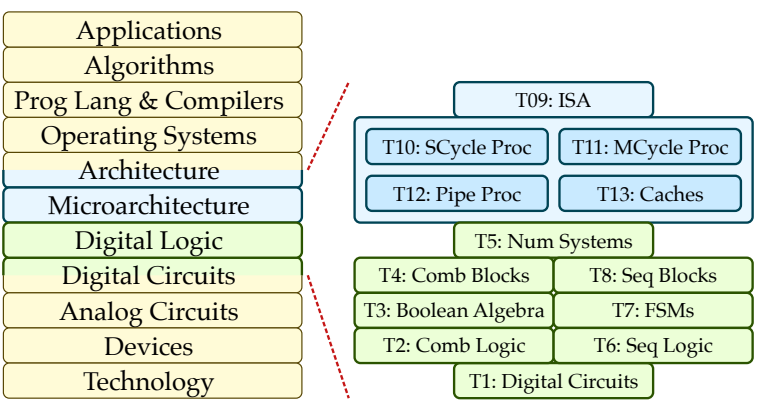
ECE 2300 Digital Logic and Computer Organization Fall 2026

Topic 4: Combinational Building Blocks

School of Electrical and Computer Engineering
Cornell University

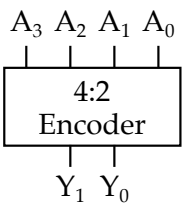
revision: 2026-09-15-10-39



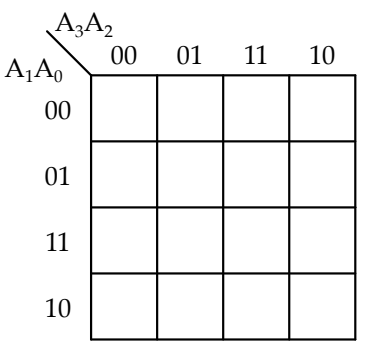
1. Encoders and Decoders

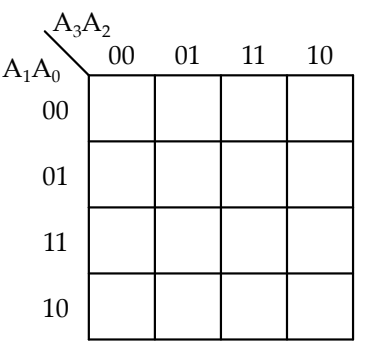
- *Binary encoder* has 2^N inputs and N outputs; sets the output value to reflect which input is one assuming exactly one input is one (encodes a one-hot value as a binary value)
- *Binary decoder* has N inputs and 2^N outputs and asserts exactly one output depending on the input value (decodes a binary value into a one-hot value)

1.1. Encoders

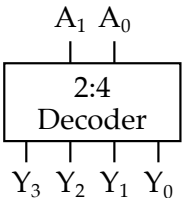


A_3	A_2	A_1	A_0	Y_1	Y_0
0	0	0	1		
0	0	1	0		
0	1	0	0		
1	0	0	0		

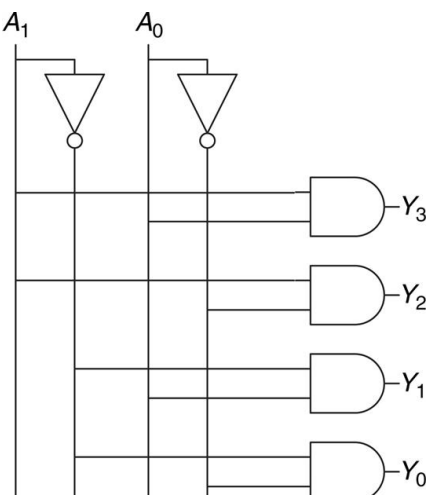




1.2. Decoders

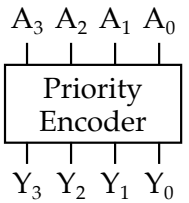


A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0				
0	1				
1	0				
1	1				



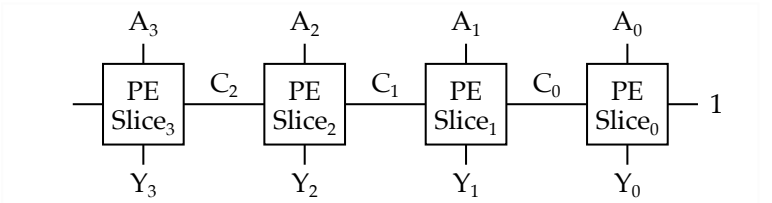
1.3. Priority Encoders

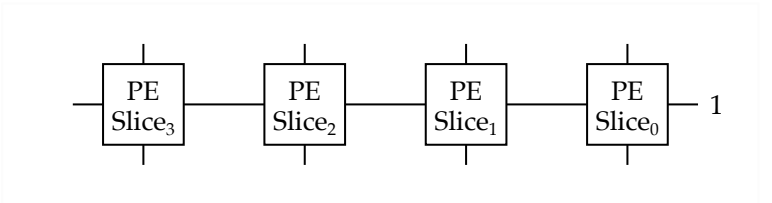
- *Priority encoder* indicates the first input that is one (i.e., the highest priority input) using a one-hot output



A_3	A_2	A_1	A_0	Y_3	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	0	1
0	1	0	0	0	1	0	0
0	1	0	1	0	0	0	1
0	1	1	0	0	0	1	0
0	1	1	1	0	0	0	1
1	0	0	0	1	0	0	0
1	0	0	1	0	0	0	1
1	0	1	0	0	0	1	0
1	0	1	1	0	0	0	1
1	1	0	0	0	1	0	0
1	1	0	1	0	0	0	1
1	1	1	0	0	0	1	0
1	1	1	1	0	0	0	1

- We can leverage modularity and regularity to manage complexity
- Implement a single *bit slice* then chain bit slices together
- Bit slice takes *continue* signal from the right and passes it to the left
  - *continue* = 1 means have not found first one, continue searching
  - *continue* = 0 means have found first one, stop searching



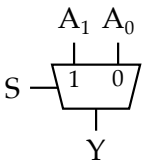


C_{in}	A	C_{out}	Y
0	0		
0	1		
1	0		
1	1		

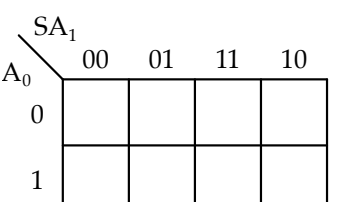
2. Multiplexors and Demultiplexors

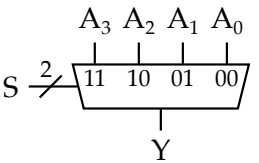
- *Multiplexor* "chooses" among several possible inputs based on the value of a *select* signal
- *Demultiplexor* "routes" an input to one of several possible outputs based on a *select* signal

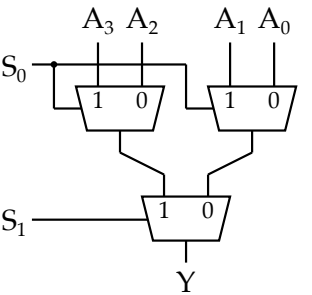
2.1. Multiplexors

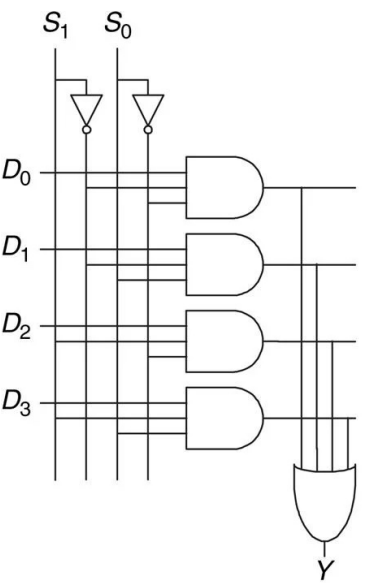


S	A_1	A_0	Y
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

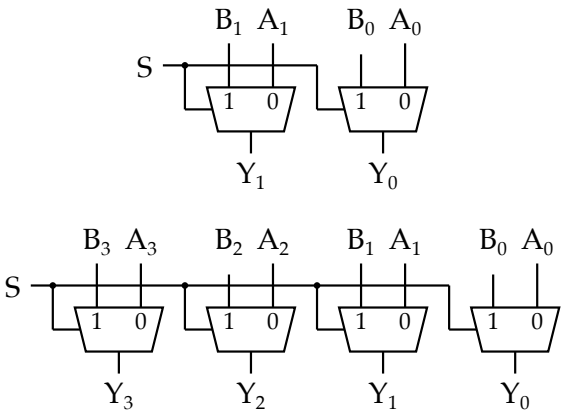



A3 A2 A1 A0
S
2
11 10 01 00
Y

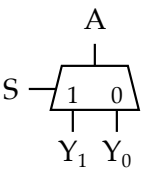

A3 A2
A1 A0
S0
1 0
1 0
1 0
S1
Y


S1 S0
D0
D1
D2
D3
Y

- Mult-bit multiplexor simply uses multiple single-bit multiplexors in parallel with a common select input



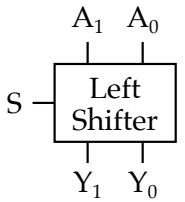
2.2. Demultiplexors

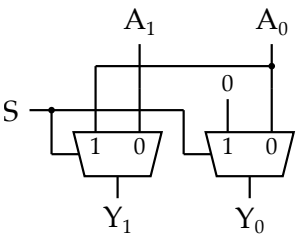


S	A	Y_1	Y_0
0	0		
0	1		
1	0		
1	1		

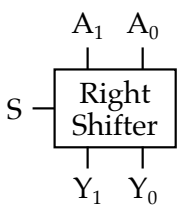
3. Shifters and Rotators

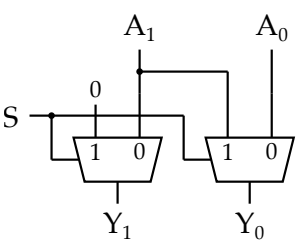
- *Shifter* shifts the input value either left or right by a variable amount specified with a *shift amount* signal; zeros are shifted in
- *Rotator* rotates the input value either left of right by a variable amount specified with a *rotate amount* signal



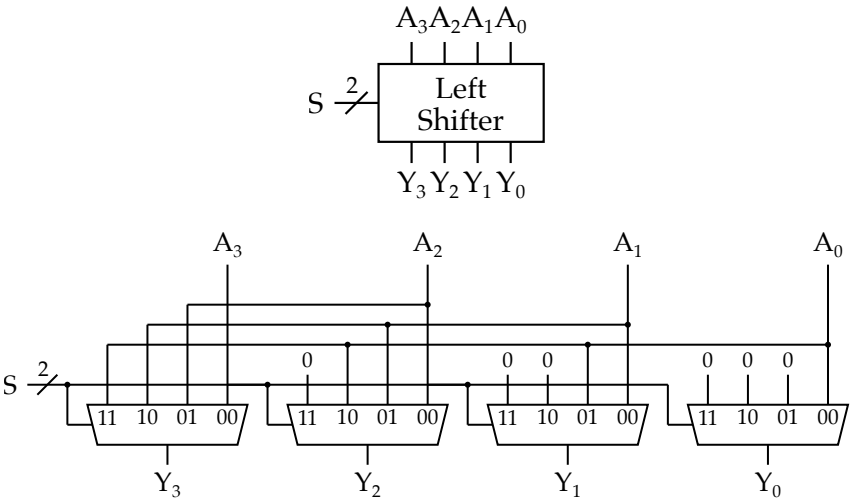


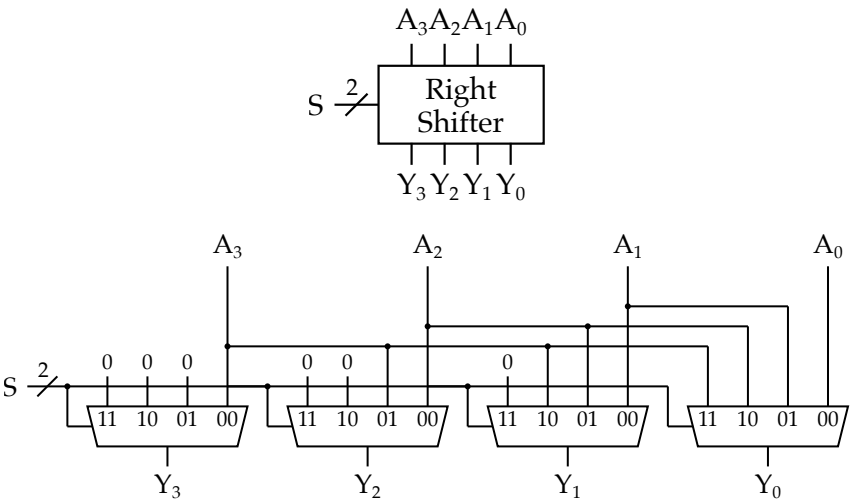
S	A_1	A_0	Y_1	Y_0
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		





S	A_1	A_0	Y_1	Y_0
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

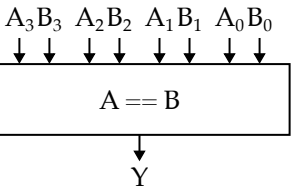

A3A2A1A0
S
2
Left
Shifter
Y3Y2Y1Y0
A3
A2
A1
A0
S
2
0
0 0
0 0 0
11 10 01 00
Y3
Y2
Y1
Y0


A3A2A1A0
S
2
Right
Shifter
Y3Y2Y1Y0
A3
A2
A1
A0
S
2
0 0 0
0 0
0
11 10 01 00
Y3
Y2
Y1
Y0

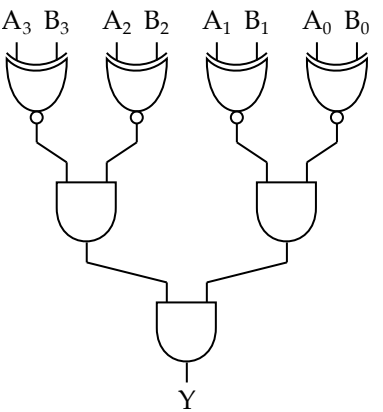
4. Comparators

- *Comparator* compares two binary numbers and outputs whether these numbers are equal, if one is greater than the other, or if one is less than the other.

4.1. Equality Comparator



A	B	Y
0	0	
0	1	
1	0	
1	1	



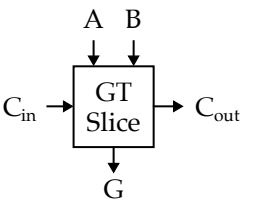
Gate	t_{pd}	**Area**
AND2	3τ	3α
XNOR2	7τ	7α

Total Area: _____

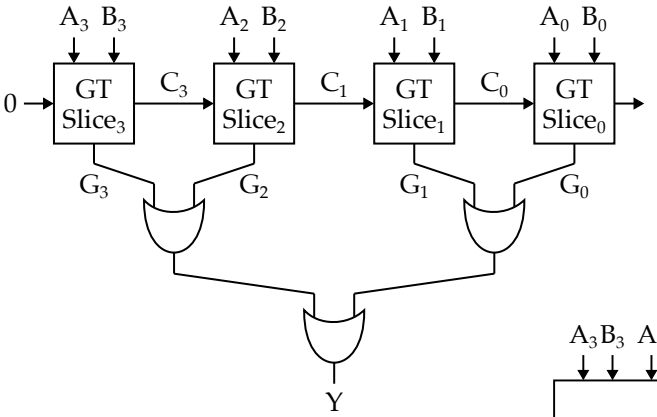
4.2. Greater-Than Comparator

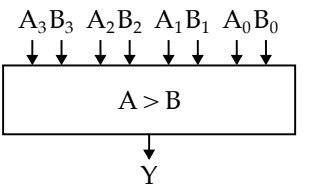
0010 (2)	0101 (5)	0111 (7)	1010 (10)	1110 (14)	1010 (10)
0001 (1)	0110 (6)	0101 (5)	1101 (13)	1011 (11)	1010 (10)
Y =	Y =	Y =	Y =	Y =	Y =

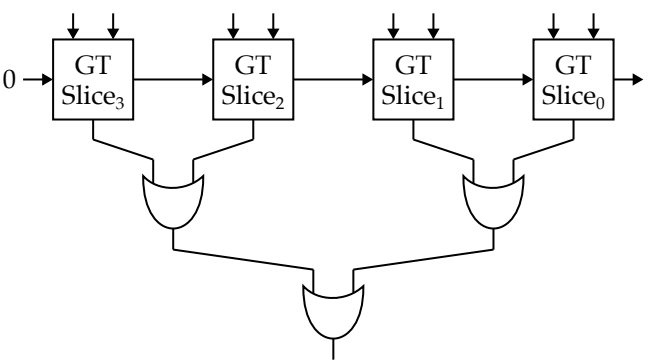
- Implement a single *bit slice* then chain bit slices together
  - *continue* = 1 means have not determined result, continue comparing
  - *continue* = 0 means have determined result, stop comparing
  - $G = 1$ means this bit slice determined that $A > B$
  - $G = 0$ means $A \leq B$ or continue comparing

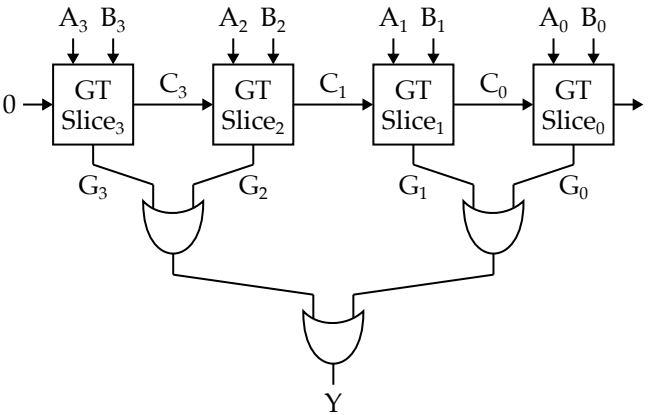


C_{in}	A	B	C_{out}	G
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		


A3 B3
A2 B2
A1 B1
A0 B0
0
GT Slice3
GT Slice2
GT Slice1
GT Slice0
C3
C1
C0
G3
G2
G1
G0
Y


A3B3 A2B2 A1B1 A0B0
A > B
Y


0
GT Slice3
GT Slice2
GT Slice1
GT Slice0



Gate	t_{pd}	Area
NOT	1τ	1α
AND2	3τ	3α
OR2	4τ	4α
XNOR2	7τ	7α

Path	Propagation Delay
$A \rightarrow G$	
$B \rightarrow G$	
$C_{in} \rightarrow G$	
$A \rightarrow C_{out}$	
$B \rightarrow C_{out}$	
$C_{in} \rightarrow C_{out}$	

Total Area: _____

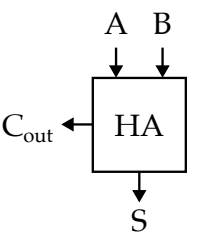
5. Adders

- We will gradually build up complexity to explore a variety of different adder designs

```
  0100      0001      0011      0011      1111
+ 1010    + 0001    + 0001    + 0011    + 0001
------    ------    ------    ------    ------
```

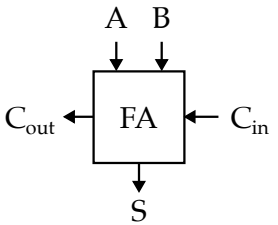
5.1. Half and Full Adders

- *Half adder* adds two one-bit numbers



A	B	C_{out}	S
0	0		
0	1		
1	0		
1	1		

- *Full adder* adds three one-bit numbers



C_{in}	A	B	C_{out}	S
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Gate	t_{pd}	Area
NOT	1τ	1α
AND2	3τ	3α
OR2	4τ	4α
XOR2	7τ	5α

Path	Propagation Delay
$A \rightarrow S$	
$B \rightarrow S$	
$C_{in} \rightarrow S$	
$A \rightarrow C_{out}$	
$B \rightarrow C_{out}$	
$C_{in} \rightarrow C_{out}$	

Total Area: _____