Основные структуры запоминающих устройств

Понятие структуры памяти

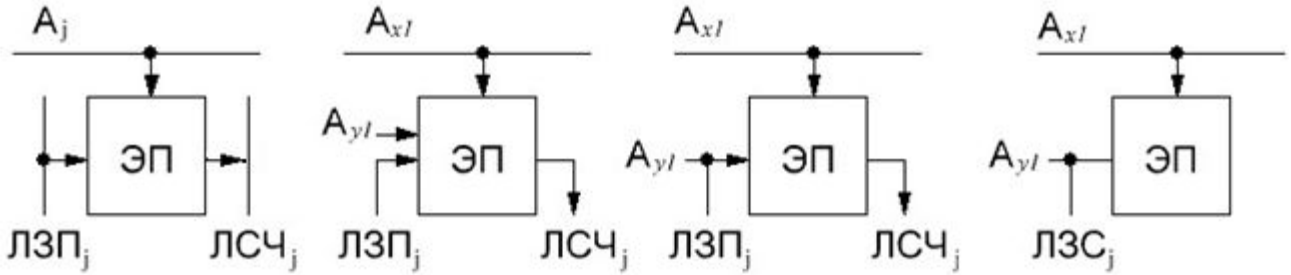
Ячейки памяти (ЯП) обычно организованы в виде *матричной структуры*. Каждая матрица **М** в устройстве памяти имеет систему *адресных* и *разрядных* линий (проводников) для работы с данными.

Адресные (словарные) **линии** служат для выделения по адресу любой ЯП. Совокупность различных адресных кодов образует **адресное пространство памяти**.

Разрядные **линии записи** (ЛЗП) служат для ввода в каждый разряд выбранной ЯП цифры 0 или 1 в соответствии с входной информацией.

Разрядные **линии считывания** (ЛСЧ) служат для съема хранимой информации с разряда выбранной ЯП. Часто используют общую **линию записи-считывания** (ЛЗС).

Адресные и разрядные линии вместе называются **линиями выборки**. Если длина адресного кода (количество разрядов) равна ***N***, то количество слов ***K***, которые хранятся в памяти как отдельные единицы данных, определяется из соотношения $K = 2^N$.



а) *б*) *в*) *г*)

Рисунок 1 − Обобщенное понятие структуры памяти: а – 2D; б – 3D; в *– 2,5D; г – 2DM*

Структуру памяти (рис. 1) определяет способ распределения *ячеек памяти* (ЯП) между *адресными* и *разрядными* линиями. По этому признаку выделяют следующие структуры памяти: **2D, 3D, 2,5D** *(D* от *Dimention* – размерность) и модифицированную – **2DM.**

Организация микросхемы ОЗУ со структурой типа 2D

Организация микросхемы ОЗУ со структурой типа 2D показана на рисунке 2.

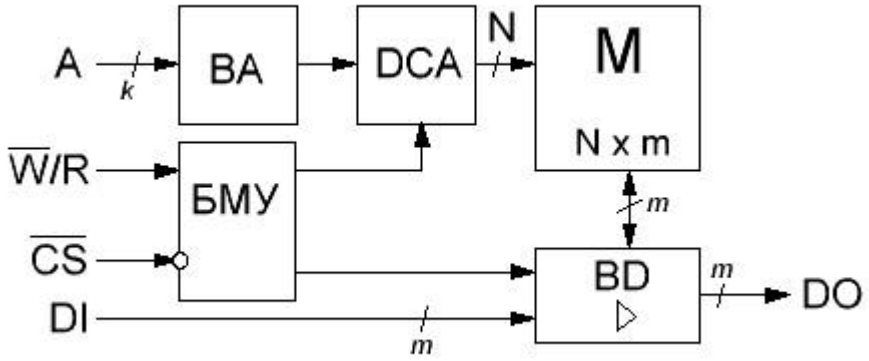


Рисунок 2 − Память со структурой 2D

В состав микросхемы памяти входят:

- *матрица элементов* памяти ***M,*** которая содержит ***N*** строк и ***m*** столбцов (по числу разрядов слова);
- *буфер адреса* ***BA*** и *дешифратор адреса* ***DCA*** адреса с числом выходов $N = 2^k$;
- *буферные формирователи* ***BD*** входных ***DI*** и выходных ***DO*** *данных* информационных сигналов в режимах записи и считывания*;*
- *блок местного управления* (БМУ).

При обращении к памяти выбираются ЯП, расположенные на выбранном выходе дешифратора адреса *DCA.*

Запись данных (***W****rite*) осуществляется при значении сигнала *W/R* = 0, а считывание (***R****ead*) при *W*/*R* = 1. Емкость памяти *2D* равна $E = N \cdot m$ бит.

Организация матрицы М памяти со структурой типа 2D

В структуре матрицы 2D (рисунок 3) запоминающие элементы или ***элементы памяти*** (ЗЭ) организованы в прямоугольную матрицу размерностью $M = k \cdot m$, где **М** – информационная емкость памяти в битах; ***k*** – число хранимых слов; ***m*** – их разрядность.

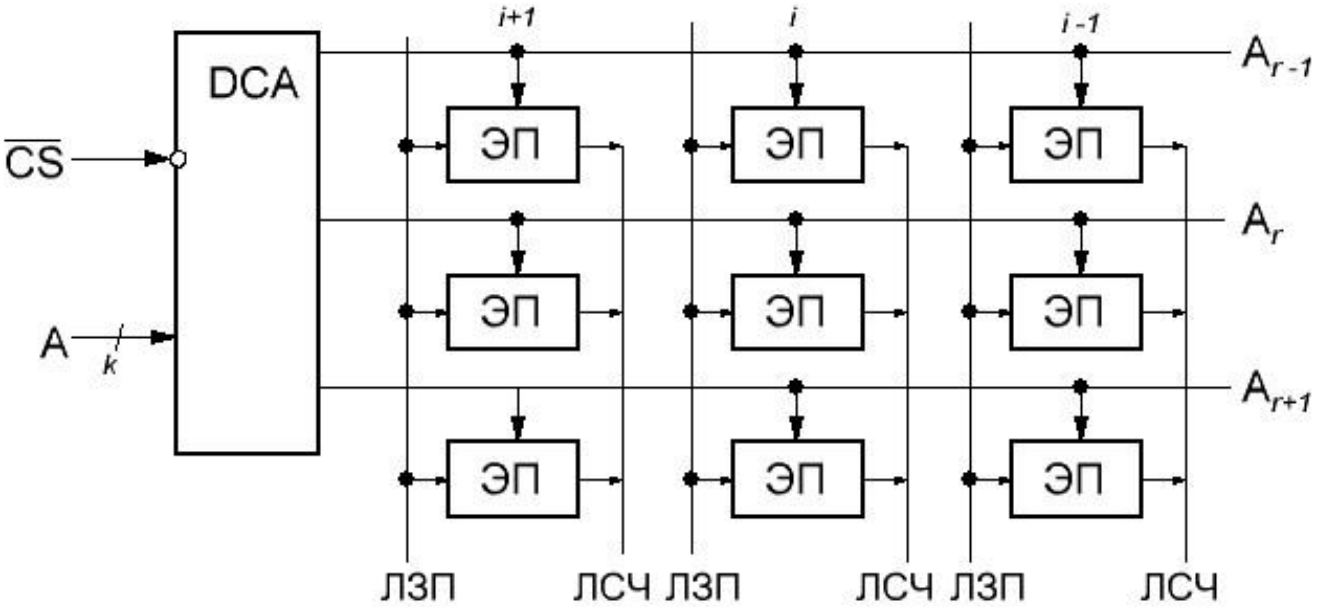


Рисунок 3 − Организация матрицы памяти М со структурой типа 2D

Недостатком структуры типа 2D является сложность построения дешифратора адреса с числом выходов *N,* равным числу хранимых в памяти слов. Поэтому структура типа *2D* используется в ЗУ малой информационной емкости.

Память со структурой типа 3D

В памяти со структурой типа 3D адресный код разделяется на две равные части – ***Ax*** и ***Ay*** (для четного ***k***), каждая из которых декодируется отдельными дешифраторами адреса соответственно ***DCX*** и ***DCY*** (рисунок 4).

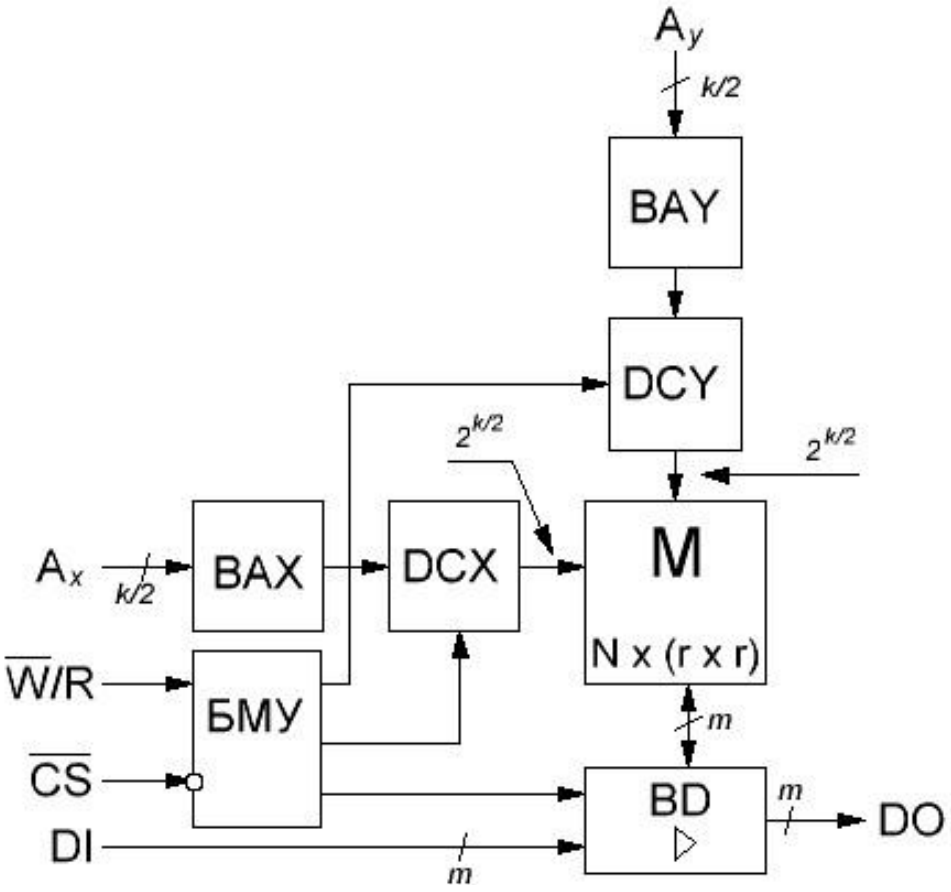


Рисунок 4 − *Память со структурой типа 3D*

Организация матрицы М памяти со структурой типа 3D

Матрица **М** состоит из ***m*** подматриц по числу разрядов слова. Каждая матрица хранит значение своего *i*-го разряда всех ***N*** слов. Каждая подматрица является квадратной: ***r*** строк и ***r*** столбцов, при этом $r = N^{1/2} = 2^{k/2}$ (рисунок 5).

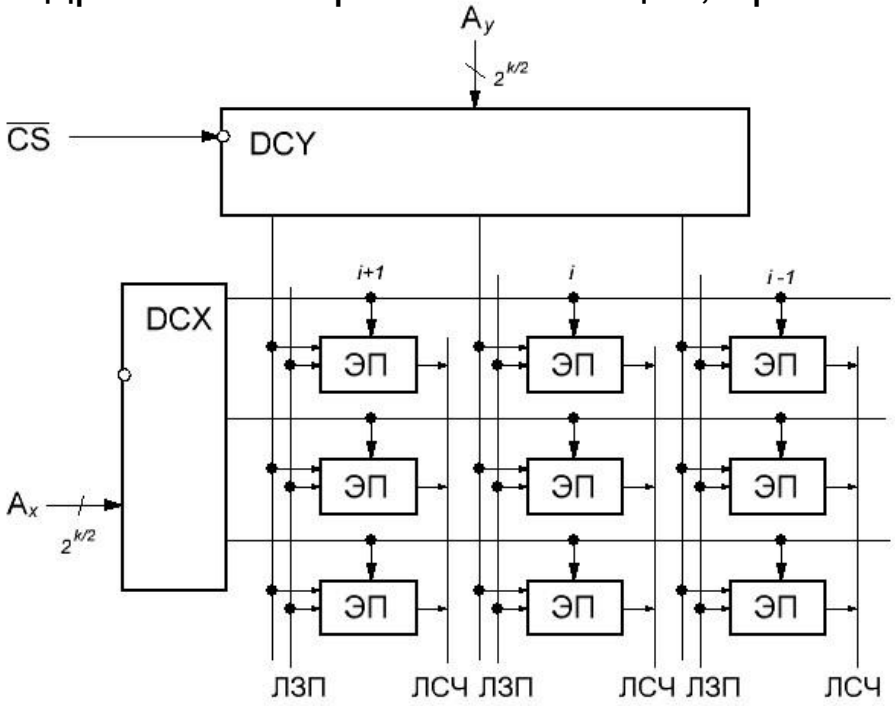


Рисунок 5 − Организация матрицы памяти М со структурой типа 3D с одноразрядной организацией

Здесь **код адреса** разрядностью ***k*** делится на две половины, каждая из которых декодируется отдельно. Выбирается запоминающий элемент – элемент памяти, находящийся на пересечении активных (выбранных) линий выходов обоих дешифраторов. Таких пересечений будет:

$$2^{k/2} \cdot 2^{k/2} = 2^{k}$$

Структура памяти типа 3D позволяет резко упростить дешифраторы адреса с помощью *двухкоординатной выборки* запоминающих элементов. Такая структура часто используется и с одноразрядной организацией *N* x 1 бит. Например, для памяти емкостью 1К слов (1024 слов = 2^{10}) требуется иметь два дешифратора с числом выходов в каждом $N = 2^5 = 32$. Для памяти со структурой 2D такой же емкости дешифратор имеет 1024 выхода.

Структура типа 3D, показанная на рисунке 5 может применяться и в ЗУ с многоразрядной организацией как показано на рисунке 6, приобретая при этом "трехмерный" характер. В этом случае несколько матриц управляются от двух дешифраторов, относительно которых они включены параллельно. Каждая матрица выдает один бит адресованного слова данных, а число матриц равно разрядности хранимых слов.

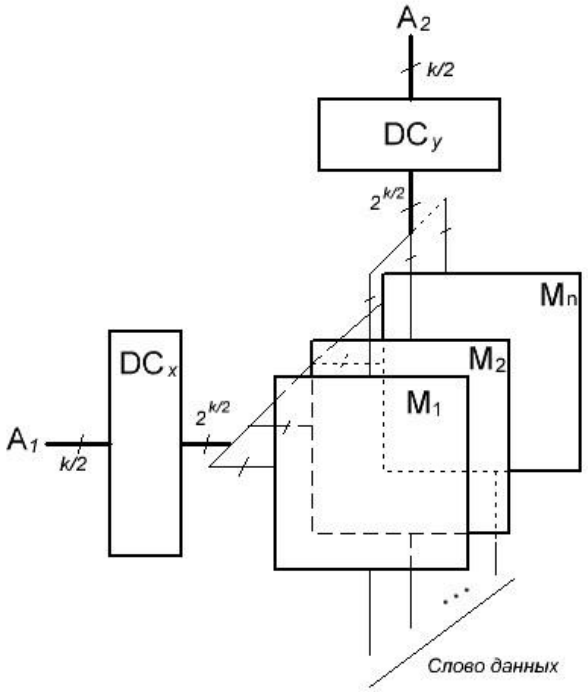


Рисунок 6 − Структура ЗУ типа 3D с многоразрядной организацией.

Недостатком структуры типа 3D является применение сложных ЭП, допускающих двухкоординатную выборку и более сложную структуру матрицы М.

Структуры типа 3D имеют довольно ограниченное применение, поскольку имеются другие структуры, например, структуры типа 2DM (2D модифицированная), где сочетаются достоинства структур как 2D так и 3D типа, где упрощается дешифрация адреса и не требуются запоминающие элементы с двухкоординатной выборкой.

Память со структурой типа 2DM

ЗУ типа ROM имеют структуры 2DM для матрицы запоминающих элементов с адресацией от дешифратора DCX имеет как бы характер структуры 2D: выбранный выход дешифратора выбирает целую строку. Однако в отличие от структуры 2D, **длина строки не равна разрядности хранимых слов**, **а многократно ее превышает**. При этом число строк матрицы уменьшается и, соответственно, уменьшается число выходов дешифратора. Для выбора одной из строк служат не все разряды адресного кода, а их часть *Ax = An-1, An-2,..., Ak* поступающую на дешифратор строк DCX, то есть дешифратор *DCX* обслуживает 2^{n-k} строк, каждая из которых хранит 2^{k} *m*-разрядных слов.

Остальные разряды адреса *Ay = Ak-1, Ak-2,..., A0* используются, чтобы выбрать необходимое слово из того множества слов, которое содержится в строке (рисунок 7). Это выполняется с помощью мультиплексоров *MUX* с организацией "$2^{k}\rightarrow1$", на адресные входы которых подаются коды адреса Ay.

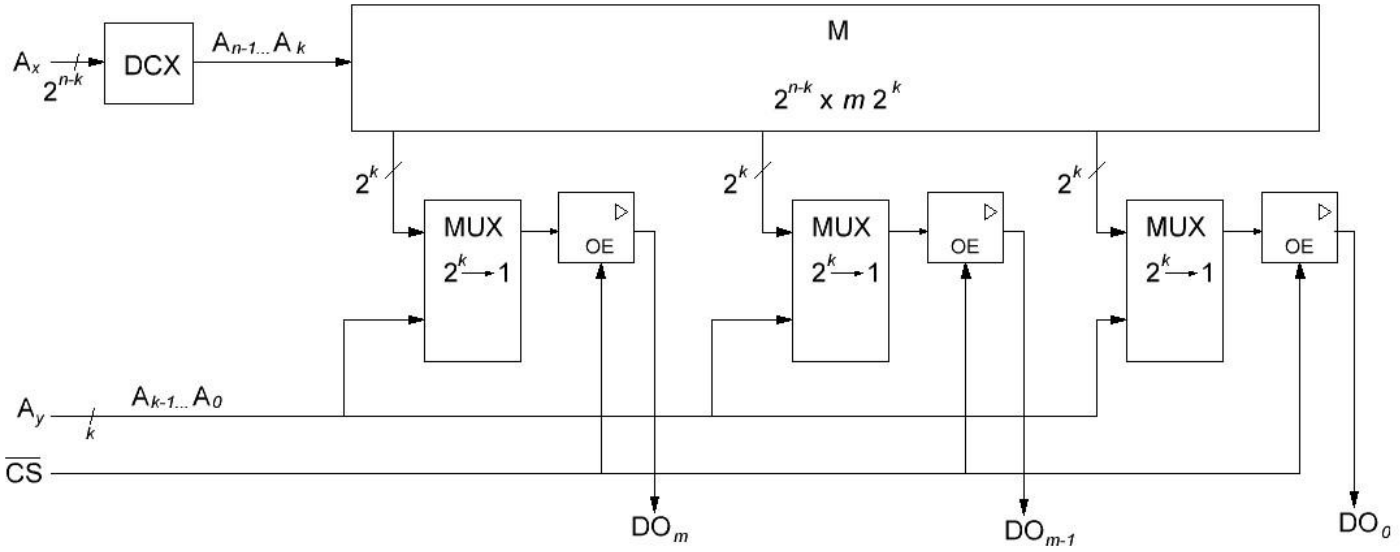


Рисунок 7 − Память SRAM и ROM со структурой 2DM

Длина строки равна $m \cdot 2^k$, где m — разрядность хранимых слов.

Из каждого "отрезка" строки (группы) длиной 2^k мультиплексор выбирает один бит.

На выходах мультиплексоров формируется выходное слово.

По разрешению сигнала CS, поступающего на входы OE управляемых буферов с тремя состояниями, выходное слово передается на внешнюю шину (рисунок 7).

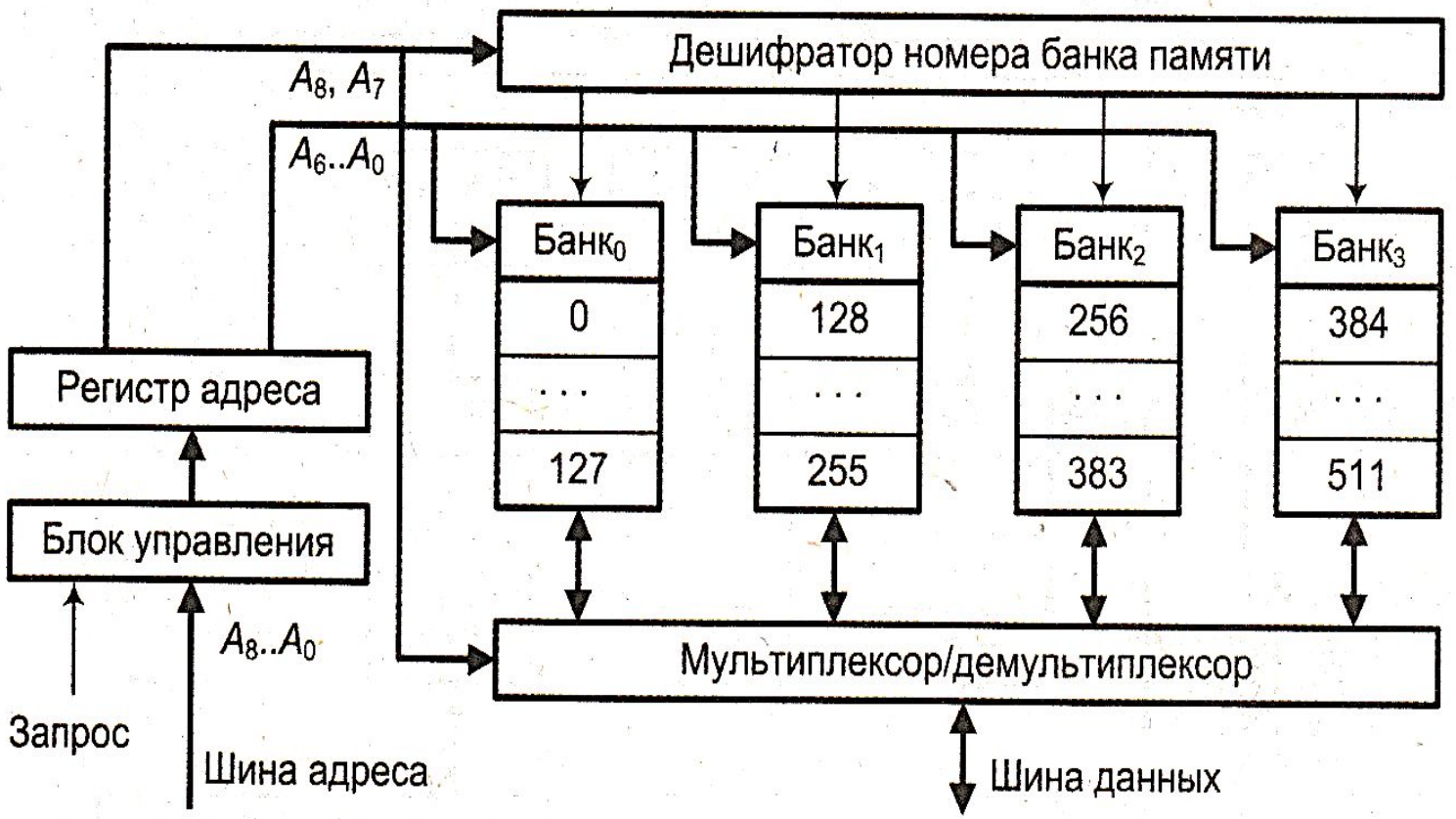


Рисунок 8 − Матрицы М с организацией 4 х 64·2

На рисунке 9 в более общем виде показана структура 2DM для ЗУ типа RAM с операциями чтения и записи.

Из матрицы **M** по-прежнему считывается "длинная" строка разрядностью *r*.

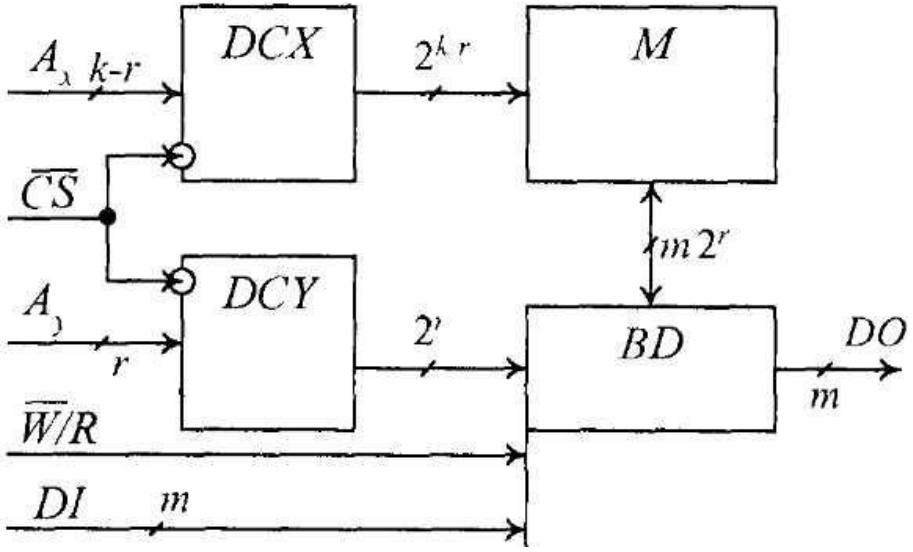


Рисунок 9 − Структура ЗУ типа 2DM для RAM

Данные в соответствующие группы строки записываются или считываются буферами данных ***BD***, которые управляются сигналами с выходов дешифратора ***DCY.***

Буфер ***BD*** также определяет направление обмена данными с помощью сигнала ***W / R***

Организация памяти со структурой 2DM является наиболее распространенной, особенно для микросхем большой емкости.

Структурная организация КЭШ-памяти

Иерархия памяти

В основе реализации иерархии памяти лежат два принципа:

а) **принцип локальности обращений** и

б) **соотношение стоимость/производительность**.

Принцип локальности обращений. Большинство программ **не выполняют обращений** ко всем своим командам и данным равновероятно, а оказывают предпочтение некоторой части своего адресного пространства.

Иерархия памяти строится на нескольких уровнях, причем более высокий уровень меньше по объему, быстрее и имеет большую стоимость в пересчете на байт, чем более низкий уровень.

Уровни иерархии взаимосвязаны: все данные на одном уровне могут быть также найдены на более низком уровне, и все данные на этом более низком уровне могут быть найдены на следующем нижележащем уровне и так далее, пока мы не достигнем основания иерархии.

Иерархия памяти обычно состоит из многих уровней, но **в каждый момент времени** мы имеем дело **только с двумя близлежащими уровнями**.

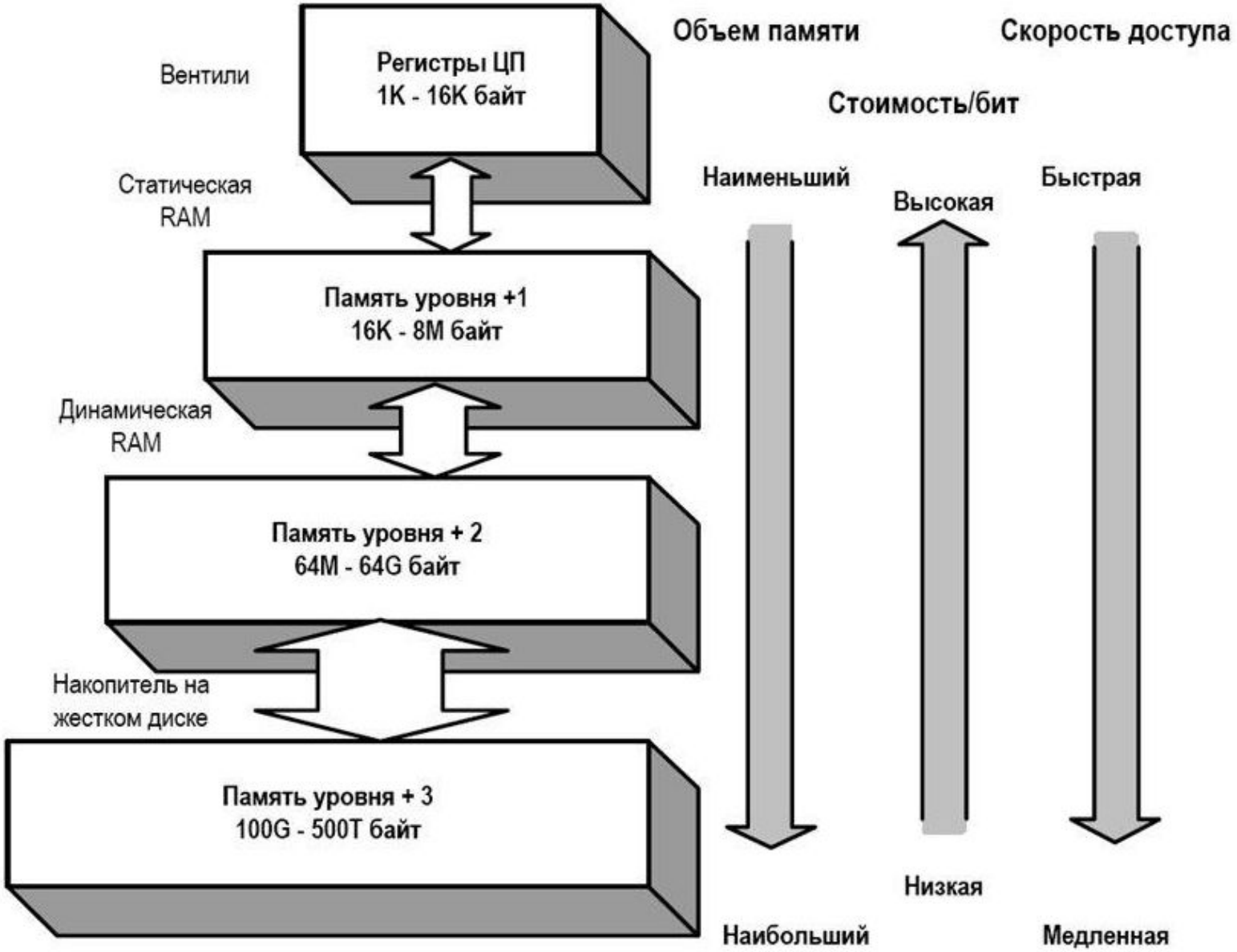


Рисунок 10 – Иерархия памяти

Минимальная единица информации, которая может либо присутствовать, либо отсутствовать в двухуровневой иерархии, называется **блоком.**

Размер блока может быть либо **фиксированным**, либо **переменным**. Если этот размер зафиксирован, то объем памяти является кратным размеру блока.

Успешное или **неуспешное** обращение к более высокому уровню называются соответственно **попаданием** (**hit**) или **промахом** (**miss**).

Попадание – есть обращение к объекту в памяти, который найден на более высоком уровне, в то время как промах означает, что он не найден на этом уровне.

Доля попаданий (**hit rate**) или **коэффициент попаданий** (**hit ratio**) это доля обращений, найденных на более высоком уровне. Представляется в процентах.

Доля промахов (**miss rate**) есть доля обращений, которые не найдены на более высоком уровне.

Главная причина появления иерархии памяти это **повышение производительности** вычислительной системы. Частота попаданий и промахов является важной количественной характеристикой производительности.

Время обращения при попадании (**hit time**) есть время обращения к более высокому уровню иерархии, которое включает в себя, в частности, и время, необходимое для определения того, является ли обращение попаданием или промахом.

Потери на промах (**miss penalty**) есть время для замещения блока в более высоком уровне на блок из более низкого уровня плюс время для пересылки этого блока в требуемое устройство (обычно в процессор).

Потери на промах включают в себя две компоненты:

- **время доступа** (**access time**) – время обращения к первому слову блока при промахе. Время доступа связано с задержкой времени при обращении к памяти более низкого уровня
- **время пересылки** (**transfer time**) – дополнительное время для пересылки оставшихся слов блока. Время пересылки связано с пропускной способностью канала связи между устройствами памяти двух смежных уровней.

Расширение функциональных возможностей микропроцессоров, возрастание программной сложности решаемых задач **обуславливает необходимость увеличения ёмкости ОЗУ.**

Проблема. Увеличение ёмкости ОЗУ увеличивает время обмена между процессором и ОЗУ. Быстродействие ОЗУ может существенно ограничить производительность процессора из-за простоя конвейера команд.

Методы решения:

1. **Модульное построение ОЗУ с расслоением**.

   Сохраняется неизменным среднее время доступа к одной команде, но увеличивается среднее число обращений к памяти, выполняемых одновременно. Частный случай расслоения – физическое разбиение памяти на две части – область хранения команд и область хранения данных

Недостатки.

1. При простом расслоении процессор запрашивая конкретный элемент информации вместе с самим элементом получает в нагрузку копии слов памяти окружающих данный элемент. Чтобы избавится от этого недостатка процессор должен где-то хранить эти копии и всякий раз при обращении к памяти проверять не является ли какое-либо из них нём, что ему нужно.
2. Расслоение не убыстряет доступа к отдельному слову. Следовательно конвейерный процессор должен делать запросы к памяти с опережением по времени, что не всегда возможно.

Организация специальной дополнительной памяти между микропроцессора и ОЗУ

Локальная быстродействующая память должна отвечать требованиям обеих устройств по быстродействию между которыми она находится. Такая память называется «**Кэш-память**».

Кэш-память (англ. *Cache* – класть в тайник; прятать что-либо про запас в потайном месте) запоминает копии информации, передаваемой между устройствами (прежде всего между процессором и основной памятью).

Конструктивно Кэш-память представлена или в виде набора отдельных микросхем или может быть размещена на кристалле процессора непосредственно.

Кэш-память обычно имеет небольшую ёмкость в сравнении с основной памятью и существенно более высокое быстродействие (реализуется на статических (триггерных) элементах памяти).

В любой момент времени Кэш-память содержит копии слов основной ОЗУ, включая фрагменты программ и данных. **Доступ к ним производится в несколько раз быстрее, чем с основной ОЗУ.**

Процессор запрашивает нужную информацию сначала в Кэш-памяти, а при отсутствии делает запрос к основной ОЗУ (рисунок 11).

Замена информации в какой-либо ячейке ОЗУ должна сопровождаться изменением её копии в Кэш-памяти (если она там имеется). Этот механизм организован на аппаратном уровне и выполняется без участия программиста.

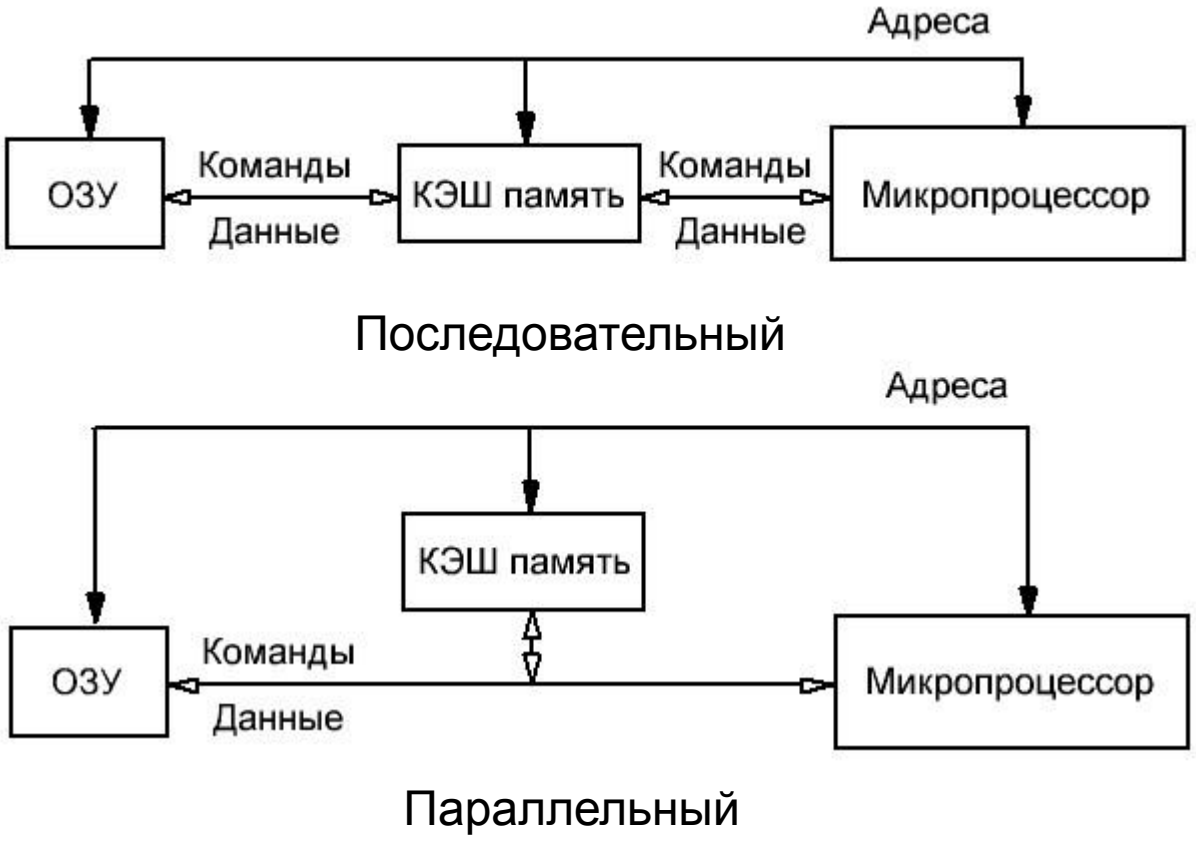


Рисунок 11 − Варианты подключения Кэш-памяти

Эффективность использования Кэш-памяти существенно **зависит от** метода локальной оптимизации **программного обеспечения**, направленной на увеличение числа попаданий в Кэш-память в процессе информационного обмена процессора с памятью.

Быстродействие системы Кэш-память – ОЗУ оценивается эффективным временем доступа.

$$T_{ЭФ} = T_0 + 0{,}5 \cdot \alpha_П \cdot t_\delta + \sum P_{КОНj} \cdot t_{КОНj}$$

$\mathbf{T_0}$ – среднее время обращения к системе Кэш-память – ОЗУ

$\boldsymbol{\alpha_П}$ – доля команд, результат выполнения которых записывается в ОЗУ и Кэш-память (команды типа «память – память»)

$\mathbf{t_\delta}$ – время блокировки памяти при записи данных

$\mathbf{P_{кон\ j}}$ – вероятность возникновения в конвейере конфликта j-го типа.

$\mathbf{t_{кон\ j}}$ – время задержки доступа к памяти из-за конфликта j-го типа в конвейере

Оптимальной по своей структуре считается та Кэш-память у которой $T_{эф} \rightarrow min$

Оптимизация Кэш-памяти обычно связано с минимизацией параметра T_0 – среднее время обращения.

$$T_0 = P_{КП} \cdot t_{КП} + (1 - P_{КП}) \cdot T_{ОЗУ}$$

$P_{КП}$ – вероятность нахождения запрашиваемой информации в Кэш-памяти
$t_{КП}$ – время доступа к Кэш-памяти
$T_{ОЗУ}$ – время доступа к ОЗУ.

$$t_{КП} = t_{КП1} + t_{КП2}$$

$t_{КП1}$ – время поиска информации в Кэш-памяти
$t_{КП2}$ – время выборки информации из Кэш-памяти

Уменьшение **T_0** может быть обеспечено при увеличении **$P_{КП}$** и уменьшении **$t_{КП}$** и **$T_{ОЗУ}$**

Время доступа к ОЗУ зависит от структуры ОЗУ и организации циклов обмена

Для ОЗУ последовательного типа

$$T_{ОЗУ} = t_{КП} + t_{В}$$

$t_{В}$ – время выборки информации из ОЗУ

Для ОЗУ параллельного типа

$$T_{ОЗУ} = t_{КП1} + t_{В}$$

Минимальные значения $T_{ОЗУ}$ достигаются **в параллельной структуре** системы памяти, когда процессор может делать запрос одновременно и к ОЗУ и к Кэш-памяти.

Это **приводит к усложнению аппаратной части подсистемы памяти**, так как необходимо организовать механизм блокирования считывания из ОЗУ.
Тем не менее при таком варианте обмена $T_{ОЗУ}$ будет фактически определяться временем цикла оперативной памяти $T_{ОЗУ} \approx T_{Ц}$.

$$T_0 = P_{КП} \cdot t_{КП} + (1 - P_{КП}) \cdot T_{Ц}$$

Функциональные характеристики Кэш-памяти определяются значениями $P_{КП}$ и $t_{КП}$.
$P_{КП}$ зависит от:

а) характеристик самой Кэш-памяти (ёмкость, применяемый алгоритм удаления редкоиспользуемой информации и т.д.) и
б) характеристики используемой программы.

$t_{КП}$ зависит от типа используемой Кэш-памяти, её ёмкости (числа ячеек $n_{КП}$).
Наибольшее влияние на $t_{КП}$ и $P_{КП}$ оказывает ёмкость Кэш-памяти $V_{КП}$.
При $V_{КП} \to \infty$ быстродействие падает, а вероятность $P_{КП} \to 1$. Поэтому

$$dT_0 / dV_{КП} = 0$$

если $V_{КП} = 0$ (нет Кэш-памяти), то $P_{КП} = 0$, $t_{КП} = 0$, $T_0 = T_{Ц}$
если $V_{КП} \neq 0$, то $P_{КП} \neq 0$, $t_{КП} < T_{Ц}$ и следовательно $T_0 < T_{Ц}$

Таблица 1 - Типовые значения ключевых параметров для кэш-памяти рабочих станций и серверов

Размер блока (строки)	4-128 байт
Время попадания (hit time)	1-4 такта синхронизации (обычно 1 такт)
Потери при промахе (miss penalty) (Время доступа - access time) (Время пересылки - transfer time)	8-32 такта синхронизации (6-10 тактов синхронизации) (2-22 такта синхронизации)
Доля промахов (miss rate)	1%-20%
Размер кэш-памяти	4 Кбайт - 16 Мбайт