Interfacing with Memory

Reminders

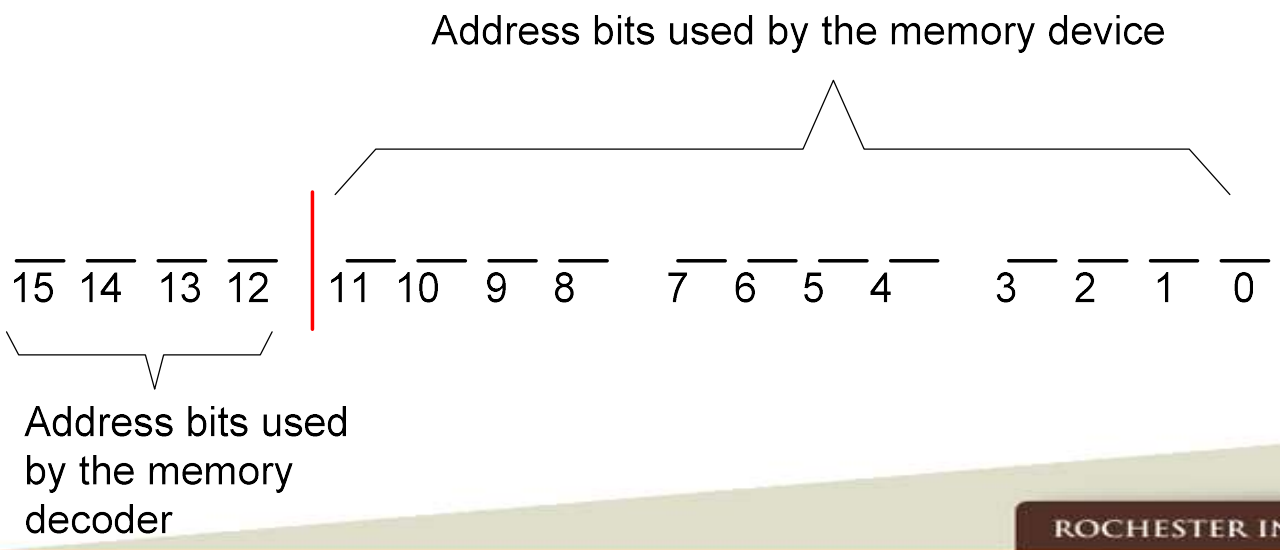
- Interfacing with Memory
- No Quiz this week
- No demo this week
- Lab 4 due Friday

Interfacing with Memory

- Full Address Decoding
  - **Using all the address bits to decode a memory device**
  - **Uses more hardware to decode all address**
  - **No aliasing of memory**
    - Every memory location is accessible from only one address
- Partial Address Decoding
  - **Not all address bits are used to decode memory device**
    - Unused bits are “don’t cares” in address decoding
  - **Typically simpler decoding logic**
  - **Unused bits causes memory fold back or aliasing**
    - Memory locations are accessible from multiple addresses
  - **Waste memory space**

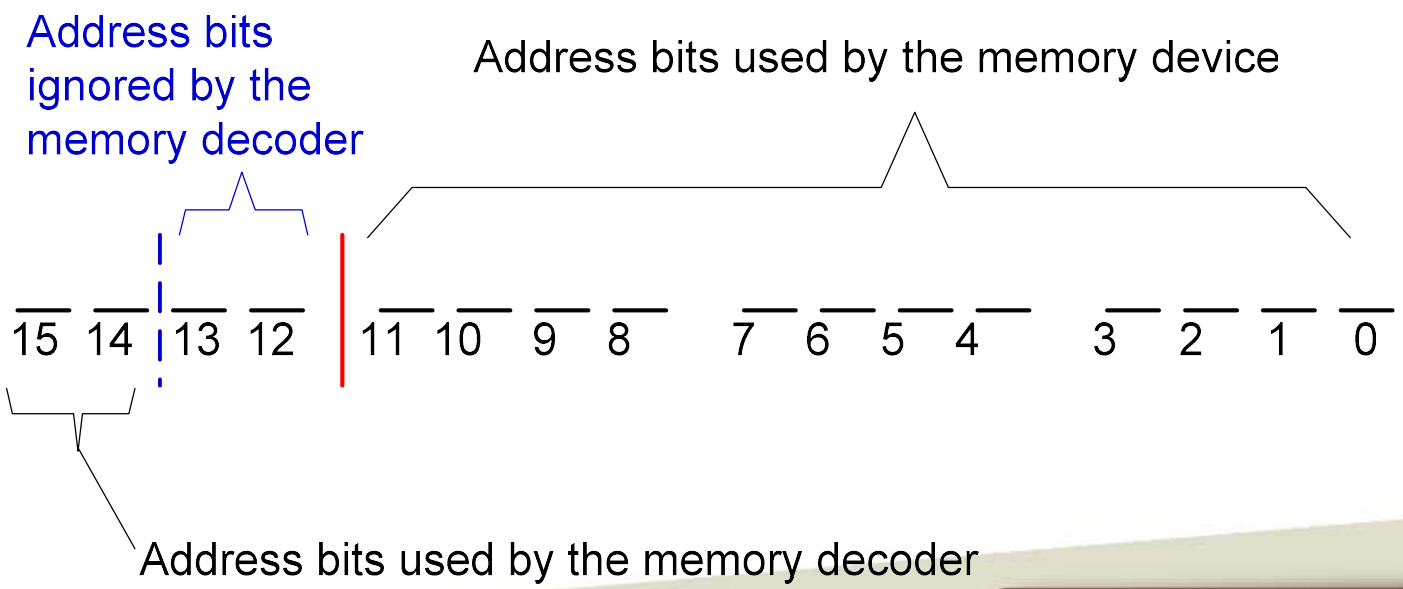
Interfacing with Memory

- Full Address Decoding
  - **Ex: 64Kx8 Memory device and 16-bit address bus**
    - All bits used so only 1 device can exist
  - **Ex: 4Kx8 Memory device and 16-bit address bus**
    - All bits used (4 used to drive chip select) so only 1 instance of device exist



Interfacing with Memory

- Partial Address Decoding
  - **Ex: 4Kx8 Memory device and 16-bit address bus**
    - 4 instances of device exist because bits 13 & 12 are not used
      - ***Total memory space used is 16KB***



Interfacing with Memory

- Byte Ordering
  - **How are bytes stored in multi-byte data element?**
    - Big Endian
      - ***High Order byte stored in lowest address***
    - Little Endian
      - ***Low Order byte stored in lowest address***
  - **Why different ordering?**
    - Developed independently with different reasoning
  - **Generally only issue when data shared in memory**
    - Both sides need to agree on ordering
  - **May cause a problem when reading memory window**
    - HEX vs Intel HEX
    - If also displaying in bytes then no problem

Interfacing with Memory

- Byte Ordering
  - **Consider 32-bit integer value 0x90ABCDEF**
    - Requires 4 byte locations to store in memory
    - Two methods of byte ordering

Memory Addresses →	00	01	02	03
	90	AB	CD	EF

Big Endian Storage

Memory Addresses →	00	01	02	03
	EF	CD	AB	90

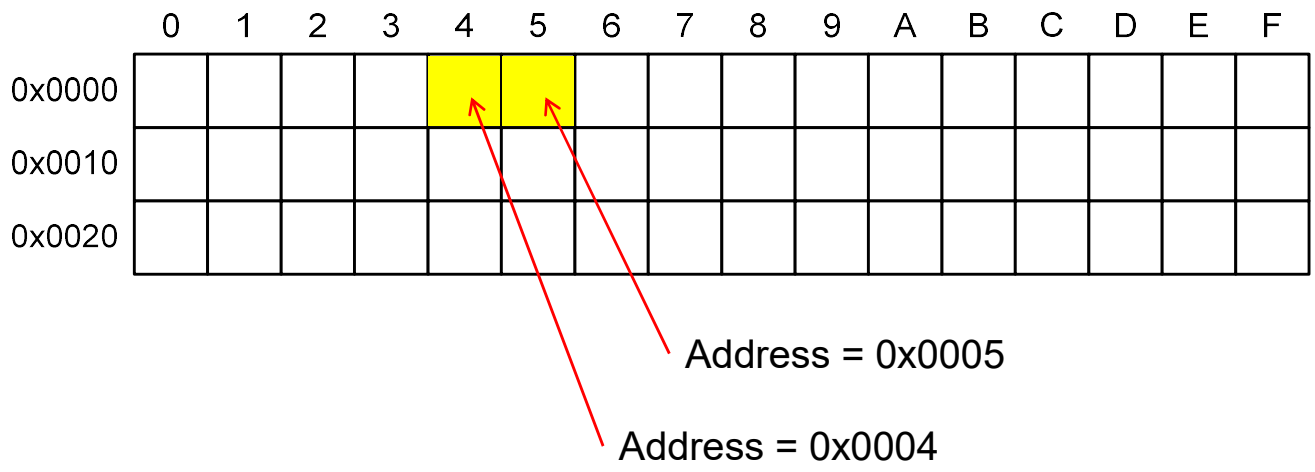
Little Endian Storage

Interfacing with Memory

- Memory Alignment
  - **Most processors require variables to reside at specific offsets in memory**
    - Ex: 32-bit processors require address evenly divisible by 4
  - **Use of misaligned data is supported but at a substantial performance penalty**
    - Some compilers automatically align data variables based on type
      - ***Makes size of structures occupy more memory than sum of parts***

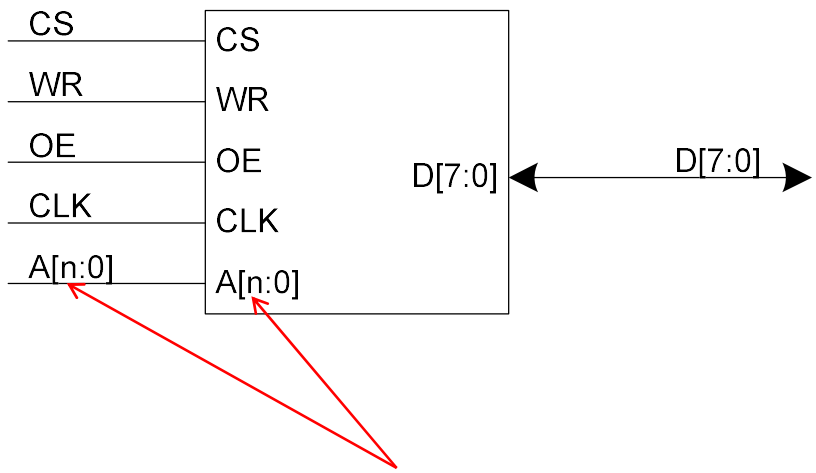
Interfacing with Memory

- Byte Accessible Memory
  - **Smallest accessible unit is 1-byte (8-bits)**
  - **Each unit is addressable by address bit0**
    - Every address is byte aligned



Interfacing with Memory

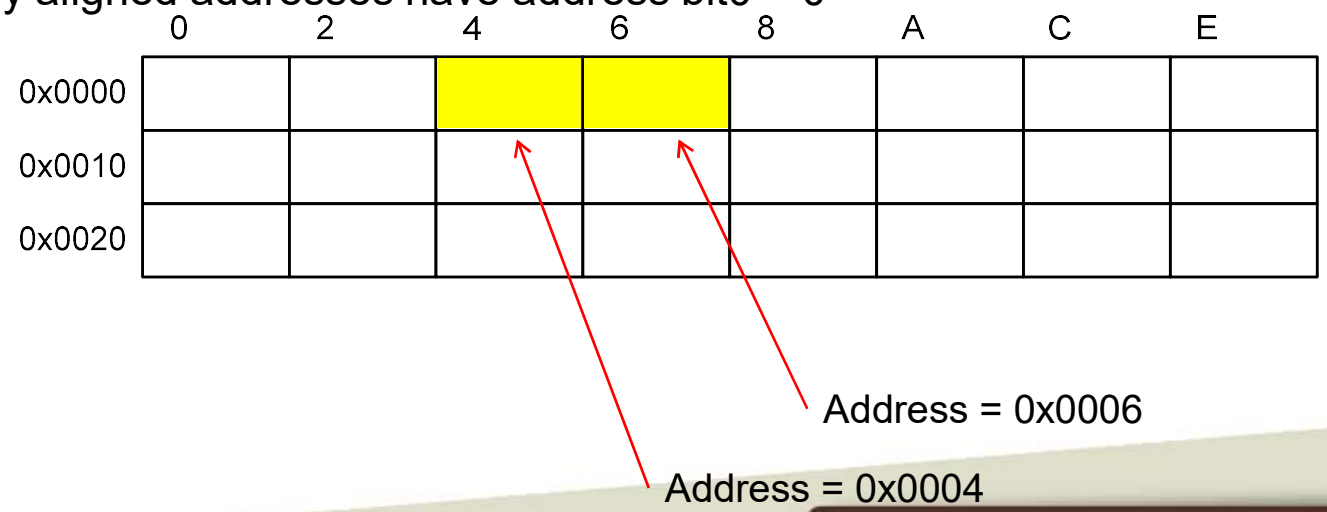
- Byte Accessible Memory
  - **Connect processor address bit0 to device bit0**



Device mapped into processor memory map

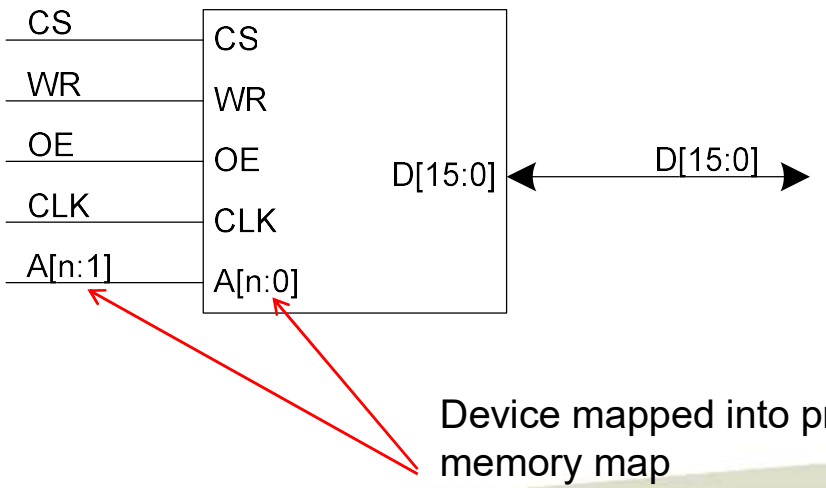
Interfacing with Memory

- 16-bit Accessible Memory
  - **Smallest accessible unit is 2-byte (16-bit)**
  - **Each unit is addressable by address bit1**
    - Bit(0) of address is always 0
      - ***Could be used to decode byte within the 2-byte word***
    - Properly aligned addresses have address bit0 = 0



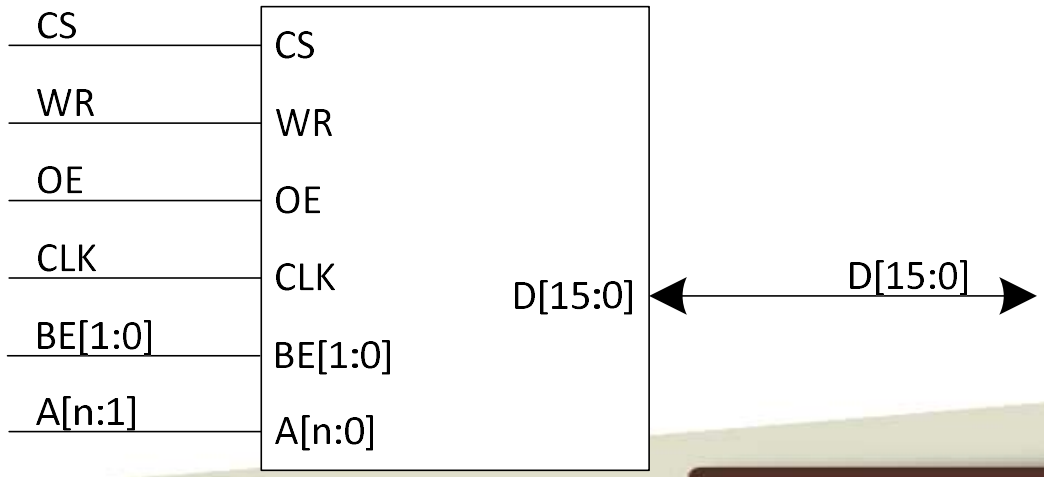
Interfacing with Memory

- 16-bit Accessible Memory
  - **Connect processor address bit1 to device bit0**
    - Device only provide 16-bit accesses
    - Processor address bit0 is always 0
      - ***Address bit (0) is not used in memory addressing***



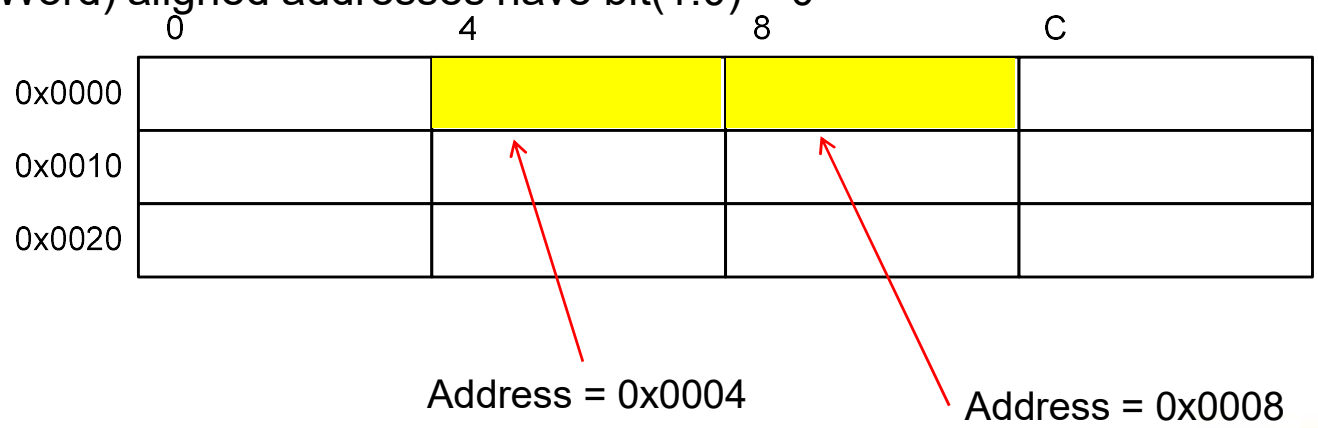
Interfacing with Memory

- 16-bit Accessible Memory w/ Byte Enable
  - **Connect processor address bit1 to device bit0**
    - Processor address bit0 is always 0
      - ***Address bit (0) is not used in memory addressing***
    - Byte Enable signals used to control access to each byte in Dout
      - ***Each byte lane has its own Byte Enable***



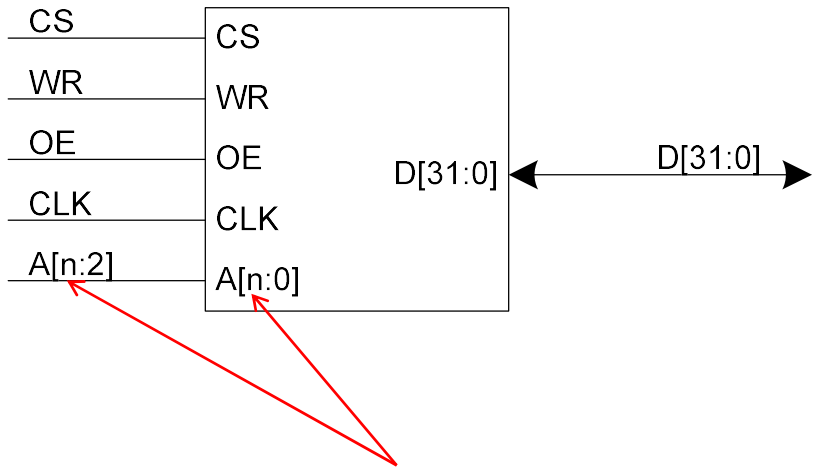
Interfacing with Memory

- 32-bit Accessible Memory
  - **Smallest accessible unit is 4-bytes (32-bit)**
  - **Each unit is addressable by address bit2**
    - Bit(1:0) of address are always 0
      - ***Two bits could be used to select byte within 4 bytes in Word (but we are not)***
    - 32-bit (Word) aligned addresses have bit(1:0) = 0



Interfacing with Memory

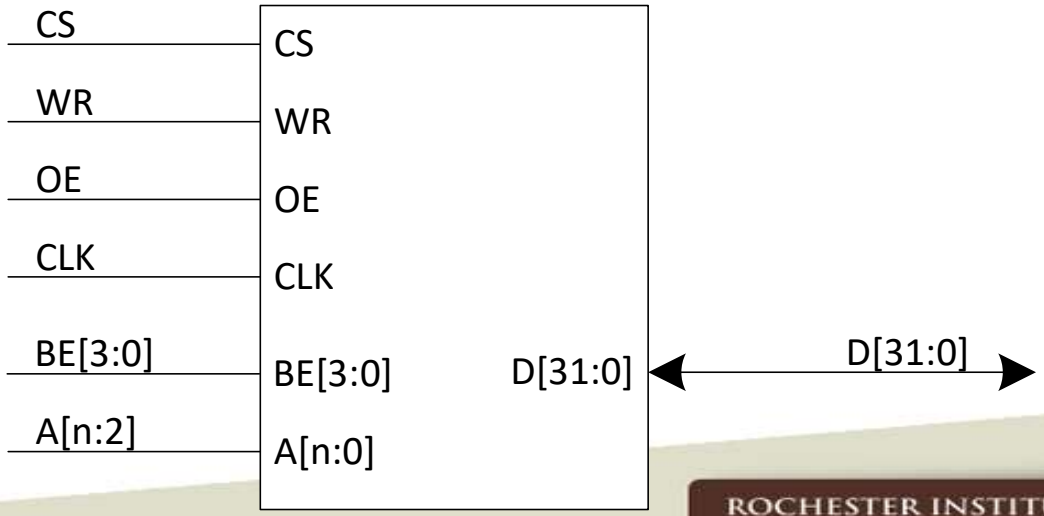
- 32-bit Accessible Memory
  - **Connect processor address bit2 to device bit0**
    - Device only provide 32-bit accesses
    - Processor address bit (1:0) are always 0
      - ***Address bits (1:0) are not used in memory addressing***



Device mapped into processor memory map

Interfacing with Memory

- 32-bit Accessible Memory w/ Byte Enable
  - **Connect processor address bit2 to device bit0**
    - Processor address bit (1:0) are always 0
      - ***Address bits (1:0) are not used in memory addressing***
    - Byte Enable signals used to control access to each byte in Dout
      - ***Each byte lane has its own Byte Enable***



Interfacing with Memory

- Timing Diagrams
  - **Representation of a set of signals in the time domain**
  - **Shows a trace through the operation of a system**
- Timing Diagram Notations
  - **Constant value**

  - **Stable Bus**

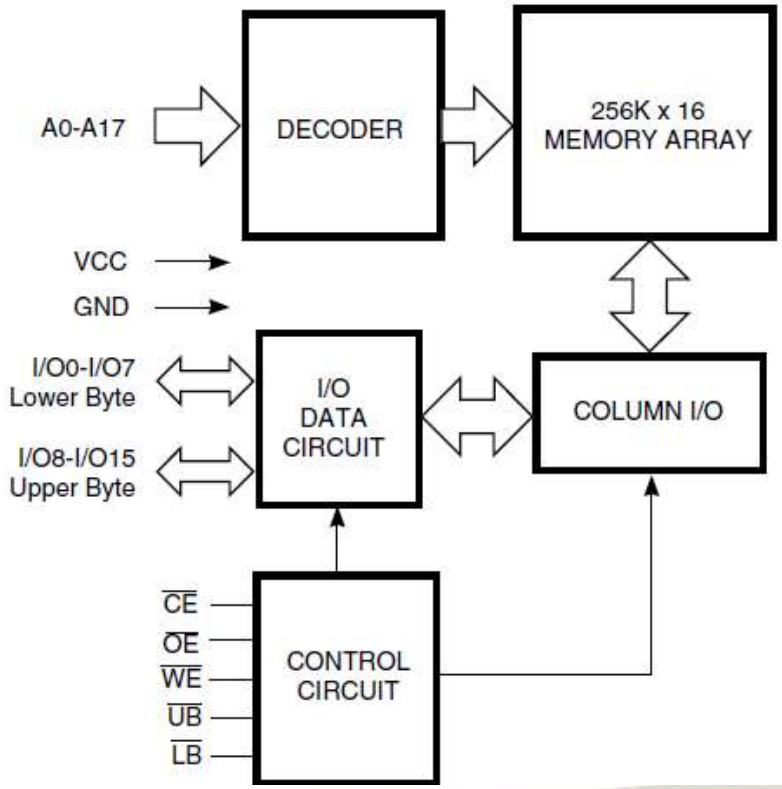
  - **Changing Value**

  - **Unknown**

Interfacing with Memory

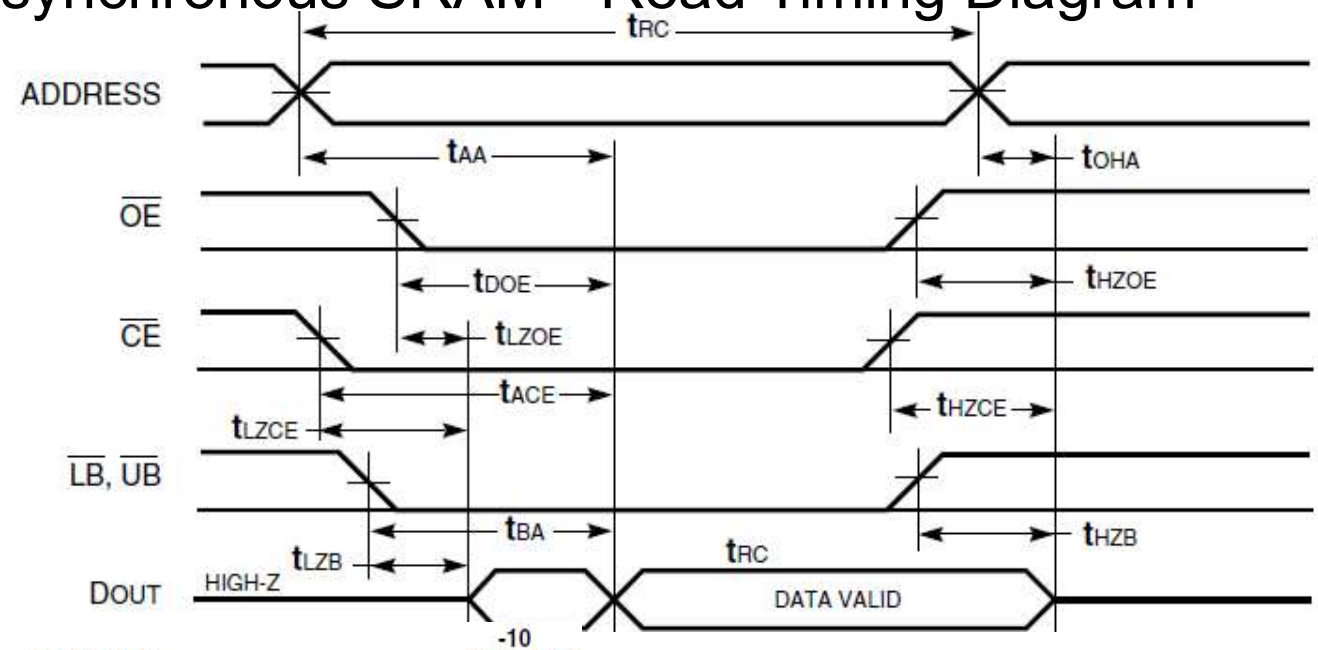
- Asynchronous SRAM
  - **256Kx16 Block Diagram**

Mode	$\overline{WE}$	$\overline{CE}$	$\overline{OE}$	$\overline{LB}$	$\overline{UB}$
Not Selected	X	H	X	X	X
Output Disabled	H	L	H	X	X
	X	L	X	H	H
Read	H	L	L	L	H
	H	L	L	H	L
	H	L	L	L	L
Write	L	L	X	L	H
	L	L	X	H	L
	L	L	X	L	L



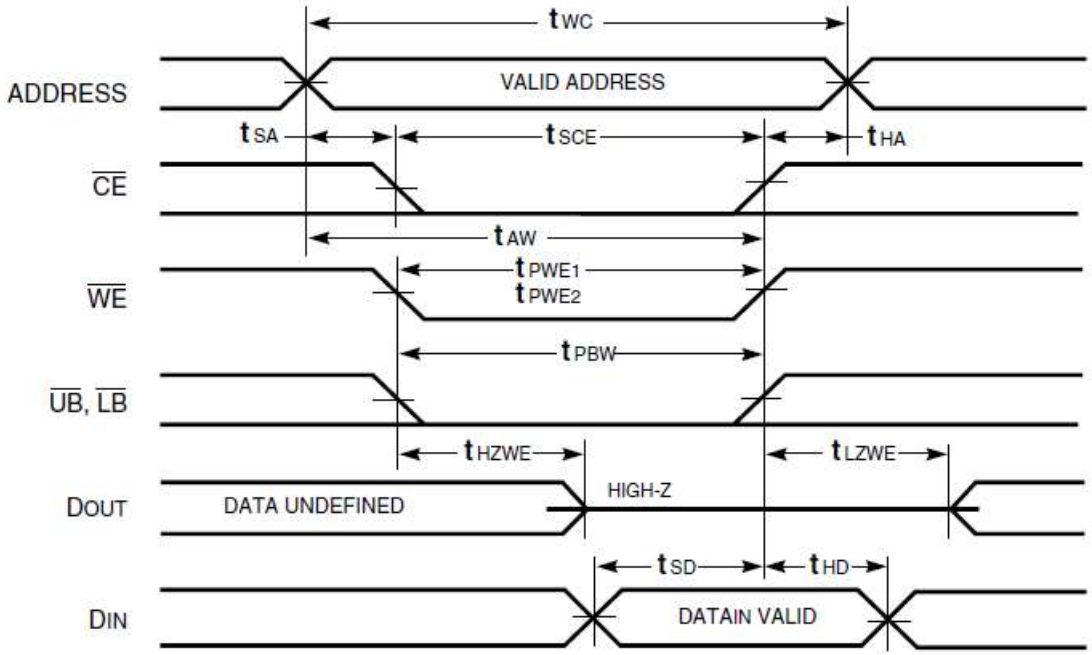
Interfacing with Memory

- Asynchronous SRAM - Read Timing Diagram



Symbol	Parameter	-10 Min.	-10 Max.
t_{RC}	Read Cycle Time	10	—
t_{AA}	Address Access Time	—	10
t_{OHA}	Output Hold Time	3	—
t_{ACE}	$\overline{CE}$ Access Time	—	10
t_{DOE}	$\overline{OE}$ Access Time	—	4
t_{HZOE}(2)	$\overline{OE}$ to High-Z Output	—	4
t_{LZOE}(2)	$\overline{OE}$ to Low-Z Output	0	—
t_{HZCE}(2	$\overline{CE}$ to High-Z Output	0	4
t_{LZCE}(2)	$\overline{CE}$ to Low-Z Output	3	—

Interfacing with Memory



- Asynchronous SRAM - Write Timing Diagram

Symbol	Parameter	Min.	Max.
t_{WC}	Write Cycle Time	10	—
t_{SCE}	$\overline{CE}$ to Write End	8	—
t_{AW}	Address Setup Time to Write End	8	—
t_{HA}	Address Hold from Write End	0	—
t_{SA}	Address Setup Time	0	—
t_{PWB}	$\overline{LB}$, $\overline{UB}$ Valid to End of Write	8	—
t_{PWE1}	$\overline{WE}$ Pulse Width	8	—
t_{PWE2}	$\overline{WE}$ Pulse Width ($\overline{OE}$ = LOW)	10	—
t_{SD}	Data Setup to Write End	6	—
t_{HD}	Data Hold from Write End	0	—