UNIT - 1 **[Contact Hours - 10]**

Contents

- Introduction: The main components of a Computer
- Historical Development: First through Fourth Generation Computers
- Moore's Law
- The Von Neumann and Non Von Neumann Model
- The Evolution of the Intel x86 Architecture
- Data Representation in Computer Systems: Signed Integer Representation
- Complement Systems: One's complement and Two's complement
- Addition and Subtraction using signed numbers
- Multiplication of Positive Numbers
- Signed Operand Multiplication
- Integer Division
- Floating Point Representation [Refer class notes]
- The IEEE-754 Floating Point Standard [Refer Class notes]

TOPIC 1: Introduction: The main components of a Computer

- Computer is a programmable machine.
- Computer is a machine that manipulates data according to a list of instructions.
- Computer is any device which aids humans in performing various kinds of computations or calculations.

Three principle characteristics of computer

- It responds to a specific set of instructions in a well-defined manner.
- It can execute a pre-recorded list of instructions.
- It can quickly store and retrieve large amounts of data.

Structure of the Computer

Input Unit

- Computers accept coded information through input units.
- The most common input device is the keyboard. Whenever a key is pressed, the corresponding letter or digit is automatically translated into its corresponding binary code and transmitted to the processor.
- Many other kinds of input devices for human-computer interaction are available, including the touchpad, mouse, joystick, and trackball. These are often used as graphic input devices in conjunction with displays.
- Microphones can be used to capture audio input which is then sampled and converted into digital codes for storage and processing.
- Similarly, cameras can be used to capture video input.
- Digital communication facilities, such as the Internet, can also provide input to a computer from other computers and database servers.

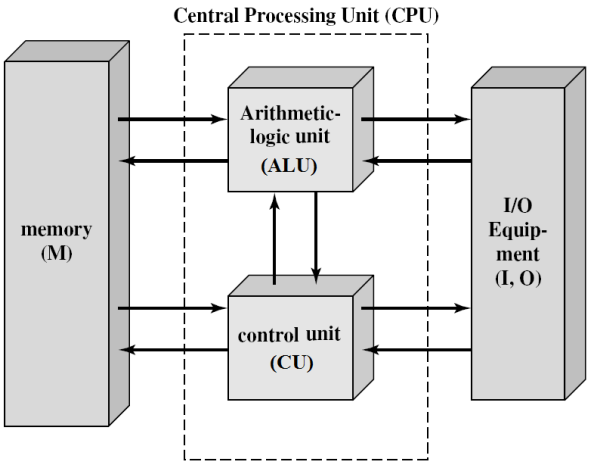


Figure: Structure of the Computer

Output Unit

- The output unit is the counterpart of the input unit.
- Its function is to send processed results to the outside world. A familiar example of such a device is a printer.
- However, printers are mechanical devices, and as such are quite slow compared to the electronic speed of a processor.
- Some units, such as graphic displays, provide both an output function, showing text and graphics, and an input function, through touch-screen capability.
- The dual role of such units is the reason for using the single name input/output (I/O) unit in many cases.

Memory Unit

The function of the memory unit is to store programs and data. There are two classes of storage, called primary and secondary.

1. Primary Memory

- Primary memory, also called main memory, is a fast memory that operates at electronic speeds.
- Programs must be stored in this memory while they are being executed.
- The memory consists of a large number of semiconductor storage cells, each capable of storing one bit of information.
- Instructions and data can be written into or read from the memory under the control of the processor.

2. Secondary Storage

- Although primary memory is essential, it tends to be expensive and does not retain information when power is turned off.
- An additional, less expensive, permanent secondary storage is used when large amounts of data and many programs have to be stored, particularly for information that is accessed infrequently.

- Access times for secondary storage are longer than for primary memory.
- A wide selection of secondary storage devices is available, including magnetic disks, optical disks (DVD and CD), and flash memory devices.

Central Processing Unit (CPU)

1. Arithmetic and Logic Unit (ALU)

- Computer operations are executed in the arithmetic and logic unit (ALU) of the processor.
- Any arithmetic or logic operation, such as addition, subtraction, multiplication, division, or comparison of numbers, is initiated by bringing the required operands into the processor, where the operation is performed by the ALU.
- When operands are brought into the processor, they are stored in high-speed storage elements called registers.
- Each register can store one word of data. Access times to registers are even shorter than access times to the cache unit on the processor chip.

2. Control Unit (CU)

- The memory, arithmetic and logic unit, and I/O units store and process information and perform input and output operations.
- The operation of these units must be coordinated in some way. This is the responsibility of the control unit.
- The control unit sends control signals to other units and senses their states.
- Control circuits are responsible for generating the timing signals that govern the transfers and determine when a given action is to take place.
- Data transfers between the processor and the memory are also managed by the control unit through timing signals.
- A large set of control lines (wires) carries the signals used for timing and synchronization of events in all units.

TOPIC 2: Historical Development: First through Fourth Generation Computers

- Electronic digital computers as we know them today have been developed since the 1940s.
- A long, slow evolution of mechanical calculating devices preceded the development of electronic computers.
- Around 400 years ago, complex mechanical devices, constructed from gear wheels, levers, and pulleys, were used to perform the basic operations of addition, subtraction, multiplication, and division.
- During World War II, the first electronic computer was designed and built at the University of Pennsylvania, using the vacuum tube technology developed for radios and military radar equipment.
- Development of the technologies used to fabricate processors, memories, and I/O units of computers has been divided into four generations.

The first generation [1945 to 1955]

- Programs and their data were located in the same memory, as they are today

- Assembly language was used to prepare programs and was translated into machine language for execution
- Basic arithmetic operations were performed in a few milliseconds, using vacuum tube technology
- Mercury delay-line memory was used
- I/O functions were performed by devices similar to typewriters
- Magnetic core memories and magnetic tape storage devices were also developed

The second generation [1955 to 1965]

- The transistor was invented at AT&T Bell Laboratories in the late 1940s and quickly re-placed the vacuum tube in implementing logic functions
- Magnetic core memories and magnetic drum storage devices were widely used in this generation
- Magnetic disk storage devices were developed in this generation
- The earliest high-level languages, such as Fortran, were developed
- Compilers were developed to translate these high-level language programs into assembly language
- IBM became a major computer manufacturer during this time.

The third generation [1965 to 1975]

- Introduction of fabrication technology for development of many transistors on a single silicon chip, called integrated-circuit technology
- This enabled faster and less costly processors and memory elements
- Integrated-circuit memories began to replace magnetic core memories.
- introduction of microprogramming, parallelism, and pipelining
- Operating system software allowed efficient sharing of a computer system by several user programs.
- Cache and virtual memories were developed.
- Cache memory makes the main memory appear faster than it really is, and virtual memory makes it appear larger.
- mainframe computers from IBM and the line of PDP minicomputers from Digital Equipment Corporation were dominant

The fourth generation [1975 to Present]

- Complete processors and large sections of the main memory of small computers could be implemented on single chips
- Tens of thousands of transistors could be placed on a single chip, and the name Very Large Scale Integration (VLSI) was coined to describe this technology.
- A complete processor fabricated on a single chip became known as a microprocessor.
- Companies such as Intel, National Semiconductor, Motorola, Texas Instruments, and Advanced Micro Devices have been the driving forces of this technology.
- Current VLSI technology enables the integration of multiple processors (cores) and cache memories on a single chip.

- Organizational concepts such as parallelism and hierarchical memories have evolved to produce the high-performance computing systems of today as the fourth generation has matured.

The fifth generation [Future]

- Use of machine learning and artificial intelligence
- Use of higher end memories for graphics
- The use of parallel processing and superconductors is helping to make artificial intelligence a reality.
- The goal is to develop devices that respond to natural language input and are capable of learning and self-organization.
- There are some applications, such as voice recognition, that are being used today.

TOPIC 3: Moore's Law

- How small can we make transistors? How densely can we pack chips?
- Every year, scientists continue to thwart prognosticators' attempts to define the limits of integration.
- In 1965, Intel founder Gordon Moore stated, "The density of transistors in an integrated circuit will double every year."
- The current version of this prediction is usually conveyed as "the density of silicon chips doubles every 18 months."
- This assertion has become known as Moore's Law. Moore intended this postulate to hold for only 10 years. However, advances in chip manufacturing processes have allowed this law to hold for almost 40 years.
- Using current technology, Moore's Law cannot hold forever. There are physical and financial limitations that must ultimately come into play.
- At the current rate of miniaturization, it would take about 500 years to put the entire solar system on a chip! Clearly, the limit lies somewhere between here and there. Cost may be the ultimate constraint.

TOPIC 4: The Von Neumann and Non Von Neumann Model

<u>The Von Neumann Model</u>

- It is also known as the von Neumann model or Princeton architecture
- Introduced by John Von Neumann in 1945
- A processing unit with an arithmetic logic unit, control unit, and processor registers
- The control unit includes an instruction register and a program counter
- It has same memory that stores data and instructions
- The mass storage is used separately/externally
- It refer to a stored-program computer in which an instruction fetch and a data operation cannot occur at the same time (since they share a common bus).

- This problem is referred to as the von Neumann bottleneck problem, which overall slower the computer processing

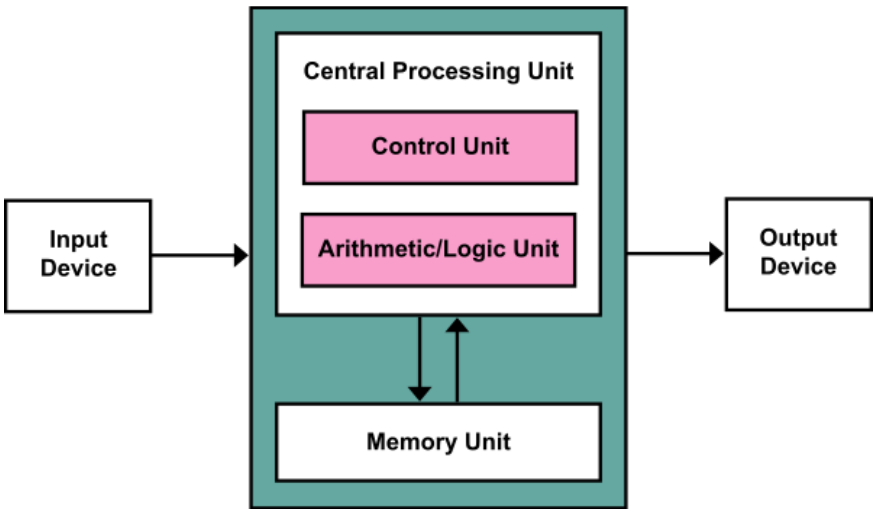


Figure: Von Neumann Computer Architecture

The Non Von Neumann Model

- Any computer architecture having separate pathways for transferring data and instructions come under the category of Non Von Neumann model
- It has separate memories for storing data and instructions
- This architecture include multiple bus system for transferring instructions and data separately
- As the data and instructions can be operated separately and simultaneously hence, multiple steps can be solved by the computer in fewer clock cycles

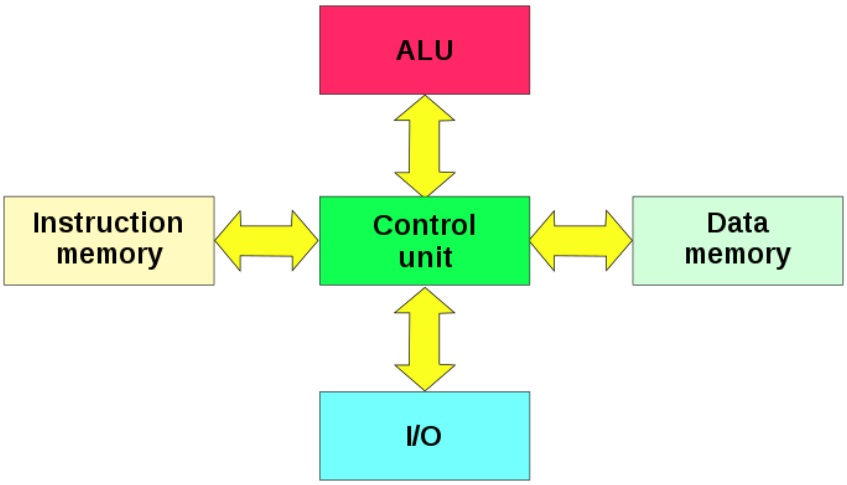


Figure: Non Von Neumann Computer Architecture (Ex: Harvard Architecture)

Comparison of The Von Neumann and Non Von Neumann Model

Parameter	Von Neumann Model	Non Von Neumann Model
Memory arrangement	Instruction and Data both are stored in **same memory**	It contains **separate memory** for storing Instruction and Data
Bus arrangement	It has **single bus** to transfer instruction and Data hence transferred sequentially	It has **multiple buses** connected to instruction memory and Data memory separately, hence instructions and data can be transferred simultaneously/parallelly
Speed of execution	As instruction and data are sent sequentially over a single bus speed of execution is **slower**	As instruction and data are sent parallelly over a multiple buses speed of execution is **Faster**
Clock cycle (instruction execution contains multiple steps)	**More clock cycles** for single instruction	**Less clock cycles** (or mostly one clock cycle) for single instruction
Physical size	**Smaller size** due to common memory and common bus	**Larger size** due to separate memories and separate busses
Complexity of operation	**Less complex controlling** as instructions and data are sent sequentially over single bus	**More complex controlling** as instructions and data are sent parallelly over multiple buses
Memory Wastage	**Less** memory is wasted as both instruction and data are stored in same memory	**High** memory wastage as data cannot be stored in instruction memory and vice versa
Cost	**Low** cost due to less hardware	**More** cost due to more hardware
Applications	Personal Computers and small level computers	Mostly in microcontrollers and signal processing

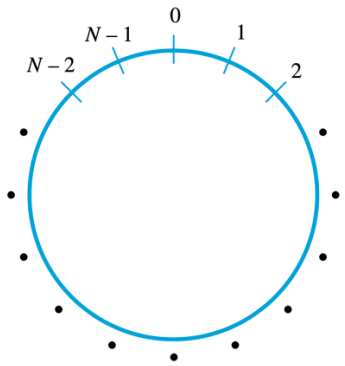
TOPIC 5:The Evolution of the Intel x86 Architecture

The x86 architecture continues to dominate the processor market outside of embedded systems. Although the organization and technology of the x86 machines has changed dramatically over the decades, the instruction set architecture has evolved to remain backward compatible with earlier versions. The first 8086 was introduced with a clock speed of 5 MHz and had 29,000 transistors. In 2008 A quad-core Intel Core 2 introduced which operates at 3 GHz, a speedup of a factor of 600, and has 820 million transistors, about 28,000 times as many as the 8086. It is worthwhile to list some of the highlights of the evolution of the Intel x86 Architecture:

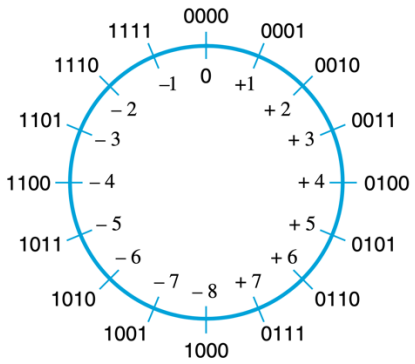
- 8080: The world's first general-purpose microprocessor. This was an 8-bit machine, with an 8-bit data path to memory. The 8080 was used in the first personal computer.
- 8086: A far more powerful, 16-bit machine. In addition to a wider data path and larger registers, the 8086 has an instruction cache, or queue, that prefetches a few instructions before they are executed. The 8086 is the first appearance of the x86 architecture.
- 80286: This extension of the 8086 enabled addressing a 16-MByte memory instead of just 1 MByte.
- 80386: Intel's first 32-bit machine, and a major overhaul of the product. With a 32-bit architecture, the 80386 rivalled the complexity and power of minicomputers and mainframes introduced just a few years earlier. This was the first Intel processor to support multitasking, meaning it could run multiple programs at the same time.
- 80486: The 80486 introduced the use of much more sophisticated and powerful cache technology and sophisticated instruction pipelining. The 80486 also offered a built-in math coprocessor, offloading complex math operations from the main CPU.

- Pentium: With the Pentium, Intel introduced the use of superscalar techniques, which allow multiple instructions to execute in parallel.
- Pentium Pro: The Pentium Pro continued the move into superscalar organization begun with the Pentium, with aggressive use of register renaming, branch prediction, data flow analysis, and speculative execution.
- Pentium II: The Pentium II incorporated Intel MMX technology, which is designed specifically to process video, audio, and graphics data efficiently.
- Pentium III: The Pentium III incorporates additional floating-point instructions to support 3D graphics software.
- Pentium 4: The Pentium 4 includes additional floating-point and other enhancements for multimedia
- Core: This is the first Intel x86 microprocessor with a dual core, referring to the implementation of two processors on a single chip.
- Core 2: The Core 2 extends the architecture to 64 bits. The Core 2 Quad provides four processors on a single chip.

TOPIC 6:Data Representation in Computer Systems: Signed Integer Representation



Circle representation of unsigned integers mod N



Mod 16 system for 2's-complement numbers

Unsigned numbers

- **This representation contains all positive numbers**
- **Total number for presentations are 2^n (Here n is number of bits for representation)**
- **Range is specified as (0→ 2^n - 1)**
- **For n=4 the number are presented with their decimal values are 0000(0), 0001(1), 0010(2), 0011(3), 0100(4),.....,1110(14), 1111(15)**

Signed numbers

- Total number for presentations are 2^n (Here n is number of bits for representation)
- This representation contains half positive numbers and half negative numbers
- Range is specified in two categories

1 Positive Range [0→ (2^n/2 - 1)]
2 Negative Range [- 2^n/2 → - 1]

- For n=4 the number are presented with their decimal values

  Signed Positive Values
  0000(0), 0001(+1), 0010(+2), 0011(+3), 0100(+4), 0101(+5), 0110(+6), 0111(+7)

  Signed Negative Values
  1000(-8), 1001(-7), 1010(-6), 1011(-5), 1100(-4), 1101(-3), 1110(-2), 1111(-1)

  Total Range

  (-8) ← (**-1**) ↔ (**0**) → (**+7**)

- When the result exceeds the positive range it is known as arithmetic overflow
- When the result comes below the negative range it is known as arithmetic underflow

How to find the value of any number presented in 2's complement system

Check the MSB;
(a) if the MSB is 0 then number is positive; find magnitude as usual procedure

(b) if the MSB is 1 the number is negative; find the magnitude by taking the 2's complement

Ex: Find the value of given signed number (1101)

Ans:

Step 1: Check MSB; Here MSB is 1 hence the number is negative

To find magnitude take 2's complement (1101) $\overrightarrow{1's\ \text{Complement}}$ (0010) $\overrightarrow{2's\ \text{Complement}}$ (0011)

So magnitude is 3; hence the number (1101) is (-3)

Ex: Find if arithmetic overflow or underflow is present for given signed number operations

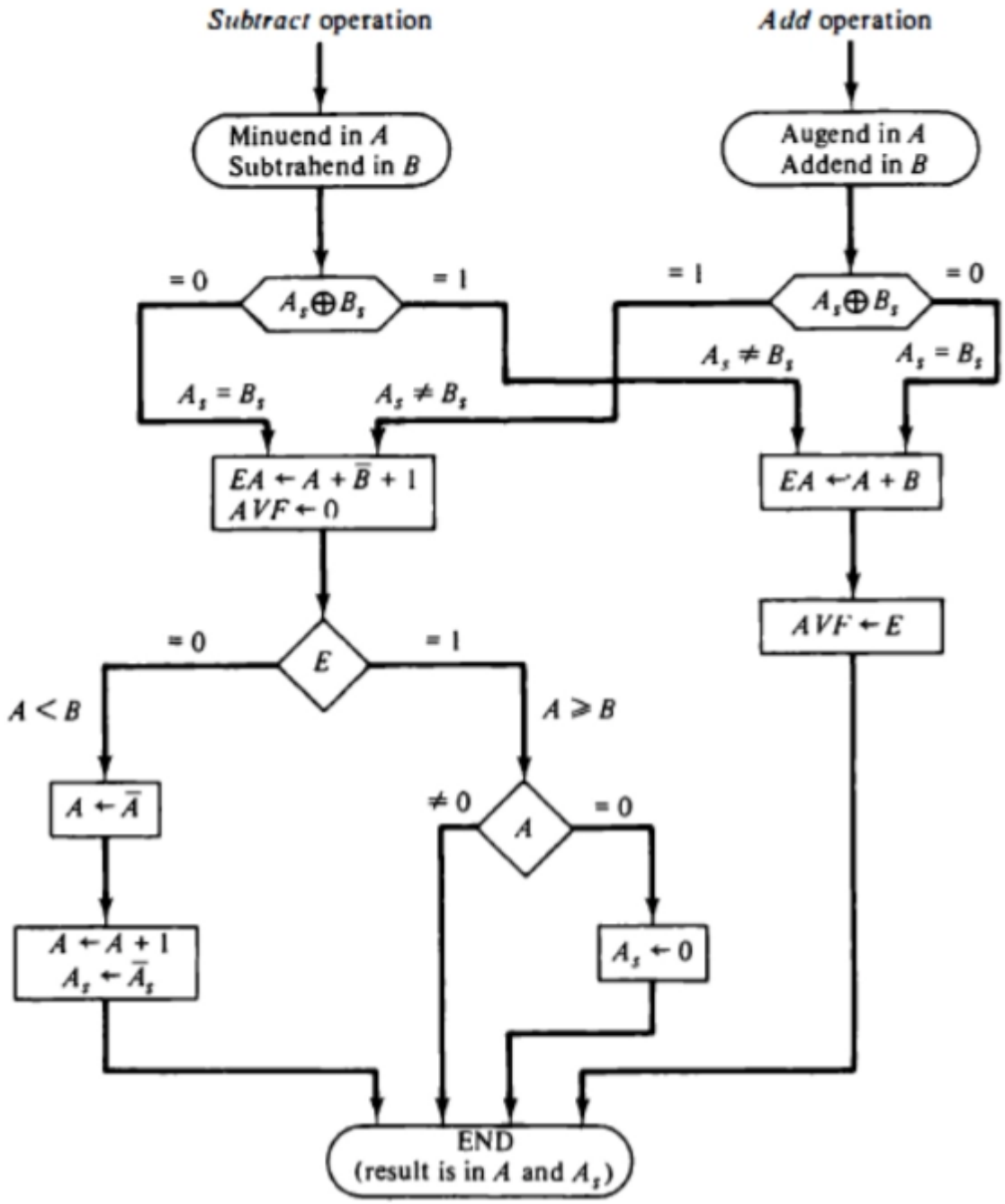
1. (0011) + (0010) : Result (0101) Adding positive number result is coming positive hence there is no arithmetic overflow

2. (0011) + (0110) : Result (1001) Adding positive number result is coming Negative hence there is arithmetic overflow

3. (1111) + (1101) : Result (1100) Adding negative number result is negative and in the range, hence no arithmetic underflow

4. (1011) + (1011) : Result (10110) in four bits result is 0110 which is a positive number Adding two negative numbers the result is coming positive hence there exist a arithmetic under flow

TOPIC 7: Addition and Subtraction using signed numbers



Question: Perform (+5) +(-4)

Step 1: Identifying values in binary

As = 0 A = 101

Bs = 1 B = 100

Step 2: Operation to be performed is addition, so we will start the algorithm from right side

Step 3: As(XOR)Bs will results as '1'

Step 4:

EA = A + B' + 1 will be performed

EA = (101) + (100)' + 1

EA = (101) + (011) + 1 = 1001

Step 5: As EA = 1011 So, E will be 1; and A = 001

Step 6: As E = 1; Hence the computer will acknowledge that A>B in magnitude

Step 7: As A is not a zero value hence As will become 0

Step 8: Finally Result will be stored as As = 0, and A = 001

Try this questions:

1. (+25) + (-22)

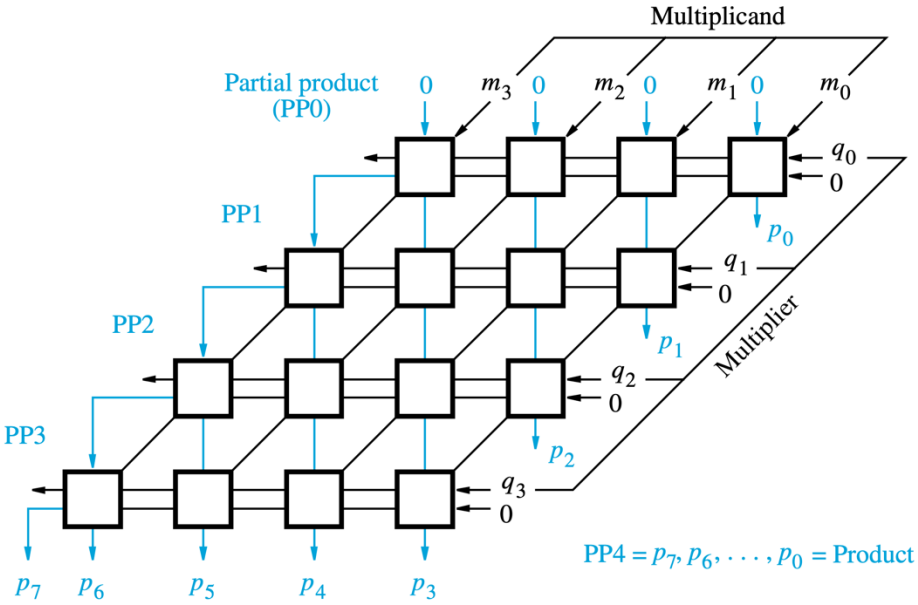
2. (-28) + (+18)

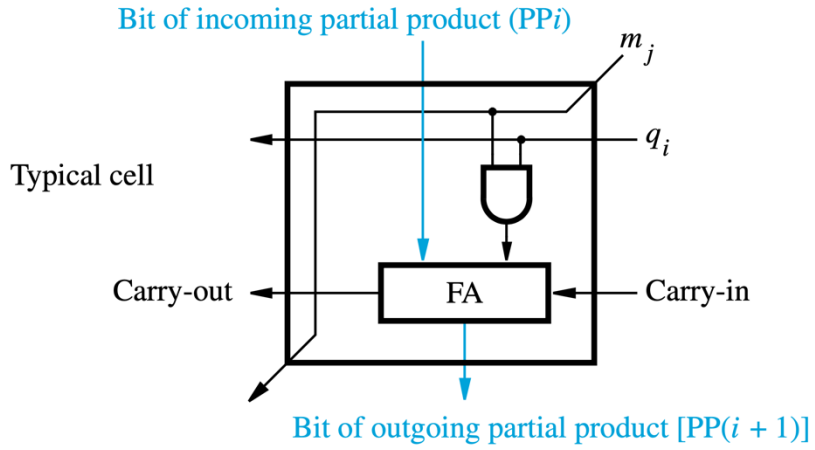
TOPIC 8: Multiplication of Positive Numbers

Conventional Method

				1	1	0	1	(13) Multiplicand M
			×	1	0	1	1	(11) Multiplier Q
				1	1	0	1	
			1	1	0	1		
		0	0	0	0			
	1	1	0	1				
1	0	0	0	1	1	1	1	(143) Product P

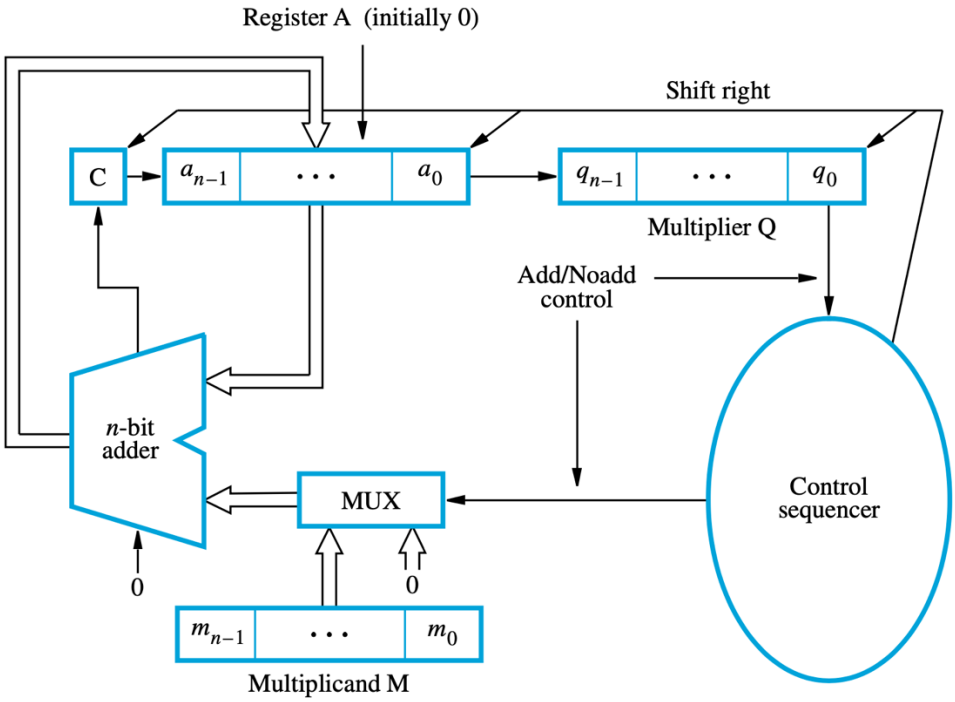
(a) Manual multiplication algorithm





(b) Array implementation

Binary Multiplication Sequencer (Used for Unsigned Numbers)



(a) Register configuration

Method

Step 1: Take two register with n bits named as A-Register and Q-Register

Step 2: Take a register C for storage of single bit and arrange in manner shown (C A Q)

Step 3: Initially all bits of A is 0; and C bit is 0 .

Step 4: From MxQ; Q is loading in Q-Register

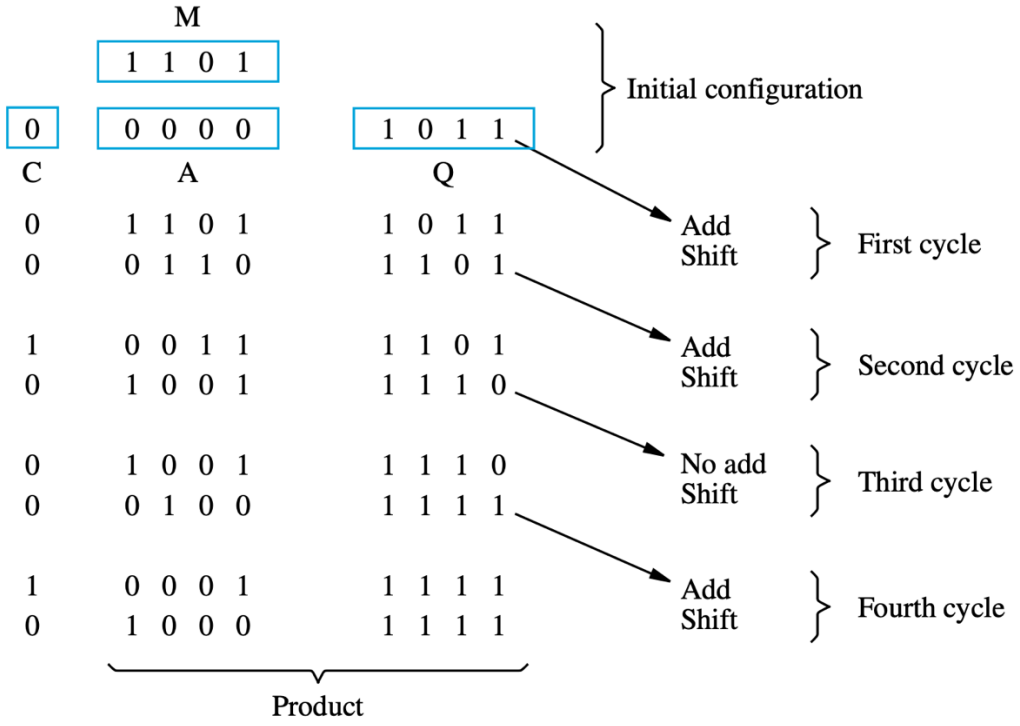
Step 5: Check the Q_0 bit which LSB of Q-Register

Step 6: If Q_0 is 0 [No Add; Perform A = A+0] and then Perform Arithmetic right shift on all registers

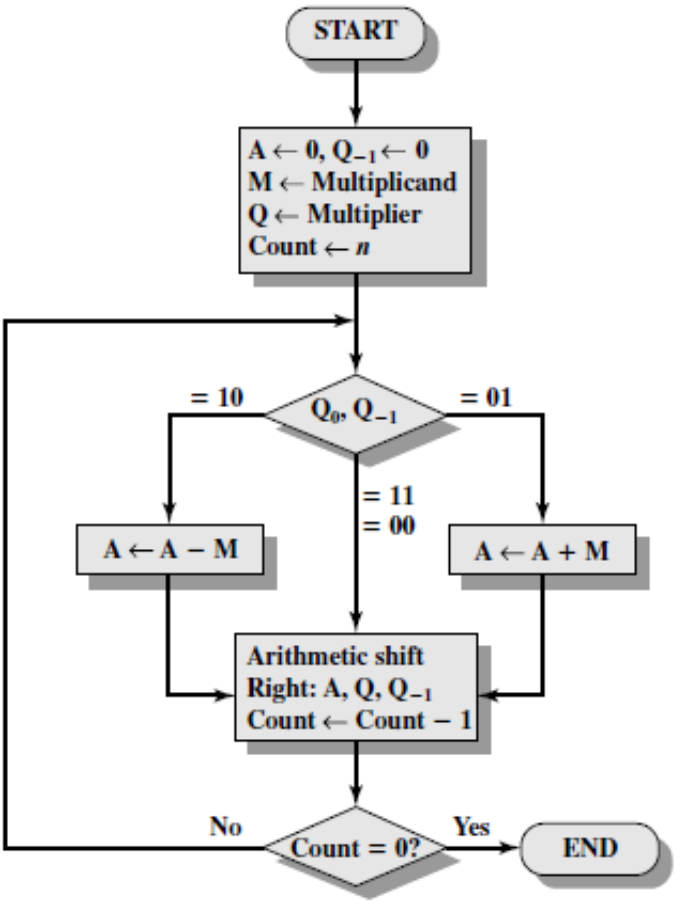
Step 6: If Q_0 is 1 [Add; Perform A = A+M] and then Perform Arithmetic right shift on all registers

Step 7: Repeat the Step 6 n number of times

Step 8: The final result is stored in AQ registers



TOPIC 9: Signed Operand Multiplication (Booth's Algorithm)



Booth's Algorithm Steps

Step 1: Take two register with n bits named as A-Register and Q-Register

Step 2: Take a register Q_{-1} for storage of single bit and arrange in manner shown (A Q Q_{-1})

Step 3: Initially all bits of A is 0; and Q_{-1} bit is also 0.

Step 4: From MxQ; Q is loading in Q-Register

Step 5: Check the Q_0 Q_{-1} bits which LSB of Q-Register and Q_{-1} bit.

Step 6: Check for following cases

if Q_0 Q_{-1} is 00; Perform A=A+0

if Q_0 Q_{-1} is 01; Perform A=A+M

if Q_0 Q_{-1} is 10; Perform A=A-M [this performed as A=A+2's Complement of M]

if Q_0 Q_{-1} is 11; Perform A=A+0

Step 7: Perform arithmetic right shift on (A Q Q_{-1})

Step 8: Repeat the Step 6 and Step 7 n-number of times

Step 9: The final result is stored in AQ registers

NOTE: Always select the n-bit such that the values should be in range of signed representation

Ex: Multiply (+3) X (+5)

Ans: this will come in the range for a 4-bit number representation as studied earlier

Ex: Multiply (-8) X (+31)

Ans: To find the representation of numbers we will first identify the numbers in 2's complement representation.

(+31) = 011111 [writing zero in front is necessary for a positive number]

To find the value of -8 first we will write the value of +8 as per the number of bit present in (+31) which is 6-bits,

(+8) = 001000; To find (-8) we will take 2's complement of (001000)

$$(001000) \xrightarrow{1's\ \text{Complement}} (110111) \xrightarrow{2's\ \text{Complement}} (111000)$$

Hence (-8) will be (111000)

So we will use booth's algorithm for (011111) X (111000)

Ex: Find the n value for multiplication of (-127) X (+66)

Ans: Based on analysis given below we will select n = 8 so that both number will come in range

for n=4 Total Number can be presented $2^4 = 16$
So 8 numbers will be positive i.e. from 0 to +7
and 8 numbers will be negative i.e. from -1 to -8
Range: (-8) ← (-1) ↔ (0) → (+7)

for n=5 Total Number can be presented $2^5 = 32$
So 16 numbers will be positive i.e. from 0 to +15
and 16 numbers will be negative i.e. from -1 to -16
Range: (-16) ← (-1) ↔ (0) → (+15)

for n=6 Total Number can be presented $2^6 = 64$
So 32 numbers will be positive i.e. from 0 to +31
and 32 numbers will be negative i.e. from -1 to -32
Range: (-32) ← (-1) ↔ (0) → (+31)

for n=7 Total Number can be presented $2^7 = 128$
So 64 numbers will be positive i.e. from 0 to +63
and 64 numbers will be negative i.e. from -1 to -64
Range: (-64) ← (-1) ↔ (0) → (+63)

for n=8 Total Number can be presented $2^8 = 256$
So 128 numbers will be positive i.e. from 0 to +127
and 128 numbers will be negative i.e. from -1 to -128
Range: (-128) ← (-1) ↔ (0) → (+127)

So numbers will be in 8-bit format

(+66) = 01000010

for (-127) = we will take 2's complement of +127

(01111111) $\xrightarrow{1's\ \text{Complement}}$ (10000000) $\xrightarrow{2's\ \text{Complement}}$ (10000001)

(-127) = 10000001

For more question on Booth's algorithm refer class notes.

TOPIC 10: Integer Division

Method 1 Restoring Division Method :

Question: Let Divisor M = 0011 (3) and Dividend Q = 1000 (8)
Step 1: Take two registers A(5-Bit) and Q (4-Bit)
Step 2: Initially A = 0 and Q is loaded with dividend value 1000
So the content of registers will be

A	Q
00000	1000

Step 3: Each operation contain 3 steps

(1) Left Shift: In this step the contents of AQ are left shifted which creates a vacant bit at Q_0

(2) A-M is performed using A + 2's Complement of M

Note: While taking 2's Complement of M the total number bits in M and should be equal to A

(3) Set the Q_0 vacant bit: After performing A = A - M (or A = A+2's Comp M) the Q_0 is set as following:

Q_0 = 0 if MSB of A is 1 and Perform Restore

Q_0 = 1 if MSB of A is 0

Perform the operation till n number of operation where n is number of bits in Q register. Finally Reminder will be in A and Quotient will be in Q

Answer:

A	Q	
00000	1000	
00001	000V	L-Shift
11110	000V	A = A-M
00001	0000	Set Q0 bit and Restore
00010	000V	L-Shift
11111	000V	A = A-M
00010	0000	Set Q0 bit and Restore
00100	000V	L-Shift
00001	000V	A = A-M
00001	0001	Set Q0 bit
00010	001V	L-Shift
11111	001V	A = A-M
00010	0010	Set Q0 bit and Restore

Note: Arrow line represents setting of Q0 bit based on MSB of A (Actually, Opposite to MSB value of A)

Final Results in AQ
A (Reminder) = 00010 Q (Quotient) = 0010

Try This Question: Divisor (M) = 00011; Dividend (Q) = 10110

Method 2 Non - Restoring Division Method :

Question: Let Divisor M = 0011 (3) and Dividend Q = 1000 (8)
Note: If Q contains n-bits than A should contain (n+1) bits
Step 1: Take two registers A(5-Bit) and Q (4-Bit)
Step 2: Initially A = 0 and Q is loaded with dividend value 1000
So the content of registers will be

A	Q
00000	1000

Step 3: Each operation contain 2 steps
(1) Left Shift: In this step the contents of AQ are left shifted which creates a vacant bit at Q_0
(2) Check MSB of A after left shift
if MSB of A = 0; Perform A = A - M
Note: While taking 2's Complement of M the total number bits in M and should be equal to A
if MSB of A = 1; Perform A = A + M
Set Q0 bit based on result
Q_0 = 0 if MSB of A is 1 and Perform Restore
Q_0 = 1 if MSB of A is 0

Perform the operation till n number of operation where n is number of bits in Q register.

Step 4: Finally after performing all operation check the MSB of A
if it is 0; No change in result
if it is 1; take A = A+M

Answer:

A	Q	
00000	1000	
00001	000V	L-Shift
11110	0000	A = A-M
11100	000V	L-Shift
11111	0000	A = A-M
11110	000V	L-Shift
00001	0001	A = A-M
00010	001V	L-Shift
11111	0010	A = A-M

Note: Arrow line represents setting of Q0 bit based on MSB of A (Actually, Opposite to MSB value of A)

As MSB of A is 1 in final result so we will take A = A + M

11111 + 00011 = 00010

Final Results in AQ

A (Reminder) = 00010 Q (Quotient) = 0010

Try This Question: Divisor (M) = 00011; Dividend (Q) = 10110