RAM & ROM Based Digital Design

ECE 152A – Winter 2012

Reading Assignment

- Brown and Vranesic
  - 10 Digital System Design
    - 10.1 Building Block Circuits
      - 10.1.3 Static Random Access Memory (SRAM)
      - 10.1.4 SRAM Blocks in PLDs

Reading Assignment

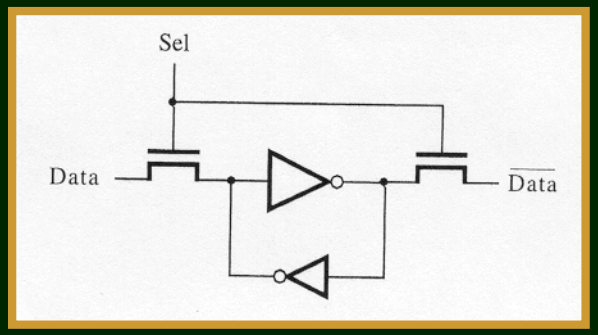
- Roth
  - 9 Multiplexers, Decoders, and Programmable Logic Devices
    - 9.5 Read Only Memories

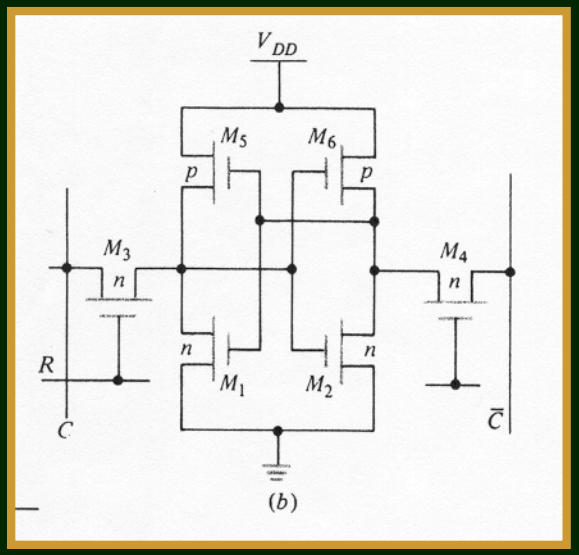
Read/Write Memories

- RAM
  - Random Access Memory
    - Same access time to all memory locations
      - As opposed to serial access memory
    - About the same time for read and write
  - SRAM
    - Static Random Access Memory
      - Built with cross coupled inverters and pass transistors

Read/Write Memories

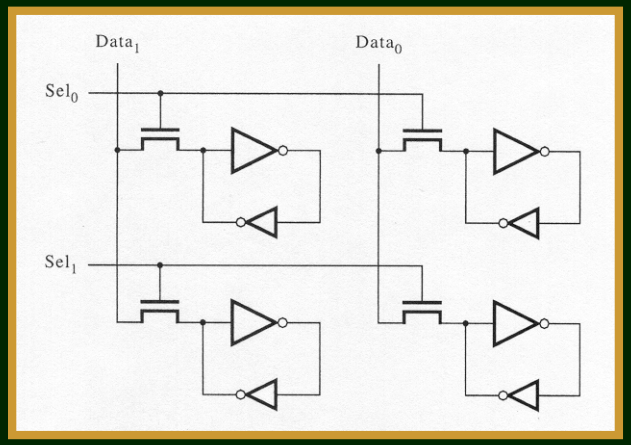
- 6T SRAM Cell
  - CMOS implementation with pass transistors
  - Sense amp at bottom of column

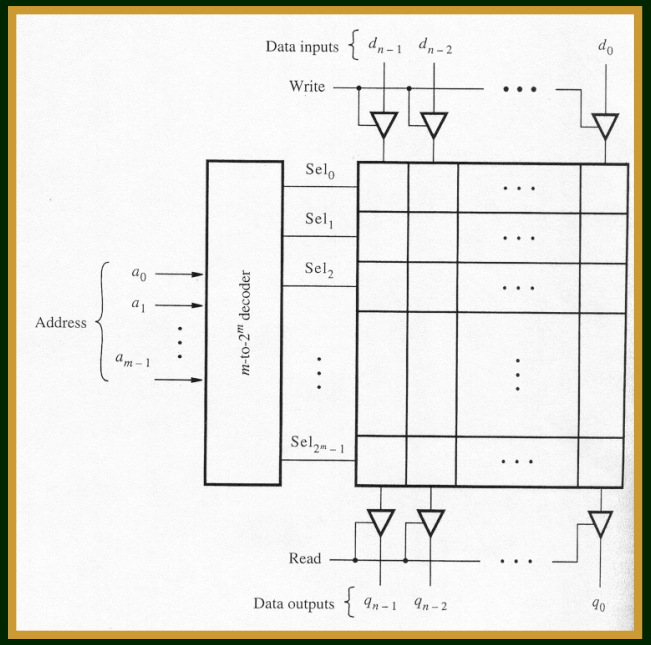




Read/Write Memories

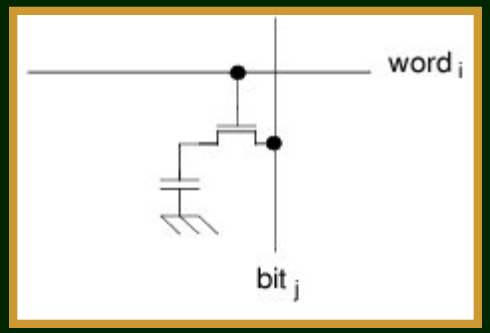
- RAM Blocks and Register Files





Read/Write Memories

- RAM (cont)
  - DRAM
    - Uses capacitance on MOSFET gate
    - Must be “refreshed” periodically
      - Restore charge on gate capacitance
      - Accomplished by reading or writing to memory location



Read/Write Memories

- RAM (cont)
  - Both SRAM and DRAM are “volatile”
    - Data lost when power is removed
  - DRAM has approximately 4 times the capacity of SRAM
    - Basic memory cell of DRAM is smaller
      - ~2 transistors vs. 6
  - SRAM is generally faster

Read Only Memories

- ROM, PROM, EPROM, EEPROM, EAROM
  - Not in-system writeable
    - Except by special means, i.e. non-standard voltages and timing
  - “Non-volatile”
    - Data retained when power removed
  - ROM : Read Only Memory
    - Mask programmed by manufacturer
    - Not erasable (programming permanent)
      - High volume applications

Read Only Memories

- ROM, PROM, EPROM, EEPROM, EAROM
  - PROM : Programmable Read Only Memory
    - Programmed by user
    - Also not erasable
  - EPROM : Erasable Programmable Read Only Memory
    - Programmed by user
    - Erased using UV light
      - “erase” sets all bits to 0 or 1
    - Development applications

Read Only Memories

- ROM, PROM, EPROM, EEPROM, EAROM
  - EEPROM : Electrically Erasable Programmable Read Only Memory
    - Programmed by user
    - Erased electrically
      - Possibly in system, but requires non-standard voltages
  - EAROM : Electrically Alterable Read Only Memory
    - Similar to EEPROM

Read Mostly Memories

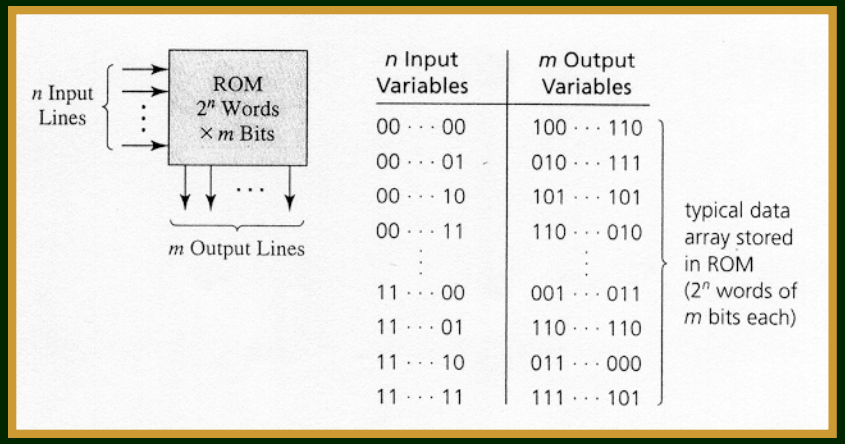
- Flash memory
  - Writable and non-volatile
    - Reads very fast
    - Writes very slowly
      - Referred to as “programming”
  - No special voltages for in system writing
  - “Flash” refers to the fact that the entire content of the memory chip can be erased in one step
  - Once erased and written, data is retained for 20+ years

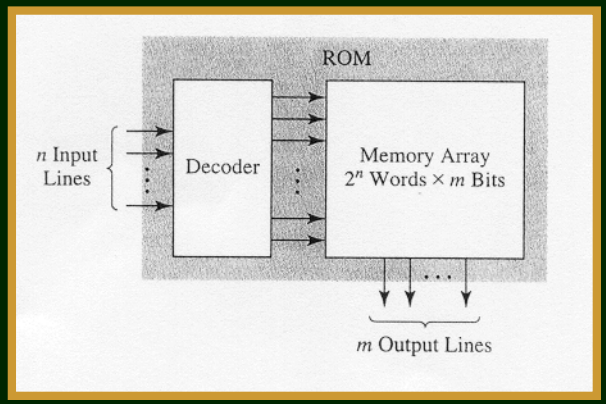
Memory Structure

- Array of memory cells
- Organization refers to number of and width of memory words
  - Example 1024 bit memory can organized as:
    - 1024 one-bit word
    - 512 two-bit words
    - 256 four-bit words
    - 128 eight-bit words
  - Internal array is the same for all organizations
    - Decoding and I/O circuitry differs

Memory Structure

- Memory Array and Address Decoder





Combinational Design with Memories

- ROM (and RAM and Flash) is a “physical” truth table
  - All addresses equal ≡ all inputs to logic network
  - Each row of truth table corresponds to a single address in the memory
  - Example: 128 x 8 ROM
    - 128, 8-bit words
    - Log_2 128 = 7 address bits (A6 – A0)
    - 8 data bits D7 – D0
    - Can implement 7 input, 8 output function

Combinational Design with Memories

- Binary to BCD converter with 128 x 8-bit ROM
  - Addresses 0 – 99
    - Output equals 2 digit BCD number
  - Addresses 100 – 127
    - All one's, indicating invalid input

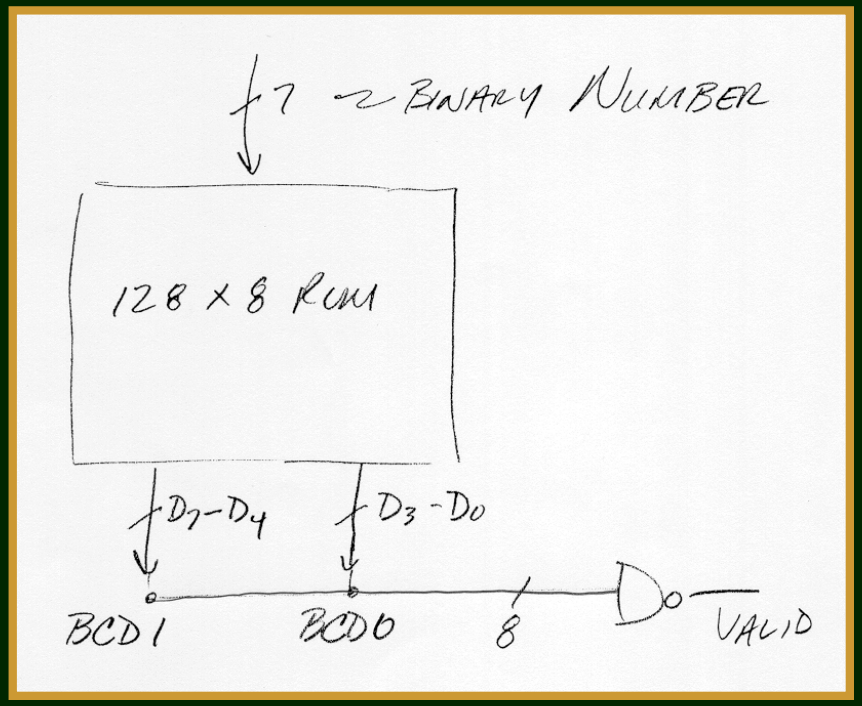
Combinational Design with Memories

- ROM Contents
  - Hex addresses 00 through 7F

Address	Data	Decimal Value
00	0000 0000	00
01	0000 0001	01
0F	0001 0101	15
10	0001 0110	16
63	1001 1001	99
64	1111 1111	Invalid
7F	1111 1111	Invalid

Combinational Design with Memories

- Final Implementation

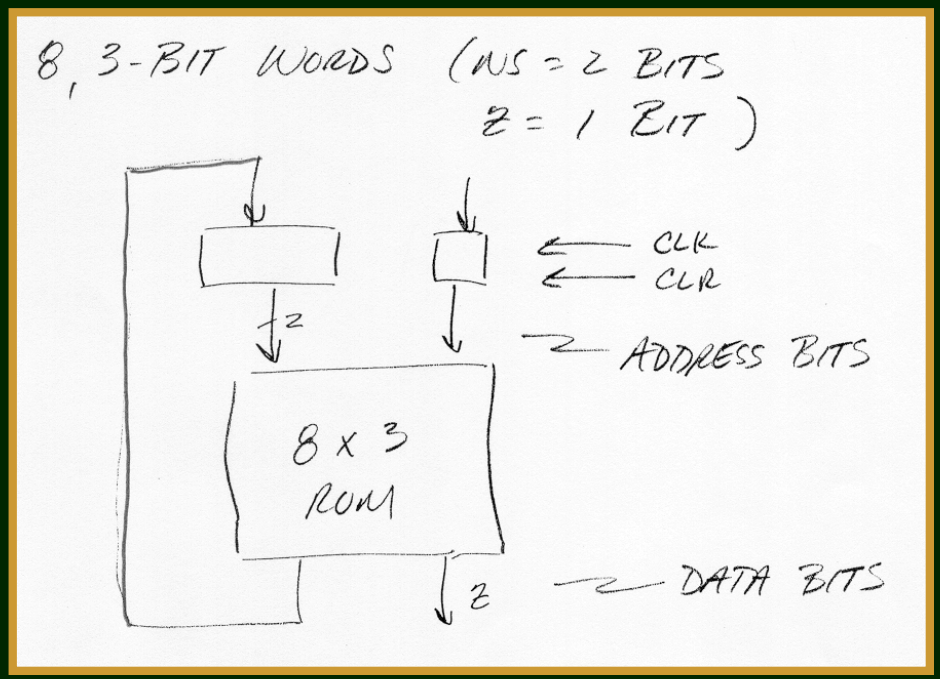


State Machine Design with Memories

- For state machine, map state table directly into memory
  - Address lines driven by present state and present input
  - Data outputs consist of next state and present output
  - Both Mealy and Moore machines can be realized
    - Output of Moore machine lags by one clock period (when state table directly mapped)

State Machine Design with Memories

- Hardware Implementation



State Machine Design with Memories

- Recall 101 sequence detector
  - Mealy machine

PS		NS	
		x=0	x=1
	AB	A^+B^+	A^+B^+
0	00	00,0	01,0
1	01	10,0	01,0
2	10	00,0	01,1

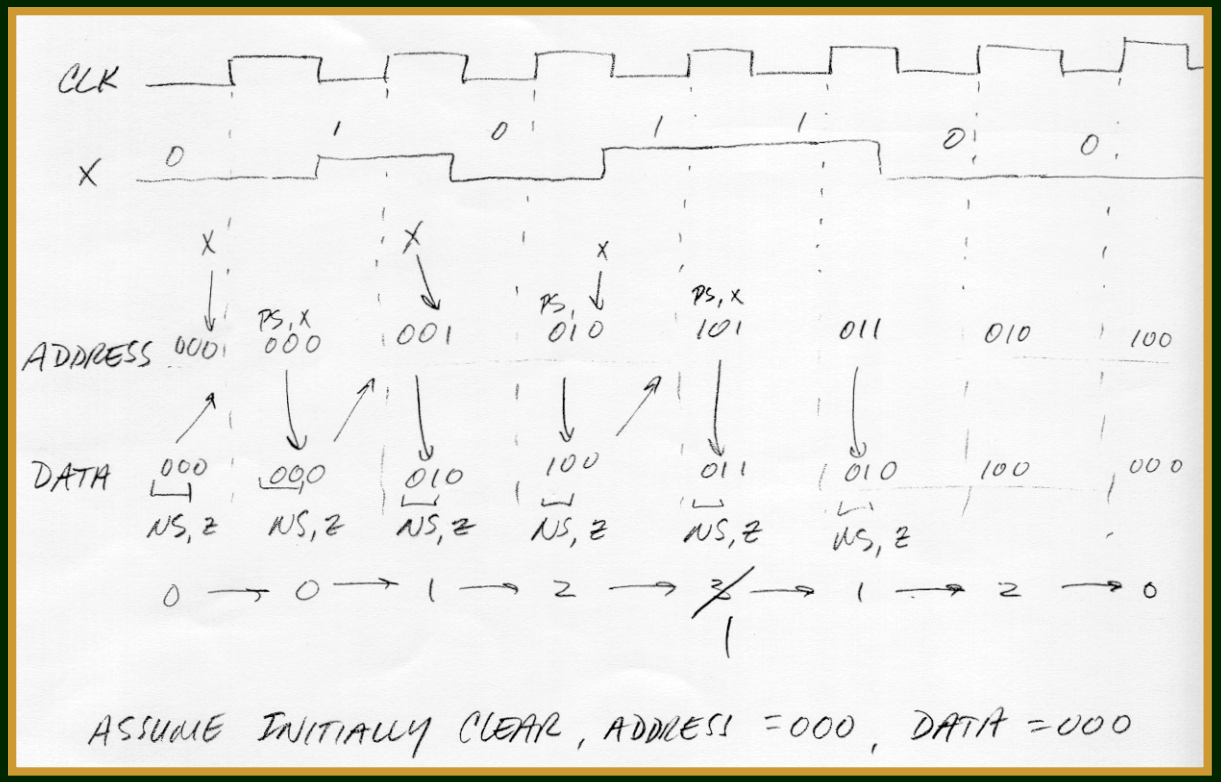
State Machine Design with Memories

- ROM Contents (8 x 3-bit)

Address		Data	
PS	x	NS	z
00	0	00	0
00	1	01	0
01	0	10	0
01	1	01	0
10	0	00	0
10	1	01	1
11	0	xx	xx
11	1	xx	xx

State Machine Design with Memories

- Timing Diagram



State Machine Design with Memories

- 101 sequence detector (again)
  - Moore machine

PS		NS x=0	NS x=1	
	AB	A^+B^+	A^+B^+	Z
0	00	00	01	0
1	01	10	01	0
2	10	00	11	0
3	11	10	01	1

State Machine Design with Memories

- Direct mapping of state table to memory
  - Output lags by one clock period
    - Introduces latency to output timing
    - “Pipelining” effect
- Implement as “Mealy-like” machine
  - Associate output with next state, not present state
    - All states have only one associated output (like Moore)
  - Eliminates one clock-period latency
    - No longer “true” Moore machine