14:332:231
DIGITAL LOGIC DESIGN

Ivan Marsic, Rutgers University
Electrical & Computer Engineering
Fall 2013

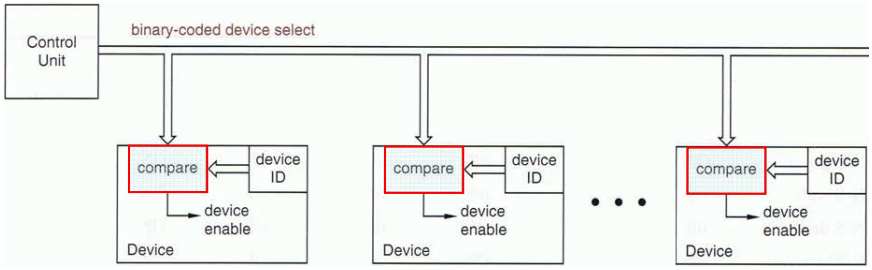
Lecture #13: Digital Comparators

Digital Comparators

- **Comparator**: A circuit that compares two binary words and indicates whether they are equal
- **Magnitude comparator**: Interprets its inputs as signed or unsigned numbers and indicates their arithmetic relationship (greater or less than)

Example Comparator Use

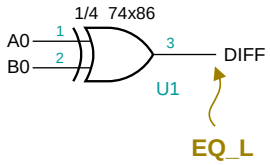
- Devices are enabled by comparing a “device select” word with a predetermined “device ID”



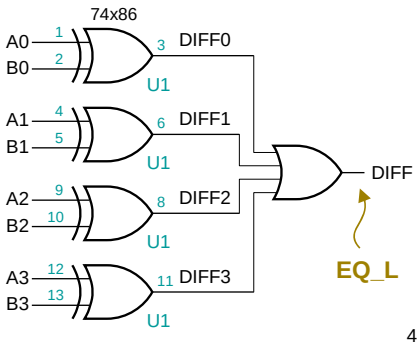
3 of 12

Equality Comparators

- 1-bit comparator
  - Active-high output (DIFF) asserted if the inputs are different



- 4-bit comparator
  - The DIFF output is asserted if *any* of the input pairs are different



4 of 12

4-Bit Magnitude Comparator

- Two input numbers to compare, 4 bits each: **A**=$A_3A_2A_1A_0$; **B**=$B_3B_2B_1B_0$
- Three outputs, reporting “greater than”, “less than”, and “equal”, respectively

Compared Inputs				*Outputs*		
A3, B3	**A2, B2**	**A1, B1**	**A0, B0**	**“A > B”**	**“A < B”**	**“A = B”**
A3 > B3	x	x	x	1	0	1
A3 < B3	x	x	x	0	1	0
A3 = B3	A2 > B2	x	x	1	0	0
A3 = B3	A2 < B2	x	x	0	1	0
A3 = B3	A2 = B2	A1 > B1	x	1	0	0
A3 = B3	A2 = B2	A1 < B1	x	0	1	0
A3 = B3	A2 = B2	A1 = B1	A0 > B0	1	0	0
A3 = B3	A2 = B2	A1 = B1	A0 < B0	0	1	0
A3 = B3	A2 = B2	A1 = B1	A0 = B0	1	0	0
A3 = B3	A2 = B2	A1 = B1	A0 = B0	0	1	0
A3 = B3	A2 = B2	A1 = B1	A0 = B0	0	0	1

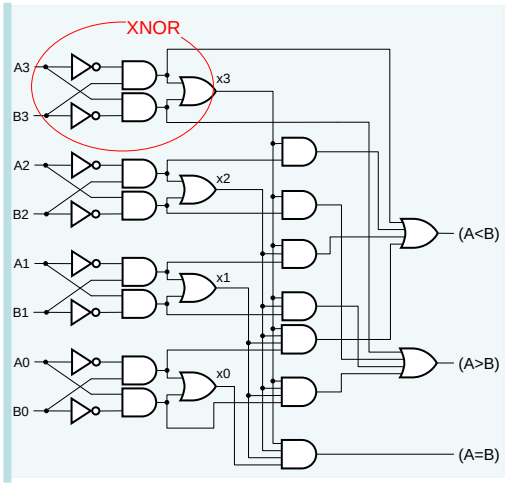
Note “x” (don’t care) notation.

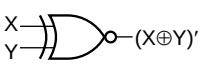
4-Bit Magnitude Comparator

- Input **A**=$A_3A_2A_1A_0$; **B**=$B_3B_2B_1B_0$
- Case **A = B** : $A_3=B_3$, $A_2=B_2$, $A_1=B_1$, $A_0=B_0$
  xi = (Ai⊕Bi)′ = Ai·Bi + Ai′·Bi′

XNOR = (Ai⊕Bi)′ = (Ai·Bi′+ Ai′·Bi)′
= (Ai′+Bi)·(Ai+Bi′)
= Ai′·Ai + Ai′·Bi′ + Ai·Bi + Bi·Bi′
= Ai·Bi + Ai′·Bi′

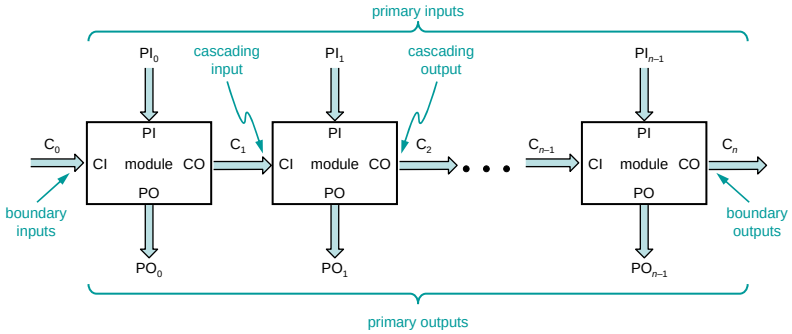
  - **Output**: x3·x2·x1·x0
- Case **A > B** :
  - **Output**:
    A3·B3′ + x3·A2·B2′ + x3·x2·A1·B1′ +
    x3·x2·x1·A0·B0′
- Case **A < B** :
  - **Output**:
    A3′·B3 + x3·A2′·B2 + x3·x2·A1′·B1+
    x3·x2·x1·A0′·B0





Iterative Combinational Circuits

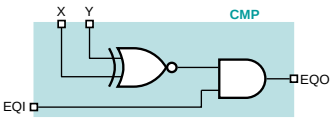
- General structure: *n* identical modules
  - For problems that can be solved by an iterative algorithm:
    1. Set C_0 to its initial value and set *i* to 0
    2. While $i < n$ repeat:
       a) Use C_i an PI_i to determine the values of PO_i and C_{i+1}
       b) Increment *i* a



An Iterative Comparator Circuit

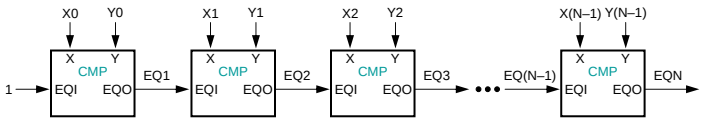
- **(a)** module for one bit
- **(b)** complete circuit
  - Comparing two *n*-bit values X and Y:
    1. Set EQ_0 to 1 and set *i* to 0
    2. While $i < n$ repeat:
       a) If EQ_i is 1 and X_i equals Y_i, set EQ_{i+1} to 1 Else set EQ_{i+1} to 0
       b) Increment *i*
- Slow because the cascading signals need time to "ripple" from left to right

(a)



$EQO = (A \oplus B)' \cdot EQI$

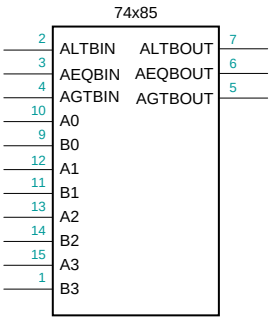
(b)



first input has to be 1

4-bit Comparator 74x85

- Outputs:
  - Greater-than output (AGTBOUT)
  - Less-than output (ALTBOUT)
  - Equal output (AEQBOUT)
- Cascading inputs:
  - AGTBIN, ALTBIN, AEQBIN
- Cascading inputs and the outputs are arranged in a 1-out-of-3 code, since normally exactly one input and output should be asserted

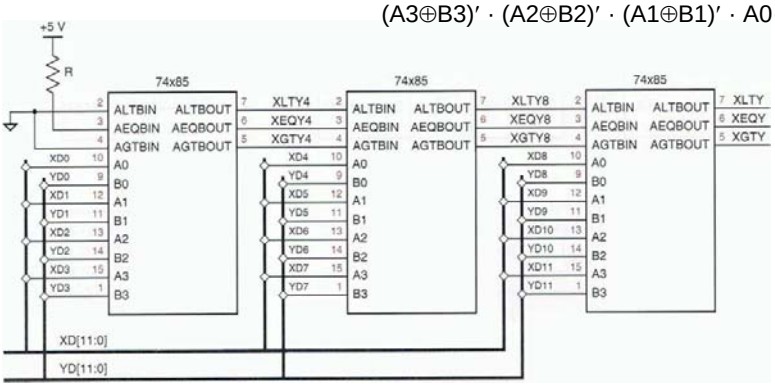


12-bit Comparator using 74x85s

AGTBOUT = (A>B) + (A=B) · AGTBIN
AGTBOUT = (A=B) · AEQBIN
AGTBOUT = (A<B) + (A=B) · ALTBIN

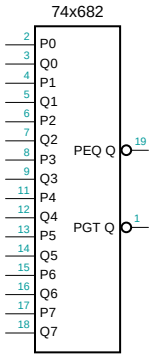
XNOR

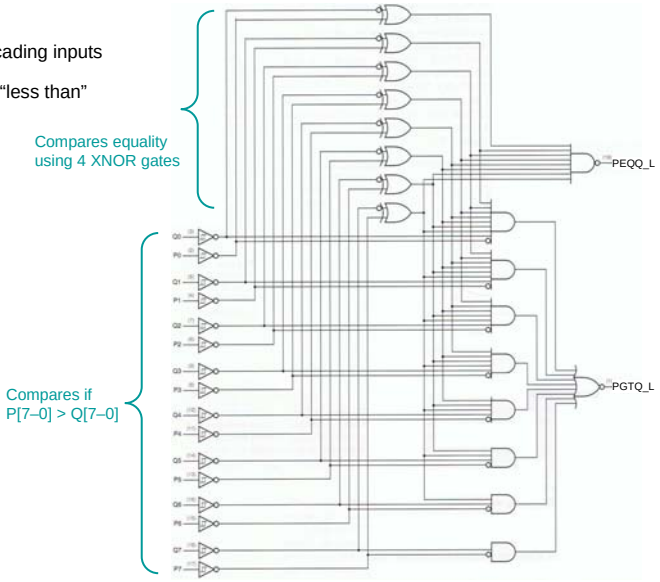
(A>B) = A3·B3′ + (A3⊕B3)′ · A2·B2′ +
(A3⊕B3)′ · (A2⊕B2)′ · A1· B1′ +
(A3⊕B3)′ · (A2⊕B2)′ · (A1⊕B1)′ · A0·B0′



8-bit Magnitude Comparator

- 74x682
  - Does not have cascading inputs (unlike 74x85)
  - Does not provide a "less than" output

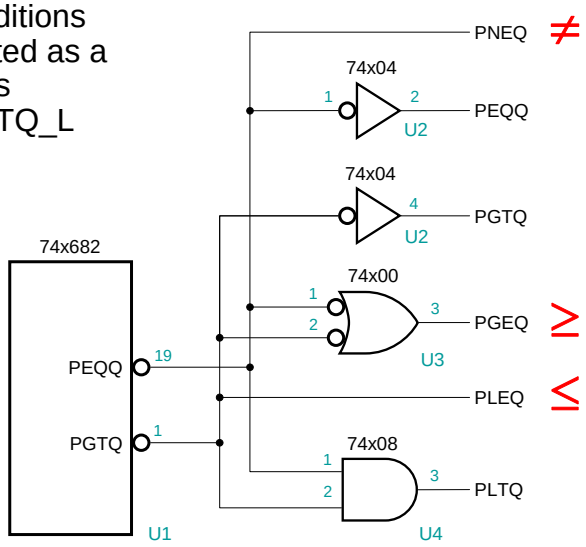




11 of 12

Arithmetic Conditions from 74x682

- Not-provided conditions can be implemented as a function of outputs PEQQ_L and PGTQ_L



12 of 12