

ELECTRONICS SYSTEM DESIGN

SECTION-2

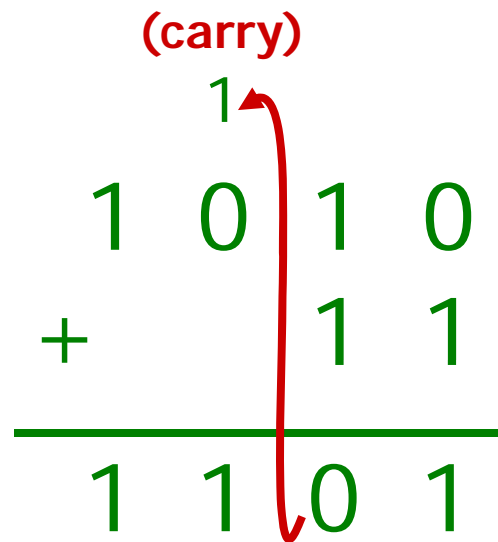
LSI AND MSI CIRCUITS AND THEIR APPLICATIONS

INTRODUCTION

- **Binary Addition**
- **Half & Full Adders**
- **Binary Subtraction**
- **Half & Full Subtractors**
- **Parallel Adders and Subtractors**
- **Using Adders for Subtraction**
- **Binary Multiplication**
- **Binary Multipliers**
- **2s Complement Notation**
- **2s Complement Adding/Subtracting**

Binary Addition

- Conceptually similar to decimal addition
- *Example:* Add the binary numbers 1010 and 11

$$\begin{array}{r} \text{(carry)} \\ 1 \\ 1010 \\ + \quad 11 \\ \hline 1101 \end{array}$$
The diagram illustrates the binary addition of 1010 and 11. The numbers are aligned by their least significant bits. A horizontal line is drawn under the second number. The result 1101 is shown below the line. A red arrow labeled '(carry)' points from the '1' in the third column (from the right) to the '0' in the second column, indicating that the sum in that column is 10, which results in a carry of 1 to the next higher column.



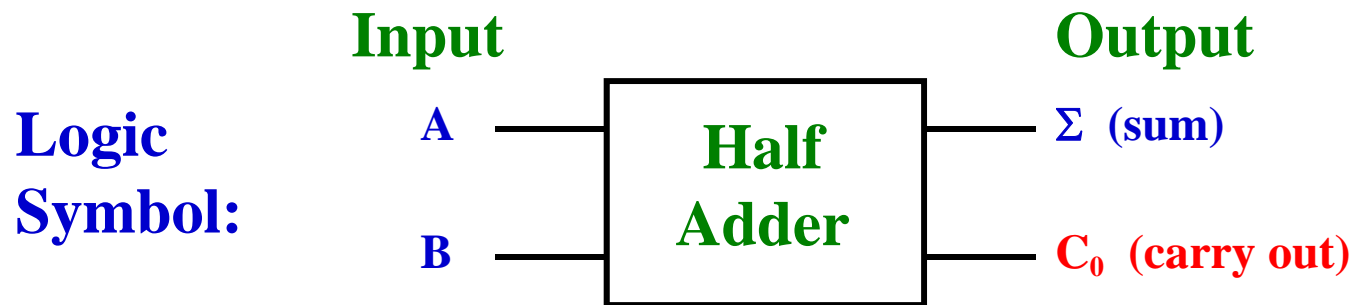
QUIZ

Add the Binary numbers 11010 and 1100

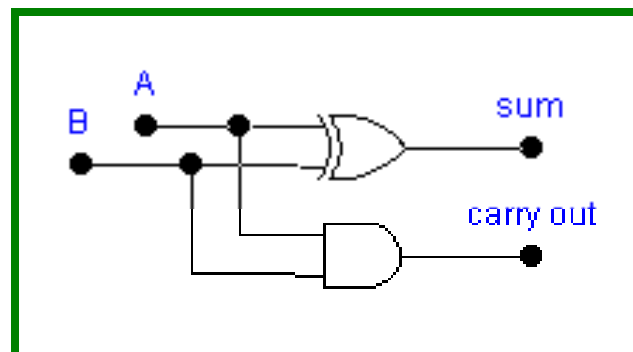
$$\begin{array}{r} \text{(carry)} \text{(carry)} \\ 1 \quad 1 \\ 1 \quad 1 \quad 0 \quad 1 \quad 0 \\ + \quad 1 \quad 1 \quad 0 \quad 0 \\ \hline 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 0 \end{array}$$

Half Adder

- Logic device that adds two binary numbers
- Only adds Least Significant Digit (LSD) column (1s column) in binary addition



Logic Diagram:

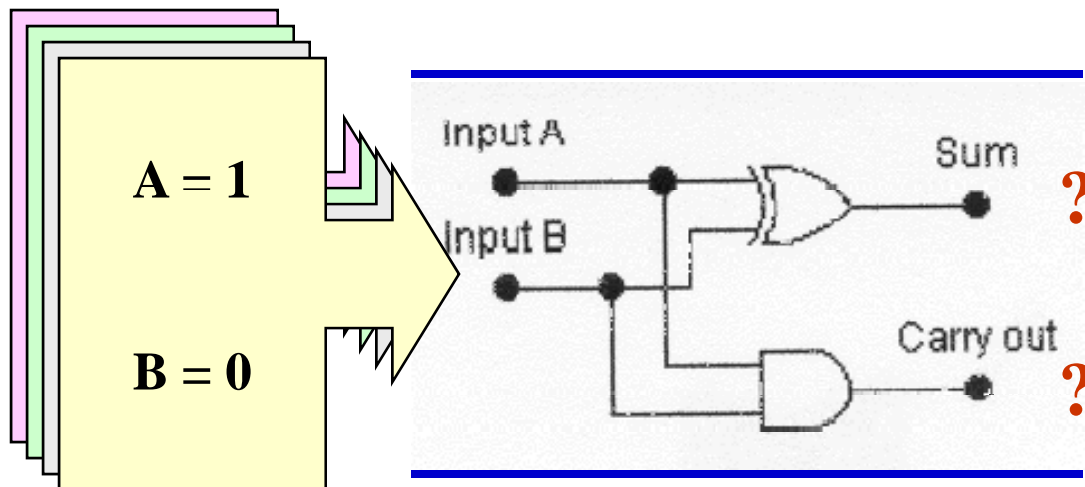




QUIZ

Q#5- What are the sum and carry out outputs from the half adder circuit?

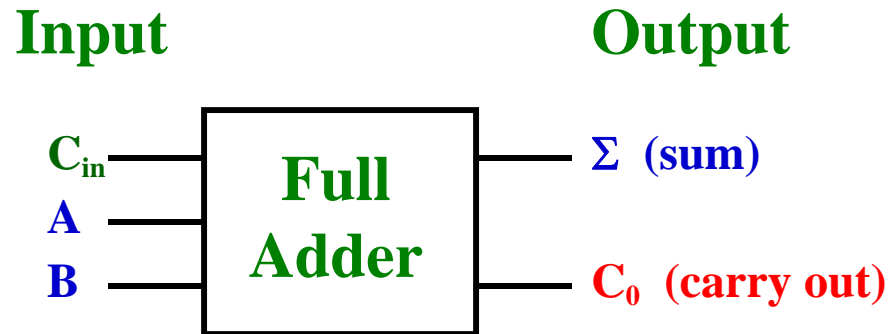
ANS: Sum=1, Carry out=0



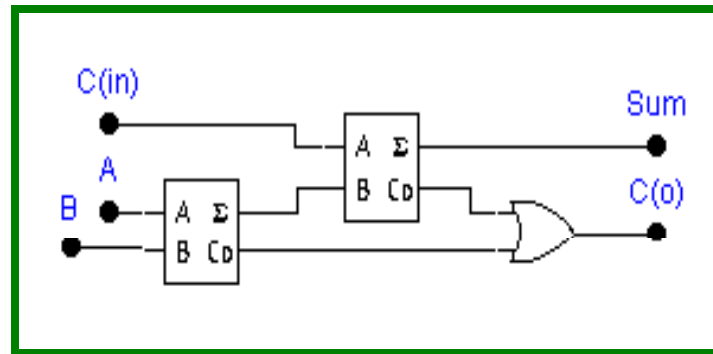
Full Adder

Used for adding binary place values other than the 1s place

Logic Symbol:



Logic Diagram:





QUIZ

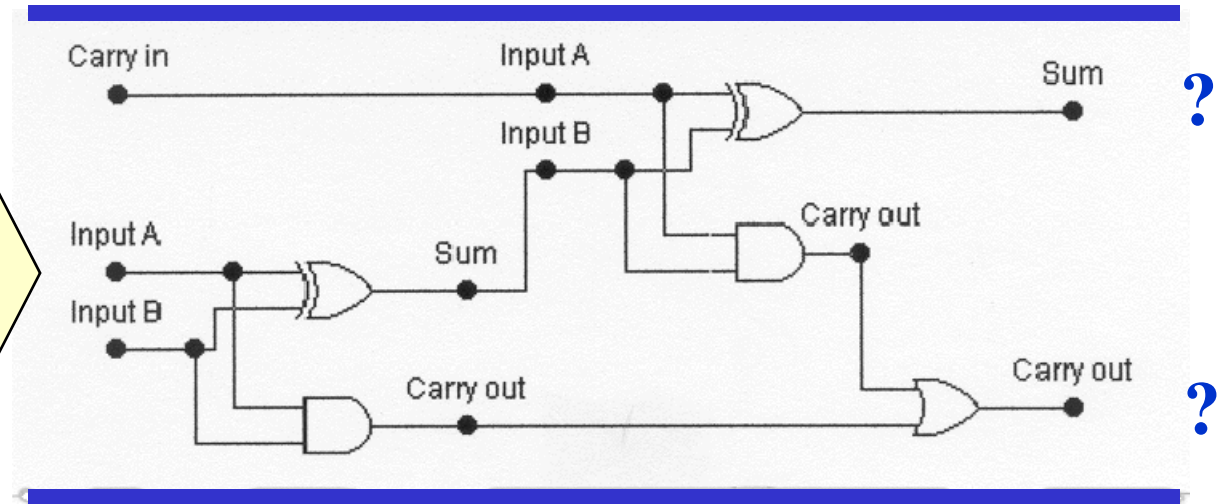
Q#6- What are the sum and carry out outputs of this full-adder circuit?

ANS: Sum=0, Carry out=1

$C_{in} = 0$

$A = 1$

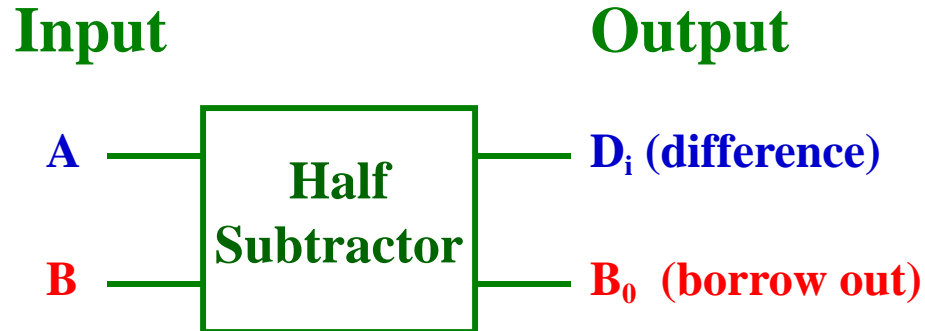
$B = 1$



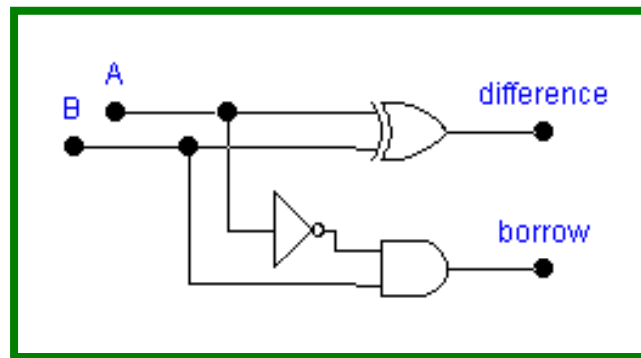
Half Subtractor

Subtracts LSD column in binary subtraction

Logic
Symbol:



Logic
Diagram:





QUIZ

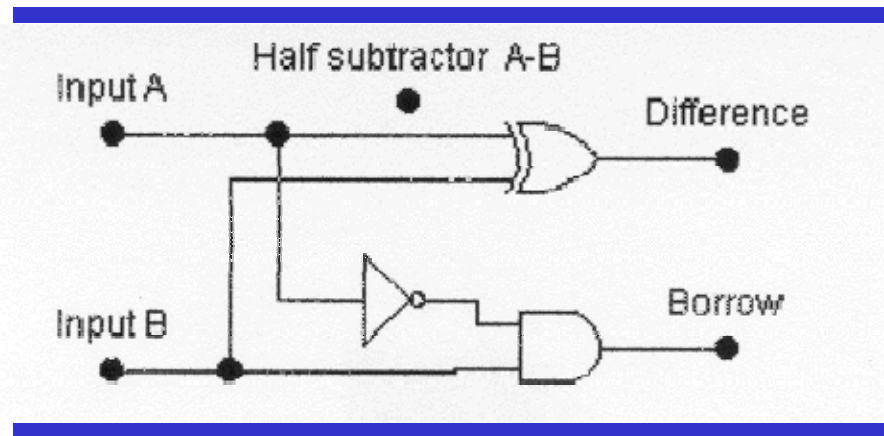
Q#4- What is the difference and borrow outputs from this half-subtractor circuit?

ANS: $D_i = 1, B_o = 1$

(A - B)

A = 0

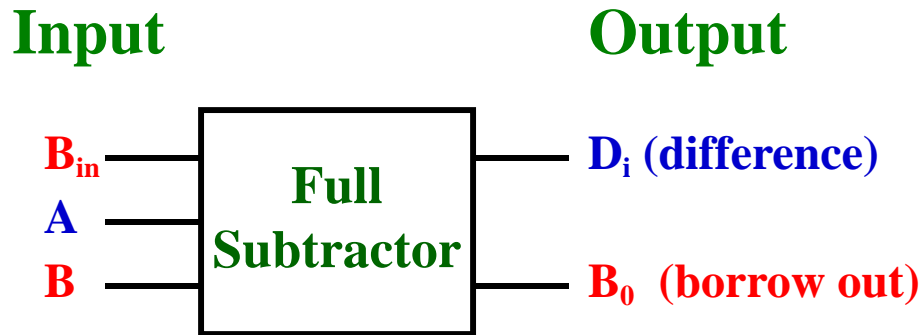
B = 1



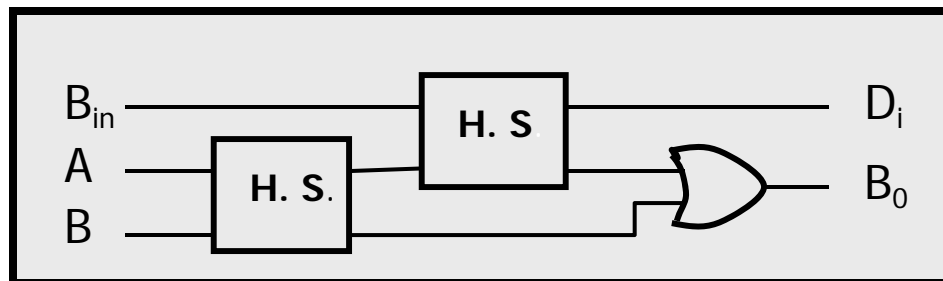
Full Subtractor

Used for subtracting binary place values other than the 1s place

Logic
Symbol:



Logic
Diagram:





QUIZ

Q#6- What are the Difference and Borrow out output from this full-subtractor circuit?

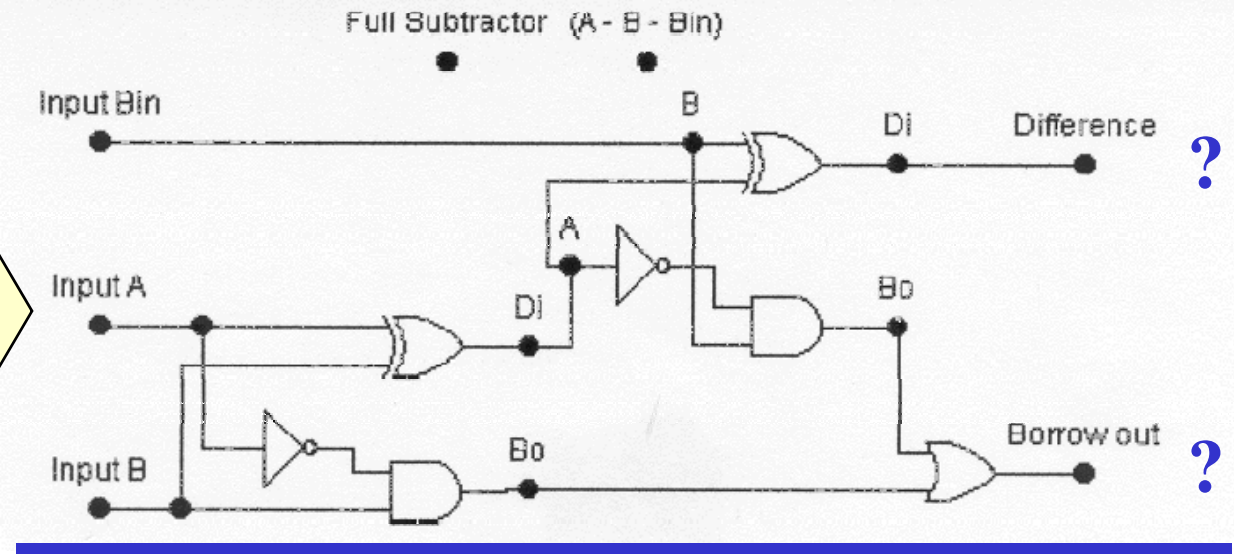
ANSWER: $D_i = 0$, $B_o = 0$

(A - B - Bin)

Bin = 0

A = 1

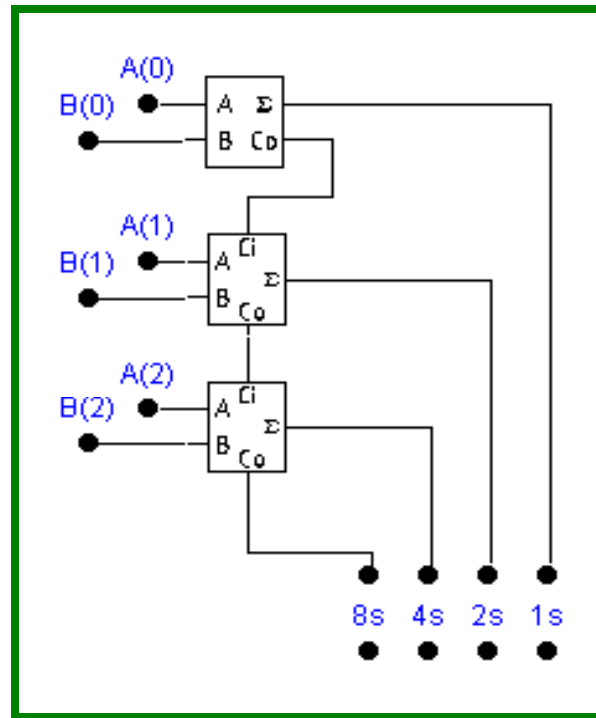
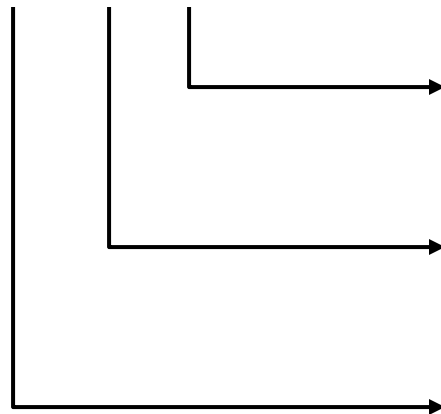
B = 1



Parallel Adding

- Use half adder for LSD
- Use full adder for other digits

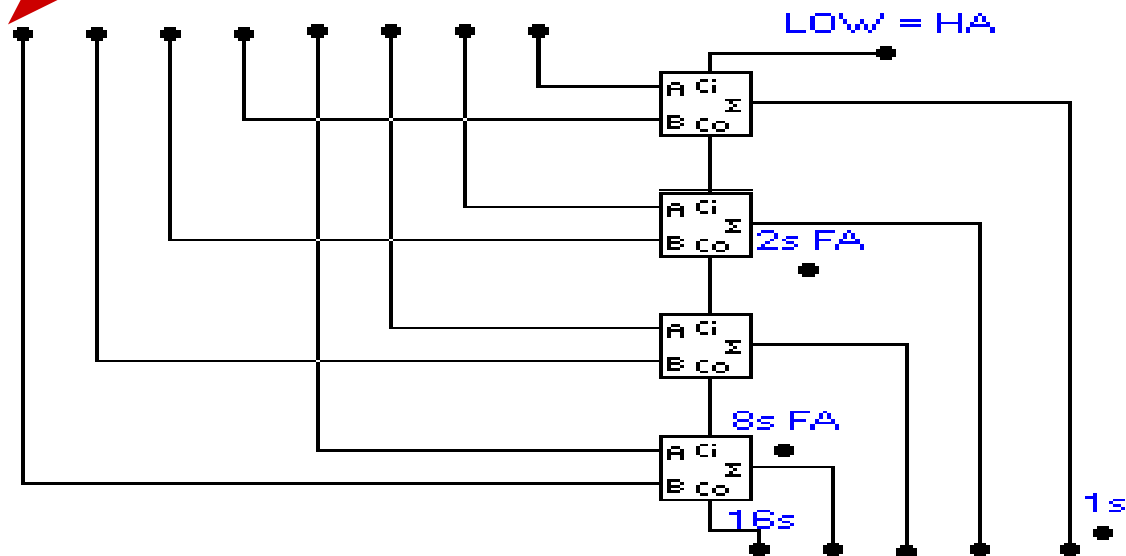
$$\begin{array}{r} A_2 \quad A_1 \quad A_0 \\ + B_2 \quad B_1 \quad B_0 \\ \hline \end{array}$$



Parallel Adder

Enter binary numbers
to be added

1 1 1 0 + 0 1 1 0



1 0 1 0 0

SUM appears here

Parallel adders are available in IC form.

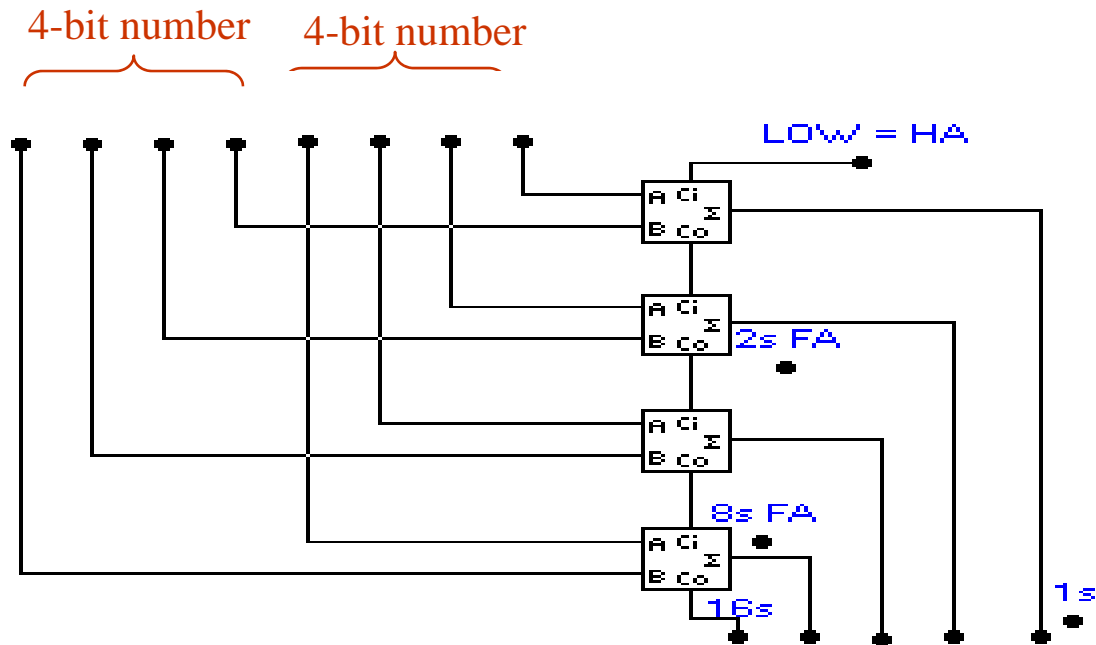
1s place uses half-adder

2s, 4s, 8s places use full adders



QUIZ

Q#7- When the 4-bit parallel adder adds binary 1010 and 1001 the sum appearing at the lower right will be ____.



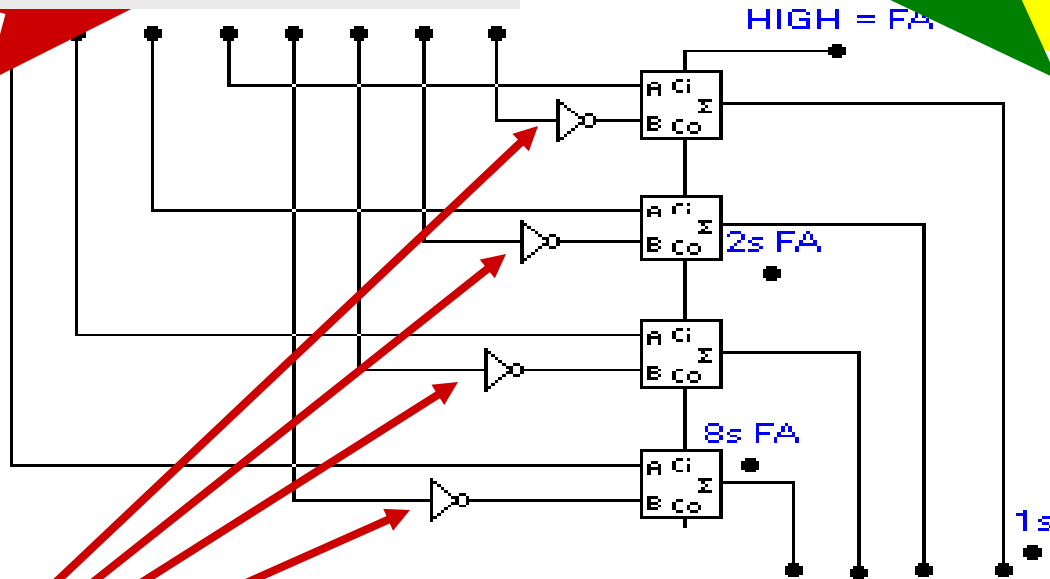
ANS: 10011

Parallel Subtractor Using Full Adders

Binary numbers to subtracted are input

1 0 0 1 - 0 1 1 1

HIGH at Carry in input acts like adding +1 to a 1s complement to form the 2s complement.
1sC is formed by four inverters.



Inverters

Note the use of

The result (difference) of the subtraction problem will appear here.

Also notice the addition of four inverters on the B inputs to the FAs

0 0 1 0

Binary Multiplication

Example:

Multiply the binary numbers 111 and 101.

								1 1 1	Multiplicand
								x 1 0 1	Multiplier
<hr/>									
								1 1 1	1st partial product
								0 0 0	2nd partial product
								1 1 1	3rd partial product
<hr/>									
								1 0 0 0 1 1	Product

111 x 101 can also be calculated: 111 + 111 + 111 + 111 + 111



QUIZ

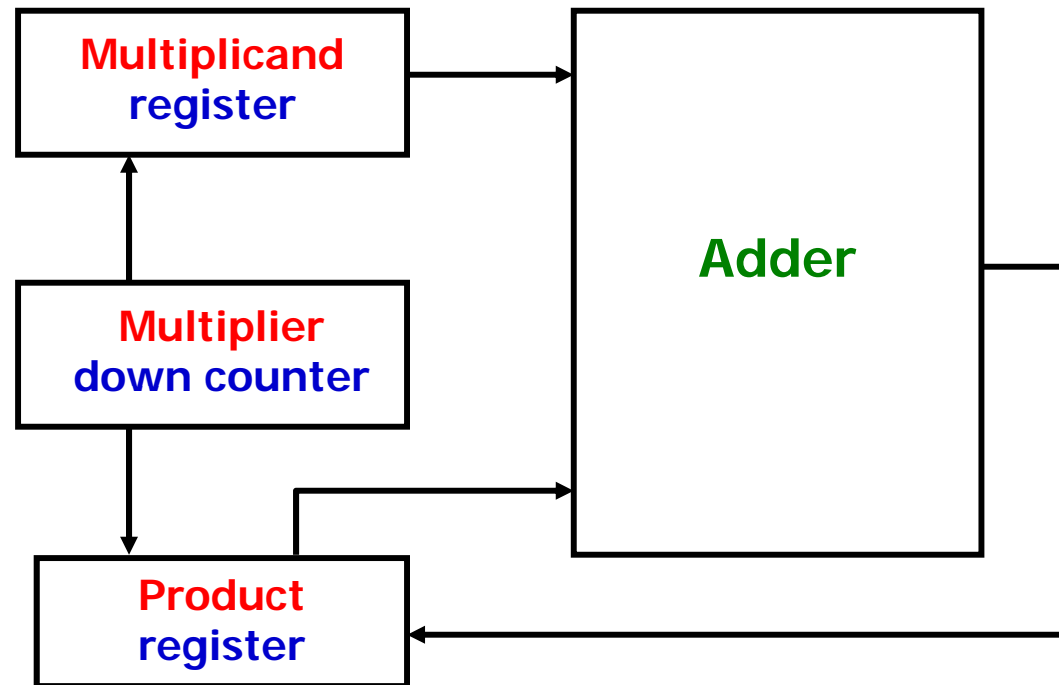
Multiply the binary numbers 101 and 100.

$$\begin{array}{r} \\ \\ \hline \\ \\ \\ \hline \end{array}$$

Binary Multipliers

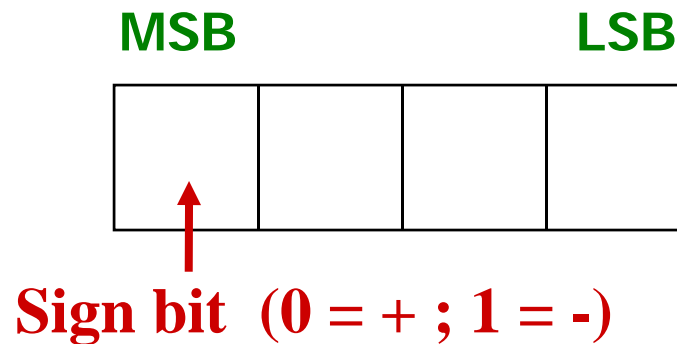
Binary multiplier circuits – utilize repeated addition.

Block Diagram:



2s Complement Notation

- **2s complement representation - widely used in microprocessors.**
- **Represents *sign* and *magnitude***

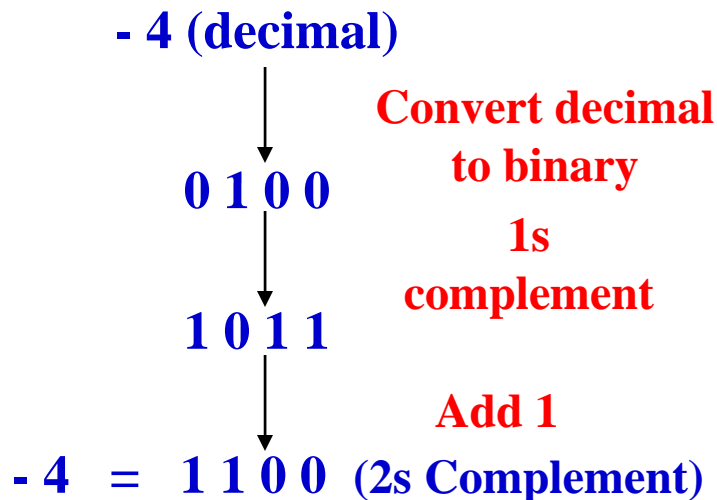


Decimal:	+7	+4	+1	0	-1	-4	-7
2s Complement:	0111	0100	0001	0000	1111	1100	1001

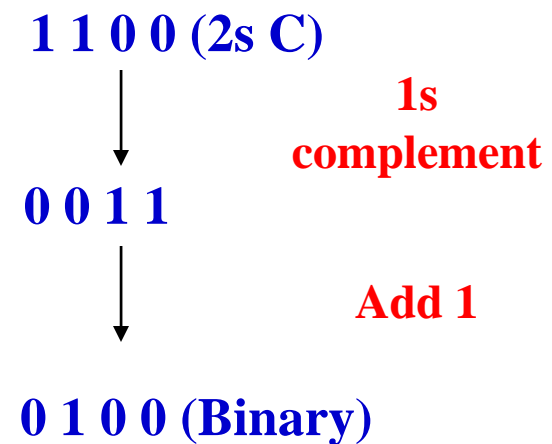
2s Complement - Conversions

- **Converting positive numbers to 2s complement:**
 - Same as converting to binary
- **Converting negative numbers to 2s complement:**

Decimal to 2s Complement



2s Complement to Binary





QUIZ

Q#5- Convert the decimal number -8 to 2s complement.

8 (decimal)

Step 1: convert decimal to binary

1000

Step 2: convert to 1s complement

0111

Step 3: add +1 equaling 2s complement

1000 2sC

ANS: -8 = 1000 2sC



QUIZ

Q#4- Convert the 2s complement number 1111 to decimal.

1111



0000



0001 binary



1 decimal

Step 1: convert to 1s complement

Step 2: Add +1

Step 3: convert binary to decimal

ANS: 1111 2sC = -1 decimal

Adding/Subtracting in 2s Complement

2s complement notation makes it possible to add and subtract signed numbers

(Decimal)	2s Complement
$\begin{array}{r} (-1) \\ + (-2) \\ \hline (-3) \end{array}$	$\begin{array}{r} 1111 \\ + 1110 \\ \hline 11101 \end{array}$ <p>Discard 1 1101 2s complement</p>
$\begin{array}{r} (+1) \\ + (-3) \\ \hline (-2) \end{array}$	$\begin{array}{r} 0001 \\ + 1101 \\ \hline 1110 \end{array}$ <p>2s complement</p>



QUIZ

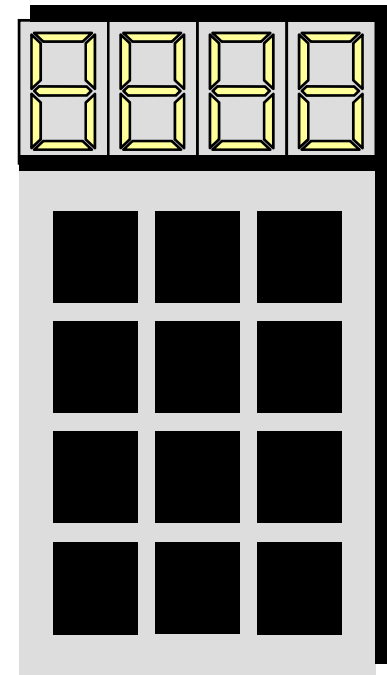
Add the following 2s complement numbers:

$$\begin{array}{r} (+5) \\ + (-4) \\ \hline (+1) \end{array} \quad \begin{array}{r} 0101 \\ + 1100 \\ \hline 10001 \end{array}$$

Discard

Practical Suggestion for Binary Math

- **Use a scientific calculator.**
- **Most scientific calculators have DEC, BIN, OCT, and HEX modes and can either convert between codes or perform arithmetic in different number systems.**
- **Most scientific calculators also have other functions that are valuable in digital electronics such as AND, OR, NOT, XOR, and XNOR logic functions.**

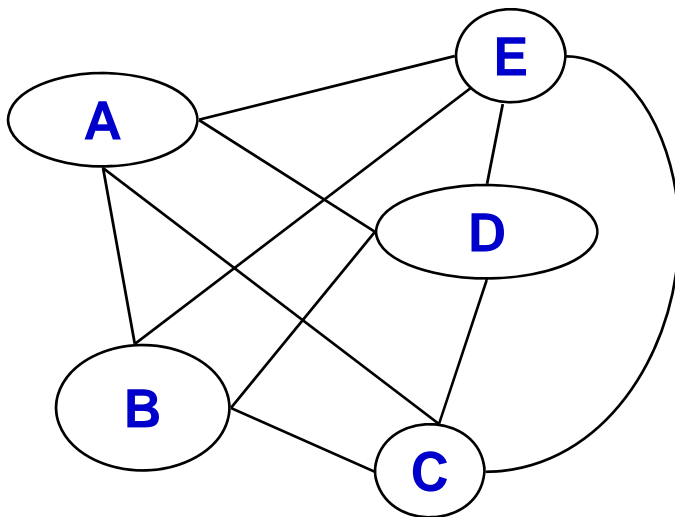


REVIEW

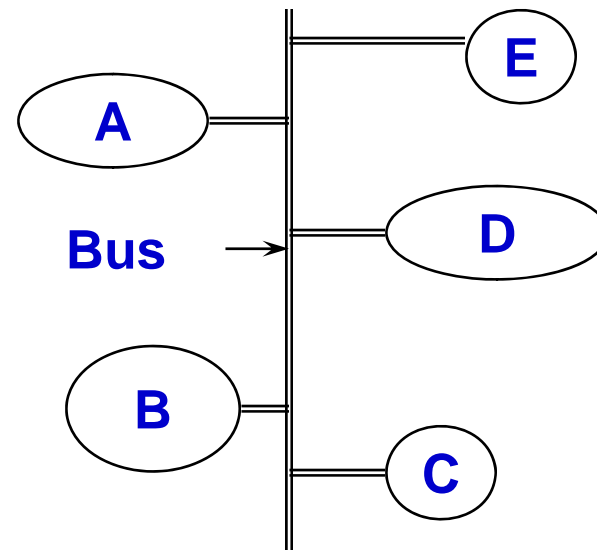
- **Binary Addition**
- **Half & Full Adders**
- **Binary Subtraction**
- **Half & Full Subtractors**
- **Parallel Adders and Subtractors**
- **Using Adders for Subtraction**
- **Binary Multiplication**
- **Binary Multipliers**
- **2s Complement Notation**
- **2s Complement Adding/Subtracting**

Buses

- Concept is to link together multiple functional units over a common data highway at a lower cost than using multiple point to point links

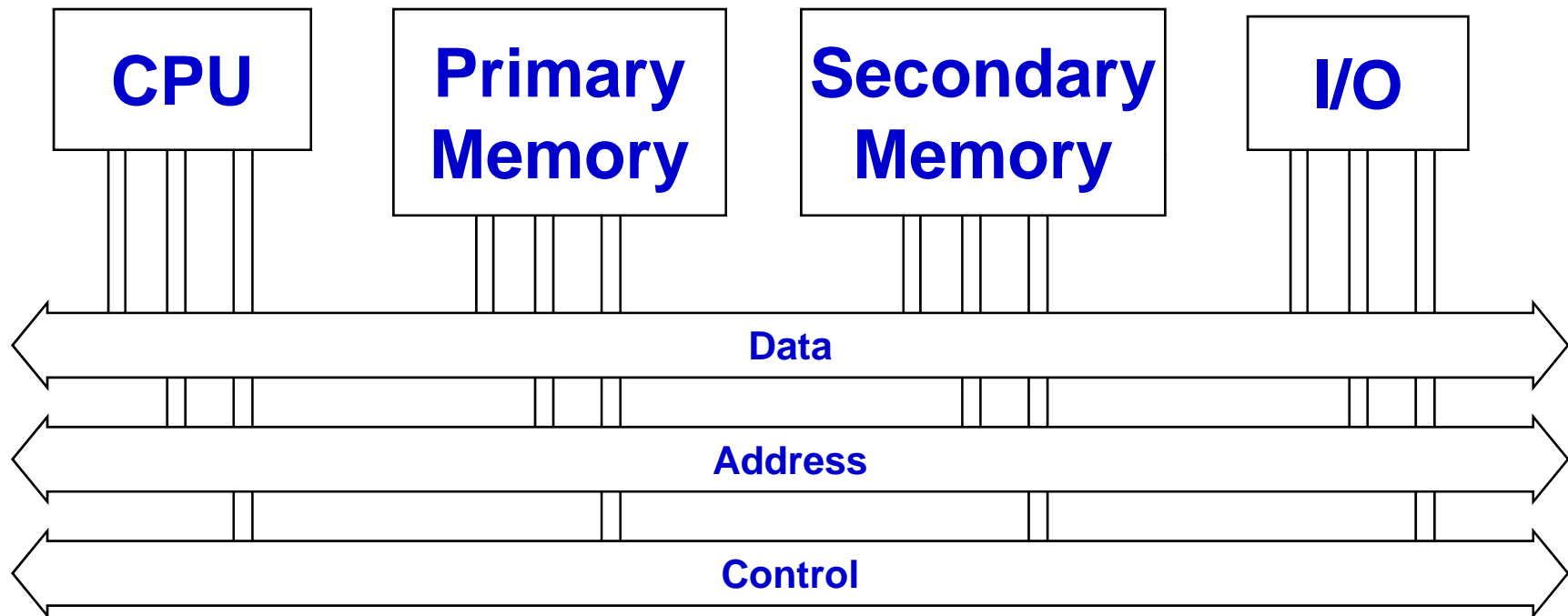


OR



Number of Links = $n * (n - 1) / 2$

Bus - Essential Part of Any Computer

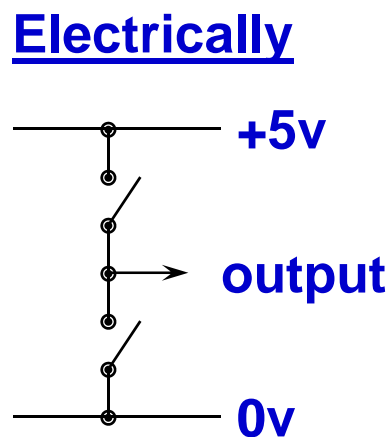
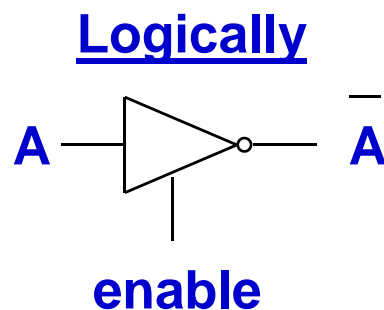


Tri-state Logic Outputs

- Since we can have multiple masters on a bus, we need Tri-state logic for attachment to a bus so that each device can choose to drive or not drive the bus depending on whether it is the bus master for a given bus cycle
- Tri-state logic prevents a bus conflict where one device is driving a signal to 1 and another device is driving it to 0 at the same time - generates high current through wires (and smoke?)

Tri-State Logic

- The problem with connecting multiple “normal” outputs together on a bus is that each has to be in one logic state (0) or the other (1) - driving voltage on each bus signal high or low
- This represents a conflict over the state of the signal
- We resolve this conflict with tri-state logic

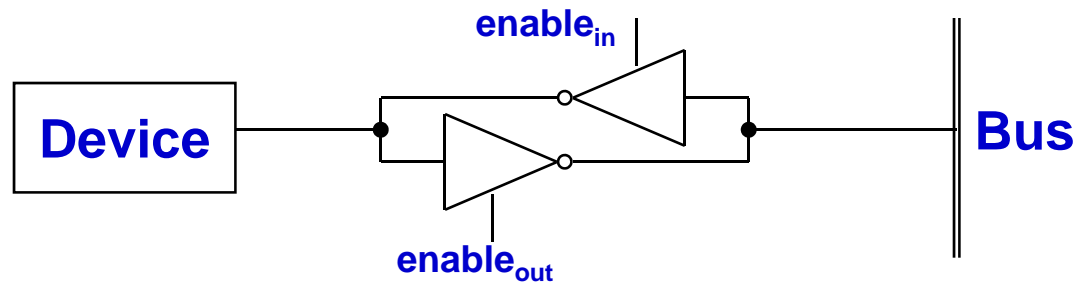


Truth Table

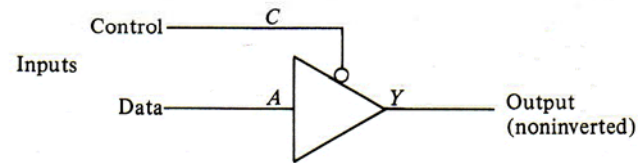
enable	A	Output
0	0	(Z)
0	1	(Z)
1	0	1
1	1	0

Tri-State Logic and Buses

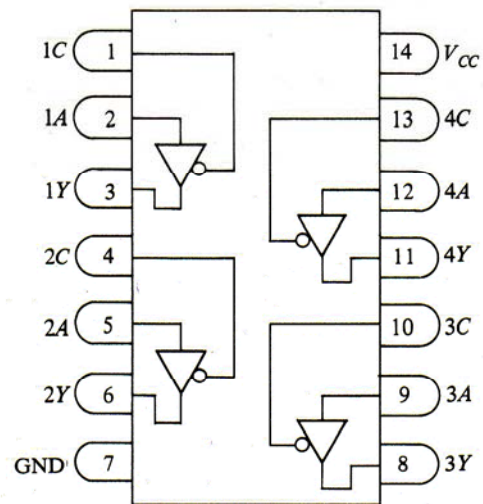
- The logical element has output enable pin to go from a floating output to drive the output from the circuit
- Inverters and buffers are used as bus drivers or buffers
 - Two such drivers or buffers in opposite directions are used to make the connection bi-directional
 - The gates also provide more “drive” onto the bus so that the bus signals are stronger and the bus can be longer



Tri-State Logic and Buses



(a) Logic symbol of a three-state buffer



(b) Pin diagram

Inputs		Output
C	A	Y
L	L	L
L	H	H
H	X	(Z)

L = LOW voltage level
H = HIGH voltage level
X = don't care
(Z) = high impedance (off)

(c) Truth table

Fig. 13-12 74125 quad three-state buffer IC

Bus Master – Slave Relationships

- During any specific bus cycle, only one device attached to the bus is allowed to drive it
- Driving the bus means that a device is forcing each signal on the bus to a high or low state
- For the data bus, the processor, a memory chip, or an I/O device may be driving the data bus during a specific read or write bus cycle
- Specific signals on the address and control bus select a device to be the master on the data bus

Bus Master – Slave Relationships

- Up till now, I have said that the address bus and the control bus are always driven by the processor, however that is NOT really true!
- That was only a “lie of simplification”!
- The processor is NOT the only device that may be driving the address and control busses
- Hopefully you are now well-prepared for me to un-simplify a bit 😊

Bus Arbitration

- Bus arbitration is used to hand off a bus between one of several potential bus masters using signals that are a part of the bus itself
- A bus arbitration protocol implements some form of bus request and bus grant handshake to determine which device will be the master on the bus for the next bus cycle

Bus Master – Slave Relationships

- Other devices that potentially can be the master on the address and control bus are:
 - Direct Memory Access (DMA) Controller
 - DRAM controller to refresh the stored bits
 - Other processors in multiprocessor architectures
- We'll only discuss the first application above, but not at the level of programming for it in detail – too hard to un-simplify that₃₉

Direct Memory Access

- You thought that handling an I/O device under interrupt control was pretty good – right?
- Wrong!
- The overhead to process an interrupt for each byte of data is still relatively costly in terms of processor time
 - Process interrupt and stack context of processor
 - Fetch and execute instructions of ISR
 - Move data from I/O device or memory to processor register
 - Move data from processor register to memory or I/O device
 - Restore context of processor and resume background code
- It is better if an I/O device like a disk can transfer data directly to or from the memory without the processor needing to execute any in or out instructions!

Direct Memory Access

- We add a DMA Controller (DMAC) to our system, e.g. Intel 8237A DMA controller chip
- The DMAC has the capability of becoming the bus master on the address and control bus for one or more “channels” transferring data between an I/O device and memory, e.g. 8237A supports 4 channels
- We connect DMA Request and DMA Acknowledge signals between the I/O device and the DMAC
- Software in the CPU sets up the DMAC to transfer an entire sequence of bytes between memory buffer and the I/O device or vice versa
- During each byte transfer, the DMAC drives address and control bus signals instead of the processor

DMAC Programming

- Generalized Steps for Programming DMAC
 - Allocate a suitably-sized memory buffer
 - Disable the DMA channel being programmed
(Note that cli does not stop DMA cycle stealing)
 - Set the Base Address Register with buffer address
 - Set the Base Count Register with size of transfer
 - Set the DMA transfer mode
 - Enable the DMA channel to start the transfer

DMAC Operation

- When requested, the DMAC arbitrates with the CPU to be the master on the address and control busses
- It executes a bus cycle to transfer a byte of data from memory (or I/O device) to I/O device (or memory)
- While DMA controller is bus master, the CPU can not access memory or I/O devices
- This is called “Cycle Stealing” (the DMA controller steals bus cycles from the processor)
- It takes less time than executing an ISR for each byte

DMAC Operation

- When the DMAC finishes transferring an entire block of data between I/O device and memory
 - It interrupts the processor (via TC to the PIC)
 - ISR software in the processor sets up DMAC again for transferring the next block of data
- Processor gets interrupted for I/O handling much less often than once per byte of data
- Hence, much better processor performance

Tri-State Bus Summary

- All devices have tri-state logic connections to the data bus – may be driving or receiving
- Memory and I/O devices don't need tri-state logic on address/control bus (never drive them)
- Because the processor may need to yield the control/address busses, it must have tri-state logic for driving those bus signals
- DMAC controller must have tri-state logic for driving the control and address bus signals

Propagation & Transmission delay

- Propagation speed - speed at which a bit travels through the medium from source to destination.
- Transmission speed - the speed at which all the bits in a message arrive at the destination. (difference in arrival time of first and last bit)

Propagation and Transmission Delay

- Propagation Delay = Distance/Propagation speed
- Transmission Delay = Message size/bandwidth bps
- Latency = Propagation delay + Transmission delay + Queueing time + Processing time

Example 3.45

What is the propagation time if the distance between the two points is 12,000 km? Assume the propagation speed to be 2.4×10^8 m/s in cable.

$$\text{Propagation time} = \frac{12,000 \times 1000}{2.4 \times 10^8} = 50 \text{ ms}$$

Solution

We can calculate the propagation time as

The example shows that a bit can go over the Atlantic Ocean in only 50 ms if there is a direct cable between the source and the destination.



Example 3.46

What are the propagation time and the transmission time for a 2.5-kbyte message (an e-mail) if the bandwidth of the network is 1 Gbps? Assume that the distance between the sender and the receiver is 12,000 km and that light travels at 2.4×10^8 m/s.

Solution

We can calculate the propagation and transmission time as shown on the next slide:

Example 3.46 (continued)

$$\text{Propagation time} = \frac{12,000 \times 1000}{2.4 \times 10^8} = 50 \text{ ms}$$

$$\text{Transmission time} = \frac{2500 \times 8}{10^9} = 0.020 \text{ ms}$$

Note that in this case, because the message is short and the bandwidth is high, the dominant factor is the propagation time, not the transmission time. The transmission time can be ignored.



Example 3.47

What are the propagation time and the transmission time for a 5-Mbyte message (an image) if the bandwidth of the network is 1 Mbps? Assume that the distance between the sender and the receiver is 12,000 km and that light travels at 2.4×10^8 m/s.

Solution

We can calculate the propagation and transmission times as shown on the next slide.

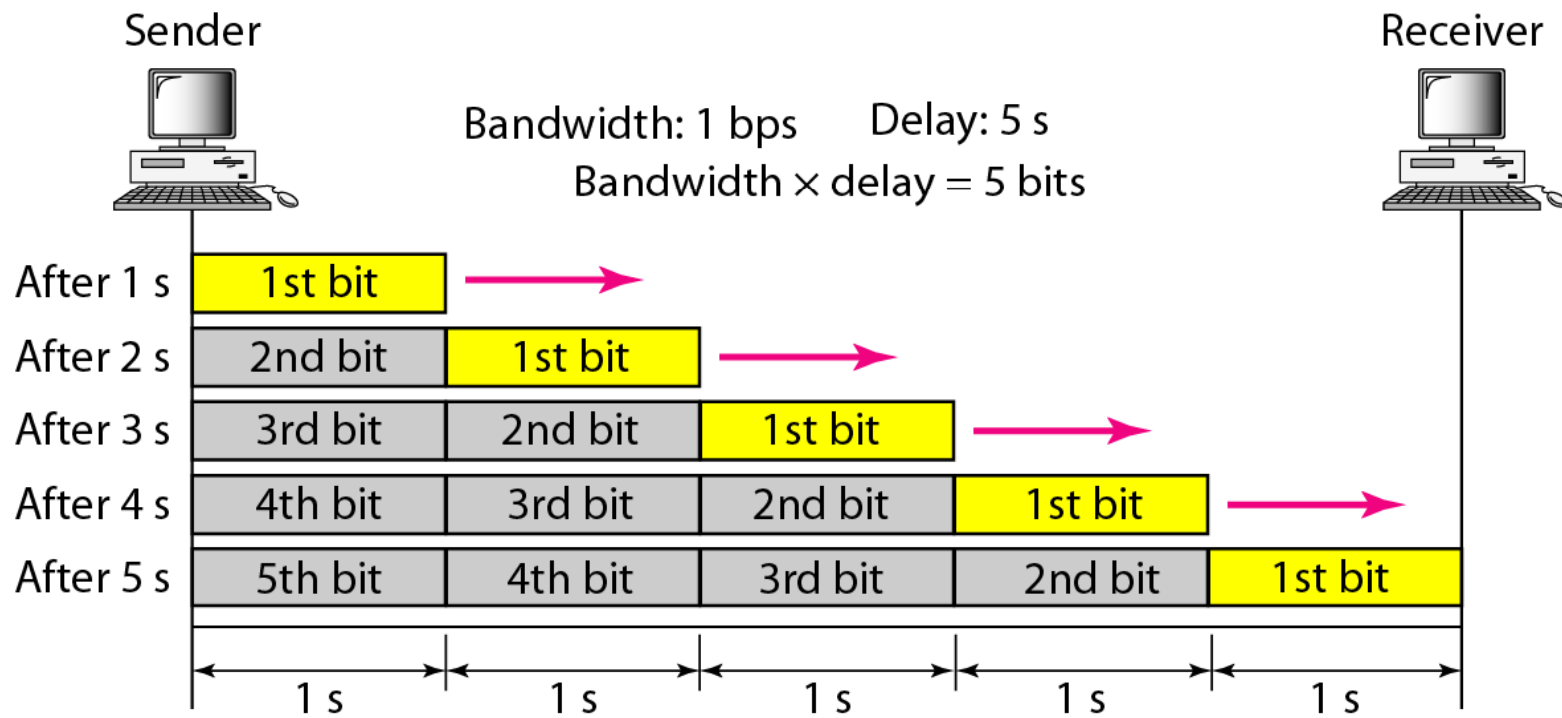
Example 3.47 (continued)

$$\text{Propagation time} = \frac{12,000 \times 1000}{2.4 \times 10^8} = 50 \text{ ms}$$

$$\text{Transmission time} = \frac{5,000,000 \times 8}{10^6} = 40 \text{ s}$$

Note that in this case, because the message is very long and the bandwidth is not very high, the dominant factor is the transmission time, not the propagation time. The propagation time can be ignored.

Figure 3.31 Filling the link with bits for case 1

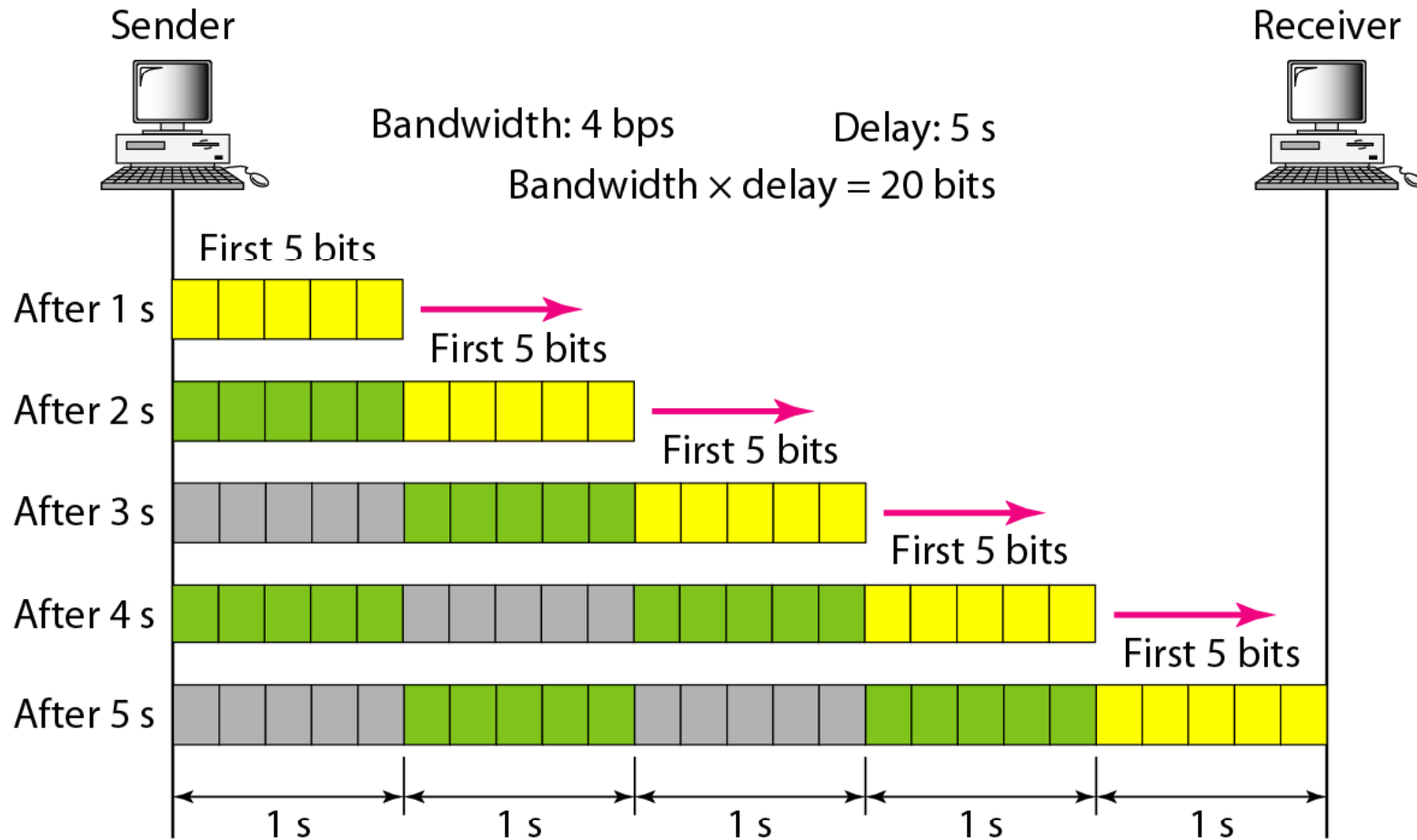


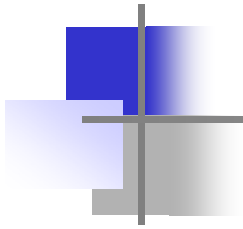


Example 3.48

We can think about the link between two points as a pipe. The cross section of the pipe represents the bandwidth, and the length of the pipe represents the delay. We can say the volume of the pipe defines the bandwidth-delay product, as shown in Figure 3.33.

Figure 3.32 Filling the link with bits in case 2





Note

The bandwidth-delay product defines the number of bits that can fill the link.

Figure 3.33 Concept of bandwidth-delay product

