

---

# EECE 218

# Microcontrollers

Address decoder design

# Connecting devices to the external bus

---

- Processor:
  - » Data bus (D0 – D15) I/O
  - » Address bus (A0 – A15) O
  - » Control bus (E, R/~W, etc.) O
- Device
  - » Data bus (D0 – D8: Typical) (I/O)
  - » Address bus (A0 – Ax) I
  - » Control bus (typical)
    - ~CS: Chip Select: 'enables' the chip
    - ~OE: Output enable: 'chip can send data' (~RE: Read enable)
    - ~WE: Write enable: 'chip should latch data internally'

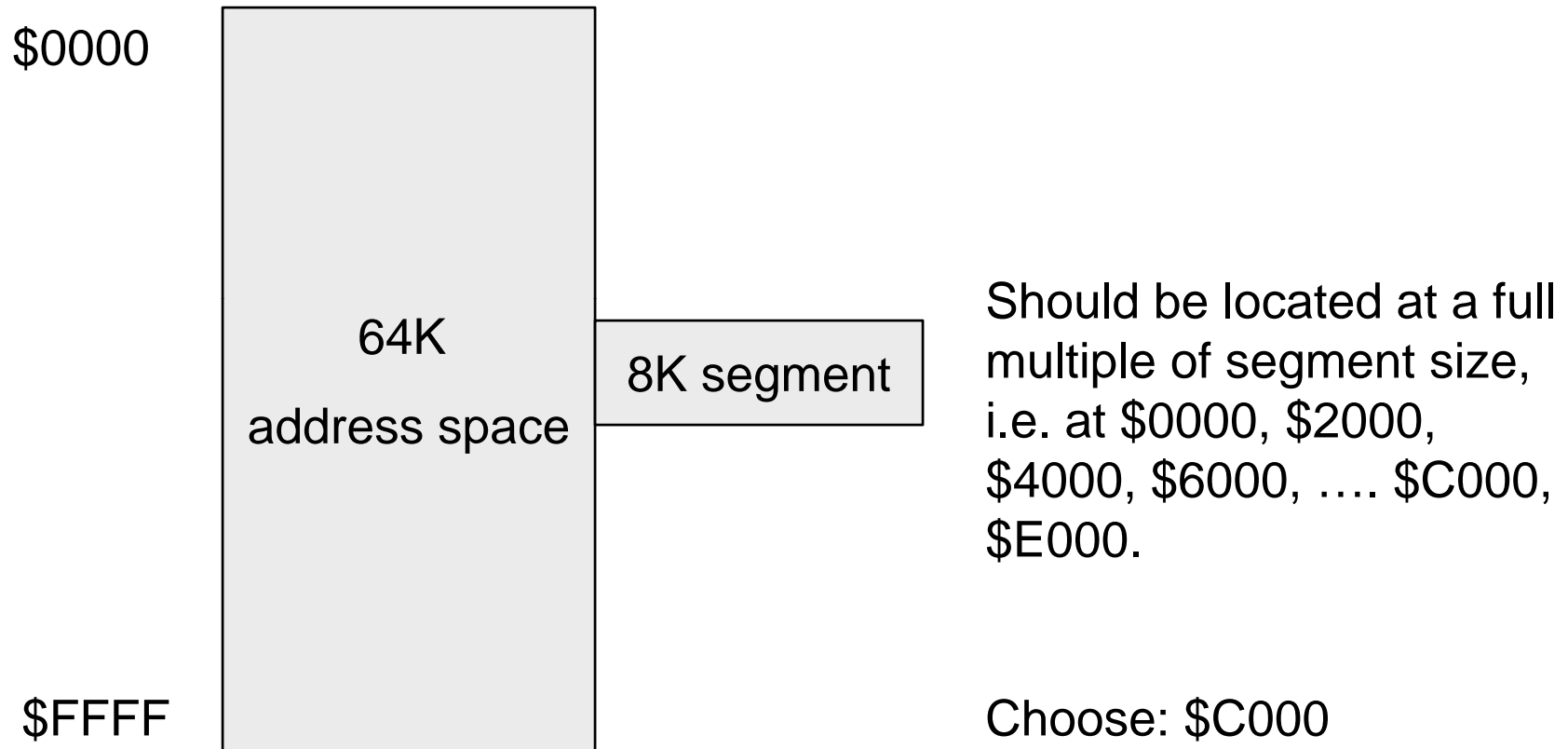
# Connecting devices to the external bus

---

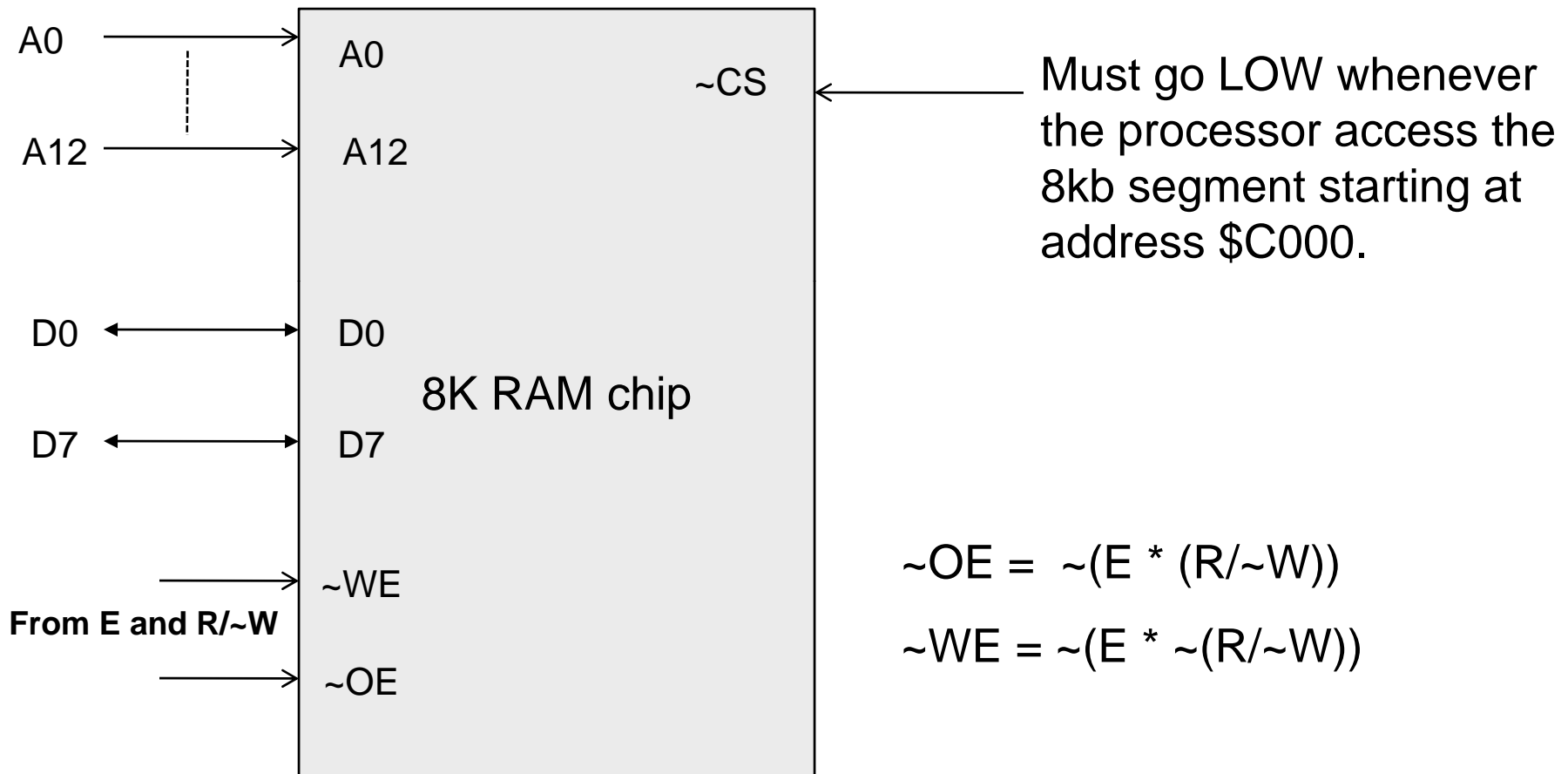
- Connections:
  - » Processor Data  $\leftrightarrow$  Device data (2 8bit chips)
  - » Processor R/ $\sim$ W  $\rightarrow$  Device R/ $\sim$ W (or equivalent)
    - On this processor:
      - E clock high means DATA ACCESS
      - Thus:  $\sim$ RE =  $\sim$ (E AND R/ $\sim$ W)  
 $\sim$ WE =  $\sim$ (E AND  $\sim$ (R/ $\sim$ W))
  - » Processor Address lower-order bits (A0 – Ax)
    - $\rightarrow$  Device address bits (A0 – Ax)
  - » Processor Address higher-order bits
    - $\rightarrow$  **Address decoder**  $\rightarrow$  Device  $\sim$ CS/ $\sim$ CE signals

# Example: 8Kbyte RAM

---



# Example: 8Kbyte RAM



# Example: 8Kbyte RAM

---

- Address map: a table showing the address bit combinations for each address range.

- Content:

**0/1 : if L/H range bits agree, X: otherwise**

- Example: Range: \$C000 -- \$DFFF

	A15	A14	A13	A12	A11	A10
\$C000	1	1	0	0	0	...
\$DFFF	1	1	0	1	1	....
i.e.:	1	1	0	X	X	....

i.e.: the address decoder:  $(A15 * A14 * \sim A13)$

---

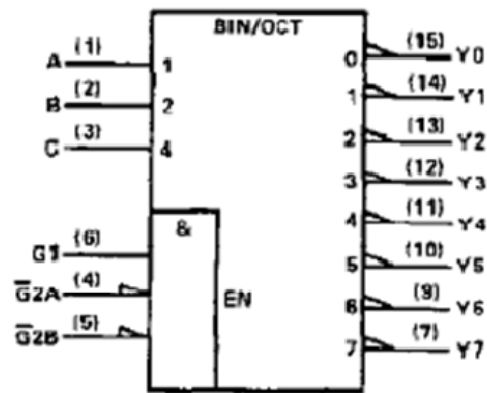
# Address decoders

---

- Rule: Lower bits of the A-bus go directly to the memory device, higher bits are used to generate the Chip Select.
- Full address decoding:
  - » All lines are used: the address decoder uses all higher-order address bits.
    - Example: 32 byte device
      - Needs 5 bits to select within the 32 byte block
      - Address decoder has 11 bits as input
  - Not economic!

# Address decoders

- Partial address ('Block') decoding:
  - » Some higher-order lines are not used in the decoding process.
- Example: Microcontroller evaluation board
  - Memory is divided into 8, 8 kbyte segments
  - One 8 k segment is divided into 8, 1kbyte segments (I/O)
- Chip to help: '138: 3-to-8-line decoder



LS138

Inputs			Outputs									
Enable		Select			Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
G1	G2*	C	B	A								
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	H	H	H	H	H	L	H	H	H
H	L	H	H	L	H	H	H	H	H	L	H	H
H	L	H	H	H	H	H	H	H	H	H	L	L

\* G2 = G2A + G2B  
 H = High Level, L = Low Level, X = Don't Care

# Address decoders

---

- Solution for the example:
  - » Use one 138 to implement the 8-way split
    - A15 → C [4]
    - A14 → B [2]
    - A13 → A [1]
    - 000 – selects:  $\sim Y_0 \rightarrow$  means: \$0000 (note 3 MSB bits!)
    - 111 – selects:  $\sim Y_7 \rightarrow$  means: \$E000
  - » Use a second 138 to implement the 2<sup>nd</sup> 8-way split
    - E.g. I/O is at \$4000-5FFF
    - [#1 138] →  $\sim Y_2$  output →  $\sim G_2$  input of [#2 138]
    - On #2 138: A12 → C / A11 → B / A10 → A

# Address decoders

---

- Useful constants:

1K = 1024 = 10 bits, 1K = \$400, \$1000 = 4K

In the example design: connect a 32-byte I/O device to  $\sim Y0$  of #2 138

Device is mapped to \$4000 --- \$43FF

**BUT: A9 --- A5 bits are NOT used in the decoding!**

Effect: the setting of these address bits do not change whether the device is selected or not. The device is selected regardless of these bits  
→ the device registers *appear* at multiple addresses.

***E.g. \$4000 → \$4020 → \$4040 → .... → \$43E0***

# Systematic design of address decoders

- Process:
  - » Given: memory layout
  - » Result: address decoding circuit
- Key: address map
- Example:

Signal:	Starts at:	Size:
ROM1	\$E000	8K
ROM2	\$C000	8K
RAM1	\$A000	4K
RAM2	\$8000	4K
IO1	\$2000	256
IO2	\$2100	256
IO3	\$2200	256
IO4	\$2300	256

LS138

Inputs					Outputs							
Enable		Select										
G1	G2*	C	B	A	Y0	Y1	Y2	Y3	Y4	Y5	Y6	Y7
X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	H	H	H	H	H	L	H	H	H	H
H	L	H	L	L	H	H	H	H	L	H	H	H
H	L	H	L	H	H	H	H	H	H	L	H	H
H	L	H	H	L	H	H	H	H	H	H	L	H
H	L	H	H	H	H	H	H	H	H	H	H	L

\* G2 = G2A + G2B

H = High Level, L = Low Level, X = Don't Care

# Systematic design of address decoders

- Address map

Signal:	Start:	End:	A15	A14	A13	A12	A11	A10	A9	A8
ROM1	\$E000	\$FFFF	1	1	1	X	X	X	X	X
ROM2	\$C000	\$DFFF	1	1	0	X	X	X	X	X
RAM1	\$A000	\$AFFF	1	0	1	0	X	X	X	X
RAM2	\$8000	\$8FFF	1	0	0	0	X	X	X	X
IO1	\$2000	\$20FF	0	0	1	0	0	0	0	0
IO2	\$2100	\$21FF	0	0	1	0	0	0	0	1
IO3	\$2200	\$22FF	0	0	1	0	0	0	1	0
IO4	\$2300	\$23FF	0	0	1	0	0	0	1	1

# Systematic design of address decoders

- Circuit:

