ответа порождаемых сообщением $\text{Ак}\overline{\text{В}}\text{к}\overline{\text{К}}\text{с}$: АкВкКс; $\overline{\text{А}}\text{кВк}\overline{\text{К}}\text{с}$; $\overline{\text{А}}\text{к}\overline{\text{В}}\text{кКс}$ только один истинный, а именно $\overline{\text{А}}\text{кВк}\overline{\text{К}}\text{с} = 1$ поэтому Аня была в синем, Валя была в красном платье, Клава в белом.

1.4.3 Алгебры Жегалкина, Шеффера и Пирса

Помимо булевой алгебры, в теории функций алгебры логики и, особенно, в приложениях этой теории к задачам синтеза и анализа схем ЭВМ используют и другие алгебраические системы, в том числе алгебры Жегалкина, Шеффера и Пирса.

Кроме того, при выборе оптимальных схем по сложности, а также при переходе от одной системы элементов ЭВМ к другой возникает необходимость в использовании систем операций, принадлежащих различным алгебрам.

Алгебра Жегалкина – это множество всех функций, рассматриваемое совместно с операциями конъюнкция, сумма по модулю 2 и

константа единицы. В алгебре Жегалкина любую функцию рассматривают (записывают) только в виде суперпозиции упомянутых выше операций.

Для алгебры Жегалкина действуют следующие законы:

коммутативности $x_1 \oplus x_2 = x_2 \oplus x_1$,

ассоциативности $x_1 \oplus (x_2 \oplus x_3) = (x_1 \oplus x_2) \oplus x_3$,

дистрибутивности $x_1(x_2 \oplus x_3) = x_1x_2 \oplus x_1x_3$.

Действия с константами:

$$x \oplus 1 = \bar{x}; \quad x \oplus x = 0.$$

Алгебра Шеффера – это множество всех функций, рассматриваемое совместно с операцией штрих Шеффера. Функция Шеффера не подчиняется закону ассоциативности (сочетательному)

$$\overline{x_1 \,\&\, \overline{(x_2 \,\&\, x_3)}} \neq \overline{\overline{(x_1 \,\&\, x_2)} \,\&\, x_3}$$
$$x_1 / (x_2 / x_3) \neq (x_1 / x_2) / x_3.$$

Доказательство этого высказывания может быть выполнено сравнением таблиц истинности логических выражений левой и правой части неравенства (таблица 1.7).

Функция Шеффера подчиняется переместительному закону (коммутативности):

$$\begin{matrix} \overline{x_1 \,\&\, x_2} = \overline{x_2 \,\&\, x_1} \\ x_1 / x_2 = x_2 / x_1 \end{matrix}.$$

Таблица 1.7

x_1	x_2	x_3	y_1	y_2
0	0	0	1	1
0	0	1	1	0
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	1
1	1	1	1	1

Функция Шеффера тождественно равна инверсии конъюнкций или дизъюнкции инверсии аргументов

$$x_1 / x_2 / ... / x_n = \overline{x_1 \,\&\, x_2 \,\&\, ... \,\&\, x_n} = \bar{x}_1 \vee \bar{x}_2 \vee ... \vee \bar{x}_n.$$

Действия с константами

$$x_1 / 0 = \overline{x_1 \,\&\, 0} = 1$$

$$x_1 / 1 = \overline{x_1 \,\&\, 1} = \bar{x}_1$$

$$x_1 / x_1 = \overline{x_1 \,\&\, x_1} = \bar{x}_1.$$

Алгебра Пирса – это множество всех функций, рассматриваемое совместно с операцией стрелка Пирса.

Функция Пирса тождественно равна инверсии дизъюнкции аргументов или конъюнкции инверсий аргументов:

$$x_1 \uparrow x_2 \uparrow ... \uparrow x_n = \overline{x_1 \vee x_2 \vee ... \vee x_n} = \bar{x}_1 \,\&\, \bar{x}_2 \,\&\, ... \,\&\, \bar{x}_n.$$

Для алгебры Пирса выполняется переместительный закон:

$$x_1 \uparrow x_2 = x_2 \uparrow x_1$$

$$\overline{x_1 \vee x_2} = \overline{x_2 \vee x_1}.$$

Также как и для алгебры Шеффера, сочетательный закон не выполняется. Действия с константами:

$$x \uparrow 0 = \overline{x \vee 0} = \bar{x}$$

$$x \uparrow 1 = \overline{x \vee 1} = 0$$

$$x \uparrow x = \overline{x \vee x} = \bar{x}.$$

Алгебры Жегалкина, Шеффера и Пирса позволяют существенно упрощать решение задач проектирования логических схем ЭВМ на интегральных микросхемах, среди которых важное место занимают элементы Шеффера (И-НЕ) и Пирса (ИЛИ-НЕ), а также для реализации контроля по четности используются сумматоры по модулю 2, составляющие базис алгебры Жегалкина.

1.4.4 Элементарные конъюнкции и дизъюнкции

Функции в булевой алгебре записываются только в виде суперпозиции операций конъюнкции, дизъюнкции, инверсии, то есть в виде формул из символов операций И, ИЛИ, НЕ. При выполнении операций в булевой алгебре следует соблюдать определенную очередность. Сначала выполняют операции инверсии, затем операции конъ-

юнкции и в последнюю очередь операции дизъюнкции. При наличии скобок в первую очередь выполняются операции в скобках.

Пример. Вычислить значение функции

$f(x_2, x_1, x_0) = (x_2 \cdot \bar{x}_0 \vee \bar{x}_1) \cdot \bar{x}_0 \vee \overline{(x_2 \vee x_0)}$ на шестом наборе.

Решение. Для шестого набора $x_2 = 1,\ x_1 = 1,\ x_0 = 0$. Подставив эти значения в формулу функции и выполнения последовательно операции, получаем:

$$f(1,1,0) = (1 \cdot \bar{0} \vee \bar{1}) \cdot \bar{0} \vee \overline{(1 \vee 0)} = (1 \cdot 1 \vee 0) \cdot 1 \vee \bar{1} = 1 \cdot 1 \vee 0 = 1 \vee 0 = 1.$$

Вычисляя значения функции на каждом наборе, можно легко построить таблицу истинности и тем самым осуществить переход от задания функции формулой к заданию ее таблицей.

Важную роль в булевой алгебре играют функции называемые элементарными конъюнкциями и элементарными дизъюнкциями.

Элементарная конъюнкция (ЭК) – это логическое произведение попарно различных отрицаемых или неотрицаемых аргументов (сомножителей).

Очевидно, ЭК равна единице тогда и только тогда, когда все ее сомножители с учетом признака отрицания равны единице. ЭК называют полной относительно n аргументов или конституентой единицы, если в нее входит каждый из этих аргументов. Конституенту единицы обозначают через $K(j)$, где j - номер единственного набора, на котором конституента равна единице. Если в наборе значение цифры двоичного кода набора равно 0, то переменная в записи конституенты инвертируется, если значение цифры – единица, то переменная записывается без инверсии.

Пример,
$$\begin{array}{l} K(3) = \bar{x}_2 \cdot x_1 \cdot x_0 \\ j = 3 \quad 0 \quad 1 \quad 1 \end{array}.$$

На всех наборах, кроме j-го значение конституенты единицы $K(j)=0$.

Элементарная дизъюнкция (ЭД) – это логическая сумма отрицаемых и неотрицаемых аргументов (слагаемых). Очевидно, ЭД равна нулю тогда и только тогда, когда каждый из ее слагаемых, с учетом признака отрицания, равен нулю.

ЭД, полную относительно n аргументов, называют конституентой нуля и обозначают через $M(j)$, где j - номер единственного набора аргументов, на котором конституента равна нулю. Если в j-м

наборе значение цифры двоичного кода набора равно 1, то переменная в записи конституенты инвертируется, если значение цифры – 0, то переменная записывается без инверсии.

Пример, $\begin{matrix} M(3) = x_2 \vee \overline{x}_1 \vee \overline{x}_0 \\ j = 3 \quad 0 \quad 1 \quad 1 \end{matrix}$.

На всех остальных наборах $M(j) = 1$.

Непосредственно из определения конституент следует справедливость следующей системы тождеств ($i \neq j$):

$$K(j) = \overline{M}(j), \qquad M(j) = \overline{K}(j);$$
$$K(i) \,\&\, K(j) = 1, \qquad M(i) \vee M(j) = 0.$$

1.5 Переключательные функции и их минимизация

1.5.1 Канонические формы функций

Для практической реализации цифровых устройств часто используют ключевые схемы с двумя устойчивыми состояниями, поэтому логические функции, описывающие работу цифровых устройств, называют переключательными функциями (ПФ).

Одна и та же функция может быть задана бесчисленным множеством формул. Неоднозначность представления функций в виде формул существенно усложняет установление различных соотношений между функциями и задающими их таблицами, например, установление тождества между функциями.

Поэтому из всех форм представления переключательных функций особый интерес имеют формы однозначного представления. Такие формы называют стандартными или каноническими. Как и таблицы истинности, канонические формы часто используют в качестве исходных форм задания переключательных функций.

В булевой алгебре используют две канонические формы – совершенную дизъюнктивную нормальную форму (СДНФ) и совершенную конъюнктивную нормальную форму (СКНФ).

Для получения СДНФ переключательной функции необходимо выполнить дизъюнкцию конституент единиц тех наборов, на которых ПФ равна 1: СДНФ $f = \bigvee\limits_{j \in W_1} K(j)$.

Правило перехода от табличного способа задания цифрового устройства к СДНФ ПФ вытекает из определения. Например, для за-

данной случайным образом таблицы истинности (таблица 1.8) СДНФ переключательной функции имеет следующий вид:

СДНФ $f(x_2, x_1, x_0) = K(1) \vee K(3) \vee K(4) = \bar{x}_2\bar{x}_1x_0 \vee \bar{x}_2x_1x_0 \vee x_2\bar{x}_1\bar{x}_0$.

Таблица 1.8

Номер набора	Значение аргументов			Значение функции $f(x_2, x_1, x_0)$
	x_2	x_1	x_0	
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	1	1	0
6	1	0	0	0
7	1	1	1	0

Рассмотрим несколько примеров получения СДНФ (таблицы 1.9, 1.10).

Таблица 1.9

Номер набора	Значение аргументов			Значение функции $f(x_2, x_1, x_0)$
	x_2	x_1	x_0	
0	0	0	0	0
1	0	0	1	1
2	0	1	0	1
3	0	1	1	1
4	1	0	0	0
5	1	1	1	0
6	1	0	0	0
7	1	1	1	1

СДНФ переключательной функции по данным таблицы 1.9 имеет вид: $f(x_2, x_1, x_0) = \bar{x}_2 \cdot \bar{x}_1 \cdot x_0 \vee \bar{x}_2 \cdot x_1 \cdot \bar{x}_0 \vee \bar{x}_2 \cdot x_1 \cdot x_0 \vee x_2 \cdot x_1 \cdot x_0$.

Пусть на входе приемника действует последовательность из 4-х импульсов. Наличие импульса отметим 1, а отсутствие 0. Заполним таблицу истинности для устройства определяющего наличие не менее 3-х импульсов в пачке (таблица 1.10).

Таблица 1.10

Номер набора	Значение аргументов				Значение функции $f(x_2, x_1, x_0)$
	x_3	x_2	x_1	x_0	
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	0
9	1	0	0	1	0
10	1	0	1	0	0
11	1	0	1	1	1
12	1	1	0	0	0
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1

Искомая СДНФ переключательной функции имеет вид:

$$y = \bar{x}_3 x_2 x_1 x_0 \vee x_3 \bar{x}_2 x_1 x_0 \vee x_3 x_2 \bar{x}_1 x_0 \vee x_3 x_2 x_1 \bar{x}_0 \vee x_3 x_2 x_1 x_0 .$$

Совершенная конъюнктивная нормальная форма ПФ получается конъюнкцией конституент нуля тех наборов, на которых переключательная функция равна нулю:

$$\text{СКНФ } f = \bigwedge_{j \in W_0} M(j).$$

СКНФ ПФ таблицы истинности $f(x_2, x_1, x_0)$(см. таблицу 1.8) в соответствии с определением равна:

$$f(x_2, x_1, x_0) = M(0) \,\&\, M(2) \,\&\, M(5) \,\&\, M(6) \,\&\, M(7) =$$
$$= (x_2 \vee x_1 \vee x_0) \,\&\, (x_2 \vee \bar{x}_1 \vee x_0) \,\&\, (\bar{x}_2 \vee x_1 \vee \bar{x}_0) \,\&$$
$$\&\, (\bar{x}_2 \vee \bar{x}_1 \vee x_0) \,\&\, (\bar{x}_2 \vee \bar{x}_1 \vee \bar{x}_0).$$

Очевидна справедливость следующих тождеств:

$$f = \bigvee_{j \in W_1} K(j) = \overline{\bigwedge_{j \in W_0} M(j)}; \quad \bar{f} = \bigvee_{j \in W_0} K(j) = \bigwedge_{j \in W_1} \overline{M(j)}.$$

Кроме канонических форм, в булевой алгебре большую роль играют более простые в смысле записи, а зачастую и в смысле схемной реализации функций формы, а именно:

Дизъюнктивная нормальная форма (ДНФ);

Конъюнктивная нормальная форма (КНФ).

ДНФ представляет собой дизъюнкцию (сумму) элементарных конъюнкций, необязательно являющихся конституентами единицы.

КНФ представляет собой конъюнкцию (произведение) элементарных дизъюнкций, необязательно являющихся конституентами нуля.

Следует помнить, что ДНФ и КНФ уже не являются формами однозначного представления функций. На множестве ДНФ и КНФ решают задачу поиска наиболее простых по сложности формул переключательной функции, эту задачу принято называть задачей минимизации функций.

1.5.2 Общие понятия о минимизации переключательных функций

Выражение переключательной функции в СДНФ или СКНФ является отправным при технической реализации логических схем цифровых устройств. Сложность логических схем (ЛС) определяется сложностью выражения ПФ, по которому ЛС реализуется. Поэтому раньше всего следует упростить ПФ или, как говорят, произвести минимизацию переключательных функций.

Сущность задачи минимизации состоит в поиске способа позволяющего по некоторой исходной формуле функции, например СДНФ, находить другую формулу, имеющую наименьшую сложность.

Сложность ПФ оценивается числом переменных, входящих в ее выражение (с учетом повторения).

Задача минимизации является исключительно сложной и в общем виде практически неразрешимой (неизбежен перебор вариантов), поэтому от поиска абсолютных минимальных форм функций приходится отказываться и ограничиваться нахождением минимальных форм лишь среди некоторого класса формул.

В булевой алгебре, в частности, ограничиваются минимизацией СДНФ и СКНФ, и нахождением минимальных по сложности ДНФ и КНФ, сокращенно обозначаемых через МДНФ и МКНФ соответственно.

Известны разные процедуры минимизации ПФ, но все они основаны на применении некоторых тождественных преобразований таких как:

операции полного и неполного склеивания;

операции поглощения;

операции удаления (или добавления) лишних членов и др.

Операция склеивания выражается тождествами:

$$A \cdot x \vee A \cdot \bar{x} = A; \quad (A \vee x) \cdot (A \vee \bar{x}) = A. \qquad (1.6)$$

В этих выражениях члены $A \cdot x$ и $A \cdot \bar{x}$ (так же, как $A \vee x$ и $A \vee \bar{x}$) отличаются только одной переменной, такие члены принято называть соседними. В отношении операции (1.6) говорят также, что члены $A \cdot x$ и $A \cdot \bar{x}$ ($A \vee x$ и $A \vee \bar{x}$) склеиваются по переменной x .

Рассмотрим две соседние конституанты единицы четырех переменных:

$$K_8 = x_3 \cdot \bar{x}_2 \cdot \bar{x}_1 \cdot \bar{x}_0 \quad \text{и} \quad K_9 = x_3 \cdot \bar{x}_2 \cdot \bar{x}_1 \cdot x_0 .$$

В результате их склеивания по переменной x_0 получаем

$$K_8 \vee K_9 = x_3 \cdot \bar{x}_2 \cdot \bar{x}_1 \cdot (\bar{x}_0 \vee x_0) = x_3 \cdot \bar{x}_2 \cdot \bar{x}_1 .$$

т.е. получаем новую конъюнкцию, ранг которой на единицу меньше ранга исходных конъюнкций. При этом переменная, по которой произведено склеивание, исключается, а результат склеивания содержит только переменные, входящие одинаковым образом (т.е. либо без инверсии, либо с инверсией) в исходные конституенты K_8 и K_9 . Это правило является универсальным и применимо к любым соседним логическим выражениям.

Операция неполного склеивания определяется соотношением

$$A \cdot x \vee A \cdot \bar{x} \vee A = A.$$

Операции поглощения выражаются тождественными соотношениями

$$A \vee A \cdot x = A; \quad A \vee A \cdot \bar{x} = A;$$

$$A \cdot (A \vee x) = A; \quad A \cdot (A \vee \bar{x}) = A.$$

Операция удаления или добавления лишних членов (т.е. таких членов, которые принимают значение, равное 1, одновременно с оставшимися членами выражения (ПФ) основана на тождестве:

$$A \vee A \vee ... \vee A = A.$$

Если в СДНФ некоторой ПФ произвести всевозможные операции склеивания и операции поглощения, получается сокращенная ДНФ. В отличие от СДНФ в нее, в общем случае, входят конъюнкции разного ранга: от первого до (n-1) - го ранга.

После отбрасывания лишних членов из сокращенной ДНФ получается ДНФ, называемая тупиковой. Ее особенностью является то, что она уже не допускает применения операций склеивание и поглощения.

Тупиковых ДНФ может быть несколько, так как при выполнении операций склеивания и поглощения СДНФ могут объединяться по-разному. Та из тупиковых ДНФ, которая имеет наименьшее число вхождений (наименьшее количество логических переменных и операций), называется минимальной ДНФ (МДНФ).

Приведем примеры, иллюстрирующие применение тождественных соотношений с целью минимизации ПФ.

Требуется минимизировать ПФ вида

$$y = f(x_2, x_1, x_0) = x_2 \cdot x_1 \cdot x_0 \vee \bar{x}_2 \cdot \bar{x}_1 \cdot x_0 \vee \bar{x}_2 \cdot x_1 \cdot x_0 \vee x_2 \cdot \bar{x}_1 \cdot x_0.$$

Производя склеивание первого и третьего членов по x_2 , а также второго и четвертого членов тоже по x_2, получим

$$y = x_1 \cdot x_0(x_2 \vee \bar{x}_2) \vee \bar{x}_1 \cdot x_0(\bar{x}_2 \vee x_2) = x_1 \cdot x_0 \vee \bar{x}_1 \cdot x_0.$$

К найденному выражению также можно применить операцию склеивания, но уже по x_1: $y = x_0(x_1 \vee \bar{x}_1) = x_0$.

Пример. Требуется упростить ПФ вида

$$y = f_2(x_2, x_1, x_0) = \bar{x}_1 \cdot x_0 \vee \bar{x}_2 \cdot x_1 \vee x_2 \cdot \bar{x}_0 \vee x_1 \cdot \bar{x}_0 \vee x_2 \cdot \bar{x}_1 \vee \bar{x}_2 \cdot x_0.$$

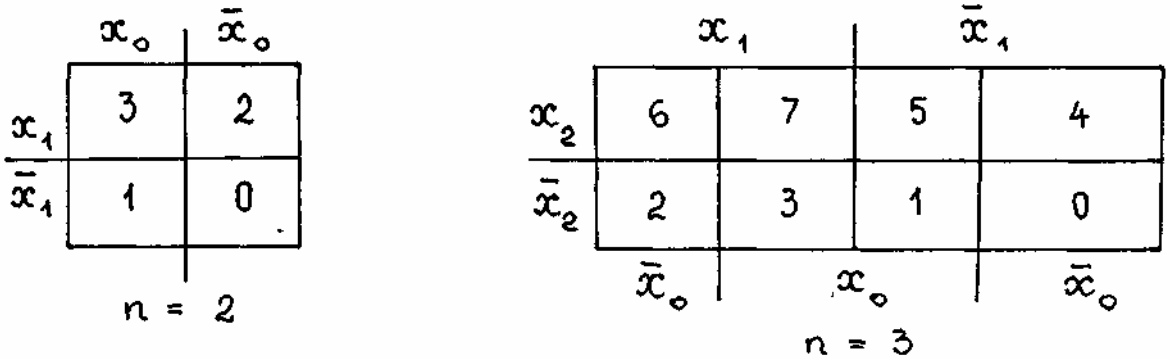
В данной ПФ нет членов, которые могли бы быть подвергнуты операциям склеивания и поглощения. Поэтому остается проверить, не являются ли лишними некоторые члены. Рассмотрим, например, член $\bar{x}_2 \cdot x_0$. Он принимает значение, равное 1, только тогда, когда x_0 = 1 и x_2 = 0. Найдем значение остальной части функции при таких значениях переменных x_2 и x_0: $\bar{x}_1 \cdot 1 \vee \bar{0} \cdot x_1 \vee 0 \cdot \bar{1} \vee x_1 \cdot \bar{1} \vee 0 \cdot \bar{x}_1 = \bar{x}_1 \vee x_1 = 1.$

Из этого следует, что член $\bar{x}_2 \cdot x_0$ является лишним, так как, отбросив его, мы не изменим значения функции. Аналогичным путем можно показать, что члены $x_2 \cdot \bar{x}_1$ и $x_1 \cdot \bar{x}_0$ также являются лишними. Следовательно, ПФ можно записать в виде

$$y = f_2(x_2, x_1, x_0) = \bar{x}_1 \cdot x_0 \vee \bar{x}_2 \cdot x_1 \vee x_2 \cdot \bar{x}_0.$$

1.5.3 Минимизация переключательных функций посредством диаграммы Вейча

Диаграмма Вейча (ДВ) для ПФ n переменных x_{n-1} ,..., x_0 представляет собой прямоугольник, разделенный на клетки, число которых равно числу $L_x = 2^n$ всевозможных наборов переменных. При n = 2, 3, 4 разбиение осуществляется так, что для каждой переменной половина площади большого прямоугольника соответствует x_i а вторая половина $-\bar{x}_i$. Способ разбиения показан на рисунке 1.4 (n = 2 и 3) и на рисунке 1.5 (n = 4).



а) Рисунок 1.4 б)

В случае n = 5 ДВ образуется путем состыковывания двух диаграмм, соответствующих n = 4, причем одна из них ставится в соответствие переменной x_4, а другая $-\bar{x}_4$. Аналогично строится ДВ для $n \geq 6$. Однако при $n > 5$ диаграмма Вейча становится громоздкой и неудобной для пользования.

Каждая клетка ДВ соответствует некоторому набору переменных с номером α. Рассмотрим, например, клетку, которая лежит на пересечении областей диаграммы, соответствующих переменным $x_3, \bar{x}_2, x_1, \bar{x}_0$.

Образуем из этих переменных конъюнкцию $x_3 \cdot \bar{x}_2 \cdot x_1 \cdot \bar{x}_0$. Она представляет собой конституенту единицы от четырех переменных для набора с номером $\alpha = 1010_{(2)} = 10_{(10)}$. Поэтому присвоим данной клетке номер α = 10. Аналогично нумеруются остальные клетки (см. рисунок 1.5). Переменные x_i можно располагать по сторонам ДВ не обязательно в порядке, указанном на рисунке 1.5, но при этом получится другая нумерация клеток.

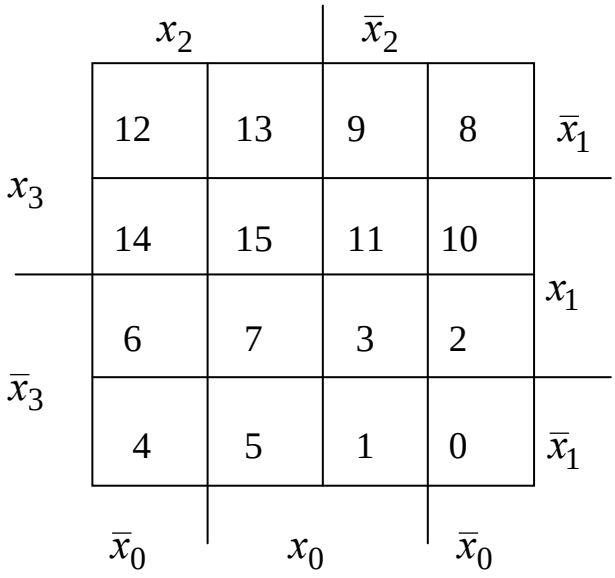


Рисунок 1.5

С помощью ДВ задаем ПФ. Пусть, например, ПФ $y = f(x_3, x_2, x_1, x_0)$ равна 1 на наборах переменных с номерами $\alpha =$ 0, 2, 4, 5, 6, 7, 8, 10, 11, 12, 13, 14 и равна нулю на всех остальных наборах. Аналитическое выражение такой ПФ в СДНФ получается весьма громоздким:

$$y = K_0 \vee K_2 \vee K_4 \vee K_5 \vee K_6 \vee K_7 \vee K_8 \vee K_{10} \vee K_{11} \vee K_{12} \vee K_{13} \vee K_{14} =$$
$$= \bar{x}_3 \cdot \bar{x}_2 \cdot \bar{x}_1 \cdot \bar{x}_0 \vee \bar{x}_3 \cdot \bar{x}_2 \cdot x_1 \cdot \bar{x}_0 \vee \bar{x}_3 \cdot x_2 \cdot \bar{x}_1 \cdot \bar{x}_0 \vee \bar{x}_3 \cdot x_2 \cdot \bar{x}_1 \cdot x_0 \vee$$
$$\vee \bar{x}_3 \cdot x_2 \cdot x_1 \cdot \bar{x}_0 \vee \bar{x}_3 \cdot x_2 \cdot x_1 \cdot x_0 \vee x_3 \cdot \bar{x}_2 \cdot \bar{x}_1 \cdot \bar{x}_0 \vee x_3 \cdot \bar{x}_2 \cdot x_1 \cdot \bar{x}_0 \vee \quad (1.7)$$
$$\vee x_3 \cdot \bar{x}_2 \cdot x_1 \cdot x_0 \vee x_3 \cdot x_2 \cdot \bar{x}_1 \cdot \bar{x}_0 \vee x_3 \cdot x_2 \cdot \bar{x}_1 \cdot x_0 \vee x_3 \cdot x_2 \cdot x_1 \cdot \bar{x}_0.$$

Ранее было показано, что при склеивании двух соседних конституент единицы ранга *n* образуется конъюнкция ранга *n* - 1, в которой содержатся только переменные, входящие одинаковым образом в исходные конституенты единицы. Применим это правило к конституентам $К_{12}$, $К_{13}$ и $К_4$, $К_5$ (рисунок 1.6), в результате получим:

$$P_1 = K_{12} \vee K_{13} = x_3 \cdot x_2 \cdot \bar{x}_1; \quad P_2 = K_4 \vee K_5 = \bar{x}_3 \cdot x_2 \cdot \bar{x}_1.$$

Соседние конъюнкции P_1 и P_2 склеиваются по переменной x_3 , при этом получается конъюнкция

$$Z_1 = P_1 \vee P_2 = x_2 \cdot \bar{x}_1.$$

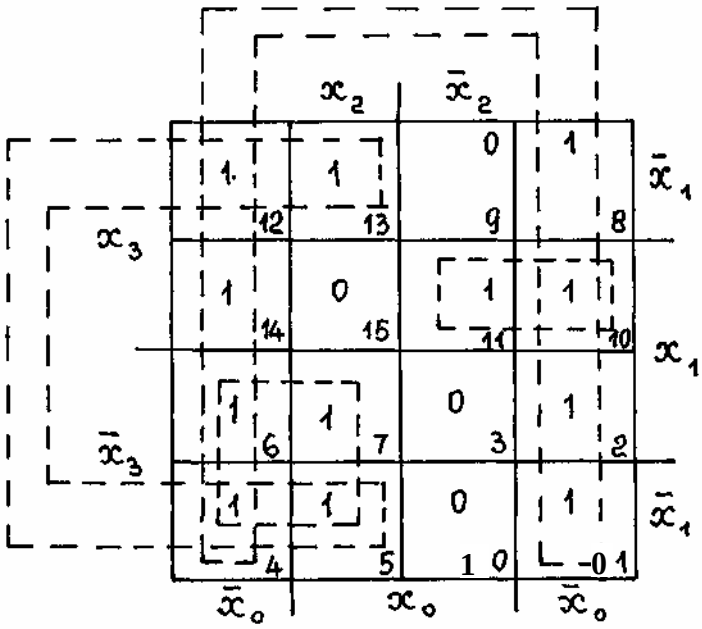


Рисунок 1.6

Таким образом, при склеивании четырех соседних конституент единицы ранга $n = 4$ образуется конъюнкция ранга $n - 2 = 4 - 2 = 2$, содержащая только переменные, одинаковым образом входящие в выражения четырех соседних конституент единицы, т.е. в $К_4$, $К_5$, $К_{12}$ и $К_{13}$. Данное правило является универсальным. Применим его к группе четырех соседних конституент единицы, включающей $К_0$, $К_2$, $К_8$ и $К_{10}$, и к группе четырех соседних конституент единицы, включающей $К_4$, $К_6$, $К_{12}$, $К_{14}$.

Производя склеивание в той или другой группе, получим

$$Z_2 = \bar{x}_2 \cdot \bar{x}_0; \quad Z_3 = x_2 \cdot \bar{x}_0.$$

Конъюнкции Z_2 и Z_3 также являются соседними, так как они различаются одной переменной. Поэтому они склеиваются и дают конъюнкцию $W = \bar{x}_0$. Таким образом, при склеивании восьми соседних конституент единицы ранга $n = 4$ образуется конъюнкция ранга $n - 3 = 4 - 3 = 1$, содержащая только те переменные, которые одинаковым образом входят во все восемь соседних конституент единицы.

Обобщая полученные результаты, можно сформулировать процедуру минимизации ПФ посредством диаграммы Вейча.

Определяются всевозможные группы соседних конъюнкций, охватываемые прямоугольными контурами и содержащие по 1,2,4,8,16 и т.д. конституент единицы (см. рисунок 1.6), при этом одна и та же конституента может входить в несколько разных групп. При $n \leq 4$ соседними следует считать не только конституенты, находящиеся в

рядом расположенных клетках, но и те, которые находятся в крайних клетках строк и столбцов. При $n = 5$ правила определения соседних конституент единицы в каждой из половин ДВ таковы же, как при $n = 4$. Однако при определении соседних конституент единицы, расположенных в левой и правой частях ДВ, следует учесть, что соседними являются конституенты, находящиеся в столбцах с одинаковым номером, если его отсчитывать от левого края каждой половины.

Следует стремиться к образованию возможно меньшего количества групп соседних конъюнкций (конституент единицы) с возможно большим числом, конституент единицы в группе.

Для каждой группы соседних конституент единицы пишутся конъюнкции, образуемые в результате склеивания, они образуются только на тех переменных, которые одинаковым образом (с инверсией или без инверсии) входят во все исходные конституенты единицы в каждой группе.

Находятся дизъюнкции конъюнкций, полученных при склеивании конституент единицы в каждой группе.

Применяя указанные правила, найдем МДНФ для ПФ, заданной на рисунке 1.6: $y = \bar{x}_0 \vee x_2 \cdot \bar{x}_1 \vee \bar{x}_3 \cdot x_2 \vee x_3 \cdot \bar{x}_2 \cdot x_1$.

Это выражение значительно проще выражения (1.7).

Группы соседних конституент единицы можно образовать поразному. Поэтому в итоге описанной выше процедуры могут получаться различные выражения ПФ, являющиеся тупиковыми ДНФ. Одна из таких ДНФ может содержать наименьшее число вхождений переменных, именно эта тупиковая ДНФ и является минимальной.

1.5.4 Минимизация неполностью определенных переключательных функций и представленных в форме СКНФ

Под неполностью определенными ПФ понимаются такие, которые описывают работу логических схем (ЛС) при условии, что на их входах некоторые наборы переменных никогда не появляются и функции на них не определены. Поэтому при заполнении ДВ такой ПФ некоторые клетки остаются пустыми. В процессе минимизации таких ПФ целесообразно доопределить их на отсутствующих наборах переменных, расставив в пустых клетках 1 и 0 так, чтобы образовались группы соседних конституент единицы с максимальным числом единиц, а количество таких групп было наименьшим.

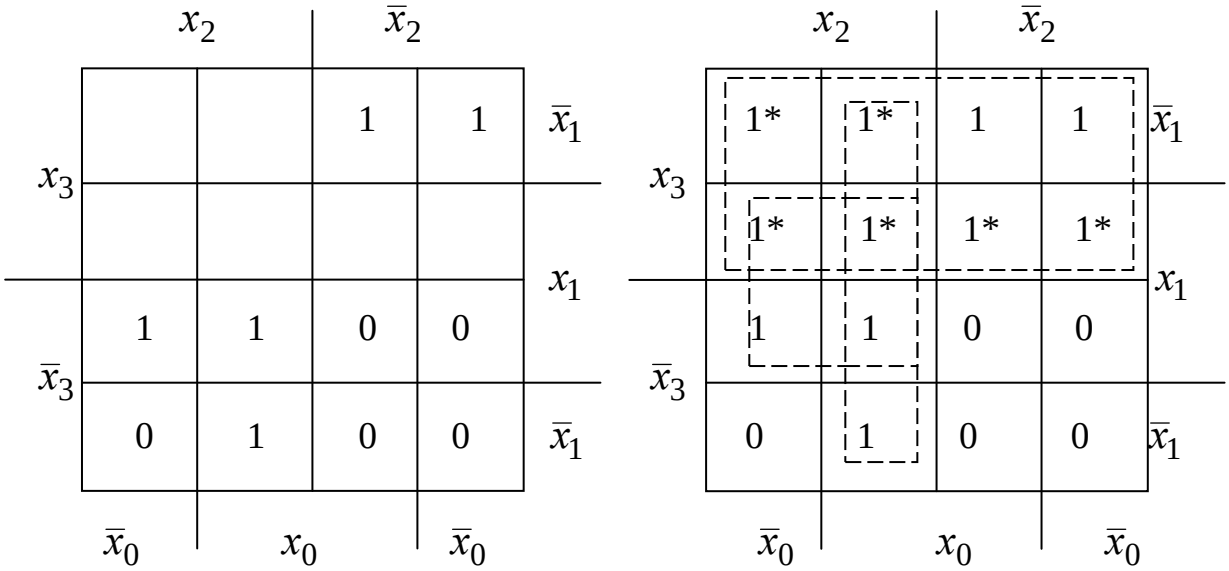
В качестве примера рассмотрим синтез ЛС, предназначенной для выделения пяти старших цифр десятичной системы счисления, представленных четырехразрядными двоичными кодами: $A = x_3x_2x_1x_0$. В данном случае номер набора A совпадает с числом, выражаемым данным кодом. Поскольку число различных наборов переменных равно 2^4=16, а цифр десятичной системы всего десять, то на наборах с номерами 10, 11, 12, 13, 14, 15 ПФ оказывается неопределенной. В соответствии с заданием на наборах с номерами 0, 1, 2, 3, 4 (пять младших цифр) ПФ должна быть равна 0, а на наборах 5, 6, 7, 8, 9 (пять старших цифр) она должна быть равна 1.

На рисунке 1.7, *а* приведена ДВ заданной ПФ, а на рисунке 1.7,*б* указан способ доопределения ПФ, приводящий к минимальной ДНФ: $y = x_3 \vee x_2x_0 \vee x_2x_1$.

Для сравнения, минимизированная ПФ без доопределения неопределенных наборов имеет вид:

$$y = x_3\overline{x}_2\overline{x}_1 \vee \overline{x}_3x_2x_0 \vee \overline{x}_3x_2x_1.$$

Сложность этого выражения значительно выше, хотя и в этом случае удалось исключить из каждой конъюнкции по одной переменной, а число слагаемых уменьшить с пяти до трех.



а) *б)*

Рисунок 1.7

Кроме того, вместо групп соседних конституент единицы следует образовывать группы соседних конституент нуля. В качестве примера на рисунке 1.8 показана ДВ для ПФ, заданной на рисунке 1.7,б. По этой диаграмме находим:

$$\overline{y} = (x_3 \vee x_2) \wedge (x_3 \vee x_1 \vee x_0).$$

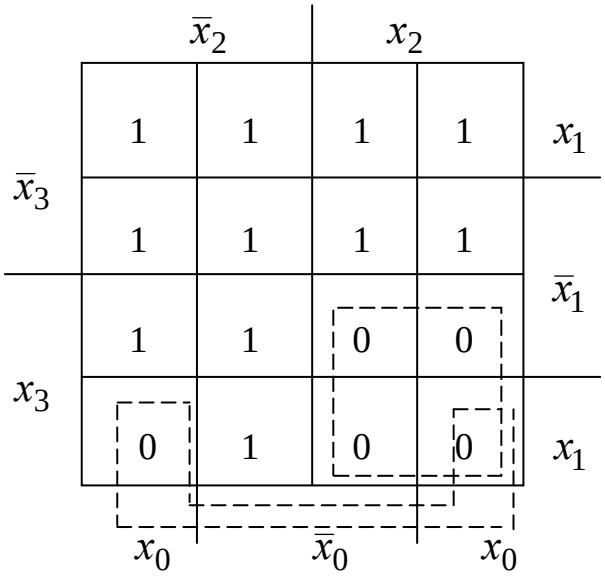


Рисунок 1.8

Задания для самостоятельной работы

Выполнить минимизацию переключательной функции методом тождественных преобразований и методом диаграмм Вейча

Номер набора	X_3	X_2	X_1	X_0	Значения выходного сигнала Y по вариантам					
					Задание 1	Задание 2	Задание 3	Задани	Задание 5	Задани
0	0	0	0	0	0	0	0	e04	0	e06
1	0	0	0	1	1	0	0	0	0	0
2	0	0	1	0	1	1	0	1	0	1
3	0	0	1	1	1	1	1	1	0	0
4	0	1	0	0	0	1	1	1	0	0
5	0	1	0	1	0	0	0	0	1	1
6	0	1	1	0	0	0	0	1	1	1
7	0	1	1	1	0	0	0	0	1	1
8	1	0	0	0	1	0	0	0	0	0
9	1	0	0	1	1	0	0	0	0	0
10	1	0	1	0	0	1	0	0	0	1
11	1	0	1	1	1	1	1	0	0	1
12	1	1	0	0	1	1	1	1	1	0
13	1	1	0	1	0	0	1	1	0	0
14	1	1	1	0	0	0	0	1	1	0
15	1	1	1	1	0	0	0	0	1	0

Глава 2. СХЕМОТЕХНИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

2.1 Синтез цифровых автоматов

Методы описания и проектирования столь сложных объектов, какими являются ЭВМ, базируются на основополагающих принципах, сформулированных в общей теории систем. Поэтому, прежде чем приступить к рассмотрению способов построения и функционирования БВМ, раскроем смысл основных понятий и принципов, которые относятся к общей теории систем и широко используются для бортовых вычислительных машин.

Система — это совокупность элементов, объединенных в одно целое для достижения определенной цели. При этом под целью понимается множество результатов, определяемых назначением системы. Вычислительная система в смысле указанного определения является системой, предназначенной для автоматизации вычислений на основе алгоритмов. Следует отметить, что понятие система приложимо как к ЭВМ в целом, так и к отдельным частям ЭВМ, например к устройствам БВМ.

Чтобы описать систему, необходимо определить функции системы и структуру системы.

Функция системы — правило получения результатов, определяемых назначением системы. Иначе говоря, функция системы — это описание процессов, которые имеют место в системе. Функции систем стремятся описывать в математической форме ввиду компактности и четкости, хотя может быть использована и другая форма, например, словесная или табличная. Функции вычислительных систем чаще всего описываются в форме алгоритмов.

Структура системы — фиксированная совокупность элементов и связей между ними. Структура наглядно изображает, как устроена система — из каких частей она состоит и как эти части связаны друг с другом. Математической, а в этом смысле наиболее абстрактной и универсальной, формой изображения структуры является граф. Граф есть совокупность вершин и дуг (ребер), представляющих одно- или двунаправленные связи между вершинами. Вершины графа отождествляются с элементами (минимальными частями) системы, а дуги и ребра графа — со связями между соответствующими элементами.

Инженерная форма отображения структуры — схема. Схема и граф тождественны по своему содержанию и различаются лишь по форме. В схемах для обозначения элементов используются различные геометрические фигуры — условные графические обозначения, разнообразие форм которых облегчает чтение схем.

Система описана, если заданы ее функция и структура. Функция определяет порядок процессов в системе, а структура — состав и взаимосвязь частей (элементов, из которых состоит система). Системам присуще следующее качество. Свойства совокупности элементов, объединенных в одну систему, не являются простой суммой свойств элементов, а имеют новое качество, отсутствующее в элементах. Например, в совокупности электронных элементов (транзисторов, резисторов и т. и.), определенным образом соединенных между собой, появляется эффект, который отождествляется с операциями математической логики, т. е. совокупность электрических элементов превращается в систему, функции которой описываются не законами электротехники, а законами математической логики. Такого рода система из электронных элементов приобретает новое свойство и становится логическим элементом. В свою очередь, объединение логических элементов, каждый из которых реализует логическую операцию, приводит к схеме, которая обладает свойством складывать числа. Принцип (способ), по которому объединение элементов приводит к появлению новых свойств, отличных от свойств элементов, называется принципом организации. Другими словами, организация - способ объединения элементов с целью осуществления определенных функций в системах, состоящих из большого числа элементов.

Организация — понятие более высокого ранга, чем функция и структура. Конкретный принцип организации — это способ построения различных систем, обладающих одинаковыми свойствами, т. е. один принцип организации, применяемый к различным случаям, приводит к системам с различными функциями и структурами. Так что функция и структура — это конкретизация принципа организации, всего лишь один вариант организации

В свою очередь, различные принципы организации приводят к системам, различающимся своими функциями и структурами, и тождественным по своим свойствам, своему назначению. Когда речь идет о принципе порождения функций, необходимых и достаточных для обеспечения определенных свойств систем пользуются термином

функциональная организация. Функциональная организация — это принципы построения абстрактных систем, заданных своими функциями. Об абстрактной системе известно только ее назначение и не известно, как она устроена, из каких элементов состоит, т. е. абстрактная система — это лишь описание, существующее на бумаге. Когда речь идет о принципе порождения структур, необходимых и достаточных для реализации заданных функций, используется термин структурная организация. Структурная организация — это принципы перевода абстрактных систем, заданных в виде функций, в материальные системы, состоящие из физически существующих элементов. Элементы бортовых цифровых вычислительных систем называют также цифровыми автоматами (ЦА).

2.1.1 Основные понятия из теории цифровых автоматов

В общем (абстрактном) виде цифровой автомат представляется как устройство с одним входом и одним выходом (рисунок 2.1), предназначенное для преобразования входной информации, отображаемой сигналами *X*, в выходную информацию, отображаемую сигналами *Y*.

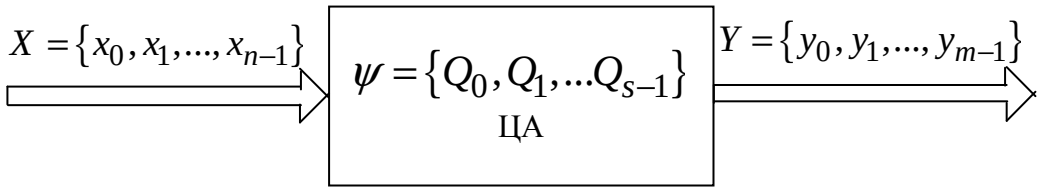


Рисунок 2.1

Основным качеством, выделяющим ЦА с памятью из других преобразователей информации, является наличие в ЦА дискретного множества внутренних состояний ψ. Это свойство ЦА определяется использованием в нем элементов памяти. Будем рассматривать только ЦА с конечным числом L_ψ внутренних состояний (конечные ЦА). Каждому состоянию присвоен определенный номер $\beta(\psi)$, где $\beta = 0,1,2,...,L_\psi - 1$.

В теории автоматов принят алфавитный способ задания информации. Алфавитом называется конечная совокупность символов, называемых словами. Число букв в слове определяет длину слова.

Входной сигнал X может принимать одно из L_x различных значений, каждое из этих значений принимается за букву. Совокупность всех значений составляет входной алфавит ЦА. Выходные сигналы могут принимать одно из L_y различных значений, составляющих выходной алфавит ЦА.

Алфавит внутренних состояний содержит L_ψ букв. Буквы входного, выходного алфавита и алфавита состояний могут принимать только два значения: 0 и 1.

Поэтому $L_x = 2^n;\ L_y = 2^m;\ L_\psi = 2^s$,

где n - количество разрядов входного сигнала,

m - количество разрядов выходного сигнала,

s - количество элементов памяти.

В общем случае поведение ЦА (см. рисунок 2.1) в момент времени t описывается зависимостью выходного сигнала $Y(t)$ от значений входных сигналов $X(t), X(t-1), X(t-2)...$, т.е. в данный и предыдущие моменты времени t, $(t-1), (t-2)$ и т.д.

При этом возможны два случая:

1) значение выходного сигнала $Y(t)$ в момент времени однозначно определяется сигналами на входе $X(t)$ в тот же момент времени. В этом случае техническая реализация ЦА соответствует схеме без элементов памяти, которая называется комбинационной схемой (КС). В качестве математического аппарата анализа и синтеза КС используется теория логических (переключательных) функций;

2) значение выходного канала $Y(t)$ в момент времени t определяется значениями сигнала на входе $X(t), X(t-1), X(t-2)$ и т.д.

Техническая реализация ЦА для этого случая соответствует схеме с элементами памяти.

Рассмотрим некоторые особенности функционирования конечных ЦА. Закон преобразования информации в конечном ЦА описывается функцией переходов $\psi(t+1)$ и функцией выходов $Y(t)$.

В зависимости от характера этих функций различают ЦА двух родов, которые нашли наибольшее применение в цифровых устройствах.

ЦА первого рода (автоматы Мили). Функция переходов

$$\psi(t+1) = F\left[\psi(t), X(t)\right]$$

определяет внутреннее состояние ЦА в момент $(t+1)$ в зависимости от входного сигнала $X(t)$ и внутреннего состояния $\psi(t)$. Функция выходов

$$Y(t) = f\left[\psi(t), X(t)\right]$$

определяет выходной сигнал в момент времени t при условии, что ЦА находится в состоянии $\psi(t)$ и подвергается воздействию входного сигнала $X(t)$.

ЦА второго рода (автоматы Мура). Функция выходов

$$Y(t) = f\left[\psi(t)\right]$$

описывает выходной сигнал в момент времени t, который определяется только внутренним состоянием ЦА $\psi(t)$ и не зависит от входного воздействия.

Функции переходов ЦА второго и первого родов совпадают.

Различают три способа описания функций $\psi(t+1)$ и $Y(t)$: аналитический, графический, табличный.

Аналитический способ является наиболее универсальным, но не очень наглядным. Наибольшей наглядностью обладают другие два способа. Графически автомат задают ориентированным графом, вершины которого отождествляют с состояниями автомата, а ребра (дуги) – с переходами из одного состояния в другое. Ребра графа автомата Мили отмечают входными и выходными сигналами (буквами), соответствующими данному переходу. В графе автомата Мура ребра отмечают входными сигналами, а вершины – состояниями и соответствующим им выходными сигналами.

На рисунке 2.2 приведен граф автомата Мура. Графическое представление имеет наибольшую наглядность, но с увеличением числа состояний автомата наглядность теряется. Кроме того, автоматизированные методы синтеза ЦА представленных графами в настоящее время не нашли широкого распространения в виду слабой формализации алгоритмов.

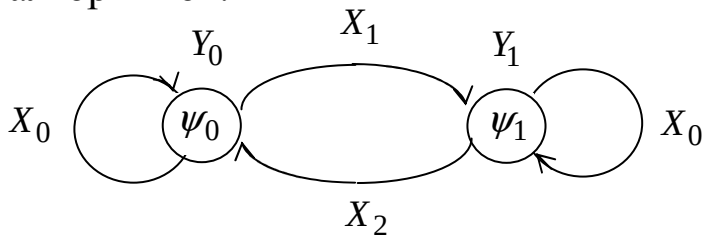


Рисунок 2.2

В дальнейшем будет использоваться табличный способ описания функций $\psi(t+1)$ и $Y(t)$ с помощью таблиц переходов и выходов, соответственно, таблицы 2.1 и 2.2.

Таблица 2.1- Таблица переходов ЦА

$X(t)$	$\psi(t)$					
	ψ_0	ψ_1	ψ_2	…	$\psi_{L_{\psi-2}}$	$\psi_{L_{\psi-1}}$
X_0	ψ_2	ψ_5	ψ_4	…	$\psi_{L_{\psi-3}}$	ψ_6
X_1	$\psi_{L_{X-1}}$	ψ_1	ψ_5	…	ψ_3	ψ_2
…	…	…	…	…	…	…
$X_{L_{X-2}}$	ψ_4	$\psi_{L_{X-1}}$	ψ_0	…	ψ_0	ψ_6
$X_{L_{X-1}}$	ψ_1	ψ_0	ψ_2	…	ψ_4	ψ_8

Таблица 2.2 – Таблица выходов ЦА

$X(t)$	$\psi(t)$						
	ψ_0	ψ_1	ψ_2	…	$\psi_{L_{\psi-3}}$	$\psi_{L_{\psi-2}}$	$\psi_{L_{\psi-1}}$
X_0	Y_3	Y_1	$Y_{L_{Y-4}}$	…	Y_0	Y_6	Y_7
X	Y_3	Y_0	Y_4	…	Y_1	$Y_{L_{\psi-1}}$	Y_5
…	…	…	…	…	…	…	…
$X_{L_{X-2}}$	Y_0	Y_6	Y_0	…	Y_5	Y_4	Y_1
$X_{L_{X-1}}$	Y_1	Y_7	Y_8	…	Y_7	Y_7	Y_0

При построении таблицы переходов (см. таблицу 2.1) в левом крайнем столбце записываются все входные сигналы из входного алфавита $\left(X_0,...X_{L_{X-1}}\right)$, а в верхней строке – все возможные состояния ЦА $\left(\psi_0,...,\psi_{L_{\psi-1}}\right)$. Если в момент t ЦА находится в состоянии $\psi(t)=\psi_\beta\left(0\le\beta\le L_{\psi-1}\right)$ и на него действует входной сигнал $X(t)$, то в момент $t+1$ ЦА перейдет в состояние $\psi(t+1)$, являющееся элементом алфавита состояний. Новое состояние $\psi(t+1)$ записывается в клетку таблицы, расположенную на пересечении столбца и строки, соответствующих $X(t)$ и $\psi(t)$.

Цифровой автомат обладает полной системой переходов, если для любой пары его состояний $\psi_{\beta 1}$ и $\psi_{\beta 2}\left(0 \le \beta_1, \beta_2 \ge L_\psi - 1\right)$ найдется хотя бы один входной сигнал X_α, который переводит ЦА из состояния $\psi_{\beta 1}$ в состояние $\psi_{\beta 2}$. При выполнении этого условия в каждом столбце таблицы переходов содержатся символы всех состояний ЦА.

Таблица выходов (см. таблицу 2.2) строится аналогично таблице переходов, при этом в клетку, лежащую на пересечении столбца и строки, соответствующих ψ_β и X_α, записывается выходной сигнал $Y(t) = f\left[\psi_\beta(t), X_\alpha(t)\right]$.

Цифровой автомат обладает полной системой выходов, если в каждом состоянии он формирует сигнал, отличающийся от выходного сигнала, соответствующего любому другому состоянию.

По содержимому таблиц переходов и выходов могут быть получены аналитические выражения логических функций переходов и выходов, соответственно.

2.1.2 Этапы синтеза схем автоматов

При синтезе схем автоматов выделяют этапы абстрактного синтеза, структурного синтеза, технического проектирования. Целью абстрактного синтеза является задание автомата графом или таблицей переходов и минимизация числа состояний автомата. Задачей структурного синтеза является построение структурной схемы автомата из заданного набора комбинационных и запоминающих элементов.

Этапы абстрактного и структурного синтеза объединяют в один этап – этап логического проектирования, в результате которого схему автомата можно представить состоящей из двух частей – комбинационной схемы и памяти в виде набора запоминающих элементов (ЗЭ).

В этой схеме можно выделить входные двоичные сигналы $X = \{x_0, x_1, ..., x_{n-1}\}$, выходные двоичные сигналы $Y = \{y_0, y_1, ..., y_{m-1}\}$, двоичные сигналы состояния памяти $Q_0, ..., Q_{s-1}$, двоичные сигналы возбуждения памяти, переключающие состояния элементарных автоматов, $g_0, ..., g_{s-1}$ (рисунок 2.3).

На этапе технического проектирования разрабатываются функциональные, принципиальные и монтажные схемы, составляется тех-

ническая документация, выполняется электрический расчет, проверяется работоспособность автомата и т.д.

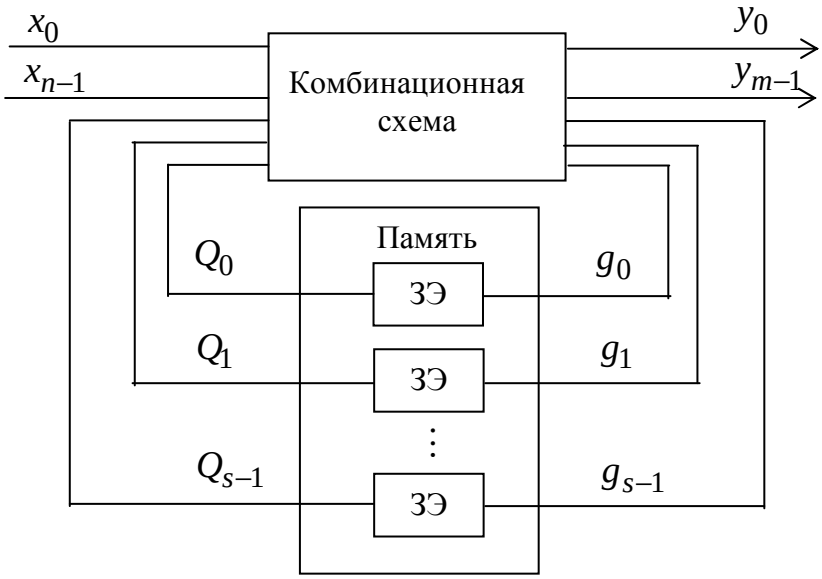


Рисунок 2.3

Для разработки функциональной схемы необходимо:

определить число запоминающих элементов и их тип;

представить переключательную функцию комбинационной схемы в форме СДНФ или СКНФ;

выполнить минимизацию ПФ;

преобразовать ПФ к виду удобному для реализации на заданном базисе логических элементов;

выполнить техническую реализацию полученной ПФ.

Комбинационная схема может быть разделена на две части. Первая по логическим функциям переходов производит изменение состояния автомата, а вторая по функциям выходов обеспечивает формирование выходных сигналов.

Синтез комбинационных схем возбуждения памяти автомата и формирования выходных сигналов (или одной обобщенной комбинационной схемы см. рисунок 2.3) осуществляется в несколько этапов.

На первом этапе осуществляется словесное или табличное задание свойств логической схемы. Для обеспечения однозначности выполняемых операций при словесном задании устройства необходимо выполнить переход к табличной форме.

На втором этапе получают совершенную дизъюнктивную или конъюнктивную нормальную форму переключательной функции.

На третьем этапе выполняется минимизация переключательной функции. Цель минимизации - упростить выражение ПФ для уменьшения числа элементов, реализующих ЦУ. Минимизация осуществляется одним из возможных методов, чаще всего методом тождественных преобразований или методом диаграмм Вейча. Поскольку техническая реализация, полученная переключательной функцией, возможна разными логическими элементами, то на четвертом этапе выполняется преобразование полученной ПФ к виду, удобному для реализации на заданном (выбранном) базисе логических элементов.

И на заключительном пятом этапе логические операции ПФ заменяются операционными элементами, реализующими эти функции.

Применение разных базисов логических элементов обусловлено понятием функциональной полноты различных базисов логических функций.

2.1.3 Функционально полные системы логических функций

Как было указано ранее, любую ПФ можно представить в виде СДНФ (СКНФ), содержащей в общем случае переменные (x_1, x_2,…, x_m) и только три логические функции - инверсию (НЕ), конъюнкцию (И) и дизъюнкцию (ИЛИ).

Набор элементарных логических функций, суперпозициями которых может быть записана любая ПФ, называют функционально полным.

Функциональной полнотой обладают следующие системы логических функций:

И, ИЛИ, НЕ (функции алгебры Буля);

И-НЕ (функция Шеффера);

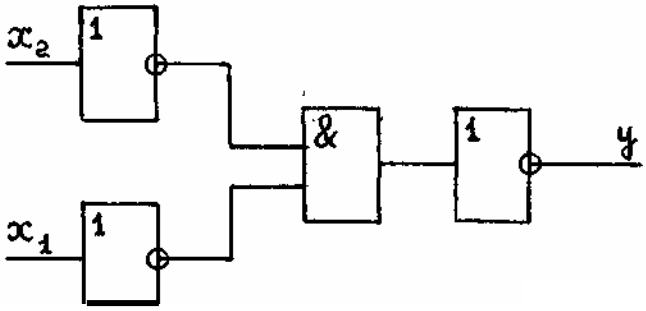
ИЛИ-НЕ (функция Пирса);

И, сумма по модулю 2 (функции алгебры Жегалкина).

Наиболее разработанными и удобными являются функции булевой алгебры, в то же время эта система обладает избыточной полнотой, т.е. для реализации любой ПФ достаточно взять только две из указанных функций: либо "И", "НЕ", либо "ИЛИ", "НЕ".

Пример:

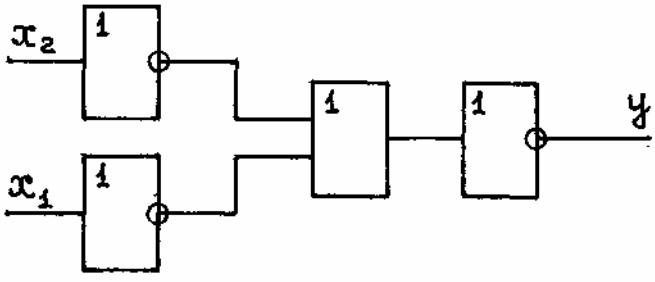
1) С помощью функции "И", "НЕ" реализуется функция "ИЛИ" (рисунок 2.4).



$$y = x_2 \vee x_1 = \overline{\overline{x_2 \vee x_1}} = \overline{\bar{x}_2 \wedge \bar{x}_1}$$

Рисунок 2.4

2) С помощью функции "ИЛИ", "НЕ" реализуется функция "И" (рисунок 2.5).



$$y = x_2 \wedge x_1 = \overline{\overline{x_2 \wedge x_1}} = \overline{\bar{x}_2 \vee \bar{x}_1}$$

Рисунок 2.5

Как следствием этого, является утверждение о том, что функциональной полнотой обладают отдельно взятые функция Шеффера (И-НЕ) и функция Пирса (ИЛИ-НЕ).

Реализация функций "НЕ", "И", "ИЛИ" посредством функции "И-НЕ" представлена на рисунках 2.6, 2.7, 2.8.

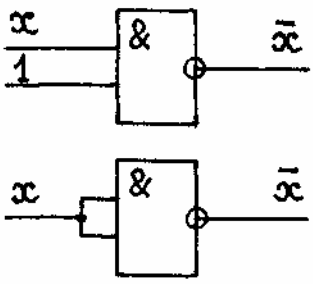
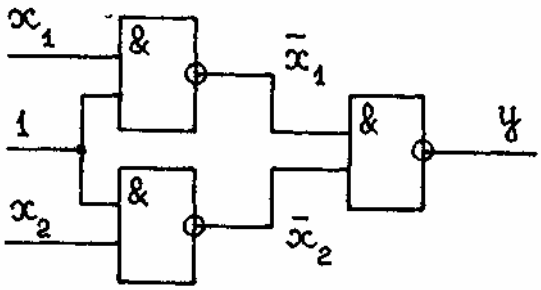
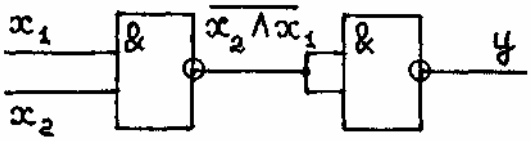


Рисунок 2.6 - Функция "НЕ" на базисе логических функций Шеффера



$$y = x_2 \vee x_1 = \overline{x_2 \vee x_1} = \overline{\bar{x}_2 \wedge \bar{x}_1}$$

Рисунок 2.7-Функция "ИЛИ" на базисе логических функций Шеффера



$$y = x_2 \wedge x_1 = \overline{x_2 \wedge x_1}$$

Рисунок 2.8 - Функция "И" на базисе логических функций Шеффера

Реализация функций "НЕ", "И", "ИЛИ" посредством функции "ИЛИ-НЕ" представлена на рисунках 2.9, 2.10, 2.11.

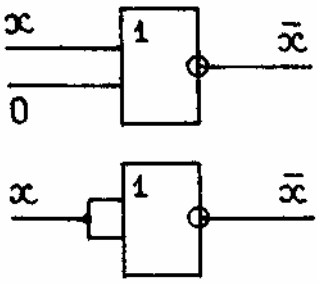
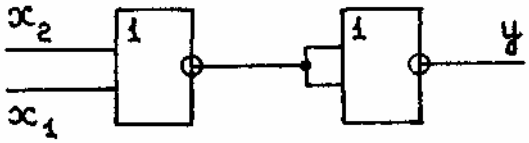
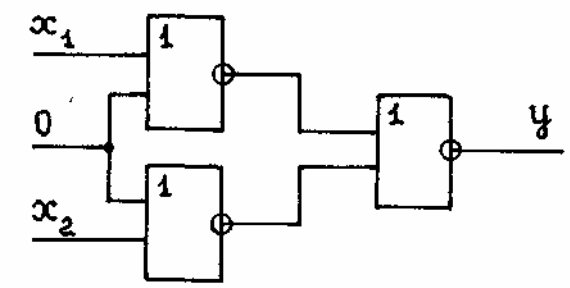


Рисунок 2.9 - Функция "НЕ" на базисе логических функций Пирса



$$y = x_2 \vee x_1 = \overline{x_2 \vee x_1}$$

Рисунок 2.10 - Функция "ИЛИ" на базисе логических функций Пирса



$$y = x_2 \wedge x_1 = \overline{\overline{x_2 \wedge x_1}} = \overline{\bar{x}_2 \vee \bar{x}_1}$$

Рисунок 2.11 - Функция "И" на базисе логических функций Пирса

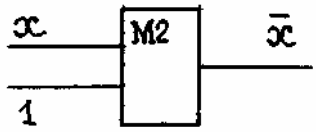
Реализация некоторой ПФ набором функций алгебры Жегалкина заключается в выполнении следующих этапов:

в СДНФ функции знаков дизъюнкции заменяются знаками сумма по модулю два;

в конституентах единицы функции всех инверсий аргументов заменяются согласно тождества $\bar{x} = x \oplus 1$;

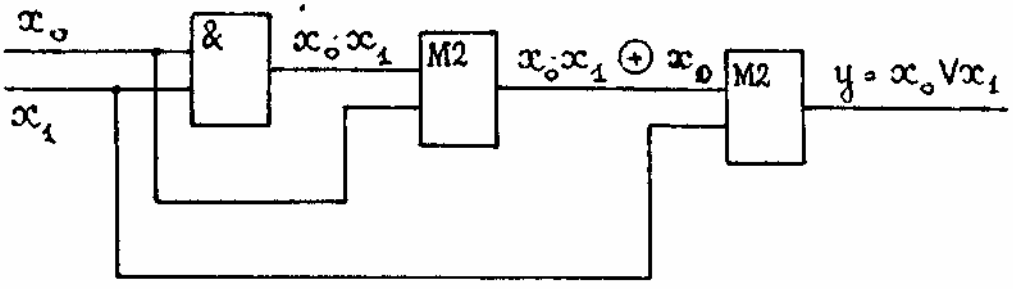
раскрываются скобки и упрощается полученное выражение согласно тождества $x \oplus x = 0$.

В соответствии с перечисленными этапами реализованные посредством функций алгебры Жегалкина функции "НЕ" и "ИЛИ" представлены на рисунках 2.12, 2.13.



$$y = x \oplus 1 = x \cdot \bar{1} \vee \bar{x} \cdot 1 = x \cdot 0 \vee \bar{x} \cdot 1 = \bar{x}$$

Рисунок 2.12 - Функция "НЕ" на базисе функций Жегалкина



$$y = x_0 \vee x_1 = x_0 \cdot (x_1 \vee \bar{x}_1) \vee x_1 \cdot (x_0 \vee \bar{x}_0) = x_0 x_1 \vee x_0 \bar{x}_1 \vee x_1 x_0 \vee x_1 \bar{x}_0 =$$
$$= x_0 x_1 \oplus x_0 \bar{x}_1 \oplus x_1 \bar{x}_0 = x_0 x_1 \oplus x_0 (x_1 \oplus 1) \oplus x_1 (x_0 \oplus 1) =$$
$$= x_0 x_1 \oplus x_0 x_1 \oplus x_0 \oplus x_1 x_0 \oplus x_1 = x_0 x_1 \oplus x_0 \oplus x_1.$$

Рисунок 2.13 - Функция "ИЛИ" на базисе функций Жегалкина

2.1.4 Техническая реализация переключательных функций

Технический синтез логических схем заключается в замене операций минимизированной и преобразованной к заданному виду переключательной функции логическими элементами заданного базиса.

Пусть, минимизированная ПФ имеет вид:

$$y = \overline{x}_3 x_1 \vee x_2 x_0 .$$

Выполним синтез данного выражения, используя широко распространенный набор логических функций алгебры Шеффера (элементы 2И-НЕ, где цифра 2 соответствует числу входов логического элемента). Для этого вначале выполним преобразование полученного логического выражения. Используя правило де Моргана, получим:

$$y = \overline{x}_3 x_1 \vee x_2 x_0 = \overline{\overline{\overline{x}_3 x_1} \wedge \overline{x_2 x_0}} .$$

Реализация данного выражения, полученная последовательной заменой логических операций соответствующими устройствами, представлена на рисунке 2.14.

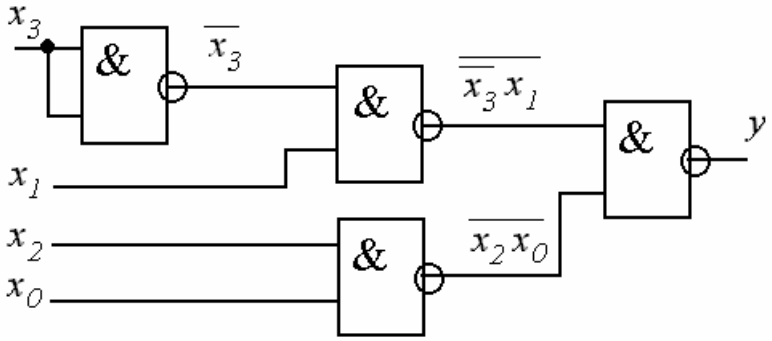
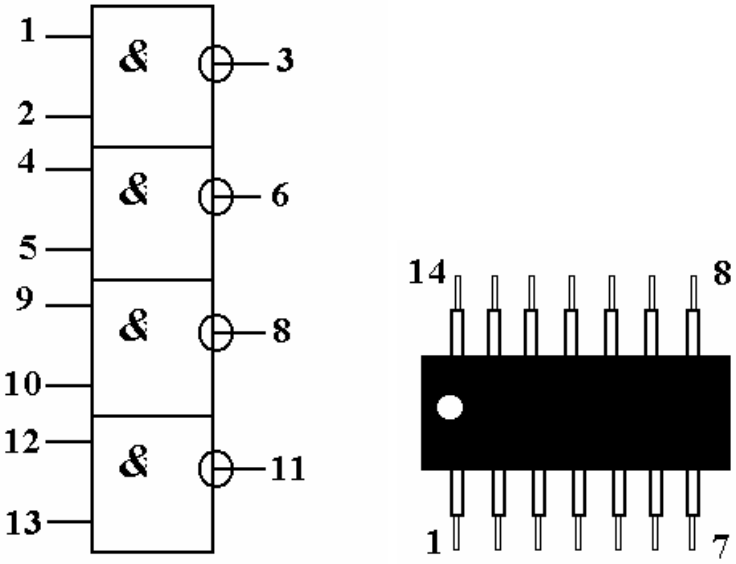


Рисунок 2.14 – Функциональная схема синтезированного устройства

Для реализации принципиальной схемы устройства по справочной литературе находят соответствующую интегральную схему. Например, для рассматриваемого примера это может быть интегральная микросхема К155ЛА3 (рисунок 2.15). На рисунке 2.15,*а* показаны номера выводов интегральной микросхемы и их назначение. Питание обеспечивается подачей на 14 вывод микросхемы +5 В и соединения 7 вывода с минусом источника питания (общим проводом). Корпус микросхемы керамический с двумя рядами выводов изображен на ри-

сунке 2.15, *б*. Первый вывод отмечен углублением в корпусе микросхемы, называемым ключом. Нумерация осуществляется по кругу от ключа против часовой стрелки.



а) *б)*

Рисунок 2.15

Выбором логических элементов, реализующих части функциональной схемы (см. рисунок 2.14) добиваются минимального числа дорожек и их пересечений на печатной плате.

Задание для самостоятельной работы

1. Выполните синтез логического выражения $y = \overline{x}_3 x_1 \vee x_2 x_0$ на базисе И-НЕ.

2. Выполните синтез логического выражения $y = \overline{x}_3 x_1 \vee x_2 x_0$ на базисе ИЛИ-НЕ.

3. Выполните синтез логического выражения $y = \overline{x}_3 x_1 \vee x_2 x_0$ на базисе И, ИЛИ, НЕ.

4. Выполните синтез логического выражения $y = \overline{x}_3 x_1 \vee x_2 x_0$ на базисе И, М2.

5. Сделайте сравнительный анализ сложности схем, полученных при выполнении заданий 1..4.

2.2 Особенности синтеза цифровых автоматов с памятью

2.2.1 Общие сведения о триггере как элементе памяти

Один из этапов синтеза цифровых автоматов с памятью заключается в выборе элементов памяти, определении их количества и характеристик.

Число элементов памяти S, нужное для реализации ЦА, имеющего L_Ψ состояний, выбирается из условия

$$S \geq \text{int}(\log_2 L_\Psi) \ ,$$

где операция int означает округление до ближайшего большего целого числа.

Каждый элемент памяти должен сколь угодно долго находиться в одном из двух состоянии - 0 или 1, иметь возможность изменения этого состояния на противоположное под действием внешнего управляющего сигнала.

Перечисленным требованиям удовлетворяют триггеры.

Триггером называется устройство, имеющее два устойчивых состояния равновесия и способное скачкообразно переходить из одного состояния в другое под воздействием управляющих сигналов.

Состояние и выходной сигнал триггера кодируют цифрами 1 и 0 и обозначают одной и той же двоичной переменной $Q = \{0, 1\}$. Кроме выходного сигнала Q триггер обычно имеет и инверсный выходной сигнал $\overline{Q}$.

Триггер в общем случае сам является элементарным цифровым автоматом с памятью и может быть задан функцией переходов:

$$\Psi^{(t+1)} = Q^{(t+1)} = F(Q^{(t)}, x_{n-1}^{(t)}, ..., x_0^{(t)}).$$

Возможны пять типов перехода триггера в новое состояние:

1. $Q^{(t+1)} = Q^{(t)}$ - триггер не меняет своего состояния (режим хранения информации.

2. $Q^{(t+1)} = 0$ - установка в нуль.

3. $Q^{(t+1)} = 1$ - установка в единицу.

4. $Q^{(t+1)} = \overline{Q}^{(t)}$ - триггер устанавливается в противоположное состояние.

5. $Q^{(t+1)} = X$ - неопределенное состояние при котором обычно наступает логическое противоречие $Q^{(t+1)} = \overline{Q}^{(t+1)}$.

Различают триггеры асинхронные (нетактируемые) и синхронные (тактируемые). Асинхронные триггеры переходят в состояние, определяемое функцией переходов, непосредственно в результате действия той или иной комбинации сигналов на информационных входах. Синхронные триггеры имеют кроме информационных входов вход синхронизации *C*. При действии некоторой комбинации информационных сигналов такой триггер устанавливается в состояние, определяемое функцией переходов, только при условии действия синхросигнала C.

На рисунке 2.16 приведено условное графическое обозначение двух интегральных схем триггеров. В основном поле находится символ триггера - буква *T*, в дополнительном левом - обозначение входных сигналов, а в дополнительном правом - выходных. Так как выходные сигналы триггера всегда обозначаются как Q и $\overline{Q}$, это дополнительное поле может не изображаться.

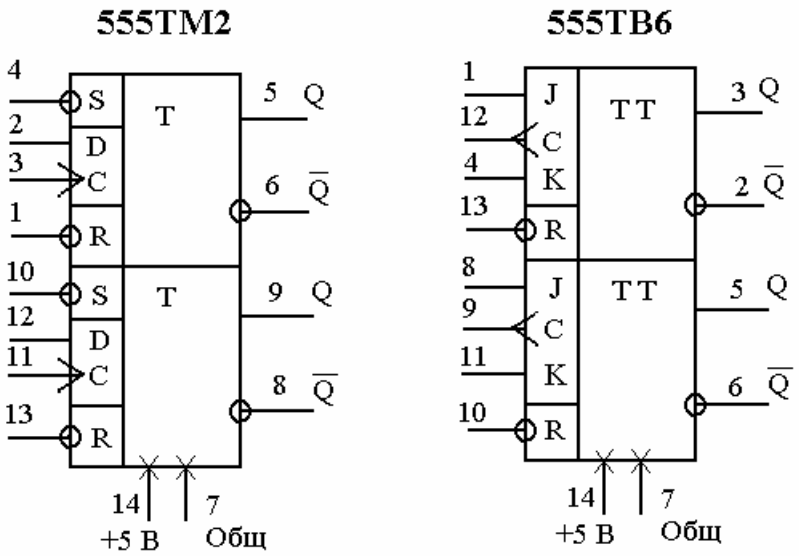


Рисунок 2.16

В зависимости от особенности реакции триггера на воздействие входных сигналов различают динамические триггеры и статические (потенциальные) триггеры.

Статические триггеры реагируют на значение потенциалов (уровней) информационных сигналов, т.е. на уровни напряжения соответствующие нулю и единице. Динамические триггеры реагируют

лишь на перепады уровней, т.е. на моменты перехода от одного уровня (например, уровня принятого за единицу) к другому (уровню нуля). Входы статических и динамических триггеров, соответственно, называют статическими и динамическими входами. Статические и динамические входы в свою очередь подразделяются на прямые и инверсные. Статический инверсный вход отличается от прямого входа тем, что если для достижения определенной реакции триггера на прямой вход подается сигнал x, то при инверсном входе для получения той же реакции триггера должен подаваться сигнал $x' = \bar{x}$. Динамический инверсный вход отличается от прямого входа тем, что на него должен подаваться перепад напряжения от уровня единицы к уровню нуля, если при прямом входе для получения той же реакции триггера подается перепад от уровня нуля к уровню единицы и наоборот. На рисунке 2.17 показаны условные обозначения статических и динамических, прямых и инверсных входов в соответствии с ГОСТ 2.743.

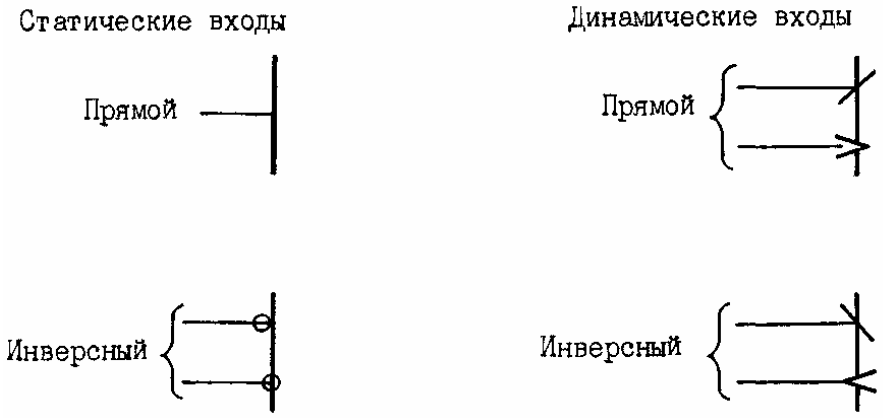


Рисунок 2.17

В общем случае количество информационных входов может быть любым. В цифровой технике в основном применяются триггеры с одним и двумя информационными входами.

2.2.2 Триггеры с одним информационным входом и законы их функционирования

Определим все возможные таблицы переходов для триггеров с одним информационным входом, учитывая, что изменение состояния их происходит под действием сигнала x (t).

Анализ таблиц показывает, что таблица 2.3 и таблица 2.4 по значению нового состояния $Q(t+1)$ с точностью до наоборот похожи соответственно на таблицы 2.5 и 2.6.

Таблица 2.3

x(t)	*Q*	
	Q(t)	*Q(t+1)*
0	0	0
1	0	1
0	1	0
1	1	1

Таблица 2.4

x(t)	*Q*	
	Q(t)	*Q(t+1)*
0	0	0
1	0	1
0	1	1
1	1	0

Таблица 2.5

x(t)	*Q*	
	Q(t)	*Q(t+1)*
0	0	1
1	0	0
0	1	1
1	1	0

Таблица 2.6

x(t)	*Q*	
	Q(t)	*Q(t+1)*
0	0	1
1	0	0
0	1	0
1	1	1

Следовательно, существует только два принципиально различных триггера с одним информационным входом это триггеры *D* и *T* - типов (от английских терминов delay - задержка, toggle - переключатель). Триггер *D* - типа имеет таблицу переходов, соответствующую таблице 2.3. Входной сигнал $x(t)$ обозначается буквой $D(t)$. В соответствии с таблицей переходов функция переходов в СДНФ имеет вид:

$$Q^{(t+1)} = \overline{Q}(t) \cdot D^{(t)} \vee Q^{(t)} \cdot D^{(t)} = D^{(t)}.$$

D - триггеры выполняются, как правило синхронными. Условное изображение *D* - триггера приведено на рисунке 2.18.

С учетом сигнала синхронизации функция переходов может быть записана в следующем виде $Q^{(t+1)} = C \cdot D^{(t)}$.

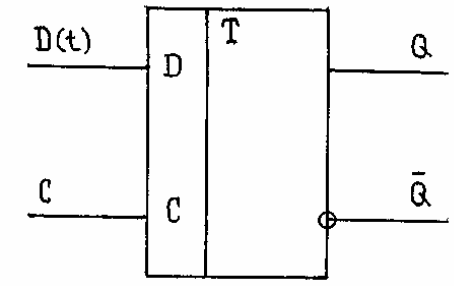


Рисунок 2.18

То есть, если в момент t действия синхросигнала $C(t)$ на входе D-триггера имеется информационный сигнал $D(t)$, то триггер устанавливается в состояние $Q^{(t+1)} = D(t)$, которое сохраняется до воздействия следующего синхроимпульса. При этом временной интервал между синхроимпульсами может быть произвольным. Следовательно, D-триггер запоминает информацию на один такт работы, определяемый интервалом между импульсами $C(t)$ и $C(t+1)$; поэтому он называется триггером запоминания информации или триггером задержки информации.

T - триггер характерен тем, что при поступлении на его вход информационного сигнала, выходной сигнал меняет значение на противоположное. Они могут быть синхронными и асинхронными. На схемах T - триггер изображается как на рисунке 2.19, порядок функционирования описывается таблицей переходов (см. таблицу 2.4) и переключательной функцией вида:

$$Q^{(t+1)} = \overline{Q}^{(t)} \cdot T^{(t)} \vee Q^{(t)} \cdot \overline{T}^{(t)},$$

где $T^{(t)}$ - информационный сигнал.

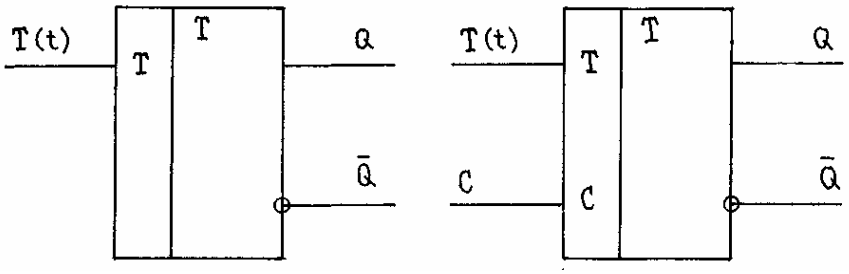


Рисунок 2.19

T - триггер реализует функцию "сумма по модулю 2", так как

$$Q^{(t+1)} = Q^{(t)} \oplus T^{(t)}.$$

Таким образом, если на информационный вход T - триггера поступает сигнал $T^{(t)} = 0$, то триггер сохраняет свое состояние, если же $T^{(t)} = 1$, то меняет на противоположное, т.е. он считает единицы. Поэтому T - триггер называют счетным триггером, а T - вход - счетным входом.

2.2.3 Триггеры с двумя информационными входами

В триггерах с двумя информационными входами, на которые подаются информационные сигналы x_1 и x_0 , возможны четыре комбинации этих сигналов, а таблица переходов для основных видов триггеров может быть представлена в виде таблицы 2.7. Анализ данных таблицы 2.7 позволяет сделать некоторые выводы:

1. При поступлении на первый информационный вход (R , S_0, R_0, E_0, K, V - входы, соответственно) сигнала $x_0 = 1$ триггер "обнуляется", поэтому этот вход и получил название входа обнуления или установки нуля (Reset- сброс).

2. При поступлении на второй вход (S, S_1, R_1, E_1, J –входы, соответственно) сигнала x_1=1 триггеры RS, S, R, E и JK устанавливаются в единичное состояние, т.е. происходит установка триггера в единицу (Set - установка), а DV - триггер при этом переходит в режим хранения информации, т.е. сохраняет предыдущее состояние.

3. Одновременное появление сигналов x_0, x_1=1 для RS-триггера является запрещенной комбинацией, а для остальных триггеров является командой на переход, обусловленный логикой его работы.

Таблица 2.7

Входные сигналы		Значение $Q(t+1)$ на выходе триггера после действия некоторой комбинации входных сигналов					
x_1	x_0	RS-триггер (S, R)	S-триггер (S_1, S_0)	R-триггер (R_1, R_0)	E-триггер (E_1, E_0)	JK-триггер (J, K)	DV-триггер (D,V)
0	0	$Q(t)$	$Q(t)$	$Q(t)$	$Q(t)$	$Q(t)$	$Q(t)$
0	1	0	0	0	0	0	0
1	0	1	1	1	1	1	$Q(t)$
1	1	X	1	0	$Q(t)$	$\overline{Q}(t)$	1

Законы функционирования всех рассматриваемых триггеров описываются переключательной функцией переходов вида

$$Q(t+1) = F\left[Q(t), x_1(t), x_0(t)\right].$$

Для построения функции для каждого конкретного случая необходимо составить таблицу истинности. Таблицы истинности для RS, S, R, JK, E, DV - триггеров представлены в таблицах 2.8...2.13 соответственно.

Таблица 2.8 - *RS*- триггер

α	$Q^{(t)}$	$S^{(t)}$	$R^{(t)}$	$Q^{(t+1)}$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	*X*
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	*X*

Таблица 2.9 - *S*- триггер

α	$Q^{(t)}$	$S_1^{(t)}$	$S_0^{(t)}$	$Q^{(t+1)}$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Таблица 2.10 - *R*- триггер

α	$Q^{(t)}$	$R_1^{(t)}$	$R_0^{(t)}$	$Q^{(t+1)}$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

Таблица 2.11 - *JK*- триггер

α	$Q^{(t)}$	$J^{(t)}$	$K^{(t)}$	$Q^{(t+1)}$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	0

Таблица 10 - *E*- триггер

α	$Q^{(t)}$	$E_1^{(t)}$	$E_0^{(t)}$	$Q^{(t+1)}$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Таблица 11 - *DV*- триггер

α	$Q^{(t)}$	$D^{(t)}$	$V^{(t)}$	$Q^{(t+1)}$
0	0	0	0	0
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Диаграммы Вейча для таблиц истинности (см. таблицы 2.8...2.13) изображены соответственно на рисунках 2.20...2.25.

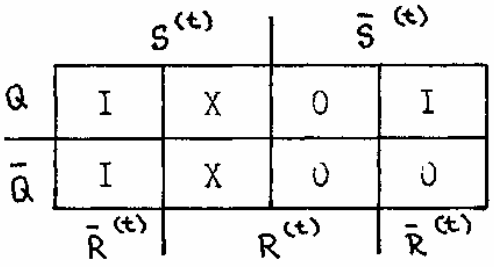


Рисунок 2.20

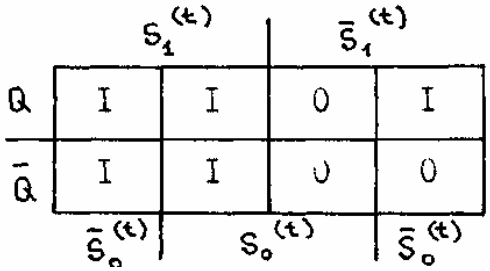


Рисунок 2.21

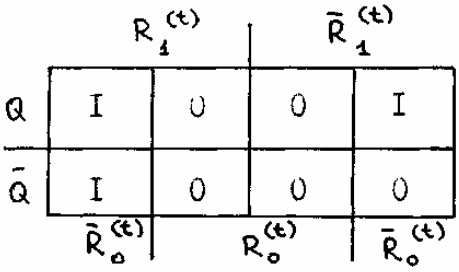


Рисунок 2.22

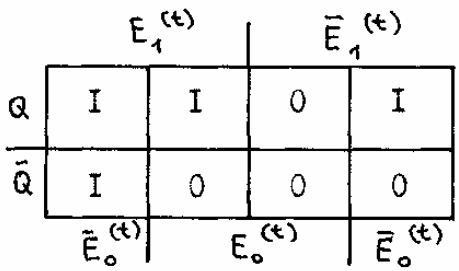


Рисунок 2.23

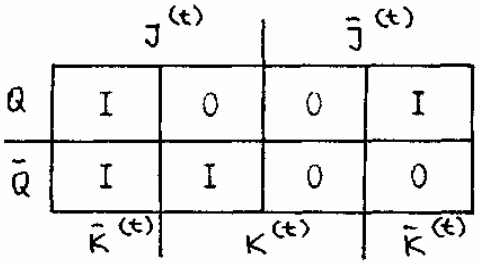


Рисунок 2.24

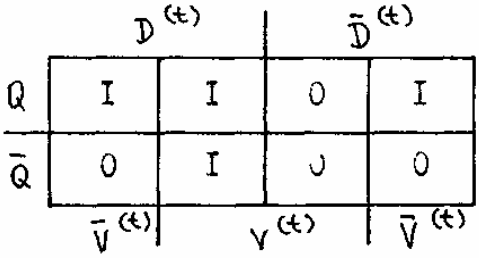


Рисунок 2.25

После преобразования получим следующие минимизированные ПФ для триггеров:

RS –триггер: $Q^{(t+1)} = S^{(t)} \vee Q^{(t)} \cdot \overline{R}^{(t)}$;

S –триггер: $Q^{(t+1)} = S_1^{(t)} \vee Q^{(t)} \cdot \overline{S}_0^{(t)}$;

R –триггер: $Q^{(t+1)} = R_1^{(t)} \overline{R}_0^{(t)} \vee \overline{R}_0^{(t)} Q^{(t)}$;

E –триггер: $Q^{(t+1)} = E_1^{(t)} \overline{E}_0^{(t)} \vee E_1^{(t)} Q^{(t)} \vee \overline{E}_0^{(t)} Q^{(t)}$;

JK –триггер: $Q^{(t+1)} = Q^{(t)} \overline{K}^{(t)} \vee \overline{Q}^{(t)} \cdot J^{(t)}$;

DV –триггер: $Q^{(t+1)} = Q^{(t)} \overline{V}^{(t)} \vee D^{(t)} \cdot V^{(t)}$.

Все вышеперечисленные триггеры могут быть как асинхронными, так и синхронными.

Полученные выражения могут быть использованы для технической реализации триггеров на логических схемах.

Пусть необходимо реализовать *RS* - триггер на элементах 2И-

НЕ. Преобразуем минимизированное выражение переключательной функции *RS* - триггера по правилу де Моргана следующим образом:

$$S^{(t)} \vee Q^{(t)}\overline{R}^{(t)} = \overline{\overline{S}^{(t)} \,\&\, \overline{\left(Q^{(t)}\overline{R}^{(t)}\right)}}.$$

Выполним замену логических операций логическими элементами 2И-НЕ, начиная с выражения в скобках. Полученная реализация *RS* - триггера приведена на рисунке 2.26.

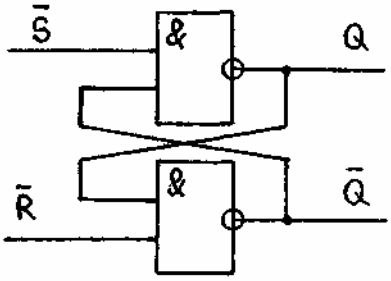


Рисунок 2.26

Для реализации *RS* - триггера на элементах 2ИЛИ-НЕ переключательная функция также по правилу де Моргана приводится к виду:

$$Q^{(t+1)} = S^{(t)} \vee \overline{\overline{Q}^{(t)} \vee R^{(t)}}.$$

Реализация данного выражения приведена на рисунке 2.27.

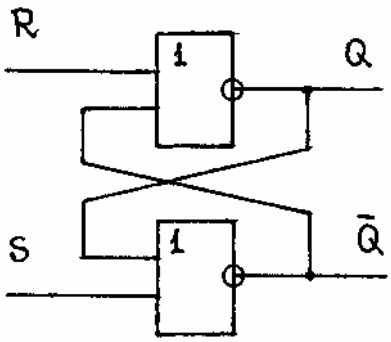


Рисунок 2.27

При построении триггера на ЛЭ ИЛИ-НЕ образуется *RS* - триггер с прямыми статическими входами $x_1 = S$ и $x_0 = R$. При построении триггера на ЛЭ И-НЕ образуется триггер с инверсными статическими входами $\bar{x}_1 = \overline{S}$ и $\bar{x}_0 = \overline{R}$.

В отличие от комбинационных схем, в цифровых автоматах с памятью используются соединения элементов с образованием петли

обратной связи (см. рисунки 2.26, 2.27). Именно обратные связи и порождают феномен памяти.

2.3 Переходные процессы в цифровых автоматах

2.3.1 Анализ переходных процессов в комбинационных схемах

Задержка сигналов логическими элементами в синхронных схемах является вполне определенной величиной и сравнительно легко может быть учтена при их синтезе и анализе. В отличие от синхронных схем задержка сигналов в элементах асинхронных схем определяется многими причинами, носит случайный характер и не может быть определена заранее. Задержки, вносимые элементами в асинхронных схемах, приводят к деформации сигналов, а именно к изменению их длительности и кратковременному отождествлению прямых и инверсных значений одних и тех же сигналов.

Анализ переходных процессов в асинхронных схемах обычно проводят по временным диаграммам работы. На рисунке 2.28 приведена схема, каждый элемент которой вносит задержку dt в распространение сигнала. Там же на графике показаны изменения длительностей выходных сигналов Z и U по сравнению с длительностью входного сигнала X (для $dt = 0$, $X = Z = Y = U$).

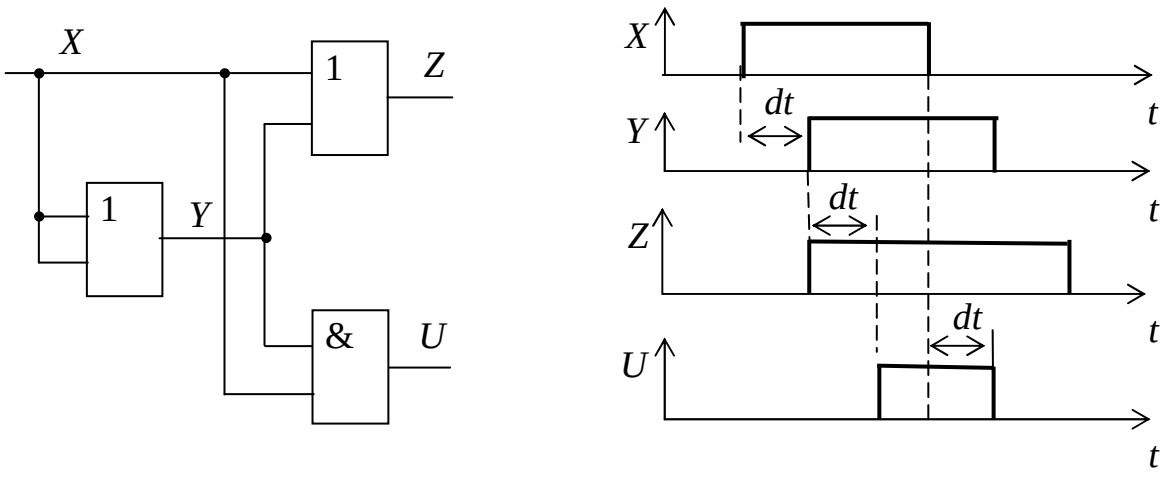


Рисунок 2.28 - Изменение длительности сигналов в схемах

Задержка и деформация сигналов в асинхронных схемах приводят к появлению ложных сигналов на выходах схемы. Возможность появления ложных сигналов в схеме называется риском сбоя сигнала. Различают риск в нуле (появление ложной единицы) и в единице (по-

явление ложного нуля). Риск сбоя в комбинационных схемах может быть статическим и динамическим.

Статическим риском сбоя называют возможность появления ложного выходного сигнала, если при смене одной комбинации значений входных сигналов на другую, изменения сигнала на выходе схемы происходить не должно. Сигнал U в схеме, изображенной на рисунке 2.29, тождественно равен нулю ($U = X \,\&\, \bar{X} = 0$), а сигнал Z - единице ($Z = X \vee \bar{X} = 1$). Однако, как показано на графике, эта схема поражена по выходу U сбоем в нуле, а по выходу Z - сбоем в единице.

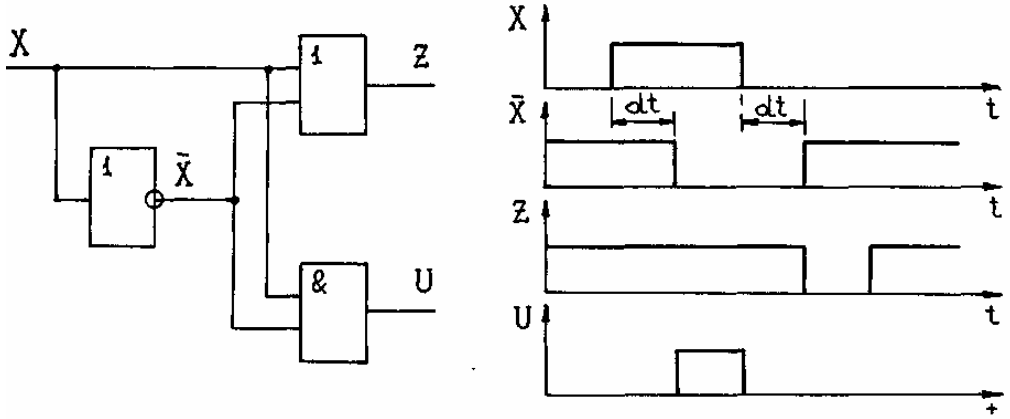


Рисунок 2.29 - Статические сбои в нуле и в единице

Динамическим риском сбоя называют возможность появления многократной смены значений выходных сигналов, если смена одной комбинации значений входных сигналов на другую должна вызвать лишь однократное изменение сигналов на выходе схемы.

Опасность сбоев, в частности, состоит в том, что они могут привести к неправильному срабатыванию элементов памяти ЭВМ. Проблема риска, очевидно, должна решаться при синтезе асинхронных схем. Чтобы искажения сигналов не приводили к сбоям в работе цифрового устройства, необходимо длительность входных сигналов в асинхронных схемах выбирать достаточно большой или применять синхронизацию работы схем.

Минимальная длительность T такта работы асинхронной комбинационной схемы должна быть равна удвоенной максимальной суммарной задержке t_{cx} распространения сигналов, вносимой схемой: $T \geq 2t_{cx}$.

Пусть каждый элемент схемы глубиной G вносит одну и ту же задержку dt в распространение сигналов, тогда $t_{cx} = Gdt$, отсюда сле-

дует, что максимальная частота $F_{\max}$ следования сигналов, подаваемая на входы схемы, будет равна: $F_{\max} = \frac{1}{T} = \frac{1}{2G \cdot dt}$.

Действительно, входные сигналы нельзя заканчивать до момента формирования переднего фронта выходного сигнала, а поскольку задерживается не только передний фронт, но и задний фронт, то новый набор значений входных сигналов нельзя подавать до окончания формирования заднего фронта выходного сигнала. Другими словами, новый набор значений входных сигналов следует подавать не ранее чем через $2t_{cx}$.

2.3.2 Синтез логических схем с учетом риска сбоя

Вследствие конечной скорости распространения сигналов в схемах, значения некоторых сигналов, представляющих прямое и инверсное значение одной и той же переменной величины X, могут оказаться (кратковременно) равными друг другу $(X = \overline{X})$.

Последнее, как известно, приводит к возможности появления ложных сигналов в комбинационных схемах, то есть к риску сбоя по переменной X. Покажем, что для устранения риска сбоя в таких схемах достаточно выполнить для них следующие условия:

на входы схемы от такта к такту подавать только соседние наборы значений аргументов;

реализовать схемой не минимальную, а сокращенную нормальную форму.

Докажем, что при выполнении первого условия представления функции в ДНФ свободно от риска в нуле, а в КНФ - в единице.

Действительно, для любого аргумента X произвольную ДНФ заданной функции f можно записать в виде $f = A\&X \vee B\&\overline{X} \vee C$, где A, B и C - функции, независимые от X.

Если при каких-то фиксированных значениях всех аргументов, отличных от X, функция f равна нулю как при $X = 0$, так и при $X = 1$, то это означает одновременное равенство нулю соответствующих значений функций A, B и C. Но тогда $f = 0$ при любом выборе значений для X и $\overline{X}$, в том числе и для значений $X = \overline{X} = 0$ и $X = \overline{X} = 1$. Тем самым отсутствие риска в нуле в произвольной ДНФ доказано. Аналогично можно показать отсутствие риска в единице для случая КНФ.

Однако ДНФ не свободна от риска в единице, а КНФ - от риска в нуле. Докажем, что при выполнении второго условия представления функций в виде сокращенных форм схемы, реализующие их, свободны от риска как в нуле, так и в единице по всем аргументам.

Действительно, пусть независимо от выбора значения аргумента X существует такой набор значений остальных аргументов, при котором заданная функция f равна единице. Это означает, что в СДНФ функции имеются две соседние конституенты единицы $X \& P$ и $\overline{X} \& P$ (P - некоторая элементарная конъюнкция, зависящая от всех аргументов за исключением X) и что функция f обладает простой импликантой, не зависящей от X. Эта импликанта сохраняет свое значение независимо от выбора значений для X и $\overline{X}$, а поскольку она обязательно содержится в сокращенной ДНФ функции f, то эта форма, свободна от риска в единице по аргументу X.

Тем самым доказано, что при соседнем кодировании последовательностей входных наборов сокращенная ДНФ свободна от риска в нуле и единице по всем переменным. Аналогичное доказательство проводится и для сокращенной КНФ.

Таким образом, при синтезе асинхронных логических схем иногда по соображениям надежности целесообразно отказываться от реализации минимальных форм, а реализовать сокращенные формы функций. В последнем случае, при минимизации частичных функций, необходимо находить наиболее простые сокращенные формы частичной функции.

Анализ переходных процессов в ЦА с памятью заключается в проверке одновременности формирования сигналов возбуждения памяти автомата, приход которых возможен по линиям, имеющим разную глубину. То есть в разрешении так называемой проблемы гонок.

2.3.3 Проблемы гонок в цифровых автоматах с памятью

В сложных автоматах, содержащих несколько элементарных автоматов (триггеров), может возникнуть ситуация, называемая гонками (состязаниями). Суть гонок заключается в том, что в общем случае сигналы каждого из элементарных автоматов, поступая по цепям обратной связи на входы комбинационной части схемы (см. рисунок 2.3), участвуют в формировании функций возбуждения всех других элементарных автоматов.

Из-за различной глубины схем, формирующих функции возбуждения, и разброса задержек в элементах схемы сигналы возбуждения элементарных автоматов поступают на их входы не одновременно. Различным может быть и время переключения самих элементарных автоматов.

Гонки, приводящие к неправильному срабатыванию автомата, называют критическими.

Если при поступлении входного сигнала одновременно должны изменить свое состояние несколько элементарных автоматов, то между ними начинаются гонки. Элементарный автомат, выигравший гонку, оказывает влияние на формирование функций возбуждения остальных элементарных автоматов, что может привести к установке автомата в состояние, не предусмотренное таблицей переходов.

Для исключения процесса гонок рекомендуется применять в качестве элементарных автоматов двухступенчатые триггеры (триггеры с внутренней задержкой). Такие триггеры состоят из двух синхронных одноступенчатых триггеров (рисунок 2.30). Левый триггер является основным (ведущим), а правый вспомогательным (ведомым). Во всех случаях (кроме случая $S = R = 0$, соответствующего режиму хранения) информация с входа ведущего триггера передается с определенной задержкой на вход ведомого триггера. Оба триггера синхронизируются взаимно инверсными сигналами C и $C' = \overline{C}$.

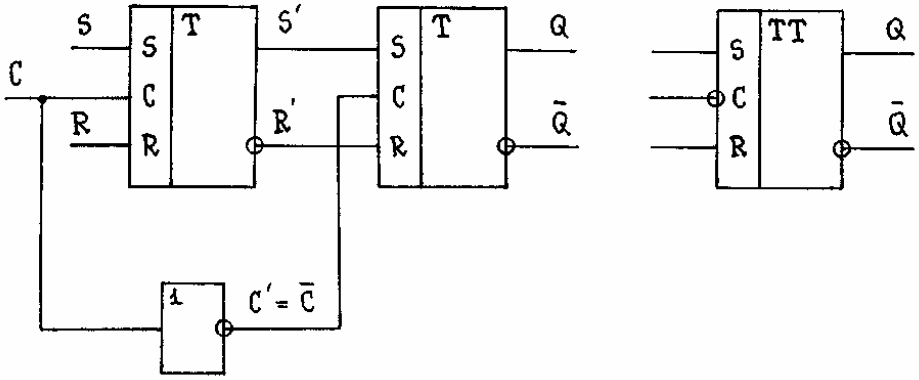


Рисунок 2.30

При действии сигнала $C = 1$ информация, поступающая на входы S и R ведущего триггера, вызывает появление на его выходах надлежащих логических переменных $Q' = S'$ и $\overline{Q}' = R'$. Хотя эта информация и появляется на информационных входах ведомого триггера, однако она не влияет на изменение его состояния, пока на его вхо-

де синхронизации действует сигнал $C' = \overline{C} = 0$. После же переключения сигнала C, когда $C = 0$, а $C' = 1$, ведомый триггер устанавливается в состояние, в котором находится ведущий триггер. Так как в это время $C = 0$, ведущий триггер будет лишен возможности воспринимать новую информацию. Тем самым устанавливается жесткая развязка стадии приема триггером новой информации от стадии передачи информации потребителю.

Поскольку выходной сигнал снимается с ведомого триггера, то работа всего устройства внешне проявляется так, как если бы прием информации со входа двухступенчатого триггера (для передачи на его выход) производится при изменении синхросигнала с уровня $C = 1$ на уровень $C = 0$. Иначе говоря, внешне вход синхронизации ведет себя как инверсный, что отражается на условном обозначении триггера (см. рисунок 2.30). Кроме того, для различения двухступенчатого триггера от одноступенчатого в основном поле условного графического обозначения (УГО) пишут две буквы T.

Не только RS - триггеры, но и некоторые другие могут иметь двухступенчатую структуру. Рассмотрим JK - триггер, построенный на основе синхронного двухступенчатого RC - триггера. Согласно обозначениям, принятым на схеме (рисунок 2.31), сигналы на выходах $\overline{S}'$ и $\overline{R}'$ асинхронного RS - триггера на $ЛЭ_1'$ и $ЛЭ_2'$ определяется выражениями

$$\overline{S}' = \overline{J \cdot \overline{Q} C}, \quad \overline{R}' = \overline{K \cdot Q C}.$$

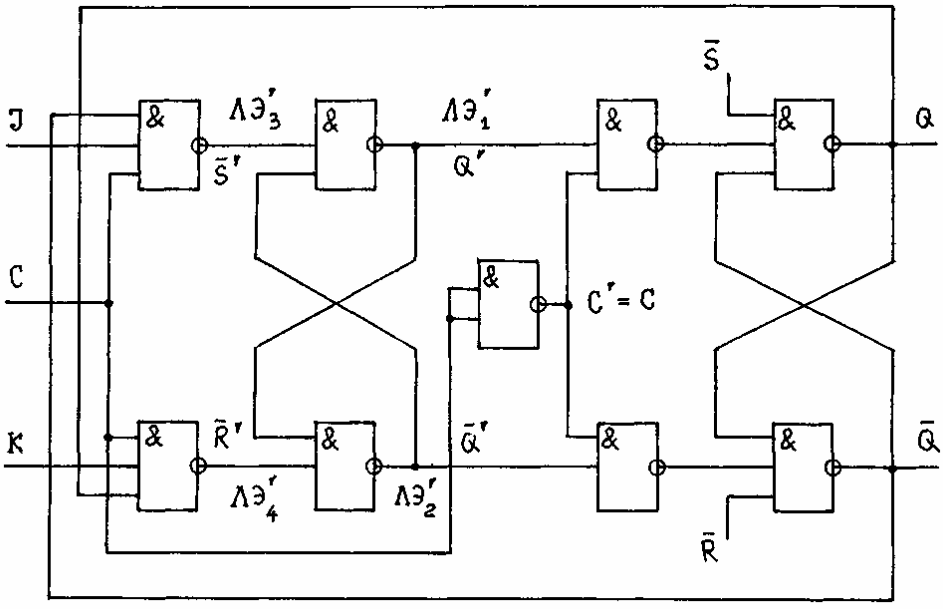


Рисунок 2.31

При $C=0$ имеем $\overline{S}'=1$ и $\overline{R}'=1$, что соответствует режиму хранения информации. При $C=1$ получаем

$$\overline{S}'=\overline{J\cdot\overline{Q}},\quad \overline{R}'=\overline{K\cdot Q}. \qquad (2.1)$$

Под действием сигналов $\overline{S}'$ и $\overline{R}'$ ведущий триггер устанавливается в состояние $Q'(t+1)$, которое при окончании синхроимпульса переписывается в ведомый триггер: $Q(t+1)=Q'(t+1)$. Чтобы убедиться в том, что рассматриваемый закон функционирования соответствует JK - триггеру, составим таблицу истинности (таблица 2.26) с учетом формулы (2.1) и того, что в режиме хранения информации:

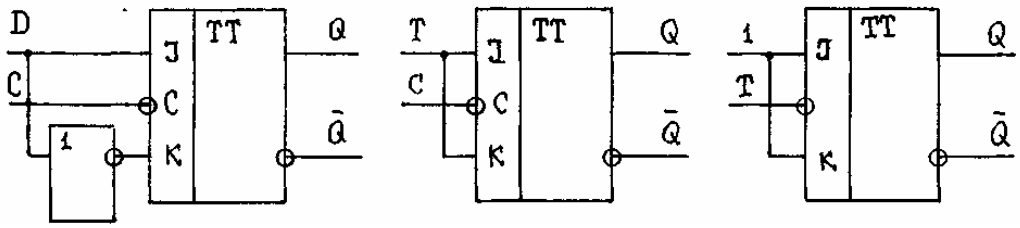
$$Q(t+1)=Q(t)=Q'(t).$$

Сравнивая полученную таблицу истинности с таблицей перехода JK - триггера, рассмотренную в п. 2.2.3 (см. таблицу 2.11), отметим их идентичность. Это означает, что рассматриваемая структура является JK - триггером.

Таблица 2.26 - Таблица истинности двухтактного триггера

α	$Q(t)$	J	K	$\overline{S}'$	$\overline{R}'$	$Q(t+1)=Q'(t+1)$
0	0	0	0	1	1	0 (режим хранения)
1	0	0	1	1	0	0 (установка 0)
2	0	1	0	0	1	1 (установка 1)
3	0	1	1	0	1	1 (инверсия)
4	1	0	0	1	1	1 (режим хранения)
5	1	0	1	1	0	0 (установка 0)
6	1	1	0	0	1	1 (установка 1)
7	1	1	1	1	0	0 (инверсия)

В свою очередь на JK - триггерах могут быть реализованы другие типы двухступенчатых триггеров путем несложных внешних соединений (рисунок 2.32).



D - триггер Синхронный T - триггер Асинхронный T - триггер

Рисунок 2.32

Глава 3. ЭЛЕМЕНТНАЯ БАЗА ЦИФРОВЫХ УСТРОЙСТВ

3.1 Дешифраторы, шифраторы и схемы сравнения

3.1.1 Дешифраторы одноступенчатого и двухступенчатого типов

Дешифратором (декодером - Decoder) называется цифровое устройство, имеющее n входов и m выходов и преобразующее входной код в сигнал на одной отдельной выходной линии. Другими словами, в дешифраторе каждому предусмотренному набору входных сигналов соответствует один вполне определенный возбужденный выход.

Дешифраторы являются преобразователями кодов, выполняющими преобразование двоичного или двоично-десятичного кодов в унитарный код. Унитарный код двоичного n - разрядного числа представляется 2^n разрядами, только один из разрядов которого равен 1.

Дешифратор называется полным, если он имеет количество выходов $m = 2^n$. При этом на каждом из выходов появляется единица (т.е. выход возбужден) только в том случае, если на вход приходит соответствующая комбинация входных сигналов. Переключательная функция каждого выхода является конституентой единицы (рисунок 3.1).

α	x_1	x_0	y_3	y_2	y_1	y_0
0	0	0	0	0	0	1
1	0	1	0	0	1	0
2	1	0	0	1	0	0
3	1	1	1	0	0	0

$$y_3 = x_1 \cdot x_0$$
$$y_2 = x_1 \cdot \bar{x}_0$$
$$y_1 = \bar{x}_1 \cdot x_0$$
$$y_0 = \bar{x}_1 \cdot \bar{x}_0$$

Рисунок 3.1 - Таблица истинности и переключательные функции двухразрядного дешифратора

Реализация переключательных функций на базисе ЛЭ «И», «НЕ» и условное графическое обозначение дешифратора приведены на рисунке 3.2.

Дешифраторы могут иметь прямые и инверсные выходы. В зависимости от наличия и отсутствия сигнала синхронизации дешифраторы могут быть синхронными и асинхронными.

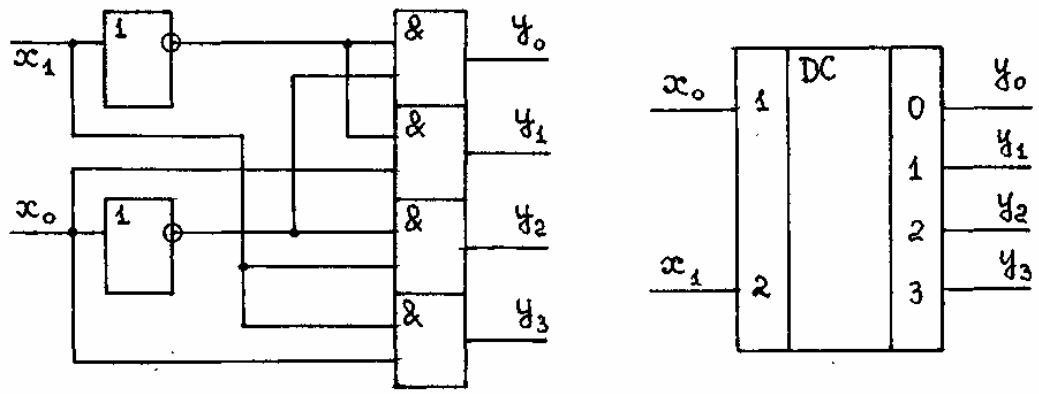


Рисунок 3.2

Реализация полных дешифраторов с большим числом n в интегральном исполнении осложняется проблемой размещения большого числа выводов на корпусе микросхемы. Действительно, даже для n = 4, $m = 2^4 = 16$ число информационных выводов (без учета управляющих выводов и выводов подключения питающих напряжений) равно 20.

При n = 5 число информационных выводов составляет 37, их размещение на стандартном корпусе интегральной микросхемы уже трудно реализуемо.

Задача дешифрирования при большом числе входных переменных решается в несколько этапов (ступеней), соответственно схемы дешифраторов такого типа называют многоступенчатыми.

Количество ступеней дешифрации может быть различным. Наиболее часто встречаются двухступенчатые дешифраторы. При построении такого n - разрядного дешифратора n - входные логические переменные делятся на две подгруппы по $n/2$ переменных в каждой, если количество переменных четное, или по $(n - 1)/2$ и $(n + 1)/2$ переменных в каждой, если n - нечетное число. Для каждой подгруппы строится одноступенчатый дешифратор. После чего выводы дешифраторов объединяются схемой «И» так как показано на рисунке 3.3.

При четном n каждый из одноступенчатых дешифраторов DC1 или DC2 имеет по $2^{n/2}$ выходов, которым соответствуют переменные y_i и z_i , являющиеся результатом $n/2$ - местных операций «И» входящих в каждую подгруппу. Для получения всех конституент единиц, определенных на совокупности n входных переменных, следует образовать всевозможные конъюнкции $y_i\, z_i$ для всех комбинаций i и j ($0 \le i, j \le 2^{n/2} - 1$). Одноступенчатые дешифраторы DC1 и DC2 образуют первую ступень дешифрации, а схемы «И», объединяющие выходы

дешифраторов, являются второй ступенью дешифрации. Необходимое количество схем "И" во второй ступени дешифрации равно 2^n .

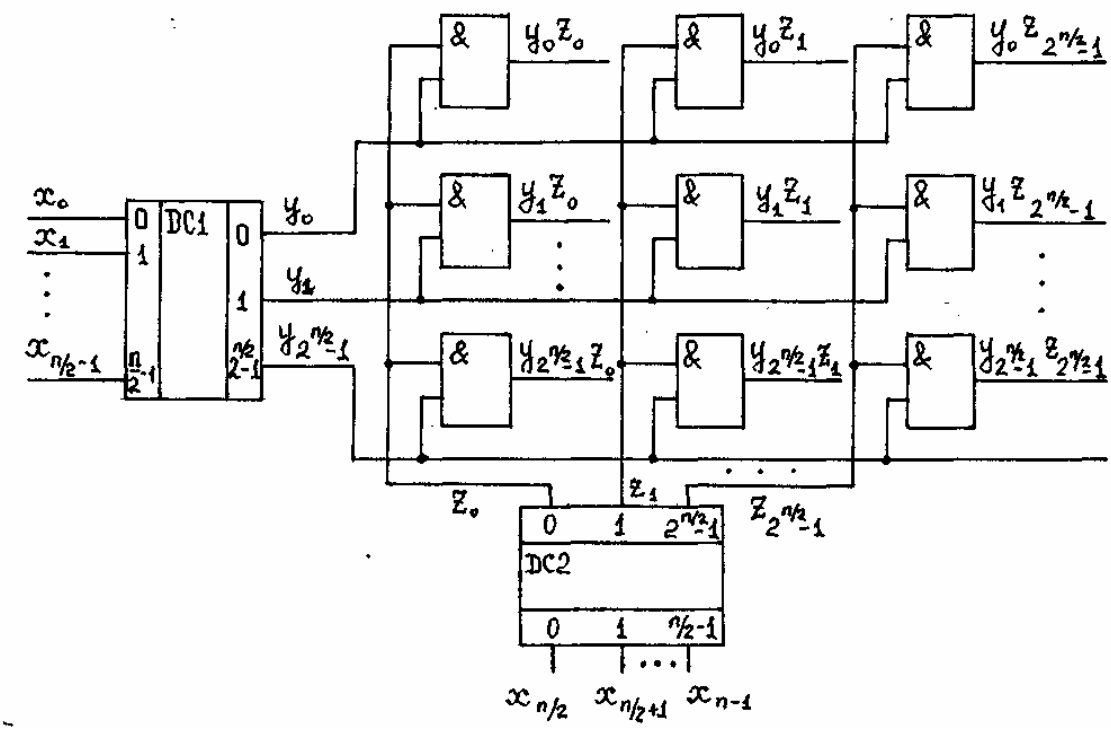


Рисунок 3.3

Применение двухступенчатого дешифратора ведет к увеличению числа корпусов интегральных схем, но используемые схемы дешифраторов в этом случае менее дорогостоящие и содержат меньшее число выводов.

Основными параметрами, по которым производится сравнение различных схем дешифраторов, являются время задержки сигнала в процессе дешифрации и аппаратурные затраты.

Поэтому многоступенчатые схемы дешифраторов обладают несомненным достоинством в экономии аппаратурных затрат, но проигрывают по сравнению с одноступенчатыми схемами во времени задержки сигнала.

Буквенное обозначение микросхем дешифраторов - ИД. Например, на рисунке 3.4 показано условное графическое обозначение неполного двоично-десятичного $(4 \rightarrow 10)$ дешифратора 561ИД1 и сдвоенного синхронного (Е – сигнал разрешения работы дешифратора) полного дешифратора $2 \rightarrow 4$ 531ИД14.

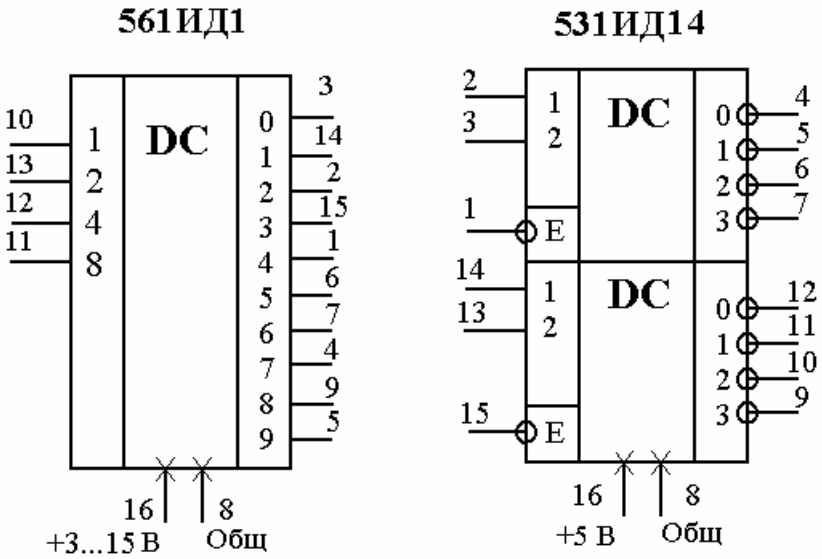


Рисунок 3.4

Задание для самостоятельной работы

1. Синтезировать двоично-десятичный дешифратор для наборов 1 и 8 на логических элементах «2И-НЕ».
2. Синтезировать двоично-десятичный дешифратор для наборов 1 и 8 на логических элементах «2ИЛИ-НЕ».
3. Синтезировать двоично-десятичный дешифратор для наборов 2 и 7 на логических элементах «3И-НЕ».
4. Синтезировать двоично-десятичный дешифратор для наборов 2 и 7 на логических элементах «3ИЛИ-НЕ».
5. Синтезировать двоично-десятичный дешифратор для наборов 3 и 6 на логических элементах «4И-НЕ».
6. Синтезировать двоично-десятичный дешифратор для наборов 3 и 6 на логических элементах «4И-НЕ».
7. Синтезировать двоично-десятичный дешифратор для наборов 4 и 5 на логических элементах «2И-НЕ».
8. Синтезировать двоично-десятичный дешифратор наборов 4 и 5 на логических элементах «2ИЛИ-НЕ».

3.1.2 Преобразователи кодов. Шифраторы

Шифраторы выполняют функцию обратную дешифраторам, т.е. преобразуют унитарный код в двоичный или двоично-десятичный. На практике часто возникает необходимость и других преобразований кодов. Например, числовая информация в БВМ может храниться в прямом коде, а выполнение отдельных операций выгоднее с точки зрения аппаратурных затрат производить в обратном и - дополнительном кодах. Следовательно, возникает необходимость преобразовать прямой код числа в обратный или дополнительный, а затем, после выполнения операции, произвести обратные преобразования.

Рассмотрим методику синтеза преобразователей кода на примере преобразователя прямого кода в обратный.

Значения положительных чисел хранятся в памяти БВМ в виде прямых машинных кодов

$$[A]_{пк} = a_n a_{m-1} \ldots a_0,$$

где a_n - цифра знакового разряда (нуль, если A >0; единица, если A < 0).

Преобразователь прямого кода в обратный должен иметь (n + 1) - входов и (n + 1) - выходов. На входы преобразователя подаются сигналы, отображающие цифры a_i прямого кода, а на выходах образуются сигналы, отображающие цифры a_i^* обратного кода. Причем, если a_n = 0, то $a_i^* = a_i$, а если a_n = 1, то значение разряда инвертируется, т.е. $a_i^* = \overline{a}_i$. Такому алгоритму соответствует таблица истинности и переключательная функция преобразователя i - го разряда, представленные на рисунке 3.5.

α	a_n	a_i	a_i^*
0	0	0	0
1	0	1	1
2	1	0	1
3	1	1	0

$$a_i^* = a_n \cdot \overline{a}_i \vee \overline{a}_n \cdot a_i$$

Рисунок 3.5 Таблица истинности и переключательная функция преобразователя прямого кода в обратный

Структурная схема такого преобразователя представлена на рисунке 3.6.

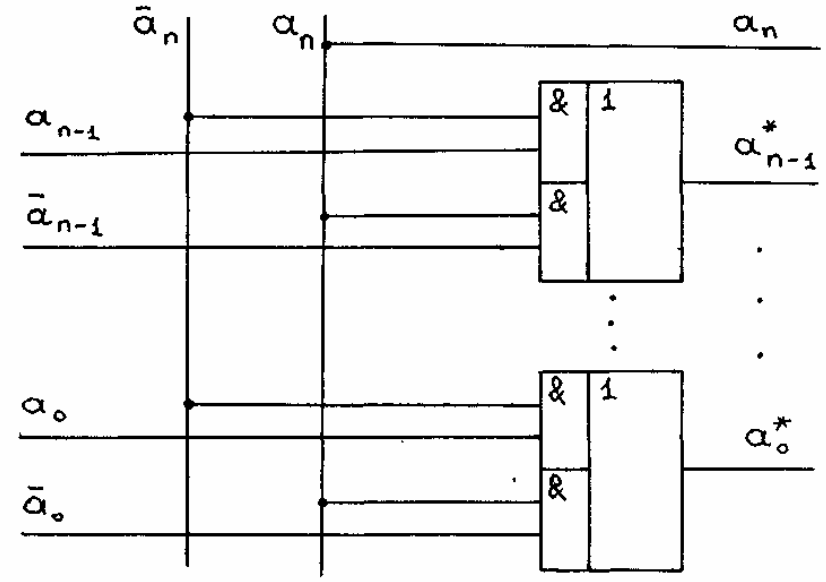


Рисунок 3.6

Для упрощения конструкции преобразователя используется парафазный входной сигнал (все входные переменные поступают как в прямом, так и в инверсном виде).

Другой пример преобразователей кодов - преобразователь двоичного кода в код Грея. Код Грея используется в датчиках углового перемещения и обеспечивает измерение угла с меньшими, чем обычный двоичный код ошибками.

Таблица истинности преобразователя трехразрядного двоичного кода (x_2, x_1, x_0) в трехразрядный код Грея (y_2 , y_1 , y_0) приведена в таблице 3.1.

Таблица 3.1 – Таблица соответствия двоичного кода и кода Грея

Номер набора	Входной двоичный код			Выходной код Грея		
	x_2	x_1	x_0	y_2	y_1	y_0
0	0	0	0	0	0	0
1	0	0	1	0	0	1
2	0	1	0	0	1	1
3	0	1	1	0	1	0
4	1	0	0	1	1	0
5	1	0	1	1	1	1
6	1	1	0	1	0	1
7	1	1	1	1	0	0

Переключательные функции в СДНФ и МДНФ имеют вид:

$y_2 = x_2\bar{x}_1\bar{x}_0 \vee x_2\bar{x}_1x_0 \vee x_2x_1\bar{x}_0 \vee x_2x_1x_0 = x_2;$

$y_1 = \bar{x}_2x_1\bar{x}_0 \vee \bar{x}_2x_1x_0 \vee x_2\bar{x}_1\bar{x}_0 \vee x_2\bar{x}_1x_0 = \bar{x}_2x_1 \vee x_2\bar{x}_1 = x_2 \oplus x_1;$

$y_0 = \bar{x}_2\bar{x}_1x_0 \vee \bar{x}_2x_1\bar{x}_0 \vee x_2\bar{x}_1x_0 \vee x_2x_1\bar{x}_0 = x_1\bar{x}_0 \vee \bar{x}_1x_0 = x_1 \oplus x_2.$

Реализация преобразователя на базе сумматоров по модулю 2 показана на рисунке 3.7

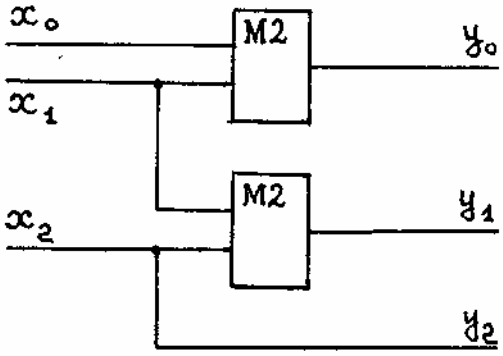


Рисунок 3.7

Наиболее часто используемые преобразователи кодов реализуются в виде интегральных микросхем (ИС). На рисунке 3.8 слева показан шифратор 8→3 (ИС 155 ИВ1). Назначение сигналов *E* – сигнал включения шифратора; *G* – сигнал, свидетельствующий о наличии хотя бы одного возбужденного входа; *E0* – выходной сигнал, свидетельствующий об отсутствии возбужденных входов. Справа на рисунке 3.8 показан шифратор 9→4 (ИС 555ИВ3).

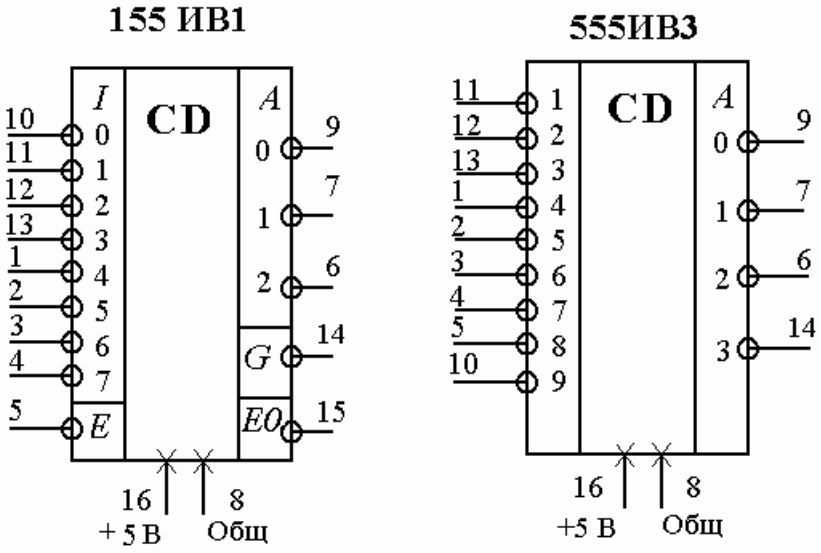


Рисунок 3.8

Задания для самостоятельной работы

1. Синтезировать шифратор унитарного кода в двоичный код семисегментного индикатора для цифр 1 и 8 на логических элементах «2И-НЕ».

2. Синтезировать шифратор унитарного кода в двоичный код семисегментного индикатора для цифр 1 и 8 на логических элементах «2ИЛИ-НЕ».

3. Синтезировать шифратор унитарного кода в двоичный код семисегментного индикатора для цифр 2 и 7 на логических элементах «2И-НЕ».

4. Синтезировать шифратор унитарного кода в двоичный код семисегментного индикатора для цифр 2 и 7 на логических элементах «2ИЛИ-НЕ».

5. Синтезировать шифратор унитарного кода в двоичный код семисегментного индикатора для цифр 3 и 6 на логических элементах «2И-НЕ».

6. Синтезировать шифратор унитарного кода в двоичный код семисегментного индикатора для цифр 3 и 6 на логических элементах «2ИЛИ-НЕ».

7. Синтезировать шифратор унитарного кода в двоичный код семисегментного индикатора для цифр 4 и 5 на логических элементах «2И-НЕ».

8. Синтезировать шифратор унитарного кода в двоичный код семисегментного индикатора для цифр 4 и 5 на логических элементах «2ИЛИ-НЕ».

3.1.3 Многоразрядные схемы сравнения

Многоразрядные схемы сравнения используются для сравнения и определения идентичности двух или нескольких n - разрядных двоичных слов.

Принципы построения и способы технической реализации многоразрядных схем сравнения рассмотрим на следующем примере.

Пусть на n выходах каких-то двух устройств формируются два n- разрядные слова

$$A = a_{n-1}...a_1a_0 \quad \text{и} \quad B = b_{n-1}...b_1b_0 .$$

Требуется зарегистрировать совпадение (идентичность) этих слов.

Для решения поставленной задачи необходимо иметь *n*- разрядную схему сравнения, которая путем поразрядного сравнения элементов a_i и b_i определяет соответствие слов *A* и *B*. Поразрядное сравнение может быть произведено одноразрядной схемой сравнения. Таблица истинности и переключательная функция одноразрядной схемы сравнения приведены на рисунке 3.9.

α	a_i	b_i	R_i $(a=b)$	$P_i = \overline{R}_i$ $(a \neq b)$
0	0	0	1	0
1	0	1	0	1
2	1	0	0	1
3	1	1	1	0

$$R_i = \overline{a}_i \cdot \overline{b}_i \vee a_i \cdot b_i;$$
$$P_i = \overline{R}_i = \overline{\overline{a}_i \cdot \overline{b}_i \vee a_i \cdot b_i} =$$
$$= a_i \cdot \overline{b}_i \vee \overline{a}_i \cdot b_i = a_i \oplus b_i$$

Рисунок 3.9 - Таблица истинности и переключательная функция одноразрядной схемы сравнения

Для обнаружения совпадения a_i и b_i во всех разрядах слов *A* и *B* необходимо образовать конъюнкцию *n* - переменных R_i . Таким образом, переключательная функция многоразрядной схемы сравнения описывается выражением:

$$У = \bigwedge_{i=0}^{n-1} R_i = \bigwedge_{i=0}^{n-1} (\overline{a}_i \cdot \overline{b}_i \vee a_i \cdot b_i).$$

Функциональная схема устройства, удовлетворяющего этому выражению, показана на рисунке 3.10. Входной сигнал подается парафазным способом (в прямом и инверсном виде).

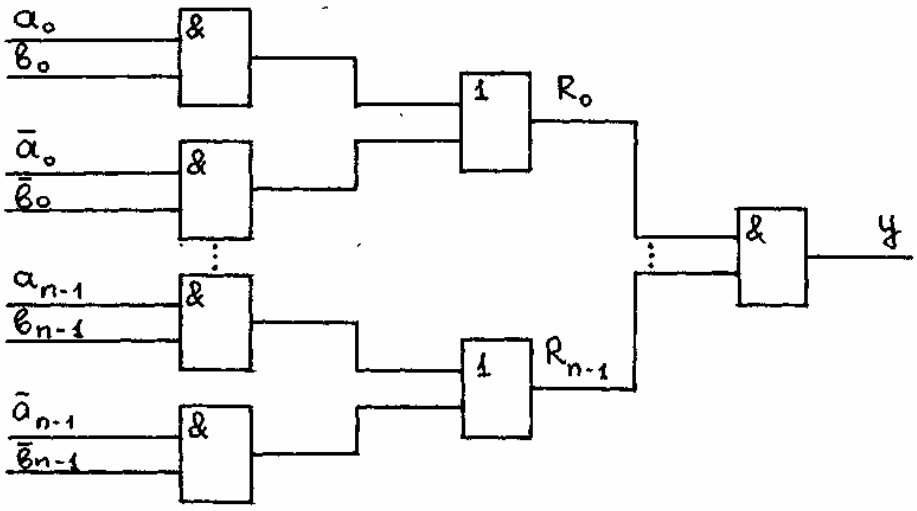


Рисунок 3.10

3.2 Цифровые коммутаторы

3.2.1 Мультиплексоры

Мультиплексором или цифровым коммутатором называется устройство, выполняющее операцию передачи сигнала (или кода) с каждой из заданных ее входным адресом линий (или шины) в выходную линию (шину). Мультиплексоры часто называют селекторами данных (Data Selector/ Multiplexer).

Мультиплексоры широко применяются во всех отраслях радиотехники, занимающихся сбором и обработкой информации в цифровом виде.

Структура мультиплексора определяется требованиями к алгоритму организации связи между источниками и потребителями информации. Мультиплексоры могут быть стробируемыми (синхронными) и нестробируемыми (асинхронными). Наиболее широко применяются стробируемые мультиплексоры. Передача информации в них происходит при наличии на входе синхронизации сигнала C = 1.

Принцип работы мультиплексора рассмотрим на примере мультиплексирования четырех одноразрядных сигналов: x_0, x_1, x_2, x_3,. Входы x_i называются информационными. Входы a_i — адресными, они задают номер входного сигнала, который необходимо передать на

выход z . Работа мультиплексора задается таблицей истинности (таблица 3.2).

Таблица 3.2

C	a_1	a_0	z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	x_0
1	0	1	x_1
1	1	0	x_2
1	1	1	x_3

В общем случае число информационных входов может быть *n*, при этом число адресных входов *m* связано с *n* формулой $n = 2^m$.

В соответствии с таблицей истинности (см. таблицу 3.2) переключательная функция мультиплексора имеет вид:

$$z = \bar{a}_1\bar{a}_0C \cdot x_0 \vee \bar{a}_1a_0C \cdot x_1 \vee a_1\bar{a}_0C \cdot x_2 \vee a_1a_0C \cdot x_3 .$$

Выражение может быть преобразовано следующим образом:

$$z = y_0x_0 \vee y_1x_1 \vee y_2x_2 \vee y_3x_3,$$

где $y_0 = \bar{a}_1\bar{a}_0C; \quad y_1 = \bar{a}_1a_0C; \quad y_2 = a_1\bar{a}_0C; \quad y_3 = a_1a_0C.$

Выражения для y_0, y_1, y_2, y_3 соответствуют переключательным функциям выходов дешифратора.

С учетом этого схема мультиплексора для случая $n = 4$, $m = 2$ представлена на рисунке 3.11. Мультиплексоры в интегральном исполнении имеют буквенное обозначение КП. Например, на рисунке 3.12 показан асинхронный мультиплексор $8 \rightarrow 1$ (ИС 155КП5).

Задания для самостоятельной работы

1. Синтезировать мультиплексор, преобразующий трехразрядный параллельный код в последовательный на логических элементах типа «2И-НЕ».

2. Синтезировать мультиплексор, преобразующий трехразрядный параллельный код в последовательный на логических элементах типа «2ИЛИ-НЕ».

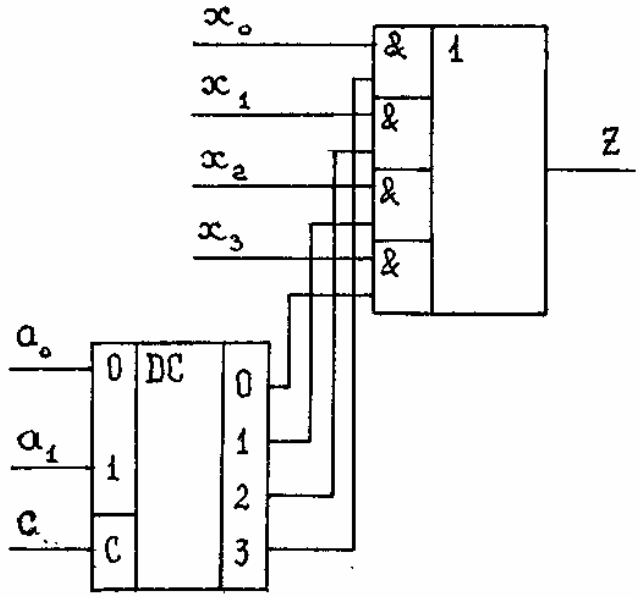


Рисунок 3.11

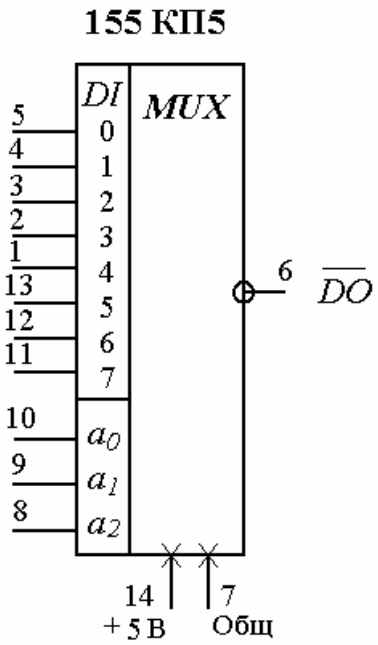


Рисунок 3.12

3. Синтезировать мультиплексор, преобразующий трехразрядный параллельный код в последовательный на логических элементах типа «ЗИ-НЕ».

4. Синтезировать мультиплексор, преобразующий трехразрядный параллельный код в последовательный на логических элементах типа «ЗИЛИ-НЕ».

5. Синтезировать мультиплексор с применением минимального числа логических элементов типа «И-НЕ», преобразующий восьмиразрядный параллельный код в последовательный.

6. Синтезировать мультиплексор с применением минимального числа логических элементов типа «ИЛИ-НЕ» преобразующий восьмиразрядный параллельный код в последовательный.

3.2.2 Демультиплексоры

Демультиплексор выполняет обратное мультиплексору преобразование: сигнал, подаваемый на вход, передается в одно из заданных направлений. Простейший демультиплексор с одного направления в четыре имеет один информационный вход x, четыре выхода z_0, z_1, z_2, z_3 и два адресных входа, задающих номер активного выхода.

Работа демультиплексора описывается таблицей истинности (таблица 3.3).

Таблица 3.3 – Таблица истинности демультиплексора

C	a_1	a_0	z_0	z_1	z_2	z_3
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	0	0	0
1	0	0	x	0	0	0
1	0	1	0	x	0	0
1	1	0	0	0	x	0
1	1	1	0	0	0	x

Переключательные функции записываются для каждого выхода отдельно:

$$z_0 = C\overline{a}_1\overline{a}_0 x = y_0 x;$$
$$z_1 = C\overline{a}_1 a_0 x = y_1 x;$$
$$z_2 = C a_1 a_0 x = y_2 x;$$
$$z_3 = C a_1 a_0 x = y_3 x.$$

Сигналы y_0, y_1, y_2, y_3 разрешения передачи входного сигнала x на соответствующий выход z_0, z_1, z_2, z_3 могут быть сформированы дешифратором. Реализованная с учетом этого схема демультиплексора представлена на рисунке 3.13.

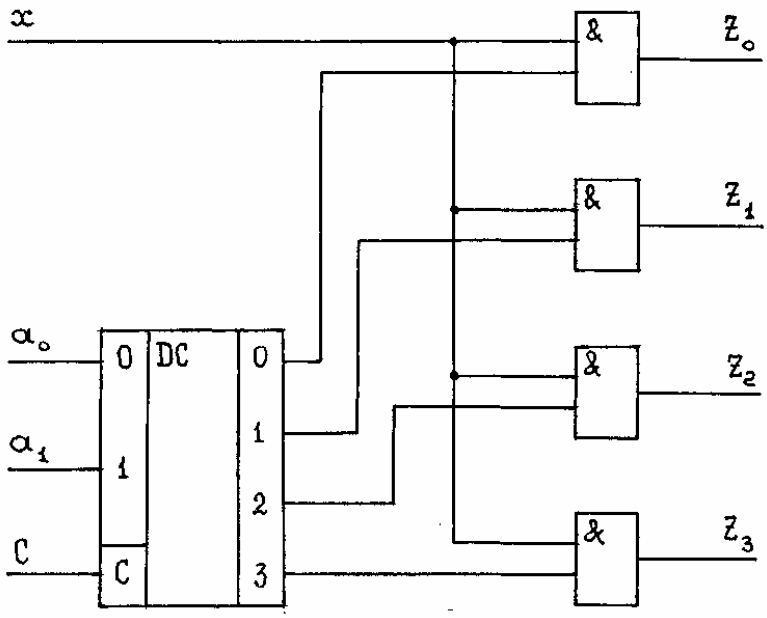


Рисунок 3.13

Выполненный в виде микросхемы демультиплексор имеет, как и дешифратор, буквенное обозначение ИД. Например, сдвоенный де-

мультиплексор 1→ 4 реализован в виде микросхемы 555ИД4, Условное графическое обозначение демультиплексора (Demultiplexer) показано на рисунке 3.14.

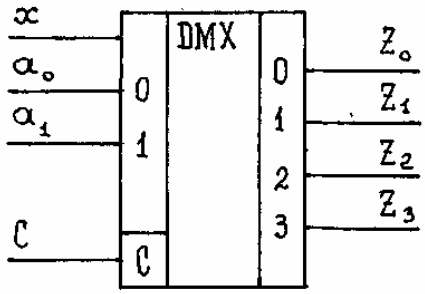


Рисунок 3.14

Задание для самостоятельной работы

1. Синтезировать демультиплексор, преобразующий последовательный код в параллельный четырехразрядный на логических элементах типа «ИЛИ-НЕ».

2. Синтезировать демультиплексор, преобразующий последовательный код в параллельный восьмиразрядный на логических элементах типа «И-НЕ».

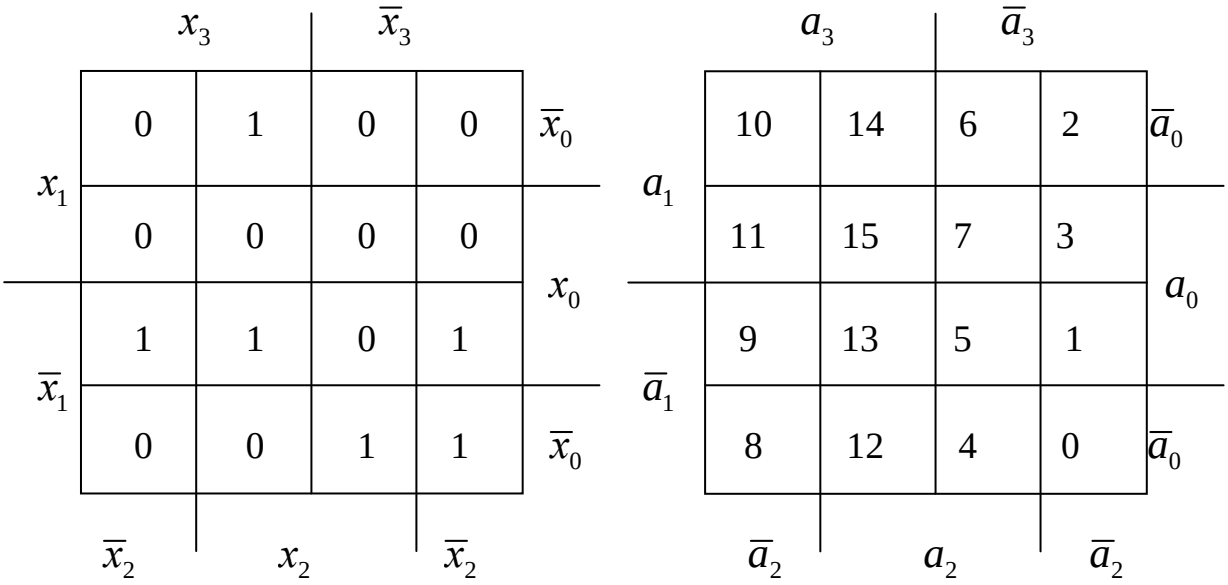
3. Синтезировать демультиплексор, преобразующий последовательный код в параллельный восьмиразрядный на логических элементах типа «ИЛИ-НЕ».

3.2.3 Синтез комбинационных схем на мультиплексорах

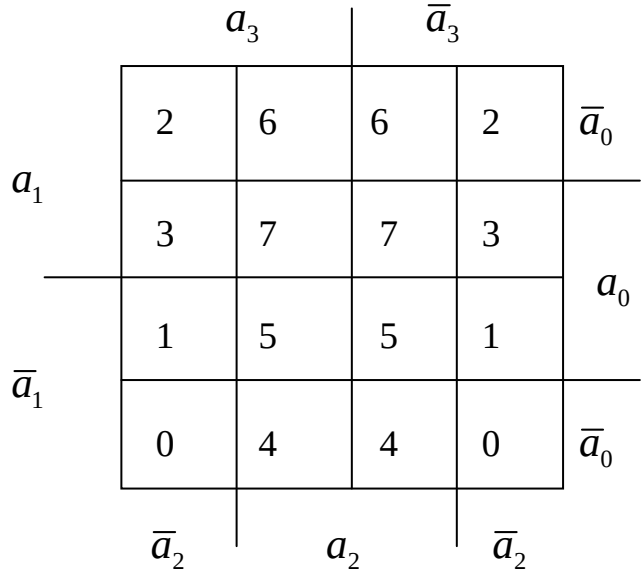
Переключательная функция, выполняемая мультиплексором $2^n \to 1$, по структуре полностью совпадает с СДНФ представления функций n переменных. Из этого следует, что любую переключательную функцию n переменных можно реализовать на мультиплексоре $2^n \to 1$, подав на входы x_i постоянные (0 или 1) или изменяемые величины.

Пусть требуется реализовать функцию $f(x)$ четырех переменных x_3, x_2, x_1, x_0, заданную диаграммой Вейча (рисунок 3.15,а), на мультиплексоре 8→1. Такой мультиплексор имеет три адресных входа, на которые можно подать три из четырех переменных. Пусть на адресные входы мультиплексора с весами 4, 2, 1 (A_2, A_1, A_0) подаются переменные x_2, x_1, x_0 соответственно. Тогда адрес входного инфор-

мационного канала мультиплексора будет определяться числом $i_{(2)} = a_2 a_1 a_0;\ i_{(10)} = 0...7$. Клетки диаграмм Вейча четырех переменных нумеруются числами $j_{(2)} = a_3 a_2 a_1 a_0;\ j_{(10)} = 0...15$ (рисунок 3.15, б). Аналогично можно пронумеровать клетки диаграммы Вейча адресами каналов i мультиплексора, не учитывая переменную a_3 (рисунок 3.15,в).



а) $f(x)$ *б)* j



в) i

Рисунок 3.15

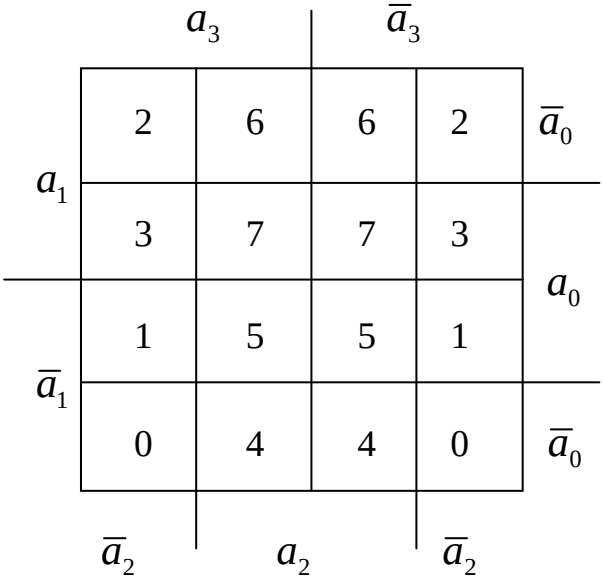
Исходная диаграмма Вейча превратилась в две однотипных восьмиклеточных диаграммы. С учетом значений функции $f(x)$, заданной единицами в диаграмме Вейча (см. рисунок 3.15,а), определим значения входных переменных, подаваемых на информационные входы мультиплексора.

Для того чтобы отличать информационные входы мультиплексора от аргументов функции $f(x)$, обозначим их буквами DI (от английского Data Input, что переводится как "входные данные"). Из рисунков 3.15,*а* и 3.15,*в* следует, что

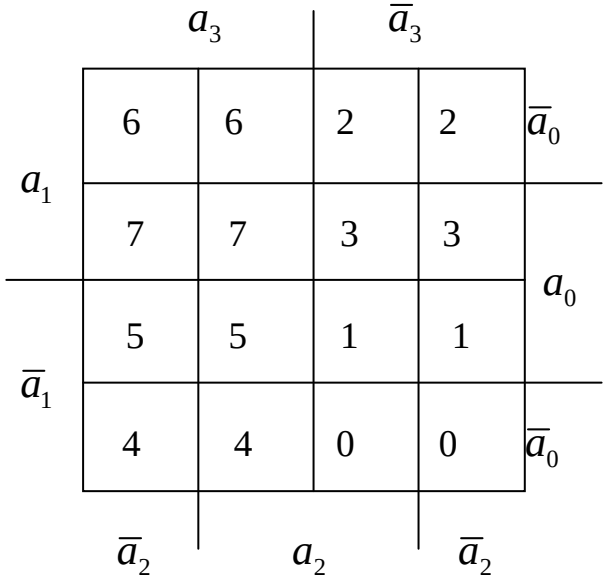
$$DI_0 = 0 \vee x_3 = x_3; \quad DI_1 = x_3 \vee \overline{x}_3 = 1;$$
$$DI_3 = 0 \vee 0 = 0; \quad DI_4 = \overline{x}_3 \vee 0 = \overline{x}_3; \quad DI_5 = x_3 \vee 0 = x_3;$$
$$DI_6 = x_3 \vee 0 = x_3; \quad DI_7 = 0 \vee 0 = 0.$$

По полученным значениям DI_i можно реализовать функцию $f(x)$ на мультиплексоре 8→1. При выборе других переменных x_i в качестве адресных получится другая схема. Предпочтение следует отдать той схеме, на информационные входы которой подается большее число констант 0 и 1. Выбрать оптимальную с этой точки зрения схему можно; выполнив синтез для всех возможных вариантов комбинаций адресных сигналов.

Все возможные варианты разбиения диаграмм Вейча показаны на рисунках 3.16, 3.17.

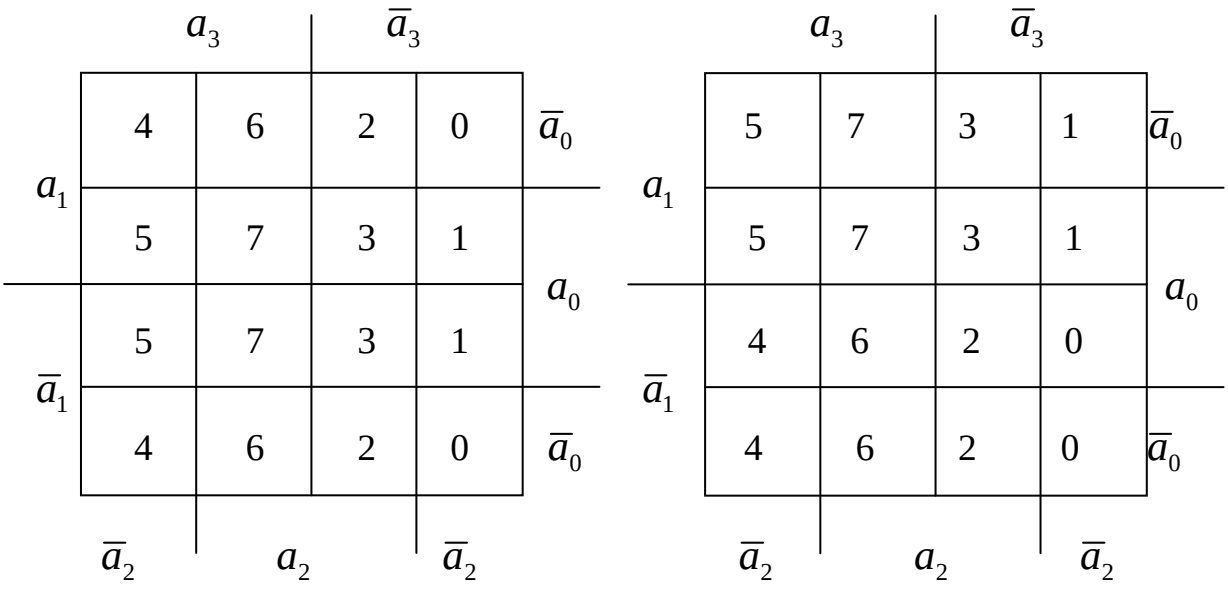


а) $i = a_2 a_1 a_0$



б) $i = a_3 a_1 a_0 j$

Рисунок 3.16



а) $i = a_3 a_2 a_0$ *б)* $i = a_3 a_2 a_1 j$

Рисунок 3.17

В таблице 3.4 указаны значения DI_i для различных комбинаций адресных сигналов, полученные на основании рисунков 3.15,*в* и рисунков 3.16, 3.17.

Таблица 3.4

Вход	i			
	$a_2 a_1 a_0$	$a_3 a_1 a_0$	$a_3 a_2 a_0$	$a_3 a_2 a_1$
DI_0	$\overline{x}_3$	1	$\overline{x}_1$	1
DI_1	1	$\overline{x}_2$	$\overline{x}_1$	0
DI_2	0	0	$\overline{x}_1$	$\overline{x}_0$
DI_3	0	0	0	0
DI_4	$\overline{x}_3$	0	0	x_0
DI_5	x_3	1	$\overline{x}_1$	0
DI_6	x_3	x_2	$\overline{x}_1$	$\overline{x}_0$
DI_7	0	0	$\overline{x}_1$	$\overline{x}_0$

Наилучший вариант схемы получается при задании адресов числами $i = a_3 a_1 a_0$, так как в этом случае на входе схемы присутствует наибольшее число констант. Функциональная схема устройства, реализующего данный вариант представлена на рисунке 3.18.

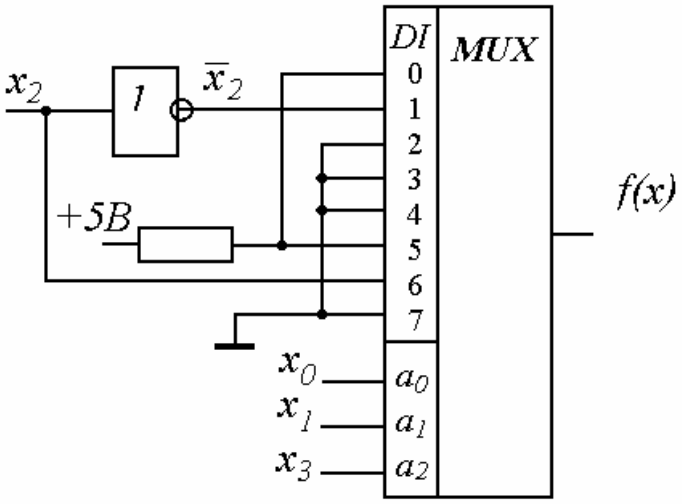


Рисунок 3.18

Из рисунка 3.15,а следует, что МДНФ функции

$$f(x) = \overline{x}_3\overline{x}_2\overline{x}_1 \vee \overline{x}_3\overline{x}_1\overline{x}_0 \vee x_3\overline{x}_1x_0 \vee x_3x_2x_1\overline{x}_0.$$

Так как внутренний дешифратор мультиплексора реализует восемь конституент единицы трех переменных, то в качестве адресных сигналов следует использовать те переменные, которые входят в МДНФ наибольшее число раз (x_3, x_1, x_0). В этом случае внутренний дешифратор мультиплексора будет нести наибольшую логическую нагрузку. Такой подход к выбору адресных сигналов позволяет исключить полный перебор всех вариантов.

3.3 Регистры

3.3.1 Общие сведения о регистрах

Регистром называется устройство, предназначенное для записи и хранения двоичных чисел.

Классификация регистров осуществляется по ряду признаков.

По способу приема и выдачи кодов чисел регистры подразделяются на параллельные и последовательные.

В параллельных регистрах информация записывается и считывается в параллельном коде, т.е. все разряды числа записываются и считываются одновременно.

В последовательных регистрах запись и считывание чисел производится в последовательном коде, т.е. разряд за разрядом, начиная со старших (или младших).

Последовательные регистры могут иметь цепи параллельной записи информации в регистр и выдачи ее из регистра. С помощью таких регистров осуществляется преобразование последовательных кодов в параллельные и наоборот.

Для сохранения считываемой из последовательного регистра информации, его выход с помощью ЛЭ соединяют со входом. В этом случае записанная в регистре информация циркулирует в нем. Такой регистр называется закольцованным.

Строятся регистры, как правило, на основе триггеров различных типов. При этом для регистрации n - разрядного двоичного слова

$$A = a_{n-1}a_{n-2}...a_1a_0,$$

используется n триггеров, каждый из которых предназначен для регистрации значения цифр строго определенного разряда.

Совокупность этих триггеров, объединенных общими цепями управления, и представляет собой регистр.

В общем виде регистр это ЦА, состояние которого оценивается совокупностью логических переменных Q_i на выходах триггеров (совокупностью состояний i-х триггеров)

$$\psi = \{Q_{n-1}, Q_{n-2}, \ldots, Q_1, Q_0\}.$$

При этом состояние каждого i - го триггера Q_i отождествляется со значением i - го разряда, хранимого в регистре числа a_i.

На рисунке 3.19 показано условное графическое обозначение двух сдвигающих регистров: 555ИР8 - последовательно-параллельный 8-разрядный регистр и 564ИР2 – два четырехразрядных сдвигающих регистра.

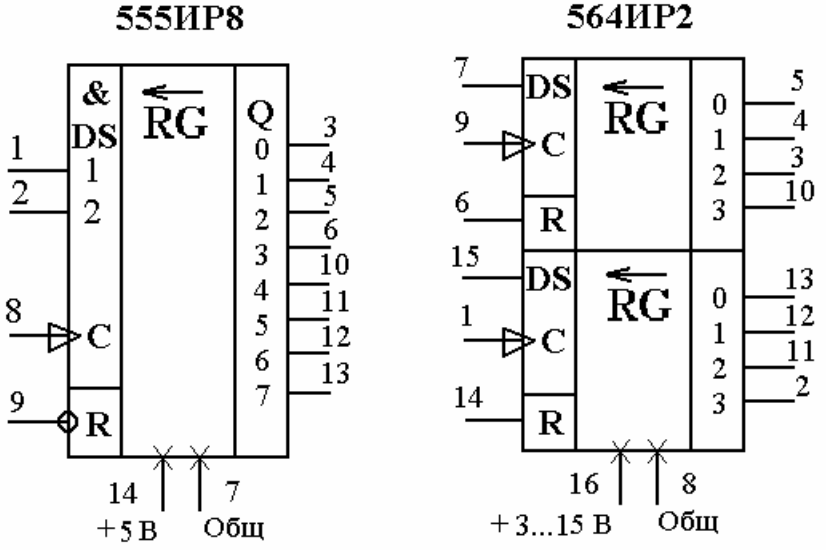


Рисунок 3.19

3.3.2 Регистры параллельного действия

В параллельных регистрах для обеспечения одновременной записи и считывания всех разрядов кода числа используются однотипные триггеры и логические схемы. Рассмотрим принцип построения таких схем.

Различают два способа записи и считывания информации: однофазный и парафазный (двухфазный). При однофазном способе записи информация о двоичном символе a_i, который надо записать в триггер i - го разряда регистра, подводится к нему по одному проводу. Двухфазный способ отличается тем, что информация к триггеру i-го разряда регистра подводится по двум проводам: по одному поступает - a_i, а по другому - $\overline{a}_i$. Учитывая способы записи и считывания, определим функциональную схему i – го разряда регистра, полагая, что запись информации в триггер осуществляется синхронно, т.е. в результате воздействия управляющего сигнала C = 1. Поэтому входными сигналами для триггера являются логические переменные a_i и C. Таблица переходов для i - го триггера в режиме приема информации соответствует таблице 3.5, а функция возбуждения триггера i - го разряда соответствует данным таблицы 3.6. Полагаем, что в качестве элемента памяти используется RS - триггер, т.е.

$$R_i = R(Q_i, a_i, C);\ S_i = S(Q_i, a_i, C).$$

Таблица 3.5

a_i	C	$Q_i(t)$	
		0	1
0	0	0	1
0	1	0	0
1	0	0	1
1	1	1	1

Таблица 3.6

$Q_i(t)$	a_i	C	R_i	S_i
0	0	0	*	0
0	0	1	*	0
0	1	0	*	0
0	1	1	0	1
1	0	0	0	*
1	0	1	1	0
1	1	0	0	*
1	1	1	0	*

Примечание. Индекс * означает то, что на соответствующий вход может поступать или нуль, или единица.

Минимизированные функции возбуждения для R_i и S_i определим с помощью диаграмм Вейча (рисунок 3.20 а,б) для неполностью определенных функций.

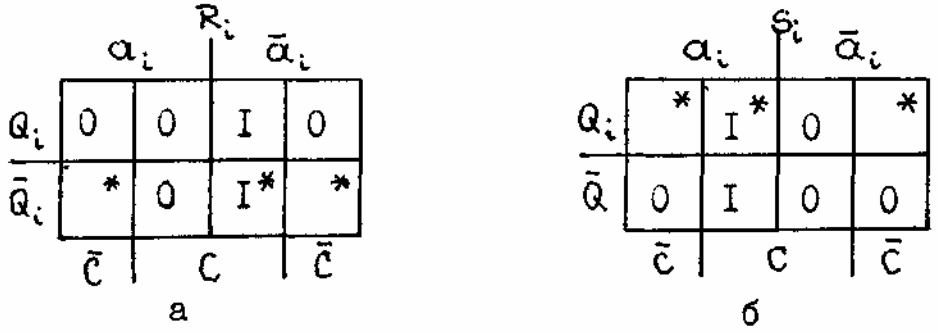


Рисунок 3.20

В результате минимизации получены следующие МДНФ: $R_i = \bar{a}_i C;\quad S_i = a_i C$.

Функция переходов i - го разряда регистра соответствует уравнению $Q_i(t+1) = a_i \left[C \vee Q_i(t) \right]|_{C=1} = a_i$.

Функциональные схемы i - го разряда регистра для двух способов записи приведены на рисунке 3.21.

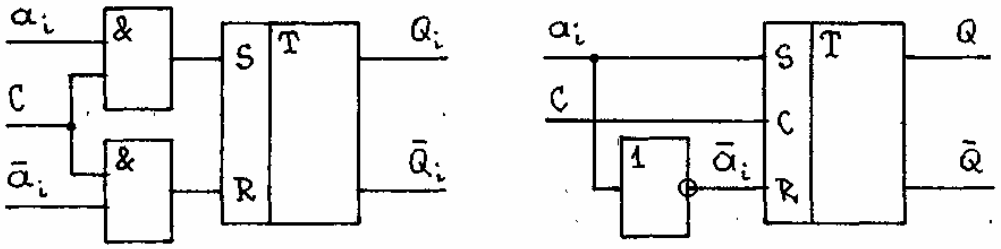


Рисунок 3.21

Объединением таких схем по входу синхронизации можно получить параллельный регистр любого числа разрядов. На рисунке 3.22 показана реализация параллельного двухразрядного регистра на универсальных JK триггерах. В данной схеме используется парафазный способ представления записываемых данных. Выводы S и R триггеров могут использоваться для начальной установки регистра в заданное состояние. При реализации параллельного регистра на дискретных триггерах выходной сигнал можно по выбору считывать как в прямом, так и инверсном виде.

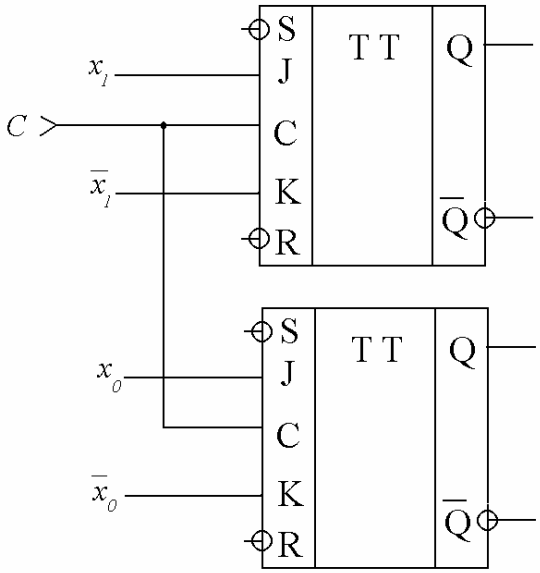


Рисунок 3.22 – Двухразрядный параллельный регистр

3.3.3 Регистры последовательного действия

В регистрах последовательного действия ввод и вывод информации осуществляется последовательно (разряд за разрядом), начиная с младшего или старшего разряда. При записи очередного разряда ранее записанная в последовательный регистр информация сдвигается, поэтому такие регистры называют еще сдвигающими регистрами. Запись или считывание одного разряда (сдвиг) выполняется под действием одного синхронизирующего импульса.

Сдвиг данных в регистре - одна из основных операций в цифровых устройствах. С ее помощью можно преобразовать параллельный код в последовательный, сдвигая по тактам двоичное число от разряда к разряду. Выполнить операции умножения и деления на число 2^n, т.к. сдвиг числа на один разряд вправо означает уменьшение его в два раза, а влево - увеличение в два раза. Таким образом, чтобы разделить или умножить число, записанное в регистре на 2^n необходимо его сдвинуть вправо или влево на *n* - разрядов.

Операция сдвига осуществляется с помощью специального последовательного соединения триггеров, составляющих последовательный (сдвигающий) регистр. Сдвигающие регистры строятся на основе синхронных RS , D, DV, JK - триггерах. В *n* - разрядном регистре применяются *n* триггеров. Наиболее приспособленными для сдвигающих регистров являются двухступенчатые триггеры, т.к. они свободны от проблемы гонок. На рисунке 3.23 показан фрагмент схемы сдвигающего регистра на JK - триггерах.

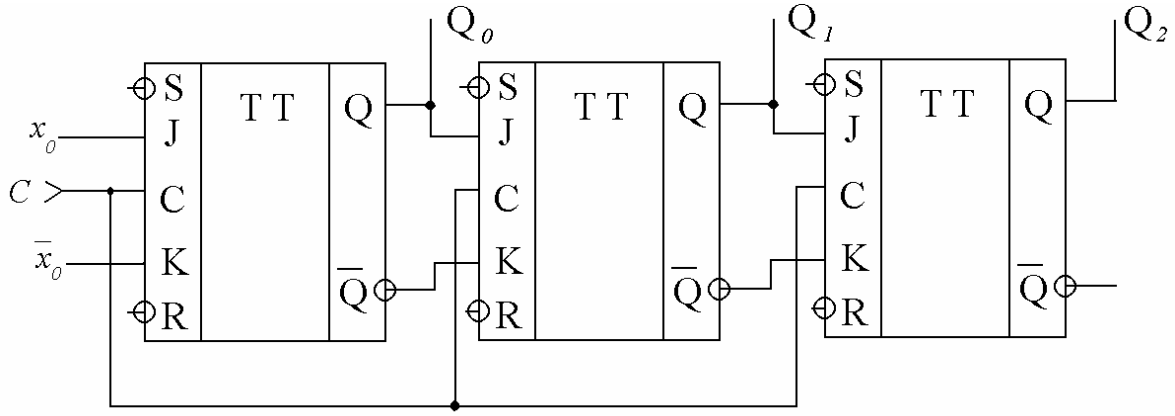


Рисунок 3.23

В общем случае *n* - разрядный сдвигающий регистр представляет собой цифровой автомат, содержащий *n* - элементов памяти, имеющий $L_\psi = 2^n$ различных состояний, которые пронумерованы числом β. При этом номер состояния сдвигающего регистра можно определить следующим образом:

а) если сдвиг информации в регистре произошел на один разряд влево, то

$$\beta(t+1) = \left[2\beta(t)\right]_{\mathrm{mod}\,2^n} + \varphi_0(t)\cdot 2^0,$$

где $\beta(t), \beta(t+1)$ - номер состояния регистра на моменты времени *t* и *t*+1 соответственно; $\varphi_0(t)$ - значение функции возбуждения на информационном входе младшего разряда в момент *t*.

Приведенное выражение отражает тот факт, что при выполнении операции сдвига влево на один разряд кроме умножения записанного в нем числа на два по модулю 2^n производится еще и запись символа a_0 в младший разряд регистра, значение которого определяется значением функции φ_0;

б) если сдвиг информации в регистре происходит на один разряд вправо, то

$$\beta(t+1) = \frac{\beta(t) - Q_0(t)}{2} + \varphi_{n-1}(t)\cdot 2^{n-1},$$

где φ_{n-1} - значение функции возбуждения триггера (*n* - I) -разряда; Q_0 - значение логической переменной на выходе триггера младшего разряда.

Таким образом, при сдвиге кода в регистре на один разряд вправо происходит не только потеря информации из младшего разряда и

деление числа на два, но также и запись в старший разряд регистра цифры $\varphi_{n-1}(t)$.

Сдвигающие регистры можно строить различным образом, зависящим от выполнения требуемой операции: сдвиг влево, сдвиг вправо, сдвиги влево и вправо. Регистры, имеющие возможность сдвигать информацию и влево, и вправо называют реверсивными. В их составе обязательно присутствует управляющая схема, которая определяет, в соответствии с командой, направление сдвига информации. Управляющая схема является схемой комбинационного типа и может быть синтезирована на основе теории ПФ для каждого конкретного случая.

3.4 Счетчики

3.4.1 Общие сведения о счетчиках

Счетчиком называется устройство, предназначенное для подсчета числа импульсов. Основными характеристиками счетчика является быстродействие и информационная емкость. Быстродействие определяется двумя параметрами:

$f_{сч}$ - максимальной частотой счета (или обратной величиной, называемой минимальным периодом счета);

$\tau_{уст}$ - временем установления, равным временному интервалу между поступлением последнего счетного импульса и моментом установления выходного кода.

При этом быстродействие счетчика часто еще оценивают разрешающим временем $T_{разр}$. Время $T_{разр}$ - это тот минимально допустимый временной интервал между двумя входными импульсами, при котором обеспечивается переход счетчика в новое состояние. Частота счета связана с $T_{разр}$ соотношением

$$f_{сч} = \frac{1}{t_и + T_{разр}} \approx \frac{1}{t_и + \tau_{уст}},$$

где $t_и$ - наименьшая длительность входного импульса.

Информационная емкость определяется модулем счета μ, равным максимальному числу импульсов, которое может сосчитать счетчик.

Счетчик представляет собой автомат Мура с μ состояниями, закодированными числовым кодом *N*, принимающим значения от 0 до

μ-1. В дальнейшем состояние счетчика будем отождествлять с кодом. Код N представляет собой результат счета. Состояние $N = 0$ называется начальным. Как правило, в счетчиках предусматривается микрооперация "Установка 0", выполняемая при подаче управляющего сигнала на установочный вход R и состоящая в переходе счетчика из любого текущего состояния в начальное.

По своему назначению счетчики делятся на суммирующие, вычитающие и реверсивные. В суммирующем счетчике единичный сигнал на входе увеличивает значение кода на единицу, а в вычитающем - уменьшает на единицу.

Реверсивный счетчик в зависимости от управляющих сигналов может работать либо как суммирующий, либо как вычитающий.

По используемой системе счисления счетчики бывают двоичные, восьмеричные, десятеричные, шестнадцатеричные и т.д. В дальнейшем будем рассматривать в основном только двоичные счетчики.

В зависимости от способа образования межразрядных связей, предназначенных для передачи сигналов (цифр) переноса от младших разрядов к старшим, различают счетчики с последовательным, параллельным и последовательно-параллельным переносом.

По степени использования сигналов синхронизации счетчики делятся на синхронные и асинхронные. Счетчик является цифровым автоматом с памятью, следовательно, для его реализации используются триггеры. Синхронные счетчики строятся на синхронных триггерах, асинхронные - на асинхронных.

Технически счетчики импульсов могут реализовываться в виде специальных интегральных микросхем, в этом случае они маркируются буквами ИЕ.

Например: 133ИЕ4.

Кроме того, счетчики могут быть реализованы на отдельных интегральных микросхемах, содержащих в своем составе регистры, триггеры, логические элементы.

3.4.2 Суммирующий счетчик

Это устройство представляет собой автомат с одним информационным (счетным) двоичным входом T_c с двоичными информационными выходами и выходом переноса P_c , которые описываются функцией переходов

$$N(t+1)=\begin{cases} N(t) \quad \text{при} \quad T_c(t)=0; \\ N(t)+1 \quad \text{при} \quad T_c(t)=0 \quad \text{и} \quad N(t)<\mu-1; \\ 0, \quad \text{при} \quad T_c(t)=1 \quad \text{и} \quad N(t)=\mu-1. \end{cases} \qquad (3.1)$$

Функция выхода переноса описывается следующим выражением:

$$P_c(t)=\begin{cases} 0 \quad \text{при} \quad T_c(t)=0 \quad \text{или} \quad N(t)<\mu-1; \\ 1 \quad \text{при} \quad T_c(t)=1 \quad \text{и} \quad N(t)=\mu-1. \end{cases} \qquad (3.2)$$

Из (3.1) и (3.2) следует, что если в некоторый момент *t* значение кода становится равным μ - 1, то дальнейший нормальный счет невозможен. В этом случае единичный сигнал на входе формирует значение сигнала P_c = 1 и переводит счетчик в начальное состояние. Единица на выходе P_c указывает на переполнение счетчика и называется единицей переноса. Наличие у счетчиков выходов P_c позволяет последовательно соединять счетчики для увеличения их емкости. Поясним принцип работы суммирующего счетчика на примере суммирующего счетчика с последовательным переносом.

Алгоритм построения такого счетчика можно определить на основе анализа правила прямого счета в двоичной системе счисления (рисунок 3.24). Согласно этому правилу, переход от числа *A* к числу *A*+1 осуществляется путем инвертирования младшего из нулей и стоящих правее него единиц.

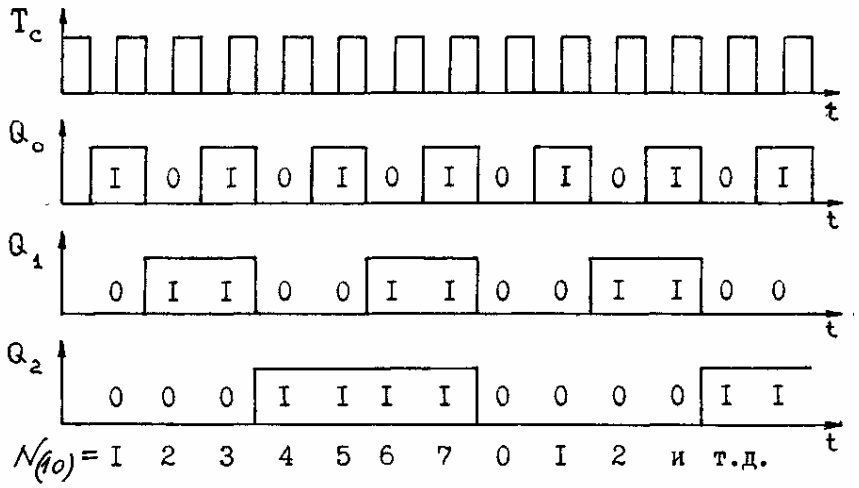


Рисунок 3.24

Для технической реализации этого правила необходимо выполнение следующих условий:

триггер младшего разряда должен переключаться при воздействии каждого единичного импульса (см. рисунок 3.24);

так как триггеры во всех разрядах счетчика должны переключаться в противоположное состояние независимо от их исходного состояния, то для счетчика целесообразнее применять *T* - триггеры;

так как триггер разряда, соответствующего младшему из нулей, должен быть последним из группы переключаемых триггеров, то переход этого триггера из состояния нуль в состояние единица не должен вызывать переключение следующего триггера, поэтому *T* - триггер каждого разряда должен быть двухступенчатым;

переключение триггера *i* - го разряда ($i \neq 0$) должно происходить только в результате переключения (i - 1) - го разряда из состояния единица в состояние нуль, поэтому для построения суммирующего счетчика целесообразнее взять двухступенчатые *T* - триггеры с инверсными входами.

Схема счетчика, удовлетворяющего данным требованиям, приведена на рисунке 3.25.

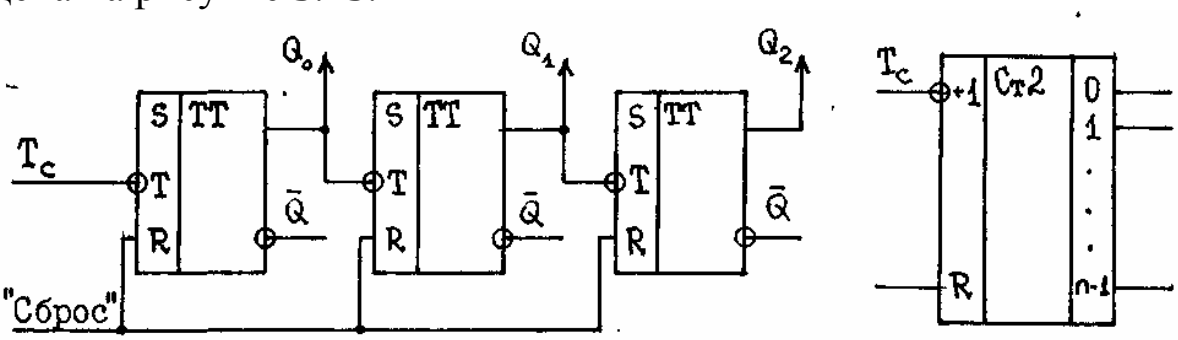


Рисунок 3.25

Особенностью этого счетчика является то, что если несколько разрядов рассматриваемого счетчика находятся в состоянии единицы, то при воздействии на его вход единичного импульса происходит последовательное переключение триггеров. Так как сигнал переключения любого триггера вырабатывается только после изменения состояния предыдущего триггера, единица как бы последовательно "переносится" от триггера младшего разряда к триггеру старшего разряда. Поэтому такой счетчик называется счетчиком с последовательным переносом.

3.4.3 Вычитающий и реверсивный счетчики

Вычитающий счетчик отличается от суммирующего видом функций переходов и выходов

$$N(t+1)=\begin{cases} N(t) \quad \text{при} \quad T_в(t)=0; \\ N(t)-1 \quad \text{при} \quad T_в(t)=1 \quad \text{и} \quad N(t)\neq 0; \\ \mu-1 \quad \text{при} \quad T_в(t)=1 \quad \text{и} \quad N(t)=0, \end{cases}$$

$$P_в(t)=\begin{cases} 1 \quad \text{при} \quad T_в(t)=0 \quad \text{или} \quad N(t)\neq 0; \\ 0 \quad \text{при} \quad T_в(t)=1 \quad \text{и} \quad N(t)=0, \end{cases}$$

где $T_в$ - двоичный вход вычитающего счетчика; $P_в$ - двоичный выход переноса.

Рассмотрим принцип работы вычитающего счетчика на примере вычитающего счетчика с последовательным переносом.

Для построения такого счетчика следует использовать правило обратного счета в двоичной системе счисления. Согласно этому правилу для перехода от числа A к числу A-1 следует инвертировать младшую из единиц и состоящие правее нее нули (рисунок 3.26).

Это означает, что триггеры, соответствующие указанным разрядам должны переключаться в противоположное состояние независимо от их исходного состояния, если триггер предшествующего разряда переходит из состояния нуль в состояние единицы.

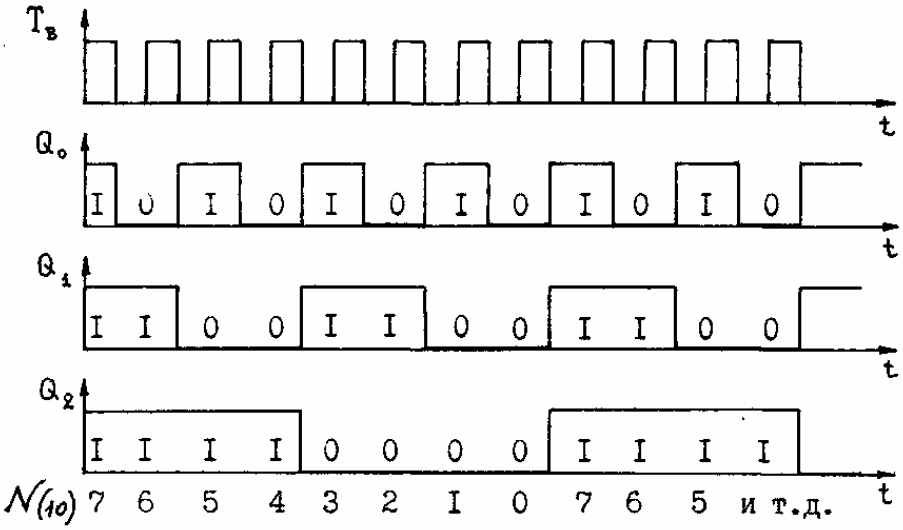


Рисунок 3.26

Поэтому при использовании в счетчике T - триггеров с инверсным входом, необходимо вход i - го триггера соединить с инверсным выходом $(i-1)$ - го триггера (рисунок 3.27).

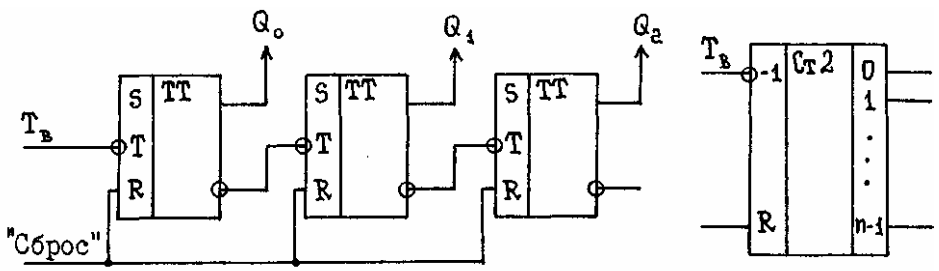


Рисунок 3.27

Таким образом, отличие суммирующего и вычитающего счетчиков (рисунки 3.25, 3.27) заключается в соединении прямого или инверсного выхода $(i-1)$ - го триггера со входом i - го триггера.

Счетчик, способный выполнять и операцию суммирования импульсов, и операцию вычитания называется реверсивным.

Управление процессом переключения осуществляется специальным сигналом Z . При этом сигнал Q_i и на входе i - го триггера определяется выражением:

$$Q_i = ZQ_{-1} \vee \overline{Z}\overline{Q}_{i-1}.$$

Если $Z = 0$, получаем $Q_i = \overline{Q}_{i-1}$, т.е. вход i - го триггера подключается к инверсному выходу $(i$ - $1)$ - го триггера. Счетчик будет работать в режиме вычитания. При $Z = 1$ имеем $Q_i = Q_{i-1}$ счетчик работает в режиме суммирования.

Схема реверсивного счетчика приведена на рисунке 3.28.

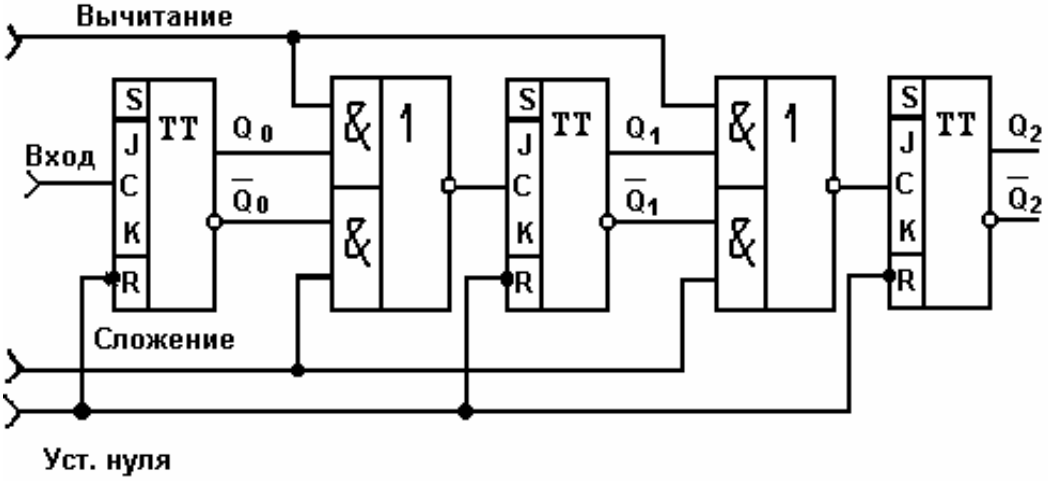


Рисунок 3.28

Реверсивный счетчик на функциональных схемах имеет два входа, обозначаемые соответственно +1 и - 1. Выбор одного из них осуществляется по значению сигнала Z. Примером реверсивного счетчика, выполненного в виде интегральной микросхемы, является 4-х разрядный двоичный счетчик 155ЗИЕI7.

3.4.4 Счетчики с заданным коэффициентом пересчета

Коэффициентом пересчета называют максимальное число, которое можно записать в этот счетчик. Все ранее рассмотренные счетчики имели коэффициент пересчета, равный $M = 2^n$, где n - количество разрядов в счетчике.

В цифровой технике часто возникает необходимость реализовывать счетчик с заданным коэффициентом пересчета, отличным от 2^n, например: M = 10.

Счетчики с произвольным коэффициентом пересчета называются пересчетными устройствами, а счетчики с коэффициентом пересчета M = 10 - декадными. Методику построения счетчика с заданным коэффициентом пересчета рассмотрим на примере декадного счетчика.

Принцип построения декадного счетчика заключается в исключении у счетчика лишних состояний r, число которых $r = 2^n - M$.

Потребное количество разрядов счетчика должно удовлетворять следующему условию

$$2^n \geq M \geq 2^{n-1}.$$

Для M = 10 это условие выполняется при n = 4, т.е. для счетчика натуральных десятичных чисел необходим четырехразрядный двоичный счетчик с особой логикой работы. Схемная логика его отличается тем, что сброс в нуль происходит на каждом десятом входном сигнале. Ниже приведена таблица истинности для такого счетчика (таблица 3.7).

Анализ данных таблицы 3.7 показывает, что нужна логическая схема, которая по приходу десятого импульса переводит счетчик в нулевое (начальное) состояние, переключательная функция для нее

$$R = Y_A = Q_3 \cdot \overline{Q}_2 \cdot Q_1 \cdot \overline{Q}_0 .$$

Таблица 3.7

$N_{сч}$	Q_3	Q_2	Q_1	Q_0	R	Y_A
	2^3	2^2	2^1	2^0		
0	0	0	0	0	0	0
1	0	0	0	1	0	0
2	0	0	1	0	0	0
3	0	0	1	1	0	0
4	0	1	0	0	0	0
5	0	1	0	1	0	0
6	0	1	1	0	0	0
7	0	1	1	1	0	0
8	1	0	0	0	0	0
9	1	0	0	1	0	0
10	1	0	1	0	1	1

Функциональная схема десятичного счетчика, удовлетворяющего таблице истинности, показана на рисунке 3.29.

Если все оставшиеся наборы двоичных кодов (11,12,13,14 и 15) принять равными 1, то выражение может быть приведено к виду:

$$R = Y_A = Q_3 \cdot Q_1,$$

а его реализация предполагает коммутацию прямых выходов тех триггеров, которые достигают единичного значения к моменту прихода десятого импульса.

Таким образом, построение счетчика с заданным коэффициентом пересчета производится в соответствии с общей методикой синтеза цифровых автоматов, на основе анализа логики его работы, описанной в виде таблицы истинности.

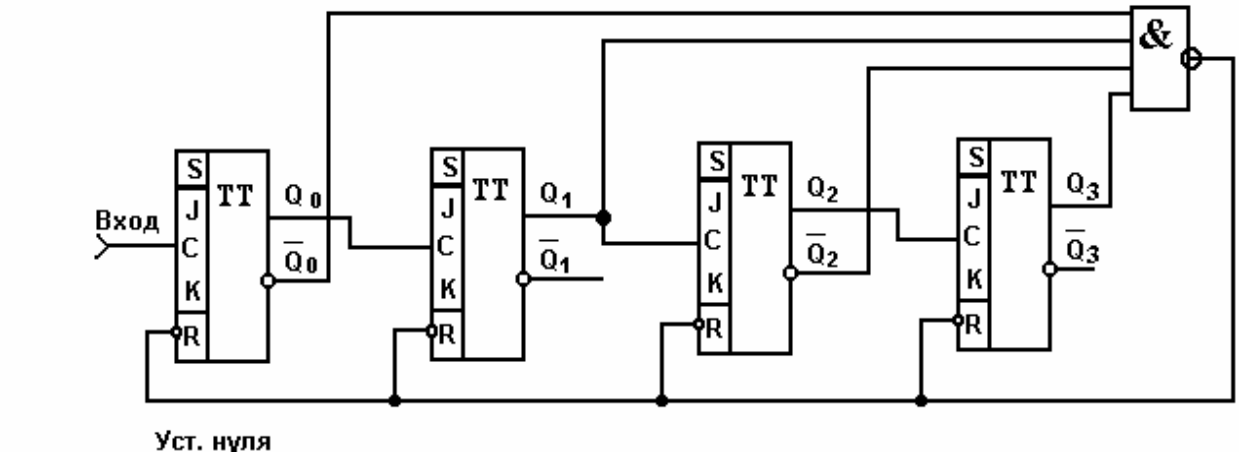


Рисунок 3.29

3.5 Комбинационные сумматоры

Наиболее распространенной операцией в устройствах обработки цифровой информации является суммирование, т.е. арифметическое сложение двух чисел, имеющих одинаковое число разрядов. Операция сложения выполняется в соответствии с правилами арифметики, используемой в данном конкретном случае системы счисления. Многоразрядные сумматоры строятся на основе одноразрядных. При этом различают многоразрядные сумматоры последовательного, параллельного и последовательно-параллельного действия.

3.5.1 Одноразрядный комбинационный сумматор

Одноразрядным комбинационным сумматором называют операционный элемент, выполняющий микрооперацию суммирования двух или нескольких одноразрядных чисел. В результате сложения получается сумма S и цифра переноса P в старший разряд. Закон функционирования одноразрядного комбинационного сумматора определяется таблицей истинности (таблица 3.8), в которой роль логических переменных играют числа a, b, S, P_i, P_{i+1}. При этом P_i это цифра переноса из предыдущего разряда, а P_{i+1} - перенос в следующий разряд.

В соответствии с таблицей 3.8 СДНФ для суммы S и переноса в последующий разряд P_{i+1} будут удовлетворять выражениям:

$$S = \overline{P}_i \overline{a} b \vee \overline{P}_i a \overline{b} \vee P_i \overline{a} \overline{b} \vee P_i a b, \qquad (3.3)$$

$$P_{i+1} = \overline{P}_i a b \vee P_i a \overline{b} \vee P_i \overline{a} b \vee P_i a b. \qquad (3.4)$$

Выполнив минимизацию выражения, получаем

$$P_{i+1} = a + b \vee P_i b \vee P_i a. \qquad (3.5)$$

Используя правила алгебры логики, учитывая (3.5), путем преобразований (3.3) получаем выражение для S в виде

$$S = \left(P_i \vee a \vee b\right) \cdot \overline{P}_{i+1} \vee a_i b_i P_i. \qquad (3.6)$$

Схема одноразрядного комбинационного сумматора, удовлетворяющая (3.5) и (3.6), показана на рисунке 3.30, а ее функциональное обозначение - на рисунке 3.31.

Таблица 3.8

α	P_i	a	b	P_{i+1}	S
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	1
3	0	1	1	1	0
4	1	0	0	0	1
5	1	0	1	1	0
6	1	1	0	1	0
7	1	1	1	1	1

Структуры, изображенные на рисунках 3.30 и 3.31, соответствуют так называемому полному сумматору.

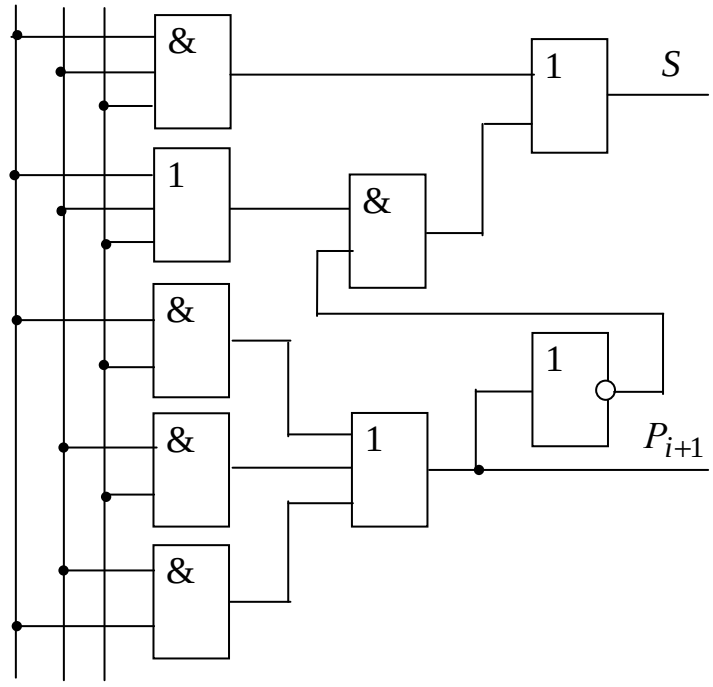


Рисунок 3.30

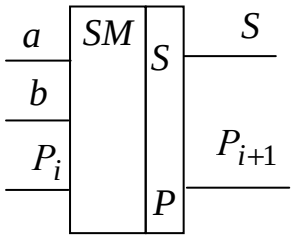


Рисунок 3.31

Часто при сложении чисел младших разрядов двоичного кода перенос $P_i = 0$, тогда переключательные функции их существенно упрощаются и удовлетворяют условиям:

$$S = \overline{a}b \vee a\overline{b}\,, \tag{3.7}$$

$$P_{i+1} = ab. \tag{3.8}$$

Устройства, ПФ которых удовлетворяют (3.7) и (3.8), называют полусумматорами, их структура и функциональное обозначение представлены на рисунках 3.32 и 3.33, соответственно.

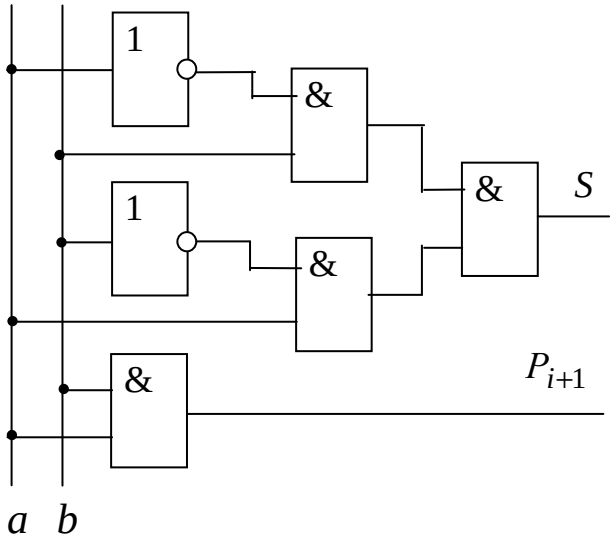


Рисунок 3.32

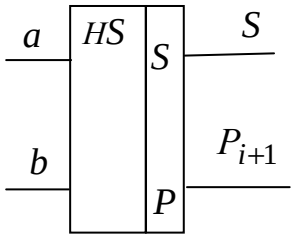


Рисунок 3.33

Интегральные схемы полных сумматоров обозначаются буквами "ИМ", а полусумматоров "ИЛ".

Например: 155ИМ1 - полный сумматор;
155ИЛ7 - полусумматор.

Кроме параметров, определяющих условия эксплуатации сумматоров как микросхем, используют следующие параметры, необходимые при решении задачи синтеза электрических схем устройства обработки цифровой информации на основе сумматоров:

разрядность;

способ суммирования двоичных чисел (последовательный или параллельный);

тип входной логики сумматора по входу A и по входу B;

наличие и организация входов управления;

способ организации цепи переноса;

тип выходной логики;

быстродействие (время задержки сигнала и переноса);

потребляемый ток питания;

особенности монтажа при увеличении разрядности сумматора.

3.5.2 Многоразрядный комбинационный сумматор последовательного действия

Многоразрядный комбинационный сумматор последовательного действия представляет собой структуру, состоящую из двух сдвигающих регистров RGA, RGB, одноразрядного сумматора и схем управления (рисунок 3.34).

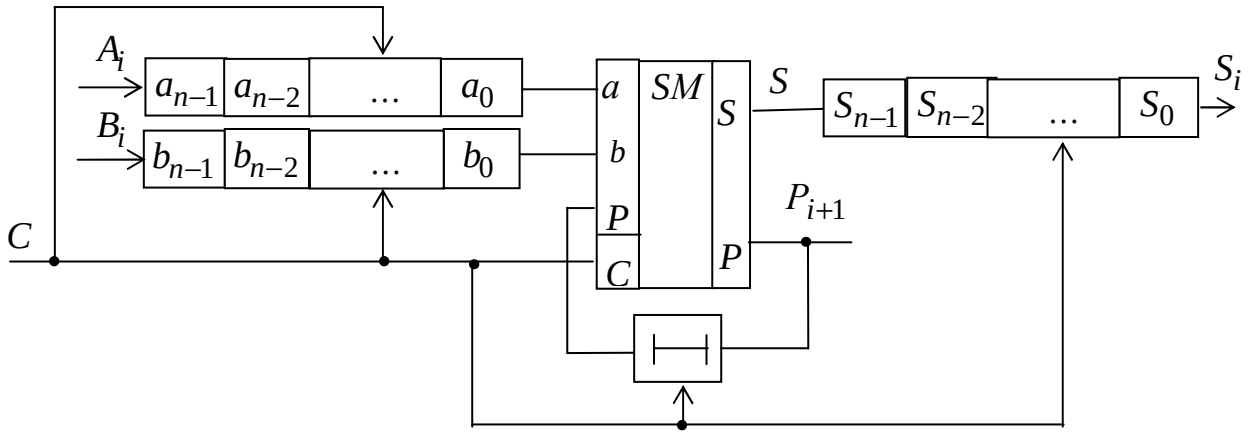


Рисунок 3.34

Порядок функционирования последовательного сумматора следующий. Перед началом суммирования сдвигающие регистры RGA, RGB, и триггер устанавливаются в нулевое состояние. Затем производится запись чисел A и B в регистры RGA, RGB соответственно. После этого производится суммирование чисел A и B поразрядно, начиная с младшего разряда. В первом такте суммирования на один вход сумматора поступает цифра $P_0 = 0$, а на другие два входа - цифры a_0 и b_0 с выходов регистров. На выходе сумматора формируется цифра младшего разряда суммы S_0 и цифра переноса P_1. Цифра переноса подается на вход триггера для задержки на один такт, а цифра S_0 поступает через схему "ИЛИ" на вход старшего разряда RGA. После окончания действия первого импульса цифра переноса P_1 появляется на выходе триггера, одновременно происходит сдвиг содержимого регистров на один разряд вправо и в освободившийся старший разряд RGA записывается S_0. На втором такте на вход сумматора будут поступать цифры a_1, b_1, P_1, а на его выходе будут образовываться

цифры S_1, P_2. Далее обработка информации будет производиться так же, как в первом цикле.

Таким образом, весь цикл суммирования занимает n тактов, результат суммирования записывается в RGA. Результат сложения будет верным, если число разрядов суммы не превышает разрядность регистра RGA. Если же указанное условие не выполняется, то будет иметь место искажения результата, вызванного переполнением разрядной сетки.

Результат сложения

$$S = A + B. \tag{3.9}$$

может быть записан в специальный регистр суммы.

Быстродействие последовательного сумматора определяется временем сложения

$$T_{сл} = n\left(T_{\text{т}} + T_S\right), \tag{3.10}$$

где n - число разрядов в регистрах; $T_{\text{т}}$ - время сдвига кода в регистрах; T_S - время образования суммы в сумматоре.

3.5.3 Многоразрядный параллельный сумматор

Функциональная схема трехразрядного комбинационного сумматора параллельного действия показана на рисунке 3.35. В его состав входят 3 одноразрядных сумматоров (в общем случае число одноразрядных сумматоров равно числу n разрядов суммируемых слов). Дополнительно используются три регистра: регистр числа A (RGA), регистр числа B (RGB) и регистр суммы (RGS). Для выполнения суммирования цифры всех разрядов чисел A и B подаются на входы сумматоров одновременно. С этого момента начинается процесс сложения, продолжающийся до завершения всех переносов и образования суммы в старшем разряде. Поскольку значение суммы цифр и переноса в i-м разряде зависят от результатов вычисления переноса в (i-1)-м разряде, значения цифр суммы на выходах одноразрядных сумматоров образуются последовательно. При неблагоприятных сочетаниях цифра переноса может распространяться через все разряды. Поэтому наибольшее время сложения двух n разрядных чисел в параллельном сумматоре определяется как

$$T_{сл\ наиб} = (n-1)T_{зр} + T_S \approx nT_{зр}, \tag{3.11}$$

где $T_{зр}$ - время задержки цифры переноса P.

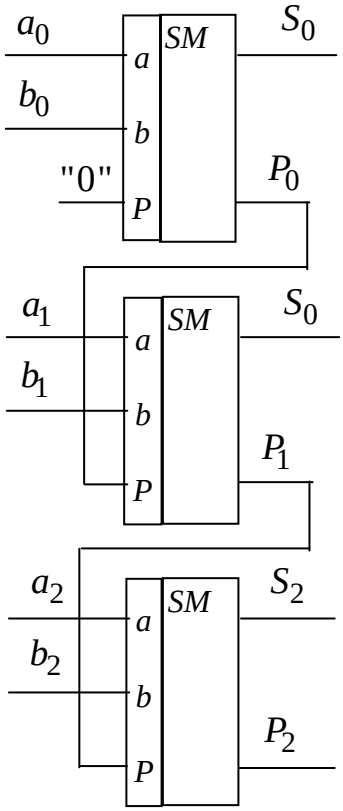


Рисунок 3.35

Анализ выражений (3.10) и (3.11) показывает, что параллельный сумматор по сравнению с последовательным выполняет сложение быстрее в η -раз, где

$$\eta = \frac{T_{\text{т}} + T_S}{T_{зр}}. \qquad (3.12)$$

Поэтому для достижения существенного выигрыша в быстродействии параллельного сумматора следует уменьшить время $T_{зр}$, то есть применять особые способы ускорения переноса.

Быстродействие параллельного сумматора обеспечивается в основном за счет увеличения аппаратурных затрат. Для снижения этих аппаратурных затрат при сохранении высокого быстродействия используют последовательно-параллельный способ суммирования. Идея этого способа основана на том, что в параллельных сумматорах в процессе последовательного формирования цифры переноса сумматоры младших разрядов простаивают, причем время простоя увеличивается с уменьшением номера разряда, поэтому имеется возможность использования освободившихся сумматоров для определения суммы цифр последующих разрядов.

3.5.4 Способы ускорения переноса в сумматорах

Быстродействие сумматоров в основном зависит от времени переноса цифры в старшие разряды (см. выражение 3.11). Поэтому повышение быстродействия сумматоров в основном может быть обеспечено за счет его уменьшения.

Для повышения скорости работы параллельных сумматоров применяют различные приемы, сокращающие время прохождения сигнала переноса через группу одноразрядных сумматоров. Значение суммы S и переноса P_i в i-м разряде, задаваемые выражениями (3.3) и (3.4), можно записать в виде

$$S_i = \left(\overline{a}_i b_i \vee a_i \overline{b}_i\right)\overline{P}_{i-1} \vee \left(\overline{a}_i \overline{b}_i \vee a_i b_i\right)P_{i-1} = R_i \overline{P}_{i-1} \vee \overline{R}_i P_{i-1}. \qquad (3.13)$$

$$P_i = a_i b_i \vee \left(\overline{a}_i b_i \vee a_i \overline{b}_i\right)P_{i-1} = C_i \vee R_i P_{i-1}, \qquad (3.14)$$

где

$$\begin{cases} R_i = \overline{a}_i b_i \vee a_i \overline{b}_i, \\ C_i = a_i b_i. \end{cases} \qquad (3.15)$$

Из выражения (3.14) следует, что появление единицы переноса в i-м разряде вызывается двумя причинами:

во-первых, перенос может возникнуть внутри одноразрядного сумматора при $a_i = b_i = 1$;

во-вторых, перенос может быть выработан в связи с возникновением единицы переноса в предыдущем разряде (сквозной перенос).

Переносы первого типа образуются во всех разрядах одновременно с поступлением на входы сумматора слагаемых. Переносы второго типа формируются последовательно от младших разрядов к старшим. Основная задача заключается в уменьшении времени распространения сигнала сквозного переноса.

Эта задача решается путем создания схем, где в формировании переноса i -го разряда участвует перенос из $(i-k)$-го разряда. Для этого в старшем (i -м) разряде необходимо анализировать не только цифры a_i и b_i, но и цифры младших разрядов $a_{i-1}, b_{i-1}, a_{i-2}, b_{i-2}, ..., a_{i-k}, b_{i-k}$. Схемы, построенные для обработки этих цифр, должны срабатывать до поступления единицы переноса из младших разрядов и тем самым ускорять распространение сквозного переноса. То есть, используя (3.14), можно записать, что

$$
\begin{aligned}
&P_1 = C_1 \vee R_i P_0,\\
&P_2 = C_2 \vee R_2 P_1 = C_2 \vee R_2 C_1 \vee R_2 R_1 P_0,\\
&P_3 = C_3 \vee R_3 P_2 = C_3 \vee R_3 C_2 \vee R_3 R_2 C_1 \vee R_3 R_2 R_1 P_0,\\
&\ldots\ldots\ldots\ldots\\
&P_i = C_i \vee R_i P_{i-1} = C_i \vee R_i C_{i-1} \vee R_i R_{i-1} C_{i-2} \vee \ldots\\
&\quad \ldots \vee R_i R_{i-1} \ldots R_2 C_1 \vee R_i R_{i-1} \ldots R_2 R_1 P_0.
\end{aligned}
\qquad (3.16)
$$

Сумматоры, схема которых реализует (3.16) называют сумматорами с параллельным переносом.

Раздел 2. ПРИНЦИПЫ ПОСТРОЕНИЯ И ФУНКЦИОНИРОВАНИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

Глава 4. ОБЩИЕ СВЕДЕНИЯ О НАЗНАЧЕНИИ И ПРИНЦИПАХ ФУНКЦИОНИРОВАНИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

4.1 Назначение, характеристики и классификация БВМ

4.1.1 Роль и место БВМ в составе авиационного радиоэлектронного комплекса

Современный летательный аппарат представляет собой комплекс технических средств, функционально-связанных между собой в процессе выполнения поставленных перед ним задач. Успешное выполнение каждым ЛА (истребителем) своих задач зависит от летно-технических и тактико-технических данных самолета, опыта летчика, а также умелых действий наземных служб, в том числе и специалистов радиоэлектронного оборудования (РЭО). Поэтому к авиационному комплексу следует относить сложную человеко-машинную систему, состоящую из ЛА совместно с системой наземных технических средств с управляющими ими людьми.

Выполнение любой задачи состоит из сбора информации об обстановке; обработке данной информации по алгоритмам, соответствующим состоянию объекта, среды и условиями применения ЛА; выработки сигналов управления и собственно процесса управления.

В соответствии с этим определяется необходимый состав основных средств авиационного комплекса:

информационные средства – источники первичной информации;

линии передачи информации;

средства обработки информации, принятия решений и формирования команд управления;

исполнительных средств (объекта управления).

Центральное место занимают средства обработки информации, принятия решений и формирования команд управления. Эти средства представляют собой совокупность вычислительных устройств различного функционального назначения, не связанных или связанных

между собой и составляющих вычислительную систему (ВС).

Таким образом, под ВС будем понимать совокупность взаимосвязанных и согласованно действующих аппаратно-программных средств передачи хранения и переработки цифровой информации, предназначенных для преобразования входных данных в выходные в соответствии с заданными целями.

Основным средством уничтожения воздушных целей являются сверхзвуковые высокоманевренные истребители, главным видом боевых действий которых считается воздушный бой. Современный воздушный бой характеризуется высокими скоростями, интенсивным маневрированием, большой информационной насыщенностью, что в совокупности обуславливает высокие психофизические нагрузки, воздействующие на экипаж. В этих условиях, несмотря на совершенствование качества подготовки летчиков, они не всегда обеспечивают безопасность пилотирования и не в полной мере используют потенциальные возможности самолетов.

Выходом из этого положения является автоматизация обработки информации и управления истребителем с помощью БВМ. Обобщенная структурная схема такой автоматизированной системы представлена на рисунке 4.1.

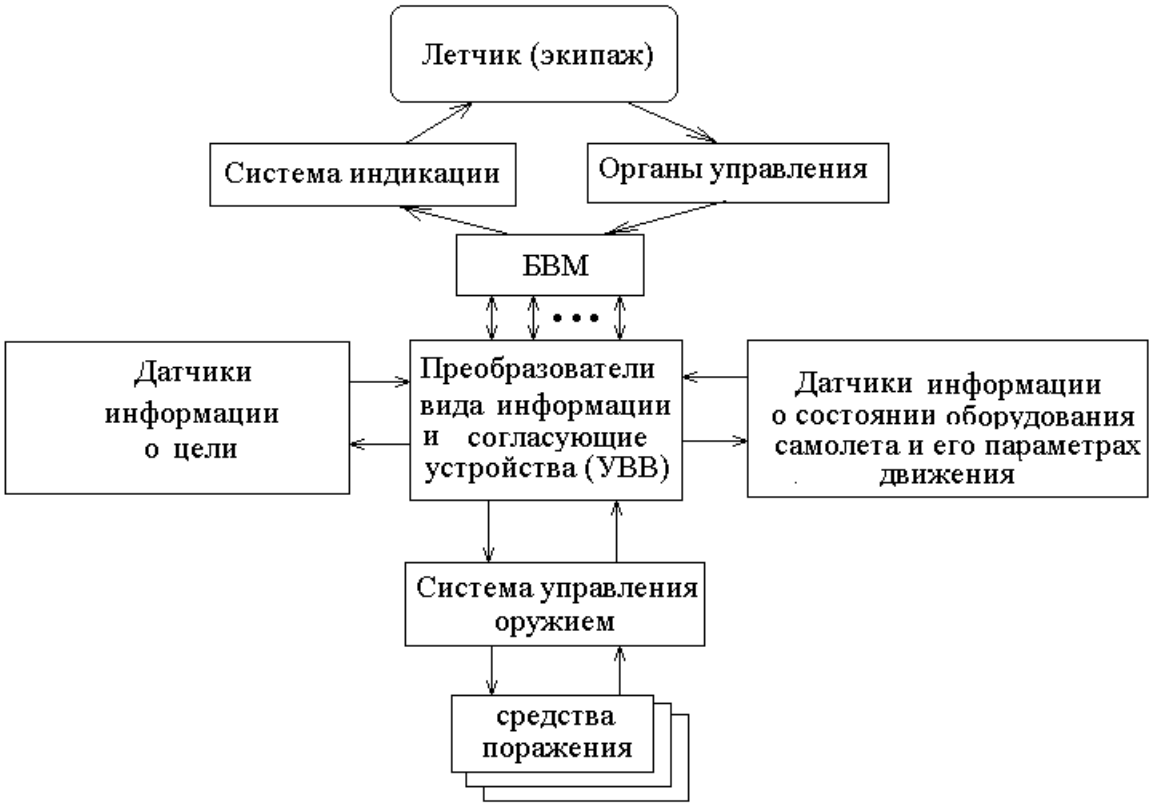


Рисунок 4.1

В состав структурной схемы входят датчики информации о цели, о состоянии оборудования истребителя и параметрах его движения. В общем случае это могут быть как отдельные устройства, например, датчик угла атаки, так и целые системы. Примером последних может быть бортовая РЛС. Так как не всегда форма представления информации является приемлемой для обработки цифровой ЭВМ, т.е. не имеет вид двоичных кодов, то для согласования вида информации и скорости ее поступления служит устройство ввода-вывода (УВВ). В некоторых вариантах конструктивного исполнения УВВ может входить непосредственно в состав БВМ.

В общем случае боевая задача истребителя заключается в поражении воздушного объекта. Для решения этой задачи необходимо обнаружить объект, распознать его как цель, определить параметры, необходимые для подготовки средств поражения (СП), и организовать применение одного или нескольких СП. Для всех перечисленных этапов в составе математического обеспечения БВМ есть соответствующие программы. Информация, полученная в результате работы этих программ, поступает на систему индикации летчику, в систему управления оружием и в датчики для изменения режимов их функционирования.

Летчик, с помощью соответствующих органов управления, задает необходимые режимы работы всех систем радиоэлектронного комплекса (РЭК), а также управляет вычислительным процессом в БВМ.

Характерными особенностями работы БВМ в контуре управления истребителем являются: сопряжение БВМ с аппаратурой иного физического характера; многократное повторение алгоритма; реальный масштаб времени; ограничение реализации алгоритмов во времени; повышенная надежность.

Первая характерная особенность состоит в том, что измеряемые датчиками физические величины по своей природе являются непрерывными, вследствие чего для ввода их в БВМ необходимо преобразование их в цифровую форму. Такое преобразование может осуществляться непосредственно датчиками или преобразователями.

Как известно, большинство исполнительных устройств работает от непрерывных сигналов. Поэтому дискретные результаты из БВМ должны быть преобразованы в аналоговые величины для последующего их использования. Различный физический характер информации приводит к значительному усложнению всей системы, увеличению ее

массы, габаритов и стоимости. Кроме того, создание преобразователей, обладающих высокой точностью, представляет определенную проблему.

Вторая особенность работы БВМ в контуре управления истребителем заключается в том, что алгоритмы решаемых задач, описанные на машинном языке, не изменяются в процессе всего периода эксплуатации ЛА и должны храниться в постоянном запоминающем устройстве. Исполнение любого алгоритма происходит путем многократного его повторения. При каждом повторении алгоритма меняются лишь исходные данные. Частота повторения того или иного алгоритма зависит от скорости изменения исходных данных, допустимой дискретности выдачи результатов в соответствующие подсистемы РЭК и условий применения.

Третья особенность состоит в реализации реального масштаба времени. Эта особенность заключается в том, что необходимо обеспечить при дискретном характере выходной информации непрерывность управления подсистемами РЭК при сохранении требуемой точности решения задачи. При этом получение результата вычислений должно произойти не позднее, чем ощутимо изменятся (устареют) исходные для работы алгоритма данные.

Четвертая особенность, заключающаяся в представлении требуемого интервала времени для реализации каждого алгоритма, определяет цикл работы БВМ и подсистем РЭК. Ограничение по времени осуществляется из соображения точности решения задач и непрерывности управления.

Пятая особенность подчеркивает важность такой характеристики, как надежность. В данном случае понимается не только техническая надежность, определяемая надежностью элементов и технологией изготовления БВМ, но и надежность программного обеспечения. Надежность последнего определяется следующей совокупностью факторов: выбором устойчивых методов решения задач; эффективностью программной защиты алгоритмов; степенью развитости программных методов контроля правильности реализации алгоритмов и т.д. Перечисленные особенности работы БВМ, круг решаемых ее задач и роль БВМ в составе радиоэлектронного комплекса предъявляют специфические требования к основным характеристикам вычислительной системы и составу ее программного и аппаратурного обеспечения.

4.1.2 Классификация БВМ

В зависимости от формы представления информации вычислительные машины делятся на два класса: аналоговые и цифровые (дискретные). Первые БВМ представляли собой аналоговые вычислители (решающие устройства). В настоящее время все более широкое применение в авиационном РЭО (в системах связи, радиолокации, при обработке речевых сигналов, изображений и др.) находят цифровые ЭВМ, имеющие ряд преимуществ по сравнению с аналоговыми: более высокая надежность; стабильность параметров при воздействии дестабилизирующих факторов; высокая точность обработки информации; значительное сокращение трудоемкости и упрощение операций регулировки и настройки; возможность создания микросхем с очень высокой степенью интеграции. Тем не менее, необходимо рассмотреть и принципы построения аналоговых вычислительных машин, так как они имеют некоторые положительные характеристики.

4.2 Общие сведения об аналоговых вычислительных машинах

В природе существует большое количество процессов имеющих разную физическую природу, но описываемых однотипными дифференциальными уравнениями. Такие аналогии можно использовать для выполнения моделирования одних процессов другими, более доступными и управляемыми. Такой принцип функционирования используется в аналоговых БВМ.

Аналоговые вычислительные машины (АВМ) удобны для воспроизведения решений дифференциальных уравнений, например уравнений динамики полета самолета, и для моделирования различных систем регулирования и управления, работа которых описывается дифференциальными уравнениями.

Машинные переменные, с которыми оперируют электронные АВМ, представляют собой непрерывно изменяющиеся во времени напряжения и токи. Им соответствуют физические или математические переменные исследуемого процесса или решаемой задачи. Соотношения между физическими и машинными переменными определяют масштабные коэффициенты:

$$U_y = m_y y; \quad t_m = m_t t,$$

где U_y, t_m; y, t; m_y, m_t, соответственно, машинные переменные, физические переменные, масштабные коэффициенты.

Коэффициенты могут быть вычислены по формуле:

$$m_y = \frac{U_{\max}}{y_{\max}},$$

где через $U_{\max}$ и $y_{\max}$ обозначены $\max|U_y|$ и $\max|y|$ соответственно.

Коэффициент m_t выбирают исходя из назначения АВМ или из требований к устройству регистрации напряжений. Длительные физические процессы, например изменение климатических условий на Земле за последние тысячелетия, целесообразно моделировать с ускорением ($t_m < t$, откуда $m_t < 1$), а быстропротекающие процессы, например процессы в ядерных реакторах, - с замедлением ($t_m > t$, откуда $m_t > 1$). Для управляющих АВМ реальных физических систем, например боевого самолета, должен быть обеспечен режим реального масштаба времени ($t_m = t$, т.е. $m_t = 1$), в противном случае управляющие воздействия от АВМ на объект управления будут поступать либо с опозданием, либо с опережением и цель управления не будет достигнута.

Первые АВМ являлись механическими устройствами и были разработаны в конце 19 начале 20 веков (братья Томпсон, 1876 год и академик Крылов 1903 год). АВМ на электронных лампах появились в 40-х годах прошлого столетия. Затем был освоен выпуск АВМ на полупроводниковых приборах, а с 70-х годов и на интегральных микросхемах.

Основу АВМ составляют операционные (решающие) блоки (ОБ), выполненные на базе решающих усилителей - дифференциальных усилителей постоянного тока с большим коэффициентом усиления. Каждый блок выполняет одну математическую операцию, блоки соединяют между собой в соответствии с аналитической зависимостью решаемой задачи. Параметры решаемой задачи как функции времени можно снимать с выходов всех блоков одновременно.

Таким образом, структура АВМ непостоянна и определяется решаемой задачей. АВМ – неалгоритмическая система, решающая задачу на множестве ОБ одновременно, а не пооперационно и последовательно во времени.

В принципе АВМ может быть реализована на различных физических переменных: пневматических, гидравлических, электромеханических, электронных и т.д. Достоинством электронных ЭВМ является их высокая надежность и точность, легкость сопряжения блоков между собой, удобство управления и регистрации решений.

По назначению АВМ делятся на специализированные АВМ, ориентированные на решение нескольких или даже одной задачи, и АВМ общего назначения, ориентированные на решение широкого круга задач. Специализированные АВМ чаще всего используют в качестве управляющих. Такими, например, являлись АВМ коллиматорных прицелов самолетов МиГ 23, Су 15 и т.д. В управляющих АВМ соединение решающих блоков фиксировано. АВМ общего назначения и сейчас используют для учебных и научно-исследовательских целей. В таких АВМ для удобства коммутации блоков между собой применяется коммутационное (наборное) поле.

Системы дифференциальных уравнений содержат в основном операции умножения на константу, сложения, интегрирования и дифференцирования. Реализация этих операций обеспечивается прецизионным операционным усилителем с различными вариантами обратных связей и внешних элементов. На рисунке 4.2 представлены схемы операционных блоков АВМ, выполняющих базовые операции.

Достоинством АВМ является высокое быстродействие, достаточно высокая точность, возможность построения сравнительно недорогих полунатурных моделей без дорогостоящих многоканальных аналого-цифровых и цифроаналоговых преобразователей. По сравнению с цифровыми аналоговые ЭВМ не универсальны, их возможности ограничены количеством и типом операционных и функциональных блоков, а точность ограничена точностью электрических параметров этих блоков. Кроме того, операционные блоки подвержены влиянию внешних факторов, таких, например, как нагрев. Поэтому цифровые ЭВМ находят в настоящее время более широкое применение. В основу построения ЦВМ положен принцип программного управления. Рассмотрим основные положения этого принципа.

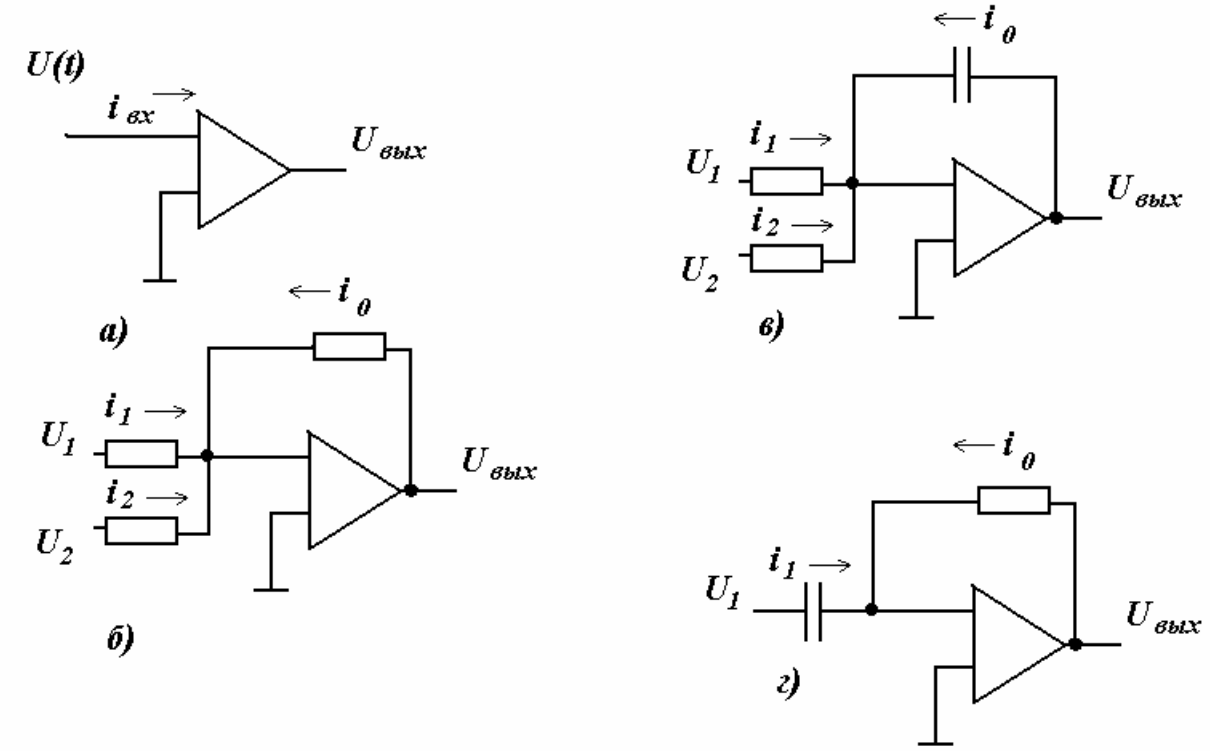


Рисунок 4.2 – Операционные блоки АВМ.
а- масштабирующий усилитель; *б*- масштабирующий сумматор; *в*- интегросумматор; *г*- блок дифференцирования

4.3 Состав и принцип работы цифровой БВМ

4.3.1 Принцип программного управления БЦВМ

Бортовая ЦВМ представляет собой автоматический преобразователь дискретной информации, построенный по принципу программного управления. Основными положениями принципа программного управления являются.

1. Информация для ЦВМ кодируется в двоичной форме. Машинной единицей информации является слово. Оно обрабатывается в машине как одно целое. Но, как и в обычной практике, машинное слово можно разбить на слоги, которые состоят из букв. При двоичном кодировании информации алфавит состоит из двух букв: 0 и 1. Место, занимаемое буквой в записи машинного слова, называют разрядом. В одном разряде содержится один бит информации. Современные БЦВМ работают со словами, длина каждого из которых (общее количество разрядов в слове) кратна 8 битам, т.е. байту.

2. В ходе вычислительного процесса в машине циркулирует информация двух видов:

числовая, относящаяся к данным решаемой задачи (исходные величины, промежуточные и конечные результаты);

управляющая, относящаяся к алгоритму решения задачи.

По форме представления машинное слово, принадлежащее к управляющей информации, нельзя отличить от числа, поскольку кодируются они одинаково. Такое единство числовой и управляющей информации позволяет обрабатывать в ЦВМ не только числа, но и управляющие слова, т.е. элементы самого алгоритма.

3. Числовая и управляющая информация в течение всего времени реализации алгоритма хранится в памяти ЭВМ. Для хранения единицы машинной информации (машинного слова или байта) служит ячейка памяти. В ЦВМ используется адресный принцип обращения к машинному слову. При этом все ячейки памяти нумеруются. Номер данной ячейки называют ее адресом. Адрес ячейки памяти выполняет роль машинного идентификатора (имени) хранящегося в ней слова. Для того чтобы занести слово в данную ячейку памяти нужно указать ее адрес. Аналогично при извлечении того или иного слова из памяти нужно указать адрес соответствующей ячейки.

4. Алгоритм решения каждой задачи представляется в машине последовательностью управляющих слов – команд. Команда задает все действия ЭВМ на одном шаге вычислительного процесса, а именно, определяет соответствующую этой команде операцию, а также участвующие в ней операнды. В соответствии с этим двоичный код команды должен содержать две части: *операционную и операндную*.

В *операционной* части записан код операции (КОП), выполняемой по данной команде.

В *операндной* части команды помещают сведения об операндах. В качестве этих сведений могут выступать адреса соответствующих операндов. Поэтому операндную часть команды называют также ее адресной частью.

Алгоритм, представленный в терминах команд, называется программой.

5. Принципиальной особенностью ЭВМ, отличающей ее от любого другого средства автоматизации вычислений, является то, что программа решения задачи помещается в память. Выполнение в ЭВМ

вычислений, предписанных алгоритмом, сводится к последовательной реализации команд. Порядок прохождения команд однозначно определяется программой. Самой первой выполняется команда, задаваемая пусковым адресом программы. Пусковой адрес (обычно это адрес самой первой команды данной программы) задается или с пульта управления ЭВМ или формируется аппаратурно.

Адрес следующей команды однозначно определяется в ходе выполнения текущей команды. Процесс вычислений продолжается до тех пор, пока не будет выполнена команда, предписывающая прекращение вычислений.

4.3.2 Структура и порядок функционирования цифровых БВМ

Бортовая ЦВМ предназначена для решения фиксированного круга задач, что определяет ее значительную специализацию. Для специализированных ЦВМ характерно наличие особенностей в составе и принципах функционирования. Но из всего многообразия структурных схем ЦВМ для рассмотрения общих вопросов примем распространенную структурную схему, изображенную на рисунке 4.3.

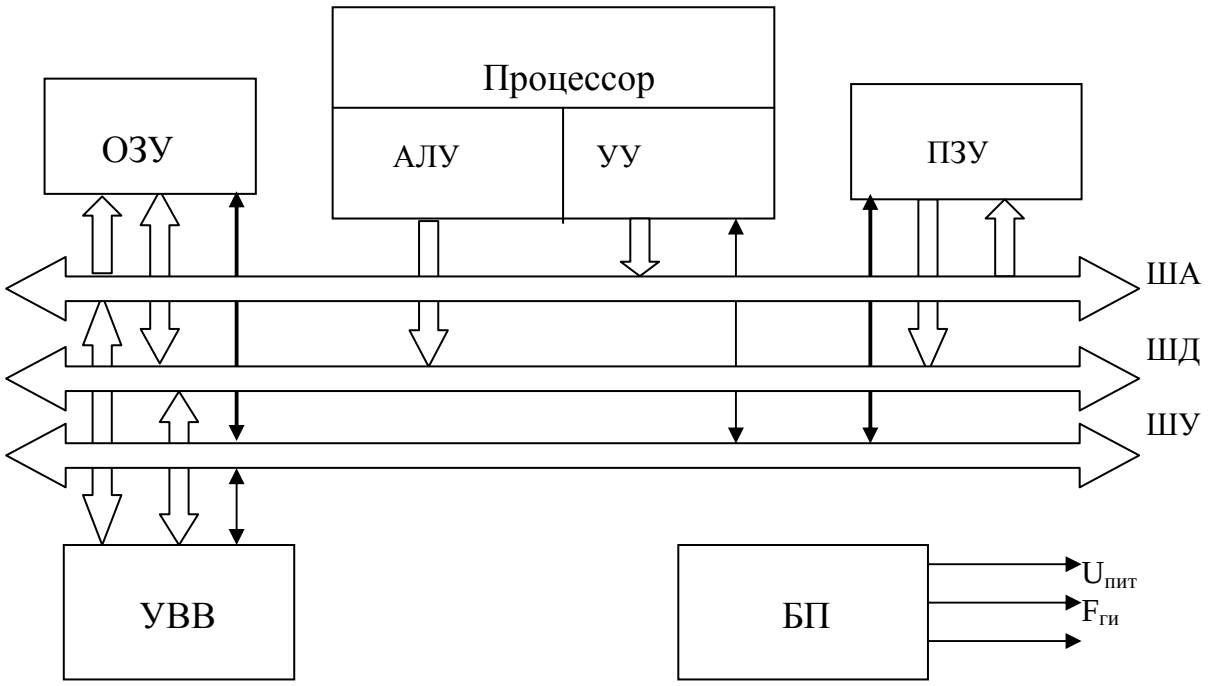


Рисунок 4.3 Обобщенная структурная схема БЦВМ

На рисунке 4.3 обозначены: УВВ – устройство ввода-вывода; ОЗУ-оперативное запоминающее устройство; ПЗУ-постоянное запоминающее устройство; АЛУ-арифметическо-логическое устройство; УУ-устройство управления; БП-блок питания; ША, ШД, ШУ – шины адреса, данных и управления.

Устройство ввода-вывода служит для связи машины с объектом управления. Конструктивно УВВ может быть оформлено отдельно от БЦВМ.

Оперативное запоминающее устройство предназначено, главным образом для хранения информации, поступающей от датчиков, а также промежуточных и конечных результатов. В ОЗУ производится как запись информации, так и считывание из него.

Постоянное запоминающее устройство предназначено для хранения неизменной в процессе эксплуатации информации. К такой информации относятся программы работы БЦВМ и константы, используемые при реализации алгоритмов. Информация записывается в ПЗУ на заводе при его изготовлении и не подлежит изменению в условиях эксплуатации. Для хранения информации, значения которой неизменны для одного или нескольких вылетов, используется долговременное запоминющее устройство, позволяющее и запись, и считывание. Изготовление такого ЗУ является наиболее трудоемким, поэтому его емкость обычно значительно меньше емкости ОЗУ и ПЗУ.

Процессор – основное устройство машины, предназначенное для автоматического выполнения последовательности операций, предусмотренных программой решения задачи.

Процессор бортовой ЦВМ функционально состоит из арифметическо-логического устройства и устройства управления. АЛУ предназначено для выполнения арифметических и логических операций над двоичными числами. Устройство управления служит для организации вычислительного процесса в соответствии с заданной программой путем принудительной координации работы всех устройств БЦВМ с помощью управляющих сигналов.

Блок питания обеспечивает формирование питающих напряжений, необходимых для работы всех устройств БЦВМ, а также для генерации серий синхронизирующих сигналов.

В радиоэлектронных комплексах современных истребителях Су-27 и МИГ-31 используются, соответственно, БЦВМ – Ц-100 (2 шт.),

БЦВМ Орбита – 20 и А-15А (К, М). В современных модификациях этих самолетов используются БЦВМ Багет 55-04 и Багет 55-06. Подробнее их состав и основные характеристики будут рассмотрены в следующих главах.

На рисунке 4.4 приведена упрощенная схема выполнения команд в ЦВМ, иллюстрирующая основные положения принципа программного управления. В соответствии с этой схемой порядок выполнения программ следующий.

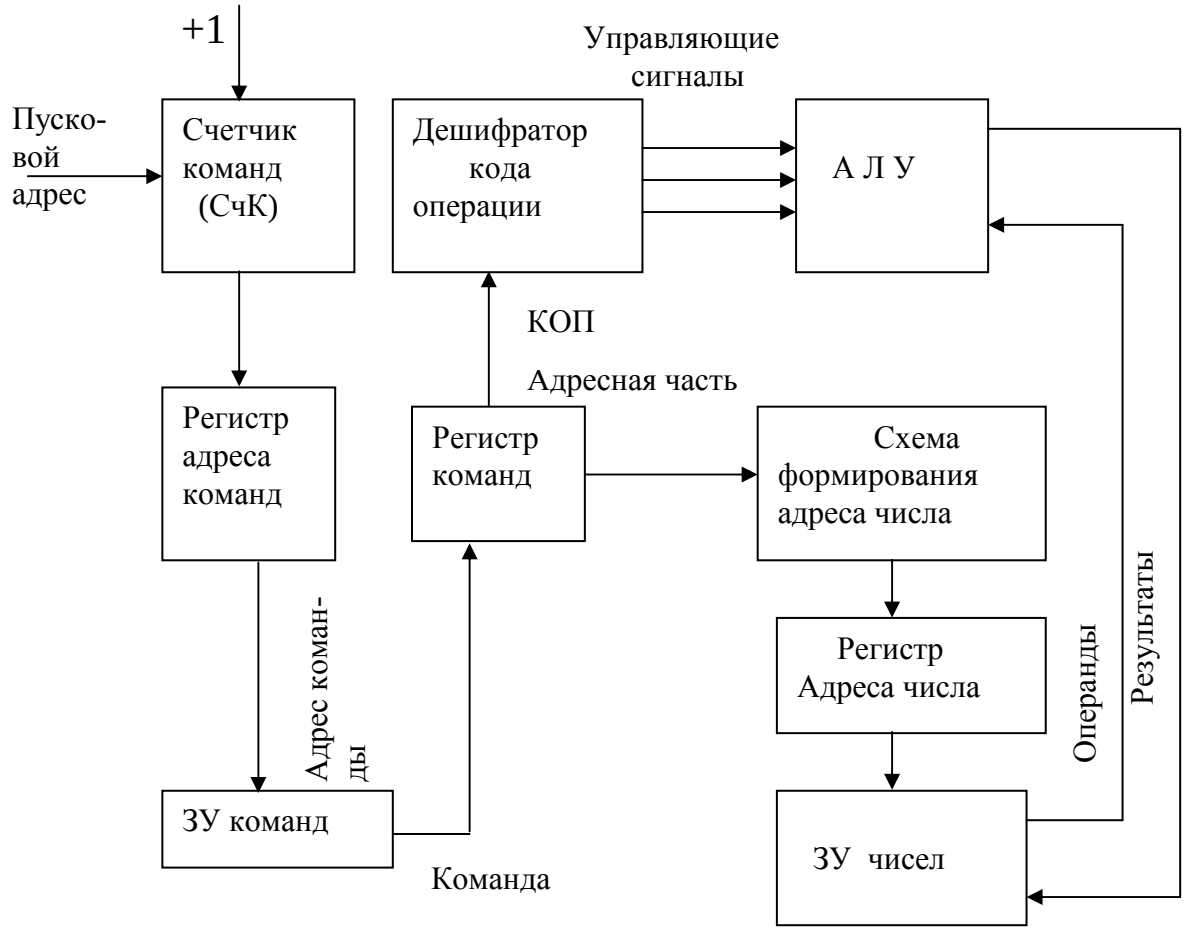


Рисунок 4.4 – Схема выполнения команд в БЦВМ

В начальный момент времени (момент включения питания или момент поступления с пульта управления соответствующей команды) в счетчик команд (СчК), предназначенный для формирования адреса текущей команды , поступает пусковой адрес программы. Хранение этого адреса в течение всего времени выполнения команды осуществляется в регистре адреса команд (РгАК). Адрес команды из РгАК поступает в запоминающее

устройство команд. По этому адресу из ЗУ считывается команда и записывается в регистр команд (РгК), где хранится на протяжении всего времени выполнения команды. С регистра РгК операционная (КОП) и адресная части по-отдельности поступают, соответственно, в дешифратор кода операции и схему формирования адреса числа. Дешифратор кода операции обеспечивает формирование набора управляющих сигналов, необходимых для реализации команды. Эти сигналы в дальнейшем поступают на все устройства, задействованные в выполнении команды. Схема формирования адреса по адресной части команды и другим признакам, характеризующим способ формирования адреса, обеспечивает получение адреса числа. В зависимости от размеров адресной части возможно одновременное формирование адресов нескольких операндов. Например, адрес первого слагаемого, адрес второго слагаемого и адрес результата. Хранение сформированного адреса в процессе выполнения команды осуществляется в регистре адреса числа (РгАЧ).

В соответствии с полученными адресами из ЗУ чисел извлекаются сперанды и поступают на арифметическо-логическое устройство. АЛУ обеспечивает необходимые действия над операндами и формирует результат операции. Длительное хранение результата осуществляется в ЗУ чисел. По окончании выполнения команды содержимое счетчика команд автоматически увеличивается на единицу младшего разряда. Таким образом обеспечивается формирование следующего адреса последовательности команд и процесс вычислений продолжается. Последняя команда последовательности содержит признаки, по которым увеличение содержимого СчК не происходит, следовательно дальнейшие вычисления прекращаются.

4.3.3 Основные характеристики БВМ

Возможности ЭВМ как вычислительных средств определяются совокупностью ее технических характеристик, основными из которых являются операционные ресурсы, емкость памяти и быстродействие.

Операционные ресурсы определяются в основном способом представления информации и системой команд. Способ

представления информации – это множество форм, используемых в машине для представления чисел, логических величин и др. Система команд – это перечень команд, определяющих номенклатуру машинных операций и способов адресации (способов формирования адресов операндов и команд).

Рассмотрим некоторые из характеристик, составляющих операционные ресурсы БЦВМ.

В зависимости от способа обработки и передачи информации различают параллельные и последовательные ЭВМ. В параллельных ЦВМ обработка информации в процессоре, а также передача ее между устройствами машины производится одновременно по всем разрядам. В последовательных ЦВМ каждое машинное слово обрабатывается в процессоре и передается между устройствами машины поразрядно. Ясно, что при прочих равных условиях быстродействие параллельных ЦВМ выше, чем у последовательных. Но последовательные ЦВМ характеризуются меньшим объемом оборудования.

По способу управления выполнением операций в процессоре БЦВМ подразделяют на синхронные и асинхронные. В синхронных БЦВМ на выполнение каждой операции отводится время, определенное для наихудшего сочетания операндов, пусть даже редко встречающегося. Следующая операция в синхронных БЦВМ не начинается, пока не истечет время, отведенное для выполнения текущей операции, даже если она фактичесчки и завершилась.

Время выполнения каждой операции связано с определенным количеством тактовых (синхронизирующих) импульсов, вырабатываемых внутренним или внешним генератором.

В асинхронных БЦВМ блоки выполнения машинных операций имеют оборудование для фиксации момента фактического окончания текущей операции и выработки сигнала начать следующую операцию. Современные БЦВМ являются синхронными машинами.

В БЦВМ применяются различные форматы команд: одноадресные и многоадресные, с фиксированной и переменной адресностью. Используются также различные способы формирования адресов операндов и команд: прямая адресация, относительная, косвенная.

Важными характеристиками БЦВМ являются емкости ее ОЗУ и ПЗУ. Емкость ПЗУ определяется объемом программ и констант,

которые необходимо хранить в нем. Емкость ПЗУ ограничена сверху следующим фактором. Процессор разрабатывается в расчете на работу с адресами длиной в $r_{ПЗУ}$ разрядов. При этом предельная емкость ПЗУ, доступная процессору, составляет величину $E_{ПЗУ} = 2^{r_{ПЗУ}}$ ячеек памяти. Емкость ОЗУ определяется характером решаемых задач, темпом прохождения программ их решения в БЦВМ. По аналогии емкость ОЗУ равна $E_{ОЗУ} = 2^{r_{ОЗУ}}$, где $r_{ОЗУ}$ – разрядность адреса ОЗУ.

Быстродействие БЦВМ характеризуется скоростью, с которой производится обработка информации. Его определяют либо числом операций, выполняемых в единицу времени, либо временем выполнения одной операции.

4.3.4 Форматы представления команд и чисел в БЦВМ

Форматом называется количество и назначение разрядов двоичного кода, используемого для записи команд, чисел и адресов цифровой ЭВМ. БЦВМ предназначена для выполнения управляющих программ, каждая из которых представляет собой последовательность команд. Команда цифровой ЭВМ должна нести информацию о том, что нужно сделать и над чем будут эти действия производиться. Поэтому структура команды представлена операционной и операндной частями (см. пункт 4.3.1). При выполнении двуместных операций, таких, например, как операция сложения, операндная часть должна содержать информацию о месте нахождения первого и второго операнда или значения этих операндов. В тоже время, результат операции (сумма) тоже следует разместить по заданному адресу, поэтому в общем случае формат команды должен быть трех адресным:

КОП	*А1*	*А2*	*А3*

,

где КОП – код операции (операционная часть команды);

А1, *А2*, *А3*- соответственно, адрес первого и второго операнда и адрес результата.

Если один из операндов для дальнейших вычислений не нужен, его адрес используется для хранения результата. Формат команд, в этом случае называется двух адресным:

КОП	*А1*	*А2*

.

Возможности вычислительной машины по обработке информации во многом определяются её системой команд, т.е. тем какие команды "умеет" выполнять данная БЦВМ. Состав системы команд современных БЦВМ от 64 до 256 различных команд. Таким образом, двоичный номер команды (код операции) должен содержать от 6 до 8 разрядов. Если емкость памяти E=64 К слов, то для кодирования адреса требуется $n = \text{int}(\log_2 E)$=16 разрядов. Поэтому даже для двухадресной команды требуется от 38 до 40 разрядов. Чтобы уменьшить занимаемое одной командой количество разрядов памяти в структуре процессора БЦВМ для хранения одного из операндов используется специальный регистр. В БЦВМ типа "Орбита" это регистр накопитель (РН), а в БЦВМ типа А-15 - регистр сумматора первый (PrC1). Формат команды в этом случае состоит из КОП и адреса второго операнда:

КОП	*А1*

.

Дальнейшее уменьшение числа разрядов команды возможно, если в операндной части команды приведен не весь адрес, а только его часть. Полный адрес в этом случае формируется по правилам, определяемым специальным двоичным кодом, называемым признаком адресации (ПА).

В таблицах 4.1,4.2 приведены форматы команд изучаемых БЦВМ.

Таблица 4.1- Формат команд БЦВМ А-15А

Нумерация разрядов команды	0-я триада			1-я			2-я			3-я	4-я			5-я			6-я		
	А	Б	В	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Функциональное назначение	Признак адресации (ПА)			Код операции (КОП)						Адресная часть (смещение)									

Таблица 2- Формат команд БЦВМ Орбита 10(20)

Нумерация разрядов команды	0-я	1-я триада			2-я			3-я			4-я			5-я		
	15	14	12	13	11	10	9	8	7	6	5	4	3	2	1	0
Функциональное назначение	ПМ	Код операции (КОП)						Адресная часть								

Обозначение ПМ, используемое в формате команд БЦВМ Орбита-10, означает "признак модификации" и характеризует способ формирования адреса. О способах формирования адреса подробно будет рассмотрено в следующей главе.

Формат команд БЦВМ Ц-100 соответствует формату трехадресных команд и в общем случае содержит три поля, каждое из которых состоит из одного или нескольких байтов:

поле кода операции;

поле, содержащее описание данных и указание о размещении результата;

поле меток.

Поле кода операции состоит из одного байта, в котором записывается номер операции (команды). Один байт кода операции позволяет закодировать 256 команд, хотя все коды пока не использованы.

В БЦВМ с целью уменьшения аппаратурных затрат на реализацию АЛУ применяются машинные двоичные коды для представления чисел. Обычно положительные числа представляются в виде правильных дробей в прямом коде, а отрицательные – в дополнительном. Формат чисел показан в таблице 4.3.

Таблица 4.3- Формат чисел

Нумерация разрядов числа	0-я	1-я триада			2-й			3-я			4-я			5-я		
	0	1	2	3	4	5	6	7	6	9	10	11	12	13	14	15
Назначение	Знак	Мантисса числа														

Наименьшее число, представимое в разрядной сетке:
0,00...01= 2^{-15}. В разрядной сетке могут быть представлены числа в диапазоне от - $(1 – 2^{-15})$ до $–2^{-15}$ и от $+2^{-15}$ до + $(1 – 2^{-15})$. Числа меньше

2^{-15} не могут быть изображены в разрядной сетке и принимаются равными нулю (число выходит за разрядную сетку вправо). Все числа большие $1-2^{-15}$ также не могут быть представлены в принятой разрядной сетке. Такое число выходит из сетки влево (переполнение разрядной сетки), и его старшие разряды (разряды слева от запятой) теряются. Поскольку в этом случае результат вычислений оказывается неверным, поэтому при переполнении происходит автоматическая остановка машины либо формируется сигнал прерывания программы.

Для исключения переполнения разрядной сетки при программировании задачи данные, участвующие в вычислениях, берутся с соответствующими масштабными коэффициентами.

При научно-технических расчетах масштабирование оказывается более простым, если все числа по модулю должны быть меньше 1, т.е. запятая фиксирована перед старшим разрядом числа, а масштабные коэффициенты представляются в виде 2^i.

Использование представления чисел с фиксированной запятой позволяет упростить схемы машины, повысить ее быстродействие, но создает трудности при программировании.

В изучаемых БЦВМ реализовано представление чисел с фиксированной запятой.

4.4 Особенности программирования в кодах команд БВМ

Бортовые цифровые вычислительные машины и системы являются управляющими и предназначены для реализации соответствующих алгоритмов управления системой "летательный аппарат - бортовое оборудование - оружие" в реальном или форсированном масштабе времени. Решение задачи в реальном масштабе времени означает получение результата вычислений до существенного изменения исходных данных. В еще более ускоренном, опережающем темпе, но с соблюдением требований по точности в так называемом форсированном масштабе времени, решаются задачи экстраполяции: прогнозирование фазовых координат цели, собственного ЛА, развития процесса функционирования различных бортовых систем и подсистем.

Реализация алгоритмов в реальном, а тем более в форсированном масштабе времени, предъявляет высокие требования к быстродействию вычислительных систем, экономичности численных методов решения и внедрению специальной пакетно-приоритетной систе-

мы организации вычислительного процесса.

В связи с изложенным, разработка программного обеспечения БВМ принципиально отличается от технологии создания программ для ЭВМ общего назначения, реализуемых не в реальном масштабе времени и обычно на языке высокого уровня.

Процесс подготовки задач для решения на БВМ можно разделить на следующие основные этапы:

формализация поставленной задачи (математическая формулировка задачи);

выбор численных методов решения;

масштабирование и составление машинных соотношений;

составление структурной схемы алгоритма решения задачи;

распределение памяти;

составление программы в условных и истинных адресах;

отладка программы и ее эксплуатация.

Математическая формулировка задачи представляет собой взаимосвязь реальных физических величин, исходных данных и результатов вычислений. В процессе разработки формульных зависимостей ставится вопрос о замкнутости (определенности) получаемых систем уравнений, решается, какая информационная поддержка потребуется для разработанного алгоритма, формируется структура управляющей системы.

Выбор численных методов решения задачи для бортовых управляющих ЦВМ предполагает замену громоздких классических выражений (интегралов, дифференциальных уравнений и т.д.) экономичными по операциям, точными и быстродействующими формулами.

При этом необходимо упрощать до разумного предела, соблюдая требования точности, математическое описание задачи.

БВМ оперирует с информацией, представляемой, главным образом, правильными дробями с ограниченной разрядной сеткой. В этом случае все физические величины решаемой задачи представляются в БВМ в некоторых масштабах. Масштаб представляет собой следующее соотношение:

$$m_x = \frac{\bar{x}}{x},$$

где x - физическое значение переменной; $\bar{x}$ представление переменной в БВМ.

На основании масштабных соотношений физические выражения

преобразуются в машинные. Рассмотрим процесс масштабирования для БВМ с фиксированной запятой на примере решения задачи измерения дальности до цели частотным методом. В простейшем случае формула вычислений имеет вид:

$$D = \frac{c \cdot F_P}{2 \cdot a}, \qquad (4.1)$$

где с - скорость распространения радиоволн; a - скорость изменения частоты; $F_p = f_{изл} - f_{прм}$ - разностная частота (разность частот излученного и принятого сигналов).

Так как максимальное значение машинных величин не может превышать единицы, то масштабы переменных, входящих в выражение (4.1), можно определить следующими соотношениями:

$$m_f = \frac{1}{f_{\max}},\ m_a = \frac{1}{a_{\max}}. \qquad (4.2)$$

Масштаб постоянных величин, используемых в выражении (4.1), определяется следующим образом:

$$m_K = \frac{1}{K - 1мл.р}, \qquad (4.3)$$

где К - значение константы; 1мл.р- единица младшего разряда (значащая цифра, определяющая точность измерения или записи константы).

Максимальные значения физических переменных известны из тактико-технических характеристик РЛС. Следовательно, масштабы имеют конкретные числовые значения.

Подставляя в соотношение (4.2) физические переменные, выраженные через масштабы, и машинные переменные в соответствии с выражением (4.1) получим:

$$\overline{D} = \frac{m_{K_1} \cdot m_f}{m_{K_2} \cdot m_a} \frac{\overline{c} \cdot \overline{f_{изл} - f_{прм}}}{\overline{2} \cdot \overline{a}},$$

или

$$\overline{D} = K_D \frac{\overline{c} \cdot \overline{f_{изл} - f_{прм}}}{\overline{2} \cdot \overline{a}}. \qquad (4.4)$$

Выражение представляет собой машинное соотношение задачи измерения дальности частотным способом, в соответствии с которым составляется программа. Все переменные в этих выражениях не превышают единицы. Коэффициент K_D является масштабным.

Выполнение арифметических операций должно производиться в такой последовательности, чтобы на каждом этапе вычислений не возникало переполнения. Коэффициент K_D можно, корректируя масштабы, изменить так, чтобы его значение было равно $2^{\pm m}$. Далее вычисляется выражение

$$\overline{D} = \frac{\overline{c} \cdot \overline{f_{изл} - f_{прм}}}{\overline{2} \cdot \overline{a}},$$

которое с помощью операции арифметического сдвига на m разрядов умножается на коэффициент $2^{\pm m}$. В этом случае отпадает необходимость хранения коэффициента K_D в памяти.

При составлении программ для БВМ с фиксированной запятой необходимо учесть следующие особенности выполнения арифметических команд.

1. Перед сложением либо вычитанием операндов их масштабы должны быть уравнены.

2. Для выполнения деления необходимо, чтобы делимое по абсолютной величине было меньше делителя: показатель степени масштаба частного равен разности показателей степеней масштабов делимого и делителя.

3. Результат выполнения операции умножения имеет масштаб, равный произведению масштабов множимого и множителя.

После подготовки задачи для решения на ЭВМ анализируется логика решения этой задачи и в результате этого анализа составляется структурная схема алгоритма.

Алгоритм - это определенная последовательность действий, математических операций над исходными данными, выполнение которых приводит к получению результата. Алгоритмы можно описывать:

словесно, используя математические соотношения;

при помощи схем в операторной форме;

на алгоритмическом языке.

Схема алгоритма представляет собой выполненное в соответствии с ГОСТ 19427 графическое изображение операторов и порядка их исполнения.

Еще одной особенностью составления программ для БВМ является необходимость рационального распределения памяти. В отличие от универсальных ЭВМ, в бортовых ЦВМ коды команд могут находиться лишь в долговременной памяти, поэтому массивы команд, ис-

ходных данных, окончательных и промежуточных результатов могут находиться в различных блоках памяти. В БЦВМ типа А-15, кроме того, следует использовать различные виды адресации для формирования исполнительного адреса. Применение различных признаков адресации позволяет существенно уменьшить число используемых в программе команд и время ее выполнения.

При программировании в кодах команд БЦВМ сначала программирование ведут в условных (символических) адресах, так как заранее не известно, сколько ячеек памяти займет программа. Это необходимо еще и потому, что при разработке сложных программ части алгоритма могут программироваться на начальном этапе несколькими исполнителями, а затем состыковываться.

Условный адрес представляет собой адрес ячейки массива, отсчитанный от некоторого, неизвестного пока, начального адреса, за которым начинается массив, Этот начальный адрес называется базисным и обозначается буквой, так как до определенного этапа программирования конкретное значение его неизвестно. Обычно выделяют следующие массивы: массивы команд, массивы исходных данных и констант, массивы промежуточных результатов, массивы окончательных результатов. Каждому массиву присваивают свой базисный адрес. Например, К - базисный адрес команд, С - базисный адрес констант, р - базисный адрес промежуточных результатов, *r* - базисный адрес окончательных результатов.

Условные адреса представляются в виде суммы базисного адреса и константы (числа и буквы).

Например, условный адрес К+1 будет адресом первой команды в массиве команд, условный адрес С + 5 – соответствует пятой константе в массиве констант.

Базисным адресам и, соответственно, условным адресам в конце программирования, когда становится известным объем каждого массива, присваиваются численные значения.

При использовании условных адресов память БВМ используется более эффективно, поскольку между массивами отсутствуют ячейки памяти, незадействованные при работе программы. Разумеется, необходимо помнить об ограниченном объеме памяти и не превышать его.

Переход к программированию в истинных адресах осуществляется после завершения программирования в условных адресах. В пер-

вую очередь принимаются конкретные значения для базисных адресов.

На этапе отладки проверяется правильность выполнения программой заданного алгоритма и правильность логического и временного взаимодействия частей программы. На этапе эксплуатации проверяется работоспособность и эксплуатационные характеристики программ в реальных условиях функционирования. Проводятся работы по внесению в программы необходимых изменений для исправления обнаруженных ошибок, улучшения характеристик и функциональных возможностей, приспособления к изменению состава и взаимосвязей оборудования АРЭК, управляемого БЦВМ.

4.5 Общие сведения о системе команд БВМ

Машинный язык или совокупность операторов, выполняемых процессором бортовой ЦВМ, составляет систему команд БЦВМ. Система команд оказывает существенное влияние на структуру и порядок работы процессора БВМ, а также на технические (производительность) и экономические (стоимость) характеристики БВМ в целом. Выбор системы команд БЦВМ осуществляется с учетом следующих основных требований:

функциональной полноты системы команд;

обеспечения максимальной производительности БЦВМ;

минимизации требуемого для хранения программ и данных объема памяти БЦВМ.

Требование функциональной полноты системы команд означает возможность создания (написания) с её помощью программ, реализующих любой заданный алгоритм управления сопрягаемым бортовым оборудованием.

Требование обеспечения, максимальной производительности БЦВМ приводит к тому, что система команд должна быть ориентирована на реализацию операций, соответствующих предметной области.

Требование минимизации объема памяти ЦВМ должно рассматриваться аналогично предыдущему.

Особенности состава системы команд (СК) рассмотрим на примере СК БЦВМ А-15А. В СК БЦВМ А-15А содержится 71 команда. Код операции занимает 6 разрядов в формате команды, поэтому всего возможно сформировать 64 различных кода операции. Не все коман-

ды требуют указания адреса операнда, поэтому свободные разряды адресной части команды используются для расширения кода операции. В настоящее время задействовано 59 кодов операции, 5 вариантов являются резервными.

Все команды разбиты на 12 групп:

арифметические операции;

логические операции;

битовые операции;

операции посылок;

операции записи;

команды управления;

операции обмена информацией между регистрами вычислительного устройства;

операции ввода-вывода и нормализации;

операции вычисления функций;

операции установки сигналов;

операции над содержимым регистров;

специальные команды.

В состав арифметических команд входят обычные команды сложения и вычитания, команды сложения и вычитания двойной длины и с замещением, т.е. с записью результата в ячейку ОЗУ, откуда был взят один из операндов. А также команды умножения и деления тоже двух видов: обычные и с округлением.

Кроме того, есть команды вычитания модулей и команда определения знака содержимого ячейки памяти.

Логические операции обеспечивают выполнение основных операций булевой алгебры: логического умножения, логического сложения и отрицания равнозначности (сложение по модулю 2) над содержимым двух многоразрядных слов. Так же как и при выполнении ряда арифметических операций, существует возможность выполнять все логические операции обычным способом и с замещением.

Битовые операции служат для выполнения действий над отдельными разрядами слова, адрес которого указан в адресной части команды. При этом возможна установка разрядов в ноль и единицу, а также логическое умножение и логическое сложение заданного разряда слова со значением признака результата ω.

Операции посылок и операции записи предназначены для посылки содержимого заданных ячеек памяти в основные регистры

процессора и записи содержимого регистров в память БЦВМ.

Команды управления служат для изменения последовательности выполнения операций и содержат команды условного и безусловного перехода.

Операция обмена и операции вычисления функций имеют один код операции 16 и обеспечивают пересылку информации между основными регистрами процессора и вычисление функций синус, косинус и корень квадратный над аргументом, находящимся в регистре сумматора.

Операции ввода-вывода служат для обмена информацией между БЦВМ и устройством ввода-вывода,

Операции установки сигналов и операции над содержимым регистров предназначены для изменения значения различных признаков и содержимого регистров, обслуживающих режим прерывания.

В состав специальных команд входят: холостая команда (команда, не выполняющая каких-либо действий и обеспечивающая задержку вычислений на время выполнения одной команды); команда увеличения содержимого заданной ячейки памяти на единицу младшего разряда; команда модификации, служащая для изменения следующей за ней команды путем суммирования ее кода с кодом ячейки, адрес которой указан в команде модификации.

Глава 5. ФУНКЦИОНАЛЬНЫЕ УСТРОЙСТВА БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИНМ

5.1 Устройства памяти БЦВМ

5.1.1 Назначение, основные характеристики и классификация запоминающих устройств

Запоминающее устройство (ЗУ) - часть вычислительной машины, предназначенная для записи, хранения и выдачи информации - команд программы, исходных данных, промежуточных и конечных результатов, представленных в виде кодов.

Основными характеристиками ЗУ являются емкость и быстродействие.

Под емкостью понимают максимальное количество кодов (машинных слов), которое одновременно может храниться в ЗУ. Каждый код размещается в одной ячейке памяти. Если емкость памяти оценивать в битах, то она равна произведению числа ячеек памяти на их длину (разрядность):

$$Q = N \cdot n,$$

где Q - емкость в битах; N - количество ячеек ; n - разрядность ячеек.

Быстродействие характеризуется временем обращения к ЗУ, необходимым для записи или чтения информации. При чтении информации время обращения складывается из времени $t_{П}$ поиска соответствующей ячейки, времени $t_{СЧ}$ считывания и времени $t_{В}$ восстановления (в случае необходимости)

$$T_{ЧТ} = t_{П} + t_{СЧ} + t_{В}.$$

При записи время обращения определяется временем поиска, временем $t_{СТ}$ стирания ранее записанной информации и временем $t_{З}$ записи:

$$T_{ЗП} = t_{П} + t_{СТ} + t_{З}.$$

Практически $T_{ЧТ} \approx T_{ЗП}$.

Кроме рассмотренных основных характеристик, важными показателями качества ЗУ являются:

надежность – способность ЗУ сохранять свои параметры в заданных пределах при различных условиях эксплуатации;

экономичность – стоимость ЗУ, отнесенная к его емкости;

габариты и потребляемая мощность.

Одновременное увеличение емкости ЗУ, уменьшение его стоимости и увеличение быстродействия не возможно, так как эти характеристики связны между собой прямой или обратной зависимостью. Так чем больше емкость памяти, то, следовательно, и больше время поиска данных. Требования к емкости и быстродействию памяти являются противоречивыми. Чем больше быстродействие, тем технически труднее достигается и дороже обходится увеличение емкости памяти. Стоимость памяти составляет значительную часть общей стоимости ЦВМ. Поэтому память организуется в виде иерархической структуры запоминающих устройств, обладающих различными быстродействием и емкостью.

По своему функциональному назначению все ЗУ, используемые в БВМ, можно разделить на следующие группы:

сверхоперативные ЗУ (СОЗУ) — представляющие собой набор регистров, содержимое которых непосредственно используется при обработке информации в процессоре;

оперативные ЗУ (ОЗУ) — хранящие оперативную информацию (операнды, части программы, требующиеся в процессе работы);

постоянные ЗУ (ПЗУ) — предназначенные для длительного хранения неизменяемой в процессе работы БВМ информации (программ, микропрограмм, констант);

полупостоянные (репрограммируемые) ЗУ (РПЗУ) — по выполняемым функциям полностью соответствуют постоянным ЗУ, но отличаются от последних возможностью относительно быстрой смены одновременно всей хранимой в них информации;

внешние ЗУ (ВЗУ) — предназначенные для хранения больших объемов информации с небольшой удельной стоимостью бита хранимой информации. ВЗУ находятся вне корпуса БВМ. Размещение на борту истребителя ВЗУ на магнитных дисках, лентах и барабанах, в настоящее время затруднено в связи с тяжелыми условиями функционирования. Поэтому ВЗУ обычно используются при проведении наземных операций, связанных с контролем работоспособности и диагностикой БВМ и всего радиоэлектронного комплекса в целом.

Запоминающие устройства могут быть реализованы на основе только БИС ОЗУ. Однако потеря информации при отключении питания, более высокая удельная стоимость бита хранимой информации, и ряд других причин привели к широкому использованию постоянных и полупостоянных ЗУ.

По способу организации обмена информацией между отдельными ЗУ и процессорами различают ЗУ: с произвольной выборкой (ЗУПВ); с последовательным обращением (ЗУПО) и с последовательно-параллельной организацией обмена.

Подавляющая часть ЗУ БВМ — это ЗУ с произвольной выборкой (или с произвольным доступом). Если память не обеспечивает произвольного доступа к любой её части, то это память с последовательным доступом, она редко используется в качестве основной памяти БВМ. При работе с такой памятью для обращения к нужной области необходимо предварительно обратиться ко всем областям, лежащим между той, к которой произошло обращение в данный момент, и требуемой областью. В последовательной форме хранится информация на магнитной ленте. Последовательный доступ применяют для запоминания значительных объемов данных, время обращения к которым не является критичным.

Одна из характеристик ЗУ — способность сохранять информацию при отключении источников питания. В этом случае различают энергозависимую или энергонезависимую память. В энергонезависимой памяти при нарушениях в работе системы питания данные не разрушаются, а в энергозависимой — разрушается.

Для современных БВМ наиболее характерно использование оперативных запоминающих устройств на униполярных МОП-транзисторах.

В некоторых микропроцессорных БВМ находят применение еще два вида полупроводниковой памяти: на приборах с зарядовой связью (ППЗС) и на цилиндрических магнитных доменах (ПЦМД). Оба типа памяти ориентированы на последовательный доступ. Как и другие устройства последовательного доступа, они имеют невысокое быстродействие, однако позволяют реализовать запоминающие устройства значительно большего объема, чем устройства памяти на биполярных и МОП-транзисторах.

Достоинства ППЗС — чрезвычайно малое потребление мощности, простота в применении. Память на ППЗС энергозависима, но малая потребляемая мощность делает возможным сохранение в ней информации с помощью резервных аккумуляторов при отказе на длительное время основного источника питания

Для ПЦМД требуется гораздо большее число вспомогательных схем, чем для ППЗС, но она является энергонезависимой.

Таким образом, этот тип памяти позволяет реализовать энергонезависимые запоминающие устройства большого объема, для которых не требуются сложные механические лентопротяжные механизмы и дисководы.

5.1.2 Оперативные запоминающие устройства

Оперативная память относится к памяти с произвольным доступом (Random access memory – RAM). По принципу хранения информации полупроводниковые оперативные ЗУ делятся на динамические и статические.

Динамические ЗУ (DRAM) строятся на основе запоминающего элемента, сохраняющего свое состояние только определенный промежуток времени, поэтому требующего периодического восстановления. Запоминающим элементом динамических полупроводниковых ЗУ служит конденсатор, в котором информация хранится в форме наличия или отсутствия заряда. Из-за имеющихся утечек заряд на запоминающем конденсаторе постоянно уменьшается. Для восстановления заряда запоминающий конденсатор периодически подключают к источнику питания.

Простейшая схема динамической ячейки памяти на МОП-транзисторах приведена на рисунке 5.1. Хранение информации обеспечивается с помощью конденсатора С3. В ячейке используются отдельные адресные (АЛз и АЛсч) и разрядные линии (РЛз и РЛсч) записи и считывания.

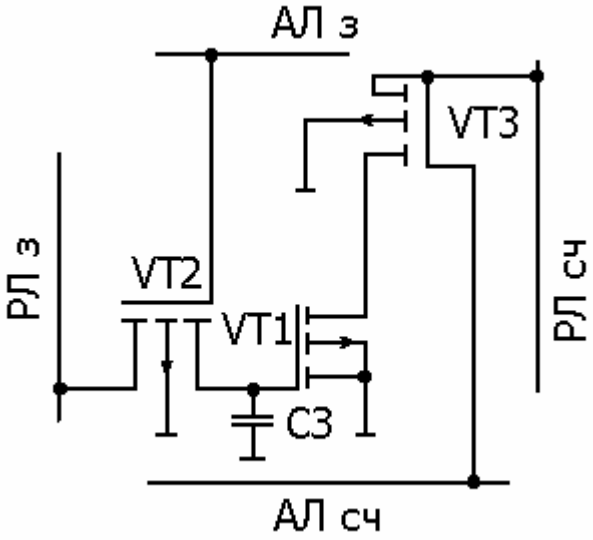


Рисунок 5.1 - Динамическая ячейка памяти

В процессе записи информации от РЛз (при разрешающем сигнале на АЛз) через открытый транзистор VT2 обеспечивается заряд емкости СЗ. Считывание информации производится по линии РЛсч через VT1 и VT3 (при разрешающем сигнале АЛсч).

Регенерация (восстановление данных) ячейки динамической памяти производится при каждом обращении к ней. Однако при обычной интенсивности работы памяти БВМ такая регенерация не гарантирует сохранности всех хранящихся в ней битов информации. БВМ может, например, затратить время, превышающее несколько миллисекунд, на выполнение простого цикла, предназначенного для выработки временной задержки, и использовать при этом лишь несколько ячеек памяти. В течение всего этого промежутка времени все другие слова памяти не будут подвергаться регенерации. Поэтому устройства памяти, выполненные на динамических ЗУ, нуждаются в логической схеме регенерации, которая автоматически обращается к каждому столбцу памяти с интервалами в несколько десятых долей миллисекунды. Динамическое ЗУ построено так, что само обращение к столбцу обеспечивает регенерацию информации во всех его ячейках. Работа логической схемы регенерации должна координироваться с другими действиями процессора. Если, например, процессор пытается обратиться к памяти в момент, когда в ней осуществляется регенерация, то схема регенерации должна отдать приоритет именно процессору.

Динамические полупроводниковые ЗУ с произвольной выборкой имеют высокий уровень интеграции и быстродействия, низкую стоимость, поэтому находят широкое применение в микропроцессорных системах. На динамических БИС ЗУ, являющихся функционально законченными устройствами, можно сравнительно просто строить ОЗУ различной информационной емкости.

Недостаток динамических ЗУ — необходимость регенерации — компенсируется большей, чем в статических ЗУ, информационной емкостью.

Статические ЗУ (SRAM) являются наиболее распространенным видом памяти микропроцессорных систем. Большинство статических ЗУ реализуется на основе МОП-технологии, но существуют и статические ЗУ на биполярных схемах.

Ячейка памяти статического ЗУ представляет собой обычный

триггер. Он может быть установлен либо в состояние «0», либо в состояние «1». Если триггер установлен в «1», то это состояние сохраняется до тех пор, пока не будет произведен сброс триггера или не будет выключено питание.

Каждый триггер хранит один бит информации. Триггеры имеют раздельные входы для установки их в состояние «0» или «1», а также вентили для соединения их с линией шины данных. Каждая ячейка памяти обычно содержит шесть МОП-транзисторов (рисунок 5.2).

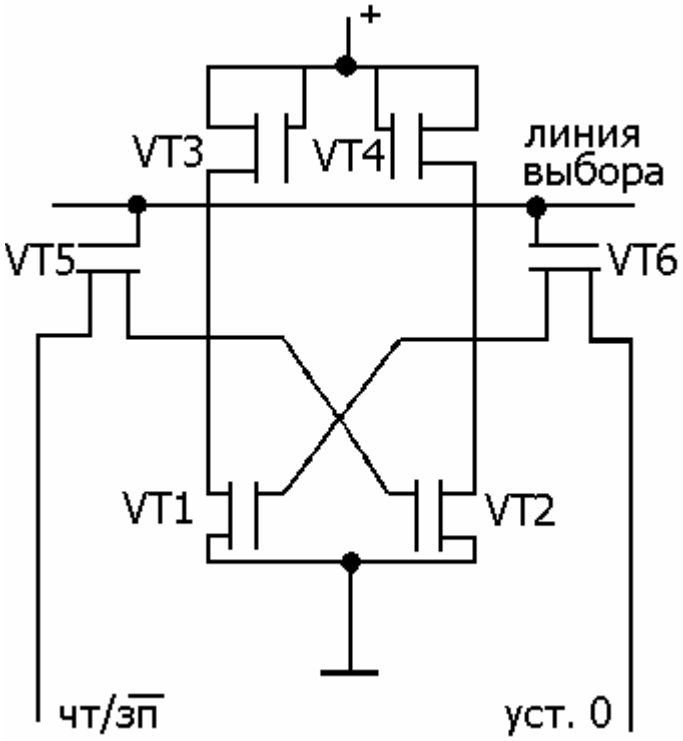


Рисунок 5.2 - Ячейка памяти статического ОЗУ

При обращении к ячейке как к элементу памяти на линии выбора устанавливается сигнал высокого уровня. Транзисторы V5 и V6 включаются, а линия чтения-записи (ЧТ/ЗП) подключается к затвору. Для записи лог. «1» в этот элемент на линии ЧТ/ЗП устанавливается высокий уровень, а на линии «Установка 0» — низкий уровень сигнала. При этом V2 отпирается, а V1 запирается. При записи лог. «0» уровни сигналов на линиях ЧТ/ЗП и «Установка 0» изменяются на противоположные. После окончания сигналов установки состояние транзисторов V1 и V2 не изменяются. Комбинацией уровней сигналов в линиях ЧТ/ЗП и «Установка 0» обеспечиваются три состояния ячейки памяти: запись, чтение и хранение информации.

Подобные ячейки памяти объединяются в матричную структуру, т. е. размещаются по строкам и столбцам. Каждому элементу памяти

присваивается адрес, являющийся n-разрядным двоичным кодом. Количество разрядов двоичного кода определяет емкость матрицы ЗУ и определяется как $Q = 2^n$, где n — разрядность адресного кода.

Разряды адресного кода разбиваются на две группы: разряды адресов строк и разряды адресов столбцов. Каждая группа разрядов подключается на вход соответствующих дешифраторов, выходы которых обеспечивают обращение к ячейке памяти с указанным адресом строки и адресом столбца.

На рисунке 5.3 изображена структура микросхемы ОЗУ с раздельными одноразрядными входами и выходами.

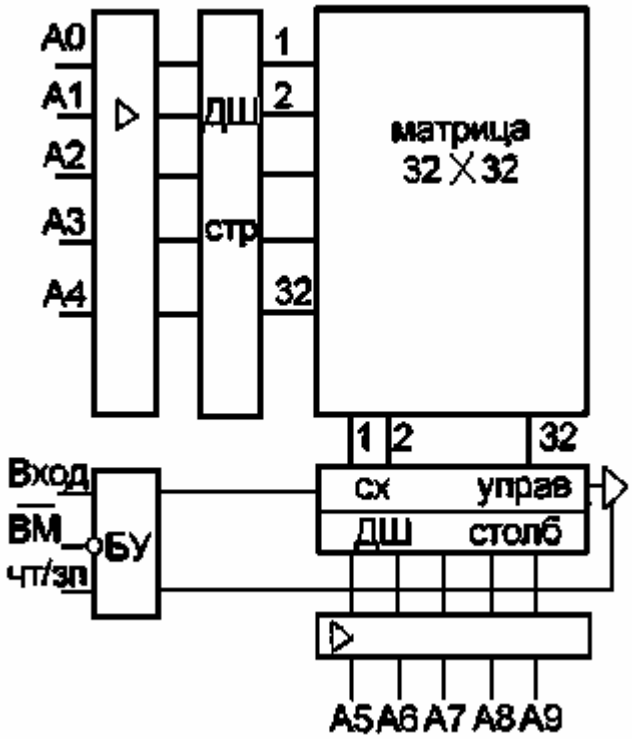


Рисунок 5.3 - Структура ИС ОЗУ

В микросхеме используются усилители адресных сигналов и дешифраторы адресов $5 \rightarrow 32$, что позволяет адресоваться к 1024 одноразрядным ячейкам памяти. Для обращения к ИС ОЗУ используется сигнал низкого уровня выборки микросхемы (ВМ) и сигнал управления ЧТ/ЗП. Бит информации, подлежащий записи, подается на вывод «Вход» блока управления (БУ), а при чтении сигнал «ЧТ» считывается с выходной линии.

Структура рассмотренной ИС 1024х1 бит. В настоящее время промышленность выпускает ИС памяти емкостью десятки и сотни мегабайт. В практическом применении более удобны ИС памяти с разрядностью, равной (или кратной) разрядности процессора.

В этом случае при однократном обращении к ИС памяти обеспечивается обмен словами, которые обрабатываются в процессоре.

Типичные БИС динамического ОЗУ: КР565РУ6 — емкость 16384х1 бит; К565РУ5 - емкость 65535х1 бит.

Типичные БИС статического ОЗУ: КР537РУ14 — емкость 4096х1 бит, КР132РУ6А — емкость 16 384х1 бит, К537РУ9 — емкость 2 048х8 бит.

5.1.3 Постоянные запоминающие устройства

Постоянные запоминающиеся устройства (ПЗУ) служат для хранения программ и другой неизменяемой информации, поэтому выполняют только операцию чтения (Read only memory – ROM). Важное преимущество ПЗУ по сравнению с ОЗУ — сохранение информации при выключении питания. Стоимость бита хранимой в ПЗУ информации может быть почти на порядок ниже, чем в ОЗУ. Постоянные ЗУ могут быть реализованы на основе различных физических принципов и элементов и отличаются способом занесения информации, кратностью занесения, способом стирания.

В качестве примера на рисунке 5.4, *а* приведена схема однотранзисторной ячейки постоянного ЗУ на биполярном транзисторе. В эмиттерной цепи транзистора предусмотрена плавкая перемычка, которая в необходимых случаях может разрушаться при программировании ЗУ. Поэтому при обращении к ЗУ по линии адреса *x* в случае неразрушенной перемычки в регистровой линии *y* будет протекать эмиттерный ток транзистора, (состояние лог. «1»), а в случае разрушенной перемычки ток течь не будет (состояние лог. «0»).

В настоящее время применяются следующие виду ПЗУ: программируемые на заводе-изготовителе или масочные ПЗУ (МПЗУ); программируемые пользователем ПЗУ и перепрограммируемые (репрограммируемые) РПЗУ. Первые два вида ПЗУ допускают только однократное программирование, репрограммируемые ПЗУ позволяют изменять хранимую в нем информацию многократно.

Рассмотрим подробнее каждый из типов ПЗУ.

Программируемые масочные ПЗУ (МПЗУ) — программируются их изготовителем, который по подготовленной пользователем информации делает фотошаблоны, с помощью которых заносит эту информацию в процессе производства на кристалл ПЗУ. Этот способ

самый дешевый и предназначен для крупносерийного производства ПЗУ. Масочные ПЗУ строятся на основе диодов, биполярных и МОП-транзисторов.

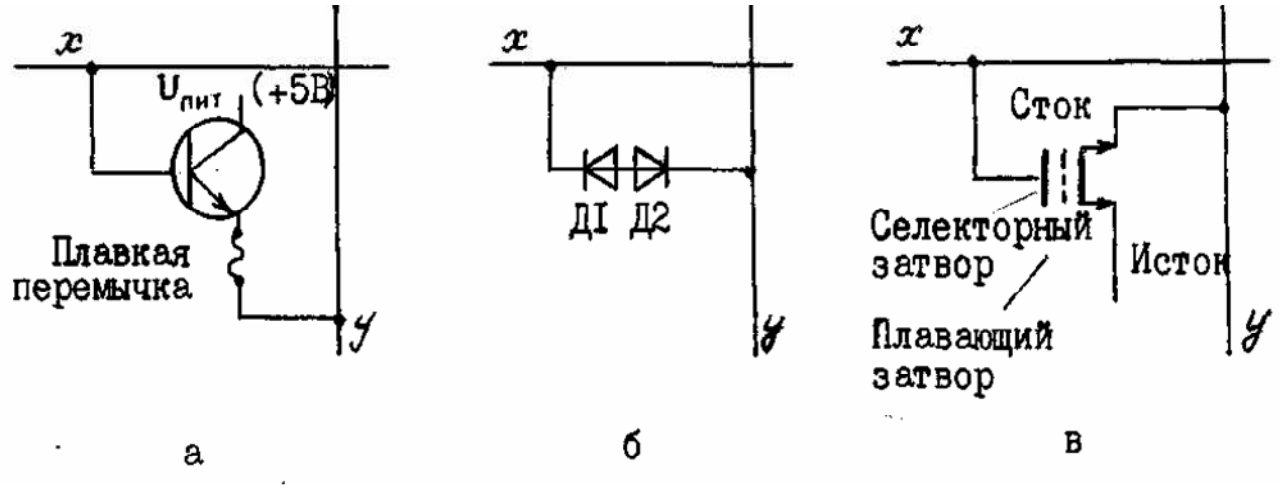


Рисунок 5.4 - Реализация ячеек памяти ПЗУ

В диодных ПЗУ диоды включены в тех пересечениях матрицы, которые соответствуют записи «1», и отсутствуют в местах, где должен быть записан «0». Внешние цепи очень просты. Так как диодная матрица представляет собой элемент с гальваническими связями, то выходные сигналы имеют ту же форму, что и входные. Таким образом, если на входы подаются напряжения постоянных уровней, то и на выходах уровни будут также постоянными, поэтому отпадает необходимость в выходном регистре для хранения информации. Масочные ПЗУ на биполярных и униполярных транзисторах также строятся в виде матриц. Постоянные ЗУ на МОП-транзисторах имеют меньшее энергопотребление, чем биполярные.

Масочные ПЗУ характеризуются высокой надежностью, но при их изготовлении возникает ряд неудобств и для заказчика, и для изготовителя. Велика номенклатура ПЗУ и мала их тиражность, поэтому от изготовителя требуются повышенные затраты на фотошаблоны, что увеличивает стоимость ПЗУ. Отсутствует возможность оперативно изменять информацию в ПЗУ без изготовления новой ИС, что особенно неудобно на этапе отработки программ системы.

Программируемые пользователем ПЗУ (ППЗУ) — являются более универсальными и, следовательно, более дорогими приборами. Они представляют собой матрицы биполярных приборов, связи которых с адресными и разрядными шинами разрушаются при занесении

на специальных программирующих устройствах соответствующих кодовых комбинаций. Эти устройства вырабатывают напряжения, необходимые и достаточные для пережигания плавких перемычек в выбранных запоминающих элементах ПЗУ. Роль плавкой перемычки может выполнять диод Д1, как показано на рисунке 5.4, б. Возможность однократного программирования пользователем сделала ПЗУ этого типа чрезвычайно удобными при разработке микро-ЭВМ.

Репрограммируемые ПЗУ (РПЗУ) — это ПЗУ с изменяемым содержанием, на затворах матрицы МОП-транзисторов длительное время могут храниться заряды, образующие заданный код. Все репрограммируемые ПЗУ представляют собой МОП-приборы.

При необходимости в перепрограммировании микросхемы предварительно записанную информацию стирают ультрафиолетовым светом через прозрачное кварцевое окошко на поверхности корпуса микросхемы. Попадая на плавающий затвор и выбивая из него фотоэлектроны, ультрафиолетовое излучение разряжает плавающий затвор МОП-транзистора. Время сохранения информации в микросхемах ПЗУ данного типа определяется качеством призатворного диэлектрика и для современных микросхем составляет десять лет и более.

Микросхемы ПЗУ с электрическим стиранием информации популярны у разработчиков микропроцессорной техники благодаря возможности быстрого стирания и записи, большому допустимому числу циклов перезаписи информации (10000 раз и более). Однако они достаточно дорогие и сложные по сравнению с ПЗУ с ультрафиолетовым стиранием, поэтому уступают последним по степени использования в микропроцессорной аппаратуре.

Основу запоминающей ячейки в ПЗУ с электрическим стиранием составляет МОП-транзистор с плавающим затвором, такой же, как и в ПЗУ с ультрафиолетовым стиранием. Но в микросхемах данного типа технологическими методами обеспечена возможность обратного туннелирования, т. е. отбора электронов с плавающего затвора, что позволяет выборочно стирать занесенную информацию. На рисунке 5.4, *в* представлена ячейка РПЗУ в виде лавинно-инжекционного МОП-транзистора с плавающим и селектирующим затворами. Интегральные ПЗУ на таких элементах допускают многократную замену хранимой информации. Плавающий (изолированный) затвор не имеет электрического подвода, он предназначен для хранения заряда.

Селектирующий затвор подсоединен к одному из выходов дешифратора строк (горизонтальной шине), а сток - к вертикальной шине. В исходном состоянии отсутствует заряд на плавающем затворе (состояние 1), транзистор имеет очень небольшое пороговое напряжение. Выбор элемента осуществляется путем подачи на селектирующий затвор выходного напряжения адресного дешифратора, при этом включается транзистор и через цепь сток-исток протекает значительный ток. Программирование (занесение лог. 0 в элементы) производится электрическим путем - подачей на сток импульса напряжения 25...50 В. При этом происходит инжекция электронов, имеющих высокую энергию, через оксид на изолированный затвор, получающий отрицательный заряд (состояние «0»). В результате увеличивается пороговое напряжение, и теперь подача на селектирующий затвор выходного напряжения дешифратора не включает этот транзистор. Сообщенное элементу состояние сохраняется сколь угодно долго. Для стирания информации, записанной в РПЗУ подается напряжение 25…50 В, но с обратной полярностью включения. При этом происходит рассасывание электронов с плавающего затвора.

5.1.4 Внешние запоминающие устройства

Внешние запоминающие устройства (ВЗУ) — это устройства большого объема (сотни миллионов байт и более), предназначенные для длительного хранения данных, создания архива данных (в том числе и на съемных носителях) и обмена этими данными с оперативными запоминающими устройствами БВМ. В общем случае ВЗУ состоит из накопителя и контролера.

Накопителем ВЗУ называется часть ВЗУ, непосредственно принимающая информацию, преобразующая ее (при необходимости) в форму, удобную для хранения, осуществляющая хранение этой информации и преобразование в электрические сигналы при считывании. Накопитель ВЗУ состоит из устройства преобразования информации, механизма (привода) и носителя информации.

Устройство преобразования информации включает усилители записи и считывания, магнитные головки (для магнитных ВЗУ) и другие средства, необходимые для прямого и обратного преобразования цифровых кодов в форму, удобную для хранения. Механизм (привод) обеспечивает перемещение носителя информации относительно органов записи и считывания.

Носителем информации является изделие, осуществляющее непосредственное хранение информации на основе определенных физических явлений. Носители информации — лента, диск, барабан, фотопленка, голограмма и т.п.

Контроллером ВЗУ называется устройство, обеспечивающее управление работой накопителя и сопряжение с интерфейсом БВМ.

Специфика ВЗУ заключается в его накопителе. Типичными накопителями ВЗУ являются:

накопитель на магнитной ленте (НМЛ);

накопитель на магнитных дисках (НМД);

накопитель на гибких магнитных дисках (НГМД);

накопитель на магнитных картах (НМК);

накопитель на цилиндрических магнитных доменах (НЦМД).

5.1.5 Способы организации памяти

Запоминающее устройство предназначено для хранения больших объемов двоичных данных, поэтому оно содержит множество одинаковых запоминающих элементов, образующих запоминающий массив (ЗМ). Массив разделен на отдельные ячейки, каждая из которых предназначена для хранения двоичного кода, количество разрядов в котором определяется шириной выборки памяти (в частности, это может быть одно, половина или несколько машинных слов).

Способ организации памяти зависит от методов размещения и поиска информации в запоминающем массиве. По этому признаку различают адресную, ассоциативную и стековую памяти.

Адресная память. В памяти с адресной организацией размещение и поиск информации в ЗМ основаны на использовании адреса хранения слова (числа, команды и т.п.), которым служит номер ячейки ЗМ, где размещено это слово.

При записи (или считывании) слова в ЗМ инициирующая эту операцию команда должна указать адрес (номер ячейки), по которому производится запись.

Типичная структура адресной памяти, показанная на рисунке 5.5, содержит запоминающий массив из N x n -разрядных ячеек и его аппаратное обрамление, включающее регистр адреса РгА, информационный регистр РгИ, блок адресной выборки (БАВ) с дешифратором адреса ДшА, блок усилителей считывания (БУС), блок разрядных

усилителей - формирователей сигналов записи (БУЗ) и блок управления памятью (БУП).

Разрядность регистра адреса определяется выражением

$$R \geq \mathrm{int}\left(\log_2 N\right).$$

По коду адреса в РгА БАВ формирует в соответствующей ячейке памяти сигналы, позволяющие произвести в ячейке считывание или запись слова.

Цикл обращения к памяти инициируется поступлением в БУП извне сигнала "Обращение". Общая часть цикла обращения включает прием в РгА с шины адреса (ША) адреса нужной ячейки и прием в БУП и расшифровку управляющего сигнала "Операция", указывающего вид запрашиваемой операции (считывание или запись).

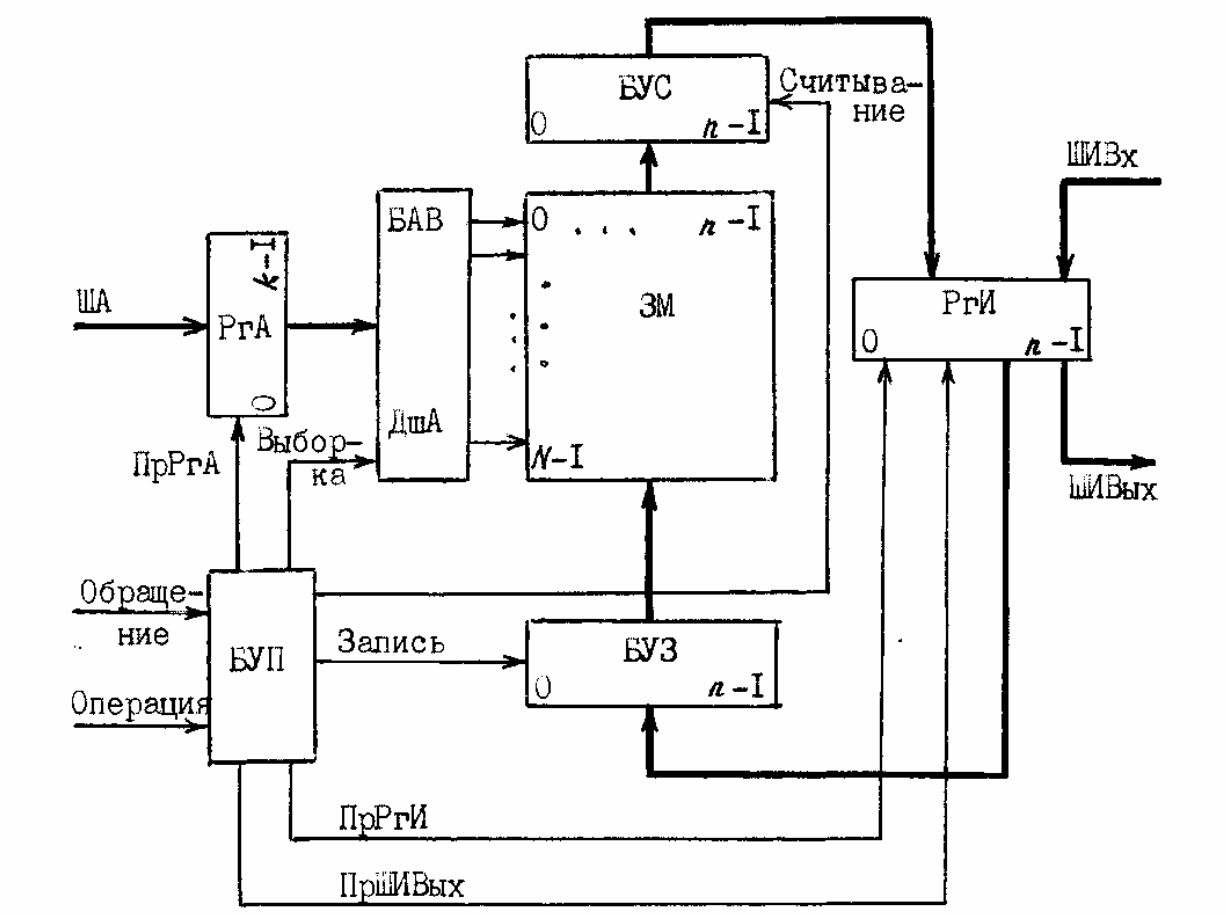


Рисунок 5.5- Структура адресной памяти

Далее при считывании БАВ дешифрирует адрес, посылает сигналы считывания в заданную адресом ячейку ЗМ. При этом код записанного в ячейке слова считывается усилителями считывания БУС и передается в РгИ.

Затем, в случае памяти с разрушающим считыванием (при считывании запоминающие элементы ячейки устанавливаются в нулевое состояние), производится регенерация информации в ячейке путем записи в нее из РгИ считанного слова. Операция считывания завершается выдачей слова из РгИ на выходную информационную шину ШИВых.

При записи после выполнения указанной выше общей части цикла обращения производится прием записываемого слова с входной информационной шины ШИВх в РгК. Сама запись состоит из двух операций: очистки ячейки (сброса в 0) и собственно записи. Для этого БАВ сначала производит выборку и очистку ячейки, заданной адресом в РгА. Очистка выполняется сигналами считывания слова в ячейке, но при этом блокируются усилители считывания, и из БУС в РгИ информация не поступает. Затем в выбранную БАВ ячейку записывается слово из РгИ.

Блок управления БУП генерирует необходимые последовательности управляющих сигналов, инициирующих работу отдельных узлов памяти.

Ассоциативная память. В памяти этого типа поиск нужной информации производится не по адресу, а по ее содержанию (по ассоциативному признаку). При этом поиск по ассоциативному признаку (или последовательно по отдельным разрядам этого признака) происходит параллельно во времени для всех ячеек запоминающего массива. Во многих случаях ассоциативный поиск позволяет существенно упростить и ускорить обработку данных. Это достигается за счет того, что в памяти этого типа операция считывания информации совмещена с выполнением ряда логических операций.

Типичная структура ассоциативной памяти представлена на рисунке 5.6. Запоминающий массив содержит $N \times (n+1)$ - разрядных ячеек. Для указания занятости ячейки используется служебный n -й разряд (0 - ячейка свободна, 1 - в ячейке записано слово).

По входной информационной шине ШИВх на регистр ассоциативного признака РгАП в разряды 0…(n-1) поступает n -разрядный ассоциативный запрос, а на регистр маски РгМ - код маски поиска, при этом n -й разряд РгМ устанавливается в 0. Ассоциативный поиск производится лишь для совокупности разрядов РгАП, которым соответствуют 1 в РгМ (незамаскированные разряды РгАП).

Для слов, в которых цифры в разрядах совпали с незамаскированными разрядами РгАП, комбинационная схема (КС) устанавливает 1 в соответствующие разряды регистра совпадения РгСв и 0 в остальные разряды. Таким образом, значение i -го разряда РгСв определяется выражением

$$РгСв(j) = \bigwedge_{i=0}^{i=n-1} \left\{ \overline{РгАП[i] \oplus ЗМ[j,i]} \vee \overline{РгМ[i]} \right\}, \quad 0 \le j \le N-1,$$

где РгАП[i] , РгМ [i] и ЗМ [j,i] - значения i -го разряда соответственно РгАП, РгМ и j -й ячейки ЗМ.

Комбинационная схема формирования результата ассоциативного обращения (ФС) формирует из слова, образовавшегося в РгСв, сигналы $\alpha_0, \alpha_1, \alpha_2$, соответствующие случаям отсутствия слов в ЗМ, удовлетворяющих ассоциативному признаку, наличию одного и наличию более чем одного такого слова.

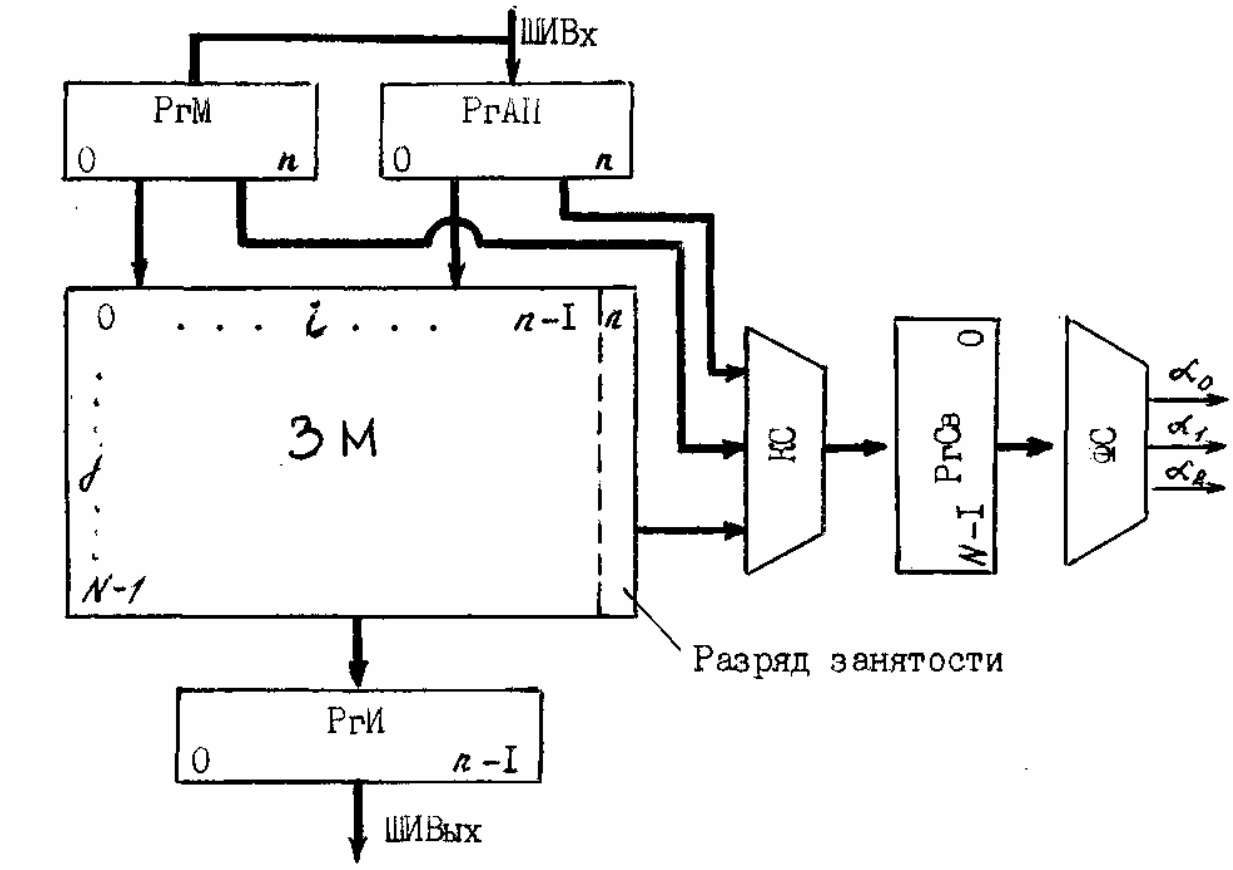


Рисунок 5.6 - Структура ассоциативной памяти

Для этого КС реализует следующие булевы функции:

$$\alpha_0 = \bigwedge_{j=0}^{N-1} \overline{РгСв[j]};$$

$$\alpha_1 = РгСв[0]\cdot\overline{РгСв[1]}\cdot\ldots\cdot\overline{РгСв[j]}\cdot\ldots\cdot\overline{РгСв[N-1]}\vee$$
$$\vee\overline{РгСв[0]}\cdot РгСв[1]\cdot\ldots\cdot\overline{РгСв[j]}\cdot\ldots\cdot\overline{РгСв[N-1]}\vee\ldots$$
$$\vee\overline{РгСв[0]}\cdot\overline{РгСв[1]}\cdot\ldots\cdot\overline{РгСв[j]}\cdot\ldots\cdot РгСв[N-1];$$

$$\alpha_2 = \overline{\alpha_1}\cdot\overline{\alpha_2}.$$

Формирование содержимого РгСв и сигналов $\alpha_0, \alpha_1, \alpha_2$ по содержимому РгАП, РгМ и ЗМ называется операцией контроля ассоциации. Эта операция является составной честью операций считывания и записи, хотя она имеет и самостоятельное значение.

При считывании сначала производится контроль ассоциации по ассоциативному признаку в РгАП. Затем при $\alpha_0 = 1$ считывание отменяется из-за отсутствия искомой информации, при $\alpha_1 = 1$ считывается в РгИ найденное слово, при $\alpha_2 = 1$ в РгИ считывается слово из ячейки, имеющей наименьший номер среди ячеек, отмеченных 1 в РгСв. Из РгИ считанное слово выдается на выходную шину ШИВых.

При записи сначала отыскивается свободная ячейка. Для этого выполняется операция контроля ассоциации при РгАП=111…10 и РгМ=000...01. При этом свободные ячейки отмечаются 1 в РгСв. Для записи выбирается свободная ячейка с наименьшим номером. В неё записывается слово, поступившее с ШИВх в РгИ.

Посредством операции контроля ассоциации можно, не считывая слов из памяти, определить по содержимому РгСв, сколько в памяти слов, удовлетворяющих ассоциативному признаку, например, реализовать запросы типа: сколько курсантов в группе имеют отличную оценку по данной дисциплине. При использовании соответствующих комбинационных схем в ассоциативной памяти могут выполняться достаточно сложные логические операции, такие как поиск большего (меньшего) числа, поиск слов, заключенных в определенных границах и др.

Отметим, что для ассоциативной памяти необходимы запоминающие элементы, допускающие считывание без разрушения запи-

санной в них информации. Это связано с тем, что при ассоциативном поиске считывание производится по всему ЗМ для всех незамаскированных разрядов и негде сохранять временно разрушаемую считыванием информацию.

Стековая память. Стековая память, также как и ассоциативная, является безадресной. Стековая память делится на два вида: FIFO – первым пришел, первым вышел и FILO – первым пришел и последним вышел. В стековой памяти FIFO (рисунок 5.7,а) ячейки образуют одномерный массив, в котором соседние ячейки связаны друг с другом разрядными цепями передачи слов. Стек заполняется с одной стороны, при этом слова записываются в свободные ячейки с последовательными номерами, начиная с верхней ячейки, а считывание с удалением слова из памяти производится с другой стороны стека и только из нижней ячейки (ячейки 0). Остальные слова в стеке сдвигаются вниз в соседние ячейки с меньшими номерами. Таким образом, порядок считывания подчиняется правилу: первым поступил - первым обслуживается.

В состав стековой памяти входит счетчик-указатель стека СчУСт. При записи нового слова в стек содержимое СчУСт увеличивается на единицу, при считывании уменьшается на единицу. При СчУСт = 0 вырабатывается сигнал УСт = 0, обозначающий, что стек пуст. При СчУСт $= N-1$ формируется сигнал УСт $N-1$, указывающий, что стек заполнен. Стековая память используется в ЦВМ для аппаратной организации различных очередей.

В стековой памяти FILO (рисунок 5.7,б) запись нового слова происходит в верхнюю ячейку. При этом все ранее записанные слова (включая слово, находившееся в ячейке 0) сдвигаются вниз, в соседние ячейки с большими на единицу номерами. Считывание возможно только из верхней (нулевой) ячейки стековой памяти. При этом если производится считывание с удалением, то все остальные слова в памяти сдвигаются вверх, в соседние ячейки с меньшими номерами. В этой памяти порядок считывания слов соответствует правилу: последним поступил - первым обслуживается. Так организована подача патронов из магазина автоматического стрелкового оружия, поэтому стековую память типа FILO называют также магазинной. Магазинная память снабжается счетчиком-указателем магазина, действие которого аналогично указателю стека. Магазинная память эффективна при реализации вложенных алгоритмов и системы прерываний.

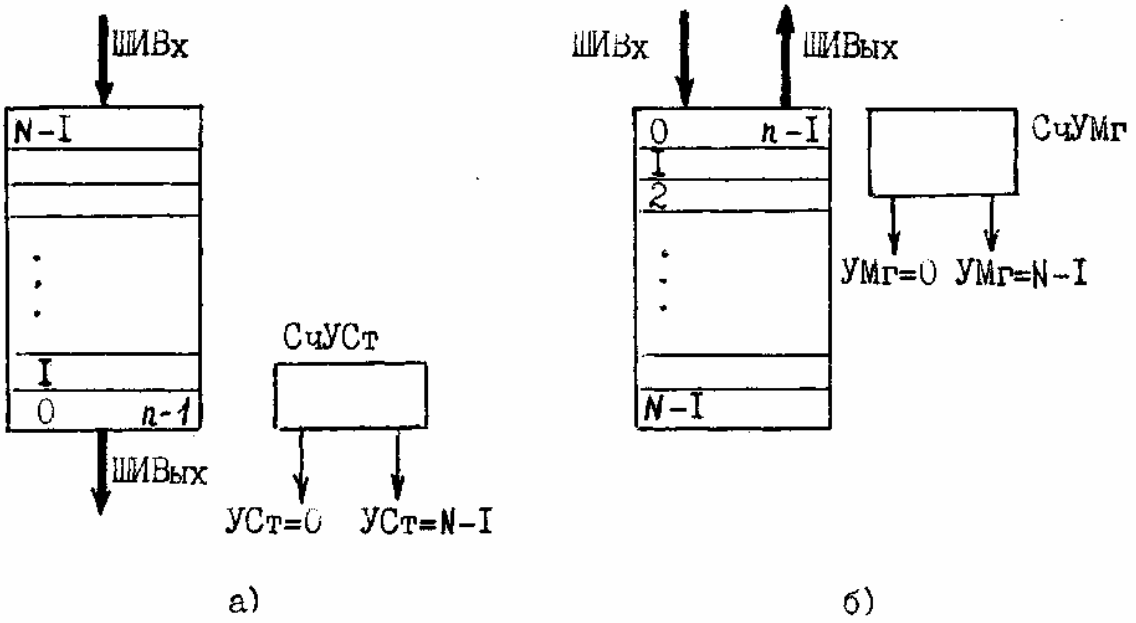


Рисунок 5.7- Стековая память типа FIFO (а) и типа FILO (б)

5.1.6 Виды адресации и способы их реализации в БЦВМ

Наиболее широкое применение в современных БЦВМ нашли ЗУ с адресной организацией. Это значит, что каждой хранимой единице информации ставится в соответствие специальное число – адрес, определяющий место ее хранения в памяти.

Сведения об адресе или его составных частях содержатся в команде. Пользуясь этой информацией, машина производит выборку из памяти требуемых программой данных. В свою очередь, сами адреса могут рассматриваться как данные, подвергаться в машине обработке и образовываться в результате вычислений.

Следует различать понятия: адресный код в команде (адресная часть команды) Ак и исполнительный адрес Аи. Адресный код – это информация об адресе, содержащемся в команде. Исполнительный адрес – это номер ячейки памяти, к которой производится фактическое обращение. В современных БЦВМ стремление к уменьшению длины кода команды и созданию удобств при программировании приводит к тому, что адресный код, как правило, не совпадает с исполнительным адресом.

Формирование исполнительного адреса может осуществляться различными способами. Наиболее часто используются следующие способы:

подразумеваемый операнд;
подразумеваемый адрес;
непосредственная адресация;
прямая адресация;
относительная адресация или базирование;
косвенная адресация;
укороченная адресация;
адресация слов переменной длины.

Подразумеваемый операнд. В этом случае в команде не содержится явных указаний об адресе операнда, операнд подразумевается и фактически задается кодом операции команды. Данный метод используется не часто, однако имеется несколько важных случаев его применения. В качестве примера можно привести команды подсчета, в которых к некоторому числу (содержимому счетчика) прибавляется фиксированное приращение, часто единица младшего разряда. Один из операндов – число в счетчике – обычно адресуется явным методом, второй операнд – приращение – не адресуется, в памяти машины не содержится и является подразумеваемым.

Подразумеваемый адрес. В команде не содержится явных указаний об адресе участвующего в операции операнда или адреса, по которому помещается результат операции, но этот адрес подразумевается. Например, команда может содержать адреса обоих операндов, участвующих в операции, при этом подразумевается, что результат операции помещается по адресу одного из операндов. Или в команде указывается только адрес одного операнда, а адрес второго, которым является содержимое специального регистра (называемого регистром результата), подразумевается.

Непосредственная адресация. В команде содержится не адрес операнда, а непосредственно сам операнд. При непосредственной адресации не требуется обращения к памяти для выборки операнда и ячейки памяти для его хранения. Это позволяет во многих случаях повысить быстродействие машины и уменьшить объем памяти, необходимый для данной задачи. Непосредственная адресация удобна для хранения различного рода констант. Однако в этом случае длина операнда должна быть меньше кода команды, поскольку часть разрядов команды занята под код операции.

Прямая адресация. Исполнительный адрес совпадает с адресной частью команды. Этот метод адресации был общепринятым в первых

вычислительных машинах и продолжает применяться в настоящее время в комбинации с другими методами.

Относительная адресация или базирование. Исполнительный адрес определяется суммой адресного кода команды Ак и некоторого числа Аб, называемого базовым адресом:

Аи = Ак + Аб.

Для хранения базовых адресов в машине могут быть предусмотрены регистры или специально выделенные для этой цели ячейки памяти (базовые регистры). Количество разрядов в базовом адресе выбирают таким, чтобы совместно с адресной частью команды можно было адресовать любую ячейку памяти. Для указания места хранения базового адреса в команде может быть отведено соответствующее поле В.

При использовании базирования с целью экономии разрядов адресный код команды Ак обычно делают короче базового адреса и называют его смещением (обозначают буквой С). Смещение С определяет положение операнда относительно начала массива, задаваемого базовым адресом Аб. Рисунок 5.8,а поясняет процесс формирования исполнительного адреса.

Большей частью исполнительный адрес при базировании образуется посредством суммирования согласно выражению (рисунок 5.8,б)

$$\text{Аи} = \begin{cases} <(B)> + (C), & если (B) \neq O; \\ (C), & если (B) = O, \end{cases}$$

где (В) и (С) – коды (числа), стоящие в полях В и С команды, <(В)> - содержимое регистра с номером (В). При (В) = 0 относительная адресация блокируется.

Суммирование при образовании Аи связано с потерей времени. Поэтому применяют также формирование исполнительного адреса методом совмещения (склеивания) (рисунок 5.8,*в*). В этом случае базовый адрес содержит старшие, а смещение – младшие разряды исполнительного адреса, которые объединяются в РгА согласно операции

РгА: = РгВ * РгК [С] ,

где * - знак объединения (склеивания); РгК [С] – разряды команды, соответствующие смещению.

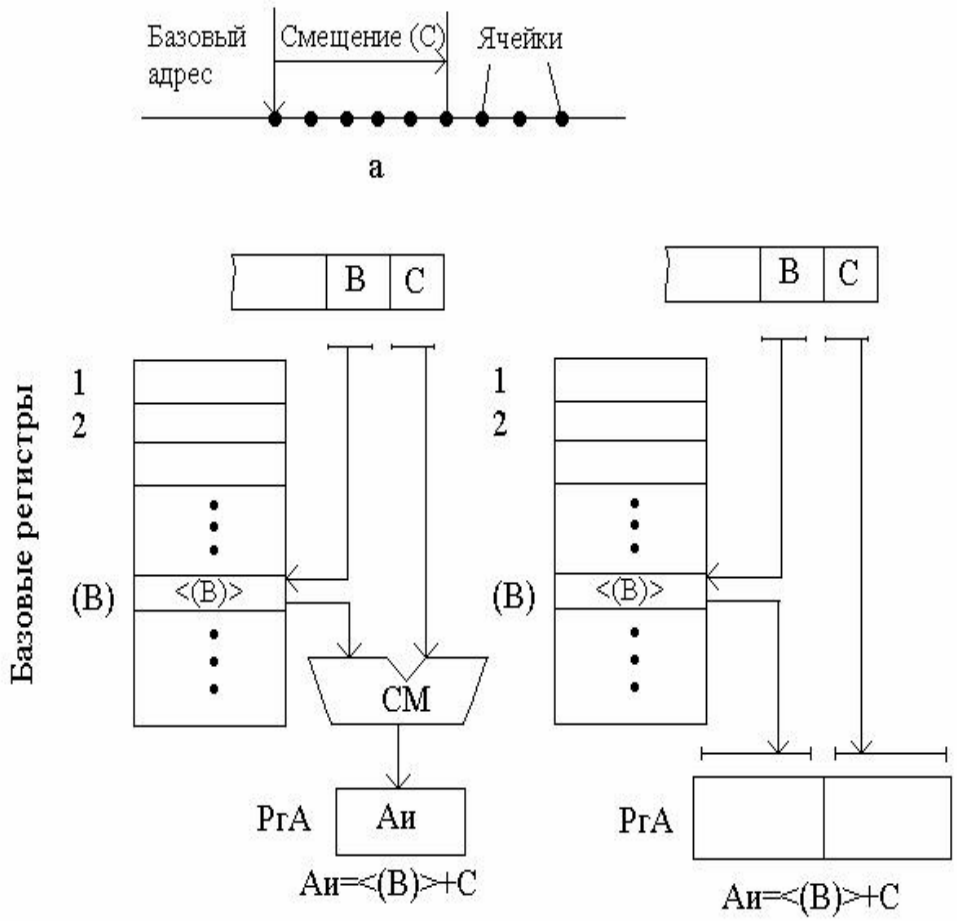


Рисунок 5.8 - Относительная (базовая) адресация:

а- образование адреса элемента одномерного массива; б- формирование исполнительного адреса при базировании суммированием; *в*-формирование исполнительного адреса совмещением (склеиванием)

Относительная адресация обеспечивает так называемую «перемещаемость» программ, т.е. возможность передвижения программ в памяти без изменений внутри самой программы.

Относительная адресация облегчает распределение памяти при составлении несколькими программистами сложных программ. Каждый программист может располагать свою часть программы в памяти, начиная с нулевого адреса. При компоновке программы достаточно только задать соответствующие значения базовых адресов.

Косвенная адресация. Адресный код команды указывает адрес ячейки памяти, в которой находится адрес операнда или команды. Таким образом, косвенная адресация может быть иначе определена как «адресация адреса».

Обращение к ячейке памяти за операндом при косвенной адресации выполняется в несколько этапов. На первом этапе формируется

исполнительный адрес ячейки памяти, где хранится адрес операнда. На втором этапе адрес операнда извлекается из ячейки и пересылается в регистр адреса РгА. И на третьем этапе происходит извлечение операнда по адресу, находящемуся в РгА.

Укороченная адресация. Для уменьшения длины кода команды часто применяется так называемая укороченная адресация. Суть ее сводится к тому, что в команде задаются только младшие разряды адресов. Старшие разряды при этом подразумеваются нулевыми. Такая адресация позволяет использовать только небольшую группу фиксированных ячеек с короткими адресами и поэтому может использоваться лишь совместно с другими методами адресации.

Адресация слов переменной длины. Эффективность вычислительных систем, предназначенных для обработки данных, повышается, если имеется возможность выполнять операции со словами переменной длины. В этом случае в машине должна быть предусмотрена адресация слов переменной длины, которая обычно реализуется указанием в команде местоположения в памяти начала слова и его длины.

Из изучаемых вычислительных машин наиболее развита система адресации БВМ А-15А. В А-15А формирование исполнительного адреса может осуществляться восемью различными способами (таблица 5.1), определяемыми признаком адресации (ПА).

Таблица 5.1 – Способы формирования исполнительного адреса БВМ А-15А

Разряды ПА			Схема формирования исполнительного адреса
А	Б	В	
0	0	0	Аи = С
0	0	1	Аи = РгБ1*С
0	1	0	Аи = П[С]
0	1	1	Аи = П[РгБ2*С]
1	0	0	Аи = СчК ± С
1	0	1	Аи = РгИ1 + С
1	1	0	Аи = РгИ2 + С
1	1	1	Аи = РгБ3*С

В таблице 5.1 использованы следующие обозначения базовых констант адресации, заносимых в регистры БВМ:

РгБ1, РгБ2, РгБ3 – содержимое 6-разрядных базовых регистров;

РгИ1, РгИ2- содержимое 16-разрядных индексных регистров;

СчК – содержимое 16-разрядного счетчика команд;

П[С] – содержимое ячейки с адресом С;

П[Б2*С]- содержимое ячейки с адресом РгБ2*С, где * - символ объединения (склеивания) двух кодов в одно 16-ти разрядное слово.

В соответствии с форматом адреса, приведенным в таблице 5.2, и распределением поля адресов памяти (таблица 5.3) формирование адреса осуществляется следующим образом.

Таблица 5.2 – Формат адреса БЦВМ А-15А

Разряды слова	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Триады	0-я			1-я			2-я	3-я			4-я			5-я		
Назначение	Номер блока						номер ячейки памяти в блоке									

При ПА = 000 исполнительный адрес равен смещению, непосредственно записанному в команде, причем его старшие 6 разрядов равны нулю. С учетом распределения поля адресов это означает прямую адресацию к ячейкам ОЗУ1.

При ПА = 001 исполнительный адрес образуется «склеиванием» 6-разрядного кода РгБ1 с 10-разрядным смещением команды. Код базы занимает старшие 6 разрядов, смещение - младшие 10 разрядов исполнительного адреса.

При ПА = 010 исполнительный адрес является содержимым ячейки, адрес которой равен смещению, записанному в команде. С учетом распределения поля адресов смещение представляет собой косвенный адрес и указывает адрес ячейки памяти в блоке ОЗУ1, чье содержимое представляет собой исполнительный адрес.

При ПА = 011 исполнительный адрес является содержимым ячейки памяти, адрес которой равен РгБ2*С, т.е. исполнительный адрес может находиться в любой ячейке памяти, как долговременной, так и в оперативной.

При ПА = 100 исполнительный адрес образуется как сумма содержимого счетчика команд и смещения. В данном случае операция суммирования производится только над младшими десятью разрядами адреса (содержимого счетчика команд). При возникновении переноса в следующий (одиннадцатый), разряд он отбрасывается. Поэтому операция может рассматриваться и как вычитание из содержимого СчК величины смещения. В результате операции получается 16-

разрядный исполнительный адрес, шесть старших разрядов которого, определяющих блок ЗУ, остаются неизменными. Этот вид адресации удобно применять для организации циклов.

При ПА = 101 исполнительный адрес образуется как сумма содержимого РгИ1 и смещения.

При ПА = 110 исполнительный адрес образуется как сумма содержимого РгИ2 и смещения.

При ПА = 111 исполнительный адрес образуется «склеиванием» шестиразрядного РгБЗ и смещения.

Следует отметить, что без косвенных и относительных видов адресации можно обратиться лишь к 1024 ячейкам памяти ОЗУ1.

Таблица 5.3 – Распределение поля адресов памяти БЦВМ А-15А

Код шести старших разрядов	Восьмеричная запись	Наименование блока	Количество ячеек
000000	00	ОЗУ1	1024
000001	01	ОЗУ2	1024
000010	02	Свободная зона	1024
000011	03	Свободная зона	1024
000100	04	Свободная зона	1024
000101	05	Свободная зона	1024
000110	06	ДЗУС	128
000111	07	Накопитель пульта контроля	8
001xxx	1X	Свободная зона	8192
010xxx	2X	Свободная зона	8192
011xxx	3X	Свободная зона	8192
100xxx	4X	ДЗУ1	8192
101xxx	5X	ДЗУ2	8192
110xxx	6X	ДЗУ3	8192
111xxx	7X	ДЗУ4	8192

В ЦВМ типа «Орбита» применяется прямая адресация к блокам ОЗУ и относительная адресация к блокам ДЗУ. Для реализации относительной адресации используются регистры, именуемые переключателями квадратов. Соответственно, имеются переключатели квадратов команд, объединенные со счетчиком команд (СКП), и переключатель квадратов числа (ПЧ).

В ЦВМ типа Ц-100 используется прямая, относительная и косвенная адресация, как на уровне команд, так и на уровне микрокоманд. Кроме этого используется регистровая адресация, при использовании которой формируется адрес не ячейки памяти, а адрес одного из регистров процессора.

5.2 Процессоры БВМ

5.2.1 Основные сведения о процессорах БВМ

Процессором называется устройство, непосредственно осуществляющее процесс обработки данных и программное управление этим процессом.

Процессор дешифрирует и выполняет команды программы, организует обращения к оперативной памяти, в нужных случаях инициирует работу каналов и периферийных устройств, воспринимает и обрабатывает запросы, поступающие из устройств машины и из внешней среды (так называемые «запросы на прерывание»). По выполняемым функциям процессор является центральным устройством ЦВМ, управляющим взаимодействием её составных частей.

Выполнение процессором программы решения задачи представляет собой последовательность операций, осуществляемых в заданном программой порядке. К таким операциям относятся арифметические и логические операции, операции, связанные с анализом получающихся результатов, с организацией и управлением процессом вычислений. Обычно каждой такой операции соответствует команда.

В свою очередь операция (выполнение команды) может быть разделена на более мелкие этапы (микрооперации), во время которых исполняются определенные элементарные действия. Последовательность микроопераций, реализующих данную операцию (команду), образует микропрограмму операции. Для характеристики временных соотношений между различными этапами операции используется понятие машинный такт. Он определяет интервал времени $T = t_i - t_{i-1}$, в течение которого выполняется одна или одновременно несколько микроопераций процессора. Таким образом, может быть установлена следующая иерархия этапов выполнения программы в процессе вычислений: программа, команда (микропрограмма), микрооперация.

Структура процессора показана на рисунке 5.9. В его состав входят устройство управления (УУ) и арифметико-логическое устройство (АЛУ).

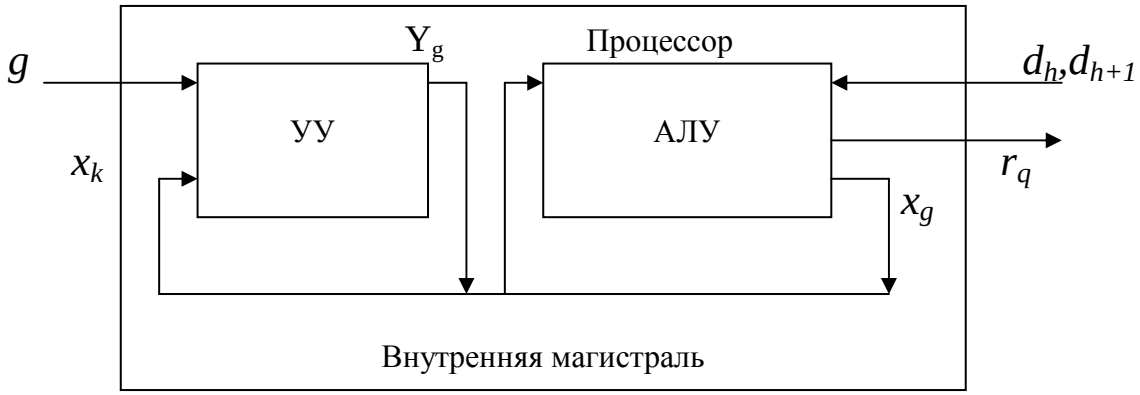


Рисунок 5.9 - Структурная схема процессора

Рассмотрим работу процессора при выполнении в АЛУ машинной операции $f_g \in F$ из состава системы команд процессора. Операндами этой операции являются слова d_h и d_{h+1}, хранящиеся в ОЗУ. Результат операции $r_q = f_g(d_h, d_{h+1})$ также записывается в ОЗУ.

На вход УУ поступает код g заданной операции и осведомительный сигнал x_k, сформированный в АЛУ при выполнении предыдущей операции. На основе этой информации УУ генерирует последовательность $Y_g = \{Y(t_1), Y(t_2), \ldots, Y(t_g)\}$. Каждый элемент $Y(t_i)$ этой последовательности есть ансамбль управляющих сигналов в такте t_i, под воздействием которых выполняется одна микрооперация (или несколько совместимых микроопераций). Одним из результатов этих микроопераций является значение осведомительного сигнала x_i, от которого зависит очередной набор управляющих сигналов $Y(t_{i+1})$. Под воздействием последовательности управляющих сигналов Y_g происходит извлечение операндов d_h и d_{h+1} из ОЗУ, выполнение над ними заданной операции f_g и запись результата r_q в ОЗУ. В последнем такте t_g генерируется ансамбль управляющих сигналов $Y(t_g)$, одним из элементов которого будет осведомительный сигнал конца операции x_g.

Основным элементом, обеспечивающим организацию вычислительного процесса, является устройство управления.

5.2.2 Классификация и основные характеристики арифметико-логических устройств

В общем случае арифметико-логическое устройство служит для выполнения над операндами машинных операций, заданных системой команд БЦВМ. Обычно операндами машинных операций являются машинные слова или отдельные поля этих слов (байты и биты). При необходимости повысить точность вычислений машинные операции выполняются над словами двойной длины.

К арифметическим операциям относятся сложение, вычитание, вычитание модулей («короткие операции»), умножение и деление («длинные операции»). Группу логических операций составляют операции дизъюнкция (логическое сложение) и конъюнкция (логическое умножение) над многоразрядными двоичными словами, а также сравнение кодов на равенство. Специальные арифметические операции включают нормализацию, арифметический сдвиг (сдвигаются только разряды мантиссы числа, знаковый разряд остается на места), логический сдвиг (знаковый разряд сдвигается вместе с цифровыми разрядами).

Классификация АЛУ может быть произведена по следующим признакам.

По способу действия над операндами АЛУ делятся на последовательные и параллельные. В последовательных АЛУ операнды представляются последовательным кодом, а операции производятся последовательно во времени над их отдельными разрядами. В параллельных АЛУ операнды представляются параллельным кодом и операции совершаются параллельно во времени над всеми разрядами операндов.

По способу представления чисел различают АЛУ для чисел с фиксированной запятой, для чисел с плавающей запятой и для десятичных чисел.

По способу организации работы различают асинхронные и синхронные АЛУ. В асинхронных АЛУ определяется момент фактического окончания операции, после чего немедленно может начаться следующая операция. В синхронных АЛУ независимо от фактической продолжительности операции, определяющейся значениями операндов, на выполнение отдельных операций отводится фиксированное время.

По характеру использования элементов и узлов АЛУ делятся на блочные и многофункциональные. В блочном АЛУ для реализации ряда операций (умножение и деление, операций с адресами и т.д.) используются отдельные блоки. При этом повышается скорость работы, так как блоки могут работать параллельно, но при этом значительно возрастают затраты оборудования. В многофункциональных АЛУ все операции выполняются одними и теми же схемами, которые коммутируются нужным образом в зависимости от требуемого режима работы.

Основные характеристики АЛУ изучаемых БЦВМ приведены в таблице 5.4.

Таблица 5.4 - Основные характеристики АЛУ

	Характеристики АЛУ				
	Назначение	Характер использования элементов	Способ обработки	Количество команд	Быстродействие оп/с (R-R)
Орбита 10	Выполнение действий над операндами	много-функциональный	последовательный	63	125000
А-15А	Выполнение действий над операндами; формирование исполнительных адресов; выполнение модификации команд	много-функциональный	параллельный	71	208000
Ц - 100	Выполнение действий над операндами	блочный	параллельный	256	250000 (800000)

5.2.3 Выполнение арифметических и логических операций в арифметико-логическом устройстве

АЛУ БВМ выполняет обычно следующие основные стандартные и специальные арифметико-логические операции:

арифметическое сложение и вычитание;

арифметический, логический и циклический сдвиги;

логическое сложение и умножение, сложение по модулю два;

поиск самой левой (правой) единицы в слове и массиве данных;

запись единицы или нуля в разряд слова и массива данных;

арифметические и логические операции с частью слова данных.

Реализация арифметико-логических операций осуществляется АЛУ, структурная схема которого представлена на рисунке 5.10.

Первый *А* и второй *Б* операнды через шины и поступают соответственно в регистры РА и РБ АЛУ. Дальнейшее прохождение операндов к узлам АЛУ и действия над ними зависят от кода операции, задаваемого управляющими сигналами $y_{i_1} ... y_{i_m}$, поступающими из устройства управления (УУ). Эти сигналы и синхросерии *С(1),...,С(К)* поступают в узел местного управления (УМУ), который вырабатывает и распределяет последовательность управляющих сигналов для выполнения заданной операции.

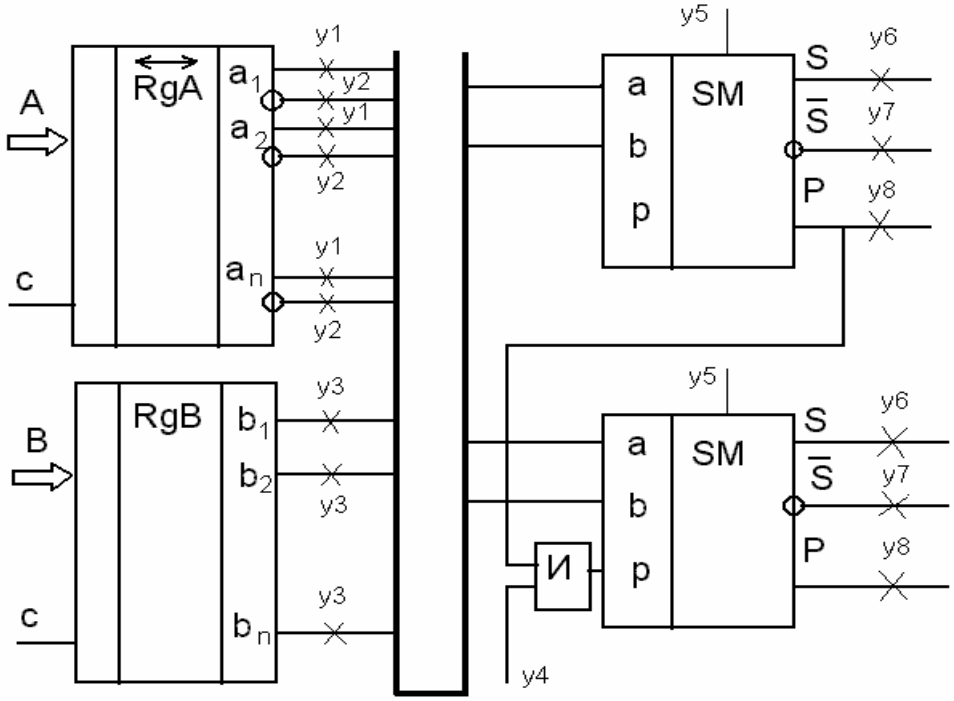


Рисунок 5.10

Выполнение операций арифметического сложения осуществляется параллельно по всем разрядам. Для этого в АЛУ используется n схем одноразрядных комбинационных сумматоров (см. п. 3.5.3). Операция вычитания реализуется через сложение чисел, одно из которых представляется дополнительным кодом отрицательного числа. Операции умножения и деления осуществляются по алгоритмам, рассмотренным в п. 1.3.3. Наибольший интерес представляет рассмотрение процесса выполнения логических операций.

Система команд БЦВМ предусматривает такие логические операции, как логическое умножение (И), логическое сложение (ИЛИ) и сложение по модулю два (ИСКЛЮЧАЮЩЕЕ ИЛИ). Реализация операций логического умножения и сложения осуществляется специальными комбинационными схемами, выполняющими поразрядную логическую обработку операндов по правилам булевой алгебры:

$$z_i = x_i y_i = x_i \,\&\, y_i, \quad z_i = x_i \vee y_i.$$

Однако можно использовать и внутренние цепи комбинационного сумматора для реализации логических операций. Для реализации поразрядного логического умножения необходимо запретить перенос из предыдущих разрядов, а результат снимать с выхода переноса сумматора

$$z_i = y_8 p_i = a_i + b_i + (\overline{y_4} p_{i-1}) = a_i \,\&\, b_i.$$

Операция логического сложения реализуется следующим образом $\overline{S} = y_5(y_2 \overline{a} \cdot \overline{b}) \vee \overline{y_5}(a \cdot b) = \overline{a \vee b}$, что следует из выражения для суммы в полусумматоре $S = a \cdot \overline{b} \vee \overline{a} \cdot b$.

Реализация операции сложения по модулю два осуществляется, как правило, сумматором. При этом «закрываются» цепи переносов сумматора, что обеспечивает $p_i = 0, \quad i = \overline{0, n-1}$, и получение

$$z_i = x_i \oplus y_i = x_i \overline{y}_i \vee \overline{x}_i y_i.$$

Разновидностью арифметических и логических операций являются операции сдвигов. Различают операции арифметического, логического и циклического сдвигов первого операнда влево или вправо на заданное вторым операндом число разрядов.

При арифметическом сдвиге первый операнд является целым числом со знаком. При выполнении этой операции значение знакового разряда сдвигаемого числа не изменяется, а значения остальных разрядов перемещаются влево или вправо на заданное число разрядов. При этом значения разрядов, сдвигаемые за пределы значащей части числа, теряются, а в разряды, освобождающиеся после сдвига, заносятся нули.

Обозначим арифметический сдвиг операнда x влево и вправо на y разрядов, соответственно, $z = x \leftarrow \frac{y}{A}{}_p$ и $z = x \frac{y}{A}{}_p \rightarrow$. Тогда имеет место

$$z = x \leftarrow \frac{y_p}{A} = x \cdot 2^y, \quad z = x\frac{y_p}{A} \rightarrow = x \cdot 2^{-y}.$$

При логическом и циклическом сдвигах первый операнд является полноразрядным логическим кодом, поэтому соответствующие операции производятся над всеми разрядами операнда аналогично арифметическому сдвигу значащей части целого числа. При циклическом сдвиге потери значений разрядов, сдвигаемых за пределы разрядной сетки операнда, не происходит, так как значения этих разрядов заносятся в разряды, освобождающиеся после сдвига.

Операции сдвига осуществляются сдвиговым регистром или за счет передачи данных с выхода обычного регистра на вход сумматора так называемым косым кодом т.е. со смещением на один разряд влево или вправо.

5.2.4 Устройство управления со схемной организацией

Устройство управления (УУ) вырабатывает необходимые управляющие сигналы для выборки очередной команды из памяти, дешифрации кода команды, формирования адресов операндов, выборки операндов из памяти, передачи их в АЛУ, выполнения в АЛУ операции, предусмотренной кодом команды, передачи полученного в АЛУ результата операции в память, инициирования работы каналов, организации реакции процессора на запросы прерывания, поступающие от узлов самого процессора, других устройств ЭВМ и внешней среды.

Для реализации перечисленных функций в состав УУ обычно входят следующие узлы (блоки):

узел управления операциями;

узел дешифрации;

узел формирования адреса;

узел управления блоками памяти;

узел обмена и реакции на запросы внешних устройств;

узел синхронизации.

Узел управления операциями предназначен для выработки последовательности управляющих сигналов, обеспечивающих выполнение команды (микрокоманды). Распознавание конкретной команды осуществляется узлом дешифрации по соответствующему данной команде коду операции.

Узел формирования адреса обеспечивает формирование исполнительного адреса операндов и команд в режиме естественного выполнения команд и выдает исполнительные адреса команд и чисел в узел управления блоками памяти и в узел обмена.

Узел управления блоками памяти предназначен для выбора блока памяти и ячейки в выбранном блоке соответственно исполнительному адресу, формирования сигналов обращения к блоку при считывании и записи кодов и пересылки считанных кодов из памяти в различные устройства ЭВМ.

Узел обмена обеспечивает обмен информацией с внешними устройствами, в том числе и в случаях, когда инициаторами такого обмена являются внешние устройства, т.е. при обмене по прерываниям.

По способу выполнения команд в ЭВМ различают синхронный и асинхронный принципы управления. В УУ, реализующем принцип синхронного управления, длительность машинного цикла выбирается постоянной, одинаковой для всех реализуемых операций и равной длительности выполнения самой продолжительной операции. В каждом цикле вырабатывается одинаковое число синхронизирующих сигналов. Нормирование синхронизирующих сигналов осуществляет узел синхронизации.

В настоящее время существует два варианта формирования управляющих сигналов (УС), предназначенных для выполнения операций машинного языка (системы команд). В зависимости от способа формирования управляющих сигналов различают устройства управления БВМ со схемным формированием УС (УУ с жесткой логикой) и УУ с программным формированием УС (микропрограммные устройства управления).

Обобщенная структурная схема управляющего устройства с жесткой логикой показана на рисунке 5.11. В состав схемы входят регистр кода операции, являющийся частью регистра команд, счетчик тактов, дешифратор тактов и дешифратор кода операции, а также логические схемы образования управляющих функциональных сигналов. На счетчик тактов поступают сигналы от блока синхросигналов, и счетчик с каждым сигналом меняет свое состояние. Состояния счетчика представляют номера тактов, изменяющиеся от 1 до n. Дешифратор тактов формирует сигнал на j -м выходе, если код, поступающий на его вход, соответствует двоичному номеру j -того синхросигнала.

Разрядность входных линий дешифратора кода операции соответствует числу разрядов, отводимых в формате команды под код операции. Для m- разрядного кода операции дешифратор обеспечивает формирование 2^m выходных сигналов. При этом активное состояние выходного сигнала (логический ноль или логическая единица, в зависимости от принятой логики) возможно лишь на той линии, двоичный номер которой соответствует двоичному коду операции. Логические схемы образования управляющих функциональных сигналов для каждой команды возбуждают формирователи функциональных сигналов, обеспечивающие выполнения требуемых в данном такте микроопераций.

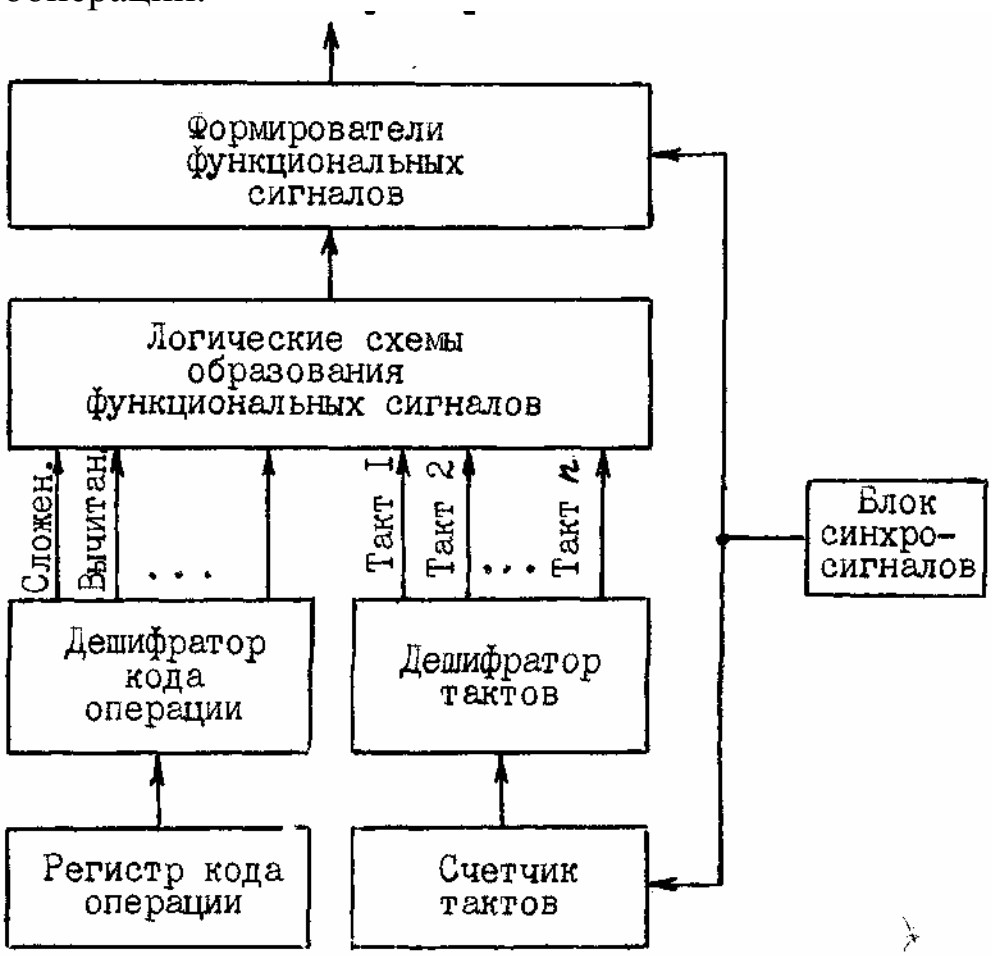


Рисунок 5.11 - Структура управляющего устройства с жесткой логикой

Синтез формирователя функциональных сигналов осуществляется по известной методике синтеза комбинационных схем. При этом с целью повышения быстродействия выполняется оптимизация разрабатываемых схем по критерию минимума времени формирования

сигналов при минимальном составе используемого оборудования. Для обеспечения минимального времени выполнения команды стараются совместить во времени основные этапы ее реализации. Обычно все этапы делят на две группы: этапы подготовки и этапы выполнения команды.

Подготовка к выполнению команды включает в себя:

этап формирования адреса команды;

этап выборки команды из ЗУ команд;

этап формирования адреса операнда (или нескольких адресов операндов).

Этапами выполнения являются: этап формирования сигналов исполнения операции и этап формирования осведомительной информации, в том числе и об окончании выполнения команды. Устройство управления обычно формирует для каждого из этапов соответствующий тактовый сигнал, который жестко привязан к основной последовательности синхроимпульсов.

Устройство управления бортовой ЦВМ А-15А обеспечивает формирование следующих основных тактовых сигналов: выборки команды (ВК), формирования адреса операнда (ФА), выборки числа (ВЧ), записи числа (ЗЧ) и исполнения операции (ИО).

В такте ВК осуществляется формирование исполнительного адреса команды, считывание её из ЗУ команд и пересылка команды на регистр команд (РгК). В зависимости от способа адресации формирование адреса операнда может происходить за один такт (прямая и относительная адресация) либо за два такта (косвенная).

В соответствии с кодом операции осуществляется формирование тактовых сигналов ВЧ и ЗЧ при выполнении команд, предполагающих обращение к ЗУ. Если обращение к ЗУ не требуется, формируется такт ИО.

На рисунке 5.12 показана последовательность включения тактов. При использовании косвенной адресации в такте ФА1 формируется адрес ячейки, в которой находится исполнительный адрес операнда. Извлечение исполнительного адреса осуществляется по специальному тактовому сигналу - выборка адреса (ВА).

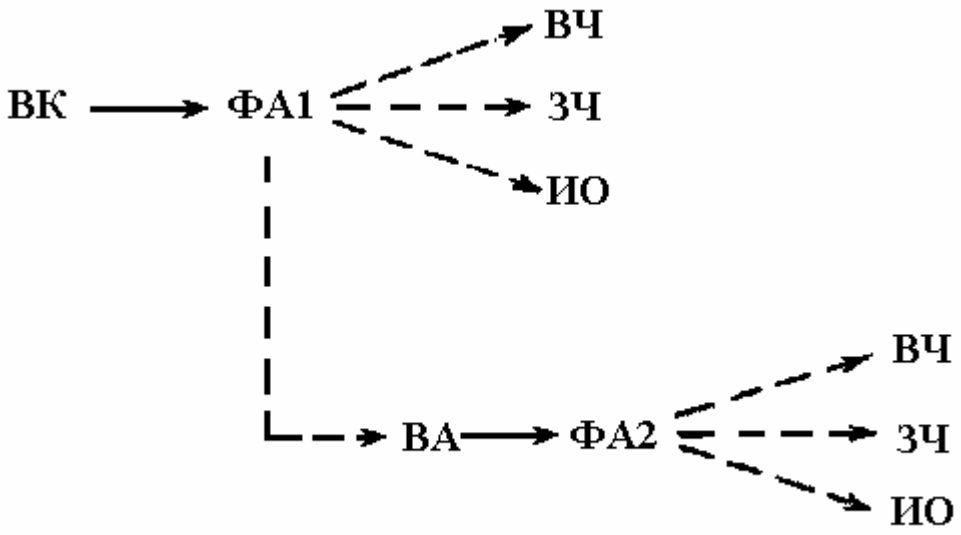


Рисунок 5.12 - Последовательность включения тактов

По окончанию такта ВА исполнительный адрес передается в узел приема информации, а именно в регистр памяти (РгП). Дальнейшая его передача в регистр адреса числа (РгАЧ) осуществляется в такте ФА2. Формирование тактовых сигналов осуществляется соответствующими одноразрядными регистрами.

Основная последовательность синхронизирующих импульсов, представляющих один машинный такт, состоит из двенадцати импульсов (1СИ, 2СИ,...,12СИ). В БЦВМ А-15А выборка и исполнение операции происходит за два машинных такта, но исполнение N- ой операции (команды) совмещается с выборкой следующей (N +1), и поэтому короткие операции выполняются практически за время одного машинного такта. Такой режим работы называется режимом совмещенного такта. Несовмещенный такт (НСТ) формируется при обращении за числом и следующей командой к одному блоку памяти, операция в этом случае выполняется за два машинных такта. При косвенной адресации время выполнения операции также увеличивается на такт выборки адреса.

Сложность устройства управления определяется структурой последовательностей управляющих сигналов $Y = \{Y_{n-1}, Y_{n-2}, ..., Y_0\}$. Чем сложнее эти последовательности, тем сложнее УУ с жесткой логикой. Кроме того, заложенная при проектировании структура схемного УУ изменению не подлежит, что не допускает никаких изменений в системе команд машины. Достоинство схемной организации управления состоит в том, что для известных алгоритмов управления можно по-

строить оптимальные в отношении быстродействия и объема оборудования УУ.

5.2.5 Микропрограммная организация управления

В этом случае каждой микрооперации или набору совместимых микроопераций ставится в соответствие микрокоманда, то есть код, содержащий полную информацию о векторе управляющих сигналов. А для каждой команды разрабатывается микропрограмма, состоящая из последовательности микрокоманд. Для всех блоков вычислительной машины формируется множество элементарных микроопераций, выполняемых в данном блоке. Все микрооперации кодируются, и в состав блоков вводятся дешифраторы микрокоманд, обеспечивающие опознавание каждой микрокоманды по её коду и формирование соответствующих ей управляющих сигналов. Так как большинство блоков ЭВМ могут работать параллельно, то на эти блоки управляющие микрокоманды подаются одновременно. Это обеспечивает большие, по сравнению со схемой жесткой логики, возможности по совмещению этапов выполнения команды.

Порядок выполнения программ при микропрограммном принципе управления вычислительным процессом рассмотрим по упрощенной структурной схеме БЦВМ Ц-100 (рисунок 5.13).

В состав центрального процессора (ЦП) БЦВМ Ц-100 входят:

блок арифметический логический (БАЛ);

блок арифметический умножения-деления (БАУД);

блок управления программой (БУП);

блок управления микропрограммой (БУМ);

блок управления данными (БУД);

блок оперативной памяти (БЗО);

блок коммутатора данных (БКД).

Обмен информацией между блоками осуществляется по четырем основным магистралям: магистрали команд и констант (LE), магистрали микрокоманд (LF), магистрали адресов (LA) и магистрали данных (LD).

Блоки БАЛ и БАУД вместе составляют арифметико-логическое устройство и используются для выполнения арифметических и логических операций над битами, байтами, словами одинарной и двойной длины.

В блоке управления программой производится формирование адресов, хранение и выдача в магистраль LE элементов программ и констант. Выдача может производиться как пословно, так и побайтно.

Блок управления микропрограммой предназначен для выдачи в ЦП кода микрокоманды, а также формирования адреса следующей микрокоманды.

Блок управления данными предназначен для формирования адресов данных, обрабатываемых процессором. Система микрокоманд блока позволяет производить также некоторые арифметические действия над данными.

Блок оперативной памяти представляет собой запоминающее устройство с произвольной выборкой емкостью 4К 16-разрядных слов. Блок допускает как пословную, так и побайтную запись и чтение информации.

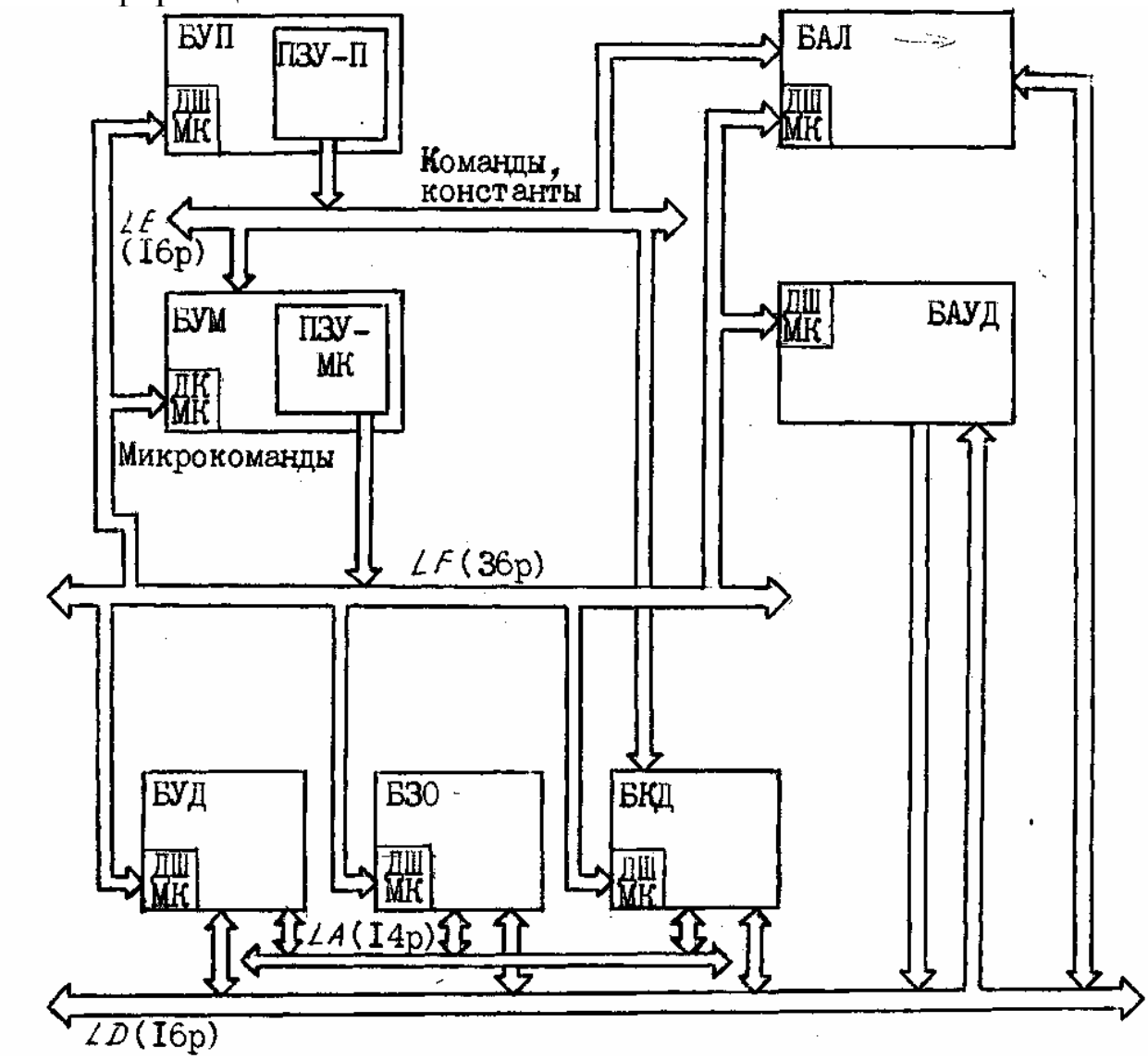


Рисунок 5.13 - Структурная схема центрального процессора БЦВМ Ц-100

Блок коммутатора данных выполняет операции по запоминанию и восстановлению содержимого магистралей LA , LD , LE, а также позволяет производить коммутацию байтов информации, передаваемых по магистрали LD .

Адрес первой команды (пусковой адрес) программы формируется в БУП аппаратурно или может быть записан с пульта контроля. По этому адресу из ПЗУ программ извлекается команда и выдается в магистраль LE. Блок управления микрокомандами считывает код команды из магистрали LE и формирует по нему первый адрес подпрограммы (микропрограммы), обеспечивающей реализацию данной команды. По адресу первой микрокоманды из ПЗУ микрокоманд считывается микрокоманда и выдается в магистраль микрокоманд.

В общем случае формат микрокоманды любого блока имеет вид, приведенный на рисунке 5.14.

Рисунок 5.14 - Обобщенный формат микрокоманд

На рисунке 5.14 приняты следующие обозначения: КСБ - поле кода структуры блока, т.е. группа разрядов, каждый из которых отнесен к определенному блоку. Если разряд КСБ для данного блока равен единице, то данная микрокоманда выполняется блоком; КОп - поле кода операции; ДКОп - поле расширения кода операции; А - адрес регистра местной памяти или блока ОЗУ; НО - поле непосредственного операнда.

В зависимости от кода операции и типа блока некоторые поля микрокоманды блока могут отсутствовать. В этом случае для записи одной микрокоманды нужно меньше 36 разрядов. Возможность размещения в объеме 36 разрядов микрокоманд нескольких блоков и стремление совместить по времени работу как можно большего количества блоков привело к тому, что в формате микрокоманды процессора поля КСБ микрокоманд блоков объединяются в поле кода структуры процессора (КСП).

На рисунке 5.15 приведен формат микрокоманды процессора, где показано размещение поля КСП и поля управления.

КСП		У		КСП	
1	7	8	31	32	34

Рисунок 5.15 - Формат микрокоманды процессора

Распределение разрядов КСП за блоками процессора показано на рисунке 5.16.

0 р	1 р	2 р	3 р	4 р	5 р	6 р	7 р	8…31	32р	33р	34р
БУМ	БАЛ	БУД	БУП	БЗО-1	БАУД	БКД	БУВВ	У	БО-5	ПУ	БКП

Рисунок 5.16 Распределение разрядов поля КСП

Кроме блоков, назначение которых уже было оговорено, на рисунке обозначены: БУВВ - блок управления вводом-выводом информации; БО-5 - блок обмена параллельными кодами; ПУ - пульт управления и контроля ЭВМ.

Дешифраторы микрокоманд всех блоков анализируют состояние разрядов поля КСП и, при наличии единицы в соответствующем разряде, воспринимают значение поля управления. В поле управления приводятся значения разрядов нескольких (до пяти) микрокоманд. Распределение разрядов в данном случае определяется кодом КСП. Дешифраторы микрокоманд, задействованных для выполнения данной микрокоманды, блоков по коду КСП выделяют соответствующие разряды кода операции и формируют необходимый набор управляющих сигналов. Все выполняемые микрооперации одной микрокоманды заканчиваются за один машинный такт. По окончании времени машинного такта блок управления микрокомандами увеличивает на единицу адрес микрокоманды. В соответствии с полученным новым адресом из ЗУ микрокоманд извлекается следующая микрокоманда и выдается на шину микрокоманд. Процесс выполнения микрокоманд i-ой команды продолжается до тех пор, пока не будет достигнуто окончание микропрограммы. Последняя микрокоманда обеспечивает увеличение адреса команды в блоке управления программами, и происходит извлечение новой команды на реализацию.

Системы микропрограммного управления можно классифицировать по ряду признаков.

По типу памяти микропрограмм. В зависимости от типа памяти различают статическое и динамическое микропрограммирование. В первом случае в качестве памяти микрокоманд (ПМК) применяется постоянная память, во втором - оперативная.

программирование позволяет оперативно изменять микропрограммы, изменяя тем самым функциональные свойства устройств ЭВМ.

По способу формирования управляющих функциональных сигналов - горизонтальное, вертикальное и смешанное микропрограммирование.

По способу кодирования микроопераций - прямое и косвенное кодирование.

По способу исполнения микрокоманд - одно- и многофазные микрокоманды.

5.2.6 Микропроцессоры и их программирование

Цифровая обработка сигналов (ЦОС) находит в настоящее время все более широкое применение. Одной из причин этому является разработка интегральных схем, обладающих свойствами целой вычислительной машины. К числу таких микросхем относятся микропроцессоры, микроконтроллеры и процессоры обработки сигналов. Ядром этих устройств является процессор и схемы информационного обмена с внешней средой. Отличие заключается в предназначении интегральных микросхем. Микропроцессоры используются для создания универсальных вычислительных машин, микроконтроллеры – для управляющих в реальном масштабе времени специализированных вычислителей, а процессоры обработки сигналов – для специализированных вычислительных машин, предназначенных для выполнения различных преобразований сигналов, например, таких как быстрое преобразование Фурье (БПФ).

Таким образом, **микропроцессор (МП)** — это программно управляемая сверхбольшая интегральная схема (СБИС), предназначенная для обработки цифровой информации. Обладая небольшими размерами, кристалл МП содержит несколько миллионов транзисторов и выполняет функции, свойственные важнейшему узлу ЭВМ — процессору. Наибольший эффект от внедрения МП достигается в устройствах и системах локальной автоматики, системах измерения, контроля и других областях, в которых применение средств цифровой обработки данных до появления МП было нерентабельным. Сравнительно низкая стоимость, малые габариты и потребляемая мощность, высокая надежность и исключительная гибкость, не свойственная другим способам обработки данных, обеспечивают возможность построения БВМ на МП.

За более чем тридцатилетнюю историю развития микропроцессоров произошло значительное повышение производительности МП, изменение архитектурных принципов построения и технологии производства. Но для того чтобы уяснить основные принципы функционирования МП прежде всего необходимо рассмотреть структурную схему одного из первых микропроцессоров отечественного производства серии К580. Этот по современным меркам скромный МП обрабатывает восьмиразрядные данные. Несмотря на то, что он обеспечивает адресацию только 64 Кбайт ячеек памяти, области его применения по-прежнему чрезвычайно широки: более 90% практических задач обработки информации в бортовом РЭО могут быть решены на его основе.

Однокристальный микропроцессор К580ВМ80А с программным управлением (УУ с жесткой логикой) и фиксированной системой команд предназначен для параллельной обработки 8-разрядной цифровой информации. По назначению относится к классу универсальных МП и применяется в различных областях техники — от одноплатных контроллеров технологических процессов до персональных ЭВМ малой производительности. Кристалл МП выполнен по n-МОП технологии и содержит 4800 транзисторов. Быстродействие МП достигает 500 тыс.оп/с простых операций типа «регистр — регистр» при длительности цикла 250 нс. Конструктивно МП выполнен в пластмассовом корпусе с 40 выводами. Для расширения функциональных возможностей разработано значительное количество микросхем поддержки, которые составляют микропроцессорный комплект (МПК) К580, состоящий из 18 БИС.

Структурная схема МП КР580ВМ80А (рисунок 5.17) состоит из двух частей: операционной (ОП) и управляющей (УП). Обе части расположены на одном кристалле. Управляющая часть содержит недоступную для пользователя управляющую память, в которую в процессе изготовления БИС записаны операции, определяющие состав команд МП.

Операционная часть МП построена на базе 8-разрядного АЛУ, на два входа которого подключены два 8-разрядных буферных регистра БР1 и БР2. Вход регистра БР1 соединен с внутренней магистралью МП, а вход регистра БР2 — с аккумулятором, выполняющим функции регистра накопителя.

Блок РОН содержит шесть 8-разрядных регистров, обозначаемых буквами B, C, D, E, H, L, которые могут использоваться как одиночные 8-разрядные регистры и как регистровые 16-разрядные пары BC, DE, HL. Объединение регистров в пары дает возможность хранить 16-разрядные двоичные числа. Все регистры имеют трехразрядные кодовые обозначения. Регистровая пара обозначается кодом старшего регистра в паре. Например, регистр D имеет кодовое обозначение 010. Буферные регистры БР1, БР2, а также 8-разрядные регистры W и Z программно недоступны и служат для хранения данных при выполнении некоторых операций.

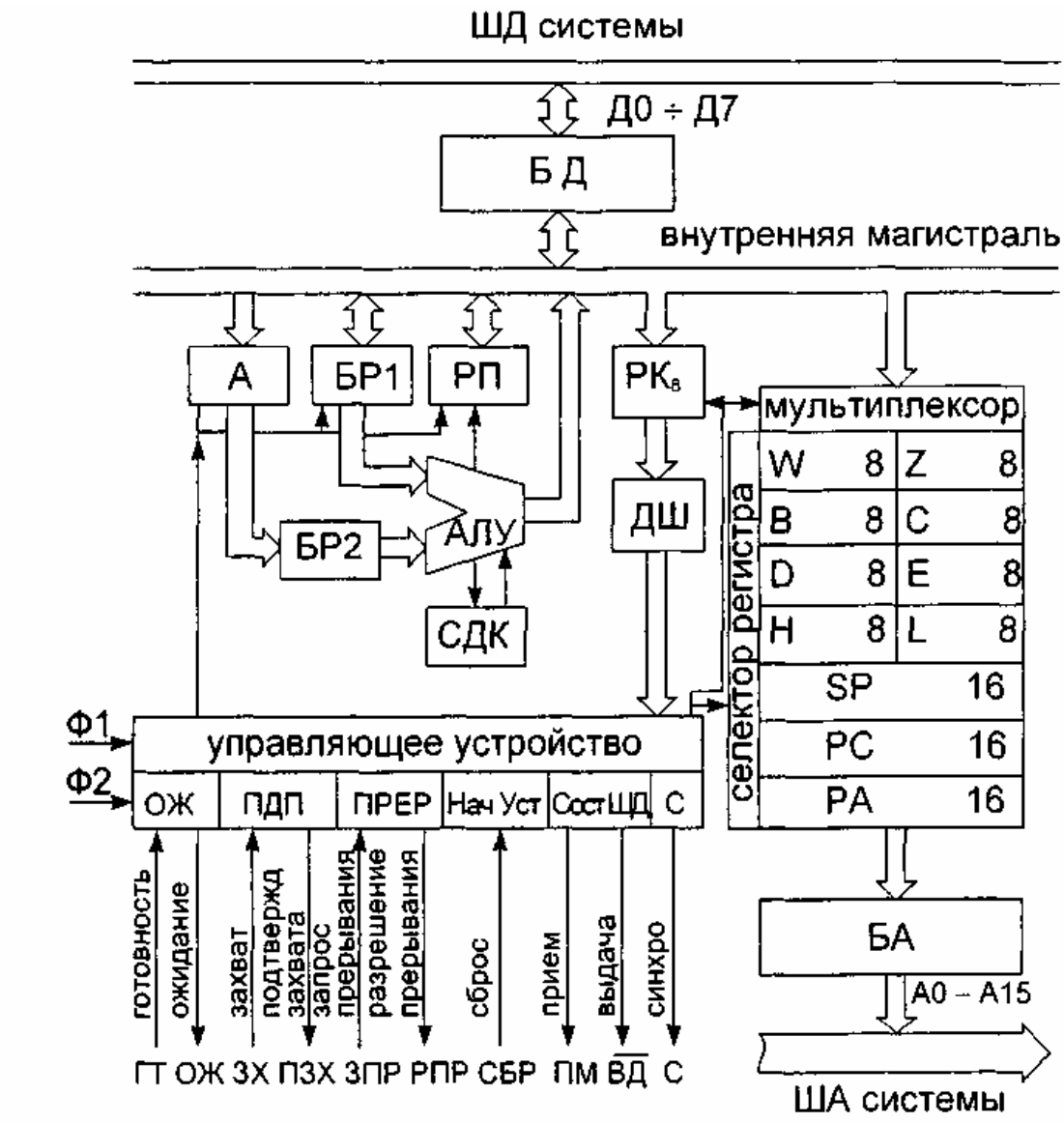


Рисунок 5.3 - Структурная схема МП КР580ВМ80А

Указатель стека SP и счетчик команд PC являются 16-разрядными регистрами и служат для хранения адресов ячеек памяти.

При обращении к памяти в качестве адреса может использоваться и содержимое любой регистровой пары блока РОН. Содержимое счетчика команд инкрементируется (увеличивается на единицу) после выборки содержимого любой ячейки памяти. Сформированный в РС адрес очередной ячейки памяти записывается в 16-разрядный регистр адреса РА и выдается на шину адреса системы через буфер адреса БА. МП имеет 16-разрядную ША и обеспечивает адресацию к памяти, содержащей 2^{16} = 64 Кбайт памяти.

АЛУ МП выполняет арифметические, логические и сдвиговые операции над 8-разрядными двоичными числами. Базовой операцией АЛУ является операция сложения двоичных чисел. Все арифметические, логические и сдвиговые операции выполняются при участии аккумулятора. Результат операции размещается в аккумуляторе. Обмен информацией МП с внешними устройствами (ВУ) возможен только через аккумулятор. МП имеет возможность выполнения операции с двухразрядными десятичными числами. С этой целью каждая десятичная цифра размещается в тетраде любого регистра МП. Коррекция двоичных чисел, необходимость которой возникает при выполнении арифметических операций над десятичными числами, выполняется специальной схемой десятичной коррекции СДК.

Восьмиразрядный регистр признаков РП используется для хранения признаков результата операции, выполняемой в АЛУ. Содержимое пяти задействованных в настоящее время разрядов РП, называемых флагами, имеет следующее назначение:

флаг переноса С устанавливается в состояние, соответствующее переносу из старшего разряда при выполнении арифметических операций или операций сдвига;

флаг нулевого результата Z устанавливается в состояние лог. «1», если результат операции в АЛУ или в любом РОН равен нулю;

флаг знака S устанавливается в состояние лог. «1», если результат операции в АЛУ или любом РОН отрицательный;

флаг четности Р устанавливается в состояние лог. «1», если количество единиц в разрядах четно;

флаг дополнительного переноса АС устанавливается в состояние лог. «1», если при выполнении операций АЛУ возникает перенос из четвертого разряда.

РП подключен к внутренней магистрали МП, что позволяет с помощью специальных команд переслать его содержимое в память

системы. Это необходимо, например, при вызове подпрограммы, когда состояние МП необходимо запомнить в стеке.

Управляющая часть МП содержит восьмиразрядный регистр команд РК, дешифратор команд и управляющее устройство. При выполнении команды программы, содержащейся в памяти системы, код команды через буфер данных БД и внутреннюю магистраль МП записывается в РК для дальнейшей обработки в дешифраторе команд и управляющем устройстве. Управляющее устройство, работа которого тактируется двумя импульсными последовательностями Ф1 и Ф2, вырабатывает три группы управляющих сигналов, необходимых для реализации четырех режимов работы МП: режима начальной установки, ожидания, прямого доступа к памяти и прерывания.

Сигналы, определяющие состояние шины данных:

С — выходной сигнал высокого уровня, стробирует появление на ШД «слова состояния»;

ПМ — выходной сигнал высокого уровня, подтверждает готовность МП к приему данных;

ВД — выходной сигнал низкого уровня, сообщает, что МП выдал информацию на ШД.

Сигналы, определяющие рабочий цикл микропроцессора:

ГТ — входной сигнал высокого уровня, сообщающий МП о готовности ВУ к обмену данными. При подаче на вход ГТ сигнала низкого уровня МП переходит в состояние ожидания, которое продолжается до момента появления на входе ГТ сигнала высокого уровня:

ОЖ — выходной сигнал высокого уровня, подтверждающий переход МП в состояние ожидания;

ЗХ — входной сигнал высокого уровня, обеспечивающий отключение МП от ШД системы за счет перевода буферов адреса БА и данных БД в высокоимпедансное состояние, характеризующееся практически бесконечным выходным сопротивлением;

ПЗХ — выходной сигнал высокого уровня, подтверждающий отключение МП от ШД;

СБР — входной сигнал низкого уровня, обеспечивающий начальную установку МП, при которой счетчик команд РС обнуляется, а сигналы ПЗХ и РПР сбрасываются. Содержимое РОН при этом изменяется.

Сигналы управления прерываниями:

ЗПР — входной сигнал высокого уровня, запрещающий чтение

кода операции очередной команды выполняемой программы и подготавливающий МП к выполнению команд программы обработки прерывания;

РПР — выходной сигнал высокого уровня, подтверждающий перевод МП в режим прерывания.

Цифровые процессоры сигналов. Цифровые программируемые процессоры сигналов (ППС) или по-другому цифровые сигнальные процессоры (Digital Signal Processors) разрабатывались специально для обработки сложных аналоговых сигналов большой размерности цифровыми способами, поэтому они содержат специальные аппаратные средства и команды. Спектр применения ППС очень широк и охватывает диапазон от простых традиционных модемов до радио- и гидролокационных систем, работающих в реальном масштабе времени. Сегодня ППС уже применяются для построения специализированных БВМ типа Багет 55. Более подробно принципы построения и использования сигнальных процессоров изложены в восьмой главе.

5.3 Интерфейсные устройства БВМ

5.3.1 Назначение, классификация и основные характеристики интерфейсов

Существенной особенностью работы БВМ в составе авиационного радиоэлектронного комплекса является интенсивный информационный обмен между БВМ и системами комплекса. Время обновления информации для большинства этих систем значительно больше, чем время выполнения команд или даже небольших программ в БВМ. Поэтому управление процедурами обмена информацией со стороны процессора БВМ не целесообразно, поскольку приводит к фактическому простою процессора и задержке в выработке управляющих сигналов для других систем.

Для организации обмена сначала в составе БВМ, а затем в виде отдельной конструкции были разработаны специальные устройства ввода-вывода (УВВ). В состав УВВ первых выпусков входило большое количество аналого-цифровых и цифроаналоговых преобразователей, поскольку элементы комплекса в большей степени работали с аналоговыми сигналами. При этом каждая новая совокупность систем (абонентов) требовала разработки своего УВВ, так как стандартиза-

ция и унификация средств обмена аналоговыми сигналами затруднена.

С разработкой систем, использующих методы цифровой обработки данных, развитием микропроцессоров и внедрением их в бортовые системы появились возможности для стандартизации бортовых средств обмена. Особенно этому способствовали развитие шинной организации связей между элементами микропроцессорных систем, а также появление понятия «интерфейс».

Термин «интерфейс» обычно трактуется как синоним слова «сопряжение». Под стандартным интерфейсом в соответствии с ГОСТ 15971 понимается совокупность унифицированных аппаратурных, программных и конструктивных средств, необходимых для реализации взаимодействия различных функциональных элементов в автоматических системах сбора и обработки информации при условиях, предписанных стандартом и направленных на обеспечение информационной, электрической и конструктивной совместимости указанных элементов.

Информационная совместимость – это согласованность взаимодействий функциональных элементов системы в соответствии с совокупностью логических условий, определяющих:

структуру и состав унифицированного набора шин;

набор процедур по реализации взаимодействия и последовательность их выполнения для различных режимов их функционирования;

способ кодирования и форматы данных, команд, адресной информации и информации состояния;

временные соотношения между управляющими сигналами, ограничение на их форму и взаимодействие.

Условия информационной совместимости определяют объем и сложность схемотехнического оборудования и программного обеспечения, их основные технико-экономические показатели – пропускную способность и надежность интерфейса, а также объем аппаратурных затрат на устройства сопряжения.

Электрическая совместимость – это согласованность статических и динамических параметров электрических сигналов в системе шин с учетом ограничений на пространственное размещение устройств интерфейса и техническую реализацию приемо-передающих элементов.

Условия электрической совместимости определяют:

тип приемо-передающих элементов;

соотношение между логическим и электрическим состояниями сигналов и пределы их изменения;

коэффициенты нагрузочной способности приемо-передающих элементов и значения допустимой емкости и резистивной нагрузки линии в устройстве и т.д.

Условия электрической совместимости влияют на скорость обмена данными, конфигурацию размещения устройств и расстояние между ними, предельно допустимое число подключаемых устройств и помехозащищенность. Обычно тип приемо-передающих элементов и большинство условий электрической совместимости регламентируется стандартом.

Конструктивная совместимость – это согласованность конструктивных элементов интерфейса для обеспечения механических и электрических соединений и замены схемных элементов, блоков и устройств.

Условия конструктивной совместимости определяют типы внешних и внутренних соединительных элементов (разъемов), конструкцию печатных плат, модулей (типовых элементов замены), блоков, стоек и т.п.

Составными физическими элементами связей интерфейса являются электрические цепи, называемые линиями интерфейса. Часть линий, сгруппированных по функциональному назначению, называется шиной, а вся совокупность линий – магистралью. В системе шин интерфейсов условно можно выделить две магистрали: информационную и магистраль управления.

По информационной магистрали передаются коды данных, адресов, команд и состояний устройств. Аналогичные названия присваиваются соответствующим шинам интерфейса.

Магистраль управления информационным каналом служит для организации обмена, формирования необходимого набора управляющих сигналов в соответствии с протоколом работы интерфейса.

Классификация интерфейсов в соответствии с ГОСТ 26.016 представлена в таблице 5.5.

Таблица 5.5

Классификационный признак	Типы интерфейсов
По способу соединения	Магистральный; радиальный; цепочечный; смешанный
По способу передачи	Параллельный; последовательный; последовательно-параллельный
По принципу обмена	Асинхронный; синхронный
По режиму передачи	Двусторонняя одновременная; двусторонняя поочередная; односторонняя

При реализации магистрального интерфейса обмен информацией между устройствами осуществляется либо отдельным контроллером интерфейса, либо одним из элементов системы, называемым центральным.

В радиальном интерфейсе обязательно один из элементов находится в центре и обеспечивает распределение информации между другими элементами по индивидуальным для каждого элемента линиям связи.

Цепочечный интерфейс обычно используется там, где формируемая устройствами информация должна переходить последовательно от устройства к устройству.

Смешанный интерфейс реализует два или более варианта соединения элементов системы.

Способ передачи информации определяет количество одновременно передаваемых разрядов двоичного кода. При последовательной передаче информационная шина содержит одну линию, по которой разряд за разрядом передается информация от источника сигнала к приемнику. Интерфейс с возможностью передачи одновременно всех разрядов машинного слова называется параллельным, а если частей слова, то последовательно-параллельным.

Асинхронный интерфейс содержит устройства, обеспечивающие возможность ведения обмена информацией в соответствии с реальным быстродействием устройств передачи и приема информации. В этом случае источник сигнала формирует сигнал запроса разрешения на передачу информации и адрес устройства, являющегося приемником информации. Устройство, идентифицировавшее адрес, определяет возможность проведения обмена и формирует соответствующий сигнал разрешения на передачу информации. По получению этого

сигнала источник информации выдает код передаваемого сообщения на информационную шину данных и до окончания передачи удерживает её. После завершения передачи сообщения источник информации формирует соответствующий сигнал «Передача завершена» и ожидает сигнала «Прием окончен» от приемника информации. По приходу этого сигнала источник информации освобождает шину данных. Время выполнения синхронного обмена соответствует быстродействию самого медленно работающего элемента системы. При организации обмена информацией источник информации выдает адрес и одновременно (или с фиксированной задержкой) данные. Время выдачи адреса и данных также ограниченно и постоянно для всех устройств. Достоинство синхронного способа – простота реализации алгоритма обмена. Недостаток – низкое быстродействие при наличии элементов системы с существенно различным быстродействием.

Для организации обмена информацией в составе интерфейсной магистрали должны быть шины трех видов:

шина данных – для передачи информационного сообщения;

шина адреса – для передачи адреса приемника и источника информации;

шина управления – для передачи сигналов, обеспечивающих протокол обмена.

В системах, не ограниченных жесткими требованиями к быстродействию, шины адреса и данных могут совмещаться с целью уменьшения аппаратурных затрат. С этой же целью обмен информацией от одного устройства к другому, и наоборот, выполняется по одним и тем же линиям, которые называются в этом случае двунаправленными (двусторонними).

В БВМ типа А-15А обмен между блоками машины и УВВ осуществляется в основном параллельными кодами. Поэтому магистрали внутриблочные, межблочные выполнены в виде однонаправленных параллельных магистралей. В качестве конструктивных средств межблочных соединений применяется в основном витая пара проводов.

Для сопряжения БВМ и УВВ применяются две 16 разрядные магистрали: К1 (для передачи параллельного кода в УВВ) и магистраль К2 (для приема параллельного кода из УВВ). Указанные магистрали являются информационными совмещенными шинами адреса и данных. Кроме однонаправленных информационных магистралей

имеются две однонаправленные магистрали управления. БВМ А-15А сопрягается с магнитным регистратором по 16 разрядной информационной магистрали КЗ и управляющей магистрали.

Таким образом, в интерфейс БВМ А-15А применяются однонаправленные параллельные магистрали, выполненные в виде многожильных кабелей, состоящих из витых пар проводов. Такие решения обеспечивают высокую пропускную способность магистралей.

5.3.2 Режимы работы БВМ при взаимодействии с внешними устройствами

В зависимости от того, кто является инициатором информационного обмена и как осуществляется передача информации, различают три основных способа информационного обмена БЦВМ и внешних устройств:

программный обмен;

обмен по прерываниям;

обмен по каналу прямого доступа.

Инициатором программного обмена является процессор БЦВМ. Причиной обмена является специальная команда ввода или вывода данных, которую процессор прочитал из памяти. Как в любой другой команде в команде обмена указан адрес источника и приемника информации. Для уменьшения длины формата команды один из адресов может быть подразумеваемым.

Обмен по прерываниям осуществляется по инициативе внешнего устройства. При готовности к информационному обмену устройство формирует запрос на прерывание выполняемой программы. Процессор анализирует уровень приоритета запроса прерывания и, в случае более важного приоритета запроса, дает разрешение на прерывание. Текущая программа останавливается, все атрибуты её сохраняются в стеке. Загружается программа информационного обмена с устройством, формировавшим запрос. По завершении обмена восстанавливается прерванная программа. Обмен осуществляется обычно небольшими по размеру данными.

Обмен большими объемами данных осуществляется по каналу прямого доступа. Инициатором является внешнее устройство. Оно формирует запрос шины данных и, получив разрешение от процессора, напрямую записывает или считывает данные в оперативную память.

Система прерываний обеспечивает не только организацию обмена информацией БВМ и внешних устройств, когда длина передаваемых сообщений требует прерывания выполняемой программы, но и реализует своевременную реакцию процессора на возникновение различных ситуаций, как в процессоре, так и во внешней среде. Важность системы прерывания определяет необходимость рассмотрения принципов её построения и функционирования более детально.

5.3.3 Особенности организации информационного обмена по прерываниям

Во время выполнения бортовой ЦВМ текущей программы внутри машины и в связанной с ней внешней среде (например, в бортовом РЭО, управляемом вычислительной машиной) могут возникать события, требующие немедленных реакций на них со стороны машины. Эти реакции состоят в том, что при возникновении подобных событий машина должна прервать обработку текущей программы и перейти к выполнению некоторой другой программы, специально предназначенной для ситуации, связанной с появлением данного события. По завершению этой программы машина должна вернуться к выполнению прерванной программы.

Рассматриваемый процесс, называемый прерыванием программ, может быть проиллюстрирован рисунком 5.18.

Принципиально важным является то, что моменты возникновения событий, требующих прерывания программ, не известны и поэтому не могут быть заранее учтены при программировании.

Каждое событие, требующее прерывания, должно сопровождаться сигналом, оповещающим БЦВМ о его возникновении. Этот сигнал называется запросом прерывания. Программы, затребованные запросами прерывания, называются прерывающими программами в отличие от прерываемых программ, выполнявшихся машиной до появления запросов.

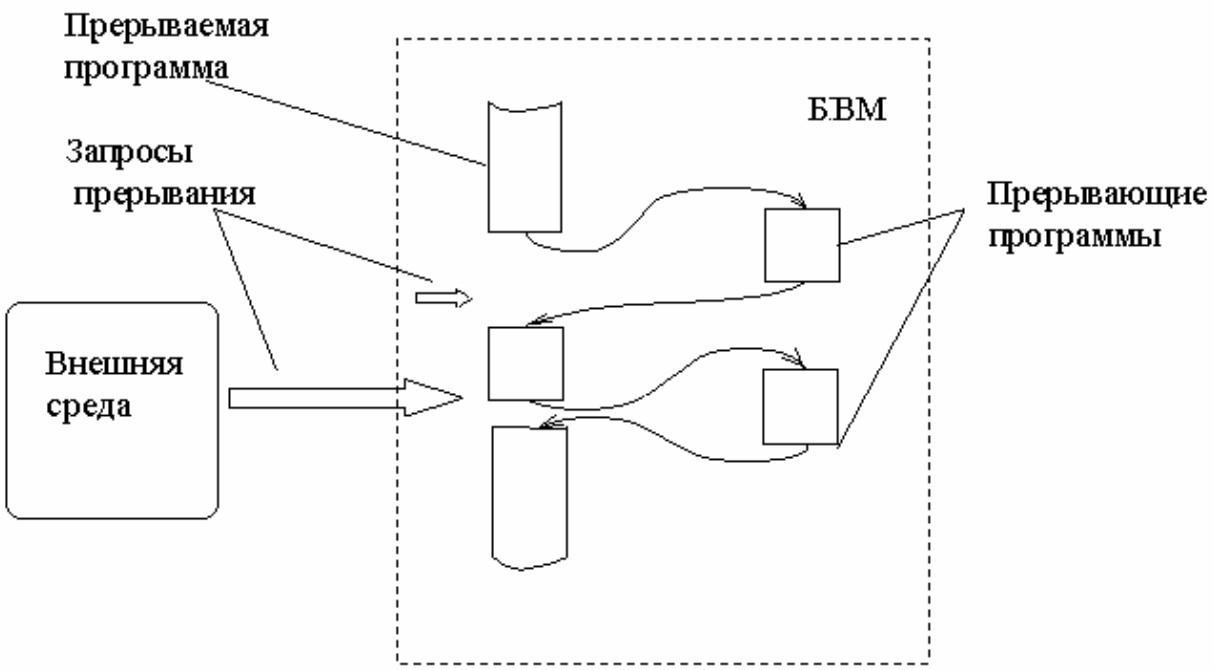


Рисунок 5.18 - Прерывание программ

Как показано на рисунке 5.18, запросы на прерывание могут возникнуть внутри самой БВМ и в ее внешней среде. К первым относятся, например, запросы при возникновении в БВМ таких событий, как появление сбоев в ее аппаратуре, переполнение разрядной сетки, попытка деления на ноль, нарушение правил адресации, завершение операции ввода-вывода и др. Хотя некоторые операции порождаются самой программой, моменты их появления, как правило, невозможно предусмотреть.

Запросы прерывания во внешней среде могут возникать от других ЦВМ, от наземных систем автоматизированного управления, от радиолокационных, оптических и других систем слежения за целью, а также от различных аварийных датчиков.

При прерывании программы БВМ должна временно прекращать выполнение текущей программы и переходить к обработке программы, соответствующей запросу прерывания, а по ее завершении возвращаться к выполнению прерванной программы. Для реализации перечисленных действий в составе устройства управления бортовой ЦВМ есть специальные аппаратные и программные средства, называемые системой прерывания. В случаях наличия нескольких источников прерывания, вырабатывающих свои запросы независимо, должен быть установлен определенный порядок обслуживания запросов.

Существуют системы прерывания с последовательным просмотром (сканированием) наличия запросов у источников прерывания и с обслуживанием запросов в порядке присвоенного им приоритета. Последние системы прерывания называются приоритетными.

Системы прерывания выполняют следующие функции:

запоминание состояния прерываемой программы;

приоритетный выбор запроса для исполнения из группы поступивших запросов прерывания и организацию перехода к прерывающей программе;

восстановление состояния прерванной программы и возврат к ней;

программное изменение приоритетов запросов.

Если управление запоминанием состояния и возвратом возложено на саму управляющую программу, то она будет состоять из трех частей: подготовительной и заключительной, обеспечивающих переключение программ, и собственно прерывающей программы, выполняющей затребованную запросом работу. На рисунке 5.19 приведена упрощенная временная диаграмма процесса прерывания.

Для оценки эффективности систем прерывания могут быть использованы следующие характеристики: время реакции, глубина прерывания (количество уровней прерывания), насыщение системы прерывания (наличие очереди), допустимые моменты прерывания программ.

Время реакции – это время между появлением запроса прерывания и началом выполнения прерывающей программы. На рисунке 5.19 это время обозначено t_P. Для одного и того же запроса задержки в исполнении прерывающей программы зависят от того, сколько программ со старшим приоритетом ждет обслуживания. Поэтому время реакции определяют для запроса с наивысшим приоритетом.

Сумма времени, затрачиваемого на запоминание состояния прерванной программы, и времени на возврат к ней называется временем обслуживания:

$$t_{об} = t_З + t_В .$$

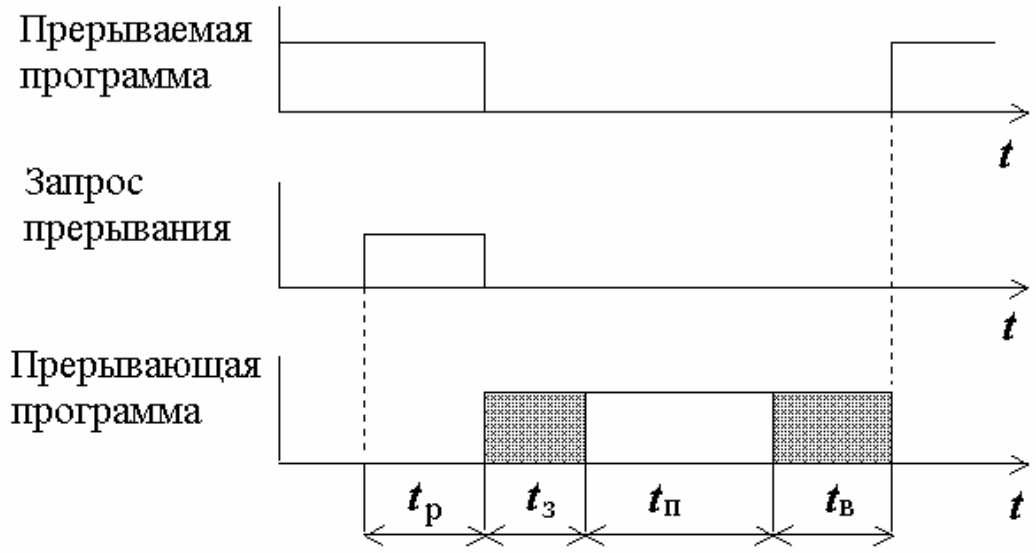


Рисунок 5.19 - Упрощенная временная диаграмма процесса прерывания:$t_р$ – время реакции; $t_з$– время запоминания состояния прерванной программы; $t_п$ – время выполнения прерывающей программы; $t_в$ – время восстановления состояния прерванной программы

Глубина (количество уровней) прерывания – максимальное число программ, которые могут прерывать друг друга. Если после перехода к прерывающей программе и вплоть до ее окончания прием других запросов запрещается, то говорят, что система имеет один уровень прерывания. Глубина равна *n*, если допускается последовательное прерывание до *n* программ. На рисунке 5.20 показано прерывание в многоуровневых системах прерывания (предполагается, что приоритет каждого следующего запроса выше предыдущего). Системы с большим значением глубины прерывания обеспечивают более быструю реакцию на срочные запросы.

К одному уровню прерывания обычно относят не один, а несколько запросов, близких по степени важности событий. Исполнение программ одного уровня (класса) прерываний осуществляется как в одноуровневой СП, т.е. по очереди.

Насыщение системы прерывания возникает, если время реакции настолько велико, что текущий запрос окажется необслуженным к моменту прихода нового запроса от этого же источника. В этом случае один запрос прерывания от данного источника будет системой утрачен, что является недопустимым. Быстродействие БВМ, логические возможности системы прерывания и количество источников прерывания должны быть согласованы таким образом, чтобы насыщение было невозможным.

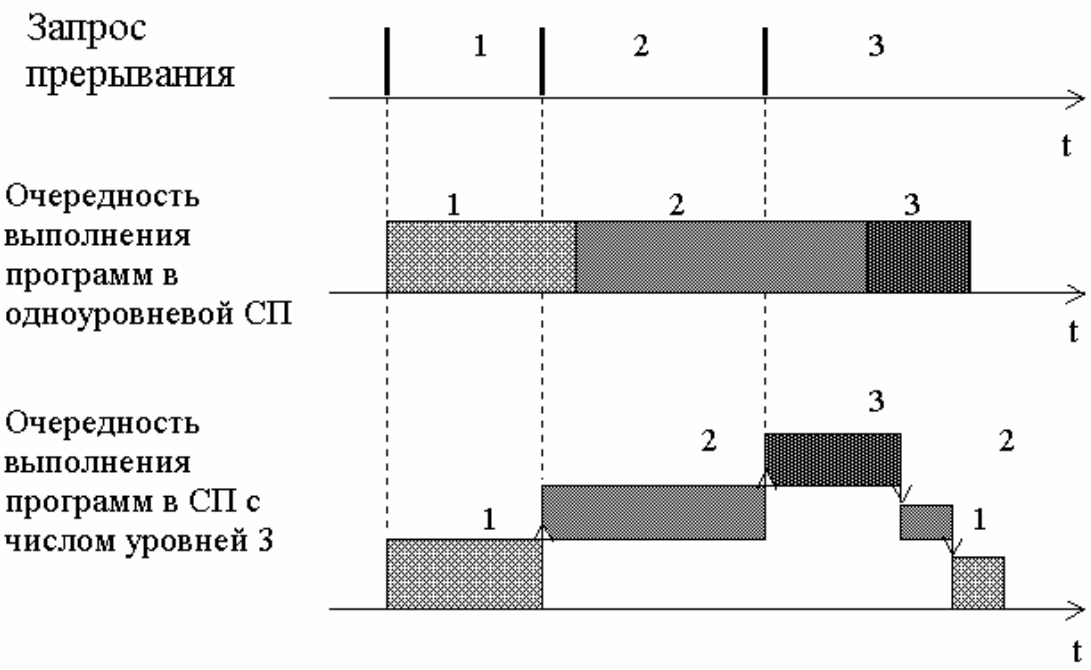


Рисунок 5.20 Прерывание в системах с различной глубиной прерывания

Допустимые моменты прерывания программ. Большей частью прерывание допускается после окончания любой текущей команды. В этом случае необходимо запоминание состояния программы, т.е. содержимого всех программно доступных регистров, так как они могут быть изменены прерывающей программой.

Время реакции определяется в основном длительностью выполнения одной команды. Это время реакции может оказаться недопустимо большим для БЦВМ т.к. они работают в реальном масштабе времени. Поэтому иногда допускается прерывание после любого такта выполнения команды. Однако в этом случае возрастает количество информации, подлежащей запоминанию и восстановлению при переключении программ. Помимо состояния прерванной программы, необходимо сохранить также и состояние процессора в момент прерывания (содержимое счетчика команд, счетчика тактов, регистра команд и т.д.). Поэтому такая организация прерывания возможна лишь в машинах со сверхоперативной памятью.

Приоритетное обслуживание программ. Понятие приоритета в прерывании программ имеет два значения: приоритет между запросами прерывания и приоритет между прерывающими программами.

Первый устанавливает лишь очередность восприятия запросов, поступивших одновременно от различных уровней, а второй, более важный – старшинство (степень срочности) в выполнении прерывающих программ разных уровней. Иначе говоря, во втором случае определяется, имеет ли право данная прерывающая программа прервать обрабатываемую в данный момент программу.

Приоритет между k запросами (уровнями) прерывания (больший приоритет имеют уровни с меньшими номерами) можно реализовать аппаратурно, например, посредством схемы последовательного поиска, или программно. Относительная степень важности программ, их длина и частота повторения в ходе вычислительного процесса могут меняться, требуя установления новых приоритетных отношений. Поэтому если приоритет между запросами может быть установлен жестко, то приоритет между прерывающими программами не может быть зафиксирован раз и навсегда. Необходимо иметь возможность изменять по мере надобности приоритетные соотношения программным путем, иначе говоря, приоритет между прерывающими программами должен быть программно-управляемым. В современных БЦВМ наибольшее распространение получило программное управление приоритетом на основе маски прерывания.

Маска прерывания представляет собой двоичный код, размеры которого поставлены в соответствие уровням или источникам (причинам) прерывания. Маска устанавливается специальной командой в регистре масок. Состояние 1 в данном разряде регистра масок разрешает, а состояние 0 запрещает («маскирует») прерывание текущей программы от запросов соответствующего уровня прерывания. Порядок расположения разрядов в регистре масок и нумерация уровней не имеет значения. Таким образом, изменяя программным способом содержимое регистра масок, можно устанавливать произвольные приоритетные соотношения между уровнями с любыми номерами без перекоммутации проводов запросов прерывания.

С замаскированным запросом поступают в зависимости от причины прерывания, либо игнорируя его, либо запоминая. Например, если прерывание вызвано окончанием операции в периферийном устройстве, то его следует запомнить, чтобы сообщить БВМ об освобождении периферийного устройства. Если замаскированное прерывание вызвано переполнением разрядной сетки при арифметической операции, например, в случае получения контрольной суммы всех ячеек

постоянной памяти, его следует игнорировать.

Все изучаемые БЦВМ имеют многоуровневые системы прерывания (А-15А – 4 уровня, Ц-100 – 7 уровней, Орбита 10 – 2 уровня) и имеют возможность программного изменения приоритета выполняемых программ посредством маскирования.

Для выполнения прерывания СП должна обеспечить запоминание прерываемой программы, организовать переход к прерывающей программе и возврат к прерванной программе.

Простейший способ указания начальных адресов прерывающих программ состоит в том, что каждому уровню отводится в памяти фиксированная ячейка, в которой хранится информация о начальном адресе прерывающей программы.

Информацию, подлежащую запоминанию при прерывании программы, можно разделить на две части: первая – основная информация, которая запоминается всегда (адрес следующей невыполненной команды прерванной программы, ее маска прерывания и др.); вторая – дополнительная информация, соответствующая содержимому тех регистров, которые данная прерывающая программа использует.

Запоминание и восстановление основной информации производится обычно аппаратными средствами, а дополнительной, состав которой зависит от самой прерывающей программы, осуществляется начальными командами последней.

Основная информация о состоянии программы (адрес очередной команды, состояния управляющих триггеров, маска прерывания и др.) формируются в так называемое слово состояния программы (ССП), занимающее одно или несколько машинных слов. Процедура перехода к прерывающей программе и возврата из нее в системе с использованием ССП поясняется рисунком 5.21. Каждому уровню прерывания отводится две фиксированные ячейки ОЗУ для хранения «старого ССП» и «нового ССП». Само прерывание программы состоит в замене старого ССП, принадлежащего прерываемой программе, новым ССП, соответствующим прерывающей программе. При выполнении текущей программы ее ССП находится в регистре текущего ССП (РгССП) и участвует в управлении вычислительным процессом. При этом отдельные поля ССП (например, счетчик команд) нужным образом изменяются. В ячейках новых ССП для всех уровней прерывания хранятся ССП, содержащие информацию, доступную для начала функционирования соответствующих прерывающих программ.

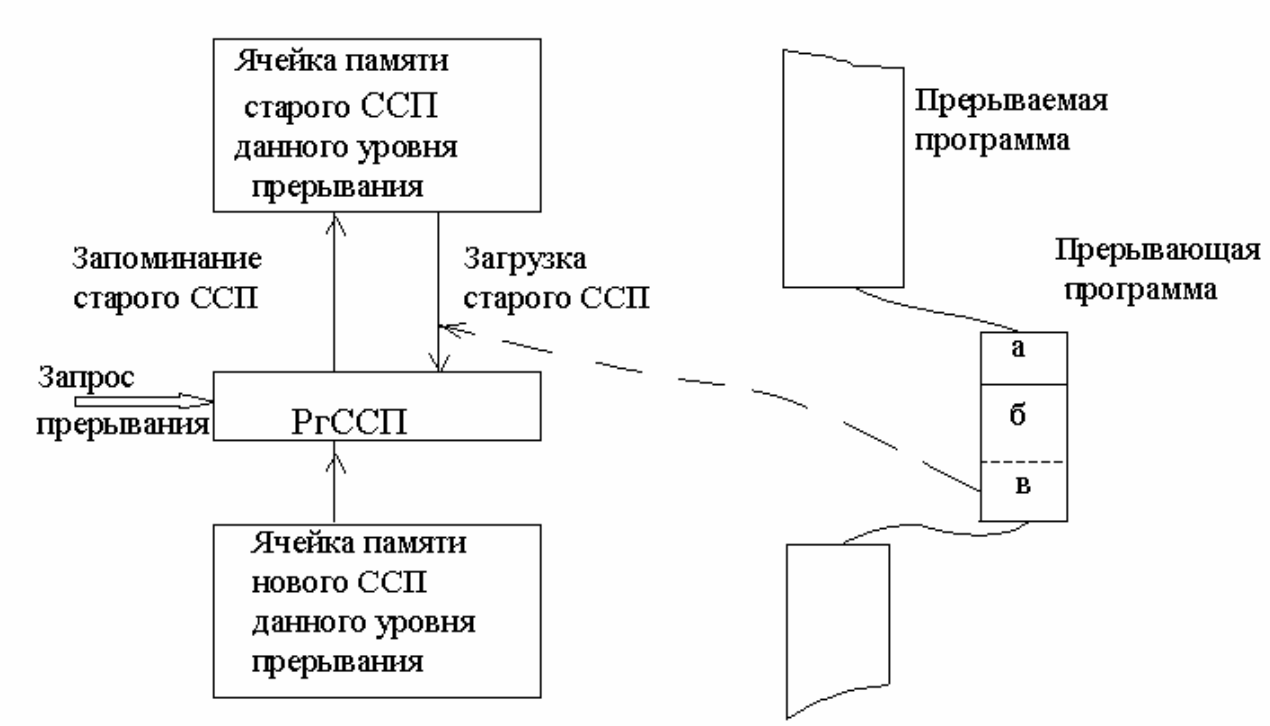


Рисунок 5.21 - Процедура перехода к прерывающей программе и возврата из нее: а – запоминание содержимого регистров; б – собственно прерывающая программа; в- восстановление содержимого регистров

При поступлении запроса от данного уровня прерывания, если этот запрос не маскирован в маске прерывания, аппаратные средства выполняют следующую процедуру. В соответствующую группу разрядов РгССП записывается код прерывания, содержащий информацию о конкретной причине прерывания. Затем ССП из РгССП передается в предусмотренную для данного уровня прерывания ячейку ОЗУ для старого ССП, а из ячейки нового ССП для этого уровня новое ССП (ССП прерывающей программы, содержащее в том числе ее начальный адрес) загружается в РгССП. С этого момента управление переходит к прерывающей программе. На время замены ССП прерывания от любых уровней запрещены.

Выполнение прерывающей программы начинается с команд запоминания в ячейках памяти дополнительной информации о состоянии программы (содержимого используемых ею регистров). Далее выполняется собственно прерывающая программа. Эта часть начинается с анализа кода прерывания, определения конкретной причины прерывания и инициирования подпрограммы, соответствующей этой

причине. Заключительная часть прерывающей программы восстанавливает сохраненное в ОЗУ содержимое использовавшихся прерывающей программой регистров и завершает свою работу командой загрузки слова состояния прерванной программы из ячейки старого ССП в РгССП. Управление переходит к прерванной программе.

Бортовая цифровая вычислительная система Б5 самолета МиГ-31 состоит из БЦВМ «Аргон» А-15А и устройства ввода – вывода Б5.01. Основная задача УВВ – обеспечить преобразование из аналоговой формы представления сигналов в цифровую и наоборот. Такие преобразования осуществляют аналого-цифровые и цифроаналоговые преобразователи (АЦП и ЦАП).

5.3.4 Аналого-цифровые и цифроаналоговые преобразователи сигналов

Процесс преобразования аналогового сигнала в цифровой можно разделить на три этапа:

дискретизацию во времени;

квантование по уровню;

кодирование.

Этап дискретизации во времени заключается в том, что из непрерывного по времени сигнала $f(t)$ выбираются отдельные его значения, соответствующие тактовыми моментами времени $t_1, t_2, ..., t_n$, следующих через определенный временной интервал – период дискретизации $T_д$. Выбор $T_д$ для представления непрерывной функции набором периодически следующих дискретных отсчетов производится на основе теоремы Котельникова В.А., которая гласит, что для представления функции $f(t)$, имеющей ограниченный спектр, набором дискретных отсчетов, необходимо чтобы частота повторения импульсов дискретизации

$$F_n = \frac{1}{T_д} \qquad (5.1)$$

была больше удвоенной максимальной частоты спектра функции $f(t)$, т.е.

$$F_n \geq 2F_{\max}, \qquad (5.2)$$

где $F_{\max}$ - максимальная частота спектра.

Операция квантовая заключается в том, что создается сетка уровней квантовая, сдвинутых относительно друг друга на шаг квантовая h (рисунок 5.22,б). Каждому уровню квантования приписывается порядковый номер (0, 1, 2, …, n-1). Далее полученные в результате дискретизации значения исходного аналогового сигнала заменяется ближайшим к ним уровнями квантования (см. рисунок 5.22,в). Таким образом, при квантовании исходная непрерывная функция заменяется ступенчатой. При этом возникают погрешности квантования $\delta_к$, так как квантование значения воспроизводят не точные значения аналоговой величины, а округленные до ближайшего уровня. Уменьшение ошибки квантования достигается уменьшением шага квантования h.

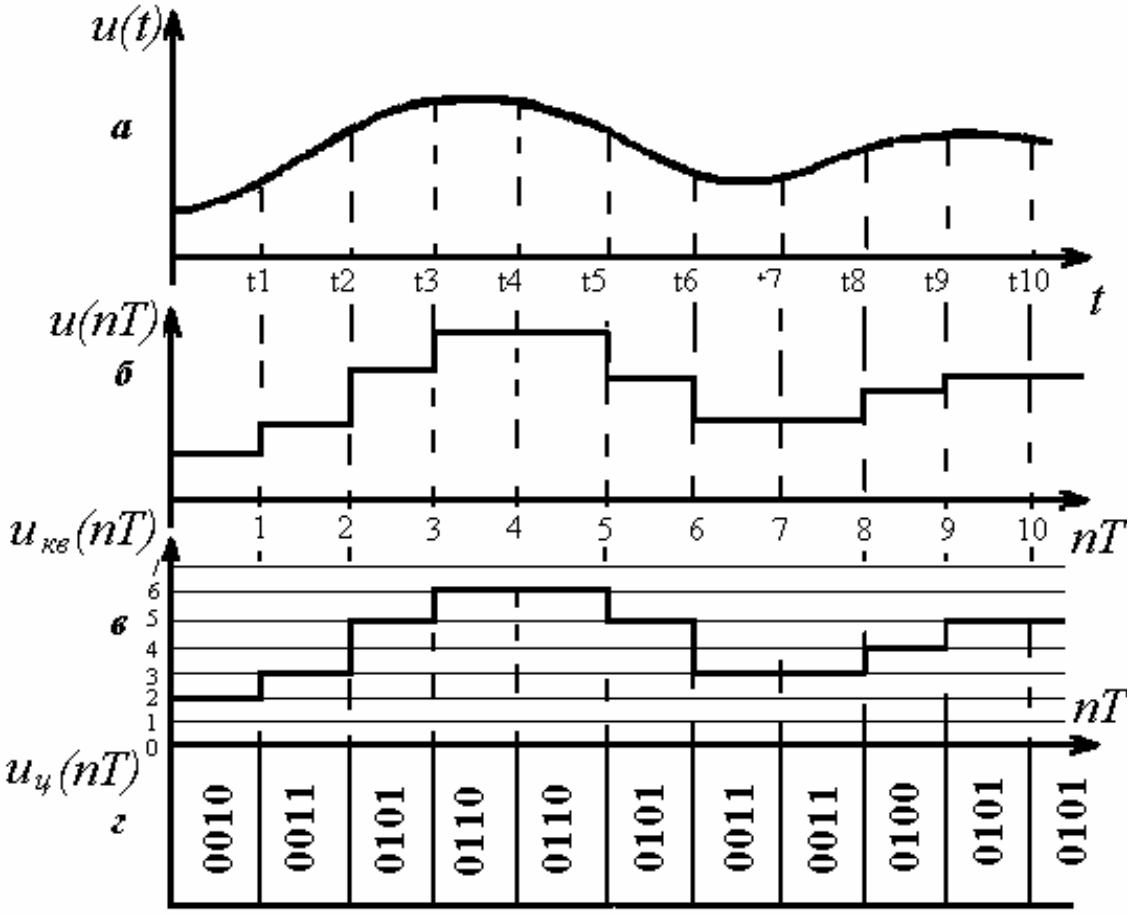


Рисунок 5.22

Ошибка квантования и дискретизации являются величинами случайными, поэтому их называют шумами квантования и дискретизации.

На этапе кодирования квантованные уровни представляются числами - номерами соответствующих уровней квантования (см. рисунок 5.22,в). Далее (см. рисунок 5.22,г), полученная таким образом последовательность чисел, представляется двоичными кодами.

Аналого-цифровые преобразователи в зависимости от области применения можно разбить на четыре основные группы:

преобразователи перемещений (линейных или угловых);

преобразователи электрических величин (напряжений, токов или сопротивлений);

преобразователи интервалов времени;

преобразователи следящего типа.

Основными характеристиками АЦП являются разрядность, диапазон изменения входного напряжения, разрешающая способность, погрешность преобразования, время преобразования.

Аналого-цифровые преобразователи характеризуются значительным разнообразием принципов и схематических вариантов реализации.

Наиболее широко получили распространение следующие из них: АЦП на основе методов параллельного преобразования (рисунок 5.23); АЦП на основе методов последовательного приближения (рисунок 5.24); АЦП на основе двойного интегрирования (рисунок 5.25).

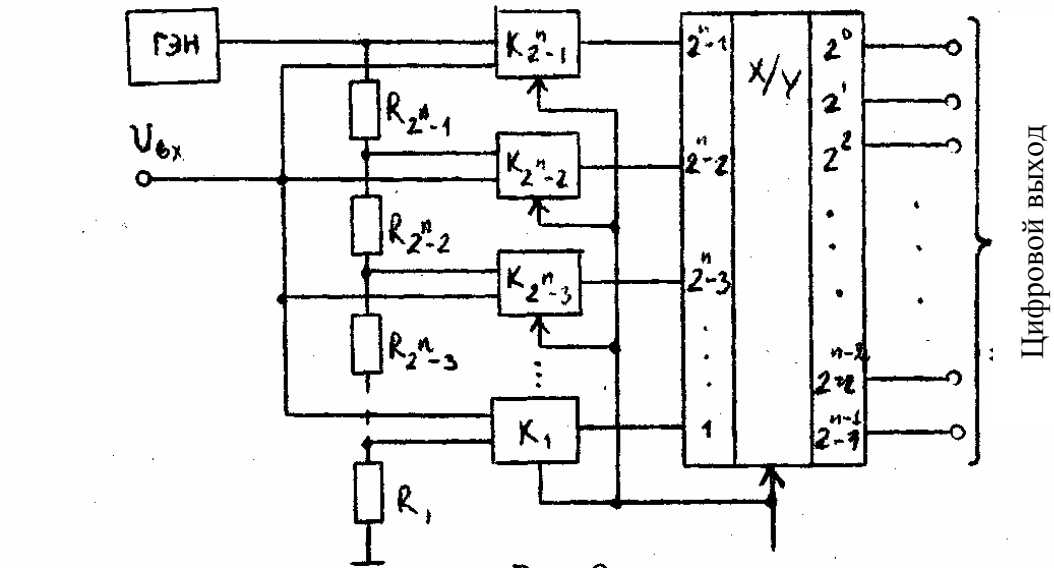


Рисунок 5.23

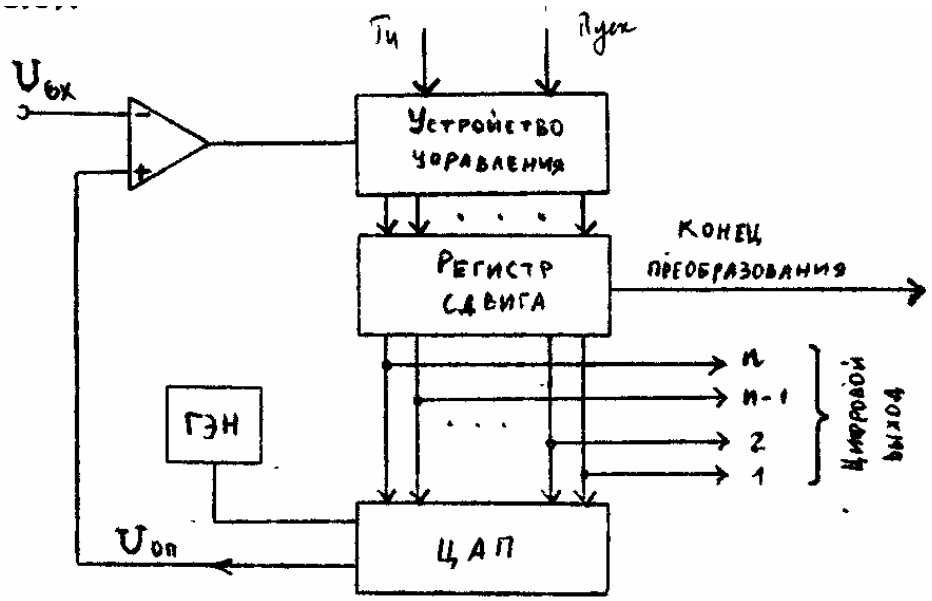


Рисунок 5.24

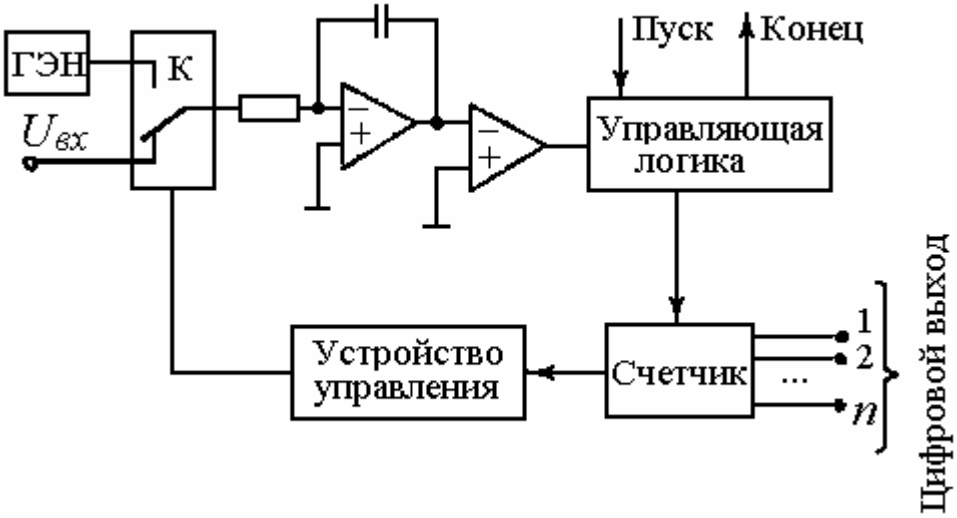


Рисунок 5.25

Самыми быстродействующими являются АЦП параллельного преобразования, самыми точными – АЦП двойного интегрирования.

Структурная схема АЦП, использующего метода параллельного преобразования (см. рисунок 5.23), включает в себя источник (генератор) эталонного напряжения (ГЭН) с делителем на прецизионных резисторах, набор компараторов (K_i) и кодопреобразователь (x/y). Аналоговый сигнал сравнивается с уровнями квантования. Уровней квантования всего

$$K_{ур} = 2^n - 1, \qquad (5.3)$$

где n – число разрядов цифрового выхода АЦП.

На выходе компараторов формируется сигнал высокого уровня (логическая единица) если аналоговый сигнал превысил значения закрепленных за ними уровней. На остальных компараторах выходной сигнал соответствует логическому нулю. Полученная комбинация из нулей и единиц преобразуется кодопреобразователем в двоичный код. Быстродействие АЦП определяется временем восстановления компаратора и задержкой кодопреобразователя. Возможности параллельных АЦП по скорости преобразования аналоговых сигналов в десятиразрядный двоичный код позволяют осуществлять это преобразование с частотой до 100 МГц.

Недостатком параллельного АЦП является резкое увеличение числа компараторов при незначительном увеличении разрядности кода. Действительно, для реализации двенадцатиразрядного быстродействующего АЦП требуется 4095 компараторов. Обычная разрядность для таких АЦП по этой причине не превышает 10. Очевидно, что при такой разрядности получить высокую разрешающую способность и точность невозможно.

Чтобы решить задачу обеспечения высокой разрешающей способности и точности при ограниченном количестве компараторов применяют комбинированный параллельно-последовательный способ преобразования.

Структурная схема АЦП, построенного на принципе последовательных приближений (см. рисунок 5.24) включает в себя компаратор, реализованный на операционном усилителе, цифро-аналоговый преобразователь (ЦАП), регистр сдвига, устройство управления и источник эталонного напряжения (ГЭН). Принцип действия такого АЦП заключается в последовательном приращении опорного для компаратора напряжения U_{on}. Это напряжение формирует ЦАП, получая от регистра сдвига с каждым тактом новый цифровой код.

Поведение регистра сдвига определяется устройством управления, которое по сигналу с выхода компаратора задает режим регистра на следующий такт (увеличение или уменьшение числа, записанного в регистр на единицу младшего разряда).

Для примера проиллюстрируем принцип работы шестиразрядного АЦП, временная диаграмма функционирования которого представлена на рисунке 5.26. Исходное состояние выходного регистра нулевое.

Начинается цикл преобразования с того, что в старший разряд регистра (СЗР) заносится единица и ЦАП вырабатывает соответствующее напряжение 5 В. Поскольку уровень входного сигнала не достигнут, единица заносится в следующий разряд. На выходе ЦАП во втором такте появляется приращение опорного напряжения, равное 2,5 В (т.е. половине напряжения СЗР). Но напряжение на выходе ЦАП по-прежнему меньше входного, поэтому в третий разряд регистра занесем единицу. Напряжение на выходе ЦАП повысилось на 1,25 В и превысило уровень входного. Компаратор переключился и по этому сигналу устройство управления обнулило третий разряд.

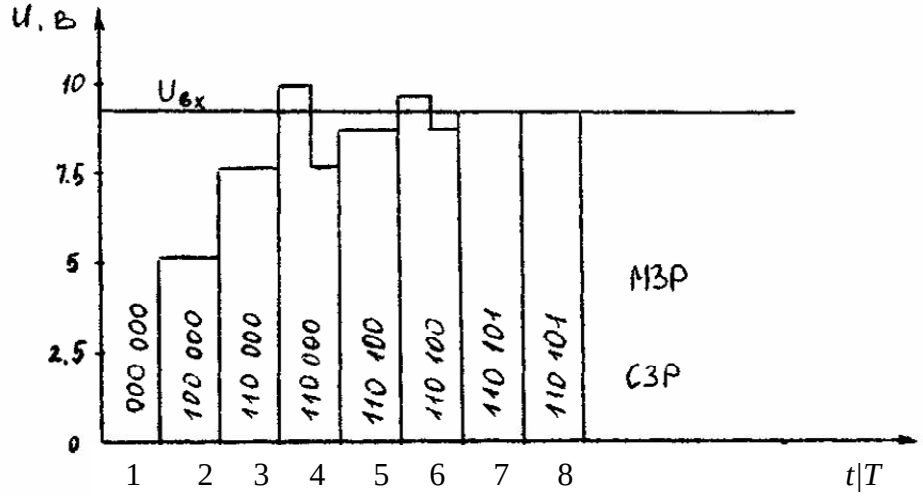


Рисунок 5.26

В четвертом такте единица заносится в четвертый разряд и т.д. Спустя шесть тактов на выходе ЦАП установится цифровой код, соответствующий с учетом разрешающей способности АЦП, входному напряжению. Быстродействие такого АЦП определяется временем, затраченным на n проб. Длительность одной пробы, в основном, определяется временем установления ЦАП $t_{уст}$, т.е.

$$T_{пр} = n t_{уст}. \qquad (5.4)$$

Следовательно, десятиразрядный АЦП, построенный с использованием ЦАП, имеющего $t_{уст}$ = 3…5 мкс, будет работать с частотой преобразования не выше 20…30 кГц. Повышение быстродействия АЦП с последовательным приближением возможно только путем повышения быстродействия ЦАП.

Построенный по принципу двойного интегрирования АЦП (см. рисунок 5.25) состоит из интегратора, компаратора, счетчика, устройства управления, управляющей логики и источника эталонного напряжения (ГЭН). Принцип действия состоит в следующем. Вначале в течение времени Т1 (рисунок 5.27) осуществляется интегрирование входного сигнала до тех пор, пока счетчик не пошлет сигнал переполнения в устройство управления, которое переключит интегратор на источник эталонного напряжения.

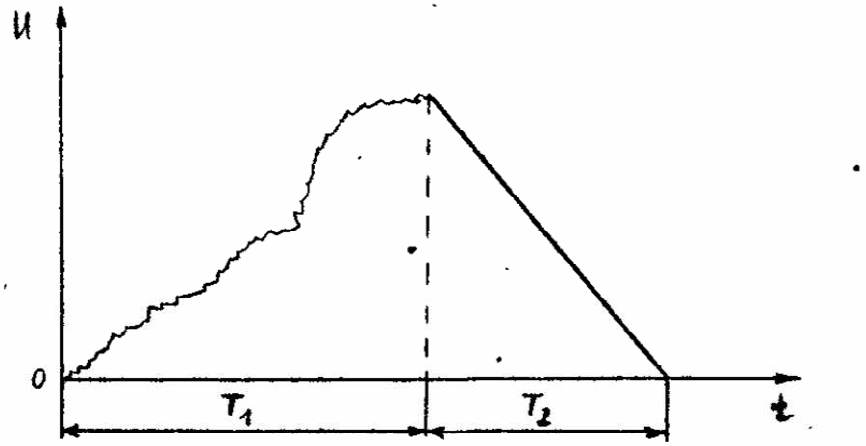


Рисунок 5.27

Конденсатор С начнет разряжаться. Счетчик, в котором после переполнения установилось нулевое состояние, с началом процесса перезаряда конденсатора продолжает работать точно так же, как и в течение времени Т1, с той лишь разницей, что теперь он не управляет, а сопровождает процесс разряда конденсатора С. По окончании разряда конденсатора сработает компаратор и через логику управления остановит счетчик. Код на выходах счетчика будет пропорционален среднему значению аналогового сигнала в интервале Т1. Такие АЦП имеют низкое быстродействие. Основная область их применения – прецизионные устройства с очень низкой частотой обработки сигналов. Таким образом, основными элементами АЦП, построенного по любой схеме, являются: ключевые схемы, компараторы, генераторы (источники) эталонных напряжений, накопительное устройство, преобразователь кода и управляющая логика, которая определяет порядок функционирования аналого-цифровых преобразователей информации.

Цифро-аналоговым преобразователем (ЦАП) называется электронное устройство, предназначенное для преобразования цифровой информации в аналоговую. Они используются для формирования сигнала в виде напряжения или тока, функционально связанного с

управляющим кодом. В большинстве случаев эта функциональная зависимость является линейной. Наиболее часто ЦАП используются для сопряжения устройств цифровой обработки сигналов с системами, работающими с аналоговыми сигналами. Кроме этого, ЦАП используются в качестве узлов обратной связи в аналого-цифровых преобразователях и в устройствах сравнения цифровых величин с аналоговыми.

Схемы ЦАП можно классифицировать по различным признакам: принципу действия, виду выходного сигнала, полярности выходного сигнала, элементной базе и др. По принципу действия наибольшее распространение получили ЦАП следующих видов: со сложением токов, с делением напряжения и со сложением напряжений. В микроэлектронном исполнении применяются только первые два типа.

По виду выходного сигнала ЦАП делят на два вида: с токовым выходом и выходом по напряжению. Для преобразования выходного тока ЦАП в напряжение обычно используются операционные усилители. По полярности выходного сигнала ЦАП принято делить на однополярные и двухполярные.

Управляющий код, подаваемый на вход ЦАП, может быть различным: двоичным, двоично-десятичным, Грея, унитарным и др. Кроме того, различными могут быть и уровни логических сигналов на входе ЦАП.

При формировании выходного напряжения ЦАП под действием управляющего кода обычно используются источники опорного напряжения. В зависимости от вида источника опорного напряжения ЦАП делят на две группы: с постоянным опорным напряжением и с изменяющимся опорным напряжением. Кроме этого, ЦАП классифицируют по основным характеристикам: количеству разрядов, быстродействию, точности преобразования, потребляемой мощности.

Все параметры ЦАП можно разделить на две группы: статические и динамические. К статическим параметрам ЦАП относят: разрешающую способность, погрешность преобразования, диапазон значений выходного сигнала, характеристики управляющего кода, смещение нулевого уровня и некоторые другие.

К динамическим показателям ЦАП принято относить: время установления выходного сигнала, предельную частоту преобразования, динамическую погрешность. Рассмотрим некоторые из этих параметров.

Разрешающая способность ЦАП определяется как величина, обратная максимальному количеству градаций выходного сигнала. Так, например, если разрешающая способность ЦАП составляет 10^{-5}, то это означает, что максимальное число градаций выходного сигнала равно 10^5. Иногда разрешающую способность ЦАП оценивают выходным напряжением при изменении входного кода на единицу младшего разряда, т. е. шагом квантования. Очевидно, что чем больше разрядность ЦАП, тем выше его разрешающая способность.

Погрешность преобразования ЦАП принято делить на дифференциальную и погрешность нелинейности. С ростом кода на входе ЦАП растет и выходное напряжение, однако при увеличении напряжения могут быть отклонения от линейной зависимости. Погрешностью нелинейности называют максимальное отклонение выходного напряжения от идеальной прямой во всем диапазоне преобразования.

Дифференциальной погрешностью называют максимальное отклонение от линейности для двух смежных значений входного кода.

Напряжение смещения нуля определяется выходным напряжением при входном коде, соответствующем нулевому значению.

Время установления $t_{уст}$ — это интервал времени от подачи входного кода до вхождения выходного сигнала в заданные пределы, определяемые погрешностью.

Максимальная частота преобразования — наибольшая частота дискретизации, при которой все параметры ЦАП соответствуют заданным значениям.

По совокупности параметров ЦАП принято делить на три группы: общего применения, прецизионные и быстродействующие. Быстродействующие ЦАП имеют время установления меньше 100 нс. К прецизионным относят ЦАП, имеющие погрешность нелинейности менее 0,1%.

Существует несколько схем, являющихся базой для построения многих разновидностей ЦАП соответствующего класса. Для формирования соответствующих уровней выходного напряжения (или тока) к выходу ЦАП подключается необходимое количество опорных сигналов $E_1, E_2 ... E_n$ (или токов $I_1, I_2 ... I_n$), либо устанавливают соответствующее дискретное значение коэффициента деления $K_1, K_2 ... K_n$.

На рисунке 5.28 приведена схема ЦАП с *суммированием токов.* В этой схеме используются *n* опорных источников тока $I_1, I_2 ... I_n$.

Входной код $b_1, b_2 ... b_n$ управляет ключами $S_1, S_2 ... S_n$, которые или подключают источники тока к нагрузке, или замыкают их накоротко. При, этом если $b_i = 0$, то соответствующий источник закорочен и в работе схемы не участвует. Если же $b_i = 1$, то соответствующий источник тока подключен к нагрузке. Результирующий ток равен сумме токов опорных источников, для которых $b_i = 1$. Напряжение на выходе будет равно результирующему току I_Σ умноженному на сопротивление $R_н$, т. е.

$$U_{вых} = I_\Sigma \ R_н \qquad (5.5)$$

Так, например, если входной код является двоичным, то результирующий ток определяется выражением:

$$I_\Sigma = I_0(b_1 2^{n-1} + b_2 2^{n-2} + ... + b_n 2^0) = I_0 N, \qquad (5.6)$$

где n — число двоичных разрядов входного тока, N — n-разрядное цифровое слово.

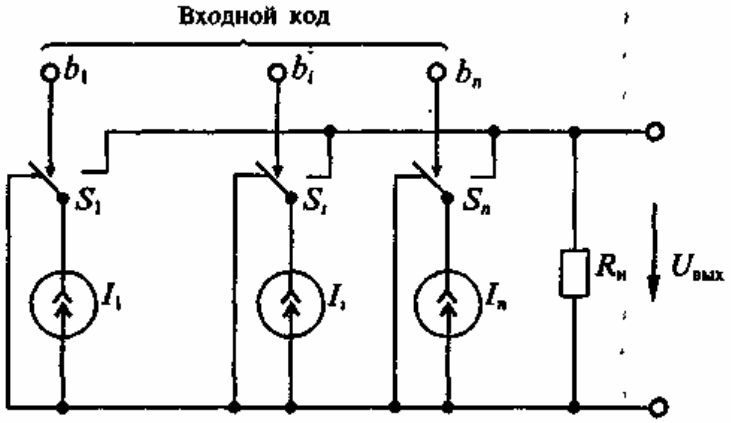


Рисунок 5.28 - Упрощенная схема ЦАП с суммированием токов

Упрощенная схема ЦАП со *сложением напряжений* приведена на рисунке 5.29. В этой схеме используется n опорных источников напряжения $E_1, E_2 ... E_n$. Входной код управляет ключами $S_1, S_2 ... S_n$, которые или подключают соответствующие источники опорного напряжения к нагрузке, или отключают их. Так же, как и для схемы с суммированием токов, при $b_i = 1$ соответствующий источник напря-

жения включен, а при b_i =0 — выключен. Результирующее напряжение на выходе равно сумме напряжений включенных опорных источников.

Так, например, для входного двоичного кода выходное напряжение определяется по формуле

$$U_{\Sigma} = U_{вых} = U_0(b_1 2^{n-1} + b_2 2^{n-2} + ... + b_n 2^0) = U_0 N. \qquad (5.7)$$

Упрощенная схема ЦАП *с делением опорного напряжения* E_0 приведена на рисунке 5.30. В этой схеме имеется один источник опорного напряжения и набор калиброванных сопротивлений $R_1, R_2 ... R_n$, с помощью которых напряжение опорного источника может быть разделено до значения, соответствующего входному коду.

Выходное напряжение для схемы, приведенной на рис. 8, определяется формулой

$$U_{вых} = \frac{E_0 R_н}{R_{\Sigma} + R_н}, \qquad (5.8)$$

где R_{Σ} — результирующее сопротивление устанавливаемое при помощи ключей $S_1, S_2 ... S_n$ которые управляются входным кодом.

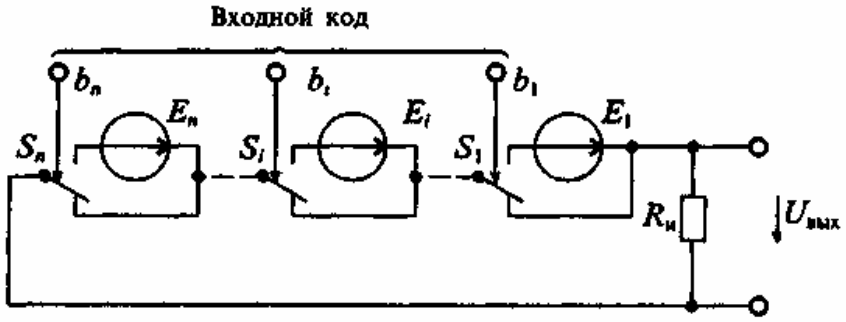


Рисунок 5.29 - Упрощенная схема ЦАП с суммированием напряжений

При $R_н$ = 0 эта схема превращается в управляемый источник тока, т. е. работает так же, как схема со сложением токов. Практически выполнить $R_н$ = 0 можно при помощи операционного усилителя с параллельной обратной связью.

Практическая схема ЦАП со сложением токов обычно выполня-

ется на различных резистивных матрацах и одном источнике опорного напряжения. На рисунок 5.31 приведена схема ЦАП с суммированием токов, в котором использован один источник опорного напряжения E_0, и резистивная матрица типа *R—2R*,

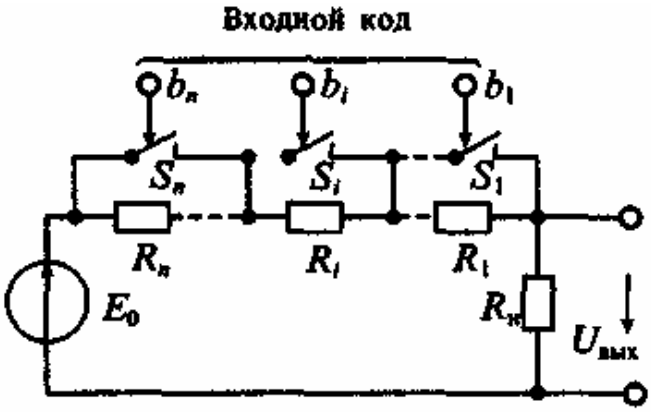


Рисунок 5.30 - Упрощенная схема ЦАП с делением напряжения

изображенная на рисунке 5.31, б. Особенность этой резистивной матрицы заключается в том, что при любом положении ключей $S_1, S_2 ... S_n$ входное сопротивление матрицы всегда равно *R*, а следовательно, ток, втекающий в матрицу, равен $I_0 = E_0 / R$. Далее он последовательно делится в узлах А, В, С по двоичному закону. Двоичный закон распределения токов в ветвях резистивной матрицы соблюдается при условии равенства нулю сопротивления нагрузки. Так как нагрузкой резистивной матрицы является операционный усилитель ОУ, охваченный отрицательной обратной связью через сопротивление R_{OC}, то его входное сопротивление равно нулю с достаточно высокой точностью.

Напряжение на выходе операционного усилителя определяется выражением

$$U_{вых} = \frac{E_0 R_{ос}}{R 2^n}(b_1 2^{n-1} + b_2 2^{n-2} + ... + b_n 2^0) = \frac{E_0 R_{ос}}{R 2^n} N, \qquad (5.9)$$

где b_i = 1, если ключ S_i, находится в положении, при котором ток протекает на инвертирующий вход ОУ, и b_i = 0, если ключ S_i, находится в положении, при котором ток протекает в общий вывод, *n* — число разрядов преобразователя.

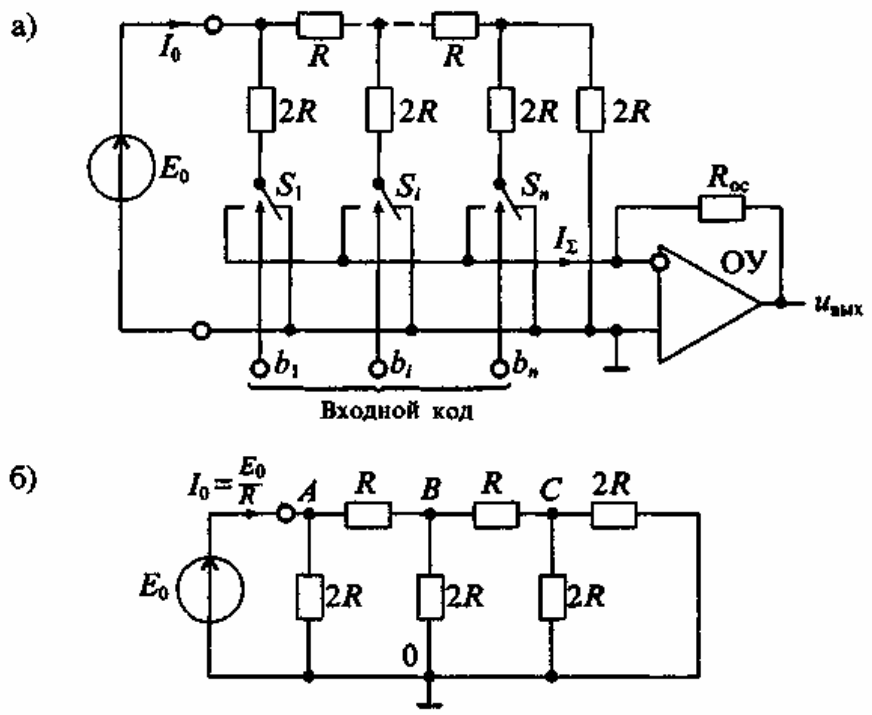


Рисунок 5.31 - Схема ЦАП со сложением токов на резистивной матрице типа *R—2R* (а) и структура резистивной матрицы (б)

Максимальное значение выходного напряжения (т. е. напряжение в конечной точке диапазона) имеет место при всех b_i = 1 и определяется по формуле:

$$U_{вых\max} = \frac{E_0 R_{oc}(1-2^{-n})}{R} = \frac{E_0 R_{oc}}{R} - h, \qquad (5.10)$$

где h — шаг квантования, т. е. приращение выходного напряжения при изменении входного кода на единицу младшего разряда:

$$h = \frac{E_0 R_{oc}}{2^n R}.$$

Как следует из формулы (5.9), выходное напряжение ЦАП зависит не только от входного кода *N*, но и от напряжения E_0 опорного источника. Если допустить, что напряжение E_0 меняется, то выходное напряжение ЦАП будет пропорционально произведению двух величин: входного кода и напряжения, поданного на вход опорного сигнала. В связи с этим такие ЦАП обычно называют перемножающими.

В интегральных микросхемах перемножающих ЦАП источник опорного напряжения отсутствует, но имеется вход для его подключения.

Интегральные микросхемы АЦП имеют буквенное обозначение ПВ, например К572 ПВ2 – АЦП двойного интегрирования с выходом на семисегментный светодиодный индикатор. Микросхемы ЦАП обозначаются буквами ПА, например К572 ПА2А – двенадцатиразрядный ЦАП.

5.3.5 Микропроцессорные средства информационного обмена

В состав МПК К580 входят интегральные микросхемы, обеспечивающие организацию информационного обмена между МП и внешними устройствами в различных режимах. Их основной особенностью является возможность программной настройки микросхемы на конкретные условия информационного обмена. К числу наиболее употребляемых схем относятся параллельный периферийный адаптер К580ИК55, программируемый контроллер прерываний К580ВН59 и контроллер прямого доступа к памяти К580ИК57.

Принципы построения микропроцессорных средств информационного обмена и порядок их программной настройки рассмотрим на примере БИС параллельного периферийного адаптера (ППА) К580ИК55.

Этот адаптер имеет 24 линии ввода-вывода, которые сгруппированы в три восьмиразрядных канала (порта) А, В и С, причем канал С разделен на две части: младшие его разряды РС3-РС0 составляют подканал С1, а старшие разряды РС7-РС4 – подканал С2. Программирование адаптера осуществляется с помощью управляющих слов, поступающих из процессора в специальный регистр ППА и задающих функциональное назначение каналов и их отдельных линий. С помощью программирования можно получить около ста различных конфигураций этой БИС, что позволяет обслуживать почти любое ВУ без дополнительной внешней логики.

Структурная схема ППА (рисунок 5.32) содержит следующие узлы: восьмиразрядный буфер данных (БД), имеющий три устойчивых состояния и обеспечивающий двунаправленную связь внутренней шины ППА с шиной данных микропроцессорного вычислительного устройства (МПВУ); три восьмиразрядных буфера каналов А, В и С, обозначенных БКА, БКВ и БКС1 БКС2; устройство управления запи-

сью и чтением (УУ Зп/Чт), которое управляет всеми внутренними пересылками данных, управляющих слов и слов состояния ППА; два устройства управления УУАС1 и УУВС2, связанные соответственно с каналами А и С1 и с каналами В и С2.

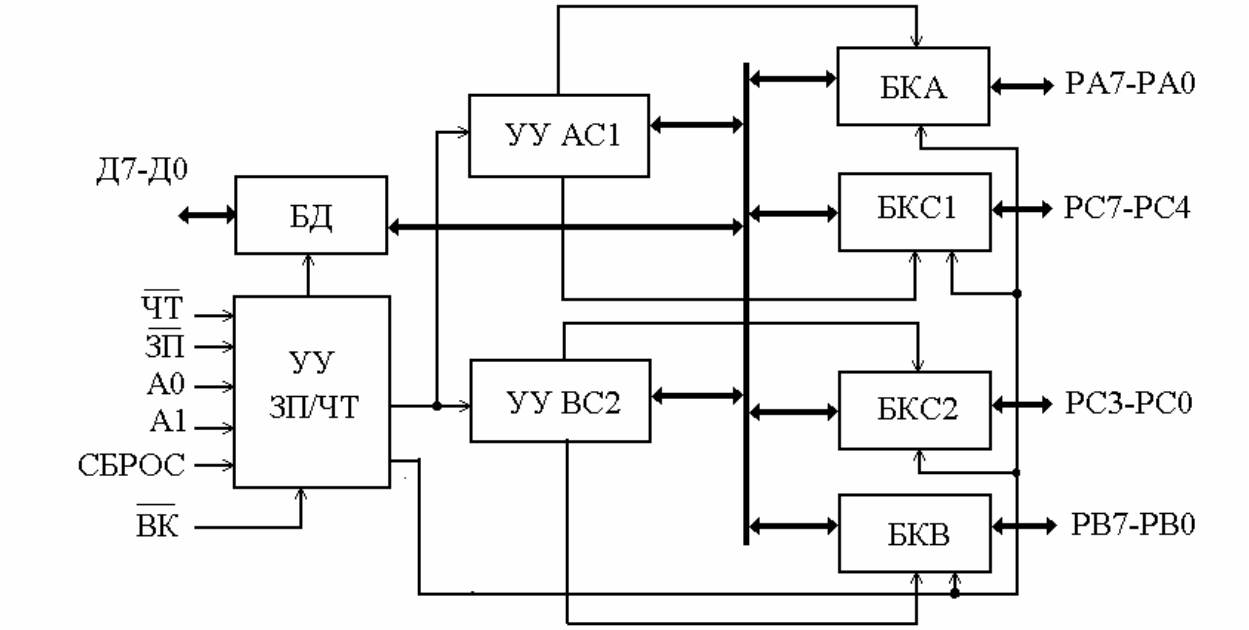


Рисунок 5.32

Выводы БИС ППА имеют следующее назначение.

ЧТ – вход для передачи сигнала считывания информации (байта данных или слова состояния) из ППА в процессор, поскольку операция считывания осуществляется с помощью команды ввода IN, то на вход ЧТ через элемент «И» подаются от микропроцессора сигналы ВВОД и ПРИЕМ, которые перекрываются во времени.

ЗП – вход для подачи сигнала занесения информации (байта данных или управляющего слова) из процессора в ППА. Поскольку эта операция осуществляется с помощью команды вывода OUT, то на вход ЗП подаются через элемент «И» сигналы процессора ВЫВОД и ВЫДАЧА.

ВК – вход для подачи сигнала разрешения работы ППА. На этом входе, как и на предыдущих двух, активным является нулевой уровень сигнала, если же ВК = 1, то буфер данных БД устанавливается в высокоимпедансное состояние. К входу ВК подключается одна из свободных линий шины адреса ША, выделенная для адресации данного ППА, либо один из выходов дешифратора адреса.

A1 , A0 – адресные входы, определяющие канал ППА, к которому производится обращение. Код 00 соответствует каналу А, 01 – каналу В, 10 – каналу С, 11 – управляющему регистру, в которой заносится управляющее слово при программировании ППА.

СБРОС – вход для подачи сигнала начальной установки ППА, этот вход подключается к системной шине сброса МПВУ.

При подаче сигнала СБРОС все каналы установлены в начальное состояние (т.е. 24 выходные линии находятся в высокоимпедансном состоянии). Перед началом выполнения операций ввода-вывода данных через ППА необходимо осуществить его программирование. С этой целью в ППА засылается управляющее слово, которое устанавливает требуемый режим работы адаптера.

Имеются три основных режима работы ППА: режим 0 – простой ввод-вывод; режим 1 – стробируемый ввод-вывод; режим 2 – двунаправленная магистраль. В режиме 0 могут работать все три канала, в режиме 1 – каналы А и В, в режиме 2 – только канал А.

При программировании ППА используются два формата управляющего слова; первый предназначен для задания режима работы и направления включения каналов; второй – для установки или сброса произвольного разряда канала С. Признак формата содержится в разряде Д7 управляющего слова: 1 соответствует формату 1, а 0 – формату 2. Остальные разряды управляющего слова в первом формате имеют следующее назначение (рисунок 5.33): Д0, Д1, Д3, Д4 – направление включения каналов С2, В, С1 и А соответственно (1 означает включение канала на ввод, а 0 – на вывод); Д2 – выбор режима работы группы каналов В и С2 (0 – режим «0», 1 – режим «1»); Д6, Д5 – выбор режима группы каналов А и С1 (00 – режим «0», 01 – режим «1», 10 или 11 – режим «2»).

При каждом изменении режима работы, т.е. при поступлении нового управляющего слова первого формата, все выходные регистры, а также триггеры состояния ППА сбрасываются. Режимы работы каналов А и В задаются независимо и они определяют режим работы подканалов С1 и С2. При работе каналов А и В в режиме 0 оба подканала С1 и С2 составляют отдельный восьмиразрядный канал, работающий в режиме 0. Если каналы А и В работают в режиме 1, или канал А работает в режиме 2, то линии канала С используются для обмена управляющими сигналами с ВУ и распределяются между каналами А и В.

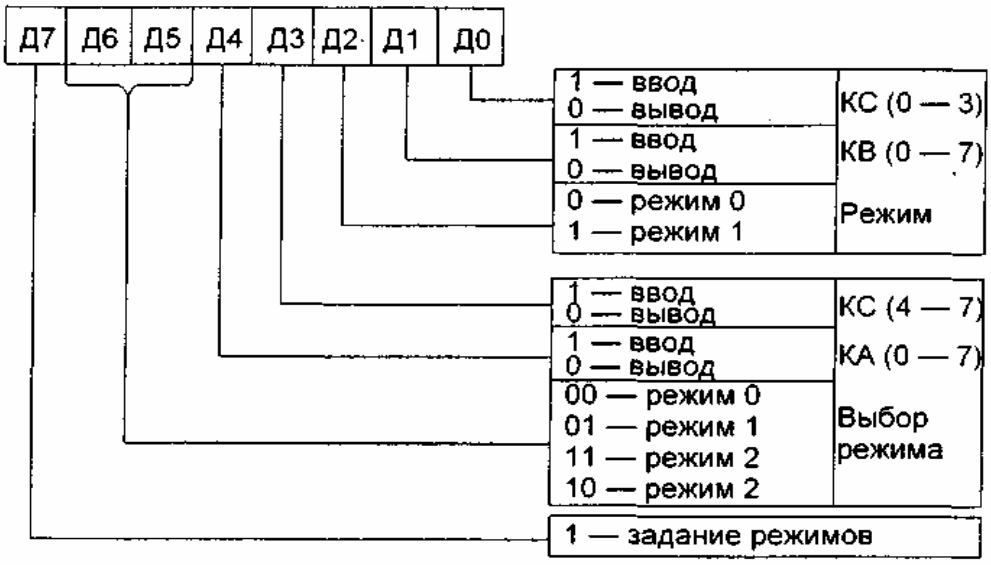


Рисунок 5.33 - Формат 1 управляющего слова

Разряды управляющего слова во втором формате имеют следующее назначение: Д0 – установка (Д0 = 1) или сброс (Д0 = 0) разряда канала С; Д3, Д2, Д1 – двоичный код номера разряда; Д6, Д5, Д4 – не используются.

Второй формат используется, например, для формирования запросов прерывания при работе ППА в режимах 1 и 2. Питание БИС К580ИК55 осуществляется от источника +5 В при токе потребления $I_n = 40$ мА.

Глава 6. ОСОБЕННОСТИ ЭКСПЛУАТАЦИИ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН

6.1 Основное содержание и этапы эксплуатации БВМ

Методы контроля работоспособности БВМ тесно связаны с эксплуатацией ее как объекта обслуживания. От того, насколько рационально они выбраны, будет зависеть эффективность эксплуатационного обслуживания БВМ.

Под эксплуатацией БВМ будем понимать комплекс организационных и технических мероприятий, направленных на поддержание БВМ в исправном состоянии и готовности к работе.

Организация эксплуатации современных бортовых ЦВМ включает в себя: обучение и постоянную проверку знаний техническим составом вверенной ему техники; учет работы БВМ и ведение технической документации; проведение регулярного контроля правильности функционирования БВМ и системы в целом; проведение ремонта БВМ.

Бортовая цифровая вычислительная машина должна содержаться в состоянии постоянной готовности к работе, ответственность за поддержание которой несут инженерно-технический состав и летные экипажи.

Из перечисленных основных положений эксплуатации БВМ следует, что высококачественную эксплуатацию БВМ можно достигнуть при хорошо организованном контроле правильности функционирования отдельных устройств и БВМ в целом. При выборе методов контроля основное внимание должно быть обращено на способность метода контроля к обнаружению ошибок, а также на объем аппаратуры и время, затрачиваемое на контроль.

Методы контроля, используемые в БВМ, можно разделить на программные и схемные (аппаратные).

6.2 Программные методы контроля работоспособности БВМ

Программные методы контроля правильности вычислений предназначены для проверки правильности реализации вычислительного процесса в различных режимах работы БВМ и правильности функционирования всех устройств БВМ.

Программные методы контроля основаны на включении в программное обеспечение БВМ специальных контролирующих программ и дополнительных соотношений в общий алгоритм. Программные методы делятся на программно-логический и тестовый контроль.

Программно-логический контроль основан на использовании информационной избыточности и предназначен для проверки правильности реализации вычислительного процесса. Информационная избыточность обеспечивается за счет включения в общий алгоритм дополнительных соотношений, позволяющих обнаруживать и исправлять ошибки, возникающие при вычислениях.

Наиболее широко применяются следующие способы программно-логического контроля: двойного просчета со сравнением результатов; контрольных соотношений; «усеченного» алгоритма; логического анализа результатов решения; подстановки.

Способ двойного просчета со сравнением результатов. Этот способ является наиболее распространенным. Основное его достоинство – простота реализации. Сущность его заключается в том, что вся рабочая программа разбивается на отдельные части и после выполнения какого-либо этапа вычислений производится контрольное суммирование всех команд, промежуточных и конечных результатов контролируемого этапа. Контрольная сумма запоминается и выполняется повторное вычисление этого этапа с последующим контрольным суммированием. Обе контрольные суммы сравниваются. При сравнении выполняется следующий этап вычислений. В случае не сравнения производится третий просчет данного этапа. При совпадении третьей контрольной суммы с одной из предыдущих выполняется следующий этап вычислений. В противном случае подается сигнал экипажу о сбоях в машине. Этот способ имеет следующие недостатки: он увеличивает время реализации алгоритма, что допустимо лишь при наличии избыточного быстродействия БВМ и позволяет обнаруживать и устранять лишь случайные ошибки, вызванные сбоями в машине.

При организации контроля способом двойного просчета необходимо правильно произвести разбиение алгоритма на контролируемые этапы. В качестве основного критерия при разбиении следует принять обеспечение требуемой вероятности обнаружения ошибки при минимальной затрате времени на контроль. Исследования показали, что существует оптимальное число контролируемых участков алгоритма, при котором время, расходуемое на контроль, минимально.

Опыт эксплуатации БВМ показывает, что поток случайных сбоев можно принять простейшим. Обозначим интенсивность потока случайных сбоев через λ, время решения задачи – $T_З$, количество контролируемых участков – N. Тогда время реализации одного контролируемого участка алгоритма

$$t = T_З / N. \qquad (6.1)$$

Следует учесть, что при организации контроля способом двойного просчета затрачивается дополнительное время $t_Д$ на каждом контролируемом участке задачи на контрольное суммирование, сравнение контрольных сумм, принятие решений на третий просчет или продолжение вычислений.

При принятых допущениях и введенных обозначениях время реализации алгоритма

$$T_A = 2N (t+t_д) + 2N (t+t_д) \lambda (t+t_д) = 2N (t+t_д) + 2 (t+t_д)^2 \lambda N. \qquad (6.2)$$

Подставляя значение t из (6.1) в (6.2), получим

$$T_A = 2N (T_З / N + t_д) + 2 (T_З / N + t_д)^2 \lambda N. \qquad (6.3)$$

Выберем N из условия минимального времени реализации алгоритма. Продифференцировав выражение (6.3) по N и приравняв производную нулю, получим

$$dT_A / dN = t_д – (T_З^2 \lambda) / N^2 + t_д^2 \lambda = 0.$$

Отсюда

$$N_{опт} = T_З \sqrt{\lambda /[t_Д (1+ \lambda t_Д)]}.$$

При экспоненциальном законе распределения сбоев и оптимальной длине участка программы вероятность бессбойной работы на одном участке будет:

$$P(t) = e^{-\lambda t} = e^{-\lambda (T_З / N_{опт})} = e^{-\lambda \sqrt{[t_Д (1+\lambda t_Д)]/ \lambda}}.$$

На основании теоремы о повторении опытов можно определить вероятность правильной реализации алгоритма при наличии двойного- тройного просчета.

Она равна:

$$P(T_A) = \{P(t)P(t) + [1 - P(t)]P(t)P(t) + P(t)[1 - P(t)]P(t)\}^N = \left[e^{-2\lambda t} + 2e^{-2\lambda t}\left(1 - e^{-\lambda t}\right)\right]^N.$$

Способ контрольных соотношений. Позволяет определить не только ошибки, появляющиеся за счет случайных сбоев, но и ошибки, появляющиеся за счет отказов. Сущность способа заключается в том, что в общий алгоритм включаются различные математические соотношения, позволяющие проверить правильность решения основной задачи путем их реализации. Такими соотношениями могут, например, быть зависимости вида

$$\sin^2\alpha + \cos^2\alpha = 1 \qquad \text{или} \qquad \sum_{t=1}^{n} P_i = 1.$$

т.е. сумма вероятностей всех возможных значений случайной величины в полной группе событий равна единице. При выполнении контрольных соотношений с заданной точностью проводится дальнейшее решение основной задачи. При невыполнении контрольных соотношений либо повторяется данный участок задачи, либо машина останавливается о чем сообщается экипажу. Способ контрольных соотношений может дать значительный выигрыш во времени по сравнению со способом двойного просчета.

Способ «усеченного» алгоритма. Данный способ позволяет выявлять и устранять ошибки за счет случайных сбоев и систематических отказов, если «усеченный» алгоритм значительно отличается от основного. Этот способ предполагает наличие упрощенного алгоритма основной задачи. Упрощенный алгоритм должен быть меньше основного. Реализация «усеченного» алгоритма совместно с основным и совпадение результатов в пределах заданной точности позволяют судить о правильности хода вычислительного процесса.

Способ логического анализа результатов решения. Способ позволяет выявлять как случайные, так и систематические ошибки. Сущность его заключается в сравнении некоторых параметров задачи и их приращений, вычисленных в ходе решения задачи, с заранее известными пределами их изменения. Способ используется в том случае, если заранее известны законы изменения некоторых параметров.

Способ подстановки. Заключается в том, что после получения ряда искомых величин решается обратная задача. В качестве исходных данных выбираются найденные значения неизвестных и определяются некоторые величины, которые при прямых вычислениях использовались в качестве исходных данных. Затем сравнивают эти величины с исходными данными при прямом вычислении. Совпадение в пределах заданной точности свидетельствует об отсутствии ошибок в вычислениях. Например, при решении систем алгебраических уравнений в качестве исходных данных могут быть приняты полученные в вычислении корни $x_1, x_2, \ldots, x_k$; в качестве неизвестных – какие-либо коэффициенты, определяемые при обратном решении задачи. Сравнение полученных коэффициентов с имеющимися в пределах заданной точности говорит о том, что система решена верно. Аналогичным способом можно воспользоваться при решении систем дифференциальных уравнений.

При разработке общего алгоритма бортовой ЦВМ необходимо использовать наиболее эффективные способы контроля правильности вычислений, позволяющие значительно повысить надежность обработки информации на борту самолета с минимальными затратами машинного времени.

Все рассмотренные способы контроля вычислений являются эффективными лишь при условии правильности хода реализации программ БЦВМ.

Контроль хода выполнения программ БЦВМ: каждая программа, реализующая тот или иной режим работы БЦВМ и системы в целом, составляется из отдельных подпрограмм программой-диспетчером. Правильность хода выполнения программы определяется: контролем последовательности включения подпрограмм; контролем длительности работы подпрограмм; контролем выполнения переходов и прерываний программы.

Контроль последовательности включения подпрограмм: режим работы БЦВМ и системы обеспечивается соответствующей программой, состоящей из отдельных подпрограмм S_1, выполняемых поочередно. Способ контроля заключается в следующем.

За время t_1 подпрограмма S_i включается в работу не менее чем один раз и в результате каждого выполнения подпрограммы S_i изменяется некоторый параметр $R_i\,(t)$, хранящийся в оперативной памяти БЦВМ. Проверкой соотношения $R_i\,(t) \neq R_i\,(t + \tau_i)$ можно установить,

что подпрограмма S_i выполнялась. Для каждого выполняемого режима работы БЦВМ заранее известны связи между отдельными подпрограммами. Детерминированность связей между подпрограммами используется для контроля последовательности их выполнения. В этом случае каждая подпрограмма при выполнении фиксирует в определенной ячейке памяти заранее закрепленный за данной подпрограммой условный код. Выполнение следующей подпрограммы начинается лишь после анализа условного кода, записанного при выполнении предшествующей подпрограммы.

Контроль длительности работы подпрограмм: предполагает контроль длительности работы отдельных участков программы и позволяет обнаруживать нарушения реализации программ за счет зацикливания и других причин, приводящих к увеличению времени реализации отдельных подпрограмм программы. Для этой цели используются счетчики относительного времени, предназначенные для подсчета временных интервалов. В этом случае накладываются определенные ограничения на длительность τ_i выполнения подпрограммы S_i. В начале каждой подпрограммы на счетчик относительного времени специальной командой заносится код, соответствующий допустимой относительной длительности исполнения данной подпрограммы. При каждой реализации подпрограммы S_i из содержимого счетчика относительного времени вычитается единица. При наличии в счетчике относительного времени нуля вырабатывается сигнал ошибки, который вызывает прерывание программы и переход к подпрограмме анализа ошибок. В противном случае происходит переход к выполнению следующей подпрограммы.

Контроль правильности выполнения переходов и прерываний программы: осуществляется с помощью команд условной и безусловной передач управления и основан на программном блокировании сигнала ошибки. Идея этого контроля заключается в том, что перед выполнением перехода специальной командой в цепь модификации адреса и выполнения перехода подается сигнал ошибки, поступление которого может быть заблокировано с помощью команд блокировки, располагаемых во всех местах программы, куда происходит переход.

При неправильном выполнении перехода сигнал ошибки не будет заблокирован и вызовет при этом прерывание и включение в работу подпрограммы анализа сбоев. Способ обеспечивает высокую вероятность обнаружения как случайных, так и систематических

ошибок любой кратности, но требует дополнительного оборудования и специальных команд в программе.

Прерывание программы является характерной особенностью работы БЦВМ, реализующей свои алгоритмы в реальном масштабе времени. В процессе работы БЦВМ в зависимости от режима работы системы и объема поступающей и выдаваемой информации производятся многочисленные прерывания программы, насчитывающие сотни прерываний в секунду.

Для контроля правильности выполнения прерываний программы часто используется рассмотренный способ программного блокирования сигнала ошибки. Отличие состоит лишь в том, что формирование сигнала ошибки осуществляется схемным путем, а прерывание может быть в любом месте программы. Блокирование сигнала ошибки осуществляется одной из первых команд вклинивающейся подпрограммы. Контроль перехода к основной программе после прерывания осуществляется проверкой правильности восстановления состояния основных результатов регистров и ячеек памяти БЦВМ, использовавшихся вклинивающейся подпрограммой. Для этой цели применяют метод многократного хранения переменной информации в памяти машины и последующее её сравнение.

Сигнал ошибки вырабатывается при не сравнении содержимого хотя бы одного регистра или ячейки памяти машины.

Тестовый контроль. Программно-логический контроль, предназначенный для защиты алгоритмов от ошибок, возникающих в ходе вычислительного процесса, не ставит своей целью установление факта наличия неисправности в каком-либо устройстве, обнаружение и устранение имеющейся неисправности.

С помощью тестового контроля обнаруживают в каком-либо устройстве неисправности, выполняют их локализацию и сигнализацию о них. В настоящее время он является основным средством проверки правильности функционирования БЦВМ на всех этапах ее эксплуатации и применения. Основным назначением тестового контроля является обнаружение систематических ошибок, возникающих в устройствах вследствие внезапных и постепенных отказов.

В процессе эксплуатации БВМ используются: испытательные тесты комплексной проверки БВМ и диагностические тесты.

Испытательные тесты комплексной проверки БВМ. Они применяются для обнаружения неисправностей в машине и позволяют

контролировать совместную работу всех устройств машины. В качестве испытательных тестов часто используют одну или несколько из решаемых БВМ задач. Исходными данными при этом выбирают такие, результаты решения для которых заранее известны. Испытательные тесты в этом случае позволяют проверить не только работоспособность БВМ, но и функционирование системы в целом. Тестовые программы этого типа не требуют дополнительной емкости памяти.

Диагностические тесты. Их применяют для определения места неисправности и используют после установления факта наличия неисправности с помощью испытательных тестов комплексной проверки БВМ. Эти тесты представляют собой программы специально подобранных примеров, которые требуют для своей реализации выполнения всех операций, имеющихся в системе команд машины. При правильном выполнении теста вырабатывается сигнал «исправно». При наличии отказа этот сигнал не вырабатывается.

Тестовые программы хранятся в постоянном запоминающем устройстве (ПЗУ) и включают в себя тесты, проверяющие АЛУ, ПЗУ, ОЗУ, УУ, УВВ и т.д.

6.3 Помехоустойчивое кодирование сигналов и его применение для схемного контроля БВМ

Схемный контроль обеспечивает контроль машины или отдельных ее устройств с помощью дополнительного оборудования, включенного БЦВМ.

Схемный контроль включает в себя: контроль с использованием корректирующих кодов и контроль по модулю.

6.3.1 Общие сведения об избыточном кодировании сигналов

Работа цифровых устройств невозможна без передачи информации по различным каналам информационного обмена. Передачу информационных сообщений всегда можно свести к передаче двоичных чисел, представленных в виде кодов.

Исследованием свойств кодов ограниченной длительности и поиском оптимальных структур технически реализуемых кодов занимается теория кодирования. Теория кодирования развивается в двух главных направлениях:

поиск кодов, позволяющих в каналах без помех максимально устранить избыточность источника и тем самым повысить эффективность передачи информации (теория экономичного кодирования);

поиск кодов, повышающих достоверность передачи информации в каналах с помехами (теория помехоустойчивого кодирования).

Поскольку под помехой для информационного канала может быть принято не только мешающее передаче информации внешнее воздействие, но и сбои в оборудовании, приводящие к потере информации, то в общем виде все информационные каналы можно считать каналами с помехами.

Рассмотрим общую постановку задачи помехоустойчивого кодирования при ограниченной длительности кодируемых сообщений. От источника сообщений поступают последовательности символов $b_1, b_2, ..., b_n$, которые называют информационными последовательностями. При кодировании каждой информационной последовательности $b_1, b_2, ..., b_n$ приводится во взаимно-однозначное соответствие передаваемое сообщение (кодовая последовательность) $\beta_1, \beta_2, ..., \beta_n$. В результате действия помех на выходе канала будет принято сообщение $\beta'_1, \beta'_2, ..., \beta'_n$, часть символов которого может отличаться от символов передаваемого сообщения: $\beta'_\nu \neq \beta_\nu$. Если при приеме каждый символ обязательно идентифицируется с одним из символов алфавита канала (канал без стирания), т.е. всегда $\beta'_\nu \in B$, то единственной формой ошибки может быть трансформация символов. Если же по некоторым из принятых символов решение об их идентификации с определенным символом алфавита B не принимается и они оказываются вовсе не опознанными (канал со стиранием), то могут иметь место два вида ошибок: трансформация символов и неопознание (стирание) символа. Неопознанные (стираемые) символы обозначим Φ, где $\Phi \notin B$.

Помехоустойчивое кодирование сообщений имеет задачей обеспечить с достаточно высокой вероятностью либо обнаружение ошибок, либо обнаружение и исправление этих ошибок. Соответственно различают коды с обнаружением ошибок и коды с исправлением ошибок. Код с обнаружением ошибок обеспечивает стирание или особую отметку той части сообщений, в которой обнаружены ошибки. Такой код эффективней в системах с обратными информационными связями, когда имеется возможность повторно запросить часть

сообщения, в котором обнаружены ошибки. При отсутствии обратных информационных связей код с обнаружением ошибок эффективен лишь в таких системах, которые менее критичны к пропуску (стиранию) части сообщения, чем к неверному его приему. Код с исправлением ошибок позволяет получить верные сообщения, несмотря на наличие искажений отдельных символов (в пределах корректирующей способности кода). Не существует кодов конечной длины, позволяющих обнаружить и, тем более, исправить все возможные ошибки. Можно построить лишь коды, обнаруживающие или исправляющие некоторое число ошибок определенного вида (наиболее вероятных или наиболее опасных).

Коды, обнаруживающие и исправляющие ошибки, называются помехоустойчивыми или корректирующими. Корректирующая способность кода основывается на его избыточности. При отсутствии избыточности, когда любая комбинация символов алфавита канала может представлять передаваемое сообщение, достоверное обнаружение ошибок невозможно: на приемной стороне неизвестно, какое именно сообщение могло передаваться (в противном случае его передача не несла бы информации и была бы лишена смысла). Для того чтобы избыточностью сообщения можно было воспользоваться для обнаружения или исправления ошибок, она должна вноситься по определенным правилам. При рассмотрении корректирующей способности кодов будем учитывать лишь избыточность, вносимую в сообщение при кодировании, сообщения же, поступающие от источника, будем считать безызбыточными. В частном случае, когда алфавиты источника и канала имеют одинаковый объем m, внесение избыточности означает увеличение числа символов в передаваемой последовательности по отношению к информационной последовательности. Например, для двоичного кода ($m=2$) с длиной информационного сообщения n и числом контрольных разрядов l передаваемое сообщение имеет вид:

$$X=\left\{x_0, x_1, \ldots, x_{n-1}, x_n, x_{n+1}, \ldots, x_{n+(l-1)}\right\},$$

где $x_i \in \{0,1\}$.

По форме внесения избыточности различают систематические и несистематические коды. В систематических кодах символы информационной последовательности входят без изменения в передаваемое сообщение, занимая в нем отведенные им позиции, а кодирование сводится к внесению в передаваемое сообщение дополнительных

(избыточных) символов, связанных определенной зависимостью с символами информационной последовательности. В явном виде они в передаваемое сообщение не входят, а могут быть установлены по известным зависимостям, связывающим их с символами передаваемого сообщения.

По способу кодирования различают блочные и непрерывные коды. При блочном кодировании информационная последовательность разбивается на блоки символов $(b_1, b_2, ..., b_k)$ фиксированной длины k, каждому из которых ставится в соответствие определенная комбинация символов алфавита канала $(\beta_1, \beta_2, ..., \beta_n)$, называемого кодовым словом. Код при блочном кодировании определяет закон формирования кодового слова, отвечающего данному блоку информационных символов.

В случае систематического блочного кода кодовое слово, отвечающее блоку информационных символов $(b_1, b_2, ..., b_k)$, может быть представлено в виде $(b_1, b_2, ..., b_k, \beta_k, \beta_{k+1}, ..., \beta_n)$, где $\beta_{k+1}, ..., \beta_n$ - избыточные символы. Блочный код, приводящий в соответствие блоку информационных символов длиной k кодовое слово длиной n, будем обозначать (k, n). Избыточные символы не несут дополнительной информации об источнике сообщения (они однозначно определяются информационными символами $b_1, b_2, ..., b_k$), Поэтому кодовое слово несет то же количество полезной информации, что и соответствующий блок информационных символов. При безызбыточности входного сообщения блок из k информационных символов (а значит, и кодовое слово) несет количество информации $H = k \log_2 m$, где m - объем алфавита источника.

Максимальное количество информации, которое может содержать слово из n символов канала при том же объеме алфавита m, равно $H_{\max} = n \log_2 m$.

Поэтому избыточность кода (k, n)

$$\rho_u = 1 - \frac{H}{H_{\max}} = 1 - \frac{k \log_2 m}{n \log_2 m} = \frac{n-k}{n} = \frac{r}{n},$$

где $r = n - k$ - число избыточных символов, вносимых при кодировании.

При непрерывном кодировании каждый символ передаваемого сообщения определяется рекуррентными соотношениями,

связывающими его с соответствующими символами информационной последовательности:

$$\beta_{i,j} = \left[\sum_{\nu=-q}^{S=1} C_{\nu j} b_{i+\nu} \right] (\mathrm{mod}\, m), \quad j = 0,1,...,l-1. \qquad (6.4)$$

Значение правой части выражения (6.4) определяется по модулю m для того, чтобы полученные значения символов $\beta_{i,j}$ принадлежали алфавиту канала, т.е. принимали одно из значений 0, 1, ..., m -1.

Соотношение (6.1) на i - м шаге кодирования группе символов входной последовательности $b_{i-q},...,b_i,...,b_{i+s-1}$ ставит в соответствие l символов кодовой последовательности $\beta_{i,0},...,\beta_{i,l-1}$; на $(i+1)$- m шаге группе символов $b_{i-q+1},...,b_{i+1},...,b_{i+s}$, смещенной на одну позицию, ставит в соответствие следующие l символов кодовой последовательности $\beta_{i+1,0},...,\beta_{i+1,l-1}$ и т.д. Таким образом, на каждый символ информационной последовательности приходится l символов передаваемой кодовой последовательности, что соответствует избыточности кода $\rho_u = (l-1)/l$. Правая часть соотношения (6.4) представляет как бы свертку соответствующего участка информационной последовательности, поэтому коды и называют сверточными.

Непрерывные (сверточные) коды могут быть как систематическими, так и несистематическими. Систематические сверточные коды получаются в частном случае, когда для одного из значений индекса j (например, $j = 0$) коэффициенты в формуле (6.4) принимают значения

$$C_{\nu 0} = \begin{cases} 0, \text{ при } \nu \neq 0, \\ 1, \text{ при } \nu = 0 \end{cases}$$

и соответственно $\beta_{i0} = b_i$. В этом случае передаваемая последовательность имеет вид $...b_i, \beta_{i,1},..., \beta_{i,l-1}, b_{i+1}, \beta_{i+1,1},..., \beta_{i+1,l+1},...$.

6.3.2 Коды, контролирующие ошибки

Одним из наиболее широко распространенных способов контроля передачи, хранения и обработки цифровой информации, в том числе и в бортовых ЭВМ, является контроль по модулю. При контроле

по модулю используется избыточная информация, представляемая дополнительными разрядами в информационном слове. Количество дополнительных разрядов определяется значением модуля, по которому производится контроль.

Суть метода заключается в определении и анализе по некоторым правилам контрольных кодов, представляющих собой наименьшие остатки от деления на модуль самих чисел (числовой контроль) или суммы их цифр (цифровой контроль).

В первом случае контрольные разряды определяются из соотношения:

$$\alpha_m = ост \frac{x}{m},$$

где x - контролируемое число;

α_m - наименьший остаток от деления числа x на модуль m.

Например, при контроле по модулю 2 для числа x = 1001 получаем

$$\alpha_2 = ост \frac{1001}{10} = 1.$$

В этом случае избыточный код имеет значение

$$x_2 = 1001\ 1$$

↑— контрольный разряд.

Очевидно, все нечетные числа будут иметь контрольный разряд равный единице, а четные - нулю. Контроль в этом случае сводится к следующим действиям: формированию контрольного разряда на передающей позиции; передаче избыточного кода; проверке сформированного по тем же правилам контрольного разряда на приемной позиции с полученным по каналу контрольным разрядом.

В случае числового контроля по модулю два ошибка не будет обнаружена в том случае, если в результате искажений четное число останется четным, а нечетное - нечетным. Вероятность такого события близка к 0,5.

Чем больше значение модуля, тем больше число контрольных разрядов добавляется в передаваемое сообщение. В то же время больше вероятность обнаружения ошибок.

Второй метод контроля (цифровой контроль) по модулю предусматривает суммирование значений разрядов числа по некоторому

модулю в соответствии с выражением

$$\alpha_m = \left[\sum_{i=0}^{k-1} b_i \right] \bmod m,$$

где b_i - значение i-го разряда числа;

k - число разрядов.

Например, для числа x = 1001 контрольный разряд при m = 2 имеет значение

$$\alpha_2 = [1+0+0+1] \bmod 2 = 0$$

и поэтому избыточный код будет следующим:

$$x_2 = 1001\,0$$

↑— контрольный разряд.

Очевидно, что при таком кодировании для m = 2 все неисправности, приводящие к искажению одного разряда (трех, пяти и т.д.), обнаруживаются. Если же искажаются два разряда (четыре, шесть и т.д.), то ошибка не будет обнаружена. Однако вероятность появления ошибок одновременно в двух разрядах существенно меньше появления одиночных ошибок.

Схемные методы контроля по модулю 2 называются также контролем по четности. Если контрольным разрядам присваиваются инверсные значения (α_2), то контроль называется проверкой по нечетности.

Контроль с проверкой по нечетности (четности) имеет небольшую избыточность и не требует больших затрат оборудования для своей реализации. Этот метод широко используется в цифровых устройствах для контроля передачи информации как внутри устройства, так и вне его, а также для контроля информации, считанной из запоминающего устройства.

6.3.3 Коды, исправляющие ошибки

В исправляющих кодах двоичное число, составленное из контрольных разрядов (синдром кода), должно определять порядковый номер искаженного разряда кодового слова. Далее значение искаженного разряда инвертируется, чем достигается его коррекция.

Примером таких кодов является код Хэмминга. В этом коде контрольные суммы составляются так, чтобы в сумму s_j входили символы кодового слова, имеющие единицу в j - м разряде в двоичной записи их порядкового номера. Нетрудно видеть, что при этом символы, порядковый номер которых в двоичной записи выражается единицей j - го разряда, т.е. равен 2^{j-1}, $j = 1, 2, 3, ...,$ войдут только в одну контрольную сумму s_j. Именно на этих позициях (1-й, 2-й, 4-й, 8-й, ..., 2^i) в коде Хэмминга и располагаются проверочные символы $\beta_1, \beta_2, \beta_4, \beta_8, ...\beta^{2i}$. Соответствующим их выбором все контрольные суммы обращаются в нуль. При отсутствии искажений все контрольные суммы останутся равными нулю. Одиночная ошибка, например искажение l- го разряда кодового слова, приведет к обращению в единицу контрольных сумм, номера которых соответствуют номерам разрядов двоичной записи числа l, содержащих единицы. Поэтому, записывая значения контрольных сумм в порядке убывания их номеров, т.е. записывая синдром $...s_4, s_3, s_2, s_1$, получаем в двоичной записи номер l искаженного разряда.

Для наглядности двоичная запись номеров разрядов кодового слова сведена в таблицу 6.1 (таблица приведена для информационных сообщений, представленных в виде пятнадцатиразрядных кодов).

Для пятнадцатиразрядного кода в соответствии с таблицей 6.1 контрольные суммы имеют вид:

$$\begin{aligned} s_1 &= \beta_1 \oplus b_3 \oplus b_5 \oplus b_7 \oplus b_9 \oplus b_{11} \oplus b_{13} \oplus b_{15}; \\ s_2 &= \beta_2 \oplus b_3 \oplus b_6 \oplus b_7 \oplus b_{10} \oplus b_{11} \oplus b_{14} \oplus b_{15}; \\ s_3 &= \beta_4 \oplus b_5 \oplus b_6 \oplus b_7 \oplus b_{12} \oplus b_{13} \oplus b_{14} \oplus b_{15}; \\ s_4 &= \beta_8 \oplus b_9 \oplus b_{10} \oplus b_{11} \oplus b_{12} \oplus b_{13} \oplus b_{14} \oplus b_{15}. \end{aligned} \qquad (6.5)$$

где b_i - значение информационных символов кода;

$\beta_1, \beta_2, \beta_4, \beta_8$ - значения проверочных символов;

$\oplus$ - знак суммирования по модулю два.

Таблица 6.1

Порядковый номер разряда	Значения разрядов двоичной записи порядкового номера разрядов кода			
	4	3	2	1 (младший)
1 (младший)	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

Значения проверочных символов β_i, отображающие в нуль контрольные суммы равны

$$\begin{aligned}
\beta_1 &= b_3 \oplus b_5 \oplus b_7 \oplus b_9 \oplus b_{11} \oplus b_{13} \oplus b_{15};\\
\beta_2 &= b_3 \oplus b_6 \oplus b_7 \oplus b_{10} \oplus b_{11} \oplus b_{14} \oplus b_{15};\\
\beta_4 &= b_5 \oplus b_6 \oplus b_7 \oplus b_{12} \oplus b_{13} \oplus b_{14} \oplus b_{15};\\
\beta_8 &= b_9 \oplus b_{10} \oplus b_{11} \oplus b_{12} \oplus b_{13} \oplus b_{14} \oplus b_{15}.
\end{aligned} \qquad (6.6)$$

Процедура формирования исправляющего кода Хэмминга включает, таким образом, следующие операции:

1) расположение информационных символов в порядке возрастания разрядов (справа налево) с сохранением незанятыми разрядов с номерами 2^{j-1}, т.е. 1-го, 2-го, 4-го, 8-го и т.д.;

2) вычисление проверочных символов, занимающих разряды с номерами 2^{j-1}, по формулам (6.6).

Процедура определения передаваемого информационного кода при использовании исправляющего кода Хэмминга включает:

1) вычисление контрольных сумм по формулам (6.5);

2) определение порядкового номера искаженного символа, если

не все контрольные суммы равны нулю, и исправление его на другой двоичный символ;

3) выделение информационного кода исключением разрядов с номерами $2^{j-1}, \quad j = 1, 2, \ldots$.

Например, если информационный код имеет вид 101001, то его символы располагаются в коде Хэмминга в следующей последовательности:

10	9	8	7	6	5	4	3	2	1
1	0	β_8	1	0	0	β_4	1	β_2	β_1

Определим по формулам (6.3) значения проверочных символов:

$$\beta_1 = b_3 \oplus b_5 \oplus b_7 \oplus b_9 = 1 \oplus 0 \oplus 1 \oplus 0 = 0;$$

$$\beta_2 = b_3 \oplus b_6 \oplus b_7 \oplus b_{10} = 1 \oplus 0 \oplus 1 \oplus 1 = 1;$$

$$\beta_4 = b_5 \oplus b_6 \oplus b_7 = 0 \oplus 0 \oplus 1 = 1;$$

$$\beta_8 = 0.$$

С учетом контрольных разрядов получим следующий код:

1001001110.

Если, например, при приеме пятый разряд этого кода ошибочно будет принят как единица, то контрольные суммы (синдромы) будут иметь значения:

$$s_1 = \beta_1 \oplus b_3 \oplus b_5 \oplus b_7 \oplus b_9 = 0 \oplus 1 \oplus 1 \oplus 1 \oplus 0 = 1;$$

$$s_2 = \beta_2 \oplus b_3 \oplus b_6 \oplus b_7 \oplus b_{10} = 1 \oplus 1 \oplus 0 \oplus 1 \oplus 1 = 0;$$

$$s_3 = \beta_4 \oplus b_5 \oplus b_6 = 1 \oplus 1 \oplus 0 \oplus 1 = 1;$$

$$s_4 = \beta_8 = 0.$$

Значение синдрома, определяющего номер искаженного разряда в двоичной записи, равно $s_4 s_3 s_2 s_1 = 0101$, что соответствует двоичному представлению цифры 5 десятичной системе счисления. Для исправления ошибки требуется изменить символ в 5-м разряде.

6.3.4 Каскадные коды

Возможности кода по обнаружению и исправлению ошибок определяются величиной избыточности кода. Причем практически безошибочная передача информации достигается лишь при весьма значительных длинах кодовых слов. Поиск эффективных схем кодирования и декодирования, сложность реализации которых с ростом длины кода увеличивалась бы как можно медленнее, привел к появлению каскадных кодов.

Обобщенный каскадный код определяется так. Двоичное слово α длины $n = n_1 \cdot n_2$ является кодовым словом обобщенного каскадного кода порядка m тогда и только тогда, когда все связанные со словом α, векторы γ_i, $i = \overline{1, m+1}$, представляют собой кодовое слово соответствующих $(i - x)$ кодов второй ступени.

Обобщенному каскадному коду можно дать достаточно наглядную и удобную при многих рассуждениях геометрическую трактовку.

Последовательность α двоичных символов, образующих кодовое слово обобщенного каскадного кода, разместим в прямоугольную таблицу размером $n_1 \cdot n_2$ так, что векторы α_j являются ее столбцами. Тогда векторы $\alpha_{i1}, \alpha_{i2}, ..., \alpha_{n2}$ образуют i-й горизонтальный блок B_i этой таблицы, общее число которых равно $m+1$, где m - порядок кода.

Каждый горизонтальный блок B_i, состоящий из α_i строк, в свою очередь, разобьем на два блока - K_i и L_i, содержащих соответственно b_i и $n_2 - b_i$ столбцов.

В блоках K_i расположены информационные символы каскадного кода, а в блоках L_i - его проверочные символы (рисунок 6.1). Если $b_{m+1} = 0$, то блок B_{m+1} совпадает с L_{m+1} и содержит одни лишь проверочные символы (рисунок 6.2).

В этом случае столбцы α_j являются кодовыми словами некоторого двоичного линейного кода длины n_1.

Отметим, что величины $a_i > 0$ и $b_i \geq 0$, определяющие внутреннюю структуру обобщенного каскадного кода, могут выбираться произвольно, при этом общее число k информационных символов и r проверочных символов равны

$$k = \sum_{i=1}^{m+1} a_i b_i; \quad r = \sum_{i=1}^{m+1} a_i \left(n_2 - b_1\right).$$

Использование многоступенчатого кодирования, где на каждой ступени используется свой избыточный код, обеспечивает высокую эффективность кодирования при сравнительно меньших значениях избыточности.

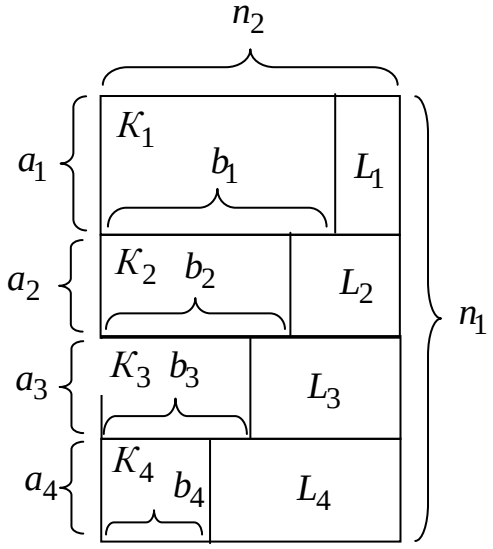


Рисунок 6.1

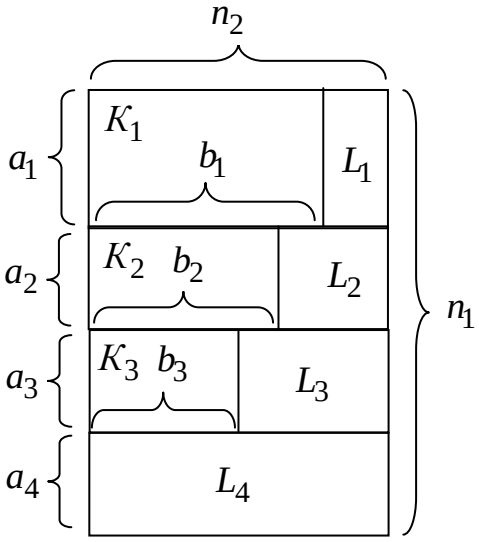


Рисунок 6.2

Вероятностная оценка эффективности схемного контроля. В БЦВМ с большой достоверностью можно считать ошибки в различных разрядах кода x независимыми случайными величинами, имеющими одинаковую вероятность появления. Тогда для i – кратных ошибок в n – разрядном коде будет справедлив биноминальный закон распределения, а вероятность появления в n – разрядном устройстве ровно i ошибок имеет вид

$$P_i = C_n^i q^i \left(1 - q\right)^{n-i}.$$

Здесь $q = 1 - e^{-\sum_{j-1}^{\overline{N}} \lambda_j t}$ - вероятность появления хотя бы одной ошибки в отдельном разряде в течение одной операции; λ_j – интенсивность отказов j – го элемента; $\overline{N}$ – количество элементов в разряде.

Если принять вероятность появления ошибки в отдельном разряде контролируемой и контрольной частях одинаковой и равной q, то вероятность появления не менее i ошибок в контролируемом и контрольном устройствах

$$P = \sum_{t=1}^{n+n'} C_{n+n'}^{i} q^{i} (1-q)^{n+n'-i},$$

где n – количество контрольных разрядов.

Для оценки эффективности схемного контроля введен коэффициент качества устройства с контролем

$$k(t) = 1 - \sum_{t=2}^{n+n'} \overline{P}_i C_{n+n'}^{i} q^{i} (1-q)^{n+n'-t},$$

где $\overline{P}_i$ - вероятность не обнаружения i – кратной ошибки (если контроль обнаруживает i – кратную ошибку, то $\overline{P}_i = 0$).

При условии $n' << n, i << n$ и $m \leq 7$ можно показать, что вероятность не обнаружения i – кратной ошибки при контроле по модулю m равна

$$\overline{P}_i = 1/m + (m-1)/m(-1/(m-1))^{i}.$$

Схемный контроль не всегда обеспечивает необходимую степень проверки правильности функционирования машины. Поэтому наиболее рациональной организацией системы контроля является комбинированный метод контроля и удовлетворяющий всем требованиям, предъявляемым к системам контроля БЦВМ. Основной проблемой использования комбинированного метода контроля является определение оптимального соотношения программного и схемного методов контроля, которое зависит от условий работы и применения БЦВМ.

Глава 7. ОБЩИЕ СВЕДЕНИЯ О БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

7.1 Краткая характеристика бортовых вычислительных систем и их классификация

Условия ведения современного воздушного боя требуют широкого внедрения вычислительных средств в состав бортового радиоэлектронного комплекса. При этом существует потребность в решении все более и более сложных задач. Для удовлетворения этой потребности необходимо совершенствовать бортовые средства вычислительной техники. Целью такого совершенствования является повышение скорости вычислений, увеличение надежности и живучести и улучшение технико-экономических показателей. Прогресс в вычислительной технике основан на поиске новых архитектурных и структурных решений, на реализации новых принципов обработки информации и принципов построения вычислительных средств.

Длительный период развития ЭВМ (и БВМ в том числе) заключался в совершенствовании структуры, предложенной в конце 40-х годов Дж. фон Нейманом. Концептуальную основу автоматизации вычислений составляла модель человека-вычислителя. Повышение быстродействия и надежности ЭВМ обеспечивалось главным образом благодаря увеличению скорости и надежности работы элементов, из которых она собиралась. В настоящее время возможности модели одиночного вычислителя исчерпаны, а скорость работы элементов близка к теоретически достижимой для используемой технологии их производства.

На смену модели одиночного вычислителя пришла модель коллектива вычислителей, предложенная академиком Э.В. Евреиновым в начале 60-х годов. Данная модель не накладывает ограничений на скорость вычислений, на значение показателей надежности и живучести.

Основным принципом модели коллектива вычислителей являются параллельность выполнения операций, программируемость структуры и конструктивная однородность элементов. Аппаратурно-программные средства, удовлетворяющие данным принципам (пусть даже частично), называют вычислительными системами.

Появление бортовых цифровых вычислительных систем (БЦВС)

было обусловлено значительным несоответствием характеристик существующих бортовых ЦВМ и требований к ним со стороны комплексов бортового оборудования по производительности и точности. Кроме того, успехи современной интегральной технологии, обеспечивающие существенное снижение массы, объема и потребляемой мощности отдельных ЦВМ и наличие естественного параллелизма решаемых ими задач, облегчают размещение на борту нескольких взаимосвязанных устройств обработки и хранения информации.

Принципы построения БЦВС отражают опыт, накопленный при создании наземных вычислительных систем.

Наличие большого числа вычислительных структур привело к появлению различных вариантов их классификации. При выборе схемы классификации необходимо обращать внимание на следующие обязательные качества:

возможность классификации всех, как существующих, так и предвидимых, вычислительных структур;

дифференциацию существенно различных вычислительных структур;

однозначность классификации любой вычислительной структуры;

наглядность, простоту и практическую целесообразность классификационной схемы.

Практическое применение в настоящее время нашли следующие классификационные схемы:

расширенная схема классификации Флинна;

классификационная система Шора;

Эрлангенская схема классификации.

В таблице 7.1 приведены классификационные признаки перечисленных классификационных схем.

Существующие бортовые ЦВС наиболее полно описываются классификационной схемой Флинна, поэтому целесообразно рассмотреть ее подробно.

В соответствии с двумя первыми признаками расширенной схемы классификации Флинна по типам потоков команд и данных (одиночный поток команд (ОК), множественный поток команд (МК), одиночный поток данных (ОД), множественный поток данных (МД)) вводятся понятия четырех основных классов вычислительных систем:

ОКОД - системы с одиночным потоком команд и одиночным

потоком данных;

МКОД - системы с множественным потоком команд и одиночным потоком данных;

ОКМД - системы с одиночным потоком команд и множественным потоком данных;

МКМД - системы с множественными потоками команд и данных.

Под потоком команд понимается последовательный ряд команд, выполняемых системой, а под потоком данных - последовательный ряд данных, вызываемых потоком команд, включая промежуточные результаты. Множественность потоков команд или данных определяется как максимально возможное число операций (команд) или операндов (данных), находящихся в одинаковой стадии обработки.

Таблица 7.1

Классификационная схема	Признаки классификации
Флинна	Тип потока команд; тип потока данных; способ обработки данных; степень связности функциональных элементов; тип связи между элементами БЦВС.
Шора	Способ соединения составных частей вычислительной системы; способ обработки данных.
Эрлангенская	Число устройств управления; число устройств обработки; длина машинного слова.

Упрощенные структурные схемы вычислительных систем приведены на рисунках 7.1 (ОКОД), 7.2 (МКОД), 7.3 (ОКМД) и 7.4 (МКМД). На рисунках поток данных для обработки обозначен {X} , а поток данных после обработки (результаты) - {У}.

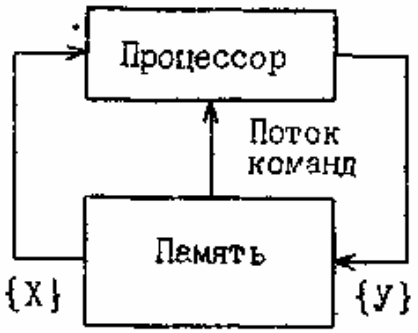


Рисунок 7.1

Рисунок 7.2

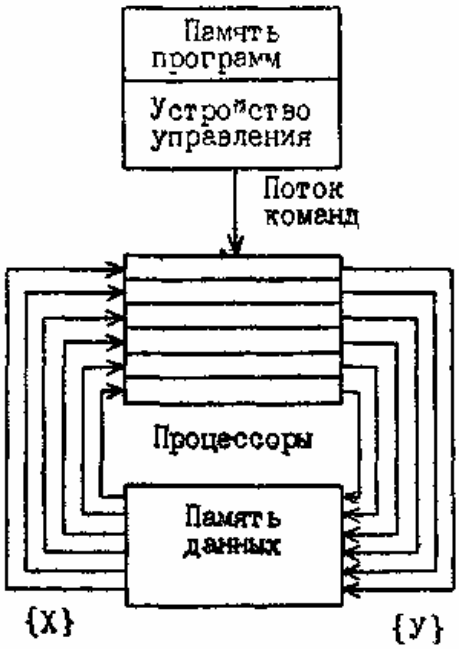


Рисунок 7.3

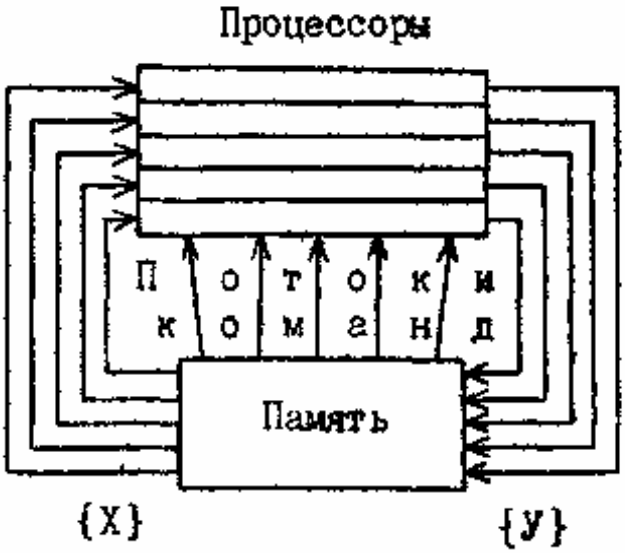


Рисунок 7.4

Основные типы наиболее часто используемых в настоящее время бортовых цифровых вычислительных систем в соответствии с классификационной схемой Флинна представлены на рисунке 7.5.

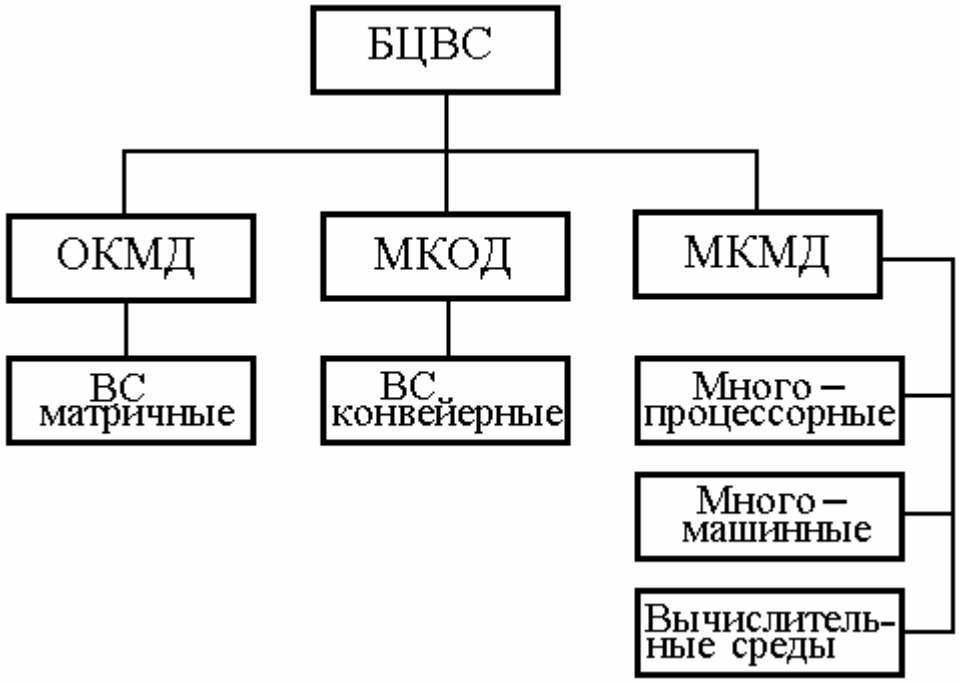


Рисунок 7.5 - Основные типы БЦВС

Матричные ВС основаны на том, что общее устройство управления определяет независимым однородным процессорным элементам одинаковые команды для каждого этапа вычислений программы (см. рисунок 7.3). Все процессоры одновременно выполняют одну и ту же операцию - каждый из них над своими данными, которые хранятся в локальных запоминающих устройствах процессоров. Данные должны быть однотипными (например, элементами матрицы).

Работа конвейерной вычислительной системы основана на выполнении двух или более команд одновременно, хотя и на разных стадиях (см. рисунок 7.2). Например, одна из команд может находиться в состоянии завершения, другая - в состоянии ожидания операнда из оперативной памяти. Общее устройство управления обслуживает несколько различных по назначению и структуре функциональных процессоров (ФП). Каждый ФП выполняет операцию над блоком данных, после чего передает его следующему процессору, а сам от предыдущего процессора принимает на обработку новый блок данных.

Многомашинная ЦВС представляет собой комплекс, объединяющий несколько вычислительных машин, каждая из которых включает в свой состав центральный процессор, память, каналы ввода-вывода. Отдельная часть такой ЦВС (одна из ЦВМ) может представлять собой также вычислительную систему.

Многопроцессорная ЦВС - это совокупность вычислительных средств, включающая два или более процессора, которые функционируют под управлением единой операционной системы на основе совместного использования общей памяти и каналов информационного обмена.

Однородные вычислительные среды (ОВС) являются одной из разновидностей многопроцессорных ЦВС и представляют собой двухмерную решетчатую структуру, в узлах которой располагаются функциональные устройства обработки информации (рисунок 7.6).

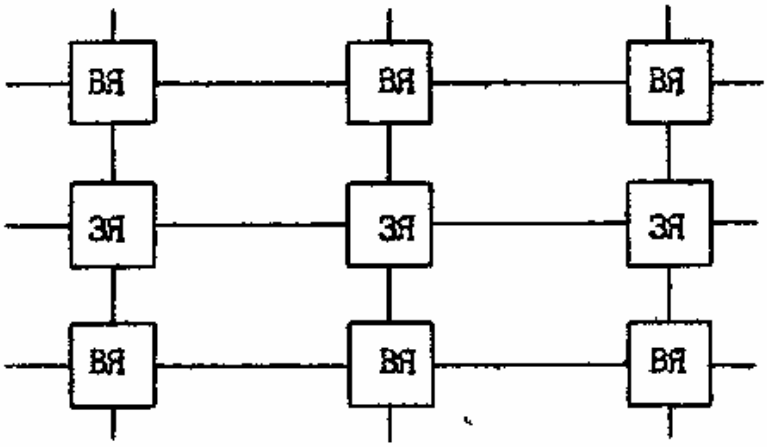


Рисунок 7.6 - Структура двухмерной решетки однородной вычислительной среды

Эти устройства представляют собой логические автоматы, называемые вычислительными ячейками (ВЯ). Вычислительные ячейки обеспечивают выполнение функционально полного набора элементарных операций. Организация функционирования ОВС строится на основе моделей коллектива вычислителей с использованием принципов параллельного выполнения произвольного числа операций. Структура вычислителя настраивается программно на конкретную решаемую задачу. При этом определяется для каждой вычислительной ячейки выполняемая ей функция (функции) и связи с соседними ячейками.

Для согласования во времени параллельно протекающих вычислительных процессов в структуру ОВС могут быть введены элементарные запоминающие ячейки (ЗЯ), осуществляющие хранение данных определенное количество тактов. Внутренняя организация вычислительного и запоминающего элементов одинакова и содержит запоминающее (вычислительное) устройство, средства программной настройки выполняемых функций и логическое устройство транзитной передачи информации (рисунок 7.7).

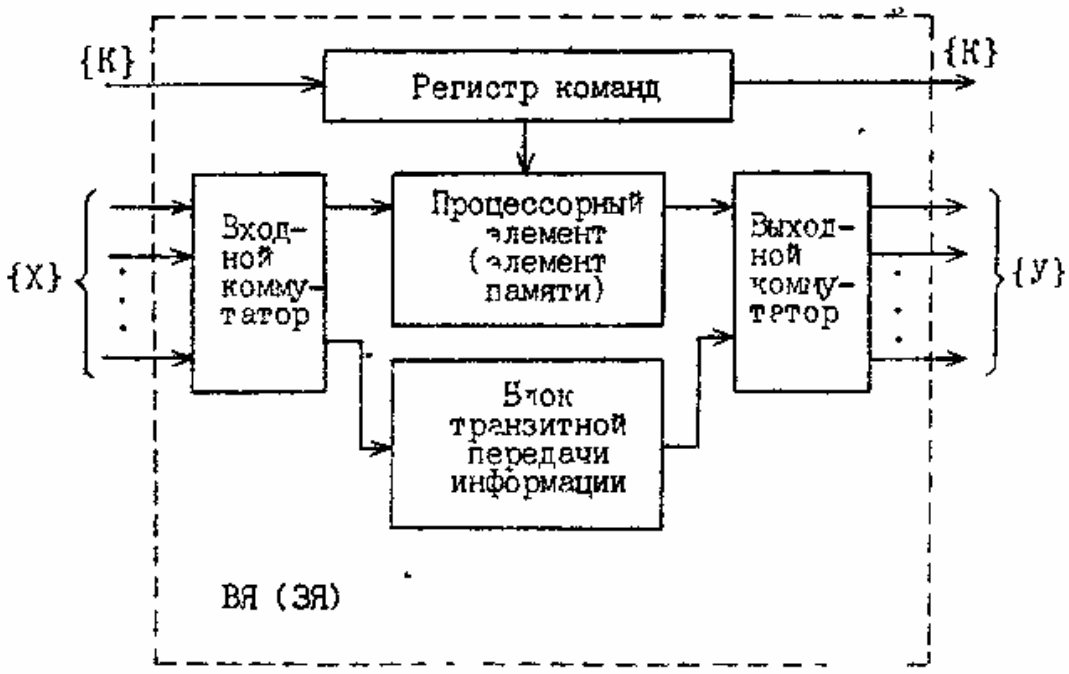


Рисунок 7.7 - Структурная схема вычислительной (запоминающей) ячейки

Бортовые вычислительные система современных истребителей являются многомашинными, причем некоторые вычислительные машины являются многопроцессорными (например, Ц-100 содержит основной процессор и процессор ввода-вывода).

Структура и характеристики многомашинных БЦВС определяются требованиями каждого конкретного применения и поэтому отличаются большим разнообразием вариантов.

Общие концепции построения многомашинных ЦВС бортового применения могут быть сформулированы следующим образом:

многоуровневая иерархическая организация структуры, соответствующая иерархическому характеру решаемых функциональных задач;

программируемость структуры с обеспечением возможности организации на верхних уровнях вычислительных сетей с динамическим перераспределением задач по ресурсам системы;

использование иерархических и программно-управляемых средств информационного обмена на основе унифицированных каналов;

модульность структуры и конструкции как элементов БЦВС, так и системы в целом за счет применения унифицированных конструктивно-функциональных модулей;

наличие развитых средств функционального контроля и диагностики.

Многоуровневая иерархическая организация многомашинной БЦВС определяется необходимостью разделения задач по уровням обработки информации, каждый из которых характеризуется различной степенью обобщения информации, используемой для решения задач данного уровня.

Например, задачи авиационного радиоэлектронного комплекса могут быть распределены по четырем иерархическим уровням:

задачи, решаемые на основе информации только одного датчика;

задачи обработки информации ряда (часто функционально близких) датчиков;

задачи общекомплексного характера - основные задачи комплекса, обобщающие в процессе решения информацию, по крайней мере, от двух различных функциональных подсистем;

задачи функционального контроля, индикации и управления комплексом, решаемые на основе обобщения всей поступающей информации.

Структурную схему реальной БЦВС рассмотрим на примере БЦВС самолета МиГ-31 (рисунок 7.8).

На самом высоком уровне расположена бортовая ЦВМ А-15А. Она служит для решения задач общекомплексного характера на основе информации, поступающей от всех сопрягаемых с комплексом систем. Эта же машина обеспечивает решение задач функционального контроля, индикации и управления работой радиоэлектронного комплекса, полетом самолета и средствами поражения. Связь БЦВМ с внешними элементами системы обеспечивается с помощью устройства ввода-вывода (УВВ). УВВ выполняет следующие функции:

выработку синхронизирующих сигналов для всех систем авиационного радиоэлектронного комплекса;

передачу данных между БЦВМ А-15А и системами авиационного комплекса с их преобразованием;

формирование требований на реализацию режимов обмена.

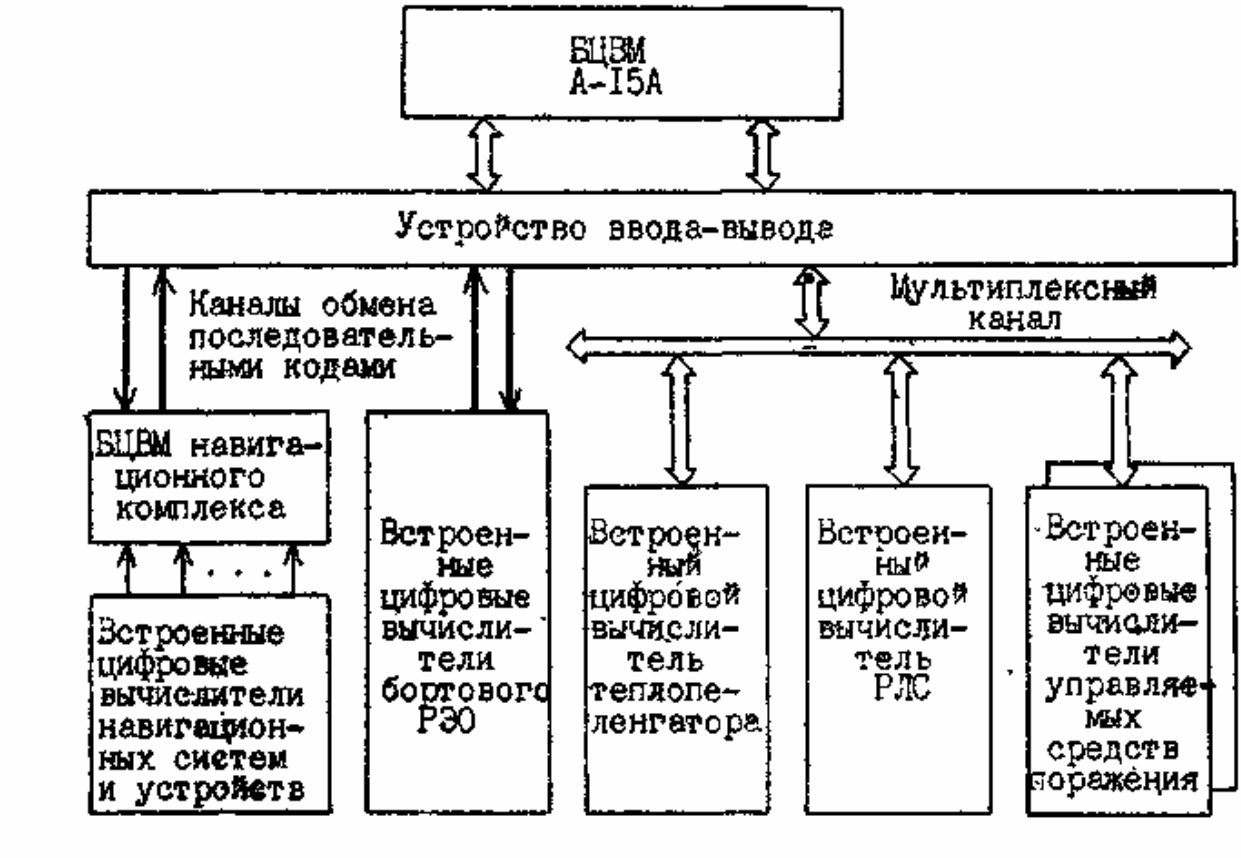


Рисунок 7.8 - Структурная схема БЦВС самолета МиГ-31

УВВ имеет каналы обмена последовательными и параллельными двоичными кодами, а также входы и выходы аналоговых сигналов в виде напряжения постоянного и переменного токов и импульсов.

БЦВМ навигационного комплекса обеспечивает вторичную обработку информации, поступающей от элементов навигационного комплекса. На самом нижнем - первом уровне расположены встроенные вычислители различных подсистем.

7.2 Особенности функционирования БЦВС авиационных радиоэлектронных комплексов

7.2.1 Программное обеспечение БЦВС АРЭК

Под программным обеспечением БЦВС АРЭК следует понимать совокупность программ, обеспечивающих решение общекомплексных задач, возложенных на ЦВМ верхнего уровня.

Функциональное назначение программного обеспечения (ПО) вычислительной системы заключается в управлении самолетом, его

вооружением и бортовым оборудованием. Для программного обеспечения управляющих систем характерны следующие особенности:

выполнение вычислений в реальном масштабе времени;

большой объем программ и используемой числовой информации, одновременно задействованной в ЦВМ;

сильная взаимосвязанность программ по последовательности их включения и по используемым переменным величинам;

мультипрограммный характер выполнения программ в режиме разделения времени и наличие обмена информацией с внешними абонентами во время решения основных функциональных задач;

многорежимность функционирования программ с использованием одной и той же программы в различных сочетаниях с другими в соответствии с изменением режимов работы управляющей системы;

применение буферных накопителей для хранения входной и выходной информации, поступающей в накопители в случайные моменты времени при взаимодействии с несколькими абонентами;

большой удельный вес в программах операций логического преобразования, поиска и упорядочивания информации с преобладанием этих операций над операциями чисто вычислительного характера;

использование программно-логических способов защиты от сбоев и отказов аппаратуры и от ошибок в поступающей информации.

В состав ПО БЦВС входят программы, выполняющие задачи управления БРЛС, самолетом, вооружением, программы навигации, программы встроенного контроля и программы связи с другими самолетами при работе в группе. Эти программы обеспечивают решение основных функциональных задач и поэтому называется функциональными. Кроме них в состав ПО БЦВС входят также программы, организующие вычислительный процесс, называемые управляющими или диспетчерскими.

При разработке управляющих программ должны учитываться следующие особенности работы ЦВС:

ЦВС выполняет множество функций и обменивается информацией с большим количеством внешних абонентов;

данные от абонентов могут поступать в ЦВМ периодически, а также в случайные моменты времени вследствие асинхронности работы разных систем комплекса, поэтому требования на решение отдельных задач формируются в ЦВС периодически с определенной частотой или возникают асинхронно при поступлении данных от

внешних абонентов;

в ЦВС необходимо непрерывно учитывать все заявки, на решение отдельных задач и определять в каждой ситуации оптимальную последовательность их реализации;

длительность решения задач и темп выдачи информации должны соответствовать режимам работы и текущим состояниям управляемых объектов.

С учетом этих требований и особенностей системы прерывания БЦВМ А-15А в БЦВС самолета МиГ-31выбрана следующая организация вычислительного процесса (рисунок 7.9). В системе использованы внешние прерывания первого (ВП-1), второго (ВП-2) и третьего (ВП-3) классов.

Внешние прерывания ВП-0 используются для контроля ЦВМ и для обслуживания режима регистрации полетной информации. Задачи этого класса являются самыми приоритетными и могут прерывать любые задачи, у которых нет маски (запрета) на ВП-0. Функциональные задачи разделены на две группы:

задачи ТАБЛИЦЫ РЕЖИМОВ (ТР), жестко связанные с временной диаграммой, периодичные, следующие друг за другом в строгой последовательности;

задачи, решаемые по заявкам, как правило, менее связанные с временной диаграммой.

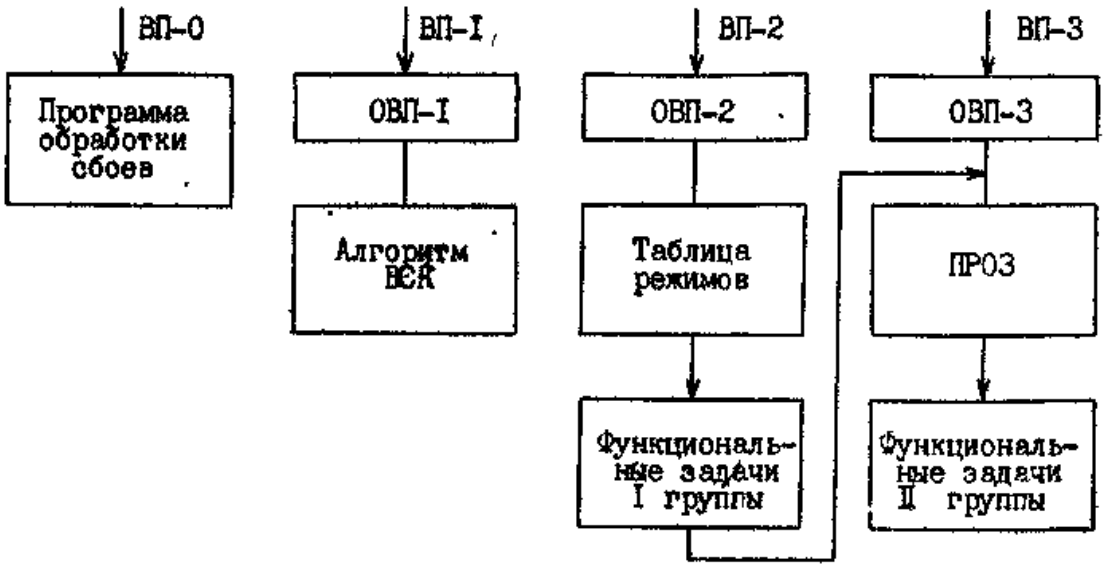


Рисунок 7.9 – Диспетчерские программы БЦВС

Время решения задач первой группы невелико, меньше длительности такта. Время решения задач, решаемых по заявкам, как правило, более одного такта.

Задачи, решаемые по заявкам, могут прерываться задачами ТР. Отсюда возникает необходимость запоминания состояния прерванных задач для последующего восстановления и продолжения решения. Задачи по заявкам ставятся на очередь и на исполнение в установленной при помощи приоритетного ряда последовательности программой обработки заявок (ПРОЗ).

Задачи ТР включаются по ВП-2, задачи встроенного контроля включаются по ВП-1. Данное разделение введено для большей помехозащищенности системы.

Для увязки работы подсистем, имеющих, как правило, собственные временные диаграммы, с работой системы в целом используются ВП-3. Они обслуживают группу срочных задач и могут прервать задачи по заявкам и задачи ТР.

Для каждого из прерываний есть соответствующая программа обработки. Программа ОБРАБОТКА ВП-1 (ОВП-1) организует включение программ встроенного контроля с частотой 96 Гц. Программа ОБРАБОТКА ВП-2 (ОВП-2) организует работу системы в реальном масштабе времени и включает программу ТАБЛИЦА РЕЖИМОВ также с частотой 96 Гц. Максимальная частота выполнения программ определяется периодом следования импульсов синхронизации радиолокационного комплекса равным 10,24 мс. Этот временной интервал называется также элементарным тактом работы комплекса. Программа ОБРАБОТКА ВП-3 (ОВП-3) организует выполнение отдельных задач в соответствии с требованиями на их включение, а также для контроля и отладки программ совместно с сервисным прибором ПС 8-01.

7.2.2 Выполнение программ, жестко связанных с временной диаграммой

Частные (функциональные) программы, жестко связанные с временной диаграммой, запускаются на решение программой Таблица режимов. Таблица режимов является частью общего диспетчера. Она обеспечивает последовательное подключение программ в соответствии с заданными временными диаграммами и логикой работы радиоэлектронного комплекса, осуществляет генерацию заявок на частотные и асинхронные задачи, а также организует все многообразие режимов работы комплекса.

Все частотные программы, входящие в ТР, подразделяются на несколько классов (уровней). Каждый уровень объединяет программы одинакового функционального назначения, но соответствующие разным режимам работы радиолокационного комплекса. Таким образом, любой конкретный режим предполагает включение на каждом уровне только одной программы. Выбор задач на каждом уровне осуществляется в соответствии с режимом работы комплекса и задается формированием ключей каждого уровня. Ключ - это ячейка оперативного запоминающего устройства, в которой находятся начальный адрес конкретной программы. Ключи формируются как в самой Таблице режимов, так и в других программах. Первоначально ключи формирует программа ПУСК. Такая структура обеспечивает простоту переключения режимов путем смены содержимого ключей. Другое достоинство такого построения ТР состоит в возможности подключения новых режимов путем наращивания программ в каждом уровне, не изменяя общей структуры программного обеспечения.

Для осуществления контроля за правильностью хода программы и восстановления программы в случае сбоя каждый уровень ТР заключен в так называемые контрольные ромбы. То есть в начале уровня запоминается слово состояния программы (ССП), одним из элементов которого является начальный адрес уровня. В конце уровня проверяется начальный адрес и, если проверяемый адрес не совпадает с записанным в ССП, значит, произошел сбой и необходимо восстановить правильность хода программы. Операторы записи и проверки ССП на схеме алгоритма изображены ромбами, поэтому они и носят названия контрольных ромбов.

Программы уровня Б (рисунок 7.10) предназначены для обработки информации от БРЛС в разных режимах. Конкретный режим для каждого интервала задает программа КОД ИНТЕРВАЛА, она же и назначает соответствующую ветвь уровня Б. Частота включения уровня Б, как и всей программы ТР, составляет 96 Гц. Но каждая конкретная ветвь этого уровня подключается со своей определенной частотой. Если в такте нет задач данного уровня, в ключ Б записывается адрес холостой ветви, которая передает управление на следующий уровень.

Второй уровень ТР называется "Дешифратор счетчика элементарных тактов" и обозначен на схеме (см. рисунок 7.10) как уровень С.

В общей ветви этого уровня с частотой 96 Гц работает программа выдачи информации в систему регистрации и на обзорный индикатор.

Уровень Т является определяющим для таблицы режимов, ему программа обязана и своим названием. На этом уровне собраны программы, называемые КОД ИНТЕРВАЛА, каждая из которых определяет содержание соответствующего интервала, а именно: назначает код интервала, рассчитывает и выдает в обменные ячейки необходимую в данном интервале информацию, назначает ветви обработки формированием ключей Б и Д. Каждому режиму работы комплекса соответствует своя определенная последовательность интервалов, которая задается таблицей начальных адресов программ КОД ИНТЕРВАЛА. Эти таблицы называются таблицами режимов. Отсюда и название всей программы.

Каждая таблица содержит 25 адресов, задающих последовательность программ КОД ИНТЕРВАЛА для полного цикла. Количество таблиц определяется количеством режимов работы комплекса и может наращиваться, если вводятся новые режимы.

С уровня Т осуществляется передача управления на уровень Д. На этом уровне собраны программы, функциональное назначение которых - расчет угловых позиций и выдача их в БРЛС. Уровень Д последний в таблице режимов, после его выполнения управление передается в программу обработки заявок.

7.2.3 Выполнение программ по асинхронным заявкам

Программное обеспечение БЦВС содержит ряд программ, время исполнения которых больше длительности такта с учетом необходимости решения в такте более приоритетных задач. Кроме этого, существует также наложение по времени исполнения двух или нескольких программ, обусловленное их приоритетностью.

Для обеспечения решения этих задач в соответствии с их приоритетностью и сохранения синхронности работы центральной ЦВМ и внешних устройств выполнение таких программ организовано по заявкам с разрешением прерывания заявок. Управляющая программа, обслуживающая данную группу функциональных программ, является составной, частью диспетчера и называется программой обработки заявок (ПРОЗ).

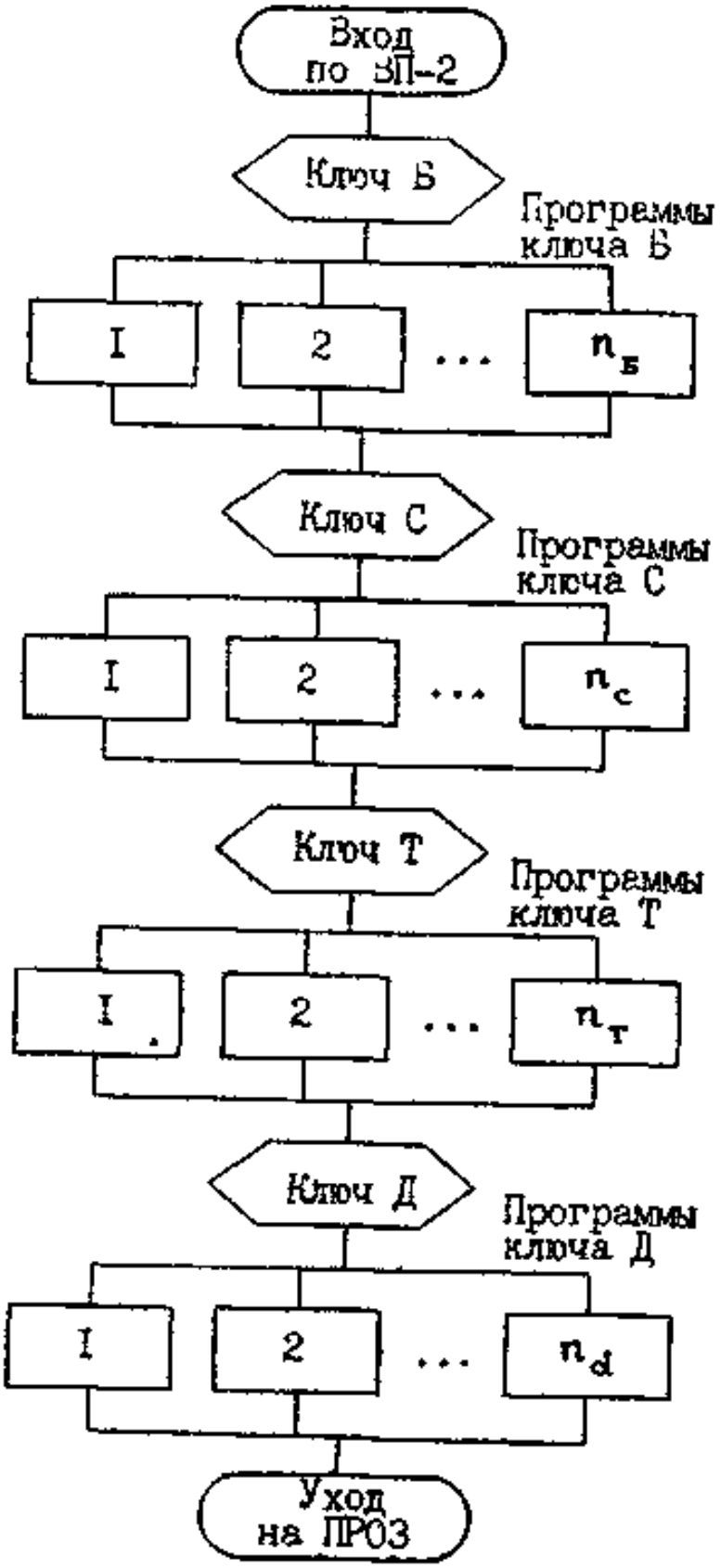


Рисунок 7.10 – Схема алгоритма программы ТР

Принцип работы ПРОЗ заключается в следующем. Все задачи, решаемые по заявкам, распределены по приоритету. Каждой частной

программе (ЧП) соответствует определенный разряд регистра заявок. Регистр заявок РЗ1*РЗ2 - это двойное слово ОЗУ. Приоритетность заявок уменьшается с увеличением номера разряда. Заявки первого слова приоритетней заявок второго слова. Ситуация, соответствующая заявке от частной программы, характеризуется наличием 1 в закрепленном за ней разряде регистра заявок. Кроме регистра заявок в ОЗУ организован регистр прерванных заявок РПЗ1*РПЗ2, а в долговременной памяти с возможностью смены информации (ДЗУС) — регистр неисполняемых заявок (РНЗ1*РНЗ2). Оба регистра имеют распределение разрядов, аналогичное РЗ1*РЗ2. Заявки на выполнение ЧП ставятся в программе ТР или при поступлении требования от внешних абонентов по ВП-3. ПРОЗ обеспечивает обслуживание в первую очередь заявки, имеющей наименьший номер из всех установленных в разрядах единиц. Поэтому возможно прерывание одной частной программы другой более приоритетной. В этом случае ПРОЗ подключает более приоритетную программу, запоминая все величины, необходимые для продолжения выполнения прерванной программы. Запоминание происходит в определенном участке ОЗУ, называемом карманом магазина. Всего имеется 5 карманов, в которых может храниться пять прерванных ЧП. По окончании исполнения текущей ЧП и отсутствия более приоритетных заявок из карманов выбираются по очереди прерванные программы и исполняются.

В каждом кармане магазина имеется 11 ячеек со следующим словами:

№ базы - в ней содержится собственный адрес данной ячейки для контроля настройки на карман магазина;

ССП1*ССП2$_{ТЕК}$ - слово состояния программы на момент прерывания;

ССП1*ССП2$_{БЛ}$ - ССП начального блока прерывающей программы;

РС1*РС2, РИ1*РИ2 - содержимое регистров сумматора и индексных регистров;

Счвр1*Счвр2 - счетчик зацикливания - момент времени, до которого задача должна быть решена. Он определяется при выборке новой задачи как текущий счетчик времени плюс время, отведенное для решения данной задачи. Время решения задачи складывается из времени на решение собственно данной задачи и времени решения более приоритетных задач, прерывающих ее.

Цена младшего разряда счетчика зацикливания равна одному такту (10,24 мс).

Для настройки на карман магазина служит ячейка РИМ (индексный регистр магазина), в которой всегда хранится адрес последнего занятого кармана магазина. Для случая, когда прерванных заявок нет, в РИМ посылается РИМ0 - адрес нулевого кармана магазина, состоящего из одной ячейки - базы кармана.

В целом временная диаграмма выполнения программ имеет вид представленный на рисунке 7.11.

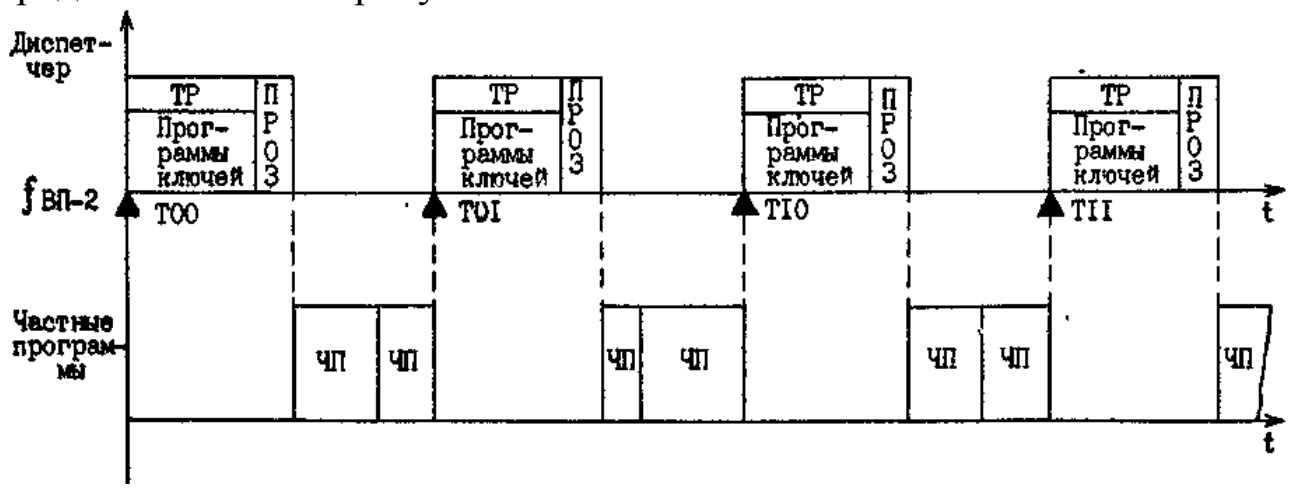


Рисунок 7.11 – Временная диаграмма выполнения программ

Программы таблицы режимов связаны с внешними прерываниями и выполняются с частотой 96 Гц за один цикл. Частные программы, исполнение которых требует большего времени, загружаются на решение программой ПРОЗ. Они могут быть прерваны как друг другом, так и программой ТР. Прерывание самой ПРОЗ невозможно. На рисунке 7.12 показана схема алгоритма ПРОЗ.

Вход в ПРОЗ осуществляется из программы ТР. После входа осуществляется маскирование всех прерываний, кроме прерываний по сбою оборудования, и из регистра заявок и регистра прерванных заявок извлекаются значения старших разрядов. По каждому из разрядов возможны четыре различные ситуации, определяемые сочетанием битов a и b.

При сочетании битов a=1 и b=0 - новая заявка сформирована, а прерванных заявок этого уровня нет, осуществляется выполнение новой частной программы. Выполнение начинается с выяснения, нет ли этой заявки в регистре неисполняемых заявок, для чего проверяется РНЗ в ДЗУС. Если заявка есть в РНЗ, то ее необходимо сбросить не выполняя, и перейти на выборку новой заявки, т.е. на начало ПРОЗ.

Если же состояние i-го разряда РНЗ равно нулю, то из ПЗУ загружается слово состояния программы, соответствующее заявке, и осуществляется её исполнение. В процессе исполнения возможно прерывание частной программы программами более высокого приоритета или управляющей программой (ТР). В этом случае в регистр прерванных заявок в i-й разряд записывается единица и происходит выход из ПРОЗ в таблицу режимов. При нормальном завершении частной программы ПРОЗ обнуляет $РЗ_i$, обеспечивает увеличение i на единицу и переходит к заявкам следующего (меньшего) приоритета.

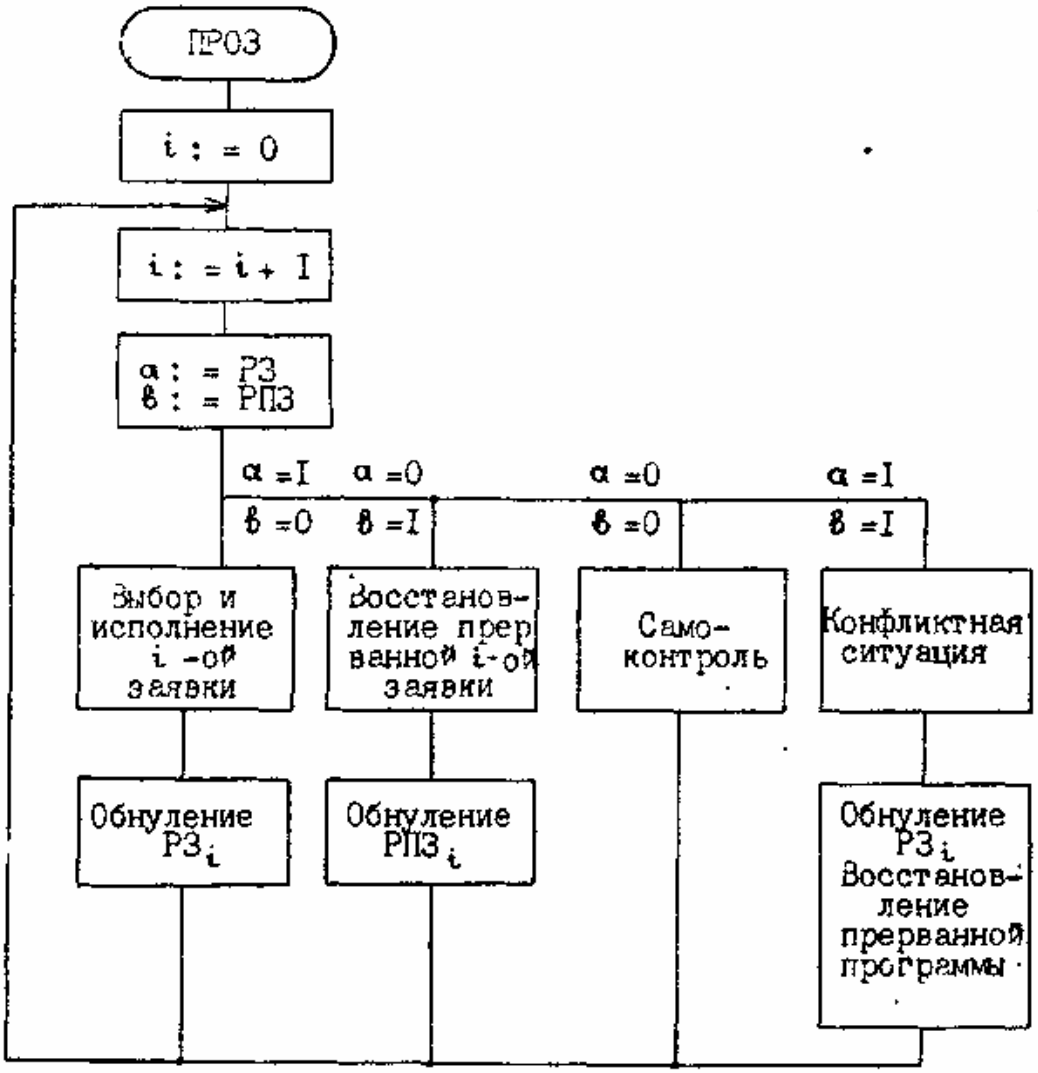


Рисунок 7.12 - Схема алгоритма программы обработки заявок

При сочетании битов a=0 и b=1 - новая заявка не сгенерирована, а старая прервана, происходит восстановление слова состояния прерванное программы по содержимому соответствующего кармана магазина и ее выполнение.

При сочетании битов a=0 и b=0 - новая заявка не сгенерирована, а старая выполнена, ПРОЗ переходит к проверке следующей пары битов РЗi+1 и РПЗi+1, соответствующих заявкам более низкого приоритета. После выполнения всех заявок происходит запуск тестов текущей самопроверки.

При сочетании битов a=1 и b=1- новая заявка сгенерирована, а старая не выполнена, возникает конфликтная ситуация. В этом случае новая заявка аннулируется и продолжается выполнение старой заявки - восстановление ССП. Таким образом, частота счета данной заявки понижается. Если время выполнения программы превысило заданное время, то заявка сбрасывается, этот сброс фиксируется в счетчике отказов. Если происходит аппаратный сбой, он также фиксируется в счетчике отказов. В этом случае регистры заявок и прерванных заявок обнуляются, остается только заявка на контроль, а в регистр индекса магазина (РИМ) посылается адрес нулевого кармана магазина.

7.3 Надежность бортовых вычислительных систем

БЦВС как объект исследования при количественном описании надежности представляет собой совокупность информационно-взаимосвязанных аппаратных и программных средств, функционирующих под управлением единой операционной системы. Отличие БЦВС от других объектов определяет необходимость учета следующих особенностей при рассмотрении вопроса ее надежности:

реальный масштаб времени реализации заданного класса алгоритмов;

многокомпонентность, определяемая большим количеством функциональных элементов БЦВС;

многорежимность, определяемая необходимостью решения большого числа задач;

функционирование в сложном информационно-управляющем поле, характеризующимся большим количеством источников и потребителей информации;

периодичность решения относительно устойчивого класса задач;

широкий диапазон внешних воздействующих факторов,

наличие различных видов избыточности (временной, структурной, информационной, функциональной, алгоритмической);

развитая система технического обслуживания и ремонта, существенно влияющая на уровень эксплуатационной надежности БЦВС;

широкое разнообразие используемых вычислительных средств с различными уровнями надежности и видами отказов;

зависимость общих критериальных показателей надежности БЦВС от состава и структуры аппаратных и программных средств.

Создание высоконадежной вычислительной системы в этих условиях связано с определением вероятности корректной реализации системой заданного класса алгоритмов в реальном масштабе времени. Эта вероятность определяется завершенностью, безошибочностью, точностью и своевременностью решения поставленных задач. При этом следует учитывать не только обеспечение требуемого уровня надежности БЦВС по отношению к устойчивым отказам, но и обеспечение надежности системы по отношению к неустойчивым, самоустраняющимся отказам - сбоям.

Если задать величину временного порога τ_{OT}, то определения для сбоя и отказа могут быть сформулированы следующим образом.

Сбой - событие, вызывающее временное нарушение реализации системой заданного класса алгоритмов, длительность которого не превышает временного порога τ_{OT}, а восстановление работоспособного технического состояния не требует реконфигурации структуры БЦВС.

Отказ - событие, приводящее к нарушению реализации заданного класса алгоритмов, длительность которого превышает временной порог τ_{OT}, а для восстановления работоспособного технического состояния необходима реконфигурация структуры БЦВС.

Результатом отказов в БЦВС являются систематические ошибки, приводящие к необходимости восстановления аппаратуры, а результатом сбоев - случайные ошибки, вызывающие необходимость восстановления информации, искаженной ошибкой.

На начальных этапах использования БЦВМ основное внимание уделялось вопросам обеспечения надежности аппаратных средств. По мере появления вычислительных систем чрезвычайно острой стала проблема повышения надежности программного обеспечения, поскольку БЦВС, надежная для одного класса решаемых задач и одних условий эксплуатации, может оказаться недостаточно надежной для другого класса задач и других условий эксплуатации.

Вероятность устойчивости БЦВС к сбоям $P_У^С$ определяется как произведение вероятностей безошибочной работы $P_В$ и своевременной реализации системой $P_С$ заданного класса алгоритмов, при условии соблюдения точностных характеристик алгоритмов (максимальных абсолютных и относительных погрешностей результирующих и промежуточных величин, количества достоверных двоичных разрядов, необходимых для представления промежуточных и результирующих величин, распределения операндных величин по количеству достоверных разрядов и т.п.):

$$P_У^С = P_В \cdot P_С .$$

Вероятность устойчивости БЦВС к отказам $P_У^О$ определяется как произведение вероятности пребывания БЦВС в работоспособном техническом состоянии $P_{РС}$ в начальный момент времени и вероятности проведения реконфигурации структуры системы $P_Р$ при возникновении отказа в пределах заданного времени:

$$P_У^О = P_{РС} \cdot P_Р .$$

В теории надежности разработано значительное количество математических моделей, использующихся для анализа уровня надежности БЦВС. К числу наиболее распространенных законов распределения, необходимых для анализа основных характеристик дискретных и непрерывных случайных величин, характеризующих сбои и отказы БЦВС, относятся следующие: ступенчатое распределение; экспоненциальное распределение; распределение Вейбулла; гамма-распределение; нормальное распределение; логарифмически нормальное распределение.

Выбор закона распределения осуществляется или по статистическим данным, полученным в процессе испытаний, или на основе анализа физических процессов, вызывающих отказы. При построении математических моделей оценки уровня надежности БЦВС целесообразно учитывать два обстоятельства: во-первых, выбранная модель не должна противоречить экспериментальным данным; во-вторых, должно быть обеспечено простое формальное описание модели и удобство ее использования при количественной оценке уровня надежности БЦВС. В соответствии с выбранным законом распределения осуществляется расчет временных и вероятностных характеристик по известным формулам.

7.4 Повышение живучести системы за счет автоматической реконфигурации

Бортовые вычислительные системы, работающие в реальном времени в составе АРЭК, должны обеспечивать непрерывное длительное функционирование, не допуская возникновения аварийных ситуаций или отказов РЭО при сбоях или отказах в оборудовании вычислительной системы. Обычно функции (программы), реализуемые БЦВС, неравнозначны, и временная утрата некоторых второстепенных функций при сохранении, основных жизненно важных допустима. Поэтому, для БЦВС основной характеристикой надежности следует считать не среднее время наработки на отказ, используемое для оценки надежности БЦВМ, а вероятность утраты на заданном интервале времени любой из основных функций вследствие отказов отдельных, элементов системы. Эта вероятность может быть принята за меру живучести системы.

Повешение живучести системы обеспечивается введением избыточного оборудования и специальной логической организацией, посредством которой достигается автоматическая реконфигурация системы для сохранения при возникновении отказов в оборудовании жизненно важных функций с, быть может, утратой второстепенных.

Основными принципами построения систем повышенной живучести на основе автоматической реконфигурации являются: многоустройственность (в том числе многопроцессорность); общие поля процессоров, оперативной памяти, каналов и периферийных устройств; динамическое распределение функций между однотипными устройствами.

Многоустройственность предполагает, что система должна содержать несколько экземпляров однотипных устройств (процессоров, модулей оперативной памяти (ОП), каналов и др.). При этом должна быть обеспечена избыточность устройств всех типов по отношению к минимальному набору, необходимому для выполнения жизненно важных функций (рисунок 7.13).

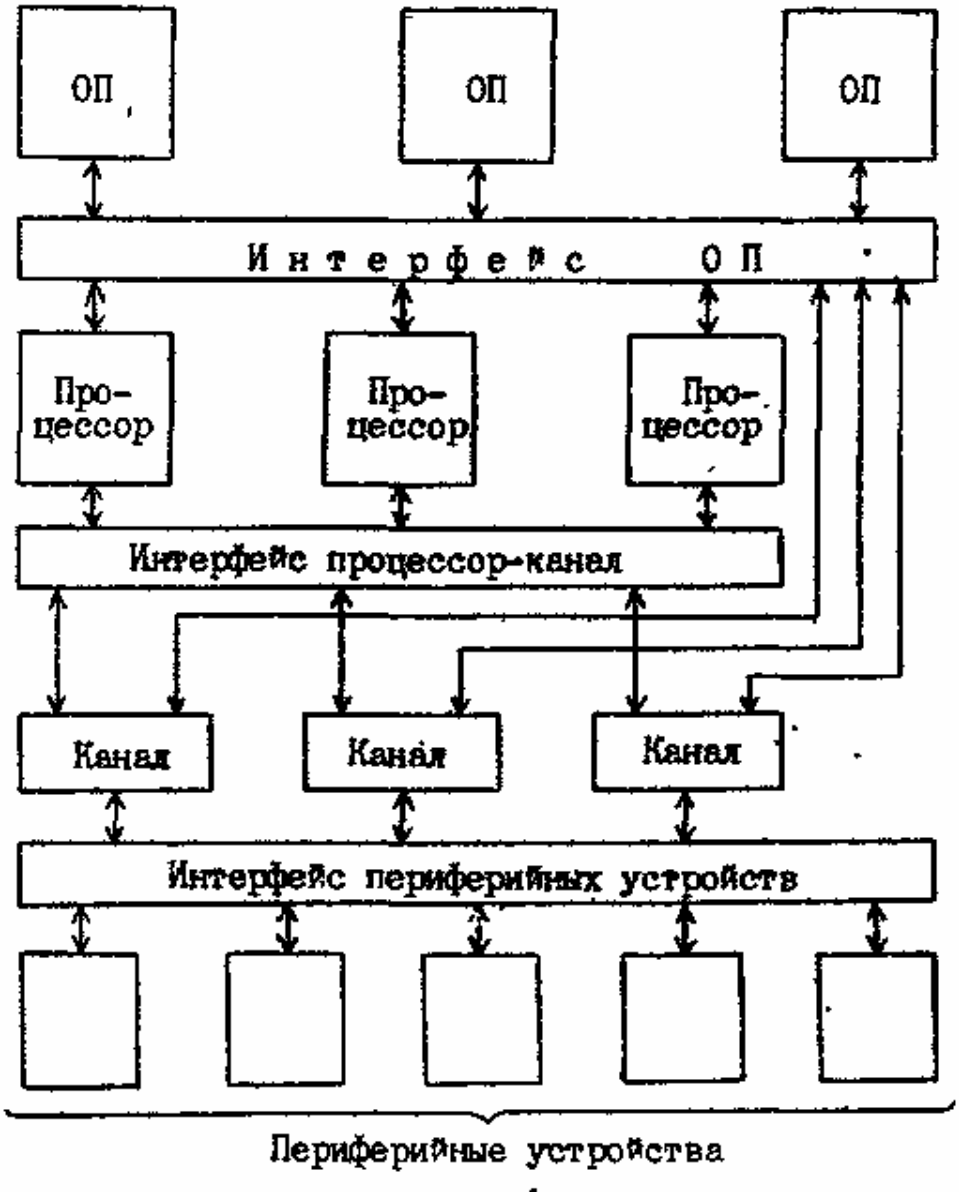


Рисунок 7.13 - БЦВС с общими полями памяти, процессоров, каналов, периферийных устройств

Под общим полем понимается равнодоступность устройств. Что в свою очередь предполагает, что все модули ОП доступны всем процессорам и каналам; все каналы идентичны и доступны любому процессору системы; все периферийные устройства доступны любому каналу.

Динамическое распределение функций означает, что программы не привязаны жестко к процессорам и могут выполняться на любом процессоре комплекса, а операции ввода-вывода не привязаны жестко к каналам и могут производиться любым каналом.

В случае динамического распределения функций заранее не известно, какое из однотипных устройств будет выполнять данную

функцию. Более того, работа может быть начата на одном, продолжена на другом и закончена на третьем устройстве.

Наличие общих полей и динамического распределения функций между однотипным оборудованием позволяет системе в принципе сохранять работоспособность, пока имеется хотя бы по одному исправному устройству каждого типа.

Для практической реализация вычислительных систем необходимо обеспечение следующих логических свойств: многосвязный интерфейс, неявную адресацию и динамическое распределение (диспетчирование) программ. Эти свойства в совокупности обеспечивают автоматическое выполнение реконфигурации за минимальное время, сохранение наиболее важных функций, автоматическое восстановление программ, прерванных сбоями и отказами.

Многосвязные интерфейсы (рисунок 7.14) позволяют устройствам связываться друг с другом по независимым шинам. В отличие от характерных для БЦВМ односвязных интерфейсов, в которых общие шины разделяются устройствами во времени, в многосвязном интерфейсе исключается выход из строя межмашинного интерфейса при неисправности одной из шин. Это требует дополнительного оборудования. Целесообразно строить интерфейс таким образом, чтобы каждое устройство имело одну выходную и несколько независимых входных систем шин.

Неявная адресация. Однотипные устройства, обслуживающие операцию ввода-вывода (каналы, блоки управления периферийными устройствами), имеют одинаковые логические номера и поэтому фактически командой ввода-вывода не адресуются. Запрос на выполнение операции ввода-вывода воспринимается первым по пути прохождения запроса исправным и свободным устройством данного типа. Неявная адресация действует не только при операции начальной выборки, но и при селекторном и мультиплексном обслуживании периферийного устройства. В разных сеансах мультиплексного обслуживания одной и той же операцией ввода-вывода могут использоваться разные каналы и блоки управления. Неявная адресация и многосвязность интерфейсов обеспечивают автоматический выбор свободного и исправного пути от ОП к периферийному устройству.

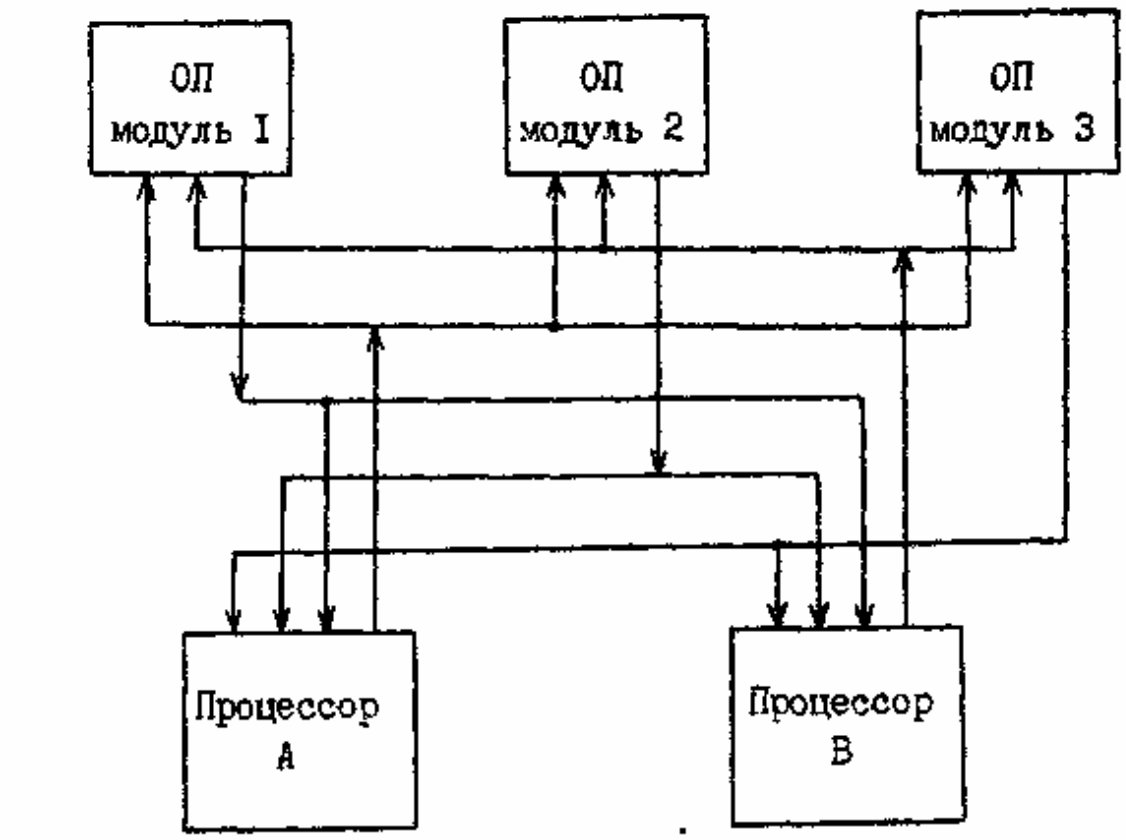


Рисунок 7.14 - Многосвязный интерфейс

Динамическое распределение (диспетчирование) программ между процессорами. В ОП организуются очереди для программ, имеющих различные уровни приоритета (рисунок 7.15). Очередь с более высоким номером соответствует более высокому приоритету. Программы заносятся в очереди своего приоритета прерываниями или идущими на процессорах программами. Аппаратура выделяет среди процессоров кандидата на прерывание, которым является процессор, обрабатывающий программу наименьшего приоритета. Аппаратура непрерывно сравнивает приоритет программы процессора-кандидата на прерывание с приоритетом непустой очереди с наибольшим номером. Если в очереди появляется программа большего приоритета, процессор-кандидат прерывает свою программу, заносит ее в очередь соответствующего приоритета и начинает выполнять программу из непустой очереди наибольшего приоритета. Обработка прерванной программы будет продолжена, причем не обязательно на том же процессоре, когда ее приоритет станет выше приоритета программы-кандидата на прерывание.

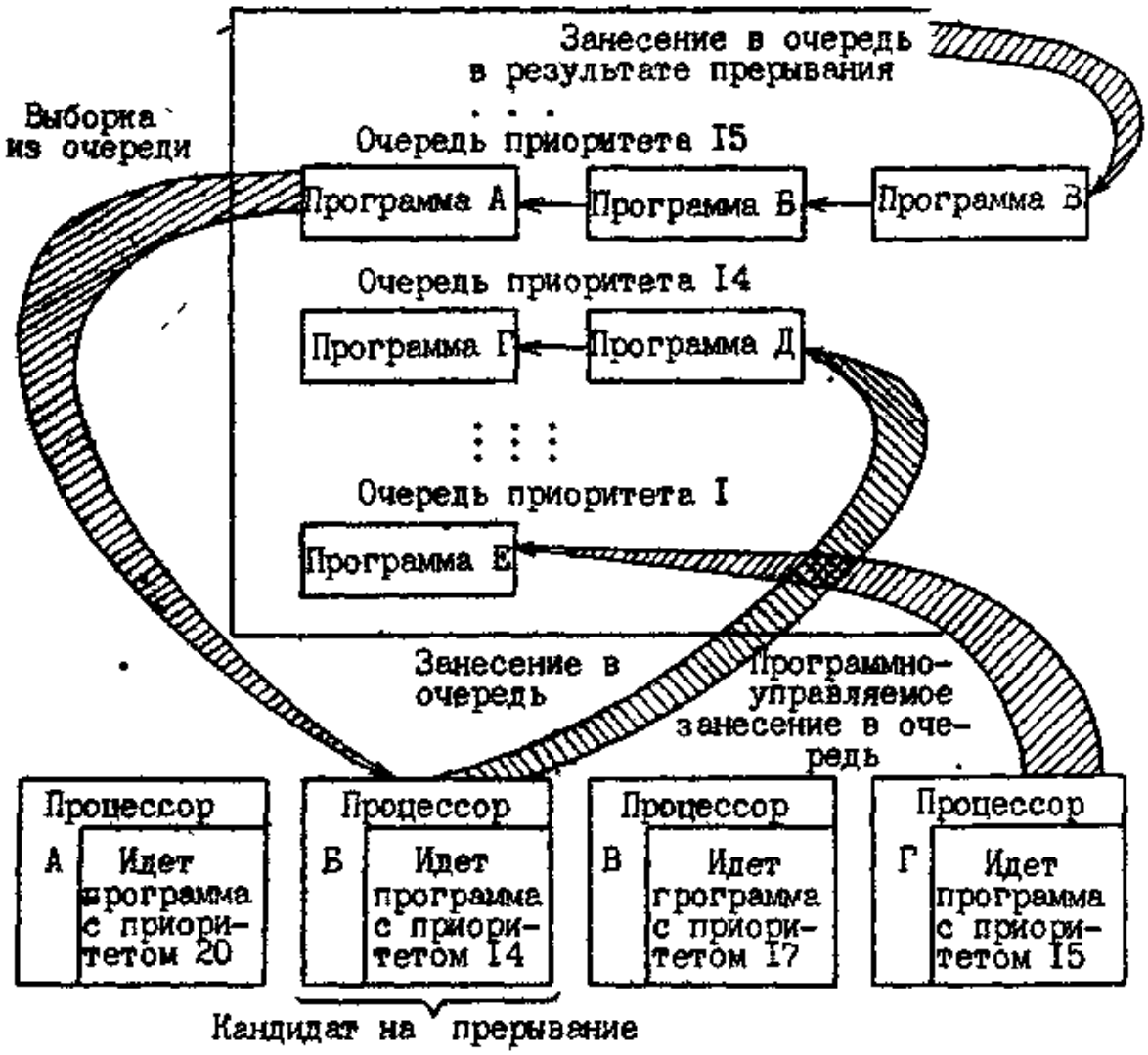


Рисунок 7.15 - Динамическое распределение программ

Описанный механизм позволяет достичь взаимной независимости программ и процессоров и обеспечивает в каждый момент времени выполнение наиболее приоритетных программ на наличных исправных процессорах.

8 ТЕХНИЧЕСКИЕ СРЕДСТВА ЦИФРОВОЙ ОБРАБОТКИ СИГНАЛОВ

8.1 Общие принципы технической реализации алгоритмов цифровой обработки

Цифровая обработка сигналов широко используется в различных системах авиационного радиоэлектронного оборудования (РЭО) военного назначения. Но наибольшее развитие цифровые методы обработки нашли в радиолокационных системах ввиду сложности и важности решаемых ими задач.

Радиолокационные системы в виде бортовых РЛС (БРЛС) и РГС входят в состав РЭО практически всех самолетов, вертолетов и ракет. Они представляют собой основное средство для получения информации о наземных, надводных, подводных и воздушных объектах (целях).

Для современных летательных аппаратов военного назначения характерно включение бортовых РЛС в состав обзорно-прицельного или прицельно-навигационного комплексов, применяемых на различных этапах выполнения боевой задачи.

Объединение в единый комплекс информационных подсистем и датчиков, функционально связанных через систему цифровой обработки (БЦВМ), существенно расширяет боевые возможности комплекса.

Обобщенная структурная схема бортовой радиолокационной системы показана на рисунке 8.1 /11/. В состав типовой радиолокационной системы входят: антенное устройство с элементами управления диаграммой направленности и антенным переключателем; синхронизатор; задающий генератор и передатчик; аналоговый приемник; аналогово-цифровой преобразователь (АЦП); БЦВМ, включающая процессор обработки сигналов (ПОС) и процессор обработки данных (ПОД); система индикации и управления РЛС, как правило, сопряжена с инерциальной системой (ИНС) и системой управления вооружением (СУВ).

Антенное устройство обеспечивает формирование основной диаграммы направленности требуемой формы и ее перемещение в пространстве по заданному закону, а также диаграммы направленности компенсационного канала. Синхронизатор определяет (задает) временной режим работы РЛС в целом и его отдельных составных частей (передатчика, приемника и др.). Антенный переключатель служит для подключения антенной системы к передатчику во время излучения зондирующего сигнала и к приемнику по окончании излучения.

Аналоговый приемник осуществляет предварительное усиление на сверхвысокой частоте (СВЧ), преобразование частоты до промежуточной, нормировку сигналов, синхронное детектирование сигналов, формирование и усиление синфазной и квадратурной составляющих. АЦП выполняет преобразование сигналов в цифровую форму. Задающий генератор формирует непрерывные высокочастотные сигналы стабильной частоты малой мощности для последующего их усиления и импульсной модуляции в передатчике, а также сигналы линейной частотной и фазовой модуляции. Передатчик обеспечивает формирование мощных радиолокационных зондирующих сигналов с ВЧП, СЧП и НЧП импульсов, а также генерирования непрерывных (прерывистых) высокочастотных сигналов подсвета РГС УР класса «воздух-воздух».

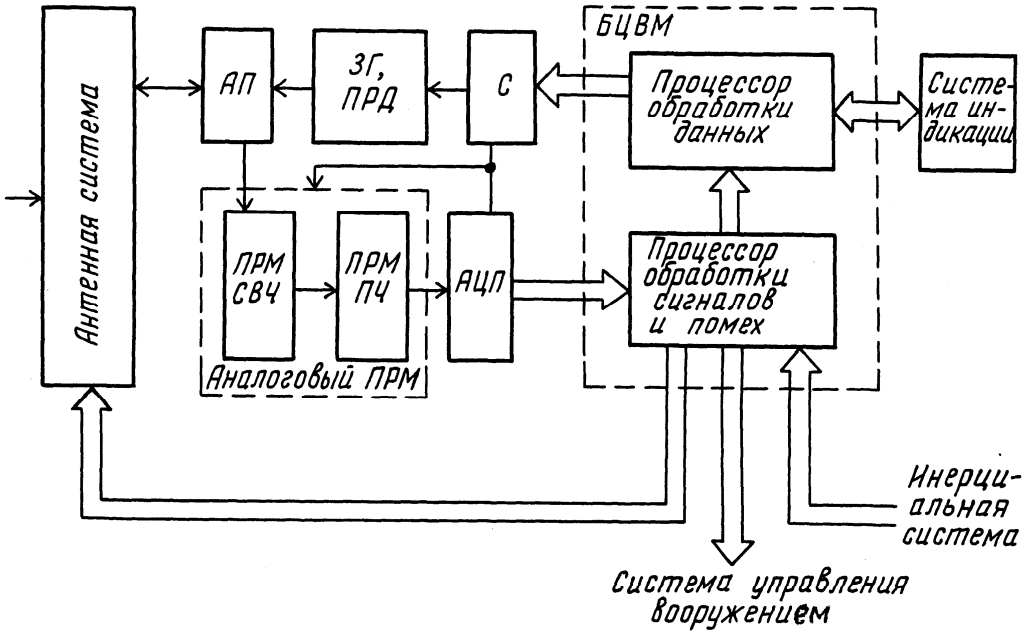


Рисунок 8.1

Процессор обработки радиолокационных сигналов (ПОС) обеспечивает решение задач первичной обработки сигналов, в частности, узкополосной доплеровской фильтрации и сжатия сигналов, пороговой их обработки и амплитудного детектирования, определения первичных отсчетов дальности до цели, скорости сближения с ней и др. Процессор обработки данных (ПОД) осуществляет управление режимами РЛС, формой и положением диаграммы направленности антенной системы, реализует алгоритмы оценивания (фильтрации) координат и параметров движения целей в различных режимах работы, формирует данные для системы индикации и т.д. Принципиальных различий в технической

реализации процессоров обработки данных и сигналов нет. Основное отличие заключается в их программном обеспечении.

При создании современной радиоэлектронной аппаратуры используются три основные подхода к технической реализации устройства цифровой обработки (УЦО): аппаратный, программный и программно-аппаратный. При аппаратном подходе получают УЦО с традиционной «жесткой» логикой, что обеспечивает наибольшее быстродействие устройств, но требует трудоемкой разработки индивидуальной структуры устройства (например, цифровой фильтр на дискретных элементах). При программном подходе УЦО реализуется в виде программы для универсальной цифровой ЭВМ. Программно-аппаратный подход предполагает разработку как программных, так и аппаратных средств и основан на применении встраиваемых микро-ЭВМ. Аппаратная часть такого УЦО содержит программируемое устройство обработки на основе микропроцессорной интегральной схемы, а также интегральные микросхемы, обеспечивающие организацию информационного обмена процессора и внешних устройств. Программная часть представлена реализованными на одном из языков программирования алгоритмами ЦОС. Этот вариант открывает широкие возможности для применения современных больших интегральных схем (БИС) микропроцессорных устройств и позволяет в наибольшей степени согласовать разрабатываемые программно-аппаратные средства с особенностями решаемых задач.

Современные микропроцессорные устройства развиваются по трем архитектурным направлениям: универсальные микропроцессоры, микроконтроллеры и цифровые сигнальные процессоры.

Микропроцессор (МП) представляет собой функционально законченное цифровое устройство, выполненное в виде одной или нескольких БИС и предназначенное для выполнения операций по обработке информации и управлению процессом обработки в соответствии с хранимой в памяти программой /12/. Универсальный микропроцессор, прежде всего, разрабатывается для выполнения разнообразных задач в составе цифровой ЭВМ или микропроцессорного вычислительного устройства (МПВУ). В узком смысле МП совпадает с центральным процессорным элементом (ЦПЭ) вычислительного устройства, выполненным на основе БИС. ЦПЭ обычно используется в качестве основного элемента микропроцессорного вычислительного устройства МПВУ, схема которого представлена на рисунке 8.2.

МПВУ минимальной конфигурации содержит ЦПЭ, блоки запоминающих устройств (ПЗУ и ОЗУ), генератор тактовых импульсов

(ГТИ) и блок интерфейса (ИФ), через который осуществляется связь с элементами радиоэлектронного комплекса, являющимися для ЦПЭ внешними устройствами (ВУ). Будем считать, что МПВУ, представляющее собой специализированное вычислительное устройство, используется в аппаратуре для выполнения некоторого заданного алгоритма обработки информации (или совокупности алгоритмов). Поэтому основная программа работы МПВУ записывается в ПЗУ, которое служит также для хранения различных подпрограмм, констант, таблиц и других данных, известных уже на этапе проектирования устройства. ОЗУ используется для хранения данных, поступивших из ВУ или подготовленных для выдачи в ВУ, а также промежуточных результатов вычислений и некоторой адресной информации. Блок ГТИ, выполняемый, как правило, на основе кварцевого генератора, предназначен для выработки серий тактовых импульсов и некоторых вспомогательных сигналов, необходимых для работы ЦПЭ и синхронизации других блоков системы.

Интерфейс представляет собой совокупность шин для передачи информации, электронных схем, специальных сигналов и алгоритмов, управляющих обменом информации. Блок интерфейса служит для сопряжения сигналов МПВУ и ВУ по временным и электрическим параметрам, а также в необходимых случаях для преобразования данных и управления обменом.

К основным узлам ЦПЭ относятся: управляющее устройство (УУ) с регистром команд (РК) и дешифратором команд (ДШК); арифметико-логическое устройство (АЛУ) с аккумулятором (А), который является основным рабочим регистром; блок регистров общего назначения (РОН) со счетчиком команд (СК).

Связь между блоками МПВУ осуществляется с помощью ряда шин: чаще всего это шины адреса (ША), шины данных (ШД), шины управления (ШУ). В некоторых случаях для увеличения быстродействия системы могут ввести отдельную шину команд (ШК).

Выбор типа МП зависит от общей элементной базы аппаратуры, в которую входит разрабатываемое МПВУ, требуемого быстродействия, допустимого объема оборудования, личного опыта разработчика и т. д. При этом следует иметь в виду, что процесс проектирования МПВУ, как правило, является итеративным. Это означает, что при выполнении каждого шага проектирования возможен возврат назад для корректировки принятых решений, причем может потребоваться смена типа МП или даже внесение изменений в исходный алгоритм работы устройства.

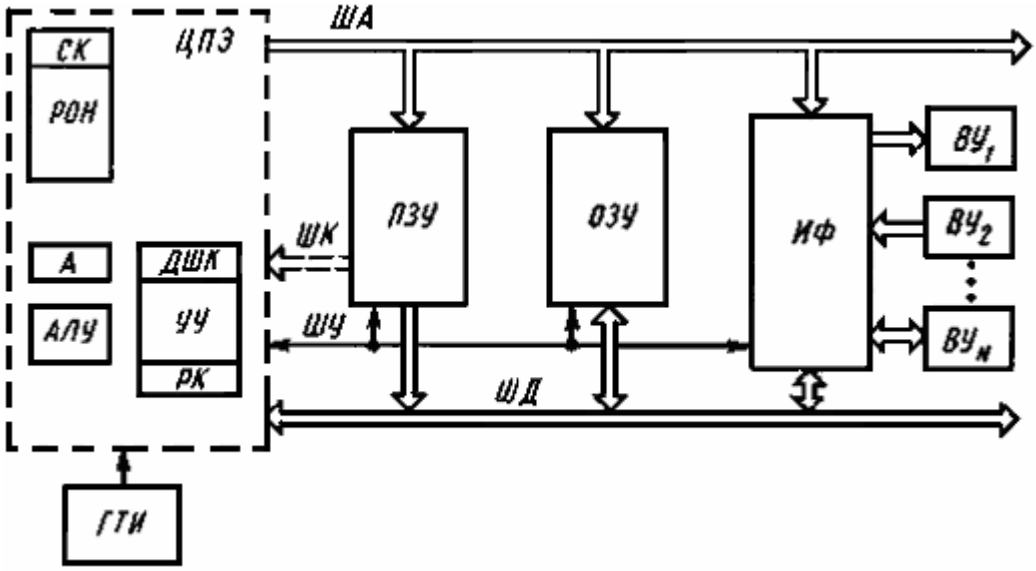


Рисунок 8.2 - Обобщенная структурная схема микропроцессорного вычислительного устройства

При использовании универсального МП структура МПВУ практически не зависит от конкретных задач, решаемых устройством. Алгоритмы цифровой обработки являются частью программного обеспечения МПВУ и представляют собой прямую реализацию рассмотренных в предыдущих главах математических выражений.

Недостатком использования универсальных МП для реализации МПВУ является функциональная избыточность оборудования, относительно большие массогабаритные характеристики и наличие большого числа элементов, что, в свою очередь, снижает надежность системы в целом.

Частично данные недостатки устраняются при использовании в качестве ЦПЭ микроконтроллеров и цифровых сигнальных процессоров.

8.2 Применение однокристальных микроконтроллеров для цифровой обработки сигналов

Современный уровень развития микроэлектроники позволил создать БИС, на кристалле которой размещены не только АЛУ, УУ и РОН, как в однокристальных МП, но и ОЗУ, ПЗУ, порты ввода-вывода и различные интерфейсные устройства вплоть до АЦП. По общей архитектуре такие БИС больше напоминают законченную ЭВМ. Однако незначительная емкость памяти, расположенная на кристалле, упрощенная и

ориентированная на выдачу сигналов управления система команд предопределяет использование их как управляющих устройств (контроллеров) для автоматизации различного радиоэлектронного оборудования. Принято называть их однокристальными микроконтроллерами (ОМК) /13, 14/. Применение ОМК позволяет существенно уменьшить число БИС, используемых в МПВУ, повысить надежность, снизить их стоимость и энергопотребление. Эти качества ОМК позволяют легко их встраивать в различные бортовые радиоэлектронные устройства.

При разработке ОМК в качестве основной задачи рассматривается задача управления некоторым объектом в реальном масштабе времени.

Отечественная микроэлектронная промышленность выпускает целый ряд ОМК, не только отличающихся разрядностью и производительностью, но и ориентированных на конкретные типы технологического оборудования. В таблице 8.1 приведены основные типы восьмиразрядных ОМК, выпускаемых отечественной промышленностью.

Таблица 8.1

Тип МК	ПЗУ команд (Кбайт)	EEPROM данных	ОЗУ данных	Тактовая частота (МГц)
КМ1816ВЕ48	1	-	64	6
КМ1816ВЕ49	2	-	128	11
КМ1816ВЕ51	4	-	128	12
КР1878ВЕ1	2	64	128	8

ОМК серии 1816 изготовлены по n-МОП-технологии и имеют похожую архитектуру. Они отличаются быстродействием, объемом ОЗУ, а также наличием или отсутствием на кристалле ПЗУ, ее типом и объемом. Наличие на кристалле портов ввода-вывода, тактового генератора, системы прерываний и таймера при высокой тактовой частоте и восьмиразрядном АЛУ обеспечивает универсальность их использования для создания микропроцессорных систем различного назначения.

При сравнительной оценке характеристик ОМК обращают внимание на наличие следующих устройств и их параметры:

- Flash ROM - объем энергонезависимой памяти программ (в килобайтах);
- EEPROM - объем энергонезависимой памяти данных (в байтах);
- RAM - объем статической памяти данных (в байтах);
- External RAM - возможность подключения к микроконтроллеру дополнительной микросхемы внешней статической памяти данных (в килобайтах);

- ISP - возможность программирования микроконтроллера в системе (на целевой плате) при основном напряжении питания;
- SPM - функция самопрограммирования Flash ROM памяти микроконтроллера в системе без участия внешнего программатора;
- JTAG - встроенный JTAG - интерфейс;
- I/O (pins) - максимальное количество доступных линий ввода / вывода;
- Timer(s) 8/16 bit - количество и разрядность таймеров/счетчиков;
- USI - универсальный коммуникационный интерфейс;
- AC - аналоговый компаратор;
- ADC (channels) - количество каналов аналого-цифрового преобразования;
- Internal RC - наличие внутренней RC-цепочки для автономной работы микроконтроллера (без внешнего источника опорной частоты);
- WDT - сторожевой таймер;
- BDC - аппаратный программируемый блок защиты от сбоев при внезапном (в том числе и кратковременном) пропадании напряжения питания микроконтроллера;
- UART - асинхронный последовательный приемопередатчик;
- SPI - синхронный трехпроводной последовательный интерфейс;
- I^2C - двухпроводной последовательный интерфейс;
- RTC - система реального времени;
- PWM (channels) - количество независимых каналов широтно- импульсной модуляции;
- Command Set - количество различных инструкций в системе команд микроконтроллера;
- Vcc - диапазон рабочих напряжений питания (в вольтах);
- Clock - диапазон рабочих частот (в мегагерцах);
- Packages - типы корпусов, в которые опрессовывается микроконтроллер, и общее количество выводов.

Наиболее производительным из ОМК 1816 является ОМК КМ1816ВЕ51, имеющий на кристалле значительную постоянную память объемом 4 Кб и возможностью расширения до 64 Кб. Его максимальная тактовая частота равна 12 МГц, а быстродействие составляет 1 млн. коротких операций в секунду. Существуют три модификации ОМК: ОМК 51 с масочным ПЗУ объемом 4 Кб рассчитан на крупносерийное оборудование; ОМК с ультрафиолетовым РПЗУ объемом 4 Кб ориентирован на системы, требующие периодической перенастройки; ОМК без встроенного ПЗУ .

ОМК 51 является развитием ОМК 48, но отличается системой команд. Однако на уровне программ на языке Ассемблера оба ОМК программно совместимы «снизу вверх».

Наиболее современной отечественной разработкой является микроконтроллерное RISC-ядро ТЕСЕЙ, производимое ОАО "АНГСТРЕМ", на базе которого уже создано несколько микроконтроллеров, как универсальных, так и специализированного применения.

Характерной особенностью ядра ТЕСЕЙ являются:

гарвардская RISC-архитектура, позволяющая выполнять любую из 52 команд 16-разрядного формата за два такта частоты процессора;

единая система команд для всего семейства с возможностью адресации до двух операндов, находящихся в памяти;

4-ступенчатый конвейер выполнения команд;

малое время отклика на прерывание и сохранение контекста;

широкий диапазон конфигураций внутренних памяти команд, памяти данных и периферийных устройств.

Микроконтроллеры отличаются наличием энергонезависимой памяти данных, возможностью многократного перепрограммирования памяти программ, небольшим количеством внешних выводов и низким энергопотреблением.

Микроконтроллер КР1878ВЕ1 является представителем семейства ТЕСЕЙ и обладает следующими характеристиками:

производительность до 4 MIPS на тактовой частоте 8 МГц;

перепрограммируемая память программ 1К х 16;

память данных RAM 128 х 8, EEPROM данных 64 х 8;

периферия: сторожевой таймер, 16-бит таймер-счётчик с предделителем, 12 линий ввода/вывода и поддержка прерываний;

ток потребления меньше 2 мА при напряжении питания 5В и тактовой частоте 5 МГц.

Микроконтроллер выпускается в 18-выводном исполнении.

Новые зарубежные разработки ОМК по своим функциональным возможностям и быстродействию ушли далеко вперед. Основные фирмы производители и номенклатура производимых микроконтроллеров представлены в таблице 8.2

Таблица 8.2 - **Фирмы-производители 8-, 16- и 32-разрядных микроконтроллерных семейств**

Фирма – производитель	Семейства МК					
	8-разрядные		16-разрядные		32-разрядные	
	Наименование	Число типов МК	Наименование	Число типов МК	Наименование	Число типов МК
Intel	MCS-48 MCS-51 MCS-151 MCS-251	10 60 2 4	MCS-96 MCS-196 MCS-296 i186xx	10 30 1 7	i960xx i386xx i486xx	12 6 6
Motorola	MC68HC05 MC68HC08 MC68HC11	200 22 43	MC68HC12 MC68HC16	2 10	MC68EC0x0 MC683xx MCF5xxx MPC500 MPC800	8 19 5 2 6
Microchip	PIC12Cxxx PIC16C5x PIC16Cxxx PIC17Cxxx PIC18Cxxx PIC14000	10 16 60 9 10 1				
Atmel	AT89 AT90S (AVR) AT94Kxx	11 28 3			AT91	10
Philips	8051	100	80C51XA	9		
Dallas Semiconductor	DS5000 DS8xCxxx	3 8				
Zilog	Z8/SAM8x Z8Plus	60 4				
Texas Instruments	TMS-370 TMS7000 TUSBxxxx	60 9 4	MSP430	20		
Hitachi	H8/300 H8/300L	17 47	H8/300H H8S/2xxx	20 45	SuperH	30
Mitsubishi	M740 M7450 M38000	600	MELPS7700 M16C	100 200	M32Rx/D M32R/E M32000	1 2 2
NEC	78K0/0S	200	78K3/4	80		
Panasonic	MN10100	35	MN10200	40		
Samsung	KS86/88	40				
Holtek	HT48/49xxx	13				

Продолжение таблицы 5.2

AMD			Am186xx	7	ElanSCxxx	5
Analog Devices	ADμC8xx	5				
Triscend	SoC-E5	5			SoC-A7	4
STMicroelectronics (бывшая SGS-Thomson)	ST6 ST7 ST9	25 50 10	ST10	10	ST100	
Fairchild Semiconductor	ACE	2				
Ubicom (Scenix)	SX	7				
Infineon (Siemens)	C500	17	C166	25	Tricore	3
Xemics	XE8000 CoolRISC	3				

Современные ОМК— это размещенная на одном кристалле сложная цифровая система, в состав которой входят 8-, 16- или 32-разрядный процессор, внутренняя память программ (до нескольких десятков килобайт) с возможностью расширения, перепрограммируемая электрическая энергонезависимая память (FLASH-память), широкий набор интерфейсных и периферийных устройств, порты ввода-вывода (параллельные и последовательные), таймеры, многоканальные аналого-цифровые преобразователи, модули ШИМ и др. Обладая малым количеством коротких команд, выполняемых за один цикл, и используя конвейерный способ их обработки, ОМК обеспечивают производительность в несколько миллионов операций в секунду.

5.2.1 RISC-архитектура современных микроконтроллеров

Одно из основных условий при выборе микроконтроллера для управленческих задач является его способность работать в реальном масштабе времени. Под реальным масштабом понимается такая скорость обработки информации, при которой не происходит существенного старения информации за время между моментом её прихода и получением результата. Для реализации реального масштаба времени требуется максимальное быстродействие процессорного ядра.

Достижение требуемого быстродействия требует новых архитектурных решений, одним из которых является RISC – архитектура, активно развиваемая фирмой Atmel. Это новое семейство высокопроизводительных 8-разрядных RISC (Reduced Instruction Set Computers) микроконтроллеров общего назначения, объединенных общей маркой ***AVR.*** Аббревиатура RISC означает усеченный перечень инструкций системы команд, причем в идеальном случае на выполнение каждой команды уходит время, не превышающее периода следования тактовых импульсов.

В основу концепции ***AVR***-микроконтроллеров легли следующие принципы:

1. Использование новейшей, наиболее скоростной и экономичной КМОП технологии фирмы Atmel в сочетании с RISC архитектурой для разработки и производства быстрых 8-разрядных микроконтроллеров, сравнимых с 16-разрядными микропроцессорами и микроконтроллерами по производительности и превосходящих микросхемы стандартной КМОП логики по скорости. Ожидаемая производительность - до 20 MIPS на частоте 20 МГц. Время выполнения короткой команды на такой тактовой частоте составляет 50 нс;

2. Разработка архитектуры и системы команд ***AVR*** в теснейшем согласии с принципами языка Си так, чтобы аппаратная часть нового микроконтроллера и его система команд были неотъемлемыми частями одного целого и использовались с максимальным коэффициентом полезного действия.

3. Функциональное расширение микроконтроллера возможностью программирования в системе путем объединения Flash-технологии фирмы Atmel со стандартным скоростным последовательным интерфейсом (SPI). Это позволяет многократно модифицировать программу не только с помощью обычного программатора, но и непосредственно в системе, в конечном устройстве пользователя. При этом не требуется вводить никаких дополнительных аппаратных узлов и вспомогательных источников питания извне как через SPI интерфейс, так и с помощью обычного программатора. Число циклов перезаписи - не менее 100000.

4. Расширенный состав периферийных устройств, таких как:

- таймер/счетчик, разрядность 8 бит;

- таймер/счетчик, разрядность 16 бит с возможностью организации функций широтноимпульсной модуляции (ШИМ) и захвата/сравнения;
- аналоговый компаратор;
- скоростной последовательный интерфейс SPI;
- встроенная система сброса микроконтроллера;
- асинхронный дуплексный последовательный порт UART;
- контроллер прерываний;
- внутренний тактовый генератор;
- сторожевой (WATCHDOG) таймер;
- многоканальный аналого-цифровой преобразователь.

Внутренний тактовый генератор может запускаться от внешнего источника опорной частоты, от внешнего кварцевого резонатора или от внутренней RC цепи. Поскольку все ***AVR*** полностью статические, минимальная допустимая частота ничем не ограничена (вплоть до пошагового режима). Максимальная рабочая частота определяется конкретным типом микроконтроллера. Ограничения верхней границы частотного диапазона связаны с технологическими проблемами при производстве микросхем и будут постепенно изменяться в сторону более высоких значений.

Сторожевой таймер предназначен для защиты микроконтроллера от сбоев в процессе работы. Он имеет свой собственный RC-генератор, работающий на частоте 1 МГц. Как и для основного внутреннего RC-генератора, значение 1 МГц является приближенным и зависит, прежде всего, от величины напряжения питания микроконтроллера и от температуры.

Порты ввода/вывода ***AVR*** имеют число независимых линий "Вход/Выход" от 5 до 32. Каждый разряд любого порта может быть запрограммирован на ввод или на вывод информации. Мощные выходные драйверы обеспечивают типовую токовую нагрузочную способность 20 мА на линию порта (втекающий ток) при максимальном значении 40 мА, что позволяет, например, непосредственно подключать к микроконтроллеру светодиоды и биполярные транзисторы. Общая токовая нагрузка на все линии одного порта не должна превышать 80 мА. Все значения приведены для напряжения питания 5 В.

Микроконтроллеры ***AVR*** работают в широком диапазоне питающих напряжений от 2,7 В до 6,0 В. Ток потребления в активном режиме зависит от величины напряжения питания и частоты, на которой работает микроконтроллер, и составляет менее 1 мА для 500 кГц, 5 ... 6 мА

для 5 МГц и 8 ... 9 мА для частоты 12 МГц. ***AVR*** также могут быть переведены программным путем в один из двух режимов пониженного энергопотребления. Первый - режим холостого хода (IDLE), когда прекращает работу только процессор и фиксируется содержимое памяти данных, а внутренний генератор синхросигналов, таймеры, система прерываний и сторожевой таймер продолжают функционировать. Ток потребления здесь не превышает 2,5 мА на частоте 12 МГц. Второй - режим микропотребления (SLEEP), когда сохраняется содержимое регистрового файла, но останавливается внутренний генератор синхросигналов. Выход из режима SLEEP возможен либо по сигналу сброса, либо от внешнего источника прерывания. При включенном сторожевом таймере ток потребления в этом режиме составляет около 80 мкА, а при выключенном - менее 1 мкА. (Все вышеприведенные значения справедливы для напряжения питания 5 В).

Температурные диапазоны работы микроконтроллеров ***AVR*** - коммерческий (0 ... 70 С) и индустриальный (-40 ... +85 С).

С точки зрения программиста ***AVR*** представляет собой 8-разрядный RISC микроконтроллер, имеющий быстрый гарвардский процессор, память программ, память данных, порты ввода/вывода и интерфейсные схемы. Структурная схема микроконтроллера приведена на рисунке 8.3.

Гарвардская архитектура ***AVR*** реализует полное логическое и физическое разделение не только адресных пространств, но и информационных шин для обращения к памяти программ и к памяти данных. Способы адресации и доступа к ним также различны. Такое построение уже ближе к структуре скоростных цифровых сигнальных процессоров и обеспечивает существенное повышение производительности за счет:

одновременной работы центрального процессора, как с памятью программ, так и с памятью данных;

расширения до 16 бит разрядной сетки шины данных памяти программ.

Следующим шагом на пути увеличения быстродействия ***AVR*** является использование технологии конвейеризации, вследствие чего цикл "выборка-исполнение" команды может быть заметно сокращен, повышая тем самым производительность процессора. Например, у микроконтроллеров семейства MCS51 короткая команда выполняется за 12 тактов генератора (1 машинный цикл), в течение которого процессор последовательно считывает код операции и исполняет ее.

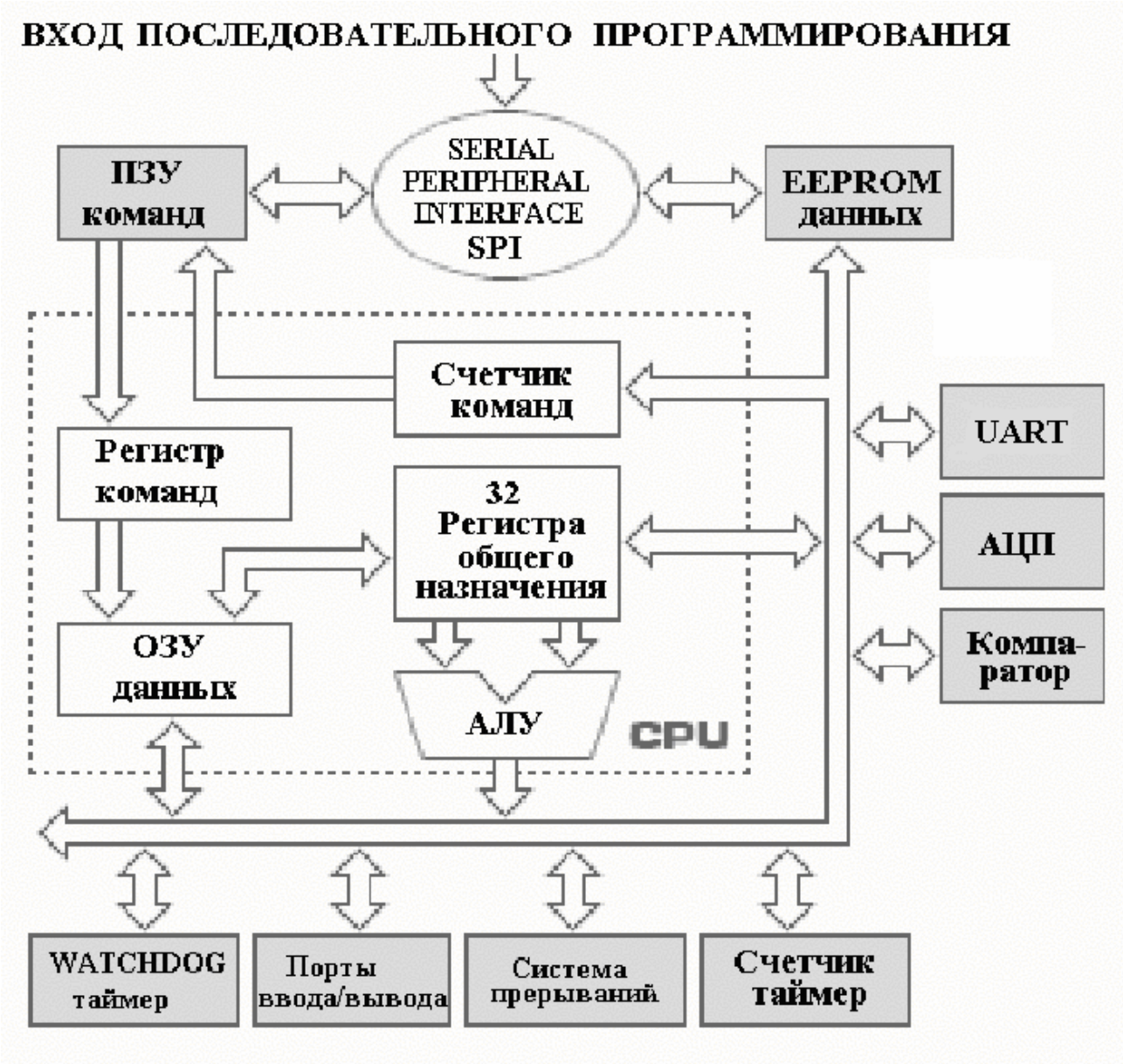


Рисунок 8.3 - Структурная схема AVR

В PIC-контроллерах фирмы Microchip уже реализована конвейерная обработка. Короткая команда выполняется у них в течение 8 периодов тактовой частоты (2 машинных цикла). За это время последовательно дешифрируется и считывается код операции, исполняется команда, фиксируется результат и одновременно считывается код следующей операции (конвейер). Поэтому одна короткая команда в общем потоке реализуется за 4 периода тактовой частоты или за один машинный цикл. В микроконтроллерах ***AVR*** тоже используется одноуровневый конвейер при обращении к памяти программ и короткая команда в общем потоке выполняется, как и в PIC-контроллерах, за один машинный цикл. Главное же отличие состоит в том, что этот цикл у ***AVR***

длится всего один период тактовой частоты по сравнению с четырьмя у PIC платформ.

Следующая отличительная черта архитектуры микроконтроллеров ***AVR*** - регистровый файл быстрого доступа (32 регистра общего назначения). Структурная схема регистрового файла показана на рисунке 8.4.

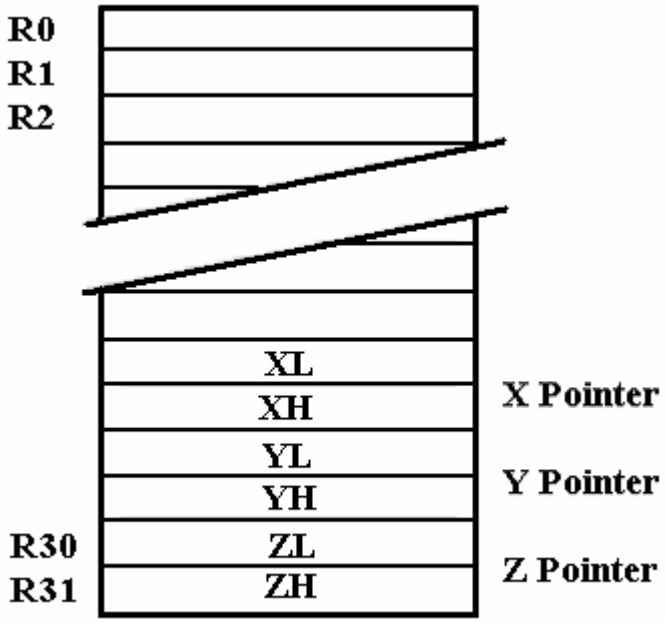


Рисунок 8.4 - Регистровый файл

Каждый из 32-х регистров общего назначения длиной 1 байт непосредственно соединен с арифметико-логическим устройством (ALU) процессора. Это означает, что в ***AVR*** существует 32 регистра-аккумулятора. Это позволяет в сочетании с конвейерной обработкой выполнять одну операцию в ALU за один машинный цикл. Например, два операнда извлекаются из регистрового файла, выполняется команда и результат записывается обратно в регистровый файл в течение только одного машинного цикла.

Шесть из 32-х регистров файла могут использоваться как три 16-разрядных указателя адреса при косвенной адресации данных. Один из этих указателей применяется также для доступа к таблицам перекодировок, записанных в памяти программ микроконтроллера. Использование трех 16-битных указателей (X, Y и Z Pointers) существенно повышает скорость пересылки данных при работе прикладной программы.

Регистровый файл занимает младшие 32 байта в общем адресном пространстве SRAM ***AVR***. Такое архитектурное решение позволяет получать доступ к быстрой "регистровой" оперативной памяти микроконтроллера двумя путями - непосредственной адресацией в коде команды

к любой ячейке и обычными (пятью) способами адресации ячеек SRAM. В технической документации фирмы Atmel это свойство носит название "быстрое контекстное переключение" и является еще одной отличительной особенностью архитектуры ***AVR***, повышающей эффективность работы микроконтроллера.

Во время переходов к выполнению процедур обработки прерываний или подпрограмм текущее состояние программного счетчика сохраняется в стеке. Обычно стек формируется программно и располагается в общем адресном пространстве оперативной памяти данных. Размер стека лимитируется только общим объемом SRAM и степенью ее использования прикладной программой. 16-разрядный указатель стека находится в общем адресном пространстве оперативной памяти и доступен для чтения и записи.

Система команд ***AVR*** весьма развита и насчитывает до 120 различных инструкций. Почти все команды имеют фиксированную длину в одно слово (16 бит), что позволяет в большинстве случаев объединять в одной команде и код операции, и операнд(ы). Различают пять групп команд ***AVR***: условного ветвления, безусловного ветвления, арифметические и логические операции, команды пересылки данных, команды работы с битами. По разнообразию и количеству реализованных инструкций ***AVR*** больше похожи на CISC (Complex Instruction Set Computer – компьютер со сложной системой команд), чем на RISC процессоры. Например, у PIC-контроллеров система команд насчитывает от 33 до 58 различных инструкций, а у MCS51 она составляет 111.

В целом архитектура ***AVR*** в сочетании с регистровым файлом и расширенной системой команд позволяет в короткие сроки создавать программы с очень эффективным кодом, как по скорости его выполнения, так и по компактности.

Еще одним достоинством семейства ***AVR*** является наличие средств поддержки разработок для ***AVR.*** Программные и аппаратные средства для новой платформы разрабатывались параллельно с самими микроконтроллерами и включают в себя компиляторы, внутрисхемные эмуляторы, отладчики, программаторы, простейшие отладочные платы-конструкторы практически на любой вкус. Более детальное знакомство с микроконтроллерами как отечественных, так и ведущих мировых производителей можно выполнить самостоятельно по имеющейся в большом количестве справочной литературе, например /13..15/.

8.3 Программируемые процессоры цифровой обработки сигналов

8.3.1 Основные операции обработки сигналов в цифровых системах и их влияние на структуру сигнального процессора

Среди микропроцессорных интегральных схем, используемых для задач цифровой обработки, наибольшее распространение нашли цифровые сигнальные процессоры (Digital Signal Processor - DSP) /13,15..18/.

Архитектура DSP определяется несколькими базовыми операциями, которые используются в алгоритмах ЦОС.

Для выделения таких операций проведем функциональный анализ основных направлений ЦОС, к которым относятся цифровая фильтрация и спектральный анализ.

Цифровая фильтрация. В области цифровой фильтрации разработчик систем ЦОС имеет дело с реализацией КИХ- и БИХ-фильтров (с конечной и бесконечной импульсными характеристиками соответственно). Оба класса фильтров относятся к классу линейных систем с постоянными параметрами, в которых входная и выходная последовательности связаны отношениями типа свертки.

В КИХ-фильтре отсчет выходного сигнала определяется только значениями входного сигнала, а в БИХ-фильтре - значениями входного и выходного сигналов. Это хорошо видно из линейных разностных уравнений с постоянными коэффициентами, которыми описывается данный класс дискретных систем.

Таким образом, для построения систем цифровой фильтрации требуется эффективная реализация соотношения типа дискретной свертки, которая раскладывается на операции умножения и накапливающего суммирования, а также операции задержки.

Спектральный анализ. В области спектрального (или гармонического анализа) используются прямое и обратное дискретное преобразование Фурье, а также рациональный способ реализации дискретного преобразования Фурье - быстрое преобразование Фурье.

Спектральный анализ основан на известных методах представления данной функции при помощи других функций, которые называются базовыми и свойства которых считаются известными.

Для получения выходной последовательности в естественном порядке необходимо выполнить двоичную инверсию отчетов входной

последовательности. Перестановка входных элементов состоит в образовании двоичных номеров выходной последовательности путем добавления единицы к старшему разряду с распространением переноса в сторону младших разрядов (вправо). Такая адресация получила название бит-реверсивной.

Вычисление коэффициента $W_N^k = \cos\left(\frac{2\pi k}{N}\right) - j\sin\left(\frac{2\pi k}{N}\right)$ можно осуществлять следующим образом:

- используя подпрограммы или таблицы синуса и косинуса;
- прямым табличным способом (выборкой готовых значений из таблицы);
- используя рекуррентную формулу

$W_N^k = \left(W_N^{k-l}\right)W_N^l$ при $W_N^0 = 1$;

- таблично-алгоритмическим способом, так как на последующих этапах коэффициенты повторяются.

При использовании алгоритма БПФ с прореживанием по частоте требуется перестановка элементов выходной последовательности, а базовая операция «бабочка» сводится к вычислению выражений:

$$x = A + BW_N^k,$$
$$y = A - BW_N^k.$$

Для получения амплитуд и фаз составляющих спектра (гармоник) необходимо также вычислить следующие выражения:

$$\frac{\sqrt{X_{\mathrm{Re}}^2 + X_{\mathrm{Im}}^2}}{N}; \quad arctg\frac{X_{\mathrm{Im}}}{X_{\mathrm{Re}}},$$

где $X_{\mathrm{Re}}, X_{\mathrm{Im}}$ - вещественная и мнимая части комплексных коэффициентов.

В гомоморфной обработке сигналов дополнительно требуется вычисление функций $\log_2 x$ и 2^x.

8.3.2 Обобщенная архитектура DSP

На рисунке 8.5 приведен операционный базис алгоритмов ЦОС. Поставив в соответствие каждой операции блок, можно получить обобщенную архитектуру DSP (рисунок 8.6). Очевидно, что реализация выделенных операций может быть различной.

Предельное быстродействие достигается при максимальной аппаратной поддержке операций, что приводит к значительным аппаратным затратам.

Это обстоятельство, а также желание обеспечить преемственность в решениях и программную совместимость приводят, в конечном счете, к аппаратной реализации не в целом той или иной операции, а отдельных ее составляющих. Причем в разных семействах (16-, 24-, 32-разрядных) архитектура отражает специфику задач ЦОС, на которые она ориентирована

Основными операциями являются: задержка, АЦП и ЦАП (функции кодека), умножение с накоплением - операция MAC (multiply and accumulate), умножение, сложение, вычисление значений функций sin, cos, arctg, $\log_2 x$, 2^x, табличное вычисление коэффициентов W_N^k. К особенностям следует также отнести необходимость использования бит-реверсивной адресации.

Обобщенная архитектура процессора DSP образуется тремя составляющими: процессорным ядром, подсистемами ввода/вывода и хранения. Структура процессорного ядра отражает выделенный операционный базис и учитывает особенности алгоритмов ЦОС. Блок генерации адреса обеспечивает необходимые способы адресации, в том число бит-реверсивную. Синтезатор тактовой частоты (PLL) дает возможность регулировать производительность и потребляемую мощность. Блок выполнения операций реализует операции с фиксированной точкой и операции с плавающей точкой для приложений, где требуются высокая точность и производительность. Очевидно, что поддержка операций с плавающей точкой может и отсутствовать (используется обычно в 32-разрядных DSP).

В процессорное ядро введен блок отладки и контроля, который содержит средства внутрисхемной эмуляции (OnCE), средства тестирования и отладки (JTAG), сторожевой таймер (WDT). Данный набор средств сегодня практически является стандартным.

Подсистема ввода/вывода включает кодек (АЦП и ЦАП), параллельные и последовательные порты ввода/вывода, host-интерфейс для связи с персональным компьютером или другой ЭВМ, аудио-интерфейс, широтно-импульсный модулятор, таймеры и другие устройства.

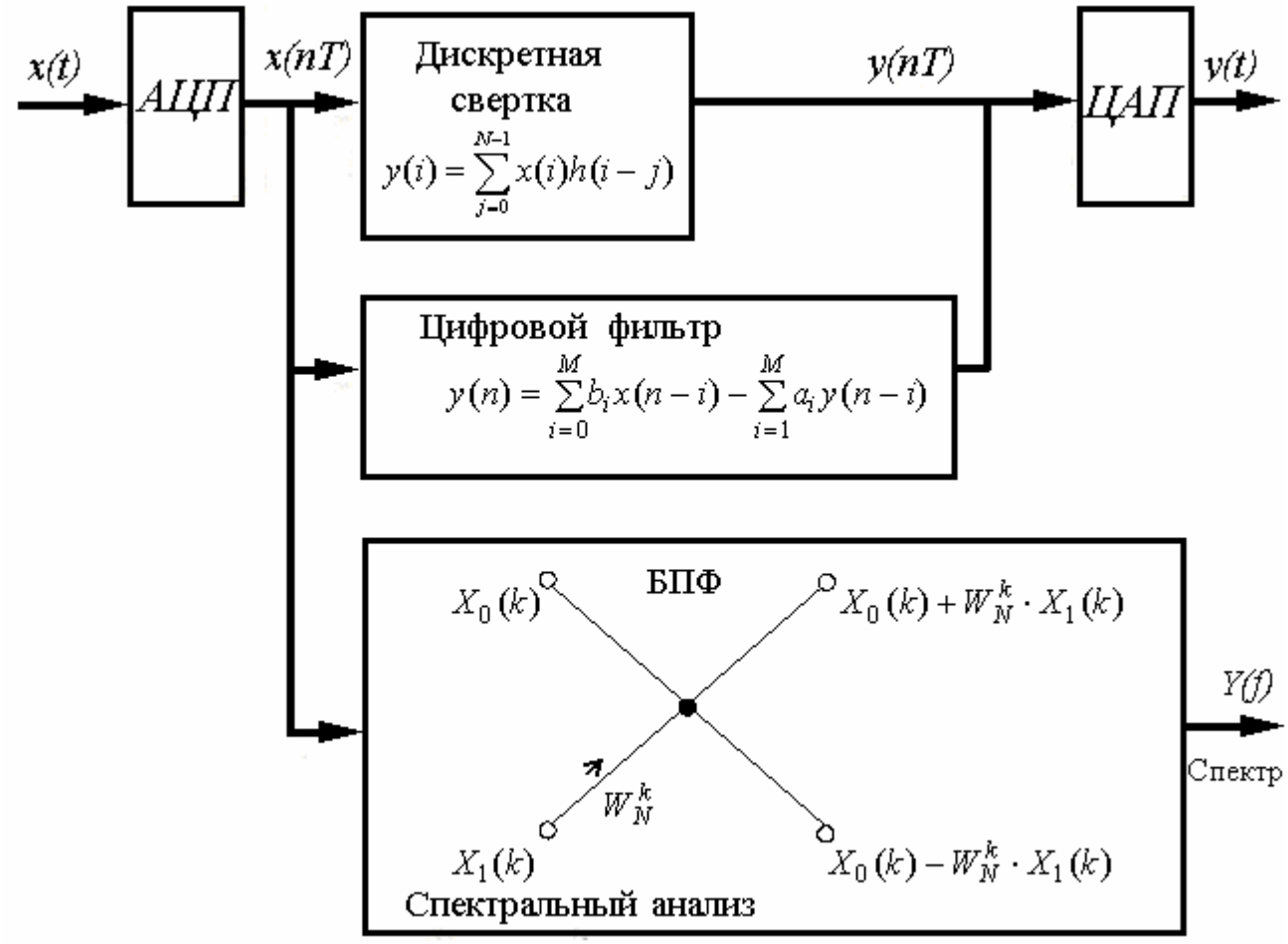


Рисунок 8.5 - Операционный базис процессоров DSP

Организация подсистемы хранения также во многом отражает специфику алгоритмов ЦОС. Так, в частности, выделение двух блоков памяти X и Y позволяет одновременно считывать операнды для свертки, а выделение памяти программ позволяет распараллелить выполнение этапов командного цикла (чтение команды и формирование адреса - чтение операндов - выполнение - запись результата). Такая модель используется в архитектурах гарвардского типа.

Необходимые сложные операции (sin, arctg, $\sqrt{}$ и др.) целесообразно выполнять табличным способом.

В случае использования A- и μ-нелинейных зависимостей, для представления отсчета требуется в данном случае 8 бит: 1 бит - знак, 3 бита - номер участка A- или μ-закона, 4 бита - номер шага на выбранном участке. Данные нелинейные характеристики позволяют устранить избыточность данных за счет компандирования (COMpressor/ exPANDER), которое заключается в сужении динамического диапазона при аналого-цифровом преобразовании и расширение при цифроаналоговом преобразовании.

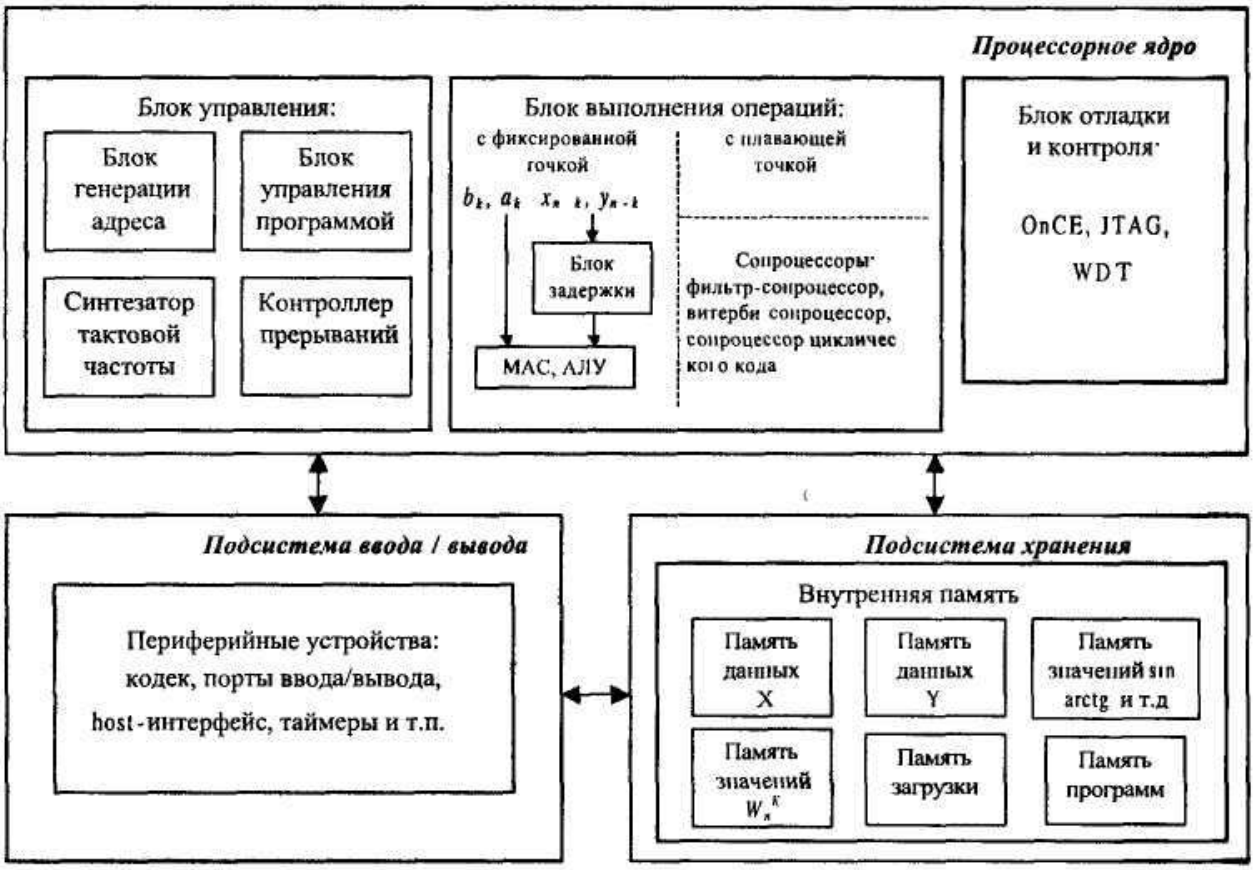


Рисунок 8.6 - Обобщенная архитектура DSP

Очевидно, что скорость выполнения алгоритма БПФ существенно возрастет, если хранить таблицы значений W_N^k. Наличие памяти загрузки позволяет упростить процесс занесения кода программы в процессор. Конкретную реализацию обобщенной архитектуры, рассмотрим на примере семейства DSP 56000, которое по организации процессорного ядра и составу периферии является базовым /12,13/.

8.4 Процессоры семейства DSP56000

8.4.1. Обзор архитектуры и шинной организации DSP56000

Высокая производительность процессоров DSP позволяет использовать их в области коммуникаций, высокоскоростного управления, обработки данных, компьютерных и аудио приложениях. Цифровая обработка сигналов эффективна в любой области электроники. А возможности DSP обеспечивают реализацию цифровой обработки для любой аналоговой электронной схемы.

Типичными задачами цифровой обработки сигналов, для решения которых успешно используется DSP, являются обработка одномерных и двумерных сигналов большой размерности, например, речь и изображение, в таких областях человеческой деятельности, как:

- теле- и радиокоммуникации (голосовая почта, телеконференции, закрытая связь);
- компьютерное оборудование (матричные процессоры, рабочие станции, графические ускорители);
- медицинская электроника;
- цифровое видео;
- радары и сонары (навигация, океанография, поиск);
- сейсмография.

Операция умножения с накоплением (MAC) является базовой операцией алгоритмов типа свертки, широко используемых в задачах цифровой обработки. Структура DSP содержит необходимые блоки, обеспечивающие эффективную реализацию операции MAC. Два операнда непосредственно участвуют в операции умножения, и результат операции затем суммируется с ранее полученными. Этот процесс происходит внутри DSP56000 с использованием двух отдельных модулей памяти за один цикл. Благодаря наличию двух модулей памяти и независимого суммирующего умножителя, можно объединить две пересылки, умножение и сложение в одну операцию.

Основу процессора составляют три параллельно работающих устройства арифметико-логическое устройство (ALU), устройство генерации адреса (AGU) и программируемый контроллер (PC).

DSP имеет периферийные устройства на кристалле в стиле микроконтроллеров: программируемую память, память данных и порты расширения памяти. Программная модель и система команд ориентированы на разработку эффективных и компактных программ.

Основные качественные и количественные характеристики семейства DSP56000:

- скорость - до 30 и более миллионов операций в секунду;
- точность представления данных - 24-разрядные данные, обеспечивающие обработку в динамическом диапазоне 144 децибел, промежуточные результаты хранятся в 56-разрядном аккумуляторе, что обеспечивает диапазон в 336 децибел;
- параллелизм - каждое из операционных устройств на кристалле, память, периферийные операции независимы и работают параллельно благодаря развитой системе шин;

• интеграция - в дополнение к трем независимым операционным устройствам DSP имеет шесть видов памяти на кристалле, три периферийных устройства (последовательный коммуникационный интерфейс (SCI), синхронный последовательный интерфейс (SSI), host-интерфейс, генератор тактовой частоты и семь шин (три адресных и четыре шины данных), при этом система является компактной и достаточно дешевой, а также обладает низким энергопотреблением;

• конвейер команд — трехступенчатый конвейер инструкций прозрачен для программиста;

• система команд - 62 инструкции, мнемоники которых совпадают с инструкциями микроконтроллеров и упрощают трансляцию программ для DSP. Дополнительные инструкции DSP служат для управления параллельными операционными устройствами;

• совместимость программного кода для всех представителей семейства;

• низкое энергопотребление - за счет использования КМОП-технологии, а также следующих дополнительных возможностей управления энергопотреблением: инструкция WAIT; инструкция STOP, останавливающая внутренний генератор тактовой частоты; уменьшение потребления энергии за счет уменьшения частоты.

Приведенные характеристики отражают специализацию архитектуры DSP для приложений цифровой обработки сигналов.

Структура DSP56002, который является базовым для семейства, представлена на рисунке 8.7.

Основные компоненты DSP:

• шины данных;
• шины адреса;
• арифметико-логическое устройство данных;
• устройство генерации адреса;
• память данных X;
• память данных Y;
• контроллер программ;
• память программ;
• устройства ввода/вывода: расширение памяти (порт A); порты ввода/вывода (B и C); host-интерфейс; последовательные интерфейсы; таймеры.

Шины данных. DSP организован на регистрах центрального процессора, составляющих три независимых операционных устройства. Передача данных на кристалле осуществляется по четырем двунаправ-

ленным 24-разрядным шинам: шине данных X (XDB), шине данных Y (YDB), шине данных программ (PDB), глобальной шине данных (GDB). Шины данных X и Y могут объединяться в 48-разрядную шину для передачи инструкций. Шины XDB и YDB сделаны локальными для увеличения скорости и уменьшения потребления мощности. Все другие передачи данных происходят по шине GDB. Структура шин поддерживает передачу данных типов «регистр - регистр», «регистр - память», «память - регистр». Размер передаваемых данных изменяется от 24- до 56-битных слов в одном командном цикле. Переходы между шинами осуществляются с помощью специального переключателя.

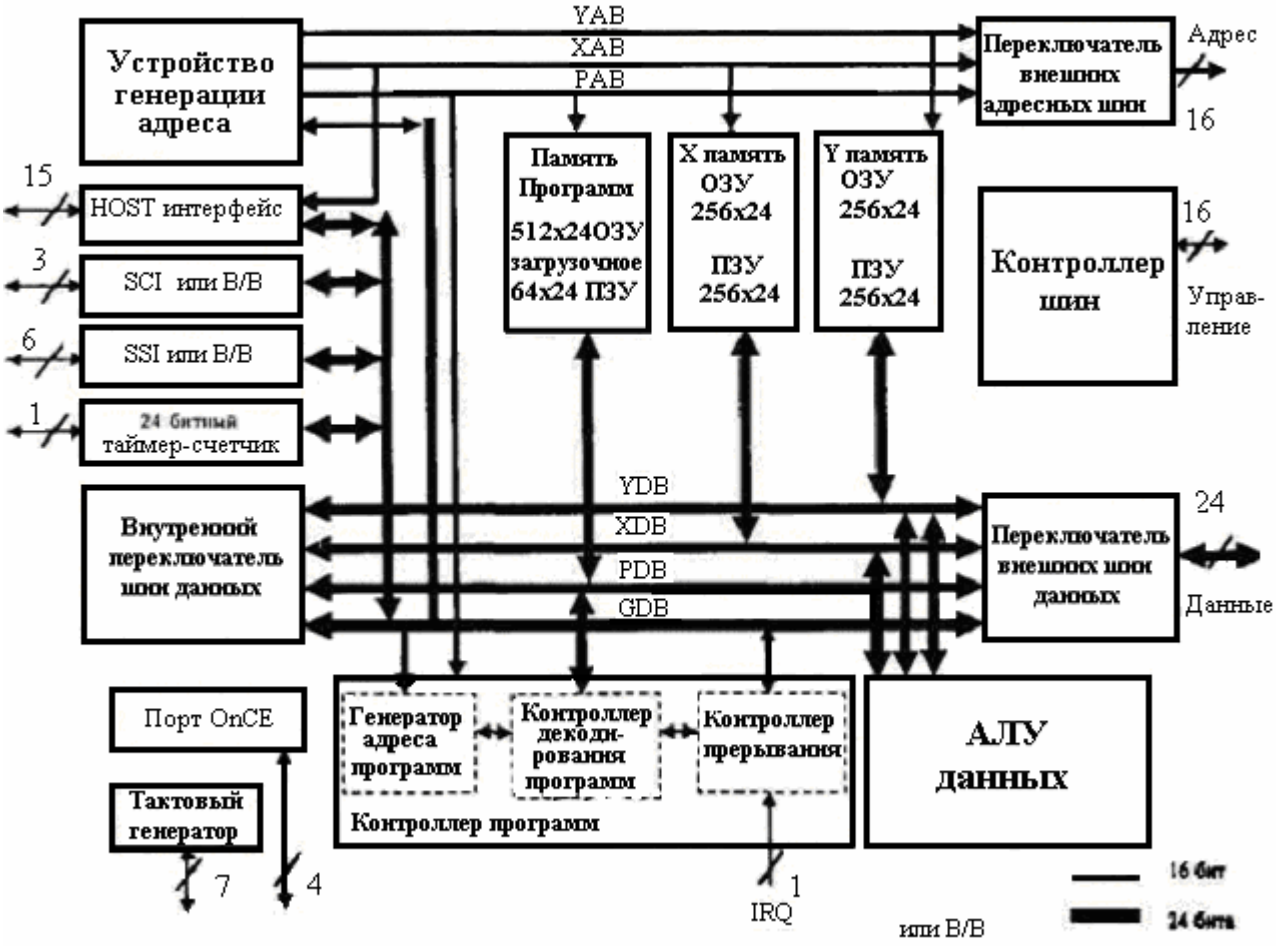


Рисунок 8.7 - Структура DSP56002

Шины адреса. Адреса внутренней памяти данных X и Y передаются по двум однонаправленным 16-битным шинам - адресной шине X (XAB) и адресной шине Y (YAB). Адреса памяти программ передаются по двунаправленной шине адресов программ (PAB).

Пространство внешней памяти адресуется с помощью однонаправленной 16-битной шины с тремя состояниями выходов, которую

можно переключить на XAB, YAB или PAB. В одном командном цикле может быть произведен только один доступ к внешней памяти.
Переход между шинами осуществляется с помощью переключателя внутренних шин. Он представляет собой программируемую матрицу ключей, позволяющую соединять две любые внутренние шины без задержек.

В блоке переключателя шин расположено устройство манипуляций с битами за счет которого переключатель обеспечивает доступ к любому адресному пространству. Устройство манипуляций с битами выполняет битовые операции (установку, сброс, сравнение и т.п.) в памяти, адресных регистрах, регистрах управления и регистрах данных через шины XDB, YDB и GDB.

Арифметико-логическое устройство данных. АЛУ данных (рисунок 8.8) разработано для улучшения возможности обработки сигналов широкого динамического диапазона. Специальные схемы обеспечивают простой контроль ошибок округления и переполнения обрабатываемых данных. АЛУ содержит четыре 24-битных входных регистра, два 48-битных аккумулятора, два 8-битных регистра расширения аккумулятора, аккумулятор сдвига, две шины данных и параллельное неконвейеризованное устройство умножения с аккумулированием (MAC). Операции АЛУ используют арифметику дополнения до двух, в которой для представления отрицательных чисел используется дополнительный машинный код. В регистры АЛУ могут быть записаны 24- или 48-разрядные операнды. АЛУ позволяет выполнять в одном командном цикле инструкции умножения, сложения, вычитания, итерационного деления, нормализации, сдвигов, логических операций. Регистры АЛУ данных могут быть прочитаны или записаны 24-битными или 48-битными операндами по шинам XDB, YDB. Операнды источника могут иметь длину 24,48 и 56 бит и всегда находятся в регистрах АЛУ данных. Результат любой операции в АЛУ данных сохраняется в аккумуляторе. Формат чисел с фиксированной запятой в виде правильных дробей с представлением отрицательных чисел в дополнительном коде.

24-битные данные обеспечивают динамический диапазон в 144дБ. Такого диапазона достаточно для большинства приложений, в которых используются параметры разрядностью не больше 24 бит. 56-битный аккумулятор АЛУ данных обеспечивает 336 дБ внутреннего динамического диапазона.

АЛУ данных имеет следующие компоненты:

- четыре 24-битных входных регистра;

- параллельный умножитель - аккумулятор (MAC);
- два 48-битных регистра аккумулятора;
- два 8-битных регистра расширения аккумулятора;
- аккумулятор сдвига;
- два устройства сдвига/ограничения по шине данных.

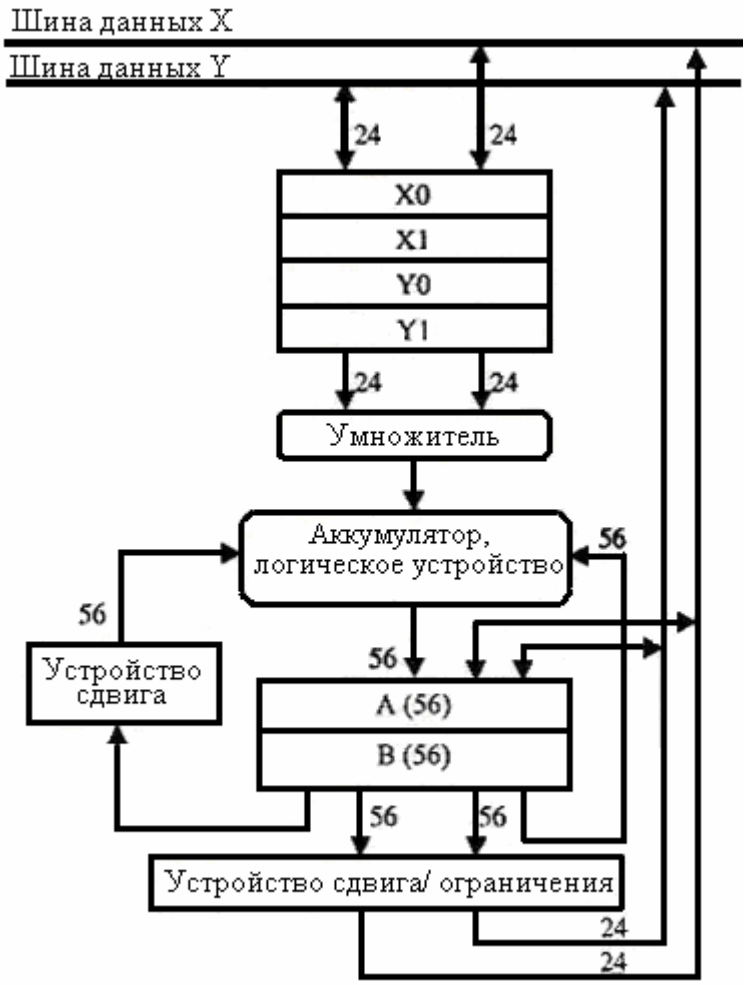


Рисунок 8.8 - АЛУ данных

Входные регистры АЛУ данных (X1, X0, Y1, Y0). Это 24-битные регистры данных общего назначения. Они могут использоваться как независимые 24-битные регистры или как два 48-битных регистра X и Y, образованные объединением X1 :X0 и Y1 :Y0 соответственно. X1 - старшее слово X, Y1 -старшее слово Y. Регистры служат входными буферами между XDB или YDB и устройством MAC. Они используются как операнды источника АЛУ данных. Содержимое регистров может быть выдано на соответствующую шину данных для процедур обработки прерываний.

Аккумуляторы АЛУ данных. Шесть регистров АЛУ данных (A2, A1, A0, B2, B1 и B0) формируют два 56-битных аккумулятора общего

назначения. Каждый из аккумуляторов содержит по три объединенных регистра (A2:A1:A0 и B2:B1:B0). 24-битная MSP хранится в A1 или B1, 24-битная LSP - в A0 или B0. 8-битное расширение - в A2 или B2.

8-битные регистры расширения предназначены для защиты от переполнения. В DSP56000/DSP56001 диапазон значений операндов составляет от -1 до +0,9999998. Если сумма двух чисел меньше -1 или больше +0,9999998, то возникает потеря значимого разряда или переполнение. 8-битный регистр расширения формирует результат в случае 8 переносов.

При использовании регистра расширения аккумулятора бит переполнения в регистре кодов условий всегда установлен в единицу.
Расширение знака обеспечивается автоматически при записи в 56-битный аккумулятор A или B 48-битного или 24-битного операнда. Если записывается 24-битный операнд, ISP автоматически заполняется нулями для представления операнда в 56-битной форме,

Если аккумуляторы A и B прочитаны, можно с помощью операций масштабировать их содержимое на один бит влево или вправо для блока арифметики с плавающей точкой.

MAC и логическое устройство. MAC и логическое устройство входят в главное арифметическое устройство DSP и выполняют все вычисления с операндами данных. В случае арифметической команды устройство обрабатывает до трех операндов и выдает 56-битный результат в следующем формате: расширение: старшая значащая часть: младшая значащая часть (EXT: MSP: LSP).

Операция MAC выполняется независимо и параллельно с операциями на шинах данных, что облегчает буферизацию входных и выходных данных АЛУ. Арифметическое устройство MAC содержит умножитель и два аккумулятора. На вход умножителя могут поступать данные только из регистров X и Y. Умножитель выполняет умножение 24 x 24 с дополнением до двух. 48-битный результат выравнивается и суммируется с 56-битным содержимым аккумулятора A или B. 56-битная сумма сохраняется в аккумуляторе (рисунок 8.9).

8-битное расширение аккумулятора фиксирует переполнение до 256 и позволяет складывать или вычитать содержимое 56-битных аккумуляторов. Операция MAC (умножение/ сложение) - неконвейеризованная одноцикловая операция. Если инструкция предусматривает умножение без сложения, то устройство MAC очищает аккумулятор и складывает его содержимое с произведением. Таким образом, результат всех арифметических инструкций представляется в формате

EXT:MSP:LSP (A2:A1:A0 или B2:B1.B0). Если результат сохраняется в 24-битном операнде, LSP просто отсекается или округляется до MSP.

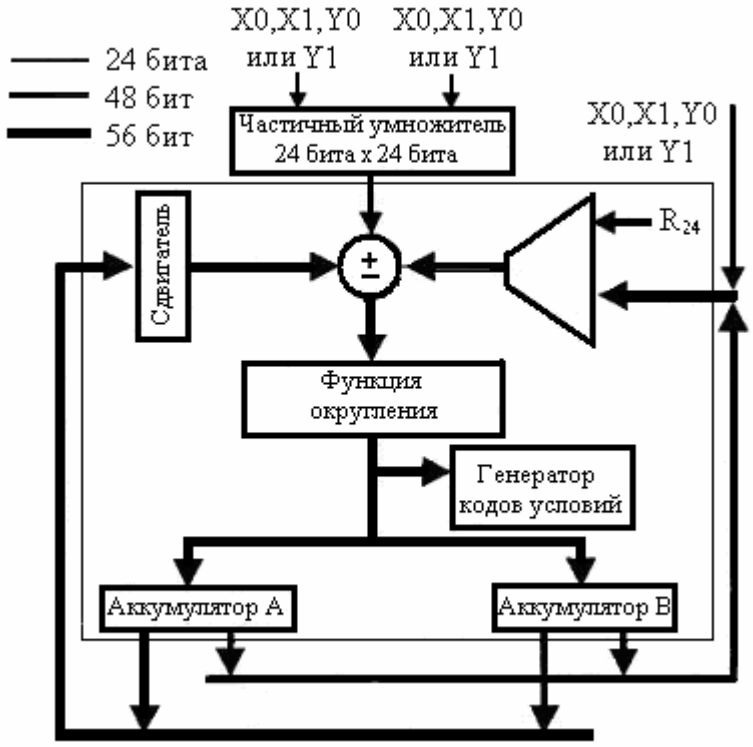


Рисунок 8.9 - Устройство MAC

Округление до ближайшего целого применяется при сложении произведения с содержимым аккумулятора в специальной команде DSP - MACR. Бит, до которого идет округление в аккумуляторе, определяется битом режима масштабирования в регистре статуса.

Логическое устройство выполняет операции логического умножения AND, логического сложения OR, исключающее ИЛИ EOR, инверсии NOT над регистрами АЛУ. Это устройство имеет разрядность 24 бита и оперирует старшей значащей частью операнда в аккумуляторе.

Устройство сдвига аккумулятора. Асинхронное параллельное устройство сдвига с 56-битным входом и 56-битным выходом подключено непосредственно к выходам MAC. Выполняет операции: нет сдвига; сдвиг на один бит влево: ASL, LSL, ROL; сдвиг на один бит вправо: ASR, ASL, ROR; обнуление.

Устройство сдвига/ограничения. Обеспечивает специальную обработку данных при выдаче из регистров аккумулятора на шины данных. Для каждой шины данных имеется отдельное устройство сдвига/ограничения. Операция ограничения в случае переполнения разрядной сетки выполняется посредством замены результата вычислений

максимально возможной величиной. Знак данных устанавливается соответственно операнду источника: $7FFFFF - для 24-битного или $7FFFFF FFFFFF - для 48-битного положительного числа, $800000 - для 24-битного, $800000 000000 -для 48-битного отрицательного числа.

MAC использует округление содержимого аккумулятора с одинарной точностью. Данный метод округления называется округлением до ближайшего целого или конвергентным округлением.

Пространство памяти DSP56002. Пространство памяти DSP56002 представлено на рисунке 8.10. Регистр OMR расширен битами YD (3 бит), MC (4 бит) SD (6 бит). Биты DE и YD задают режимы памяти (таблица 8.1).

Таблица 8.1 - Режимы работы памяти

DE	YD	Память данных
0	0	Внутренние ПЗУ запрещены и их адреса являются частью внешней памяти
0	1	Внутреннее ПЗУ данных X запрещено и является частью внешней памяти. Внутренние ОЗУ и ПЗУ данных Y запрещены и являются частью внешней памяти
1	0	ПЗУ данных X и Y разрешены
1	1	Внутренние ОЗУ и ПЗУ данных Y запрещены и являются частью внешней памяти. Внутреннее ПЗУ данных X разрешено

Однокристальный режим. В однокристальном режиме разрешены все внутренние памяти данных и программ. Программный сброс вызывает переход по адресу $0000 и выполнение соответствующей процедуры.

Карты памяти для режима 0 и режима 2 идентичны, но в режиме 2 вектор сброса находится по адресу $E000.

Режим загрузки из EEPROM. Загрузочные режимы предназначены для загрузки программ из байтового ПЗУ во внутреннюю память программ в течение сброса по питанию. После включения питания генератор состояний ожидания добавляет 15 состояний ожидания ко всем обращениям к внешней памяти, что позволяет использовать медленную память.

Программа загрузки использует байты в трех последовательно расположенных ячейках внешнего ПЗУ для образования слова внутренней памяти программ. В загрузочном режиме разрешены загрузочное ПЗУ на кристалле и выполнение программы загрузки.

Нормальный расширенный режим. В этом режиме разрешено внутреннее ОЗУ программ и вектор аппаратного сброса по адресу $E000.

Режим разработки. В этом режиме внутреннее ОЗУ программ запрещено, а вектор аппаратного сброса находится по адресу $0000. Все ссылки на пространство памяти программ переадресуются прямо во внешнюю память.

Режим загрузки по Host. В этом режиме разрешены загрузочное ПЗУ и выполнение программы загрузки. Режим аналогичен режиму 1, за исключением того, что программа загружается из внешней памяти программ по host-интерфейсу.

Режим загрузки по SCI. В этом режиме разрешены загрузочное ПЗУ и выполнение программы загрузки. Внутреннее и/или внешнее ОЗУ программ загружается по интерфейсу SCI Количество слов программы и стартовый адрес должны быть определены.

Программа загрузки по SCI ожидает приема трех байтов, определяющих количество слов программы, трех байтов, определяющих адрес, с которого будут загружаться слова программы, и трех байтов для загрузки каждого слова программы. Режимы DSP56002 и управляющие сигналы приведены в таблице 8.2.

Данные принимаются в SCI, начиная с младших битов. После окончания приема слов программы начинается выполнение загруженной программы. SCI запрограммирован на работу в асинхронном режиме с длиной данных - 8 бит, 1 стоп-бит, проверка на четность отсутствует.

Таблица 8.2 -Операционные режимы работы DSP56002

Опер. режим	MC	MB	MA	Описание
0	0	0	0	Однокристальный режим - ОЗУ программ разрешено
1	0	0	1	Режим загрузки с EPROM, выход в режим 0
2	0	1	0	Нормальный расширенный режим - ОЗУ программ разрешено
3	0	1	1	Режим разработки - ОЗУ программ запрещено
4	1	0	0	Резерв
5	1	0	1	Загрузка по Host, выход в режим 0
6	1	1	0	Загрузка по SCI (внешняя синхронизация), выход в режим 0
7	1	1	1	Резерв

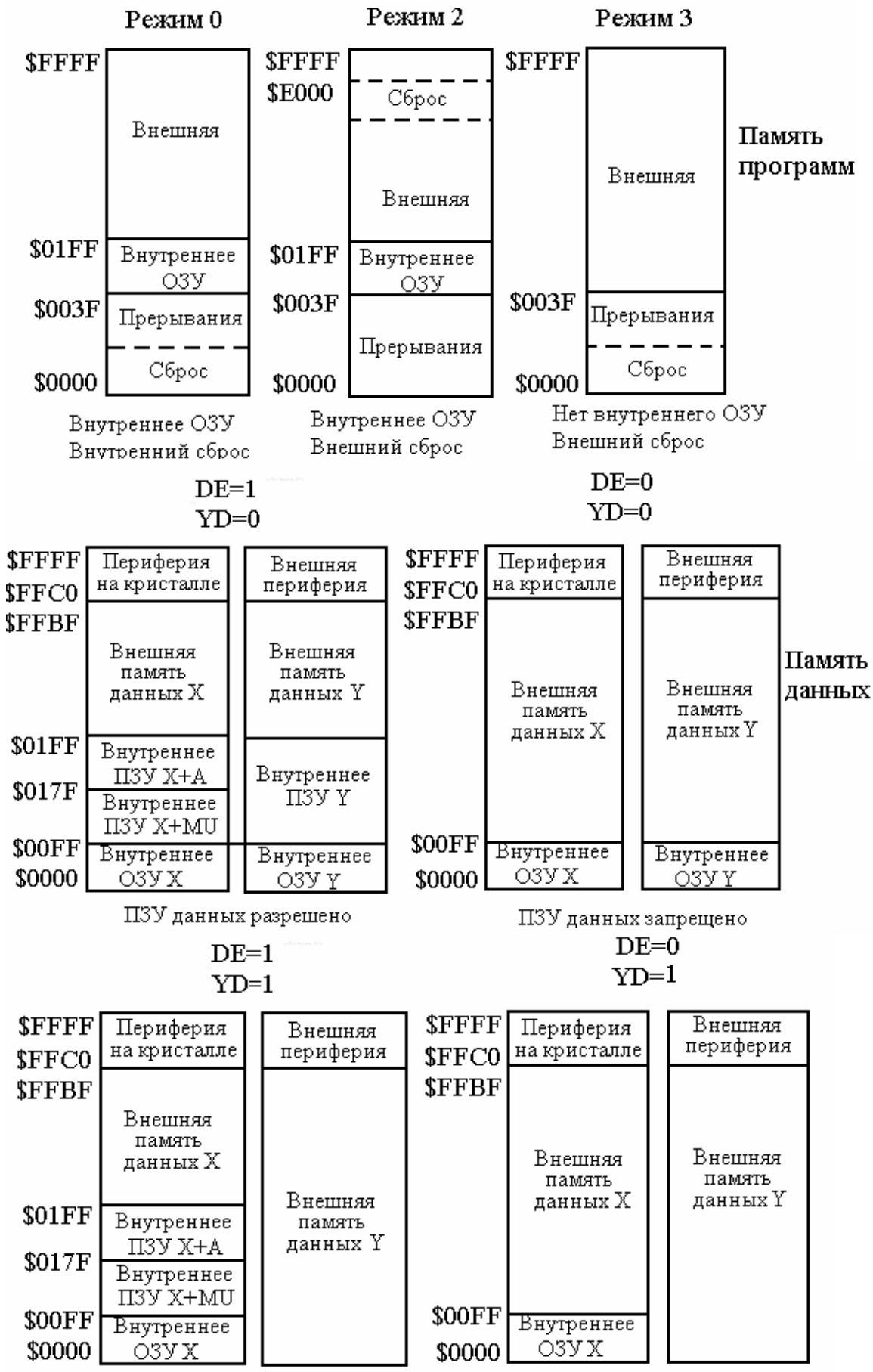


Рисунок 8.10 – Пространство памяти DSP56002

Устройство генерации адресов (address generate unit - AGU) и режимы адресации.

Структура AGU. AGU выполняет вычисление истинных (физических) адресов данных в памяти. Это устройство использует три типа адресной арифметики: линейную модификацию адреса, модульную модификацию адреса, адресацию с реверсивным переносом - и работает параллельно с остальными устройствами на кристалле, что сокращает время на генерацию адресов. Структура AGU показана на рисунке 8.11. Все регистры AGU являются 24-разрядными, в которых 16 младших разрядов являются значащими. Старшие 8 разрядов недоступны для записи и при чтении заполняются нулями.

Блок регистров адреса (R0-R3 и R4-R7). Каждый из двух подблоков регистров адреса содержит четыре 16-битных регистра, которые содержат адреса для обращения к памяти. Каждый регистр может быть прочитан или записан с использованием глобальной шины данных. При выдаче содержимого регистров на шину данных 16-битные регистры записываются в два младших значащих байта шины данных, а старший значащий байт заполняется нулями.

При записи в регистры старший значащий байт шины данных отсекается. Каждый адресный регистр может использоваться как вход АЛУ адресов для модификации регистров. Регистр из АЛУ нижних адресов и регистр из АЛУ верхних адресов доступны в одной инструкции.

Если параллельно осуществляется пересылка данных из памяти X и из памяти Y, адресные регистры разделяются на два блока: R0 - R3, R4 - R7. Содержимое регистров адреса может модифицироваться соответственно режиму адресации. Тип модификации определяется содержимым регистров модификации (Mn). Регистры смещения (Nn) используются для режима адресации «модификация со смещением».

Большинство режимов адресации модифицируют регистры адреса в цикле «чтение - модификация - запись».

Блок регистров смещения (N0-N3 и N4-N7). Каждый из двух подблоков регистров смещения содержит четыре 16-битных регистра, которые содержат величину смещения, используемую для модификации адресных указателей или данных. Каждый регистр может быть прочитан или записан с использованием глобальной шины данных. При выдаче содержимого регистров на шину данных 16-битные регистры записываются в два младших значащих байта шины данных, а старший значащий байт заполняется нулями. При записи в регистры старший значащий байт шины данных отсекается.

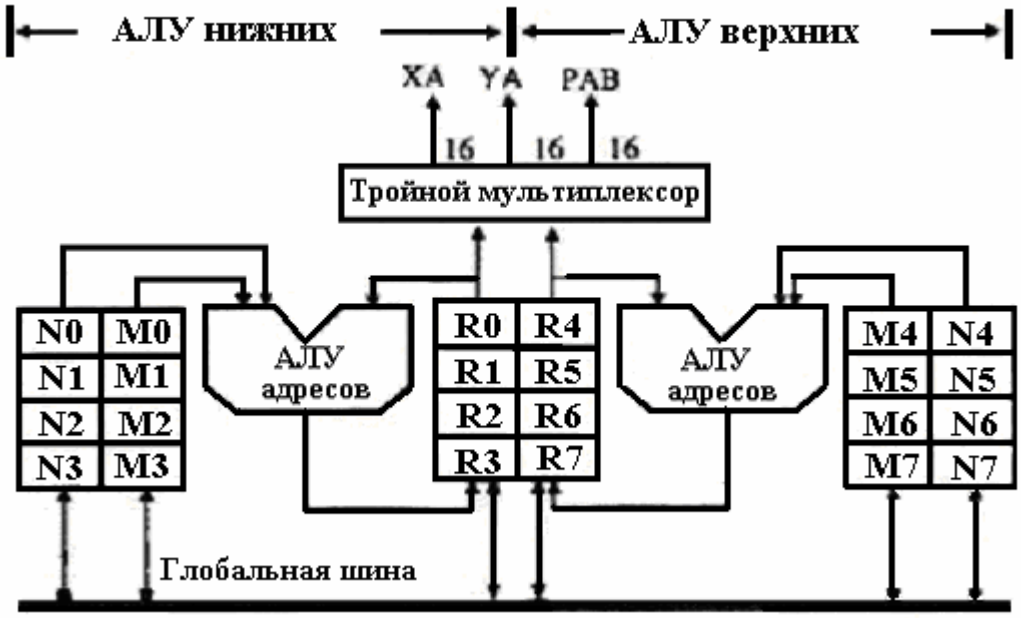


Рисунок 8.11 - Структура AGU

Блок регистров модификации (M0-M3 и M4-M7). Каждый из двух подблоков регистров модификации содержит четыре 16-битных регистра, которые определяют тип адресной арифметики для вычисления модификаций регистров адреса или данных. Каждый регистр может быть прочитан или записан с использованием глобальной шины данных. При выдаче содержимого регистров на шину данных 16-битные регистры записываются в два младших значащих байта шины данных, а старший значащий байт заполняется нулями. При записи в регистры старший значащий байт шины данных отсекается. Каждый регистр модификации устанавливается в $FFFF после сброса, что определяет линейную арифметику при вычислении модификаций
адреса.

АЛУ адресов. Два идентичных АЛУ адресов содержат 16-битный полный адрес, который может быть инкрементирован, декрементирован или к которому может быть добавлено содержимое регистра смещения. Второе полное слагаемое (называемое модулем) определяется результатом суммирования первого полного слагаемого с величиной модуля, хранящейся в регистре модификации. Третье полное слагаемое определяется инкрементированием, декрементированием содержимого адресного регистра или его суммированием с величиной смещения и переносом. Смещение и реверсивный перенос подаются параллельно на разные входы. Тестовая логика определяет, какой из трех результатов подается на выход в качестве полного адреса

Каждое АЛУ адресов может модифицировать один регистр адреса в течение одного командного цикла. Содержимое регистра модификации определяет тип арифметики для модификации адреса. Величина модификатора декодируется в АЛУ адресов.

Адресация. DSP56000/DSP56001 обеспечивают три различных типа адресации: прямая регистровая, косвенная регистровая и специальная. При прямой регистровой адресации и специальных режимах использование AGU не является обязательным. При прямой регистровой адресации в команде указывается физический адрес регистра, в котором находится операнд. Если регистр адреса используется для указания ячейки памяти, адресация называется косвенной регистровой, т.е. содержимое регистра указывает не на сам операнд, а на адрес операнда. Специальная адресация определяет адрес с учетом его модификации. Модификация заключается в увеличении или уменьшении на единицу содержимого адресного регистра до или после выполнения команды.

Программный контроллер. Программный контроллер обеспечивает предварительную выборку инструкций, декодирование инструкций, управление аппаратными циклами и обработку исключений Контроллер содержит 15-уровневый 32-разрядный системный стек и шесть непосредственно адресуемых регистров (рисунок 8.12), программный счетчик PC (Program Counter), регистр адреса цикла LA (Loop Address), счетчик петли LC(Loop Counter), регистр статуса SR (Status Register), регистр режима операций OMR (Operating Mode Register) и указатель стека SP (Stack Pointer).

Системный стек представляет собой отдельный блок внутренней памяти, используемый для хранения регистра статуса и программного счетчика при вызовах подпрограмм и длительных прерываниях. В стеке также хранятся LA и LC для организации программных циклов.

Каждая ячейка стека включает два 16-разрядных регистра: старшую часть (SSH) и младшую часть (SSL) стека.

Все эти регистры могут быть прочитаны или записаны для упрощения отладки системы. Хотя ни один из регистров программного контроллера не является 24-битным, все они читаются и записываются через 24-битную шину PDB. При чтении регистров младшая часть битов (LSB) является значащей, а старшая значащая часть битов (MSB) заполняется нулями. При записи регистров младшая часть битов (LSB) является значащей, а старшая значащая часть битов просто отбрасывается, поскольку имеет смысл только младшая значащая часть битов (биты 15-0).

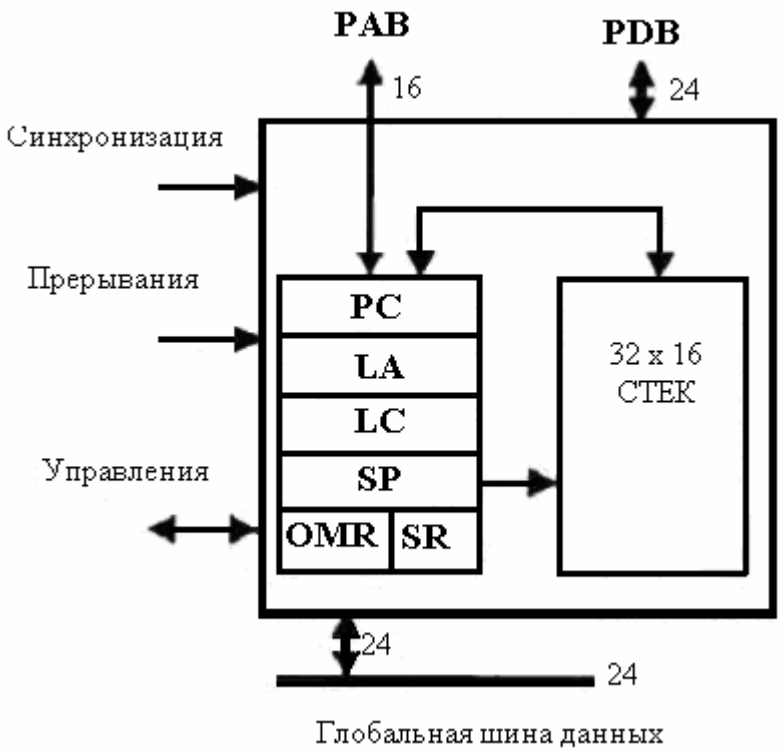


Рисунок 8.12 - Регистры контроллера

Программный контроллер содержит конвейер с тремя ступенями и управляет пятью состояниями процессора: нормальным, обработкой исключений, сбросом, ожиданием и остановом.

Структура программного контроллера. Программный контроллер состоит из трех аппаратных блоков: контроллера декодирования программ (PDC), генератора адресов программ (PAG) и контроллера прерываний (PIC).

Контроллер декодирования программ. PDC включает в себя программную логику для декодирования, генератор адреса регистра, механизм выполнения циклов, механизм выполнения повторений, генератор кодов условий, механизм выполнения прерываний, регистр-защелку инструкций и его копию. PDC декодирует 24-битную инструкцию, загруженную в регистр-защелку, и вырабатывает все сигналы, необходимые для управления конвейером. Копия регистра-защелки инструкций оптимизирует выполнение инструкций повторения и перехода.

Генератор адресов программ. PAG содержит: регистры PC, SP, SS (системный стек), OMR (регистр операционного режима), SR, LC и LA. Циклы, являющиеся основной конструкцией алгоритмов цифровой обработки сигналов, поддерживаются аппаратно.

При выполнении инструкции цикла DO в регистр счетчика цикла загружается количество повторений цикла, а в регистр адреса цикла -

адрес последней инструкции цикла, и устанавливается флаг цикла в регистре статуса. Перед выполнением инструкции DO содержимое регистров LA, LC и SR сохраняется в стеке. Под управлением механизма выполнения циклов адрес первой инструкции цикла помещается в стек. Пока флаг цикла в регистре статуса не сброшен, механизм выполнения циклов сравнивает содержимое PC с содержимым LA для определения последней инструкции цикла. Когда последняя инструкция выбрана, содержимое LC сравнивается с единицей. Если равенство не выполняется, содержимое LC декрементируется и из SS читается адрес первой инструкции цикла. Если равенство выполняется, то значения LA, LC и флага цикла в SR восстанавливаются из стека, а выборка инструкций продолжается с адреса LA + 1.

Пересылка массива данных может быть выполнена с использованием механизма повторений. Инструкция REP загружает в LC количество повторений следующей за ней инструкции. Так как команда, которая будет повторяться, выбрана только один раз, это увеличивает производительность за счет уменьшения обращений к внешней шине. Однако инструкция REP не может быть прервана, поскольку выбирается только один раз.

Контроллер прерываний. PIC принимает все запросы прерываний, осуществляет арбитраж в каждом цикле и генерирует адрес вектора прерывания. Прерывания могут вызывать четыре внешних и 16 внутренних источников прерываний.

Используется структура гибкого приоритета прерываний. Каждое прерывание получает свой уровень приоритета (IPL) - от 0 до 3. Уровень 0 - самый низкий, а уровни 1 и 2 маскируются. Уровень 3 является высшим и не маскируется. Биты маски прерываний в регистре статуса (SR) показывают текущий уровень приоритета прерываний в процессоре. Прерывания, имеющие уровень приоритета меньше текущего, не учитываются при арбитраже. Уровень приоритета 3 всегда вызывает прерывание процессора. Уровни приоритета прерываний для каждого периферийного устройства на кристалле (HI, SSI, SCI) и для каждого внешнего источника прерываний (¬IRQA, ¬IRQB) могут задаваться программным путем от 0 до 2. Уровни приоритета устанавливаются при записи в регистр уровней приоритета, представленный на рисунке 5.13.

В DSP56002 регистр уровней приоритета прерываний расширен: добавлены биты 16 -TIL1 и 17 - TIL0, которые задают уровень приоритета прерываний от таймера. Источники прерываний и их уровни приоритета указаны в таблице 8.3. Каждый источник прерываний имеет

свой вектор для вызова процедуры обработки прерывания, расположенной в младших 64 словах памяти программ. При переходе к обработке исключений текущая инструкция выполняется, если только выбранное слово не является первым словом двухсловной инструкции, иначе выполнение инструкции прерывается.

Далее выполняется две выборки адреса, во время которых PC не изменяется. PIC генерирует выборку адреса инструкции прерывания, указывающего на первое слово двухсловной инструкции подпрограммы быстрой обработки прерывания. Затем производится выборка двух слов инструкции по вектору прерывания.

Если одно из этих слов - переход к подпрограмме, то быстрая обработка прерывания переводится в длительную. Процедура длительной обработки прерывания помещает текущий контекст процессора в стек. Подпрограммы и прерывания могут быть вложенными. Стек может быть расширен в память за счет использования программного доступа к регистрам SSH и SSL. Контакты внешних прерываний ¬IRQA и ¬IRQB могут быть настроены на прерывания по переднему или заднему фронту. Немаскируемое прерывание NMI осуществляется по уровню и выполняется по первому появлению на контакте ¬IRQB 10B после последнего обслуживания NMI или сброса. NMI имеет уровень приоритета 3 и не маскируется. NMI не используется как вход прерывания общего назначения, поскольку зарезервировано для аппаратных разработок.

Вектор аппаратного сброса может указывать на внутреннюю или внешнюю (P.$E000) память программ. Немаскируемое прерывание, трассировка и программное прерывание используются при отладке.

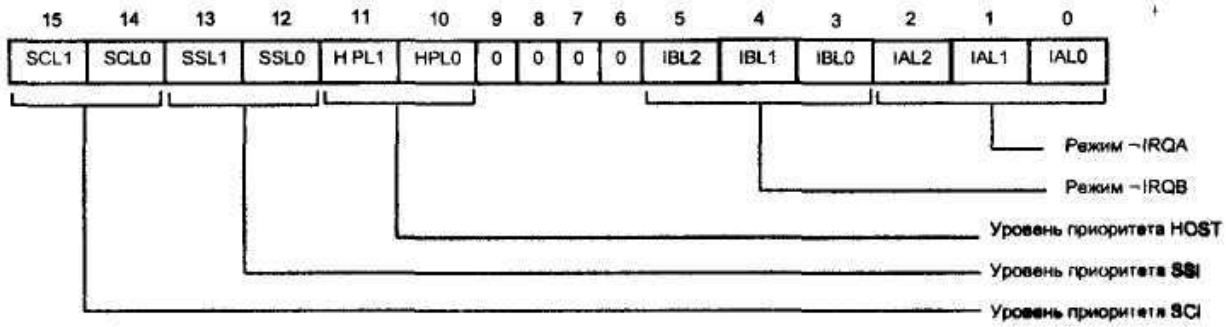


Рисунок 8.13- Формат регистра уровней приоритета

Программное прерывание используется для организации точек останова. При трассировке инструкции выполняются в пошаговом режиме.

Таблица 8.3 - Уровни приоритета

Стартовый адрес	IPL	Источник
P.$0000/P.$E000	3	Аппаратный внешний сброс
P.$0002	3	Ошибка стека
P.$0004	3	Трассировка
P.$0006	3	Программное прерывание SWI
P.$0008	0-2	Внешнее прерывание LLJIRQA
P.$000A	0-2	Внешнее прерывание LUIRQB
P.$000C	0-2	SSI - Приемник данных
P.$000E	0-2	SSI - Приемник данных в состоянии исключения
P.$0010	0-2	SSI - Передатчик данных
P.$0012	0-2	SSI - Передатчик данных в состоянии исключения
P.$0014	0-2	SCI - Приемник данных
P.$0016	0-2	SCI - Приемник данных в состоянии исключения
P.$0018	0-2	SCI - Передатчик данных
P.$001A	0-2	SCI - Простой линии
P.$001C	0-2	SCI - Таймер
P.$001E	3	NMI - Зарезервировано для разработок
P.$0020	0-2	Host - Приемник данных
P.$0022	0-2	Host - Передатчик данных
P.$0024	0-2	Host - Команда
P.$0028	0-2	Доступно для команд Host
P.$002A	0-2	Доступно для команд Host
P.$002C	0-2	Доступно для команд Host
P.$002E	0-2	Доступно для команд Host
P.$0030	0-2	Доступно для команд Host
P.$0032	0-2	Доступно для команд Host
P.$0034	0-2	Доступно для команд Host
P.$0036	0-2	Доступно для команд Host
P.$0038	0-2	Доступно для команд Host
P.$003A	0-2	Доступно для команд Host
P.$003C	0-2	Доступно для команд Host
P.$003E	0-2	Неправильная инструкция

Все периферийные устройства на кристалле используют одинаковый механизм запроса прерываний. Каждое устройство имеет одну линию запроса прерывания в PIC и две входные линии: линию чтения вектора и линию ответа.

Устройство, имеющее более одного источника прерывания, проводит собственный арбитраж прерывания в соответствии с его внутренним уровнем приоритета. PIC выбирает одно из периферийных устройств и посылает ему адрес вектора прерывания.

Конвейер инструкций. Программный контроллер содержит трехуровневый конвейер, выполняющий выборку, декодирование и испол-

нение инструкций. Устройства выборки, декодирования и исполнения расположены последовательно.

Синхронизация. DSP56000/DSP56001 использует четырехфазную синхронизацию выполнения инструкций. Синхронизация обеспечивается внутренним генератором, соединенным с внешним кристаллом контактами XTAL и EXTAL, или внешним генератором по линии EXTAL.

Программная модель. Программная модель контроллера представлена на рисунке 8.14. Она содержит наряду с программным счетчиком, регистром статуса и системным стеком дополнительные регистры счетчика цикла и адреса цикла, предназначенные для аппаратной поддержки инструкции DO.

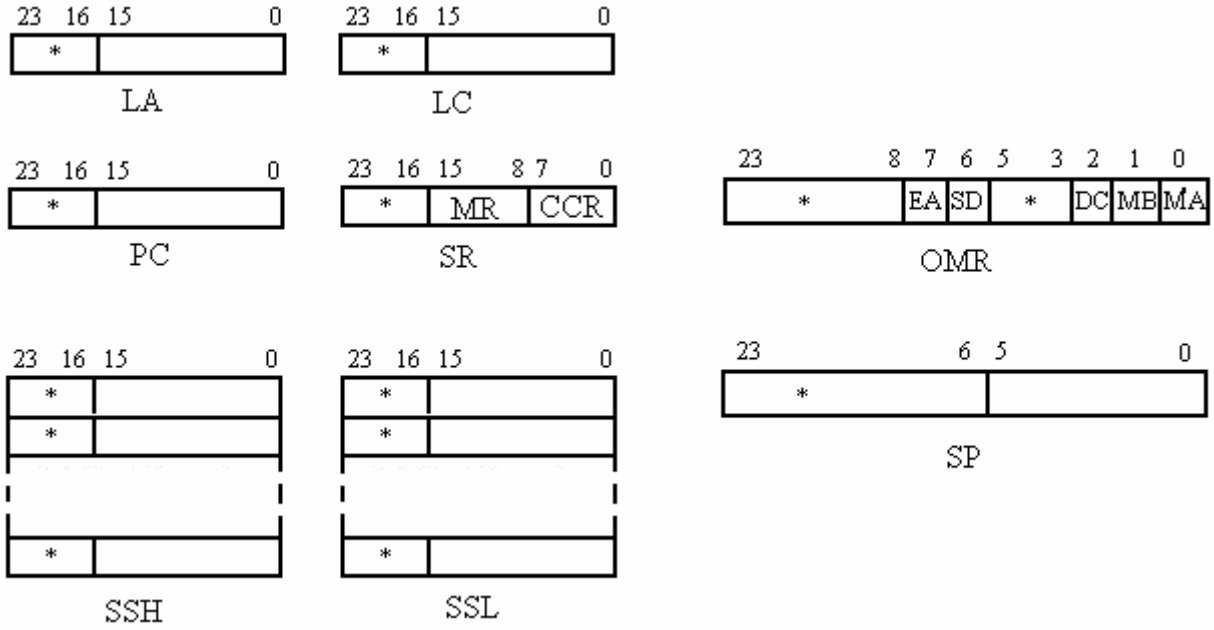


Рисунок 8.14 - Программная модель контроллера
(* - запрещены для изменения, читаются как нуль)

Программный счетчик (PC), 16-битный регистр, содержащий адрес следующей ячейки в памяти программ. PC может указывать на инструкции, операнды данных или адреса операндов. Ссылки на этот регистр подразумеваются в большинстве инструкций. PC запоминается в стеке при выполнении программного цикла, подпрограммы или прерывания.

Регистр статуса (SR). 16-битный регистр статуса SR состоит из регистра режима MR, занимающего 8 старших битов и регистра кодов условий CCR, занимающего 8 младших битов. Регистр статуса сохраняется в стеке при выполнении программного цикла, подпрограммы или прерывания. Формат регистра показан на рисунке 8.15.

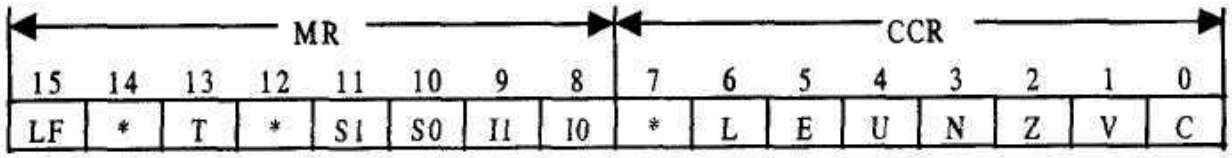


Рисунок 8.15 - Формат регистра статуса SR

Регистр управления (MR). MR - специальный управляющий регистр, определяющий текущее состояние процессора для супервизора. На биты регистра режима MR воздействуют сброс процессора, выполнение цикла, возврат из прерывания, программное прерывание и инструкции, оперирующие непосредственно с содержимым регистра. После аппаратного сброса в регистре устанавливается маска прерывания, биты масштабирования, флаг цикла, бит трассировки сбрасываются в «0».

Регистр кодов условий (CCR). Регистр кодов условий (флагов) определяет текущее состояние процессора для пользователя. На регистр кодов условий воздействуют арифметические и логические операции, операции пересылки и инструкции, оперирующие непосредственно с содержимым регистра. После сброса все биты CCR устанавливаются в «0».

Флаг переноса (C - бит 0) устанавливается в «1» при наличии переноса из старшей значащей части числа в результате сложения. Перенос или заем генерируется из бита 55 результата. Перенос также возникает при манипуляциях с битами, сдвигах и циклических сдвигах.

Флаг переполнения (V-бит 1) устанавливается при арифметическом переполнении 56-битного результата. Это означает, что результат не может быть представлен в аккумуляторе.

Флаг нуля (Z- бит 2) устанавливается в «1», если результат равен нулю.

Флаг отрицательного результата (N - бит 3) устанавливается в «1», если результат имеет отрицательный знак.

Флаг денормализации (U - бит 4) устанавливается, если два старших бита идентичны.

Флаг расширения (E- бит 5) сброшен в «0», если все биты целой части 56-битного результата - нули или единицы, иначе бит установлен. Целая часть определяется режимом масштабирования и битом E. Если бит расширения равен нулю, то младшая часть числа содержит все значащие биты, а старшая часть числа является расширением знака. В этом случае регистр расширения аккумулятора игнорируется. Если бит расширения равен единице, то регистр расширения аккумулятора используется.

Флаг ограничения (L - бит 6) установлен, если установлен бит переполнения. Также устанавливается при выполнении операций ограничения устройством сдвига/ограничения данных. Бит сбрасывается только при сбросе процессора или специальной командой.

Маска прерывания (I1,I0 - биты 8 и 9). Показывает текущий уровень приоритета прерываний в процессоре. Уровень приоритета изменяется программным путем. Биты маски прерываний устанавливаются в «1» при аппаратном сбросе процессора и не изменяются при программном сбросе.

Режим масштабирования (S1,S0 - биты 10 и 11). Биты определяют режим масштабирования для устройства сдвига/ограничения данных АЛУ и позиции округления для MAC. Действие режима масштабирования проявляется при чтении данных из аккумуляторных регистров A и B на шины данных и при округлении в MAC. При сбросе процессора и начале длительной обработки прерывания эти биты равны нулю.

Режим трассировки (T- бит 13). Бит устанавливается в начале выполнения каждой инструкции в пошаговом режиме. Если бит равен нулю, трассировка запрещена и выполнение инструкций идет нормальным образом. При сбросе процессора бит T равен нулю.

Флаг цикла (LF- бит 15). Бит устанавливается при выполнении программного цикла и разрешает определение окончания цикла. По завершении цикла бит устанавливается в «0».

Регистр операционного режима (OMR). OMR -24-битный регистр (рисунок 8.16), в котором определены 5 битов для DSP56000/DSP56001 и 7 битов для DSP56002. Устанавливает текущий операционный режим процессора.

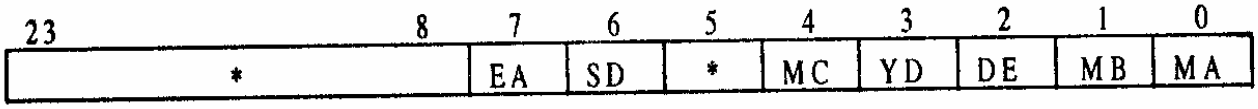


Рисунок 8.16 –Формат регистра операционного режима DSP56002

На биты регистра воздействует только сброс процессора и непосредственные операции с содержимым регистра. После сброса биты MA и MB могут быть загружены извне через контакты A и B.

Биты операционного режима кристалла (MA, MB - биты 0 и 1) показывают режим расширения шин в процессоре. После сброса эти значения загружаются извне через внешние контакты выбора операционного режима ¬MODA и ¬MODB, либо могут быть изменены программным путем.

Бит разрешения ПЗУ данных (DE- бит 2). Установка бита разрешает использование двух ПЗУ данных размером 256 х 24, находящихся по адресам $0100 - $01FF в пространствах памяти X и Y.

Бит запрещения внутренней памяти Y (YD - бит 3).Установка бита запрещает доступ к внутренней памяти данных Y. Если бит равен нулю, внутренняя память данных Y доступна в соответствии с состоянием управляющего бита DE. Аппаратный сброс очищает бит YD

Бит операционного режима кристалла (MC - бит 4). Бит MC наряду с битами MA и MB определяет карту памяти и операционный режим кристалла. После сброса процессор загружает этот бит через внешний контакт MODC После выхода процессора из состояния сброса бит MC может быть изменен программным путем.

Бит задержки останова (SD - бит 6). При выходе из состояния останова (по инструкции STOP) если этот бит равен нулю, генерируется задержка в 64К циклов синхронизации (131072 T состояний). Если бит установлен в «1», задержка составляет 16 T состояний. Длительная задержка используется для стабилизации синхронизации

Бит доступа к внешней памяти (EA - бит 7). Бит определяет функции двух линий порта A, указанные в таблице 8.4.

Таблица 8.4 – Функции линий порта A

EA	Линия ¬BR (Вход)	Линия ¬BG (Выход)
0	Запрос шины (¬BR)	Предоставление шины (¬BG)
1	Ожидание (WT)	Стробирование шины (¬BS)

Регистр адреса цикла и регистр счетчика цикла (LA и LC). Содержимое регистра адреса цикла LA указывает на местонахождение слова последней инструкции программного цикла LC -специальный 16-битный счетчик, определяющий число повторений цикла.

При выполнении инструкции DO содержимое регистра LA помещается в SSH, а регистра LC - в SSL, откуда оно извлекается только после окончания цикла по инструкции ENDDO

При выборке инструкции по адресу, содержащемуся в LA, проверяется содержимое LC. Если содержимое LC не равно единице, оно декрементируется и следующая инструкция выбирается по адресу, находящемуся в верхней ячейке стека В противном случае PC инкрементируется, флаг цикла сбрасывается, регистры LA и LC восстанавливаются из стека и продолжается нормальное выполнение программы.

Регистр LC программно доступен, и выполнение цикла может

быть прекращено программным путем. Регистр LC также используется в инструкции REP.

Системный стек (SS). SS -отдельная внутренняя память размером 15 х 32 бита, разделенная на два блока SSL и SSH. SSH сохраняет содержимое программного счетчика, SSL -регистра статуса при вызовах подпрограмм и длительных прерываниях. В стеке сохраняются также значения LA и LC при выполнении программных циклов. Стек обеспечивает выполнение 15 длинных прерываний, семи циклов DO, 15 вызовов подпрограмм. Если стек переполняется, возникает немаскируемое прерывание - ошибка стека. Значение PC теряется, что приводит к ошибке программы, выполняющейся, когда возникло прерывание по переполнению стека.

Указатель стека (SP). 6-битный указатель стека определяет последнюю занятую ячейку стека и состояние стека (потеря содержимого, пустой, полный, переполнение). Формат регистра SP указан на рисунке 8.17.

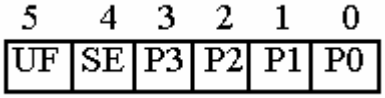


Рисунок 8.17- Формат регистра SP

Указатель стека (P0.P1.P2.P3—биты 0-3) показывает последнюю используемую ячейку стека. После аппаратного сброса все биты равны нулю, что означает - стек пуст.

Флаг ошибки стека (SE- бит 4). Показывает возникновение ошибки стека. Если стек заполнен, указатель стека равен 001111, и любая операция, вызывающая запись в стек, приведет к возникновению ошибки стека, т е SP будет равен 010000 Аналогично, если SP равен 0, любая операция чтения из стека приведет к возникновению ошибки стека. Флаг ошибки стека остается равным единице, пока пользователь не сбросит его в «0».

Флаг потери содержимого (UF- бит 5). Устанавливается при потере содержимого стека. Сбрасывается в «0» пользователем.

Остальные биты стека зарезервированы для будущих разработок

Состояния процессора. DSP всегда находится в одном из пяти состояний: нормальное, обработка исключения, сброс, ожидание и останов

Нормальное состояние процессора связано с выполнением инструкций. Инструкции (команды) выполняются с использованием трехуровневого конвейера.

Состояние обработки исключения связано с прерываниями, которые генерируются как внутренними устройствами DSP, так и поступают от внешних источников.

Состояние сброса возникает при появлении сигнала активного уровня на контакте RESET. В этом состоянии происходит сброс внутренних периферийных устройств, регистры модификации устанавливаются в $FFFF, регистр уровней приоритета прерываний очищается, регистр BCR устанавливается в $FFFF, что добавляет задержку к состоянию ожидания при всех обращениях к внешней памяти, указатель стека очищается, биты режима масштабирования, режима трассировки, флаг цикла и биты условий в SR сбрасываются, биты маски прерываний в SR устанавливаются в «1»; бит разрешения ПЗУ данных, бит задержки и бит стробирования памяти сбрасываются. DSP не выходит из состояния сброса, пока на контакте RESET сохраняется сигнал активного уровня. После выхода из состояния сброса биты операционного режима в OMR загружаются извне по линиям MODA, MODB и выполнение программы начинается с адреса $E000 в памяти программ в нормальном расширенном режиме или с адреса $0000 в остальных операционных режимах.

Состояние ожидания - это состояние пониженного энергопотребления, в которое процессор переходит при выполнении инструкции WAIT. В состоянии ожидания запрещена внутренняя синхронизация всех устройств на кристалле, за исключением внутренней периферии (генератор опорной частоты работает). Все внутренние процессы остановлены до тех пор, пока не поступит немаскируемое прерывание или не будет произведен сброс.

Состояние останова - это состояние самого низкого энергопотребления, в которое процессор переходит при выполнении инструкции STOP. В этом состоянии генератор опорной частоты отключен. При переходе в состояние останова кристалл сбрасывает все прерывания от периферийных устройств и внешние прерывания. Уровни приоритетов прерываний остаются такими же, как перед выполнением инструкции STOP. Процессор остановлен до тех пор, пока не появится сигнал низкого логического уровня на контакте ¬IRQA или на контакте ¬RESET. Эти события включают генератор, и после стабилизации частоты включается синхронизация всех устройств на кристалле. Период стабилизации частоты определяется битом SD в OMR.

8.4.2 Порты ввода/вывода

Порт А. Порт А может использоваться для расширения памяти или обычного ввода/вывода. Шина данных порта А имеет разрядность 24 бита, шина адреса разрядностью 16 бит обеспечивает доступ к памяти за один машинный цикл (рисунок 8.18).

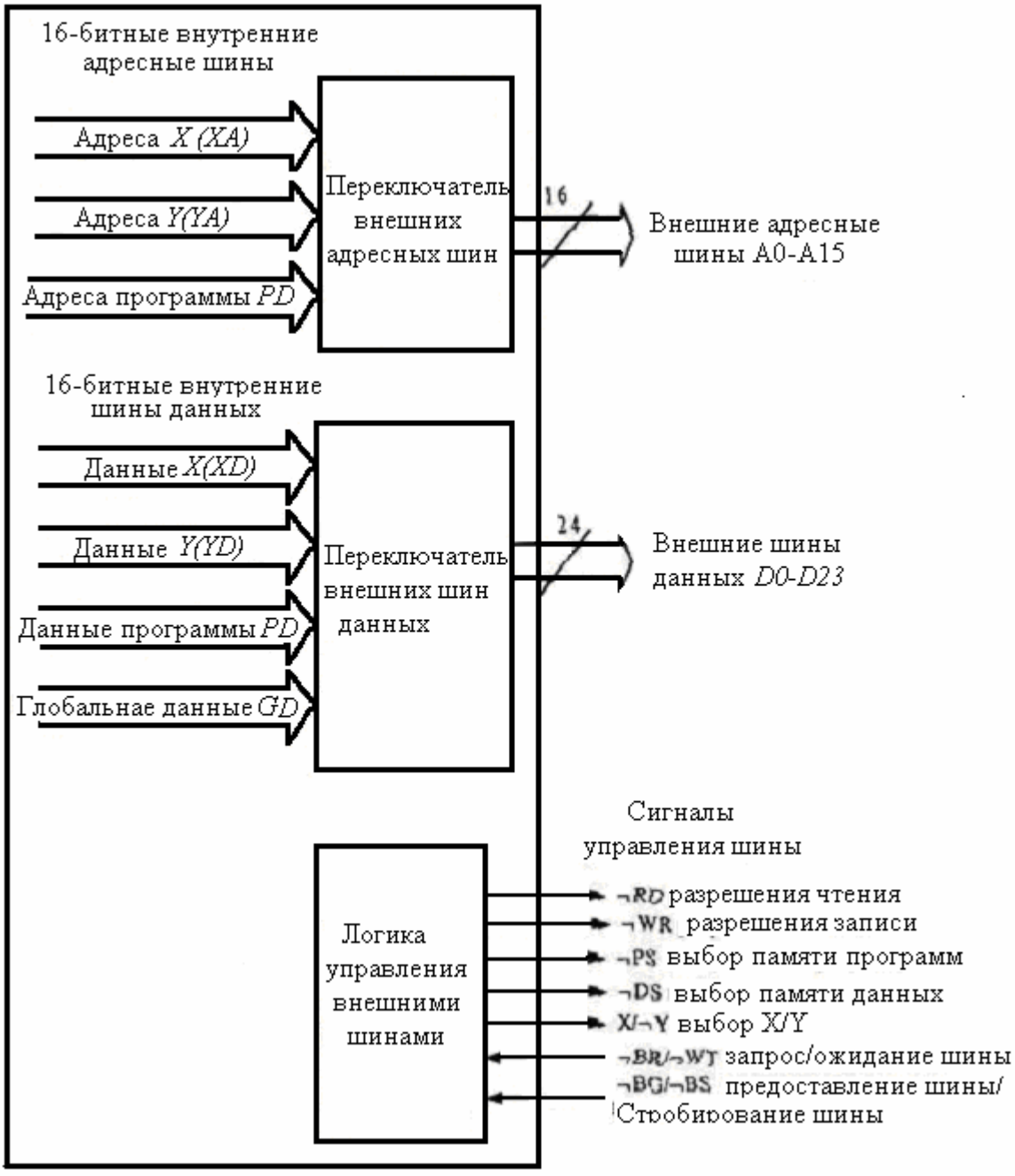


Рисунок 8.18 - Структура порта А

Во время выполнения инструкции могут быть доступны следующие блоки памяти DSP: X-память данных, Y-память данных, память программ или несколько блоков сразу. Каждый из этих блоков памяти может быть внутренним или внешним. Шины данных и адреса порта А

обеспечивают доступ к внешней памяти.

На рисунке 8.18 показаны сигналы порта А. Все сигналы разделены на три функциональные группы: адресные, данных и управления.

Сигналы шины управления также подразделяются на три группы: управление чтением и записью, выбор адресного пространства, управление доступом к шине. Сигналы управления чтением и записью могут быть декодированы как сигналы чтения и сигналы записи или сигнал записи может использоваться в качестве сигнала чтения/записи, сигнал чтения - в качестве выхода разрешения доступа к памяти. Сигналы выбора памяти программ, памяти данных и типа памяти данных могут рассматриваться как дополнительные адресные линии, позволяющие расширять пространство адресуемой памяти от 64К до 192К слов.

Порт В. Порт В может использоваться как 15 линий ввода/вывода общего назначения, каждая из которых конфигурируется индивидуально как вход или выход или как 8-битный двунаправленный host-интерфейс. Линии порта В представлены на рисунке 8.19. Сконфигурированный как порт ввода/вывода общего назначения, порт В может использоваться для управления внешними устройствами. В качестве host-интерфейса порт В обеспечивает соединение с другим процессором.

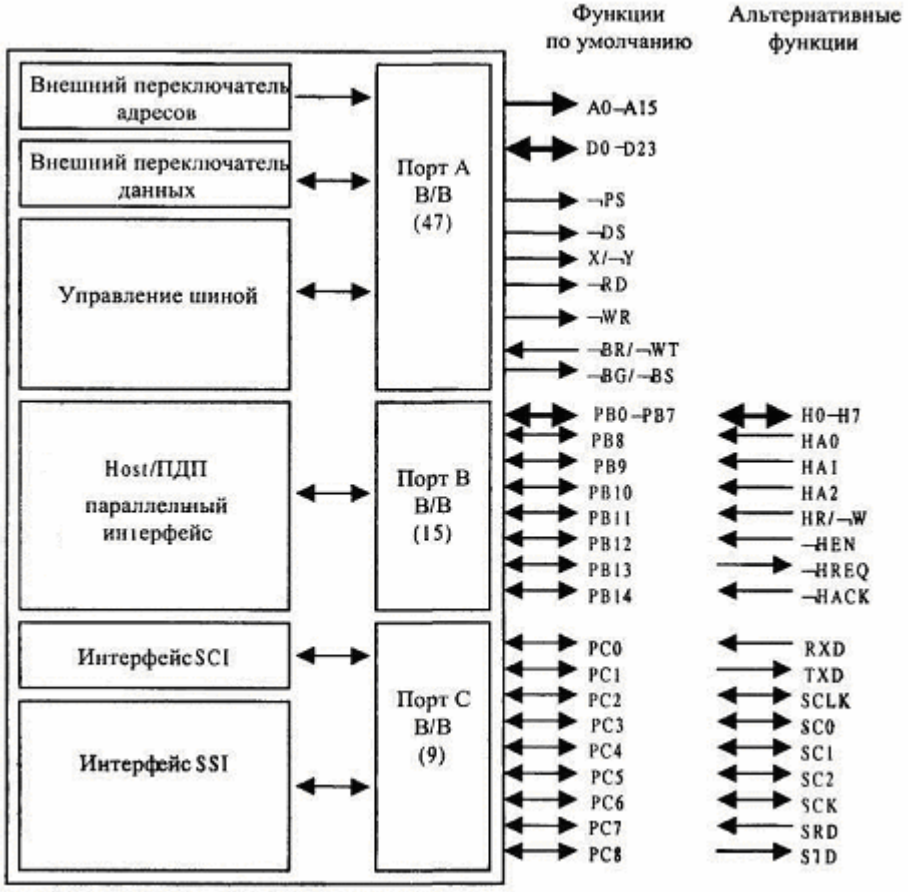


Рисунок 8.19 - Структура порта В

Порт С. Порт С имеет девять линий и три функциональных назначения (рисунок 8.20). Три из девяти линий могут использоваться для стандартного ввода/вывода или для последовательного коммуникационного интерфейса SCI, другие шесть линий могут использоваться для стандартного ввода/вывода или для синхронного последовательного интерфейса SSI.

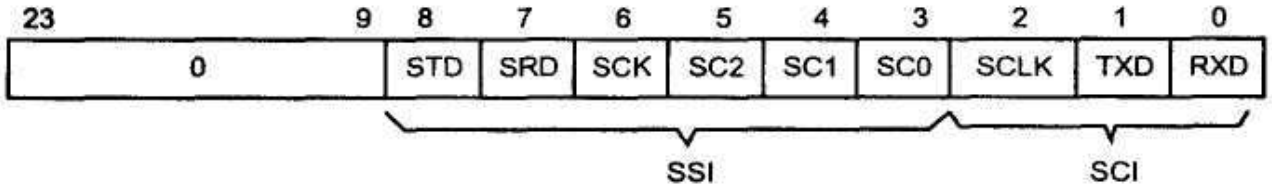


Рисунок 8.20 - Назначение линий порта С

Таким образом, порт С может использоваться для управления устройствами, если он сконфигурирован как порт стандартного ввода/вывода, и для соединения с другими DSP, процессорами, АЦП и ЦАП, если он сконфигурирован как последовательный интерфейс.

При использовании в качестве линий стандартного ввода/вывода порт С может быть представлен в виде трех регистров, которые управляют девятью контактами ввода/вывода. Это регистр управления порта С (PCC, адрес X.$FFE1), регистр направления данных порта С (PCDDR, адрес X.$FFE3) и регистр данных порта С (PCD, адрес X.FFE5).

Регистры PCDDR и PCD имеют разрядность 24 бита, из которых используются только младшие девять битов. Соответственно, установка в «1» бита в регистре PCDDR означает, что данная линия порта С используется для вывода, в противном случае - для ввода.

Выбор между стандартным вводом/выводом и последовательными интерфейсами осуществляется с помощью регистра PCC, в котором установка соответствующего бита в «1» означает использование последовательного интерфейса, а сброс в «0» - стандартный ввод/вывод.

Последовательный коммуникационный интерфейс SCI обеспечивает полнодуплексную последовательную связь с другими DSP, микропроцессорами или периферийными устройствами типа модемов. Сигналы интерфейса могут иметь ТТЛ-уровень или стандарт RS232C, RS422 и т. д. Этот интерфейс использует три линии: передачи данных (TXD), приема данных (RXD) и синхронизации (SCLK).

Прием данных (RXD - бит 0). Этот вход принимает байт последовательных данных и передает их в регистр сдвига. Асинхронный ввод данных осуществляется по положительному фронту сигнала синхронизации 1 x SCLK, если SCKP = 0.

Передача данных (TXD- бит 1). На этот выход передаются последовательные данные из регистра сдвига. Данные меняются по отрицательному фронту сигнала синхронизации SCLK, если SCKP = 0.

Синхронизация (SCLK— бит 2). Двунаправленная линия обеспечивает синхронизацию при приеме или передаче данных в асинхронном режиме, а также при передаче данных в синхронном режиме.

Программная модель SCI. Программная модель SCI приведена на рисунке 8.21. Программная модель может быть представлена в виде регистров трех типов: 1) управления (SCR); 2) статуса (SSR); 3) управления синхронизацией (SCCR).

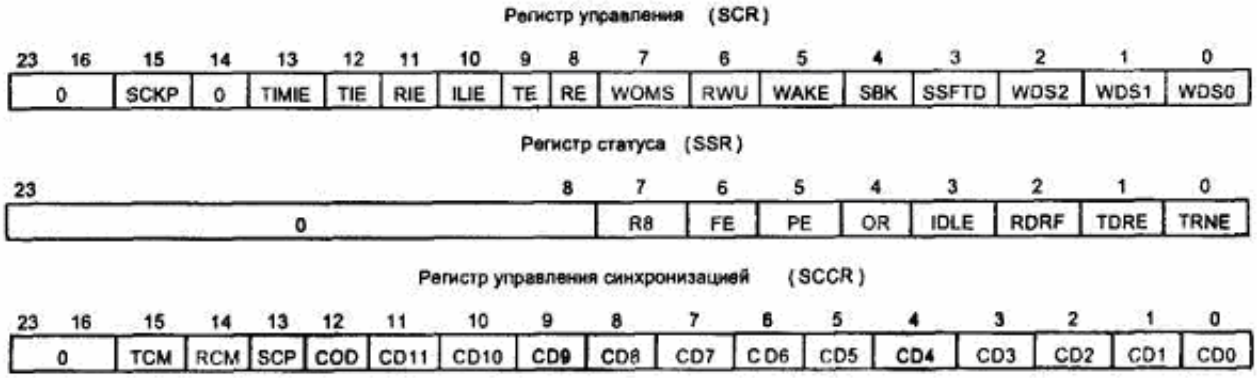


Рисунок 8.21 - Программная модель SCI

Регистр управления (SCR). 16-битный регистр чтения/записи, управляющий операциями последовательного интерфейса. Младшие пятнадцать из 16 битов имеют функциональное назначение (рисунок 8.21). Выбор слова осуществляется битами WDSO, WDS1, WDS2. Эти биты выбирают формат данных для передачи и приема в соответствии с таблицей 8.5.

Таблица 8.5 - Выбор формата данных

WDS2	WDS1	WDS0	Формат слова
0	0	0	8-битные синхронные данные (режим сдвигового регистра)
0	0	1	Резерв
0	1	0	10-битные асинхронные данные (старт-бит, 8 бит данных, стоп-бит)
0	1	1	Резерв
1	0	0	11-битные асинхронные данные (старт-бит, 8 бит данных, бит четности, стоп-бит)
1	0	1	11-битные асинхронные данные (старт-бит, 8 бит данных, бит нечетности, стоп-бит)
1	1	0	11-битные мультиточечные данные (старт-бит, 8 бит данных, бит типа данных, стоп-бит)
1	1	1	Резерв

Асинхронные режимы совместимы с большинством последовательных устройств типа DUART. Эти режимы поддерживают стандарт RS-232. Мультиточечный режим совместим с MC68681 DUART, интерфейсом SCI M68HC11 и последовательным интерфейсом Intel 8051. Синхронный режим, по существу, представляет собой сдвиговый регистр для расширения ввода/вывода. Синхронизация данных выполняется с использованием синхронизации приема и передачи, которая совместима с режимом 0 последовательного интерфейса Intel 8051. При аппаратном сбросе биты очищаются.

Направление сдвига определяется битом SSFTD. Регистр сдвига данных может быть запрограммирован на сдвиг битов, начиная с младших (SSFTD = 0) или - со старших битов (SSFTD = 1). Местоположение битов четности и типа данных не изменяется - они находятся рядом со стоп-битом. Программный и аппаратный сбросы очищают этот бит.

Посылка маркера паузы (SBK- бит 4). Маркер паузы представляет собой нулевой фрейм данных. Если бит установлен и затем очищен, передатчик завершает передачу данных, посылает нулевой фрейм и возвращается к холостому режиму или посылке данных. Программный и аппаратный сбросы очищают этот бит.

Выбор режима пробуждения (WAKE- бит 5). Если бит равен нулю, выбирается режим пробуждения по холостой линии (свободной от выполнения каких-либо действий). В данном режиме приемнику повторно разрешен прием данных. Программное обеспечение передатчика должно обеспечить формирование пустой (холостой) строки между последовательными сообщениями. Холостая строка не должна устанавливаться между корректными сообщениями, поскольку каждый фрейм содержит старт-бит, равный нулю.

Если WAKE = 1, выбирается режим пробуждения по биту адреса. В этом режиме приемнику разрешено принимать данные, если последний бит принимаемых данных (8 или 9) равен единице.

Последовательный аудиоинтерфейс (SAI). DSP осуществляет объединение источников и приемников данных через последовательный аудиоинтерфейс (Serial Audio Interface -SAI). SAI является синхронным и специально предназначен для передачи аудиоданных. Он обеспечивает взаимодействие полнодуплексного последовательного порта с различными аудиоустройствами, в том числе такими как АЦП, ЦАП, устройствами компакт-дисков и т. д. SAI реализует широкий диапазон форматов последовательных данных, используемых в настоящее время в аудио-промышленности:

- формат I^2S (Philips);
- формат CDP (Sony);
- формат MEC (Matsushita);
- промышленные А/Ц- и Ц/А-стандарты.

SAI состоит из независимых секций приемника, передатчика и разделяемого (совместно используемого) генератора скорости передачи данных. Каждая из секций приемника и передатчика может работать в режиме ведомого или ведущего. В режиме ведущего в соответствии с программой работы генератора скорости передачи данных производится внутреннее управление последовательным тактовым сигналом и выбором линий данных. В режиме ведомого от внешнего источника передаются три вида сигналов. Передатчик состоит из трех регистров передаваемых данных, трех полностью синхронизируемых регистров сдвига на выходе и трех линий выхода последовательных данных, управляемых одним контроллером передатчика. Это обеспечивает возможность осуществления одновременной передачи для одного, двух или трех стерео-, аудиоустройств. Приемник состоит из двух регистров принимаемых данных, двух полностью синхронизируемых регистров сдвига на входе и двух линий входа последовательных данных, управляемых одним контроллером приемника. Это обеспечивает одновременный прием из одного или двух аудиоустройств.

8.5 Специализированная цифровая вычислительная машина «Багет-55»

Специализированная цифровая вычислительная машина (СЦВМ) «Багет-55» представляет собой комбинированное устройство, включающее высокопроизводительный программируемый процессор сигналов (ППС) с возможностью доукомплектования специализированным вычислительным устройством (СВУ) /17/.

СЦВМ предназначена для использования в качестве вычислителя для многоканальной цифровой обработки сигналов в реальном масштабе времени в составе авиационных бортовых систем различного назначения, а также для использования в составе систем управления различного класса.

СЦВМ предназначена для эксплуатации в следующих климатических условиях при охлаждении её воздухом с температурой от минус 10° до плюс 23° С, очищенным от капельной влаги, масла и механических частиц и расходе воздуха не менее 100 кг/ч:

1) пониженная рабочая температура - от минус 55° С;

2) повышенная рабочая температура - до плюс 60° С;

3) относительная влажность воздуха окружающей среды - не более 98% при температуре не более плюс 35° С;

4) пониженное атмосферное давление - до 5,47 КПа (41мм. рт. ст.).

Работоспособность изделия сохраняется после воздействия предельных температур: минус 60°, плюс 85°С, а также в условиях воздействия соляного (морского) тумана, атмосферных конденсированных осадков - росы и внутреннего оледенения, статической пыли, плесневых грибов и акустического шума.

Основные технические параметры и технические характеристики СЦВМ Багет-55-04 /30/:

тип основного микропроцессора ……1В577 (аналог DSP96002);

тип дополнительного микропроцессора……. 1В578;

разрядность обрабатываемых данных:

- фиксированная запятая, бит ……. 32;
- плавающая запятая, бит …… 32/64;

тактовая частота, МГц …….. до 40;

внешняя тактовая частота, МГц …….. до 10;

внешние интерфейсы ввода/вывода:

- последовательный интерфейс типа RS232 2 шт.;
- интерфейс дискретных сигналов: 1 шт.;
- количество входных линий: 4;
- количество выходных линий: 4;
- тип логики сигнала: ТТЛ;
- специализированный 32-разрядный параллельный интерфейс приема информации с выхода АЦП: 1 шт.;

код приема данных по специализированному параллельному интерфейсу …… дополнительный;

максимальный темп приема данных по специализированному параллельному интерфейсу, Мслов/с……. 20;

количество модулей, устанавливаемых в корпус СЦВМ…… 10;

тип входной сети питания…… 200/220 В, 50-400 Гц

потребляемая мощность, не более, Вт …….. 60;

габаритные размеры, не более, мм :

- ширина …… 257;
- высота …….. 197,5;
- длина (без ловителей) …… 376;

- длина (с ловителями) ……. 405;

вес, не более, кг …….. ..15,2 .

СЦВМ Багет-55-04 включает в себя следующие составные части:

- программируемый процессор сигналов (DSP);
- корпус с установленными в нем модулем электропитания и кросс-платой;
- внутренние соединительные кабели.

В состав ППС изделия должны входить следующие функциональные модули:

- модуль обработки сигналов;
- модуль процессора данных;
- модуль приема и буферизации.

Модули, кабели, источник питания, объединительная панель (кросс-плата) размещены в герметизированном корпусе. Для потребителя доступны разъемы на передней панели корпуса, на которые выведены все интерфейсные сигналы и сигналы управления СЦВМ.

С целью расширения функциональных возможностей СЦВМ и ее исполнение могут быть доукомплектованы специализированным вычислительным устройством (СВУ), имеющим общий канал обмена данными с ППС и состоящим из функциональных модулей центрального процессора (ЦП).

СВУ СЦВМ имеет следующие технические характеристики:

- центральный процессор типа 1В578;
- системная шина VME;
- разрядность системной шины - 32 разряда;
- разрядность обрабатываемых данных:
  целые числа - 8,16,32 разряда;
  числа с плавающей запятой - 32 разряда;
- тактовая частота не менее 50 МГц;
- емкость ОЗУ не менее 16 Мбайт;
- электрически стираемая программируемая постоянная память EEPROM (flash type) емкостью не менее 32 Мбайт;
- количество таймеров не менее трех.

При необходимости использования дополнительных интерфейсов взаимодействия СЦВМ с внешними устройствами объекта может быть произведено доукомплектование платами мезонинными в части ППС и СВУ.

Все электронные устройства СЦВМ размещаются в корпусе, защищающем их в процессе эксплуатации от механических повреждений

других внешних воздействующих факторов.

В нижней части корпуса установлена панель объединительная, при помощи которой организовано объединение внутренних магистралей модулей, входящих в состав СЦВМ, и преобразование входных сигналов СЦВМ, поступающих в виде парафазных сигналов, в сигналы вида ТТЛ.

В передней части объединительной платы установлены разъемы, через которые внешние интерфейсы СЦВМ с помощью внутренних кабелей выводятся на переднюю панель корпуса. На внутренней стороне задней стенки корпуса установлен источник электропитания, формирующий вторичные напряжения + 5В,+12В, и сигналы управления питанием. Связь источника электропитания с объединительной платой осуществляется через два 15-контактных разъема.

Доступ к объединительной плате осуществляется через съемную нижнюю крышку, доступ к модулям - через съемную верхнюю крышку корпуса.

С внешней стороны задней стенки корпуса имеются резьбовые отверстия для присоединения воздуховодов внешней системы вентиляции или для установки блока вентиляторов.

Структурная схема СЦВМ приведена на рисунке 8.22.

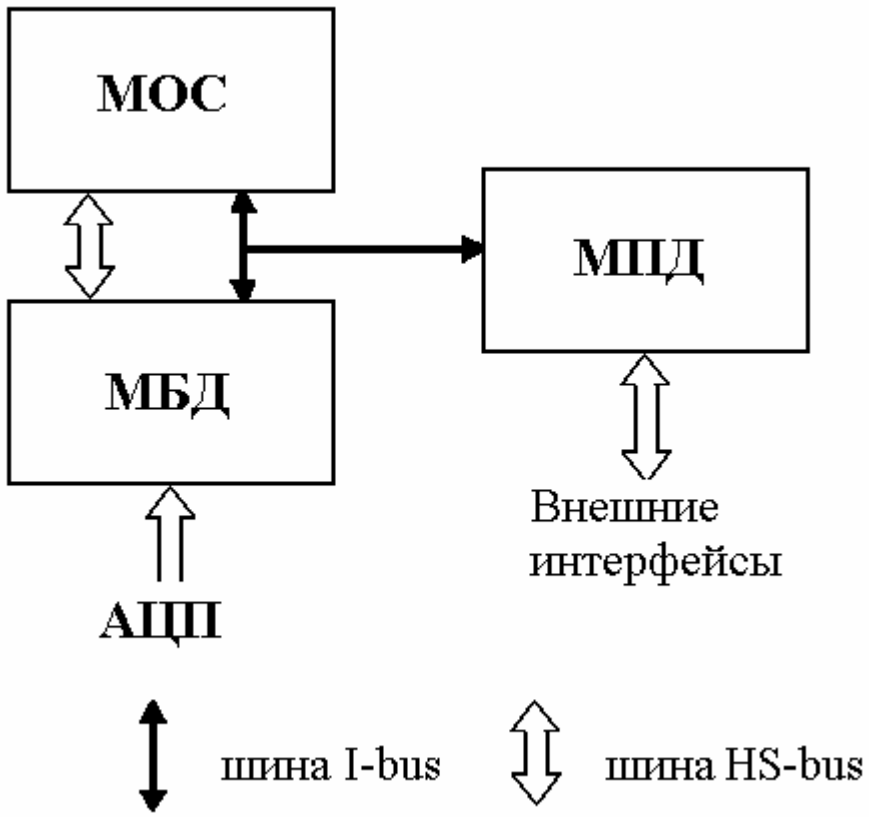


Рисунок 8.22 – Структурная схема СЦВМ

СЦВМ может функционировать в двух режимах - штатном и технологическом.

В штатном режиме работы обеспечивается:

- начальная инициализация СЦВМ по включению питания;
- определение режима начального тестирования по команде пользователя;
- начальное тестирование СЦВМ в соответствии с выбранным режимом;
- выдача результатов тестирования в память;
- передача управления программе пользователя, размещаемой в репрограммируемом постоянном запоминающем устройстве (РПЗУ).

В технологическом режиме работы обеспечивается:

- начальная инициализация СЦВМ по включению питания;
- определение режима начального тестирования по команде пользователя;
- начальное тестирование СЦВМ в соответствии с выбранным режимом;
- выдача результатов тестирования в память;
- дистанционная загрузка программ пользователя с инструментальной ПЭВМ;
- запуск на исполнение программы пользователя;
- запись отлаженного рабочего программного обеспечения в РПЗУ.

Модуль обработки сигналов (МОС). Модуль обработки сигналов представляет собой вычислительное устройство с симметричной внутренней архитектурой, содержащее четыре сигнальных микропроцессора, и является основной вычислительной единицей для программируемых процессоров сигналов.

МОС предназначен для обработки потоков информации, поступающих по I-шине от буфера данных или других МОС в реальном масштабе времени, под управлением модуля процессора данных.

МОС содержит 4 сигнальных процессора 1В577 (DSP-процессора), работающих на частотах до 40.0 МГц, и имеет пиковую производительность 160 MIPS или 240 MFLOPS. Структура модуля приведена на рисунке 8.23.

В состав МОС входят следующие функциональные блоки:

- вычислительные элементы ВЭ_А - ВЭ_D, содержащие процессор 1В577, локальную статическую память (SRAM) объемом 2Мб, локальное запоминающее устройство для организации очередей типа

«первым пришел – первым вышел» (first in – first out FIFO) емкостью 8К х 32-разрядных слов и логику управления;

- общая статическая память объемом 2 Мб;
- загрузочное ПЗУ объемом не менее 32 Кб (PROM);
- выходное FIFO, расположенное на общей шине (M-bus) процессоров;
- контроллер HS-шины (HS-CTRL);
- многофункциональный контроллер (MF-CTRL);
- энергонезависимое последовательное ЗУ объемом 2Кбит (EEPROM);
- контроллер I-шины (I-CTRL).

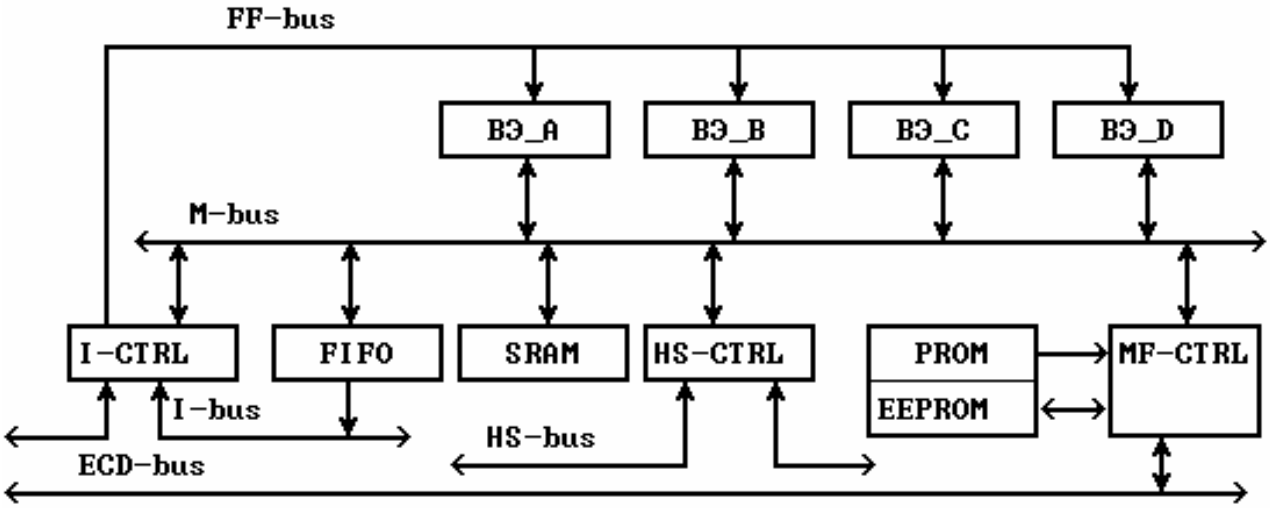


Рисунок 8.23 – Структура модуля МОС

К внутренним шинам модуля относятся:

- M-шина - общая шина сигнальных процессоров;
- Lx-шины - локальные шины ВЭ_A...ВЭ_D;
- FF-шина - входная шина данных локальных FIFO.

К внешним шинам относятся:

- I-шина - шина приема/передачи данных от АЦП, может использоваться для локального межмодульного обмена данными;
- HS-шина - шина межмодульного обмена данными;
- ECD-шина - шина управления процессами в системе.

Модуль процессора данных (МПД) является комбинированным устройством и выполняет функции управляющего устройства в СЦВМ и обеспечивающего устройства при обмене данными СЦВМ с внешними системами.

Условно модуль можно разделить на две логические части:

субмодуль А и субмодуль Б. Субмодуль А реализован на базе

сигнального микропроцессора 1В577 и предназначен для управления работой СЦВМ (управление событиями, процедурами смены режимов, отладка и т.п.). Субмодуль Б реализован на базе однокристального RISC-микропроцессора 1В578-ОМП и обеспечивает сопряжение СЦВМ с внешними системами. Связь между этими двумя субмодулями осуществляется через двухпортовую память.

В состав МПД входят следующие функциональные узлы:

субмодуль А:

- микропроцессор 1В577, работающий на частотах до 40 МГц;
- локальная статическая память (Local SRAM) объемом 1Мб;
- загрузочное последовательное ППЗУ объемом 256 Кбит;
- глобальная статическая память (Global SRAM) объемом 1Мб;
- контроллер HS-шины (HS-CTRL);
- многофункциональный контроллер (Multy-CTRL);

субмодуль Б:

- RISC-микропроцессор 1В578-ОМП, работающий на частоте 50МГц;
- статическая память микропроцессора (SRAM) объемом 8Мб;
- загрузочное ППЗУ объемом 4 Мбит;
- флэш-память объемом до 16 Мб;
- буферная статическая память (Buffer SRAM) объемом 1Мб, доступная как со стороны процессора, так и со стороны магистрали VME;
- контроллер последовательных каналов обмена данными типа RS232;
- универсальный контроллер (Uni-CTRL);
- контроллер шины PCI (PCI-CTRL);
- контроллер шины VME (VME-CTRL);
- двухпортовая память объемом до 64К 32-разрядных слов.

Модуль имеет набор интерфейсов, обеспечивающих его сопряжение как с модулями СЦВМ, так и с внешними устройствами. Обмен данными с модулями СЦВМ осуществляется по шинам HS и ECD. Для реализации сопряжения с внешними устройствами модуль имеет два слота с интерфейсом шины PCI, в которые могут быть установлены стандартные PMC-мезонины с необходимыми интерфейсами. Помимо перечисленного, модуль МПД содержит два последовательных канала обмена данными, имеющие электрические интерфейсы RS-232 , а также набор дискретных сигналов, которые используются как коман-

ды управления, с уровнями ТТЛ (4 входных линии и 4 выходных линии).

Модуль приема и буферизации данных (МПД). Модуль приема и буферизации данных, или просто буфер данных (МБД), предназначен для приема данных, подлежащих обработке, от какого-либо внешнего устройства (АЦП),

МБД выполняет следующие функции:

- прием и буферизацию информации от внешних источников;
- присвоение идентификатора ("цвета" входному потоку данных;
- программируемую коммутацию потоков с разными идентификаторами;
- выдачу информации из данного МБД в последующий (мезонинная плата);
- выявление сбоев в работе входных каналов и формирование по ним события на ECD-шине;
- программно маскируемое формирование события на ECD-шине при поступлении сигналов "начало кадра" - ИНК.

Структурная схема МБД, приведена на рисунке 8.24.

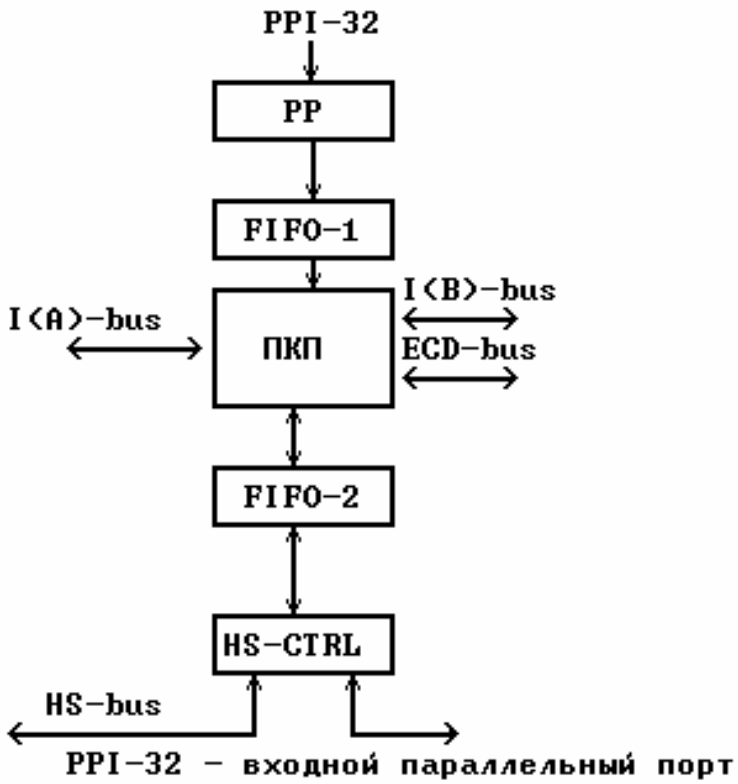


Рисунок 8.24 – Структурная схема модуля приема и буферизации данных

МБД содержит один 32-разрядный параллельный порт (PP-32), работающий только на прием данных от АЦП.

Модуль содержит следующие функциональные блоки:

- входной интерфейс параллельный порт;
- выходной интерфейс для мезонинной платы выходного порта;
- буферную память FIFO-1 глубиной 8К 32-разрядных слов на входе от АЦП;
- программируемый коммутатор потоков данных (ПКП) SWITCH;
- буферную память FIFO-2 глубиной 8К 32-разрядных слов на входе от HS-шины;
- контроллер HS-шины (HS-CTRL).

Модуль БД имеет многошинную (многоканальную) внутреннюю архитектуру, которая позволяет коммутировать шины между собой, обеспечивая тем самым разнообразное распределение потоков данных.

Максимальная пропускная способность внешнего выходного канала составляет 20 миллионов 32-разрядных слов / с (при системной частоте 40 МГц). Источниками данных для выходного канала могут выступать любые каналы, подключенные к программируемому коммутатору.

Таким образом, характеристики СЦВМ «Багет-55» обеспечивают эффективную цифровую обработку одно- и двумерных сигналов.

ЗАКЛЮЧЕНИЕ. ПЕРСПЕКТИВЫ РАЗВИТИЯ БОРТОВЫХ ВЫЧИСЛИТЕЛЬНЫХ МАШИН И СИСТЕМ

Перспективные направления развития вычислительных машин и систем можно разделить на следующие группы:

совершенствование элементной базы вычислительных машин классической архитектуры;

разработка новых архитектурных подходов к построению ЭВМ;

расширение области применения БВМ на основе новых алгоритмов и методов цифровой обработки информации;

применение новых физических принципов для выполнения обработки и хранения данных.

Перспективы развития БЦВС связаны с разработкой ЭВМ пятого поколения. Одна из основных целей разработки ЭВМ пятого поколения состоит в создании ЭВМ, рассчитанной на простое и удобное взаимодействие в режиме диалога с оператором (экипажем) на естественном языке. Другая особенность ЭВМ пятого поколения – наличие системы логического решения задач. Она обеспечит ЭВМ способность к ассоциативному поиску информации, логическим заключениям и самообучение. В идеальном случае эта ЭВМ должна уметь извлекать из общего массива заключенной в ней информации некоторые данные, связанные между собой различными признаками и необходимые для формирования логических заключений. ЭВМ пятого поколения будет реагировать даже на нечетко поставленные вопросы и требовать от человека их повторной более четкой формулировки или самостоятельно прояснять их либо с помощью имеющегося в ней огромного объема информации, либо в результате взаимодействия с другими ЭВМ.

В состав ЭВМ пятого поколения входит еще одна функциональная система - база знаний с соответствующей системой управления. На основе информации, полученной из базы знаний, в машине логического решения задач будет формироваться решение поставленной оператором проблемы, подвергаемое затем смысловому анализу. Цель такого анализа - устранение несущественной информации в полученном решении и выделении в нем фактов и правил, которые могут быть использованы в дальнейшем при решении других задач. Результаты, представляющие общий интерес, будут накапливаться в базе знаний ЭВМ.

Предполагается, что ЭВМ пятого поколения будут обладать

производительностью и функциональными возможностями, значительно превосходящими характеристики современных ЭВМ. Вычислительная система, использующая ЭВМ пятого поколения, будет представлять собой неоднородную сеть вычислительных машин, к которой могут быть подключены универсальные высокопроизводительные ЭВМ общего назначения, вычислительные машины малой производительности, а также специализированные вычислители, причем все они должны будут использовать параллельную обработку информации. Поэтому можно считать все ЭВМ пятого поколения малинами с параллельной архитектурой.

Другим перспективным направлением совершенствования БВМ является применение новых материалов и принципов обработки информации для построения аппаратной части вычислительных машин. Среди таких перспективных БВМ следует выделить оптические вычислительные машины и вычислительные машины на основе биотехнологий.

Оптические цифровые вычислительные машины. Кроме совершенствования технологии производства электронных схем и архитектуры вычислительных устройств в вычислительной технике будущего наметилось новое научно-техническое направление - цифровая оптическая вычислительная техника. Считается, что она позволит перейти от электронных вычислительных машин к оптическим цифровым вычислительным машинам (ОЦВМ), которые будут обладить неизмеримо большим быстродействием за счет использования фотонов в качестве носителя информации.

Основными достоинствами ОЦВМ являются: способность выполнять операции над двухмерными операндами (картинная логика); высокая скорость выполнения операций, определяемая только временем прохождения света от входа до выхода процессора; использование в составе средств обработки информации голографических (оптических) запоминающих устройств большой емкости и быстродействия; применение в качестве интерфейса волоконно-оптических каналов ввода-вывода информации и внутримашинных связей.

Главным блоком ОЦВМ является оптический комбинаторный логический матричный процессор, который выполняет арифметические и логические операции. С оптическим процессором совмещены оперативное запоминающее устройство и каналы ввода-вывода. Конструктивно все они размещены на одной большой

оптической двоичной вентильной матрице (кристалле), обладающей свойствами пространственности и параллельности при обработке двухмерной информации, ввод и вывод которой происходит параллельно информационными страницами.

В качестве соединительных каналов (связей) предполагается использовать оптические волокна, интегрально-оптические волноводы и синтезированные голограммы.

Оптические связи по сравнению с электрическими имеют ряд преимуществ: способность к перепрограммированию с помощью динамических оптических компонентов, отсутствие влияния емкостных нагрузок и взаимных помех, гибкость связей и др.

Перепрограммируемость системы перестраиваемых связей позволяет организовать новые типы машинной архитектуры, что существенно повышает эффективность обработки информации.

Вычислительные машины на основе биотехнологий. В настоящее время ведутся исследования по созданию ЭВМ с использованием элементов нервной системы человека - так называемых нейрокомпьютеров, которые будут иметь датчики на биологической основе и исполнительные устройства на основе механизмов мышечного сокращения.

Благодаря успехам молекулярной биологии становится возможной разработка датчиков с заранее заданными свойствами, избирательной реакцией и очень высокой чувствительностью. Кроме того, возможно использование биосистем в качестве запоминающих устройств, в частности, активных биологических пленок, позволяющих получать бессеребряные фотоносители.

В сочетании с лазером, обеспечивающим быструю запись и стирание информации, биохромные материалы могут составить основу построения запоминающих устройств.

Полагают, что нейрокомпьютеры как аналоговые устройства на новом, более высоком уровне будут способны решать такие задачи, как исследование нелинейных систем, описываемых системами дифференциальных уравнений в частных производных, которые сейчас не под силу самым мощным современным ЭВМ.

Нейрокомпьютеры имеют молекулярный уровень организации, при котором каждая молекула белка или элементарная ячейка пленки представляют активный элемент среды, способный находиться в нескольких устойчивых состояниях и играть роль ячейки памяти. Например, пленка в 1 $см^2$ может содержать свыше 10^{12} активных

элементов.

Развитие вычислительной техников том числе и бортового применения, столь стремительно, что даже в недалеком будущем вполне вероятно появление совершенно новых аппаратных, архитектурных, схемотехнических, программных и иных решений.

Все содержание книги и особенно её последней главы, посвященной внедрению вопросов цифровой обработки информации в бортовое радиоэлектронное оборудование, убедительно подтверждает необходимость всестороннего (системного) рассмотрения вопросов повышения эффективности бортового РЭО.

Радиоинженер должен умело применять методы оптимизации сложных систем, широко используя достоинства цифровой обработки информации. Это предъявляет высокие требования к образованию современного радиоинженера и определяет актуальность изучения дисциплины «Основы бортовых вычислительных машин».

ЛИТЕРАТУРА

1. **Баженов, А.В.** Основы цифровой схемотехники. Ч.1. Теоретические основы цифровой обработки сигналов / А.В. Баженов, В.В. Закалюжный. Конспект лекций. - Ставрополь: СВАИУ, 1998.- 136 с.

2. **Баженов, А.В.** Авиационные вычислительные машины и системы. Ч.2. Бортовые цифровые вычислительные машины / А.В. Баженов, А.А. Федотов. Конспект лекций. - Ставрополь: СВАИУ, 1995.- 139 с.

3. **Шевцов, В.К.** Вычислительные машины авиационных радиоэлектронных систем. Выпуск 1. Пособие по изучению процессора бортовой цифровой ЭВМ / В.К. Шевцов. - Даугавпилс: ДВВАИУ, 1985.- 63 с.

4. **Микропроцессорная лаборатория Микролаб КР580** / Учебное пособие. – М.: Внешторгиздат, 1988.- 95 с.

5. **Дулепов, Е.Г.** Теоретические основы вычислительной техники. Учеб.пособие. – Иркутск: Изд-во Иркутского университета. 1997. – 264 с.

6. **Марков, С.** Цифровые сигнальные процессоры. Книга 1.- М.: фирма МИКРОАРТ, 1996. – 114 с.

7. **Гольденберг, Л.М.** Цифровая обработка сигналов: Справочник /Л.М.Гольденберг, Б.Д.Матюшкин, М.Н.Поляк. – М.: Радио и связь, 1985. – 312 с.

8. **Ярославский, Л.П.** Цифровая обработка сигналов в оптике и голографии: Введение в цифровую оптику. – М.: Радио и связь, 1987. – 296 с.

9. **Майоров, С.А.** Структура электронных вычислительных машин/ Майоров С.А., Новиков Г.И. – Л.: Машиностроение, 1979. – 384 с.

10. **Пухальский, Г.И.** Проектирование дискретных устройств на интегральных микросхемах: Справочник /Пухальский Г.И., Новосельцева Т.Я.- М.: Радио и связь, 1990.- 304 с.

11. **Алексеев, Ю.Я.** Помехозащищенность авиационных радиолокационных систем/ Ю.Я. Алексеев, В.Н. Антипов, В.А. Ефимов, В.И Меркулов, И.Н. Умнов / Под редакцией В.А. Ефимова.- М.: ВАТУ, 2001.-271 с.

12. **Проектирование** импульсных и цифровых устройств радиотехнических систем: Учеб. пособие./Под ред. Ю.М. Казаринова. - М.: Высш. шк. , 1985.- 319 с.

13. **Микропроцессорные** системы: Учебное пособие для вузов/ Е.К. Александров, Р.И. Грушвицкий, М.С. Куприянов и др. Под общ. Ред. Д.В. Пузанкова. – СПб.: Политехника, 2002.- 935 с.

14. **Современные** микроконтроллеры: Архитектура, средства проектирования, призеры применения, ресурсы сети Интернет. «Телесистемы». Под ред. Коршуна И.В.; Составление, пер. с англ. и литературная обработка Горбунова Б.Б. -М.: Издательство «Аким», 1998.-272 с.

15. **Микроконтроллеры** фирмы Analog Devices. http:// css.vstu.edu.ru/sprav.spr2/ analog.html

16. **Марков, С.** Цифровые сигнальные процессоры. Книга I.- М.: фирма МИКРОАРТ, 1996. – 114 с.

17. **Куприянов, М. С.** Цифровая обработка сигналов: процессоры, алгоритмы, средства проектирования / М. С. Куприянов, Б. Д. Матюшкин. - 2-е изд. , перераб. и доп. - СПб.: Политехника, 2002. - 592 с.

18. **Сикарев, А.А.** Микроэлектронные устройства формирования и обработки сложных сигналов/ А.А. Сикарев, О.Н. Лебедев. – М.: Радио и связь, 1983.- 216 с.

19. **Миронов, С.** Цифровая обработка радиолокационных сигналов на основе процессора Л1879ВМ1/С.Миронов, В.Дударев, А.Богатов. Электронный журнал. Электроника. Раздел радиолокация. Выпуск №3- 2003.

20. **СЦВМ «Багет-55-04».** Техническое описание и инструкция по эксплуатации.

ЛИТЕРАТУРА

1. **Баженов, А.В.** Основы цифровой схемотехники. Ч.1. Теоретические основы цифровой обработки сигналов / А.В. Баженов, В.В. Закалюжный. Конспект лекций. - Ставрополь: СВАИУ, 1998.- 136 с.
2. **Баженов, А.В.** Авиационные вычислительные машины и системы. Ч.2. Бортовые цифровые вычислительные машины / А.В. Баженов, А.А. Федотов. Конспект лекций. - Ставрополь: СВАИУ, 1995.- 139 с.
3. **Шевцов, В.К.** Вычислительные машины авиационных радиоэлектронных систем. Выпуск 1. Пособие по изучению процессора бортовой цифровой ЭВМ / В.К. Шевцов. - Даугавпилс: ДВВАИУ, 1985.- 63 с.
4. **Микропроцессорная лаборатория Микролаб КР580** / Учебное пособие. – М.: Внешторгиздат, 1988.- 95 с.
5. **Дулепов, Е.Г.** Теоретические основы вычислительной техники. Учеб.пособие. – Иркутск: Изд-во Иркутского университета. 1997. – 264 с.
6. **Марков, С.** Цифровые сигнальные процессоры. Книга 1.- М.: фирма МИКРОАРТ, 1996. – 114 с.
7. **Гольденберг, Л.М.** Цифровая обработка сигналов: Справочник /Л.М.Гольденберг, Б.Д.Матюшкин, М.Н.Поляк. – М.: Радио и связь, 1985. – 312 с.
8. **Ярославский, Л.П.** Цифровая обработка сигналов в оптике и голографии: Введение в цифровую оптику. – М.: Радио и связь, 1987. – 296 с.
9. **Майоров, С.А.** Структура электронных вычислительных машин/ Майоров С.А., Новиков Г.И. – Л.: Машиностроение, 1979. – 384 с.
10. **Пухальский, Г.И.** Проектирование дискретных устройств на интегральных микросхемах: Справочник /Пухальский Г.И., Новосельцева Т.Я.- М.: Радио и связь, 1990.- 304 с.
11. **Алексеев, Ю.Я.** Помехозащищенность авиационных радиолокационных систем/ Ю.Я. Алексеев, В.Н. Антипов, В.А. Ефимов, В.И Меркулов, И.Н. Умнов / Под редакцией В.А. Ефимова.-М.: ВАТУ, 2001.-271 с.

12. **Проектирование** импульсных и цифровых устройств радиотехнических систем: Учеб. пособие./Под ред. Ю.М. Казаринова. - М.: Высш. шк. , 1985.- 319 с.

13. **Микропроцессорные** системы: Учебное пособие для вузов/ Е.К. Александров, Р.И. Грушвицкий, М.С. Куприянов и др. Под общ. Ред. Д.В. Пузанкова. – СПб.: Политехника, 2002.- 935 с.

14. **Современные** микроконтроллеры: Архитектура, средства проектирования, призеры применения, ресурсы сети Интернет. «Телесистемы». Под ред. Коршуна И.В.; Составление, пер. с англ. и литературная обработка Горбунова Б.Б. -М.: Издательство «Аким», 1998.-272 с.

15. **Микроконтроллеры** фирмы Analog Devices. http:// css.vstu.edu.ru/sprav.spr2/ analog.html

16. **Марков, С.** Цифровые сигнальные процессоры. Книга I.- М.: фирма МИКРОАРТ, 1996. – 114 с.

17. **Куприянов, М. С.** Цифровая обработка сигналов: процессоры, алгоритмы, средства проектирования / М. С. Куприянов, Б. Д. Матюшкин. - 2-е изд. , перераб. и доп. - СПб.: Политехника, 2002. - 592 с.

18. **Сикарев, А.А.** Микроэлектронные устройства формирования и обработки сложных сигналов/ А.А. Сикарев, О.Н. Лебедев. – М.: Радио и связь, 1983.- 216 с.

19. **Миронов, С.** Цифровая обработка радиолокационных сигналов на основе процессора Л1879ВМ1/С.Миронов, В.Дударев, А.Богатов. Электронный журнал. Электроника. Раздел радиолокация. Выпуск №3- 2003.

20. **СЦВМ «Багет-55-04».** Техническое описание и инструкция по эксплуатации.