Memory Basics

• Uses:
  Whenever a large collection of state elements is required.
  – data & program storage
  – general purpose registers
  – buffering
  – table lookups
  – CL implementation

• Types:
  – RAM - random access memory
  – ROM - read only memory
  – EPROM, FLASH - electrically programmable read only memory

• Example RAM: Register file from microprocessor

  regid = register identifier
  (address of word in memory)
  sizeof(regid) = log2(# of reg)
  WE = write enable
Definitions

Memory Interfaces for Accessing Data

- **Asynchronous** (unclocked):
  A change in the address results in data appearing

- **Synchronous** (clocked):
  A change in address, followed by an edge on CLK results in data appearing or write operation occurring.

  A common arrangement is to have synchronous write operations and asynchronous read operations.

- **Volatile**: Looses its state when the power goes off.
- **Nonvolatile**: Retains its state when power goes off.

Register File Internals

- For read operations, functionally the regfile is equivalent to a 2-D array of flip-flops with tristate outputs on each:

- Cell with added write logic:

These circuits are just functional abstractions of the actual circuits used.

How do we go from "regid" to "SEL"?
Regid (address) Decoding

- The function of the address decoder is to generate a one-hot code word from the address.
- The output is used for row selection.
- Many different circuits exist for this function. A simple one is shown to the right.

Standard Internal Memory Organization

- Special circuit tricks are used for the cell array to improve storage density. (We will look at these later)
- RAM/ROM naming convention:
  - examples: 32 X 8, "32 by 8" => 32 8-bit words
  - 1M X 1, "1 meg by 1" => 1M 1-bit words
Read Only Memory (ROM)

- Simply for of memory. No write operation needed.
- Functional Equivalence:

  ![Diagram of Read Only Memory](image)

  - Connections to Vdd used to store a logic 1, connections to GND for storing logic 0.

  - Full tri-state buffers are not needed at each cell point.
  - In practice, single transistors are used to implement zero cells. Logic one’s are derived through precharging or bit-line pullup transistor.

Column MUX in ROMs and RAMs:

- Controls physical aspect ratio
  - Important for physical layout and to control delay on wires.
- In DRAM, allows time-multiplexing of chip address pins

![Diagram of Column MUX](image)

- 1K X 1 ROM
- 256 X 4 ROM
Cascading Memory Modules (or chips)

- Example: assemblage of 256 x 8 ROM using 256 x 4 modules:

- Example: 1K x * ROM using 256 x 4 modules:

- Each module has tri-state outputs:

Memory Components Types:

- Volatile:
  - Random Access Memory (RAM):
    - DRAM "dynamic"
    - SRAM "static"

- Non-volatile:
  - Read Only Memory (ROM):
    - Mask ROM "mask programmable"
    - EPROM "electrically programmable"
    - EEPROM "erasable electrically programmable"
    - FLASH memory - similar to EEPROM with programmer integrated on chip
**Volatile Memory Comparison**

The primary difference between different memory types is the bit cell.

- **SRAM Cell**
  - Larger cell ⇒ lower density, higher cost/bit
  - No refresh required
  - Simple read ⇒ faster access
  - Standard IC process ⇒ natural for integration with logic

- **DRAM Cell**
  - Smaller cell ⇒ higher density, lower cost/bit
  - Needs periodic refresh, and refresh after read
  - Complex read ⇒ longer access time
  - Special IC process ⇒ difficult to integrate with logic

---

**Multi-ported Memory**

- **Motivation:**
  - Consider CPU core register file:
    - 1 read or write per cycle limits processor performance.
    - Complicates pipelining. Difficult for different instructions to simultaneously read or write regfile.
    - Common arrangement in pipelined CPUs is 2 read ports and 1 write port.

- **Motivation:**
  - I/O data buffering:
    - Dual-porting allows both sides to simultaneously access memory at full bandwidth.

---
Dual-ported Memory Internals

- Add decoder, another set of read/write logic, bits lines, word lines:
- Example cell: SRAM
- Repeat everything but cross-coupled inverters.
- This scheme extends up to a couple more ports, then need to add additional transistors.

Memory in Desktop Computer Systems:

- **SRAM** (lower density, higher speed) used in CPU register file, on- and off-chip caches.
- **DRAM** (higher density, lower speed) used in main memory

Closing the GAP:
1. Caches are growing in size.
2. Innovation targeted towards higher bandwidth for memory systems:
   - SDRAM - synchronous DRAM
   - RDRAM - Rambus DRAM
   - EDORAM - extended data out SRAM
   - Three-dimensional RAM
   - hyper-page mode DRAM video RAM
   - multibank DRAM
Important DRAM Examples:

- **EDO** - extended data out (similar to fast-page mode)
  - RAS cycle fetched rows of data from cell array blocks (long access time, around 100ns)
  - Subsequent CAS cycles quickly access data from row buffers if within an address page (page is around 256 Bytes)

- **SDRAM** - synchronous DRAM
  - clocked interface
  - uses dual banks internally. Start access in one bank then next, then receive data from first then second.

- **DDR** - Double data rate SDRAM
  - Uses both rising (positive edge) and falling (negative) edge of clock for data transfer. (typical 100MHz clock with 200 MHz transfer).

- **RDRAM** - Rambus DRAM
  - Entire data blocks are access and transferred out on a high-speed bus-like interface (500 MB/s, 1.6 GB/s)
  - Tricky system level design. More expensive memory chips.

Non-volatile Memory

*Used to hold fixed code (ex. BIOS), tables of data (ex. FSM next state/output logic), slowly changing values (date/time on computer)*

- **Mask ROM**
  - Used with logic circuits for tables etc.
  - Contents fixed at IC fab time (truly write once!)

- **EPROM** (erasable programmable) & **FLASH**
  - requires special IC process (floating gate technology)
  - writing is slower than RAM. EPROM uses special programming system to provide special voltages and timing.
  - reading can be made fairly fast.
  - rewriting is very slow.
    - erasure is first required, EPROM - UV light exposure, EEPROM – electrically erasable
FLASH Memory

- Electrically erasable
- In system programmability and erasability (no special system or voltages needed)
- On-chip circuitry (FSM) and voltage generators to control erasure and programming (writing)
- Erasure happens in variable sized "sectors" in a flash (16K - 64K Bytes)

See: http://developer.intel.com/design/flash/ for product descriptions, etc.

- Compact flash cards are based on this type of memory.

Memory Specification in Verilog

- Memory modeled by an array of registers:

```verilog
reg[15:0] memword[0:1023]; // 1,024 registers of 16 bits each

//Example Memory Block Specification
// Uses enable to control both write and read
//-----------------------------
//Read and write operations of memory.
//Memory size is 64 words of 4 bits each.
module memory (Enable,ReadWrite,Address,DataIn,DataOut);
  input  Enable,ReadWrite;
  input [3:0] DataIn;
  input [5:0] Address;
  output [3:0] DataOut;
  reg [3:0] DataOut;
  reg [3:0] Mem [0:63]; //64 x 4 memory
  always @ (Enable or ReadWrite)
    if (Enable)
      if (Read) //Read
        DataOut = Mem[Address];
      else
        Mem[Address] = DataIn; //Write
    else DataOut = 4'bz; //High impedance state
endmodule
```

Spring 2005      EECS150 - Lec17-mem1      Page 18
Memory Blocks in FPGAs

- LUTs can double as small RAM blocks:
  - 4-LUT is really a 16x1 memory. Normally we think of the contents being written from the configuration bit stream, but Virtex architecture (and others) allow bits of LUT to be written and read from the general interconnect structure.
  - achieves 16x density advantage over using CLB flip-flops.
  - Furthermore, the two LUTs within a slice can be combined to create a 16 x 2-bit or 32 x 1-bit synchronous RAM, or a 16x1-bit dual-port synchronous RAM.
  - The Virtex-E LUT can also provide a 16-bit shift register of adjustable length.

- Newer FPGA families include larger on-chip RAM blocks (usually dual ported):
  - Called block selectRAMs in Xilinx Virtex series
  - 4k bits each

Verilog Specification for Virtex LUT RAM

```verilog
module ram16x1(q, a, d, we, clk);
output q;
input d;
input [3:0] a;
input clk, we;
reg mem [15:0];
always @(posedge clk) begin
  if(we)
    mem[a] <= d;
  end
assign q = mem[a];
endmodule
```

Note: synchronous write and asynchronous read.

- Deeper and/or wider RAMs can be specified and the synthesis tool will do the job of wiring together multiple LUTs.
- How does the synthesis tool choose to implement your RAM as a collection of LUTs or as block RAMs?
Virtex “Block RAMs”

- Each block SelectRAM (block RAM) is a fully synchronous (synchronous write and read) dual-ported (true dual port) 4096-bit RAM with independent control signals for each port. The data widths of the two ports can be configured independently, providing built-in bus-width conversion.
- CLKA and CLKB can be independent, providing an easy way to “cross clock bounders”.
- Around 160 of these on the 2000E. Multiples can be combined to implement wider or deeper memories.
- See chapter 8 of Synplify reference manual on how to write Verilog for implied Block RAMs. Or instead, explicitly instantiate as primitive (project checkpoint will use this method).

Table 5: Block SelectRAM Port Aspect Ratios

<table>
<thead>
<tr>
<th>Width</th>
<th>Depth</th>
<th>ADDR Bus</th>
<th>Data Bus</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4096</td>
<td>ADDR&lt;11&gt;</td>
<td>DATA&lt;0&gt;</td>
</tr>
<tr>
<td>2</td>
<td>2048</td>
<td>ADDR&lt;10&gt;</td>
<td>DATA&lt;1&gt;</td>
</tr>
<tr>
<td>4</td>
<td>1024</td>
<td>ADDR&lt;9&gt;</td>
<td>DATA&lt;3&gt;</td>
</tr>
<tr>
<td>8</td>
<td>512</td>
<td>ADDR&lt;8&gt;</td>
<td>DATA&lt;7&gt;</td>
</tr>
<tr>
<td>16</td>
<td>256</td>
<td>ADDR&lt;7&gt;</td>
<td>DATA&lt;15&gt;</td>
</tr>
</tbody>
</table>

Relationship between Memory and CL

- Memory blocks can be (and often are) used to implement combinational logic functions:
- Examples:
  - LUTs in FPGAs
  - 1Mbit x 8 EPROM can implement 8 independent functions each of \( \log_2(1M) = 20 \) inputs.
- The decoder part of a memory block can be considered a “minterm generator”.
- The cell array part of a memory block can be considered an OR function over a subset of rows.
- The combination gives us a way to implement logic functions directly in sum of products form.
- Several variations on this theme exist in a set of devices called Programmable logic devices (PLDs)
A ROM as AND/OR Logic Device

![Diagram of a ROM as AND/OR Logic Device]

(a) Conventional symbol  
(b) Array logic symbol

Fig. 7-1 Conventional and Array Logic Diagrams for OR Gate

PLD Summary

![Diagram of PLD Summary]

(a) Programmable read-only memory (PROM)
(b) Programmable array logic (PAL)
(c) Programmable logic array (PLA)

Fig. 7-13 Basic Configuration of Three PLDs
PLA Example

Fig. 7.14 PLA with 3 Inputs, 4 Product Terms, and 2 Outputs

PAL Example

Fig. 7.17 Fuse Map for PAL as Specified in Table 7-6