ЛАБОРАТОРНАЯ РАБОТА № 3
"Использование модулей памяти"

1. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ
1.1. Общие сведения о памяти ЭВМ

Память предназначена для фиксации, хранения и выдачи информации в процессе работы ЭВМ. Процессы чтения и записи информации определяются как процессы обращения к запоминающему устройству (ЗУ). ЗУ характеризуются:

- местом расположения (на кристалле ЦП, мат. плате, внешняя память);
- ёмкостью;
- единицей пересылки;
- методом доступа;
- быстродействием;
- физическим типом (полупроводники, магнитный носитель, оптика);
- физическими особенностями (энергозависимая /энергонезависимая);
- стоимостью.

Емкость ЗУ характеризуют числом битов либо байтов, которое может храниться в запоминающем устройстве.

Единица пересылки обычно равна ширине шины данных (ШД) (слову), но не обязательно.

Методы доступа к ЗУ:

Последовательный доступ. ЗУ с последовательным доступом, ориентированные на хранение информации в виде последовательности блоков данных, называемых записями. Для доступа к нужному элементу (слову или байту) необходимо прочитать все предшествующие ему данные. Пример - Магнитные ленты.

Прямой доступ. Каждая запись имеет уникальный адрес, отражающий ее физическое размещение на носителе информации. Обращение осуществляется как адресный доступ к началу записи с последующим последовательным доступом к определенной единице информации внутри записи. Пример - жесткий диск.

Произвольный доступ. Каждая ячейка памяти имеет уникальный физический адрес. Обращение к любой ячейке занимает одно и то же время и может проводиться в произвольной очередности. Пример - ОЗУ.

Ассоциативный доступ. Этот вид доступа позволяет выполнять поиск ячеек, содержащих такую информацию, в которой значение отдельных битов совпадает с состоянием одноименных битов в заданном образце. Сравнение осуществляется параллельно для веек ячеек памяти, независимо от ее емкости. Пример – КЭШ-память.

Быстродействие ЗУ:

Время доступа. Для памяти с произвольным доступом оно соответствует интервалу времени от момента поступления адреса до момента, когда данные заносятся в память или становятся доступными. В ЗУ с подвижным носителем информации это время, затрачиваемое на установку головки записи/считывания (или носителя) в нужную позицию.

Длительность цикла памяти или период обращения (ТЦ). Понятие применяется к памяти с произвольным доступом, для которой оно означает минимальное время между двумя последовательными обращениями к памяти. Период обращения включает в себя время доступа плюс некоторое дополнительное время. Дополнительное время может требоваться для затухания сигналов на линиях, а в некоторых типах ЗУ, где считывание информации приводит к ее разрушению, - для восстановления считанной информации.

Скорость передачи. Это скорость, с которой данные могут передаваться в память или из нее. Для памяти с произвольным доступом она равна 1/ТЦ. Для других видов памяти скорость передачи определяется соотношением:

$$TN = TA + N/R,$$

где TN - среднее время считывания или записи N битов; TA - среднее время доступа; R - скорость пересылки в битах в секунду.

Стоимость - отношение общей стоимости ЗУ к его ёмкости в битах, стоимость хранения одного бита информации.

Основная память (ОП) - единственный вид памяти, к которой ЦП может обращаться непосредственно. Основная память - ЗУ с произвольным доступом.

Основная память может включать в себя два типа устройств:

- оперативные запоминающие устройства (ОЗУ) – RAM (Random Access Memory) и
- постоянные запоминающие устройства (ПЗУ) - ROM (Read Only Memory).

ПЗУ обеспечивает считывание информации, но не допускает ее изменения в отличие от ОЗУ.

Традиционно, понятие RAM противопоставляется ROM. Исходя из полных английских названий, можно сделать вывод, что память типа ROM не является памятью с произвольным доступом. Однако это неверно, потому как доступ к устройствам ROM может осуществляться в произвольном, а не строго последовательном порядке. И на самом деле, наименование «RAM» изначально противопоставлялось ранним типам памяти, в которых операции чтения/записи могли осуществляться только в последовательном порядке. В связи с этим, более правильно назначение и принцип работы оперативной памяти отражала бы аббревиатура RWM (Read/Write Memory), которая, тем не менее, практически не встречается. Стоит отметить, что русскоязычным названиям и сокращениям (ОЗУ и ПЗУ) подобная путаница не присуща.

Энергозависимые ОЗУ можно также подразделить еще на две основные подгруппы: динамическую память (DRAM – Dynamic Random Access Memory) и статическую память (SRAM - Static Random Access Memory).

1.2. Использование параметризированных модулей памяти в проектах САПР Quartus

В библиотеке параметризированных модулей можно найти следующие модули ROM/RAM-памяти, поддерживаемые семейством Flex10K:

lpm_ram_dq	Parameterized RAM with separate input and output ports megafunction.
lpm_ram_dp	Parameterized dual port RAM megafunction.
alt3pram	Parameterized triple-port RAM megafunction.
lpm_rom	Parameterized ROM megafunction.
lpm_ram_io	Parameterized RAM with a single I/O port megafunction.
altdpram	Parameterized dual-port RAM megafunction.
csdpram	Parameterized cycle-shared dual port RAM megafunction.

Рассмотрим некоторые из этих блоков более подробно.

Встраивание параметризированных модулей в проект, как вы уже могли заметить, можно производить с помощью мастера, который запускается автоматически после выбора соответствующего модуля из библиотеки, если выставлена галочка “Launch MegaWisard Plugin”. Ручная настройка параметров имеет некоторые плюсы, в частности для блоков памяти это вход, позволяющий перевести выходы в третье состояние. При запуске мастера создается VHDL-описание фактически нового блока с указанными вами параметрами, при ручной же настройке модуль берется стандартный, а уже настройки указываются вручную в его свойствах. Задание рассчитано на использование ручных настроек. Однако это не означает, что нельзя использовать мастер, просто понадобится дополнительные буферы для перевода линий в третье состояние, чтобы исключить конфликт шин. Советы и пояснения по работе с мастером приводятся в отдельном файле megawisard.pdf. Дальше приводится описание настроек выставляемых вручную.

1.2.1. Работа с модулем lpm_rom.

Находим модуль в библиотеке и добавляем его на рабочую область.

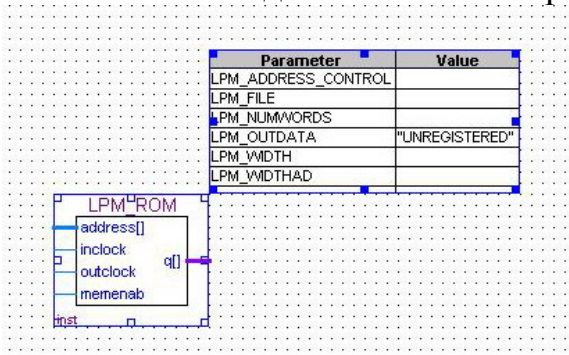


Рис. 1.1. Модуль lpm_rom

Далее необходимо выбрать используемые и неиспользуемые порты. Сделать это можно в свойствах модуля, на закладке портов (рис. 1.2).

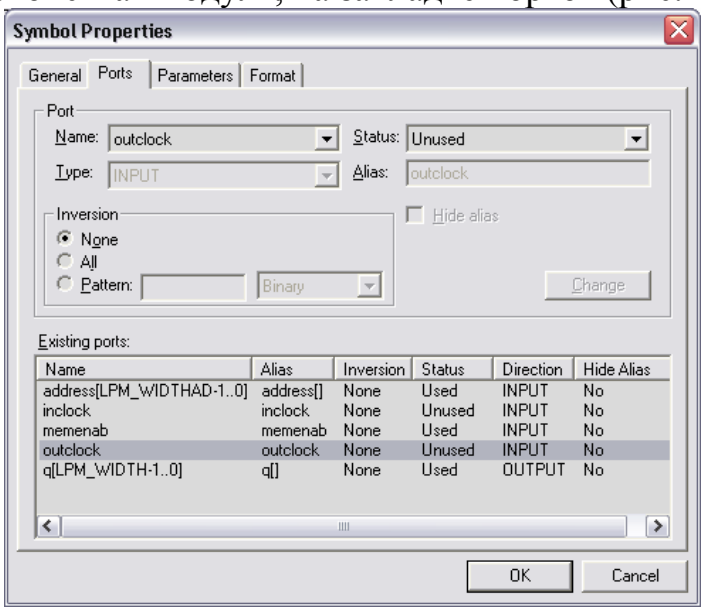


Рис. 1.2. Свойства модуля lpm_rom – закладка Порты

В настройках присутствуют порты address и q – соответственно порт ввода адреса и порт вывода данных. Присутствуют также два порта тактирования – inclock и outclock. Стоит обратить внимание, что если порты inclock и outclock не используются (Unused), то ввод и вывод данных производится асинхронно. В противном случае ввод синхронизируется передним фронтом inclock, а вывод – передним фронтом outclock. Порт memenab позволяет перевести выходную шину в состояние высокого импеданса (отключено).

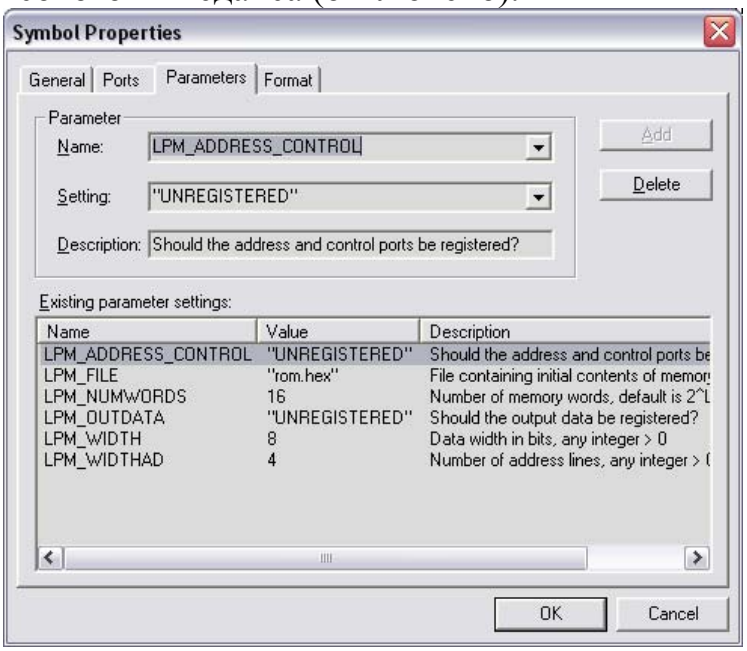
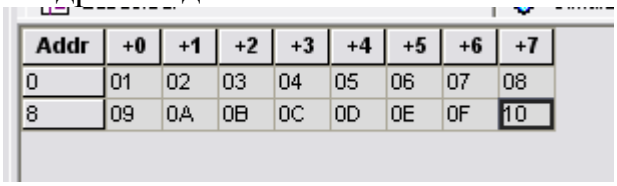


Рис. 1.3. Свойства модуля lpm_rom – закладка Параметры

Переходим к закладке параметров модуля (рис. 1.3). Здесь указываются такие параметры, как разрядность адресного входа, разрядность выхода данных, количество слов в памяти, а также параметры указывающие присутствуют

ли на входе и выходе регистры (необходимы для синхронных операций). Также здесь указвыается путь и название файла инициализации памяти (т.н. файл «прошивки»). Обратите внимание на то, что все строковые параметры записываются в кавычках. По умолчанию файл инициализации памяти ищется в директории проекта. Файл может иметь формат Memory Initialization File (.mif) или Hexadecimal (Intel-Format) File (.hex). Если вы указали несуществующий файл инициализации, то перед компиляцией его необходимо создать либо в текстовом редакторе, либо в самом Quartus'е через меню New. При редактировании в Quartus'е (рис. 1.4) в меню View->Address Radix и Memory Radix задается система счисления для адреса и данных соответственно.



Addr	+0	+1	+2	+3	+4	+5	+6	+7
0	01	02	03	04	05	06	07	08
8	09	0A	0B	0C	0D	0E	0F	10

Рис. 1.4. Редактирование файла прошивки в среде Quartus

После окончания настройки модуля можно попробовать промоделировать его работу. Дня этого необходимо проинициализировать память значениями из файла и получить на выходе значения по заданному адресу. Для файла rom.hex, содержимое которого приводилось на рис. 1.4 моделирование будет выглядеть следующим образом (рис. 1.5).

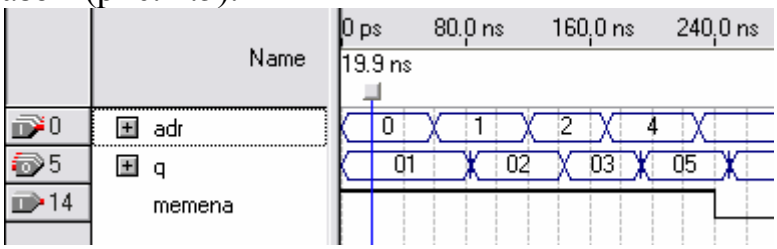


Рис. 1.5. Моделирование блока lpm_rom

Стоит также отметить тот факт, что изменение файла инициализации должно сопровождаться перекомпиляцией проекта.

Также нельзя забывать, что модуль имеет достаточно большое время срабатывания (>20 нс). Поэтому очень часто работа с памятью является самым узким местом системы.

Таблицу истинности для данного модуля можно посмотреть в помощи Quartus.

1.2.2. Работа с модулем lpm_ram_io.

Работа с модулем lpm_ram_io производится аналогично. Отличием являются наличие входа разрешения записи и возможность выполнения операции записи в этот модуль.

Таблица истинности модуля также находится в помощи Quartus.

Особое внимание стоит уделить тактированию входа, поскольку под входом понимается не только занесение данных в модуль, но и адреса.

Результаты операций записи необходимо искать не в файле инициализации, а в отчете моделирования в пункте Simulator->Logical Memories.

2. ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Разработать блок, включающий в себя модули lpm_rom и lpm_ram выходящие на общую шину. Продемонстрировать операции чтения из памяти. Синхронность/асинхронность операций указана по варианту.

2. Используя 8-разрядный регистр, прочитать данные из ячейки-источника памяти ПЗУ и записать их в ячейку-приемник памяти ОЗУ.

Семейство ПЛИС для реализации - Flex10K (изменить семейство ПЛИС(Family) можно в настройках проекта в ветке Device или в пункте меню Assignments->Device)

ВАРИАНТЫ ЗАДАНИЙ К ЛАБОРАТОРНОЙ РАБОТЕ № 3

№ варианта	lpm_rom		lpm_ram_io		Ячейка Источник	Ячейка Приемник
	Ввод	Ввод	Ввод	Вывод		
1	Синхр.	Асинхр.	Синхр.	Синхр.	1	8
2	Асинхр.	Асинхр.	Синхр.	Асинхр.	2	7
3	Синхр.	Синхр.	Синхр.	Асинхр.	3	6
4	Асинхр.	Синхр.	Синхр.	Синхр.	4	5
5	Синхр.	Асинхр.	Синхр.	Асинхр.	5	4
6	Асинхр.	Асинхр.	Синхр.	Синхр.	6	3
7	Синхр.	Синхр.	Синхр.	Асинхр.	7	2
8	Асинхр.	Синхр.	Синхр.	Синхр.	8	1

СОДЕРЖАНИЕ ОТЧЕТА ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

1. Задание.
2.1. Схема блока.
2.2. Результаты моделирования.