

CS222: Computer Architecture

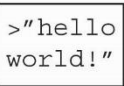


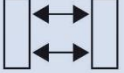
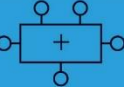
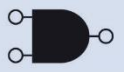
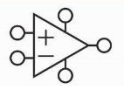


Instructors:

Dr Ahmed Shalaby <http://bu.edu.eg/staff/ahmedshalaby14#>

الاحترام - الادب - الاخلاق
الطالب - المعيد - الدكتور

Chapter 5 :: Topics

- Introduction
- Arithmetic Circuits
- Number Systems
- Sequential Building Blocks
- Memory Arrays
- Logic Arrays

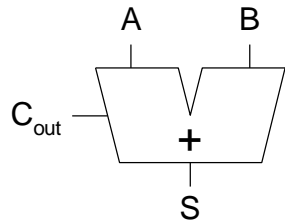
Application Software	
Operating Systems	
Architecture	
Micro-architecture	
Logic	
Digital Circuits	
Analog Circuits	
Devices	
Physics	

Introduction

- **Digital building blocks:**
 - Gates, multiplexers, decoders, registers, arithmetic circuits, counters, memory arrays, logic arrays
- **Building blocks demonstrate hierarchy, modularity, and regularity:**
 - Hierarchy of simpler components
 - Well-defined interfaces and functions
 - Regular structure easily extends to different sizes
- **Will use these building blocks in Chapter 7 to build microprocessor**

1-Bit Adders

Half Adder

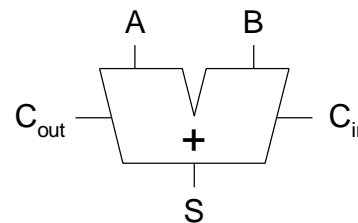


A	B	C_{out}	S
0	0		
0	1		
1	0		
1	1		

$$S = A \oplus B$$

$$C_{out} = A \cdot B$$

Full Adder



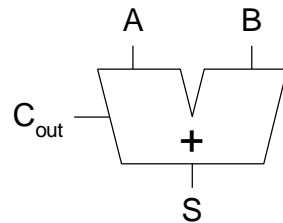
C_{in}	A	B	C_{out}	S
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

$$S = A \oplus B \oplus C_{in}$$

$$C_{out} = A \cdot B + (A \oplus B) \cdot C_{in}$$

1-Bit Adders

Half Adder

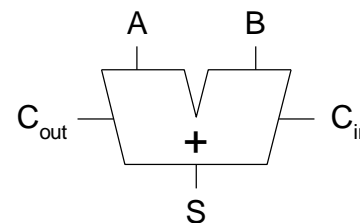


A	B	C_{out}	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$S = A \oplus B$$

$$C_{out} = A \cdot B$$

Full Adder



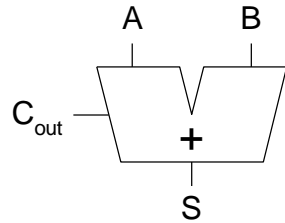
C_{in}	A	B	C_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$S = A \oplus B \oplus C_{in}$$

$$C_{out} = A \cdot B + (A \oplus B) \cdot C_{in}$$

1-Bit Adders

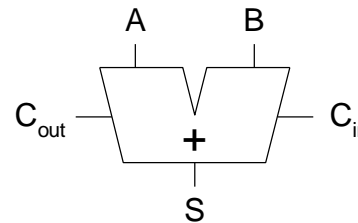
Half Adder



A	B	C _{out}	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$S = A \oplus B$$
$$C_{out} = AB$$

Full Adder



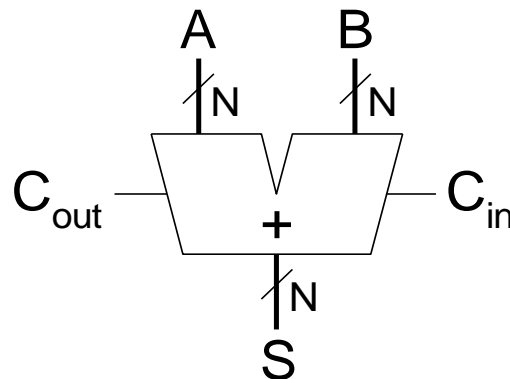
C _{in}	A	B	C _{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$S = A \oplus B \oplus C_{in}$$
$$C_{out} = AB + AC_{in} + BC_{in}$$

Multibit Adders (CPAs)

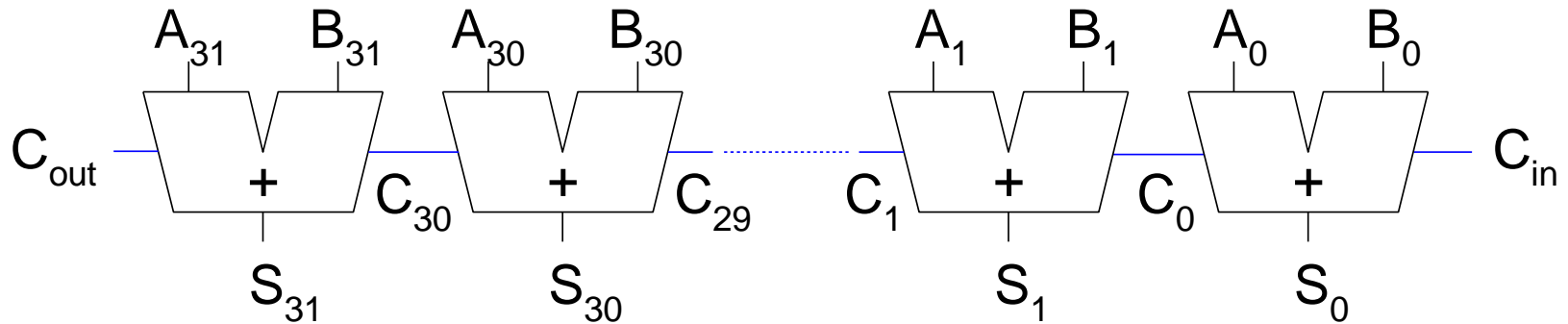
- Types of carry propagate adders (CPAs):
 - Ripple-carry (slow)
 - ~~Carry lookahead~~ (fast)
 - ~~Prefix~~ (faster)
- Carry-lookahead and prefix adders faster for large adders but require more hardware

Symbol



Ripple-Carry Adder

- Chain 1-bit adders together
- Carry ripples through entire chain
- Disadvantage: **slow**



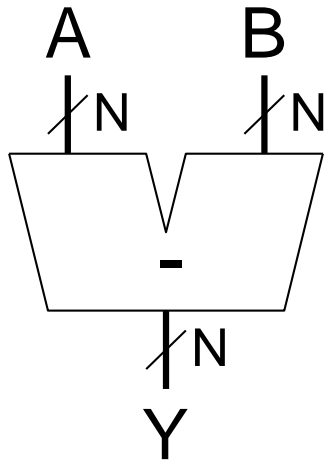
Ripple-Carry Adder Delay

$$t_{\text{ripple}} = Nt_{FA}$$

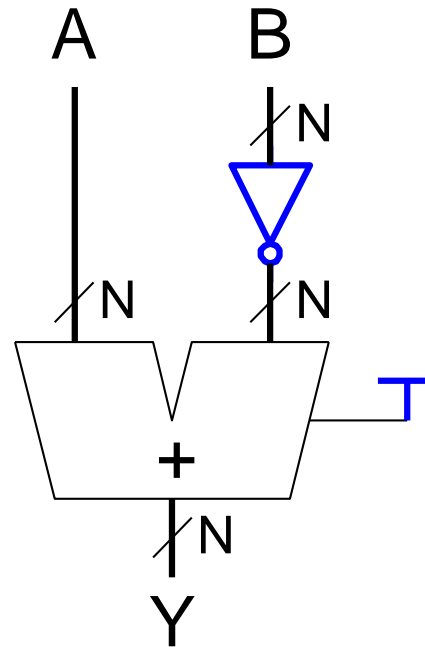
where t_{FA} is the delay of a 1-bit full adder

Subtractor

Symbol

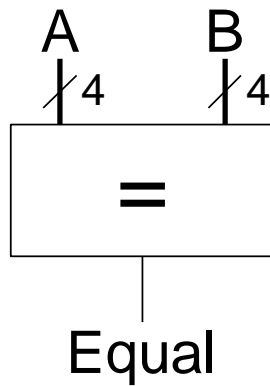


Implementation

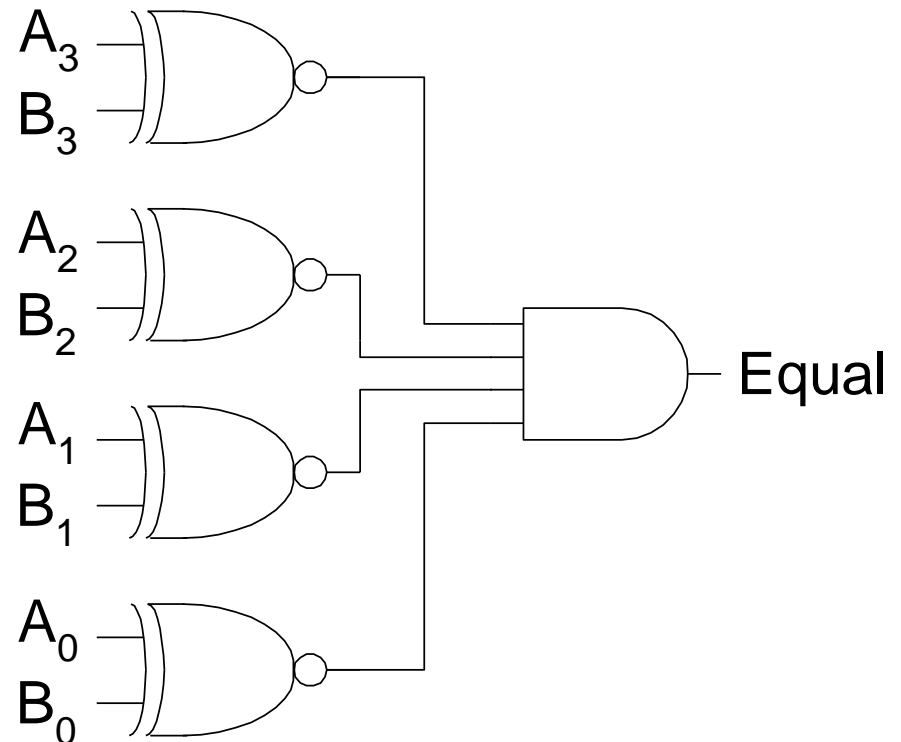


Comparator: Equality

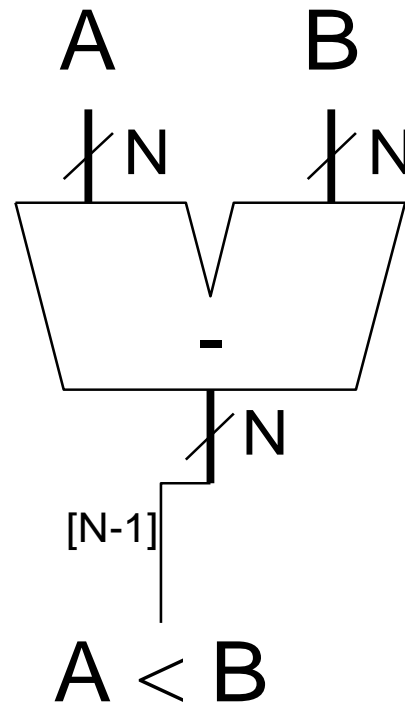
Symbol



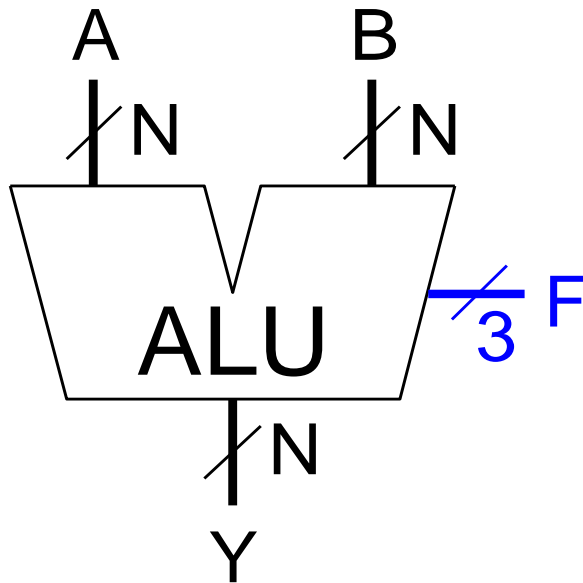
Implementation



Comparator: Less Than

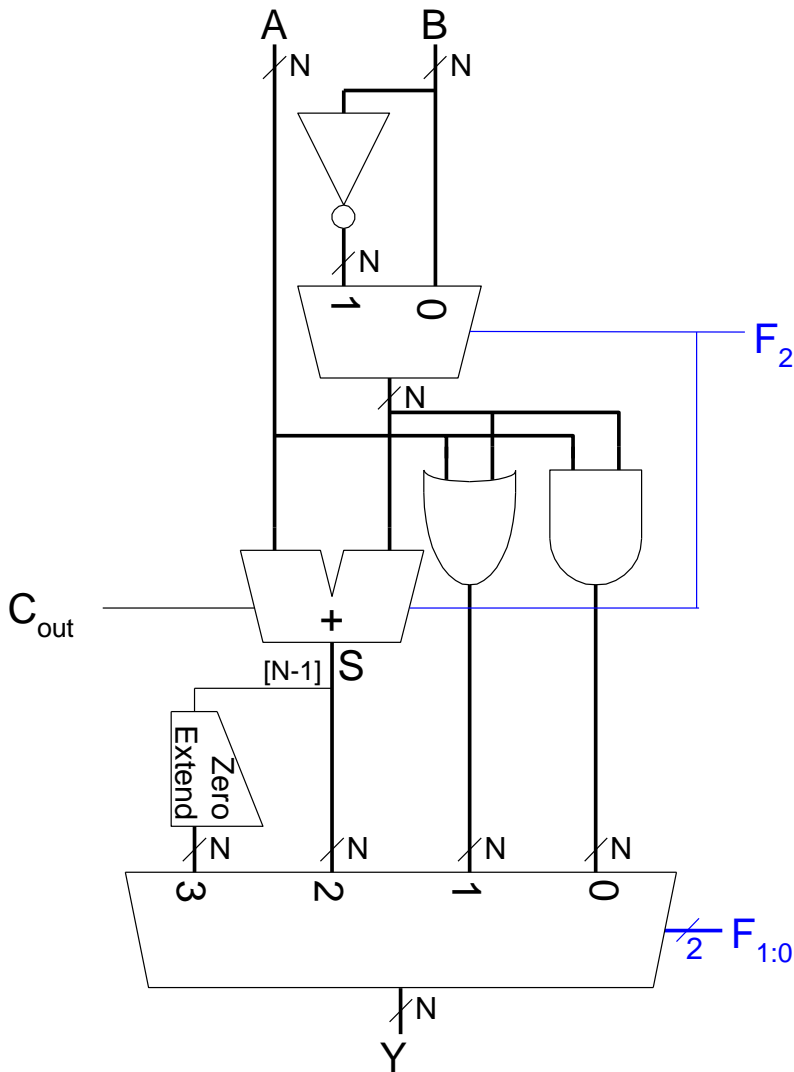


Arithmetic Logic Unit (ALU)



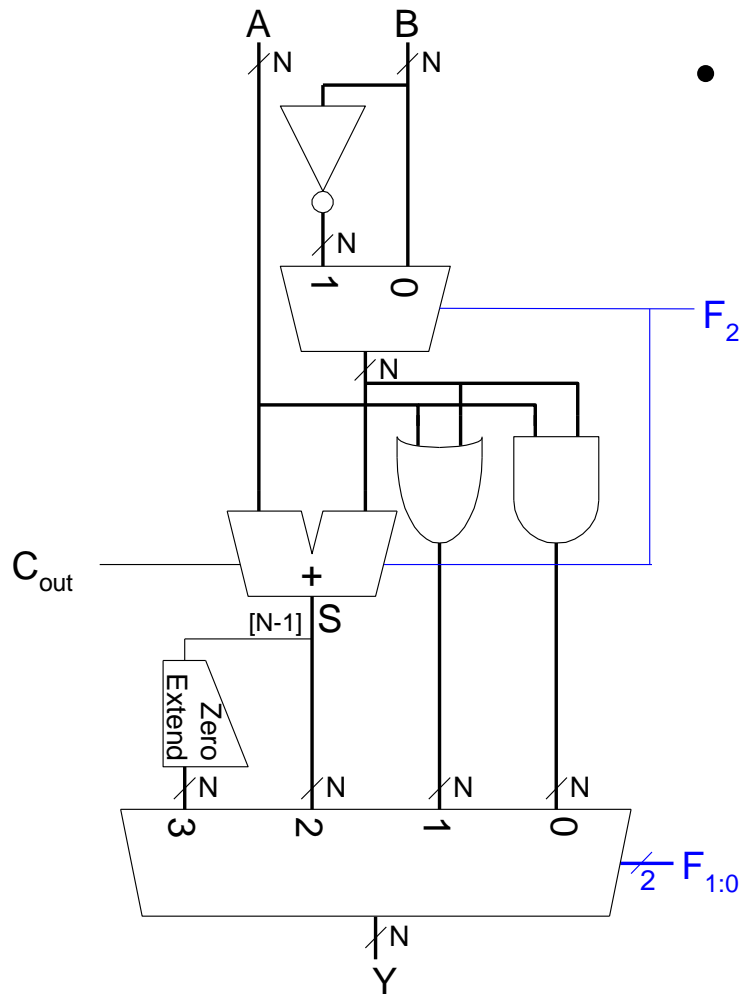
$F_{2:0}$	Function
000	$A \& B$
001	$A B$
010	$A + B$
011	not used
100	$A \& \sim B$
101	$A \sim B$
110	$A - B$
111	SLT

ALU Design



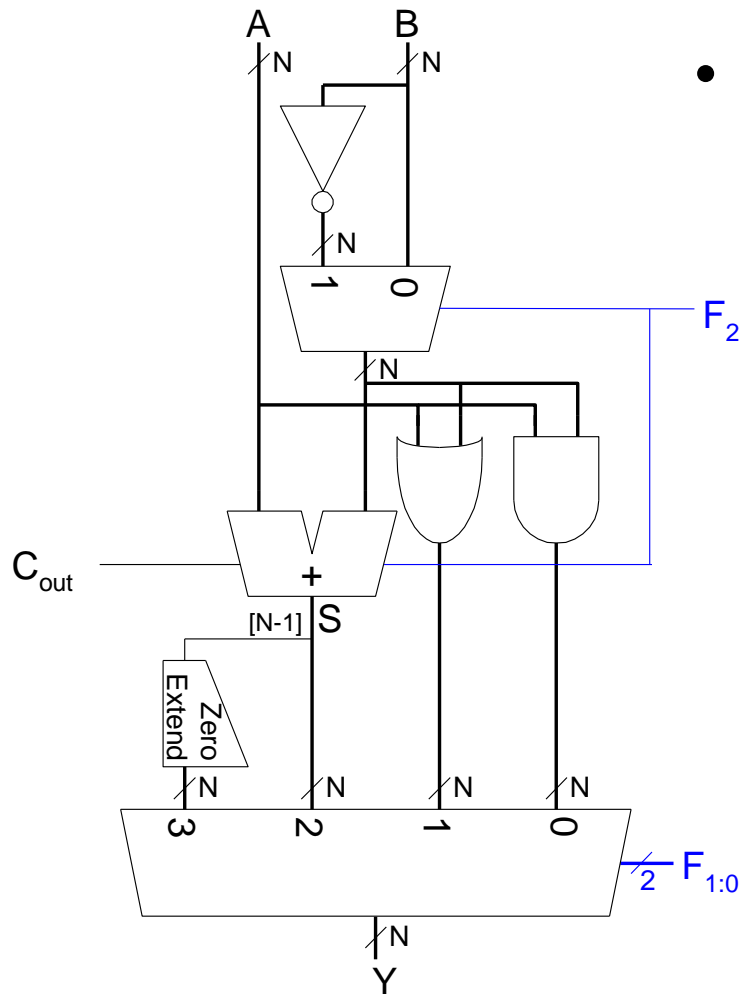
$F_{2:0}$	Function
000	$A \& B$
001	$A B$
010	$A + B$
011	not used
100	$A \& \sim B$
101	$A \sim B$
110	$A - B$
111	SLT

Set Less Than (SLT) Example



- Configure 32-bit ALU for SLT operation: $A = 25$ and $B = 32$

Set Less Than (SLT) Example



- Configure 32-bit ALU for SLT operation: $A = 25$ and $B = 32$
 - $A < B$, so Y should be 32-bit representation of 1 (0x00000001)
 - $F_{2:0} = 111$
 - $F_2 = 1$ (adder acts as subtracter), so $25 - 32 = -7$
 - -7 has 1 in the most significant bit ($S_{31} = 1$)
 - $F_{1:0} = 11$ multiplexer selects $Y = S_{31}$ (zero extended) = 0x00000001.

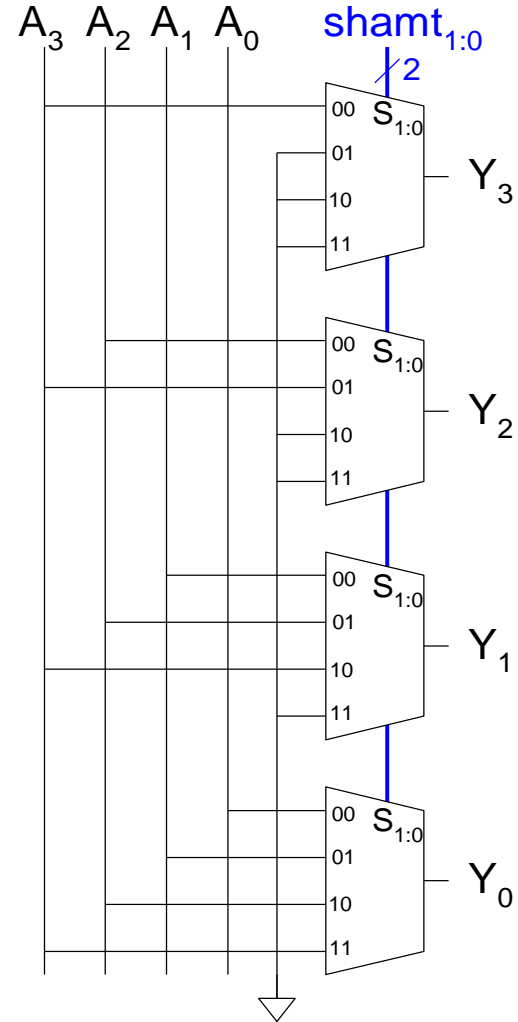
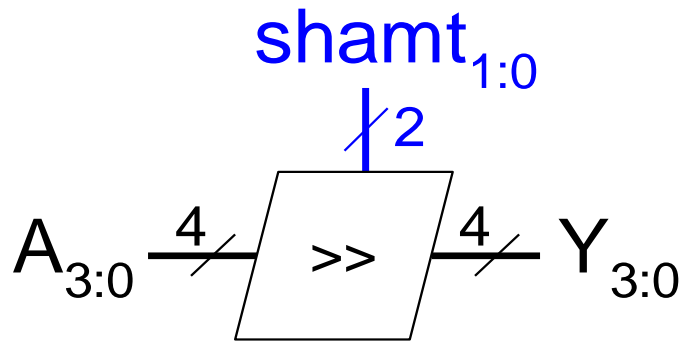
Shifters

- **Logical shifter:** shifts value to left or right and fills empty spaces with 0's
 - Ex: $11001 \gg 2 =$
 - Ex: $11001 \ll 2 =$
- **Arithmetic shifter:** same as logical shifter, but on right shift, fills empty spaces with the old most significant bit (msb).
 - Ex: $11001 \ggg 2 =$
 - Ex: $11001 \lll 2 =$
- **Rotator:** rotates bits in a circle, such that bits shifted off one end are shifted into the other end
 - Ex: $11001 \text{ ROR } 2 =$
 - Ex: $11001 \text{ ROL } 2 =$

Shifters

- **Logical shifter:**
 - Ex: 11001 >> 2 = 00110
 - Ex: 11001 << 2 = 00100
- **Arithmetic shifter:**
 - Ex: 11001 >>> 2 = 11110
 - Ex: 11001 <<< 2 = 00100
- **Rotator:**
 - Ex: 11001 ROR 2 = 01110
 - Ex: 11001 ROL 2 = 00111

Shifter Design



Shifters as Multipliers, Dividers

- $A \ll N = A \times 2^N$
 - **Example:** $00001 \ll 2 = 00100$ ($1 \times 2^2 = 4$)
 - **Example:** $11101 \ll 2 = 10100$ ($-3 \times 2^2 = -12$)
- $A \gg N = A \div 2^N$
 - **Example:** $01000 \gg 2 = 00010$ ($8 \div 2^2 = 2$)
 - **Example:** $10000 \gg 2 = 11100$ ($-16 \div 2^2 = -4$)

Multipliers

- **Partial products** formed by multiplying a single digit of the multiplier with multiplicand
- **Shifted** partial products **summed** to form result

Decimal

$$\begin{array}{r} 230 \\ \times 42 \\ \hline 460 \\ + 920 \\ \hline 9660 \end{array}$$

$$230 \times 42 = 9660$$

multiplicand
multiplier
partial
products

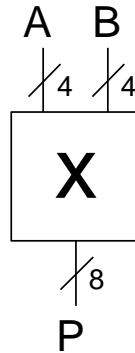
result

Binary

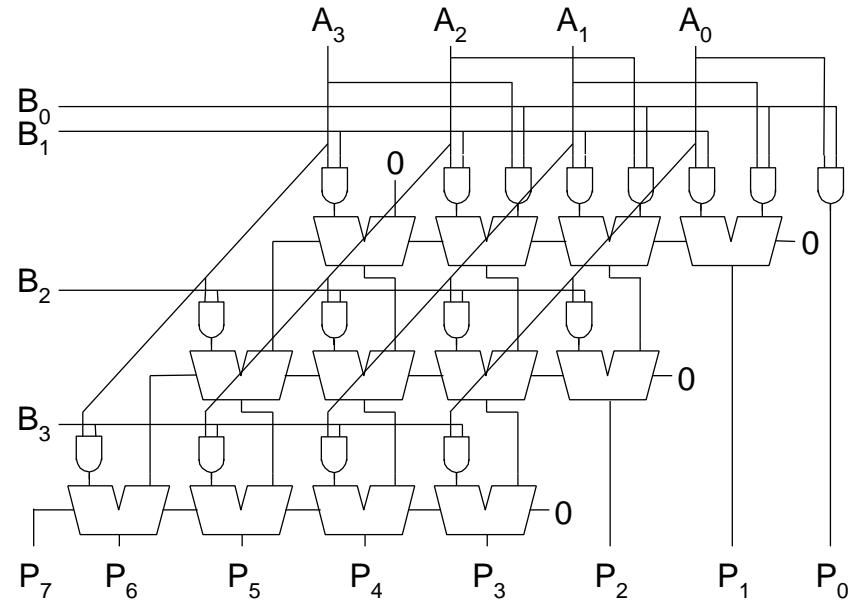
$$\begin{array}{r} 0101 \\ \times 0111 \\ \hline 0101 \\ 0101 \\ 0101 \\ + 0000 \\ \hline 0100011 \end{array}$$

$$5 \times 7 = 35$$

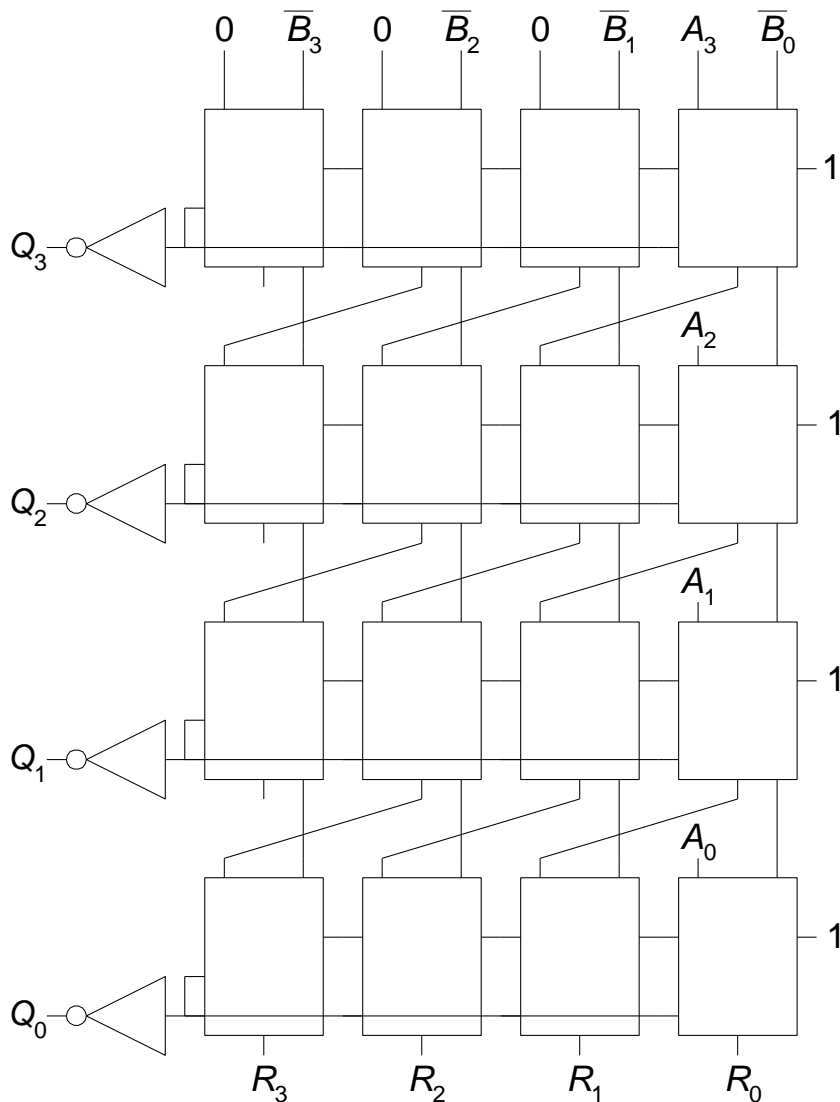
4 x 4 Multiplier



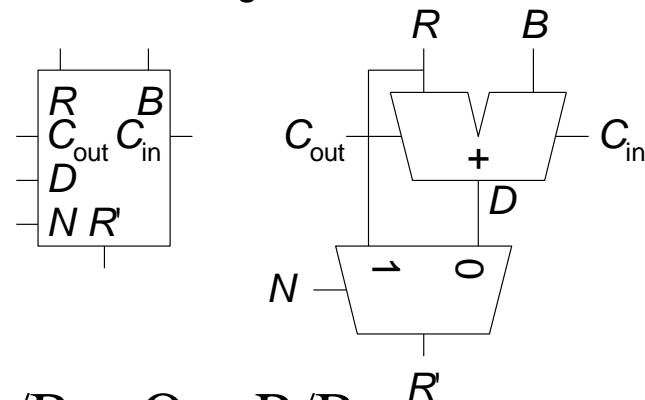
$$\begin{array}{r}
 \\
 \\
 \\
 \\
 + \\
 \hline
 P_7 \\
 P_6 \\
 P_5 \\
 P_4 \\
 P_3 \\
 P_2 \\
 P_1 \\
 P_0
 \end{array}$$



4 x 4 Divider



Legend



$$A/B = Q + R/B$$

```

R' = 0
for i = N-1 to 0
    R = {R' << 1, Ai}
    D = R - B
    if D < 0 then    Qi = 0, R' = R    // R < B
    else             Qi = 1, R' = D    // R ≥ B
R = R'
    
```



4 x 4 Divider

$$A/B = Q + R/B$$

Algorithm:

$$R' = 0$$

for $i = N-1$ to 0

$$R = \{R' \ll 1, A_i\}$$

$$D = R - B$$

if $D < 0$, $Q_i=0$, $R' = R$

else $Q_i=1$, $R' = D$

$$R' = R$$

Example $9/3 =$

$A = 9$ [1001], $B = 3$ [0011]

$A_3 = 1, A_2 = 0, A_1 = 0, A_0 = 1$

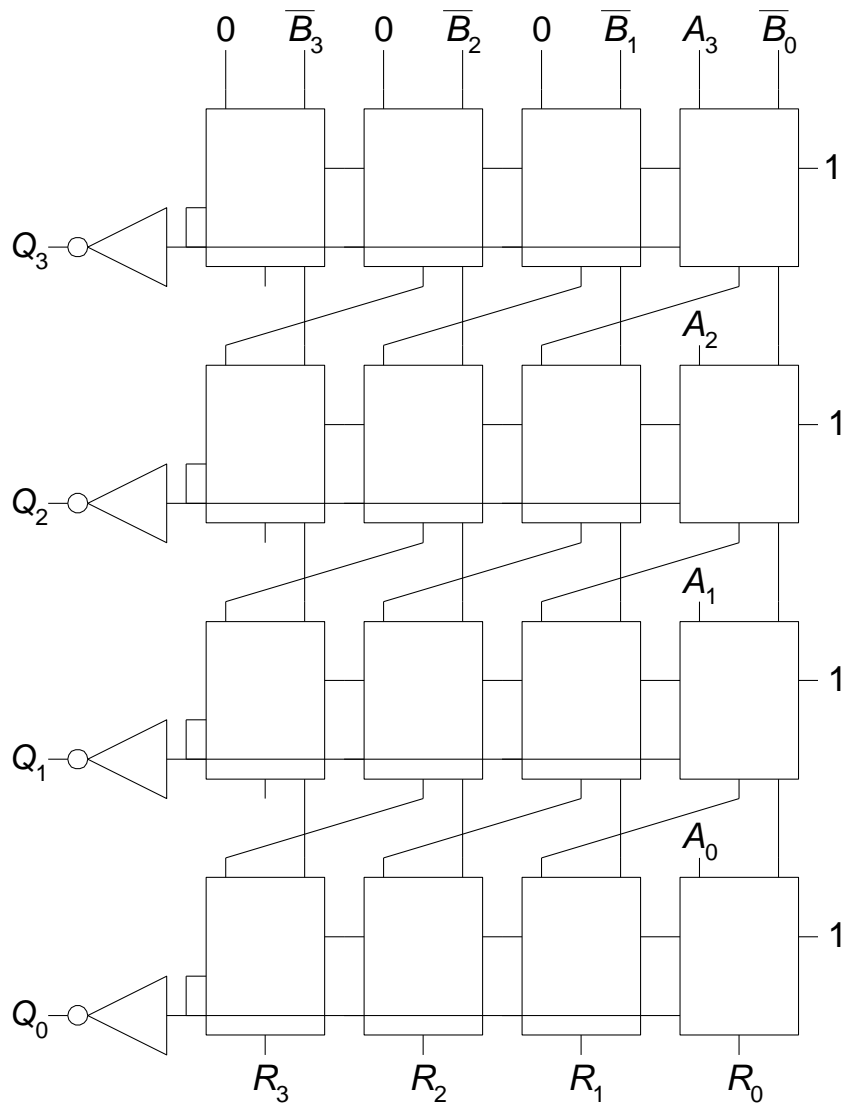
Iter (i)	R	D	Q	R'
3	{000,1}	1-3<0	0	0001
2	0010	2-3<0	0	0010
1	0100	4-3>0	1	0001
0	0011	3-3=0	1	0000

$$Q_3 = 0, Q_2 = 0, Q_1 = 1, Q_0 = 1$$

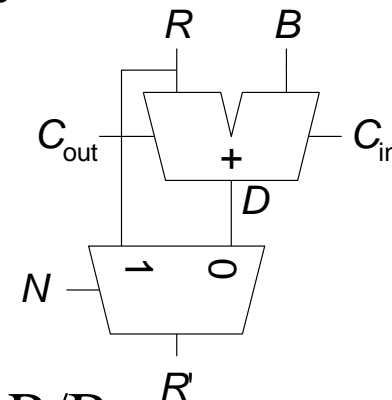
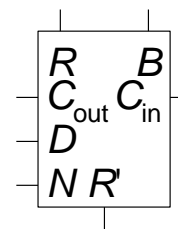
$$Q = 3 \quad R' = 0$$



4 x 4 Divider



Legend



$$A/B = Q + R/B$$

Algorithm:

$$R' = 0$$

for $i = N-1$ to 0

$$R = \{R' \ll 1, A_i\}$$

$$D = R - B$$

if $D < 0$, $Q_i = 0$, $R' = R$

else $Q_i = 1$, $R' = D$

$$R' = R$$

