Chapter 2

ALU and Data Path, CPU Control Design

LEARNING OBJECTIVES

- *Arithmetic and logic unit*
- *Fixed-point arithmetic operation*
- *Floating point arithmetic operation*
- *BCD*
- *Data path*
- *CPU control design*
- *Instruction cycle*
- *Control unit*
- *Control of processor*
- *Function of control unit*
- *Design of control unit*
- *Types of micro-instructions*
- *Micro-instruction sequencing*
- *RISC and CISC*
- *RISC characteristic*
- *CISC characteristic*

ALU (Arithmetic and Logic Unit)

ALU performs arithmetic and logical operations on data (*see* Figure 1).

Figure 1 ALU inputs and outputs

- Data are presented to ALU in registers and the results of an operation are stored in registers.
- Registers are temporary storage locations within the processor that are connected by signal paths to ALU.
- The control unit provides signals that control the operation of ALU and the movement of data into and out of the ALU.
- Here we will discuss
  1. Fixed-point arithmetic operations
  2. Floating-point arithmetic operations
  3. BCD data arithmetic operations

Fixed-point Arithmetic Operations

Fixed-point representation

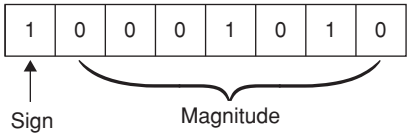
The numbers may be positive, zero or negative. So we have two types of numbers:

Unsigned numbers Only zero and positive integers can be represented. All bits represent magnitude and no need of sign.

Signed numbers In signed representation, the most significant bit represents the sign. If the number is positive, the MSB is 0 and remaining bits represent magnitude. If the number is negative, we have three techniques to represents that number:

1. **Signed magnitude representation:** In signed magnitude representation, the MSB represents sign and remaining bits represents magnitude. If the number is negative then the MSB is 1.

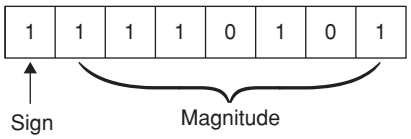
   Example: Signed magnitude representation of –10 =



2. **Signed 1's complement representation:** In signed 1's complement representation, the MSB bit is 1. The remaining bits of its signed magnitude bits are inverted i.e., convert 0's to 1's and 1's to 0's to obtain 1's complement.

Example:

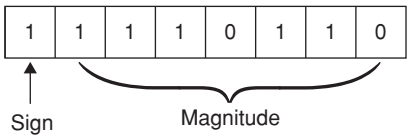
Signed 1's complement (–10) =



3. **Signed 2's complement representation:** To get signed 2's complement representation, add 1 to the signed 1's complement of that number.

Example:

Signed 2's complement (–10) =



Fixed-point arithmetic operations

We will discuss the following operations using signed magnitude data and signed 2's complement data.

1. Addition
2. Subtraction
3. Multiplication
4. Division

Addition and subtraction using signed magnitude data Consider two numbers whose magnitude is represented as A and B. When the signed numbers are added or subtracted, there are eight different conditions to consider, depending on the sign of the numbers and operation performed.

Operation	Add Magnitudes	Subtract Magnitudes ($A > B$)
$(+A) + (+B)$	$+(A + B)$	
$(+A) + (-B)$		$+(A - B)$
$(-A) + (+B)$		$-(A - B)$
$(-A) + (-B)$	$-(A + B)$	
$(+A) - (+B)$		$+(A - B)$
$(+A) - (-B)$	$+(A + B)$	
$(-A) - (+B)$	$-(A + B)$	
$(-A) - (-B)$		$-(A - B)$

Algorithm for addition (subtraction): When the signs of A and B identical (different), add the two magnitudes and attach the sign of A to the result. When the signs of A and B are different (identical), compare the magnitudes and subtract the smaller number from the larger. Choose the sign of result based on magnitudes of A and B.

Example: All eight cases for the numbers $A = 5$, $B = 2$.

$(+A) + (+B) = (+5) + (+2)$

$= 0101 + 0010 = 0111 = +7$

$(+A) + (-B) = (+5) + (-2)$

$= 0101 + 1010$

Take 2's complement of –2 and add it to 5

101

110

1] 011

↑

Discard

$\therefore$ result = + 3 ($A > B$)

$(-A) + (+B) = (-5) + (+2)$

$= 1101 + 0010$

add 2's complement of –5 to 2

011

010

101

As MSB is 1 take 2's complement to get original number i.e., 011.

Result = 1011 = –3 ($\because A > B$)

$(-A) + (-B) = (-5) + (-2)$

$= 1101 + 1010$

101

010

Result = 1111 = –7

Similarly we can perform the subtractions using signed magnitude data.

Hardware implementation:

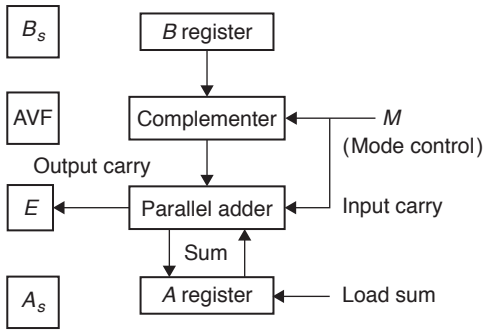


Figure 2 Hardware implementation for addition and subtraction.

Figure 2 shows the hardware implementation for addition and subtraction operations. It consists of registers A and B and sign flip-flops A_s and B_s. Subtraction is done by adding A to the 2's complement of B. The output carry is transferred to flip-flop E and add overflow flip-flop AVF holds the overflow bit when A and B are added. The addition is done through the parallel adder. The output of adder is sent to 'A' register. The complementer provides an output of B or complement of B depending on the state of mode control M. When $M = 0$, the output equal to $A + B$, when $M = 1$, the output equal to $A + \bar{B} + 1$, i.e., $A - B$.

Addition and Subtraction with signed 2's complement data

Addition: In 2's complement representation, addition proceeds as if the two numbers were unsigned integers. If the result of the operation is positive, we get a positive number

in 2's complement form, which is same as in unsigned integer form. If the result of the operation is negative, we get a negative number in 2's complement form.

Example:

+5 = 0101
+2 = 0010
0111 = +7

+5: 0101
–2: 1110
10011 = +3
–5: 101
+2: 0010
1101

As the result is negative, take 2's complement of result to get original number, i.e., 0011 and the answer is –3.

–5: 1011
–2: 1110
11001

As the result is negative take 2's complement to get original number, i.e., 0110 + 1 = 0111.

∴ Answer is –7.

Note: If two numbers are added and they are both positive or both negative, then overflow occurs if the result has the opposite sign.

Subtraction: To subtract subtrahend from minuend, take the 2's complement of subtrahend and add it to the minuend.

Example:

+5: 0101
+2: 0010

To subtract these two numbers add 2's complement of 2 to 5.

+5: 0101
–2: 1110
10011 = +3
+5: 0101
– 2: 1110
2's complement of –2 = 0010.
+5: 0101
+2: 0010
0111 = +7

Similarly for the other cases we can perform the subtraction.

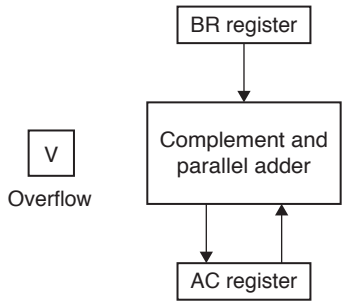


Figure 3 Hardware implementation for signed 2's complement addition and subtraction:

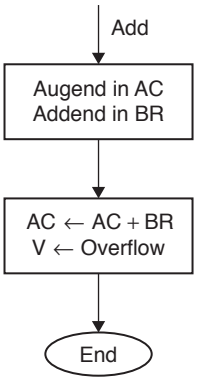


Figure 4 Flowchart for addition in 2's complement form

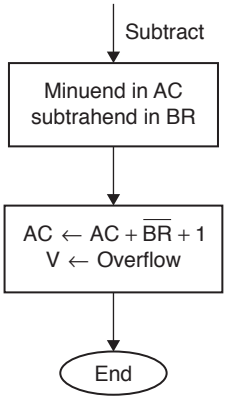


Figure 5 Flowchart for subtraction of 2's complement data

Multiplication of signed magnitude data

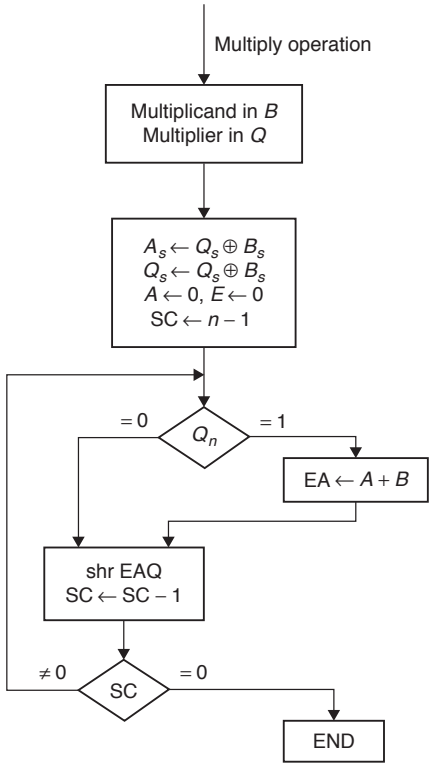


Figure 6 Flowchart for multiplication of signed magnitude data

Multiplication of two fixed point binary numbers in signed magnitude representation is a process of successive shift and add operations (*see* Figure 6).

Example 1: Multiply the two numbers −7 and +8, using 5-bit registers.
−7 = 10111
+8 = 01000

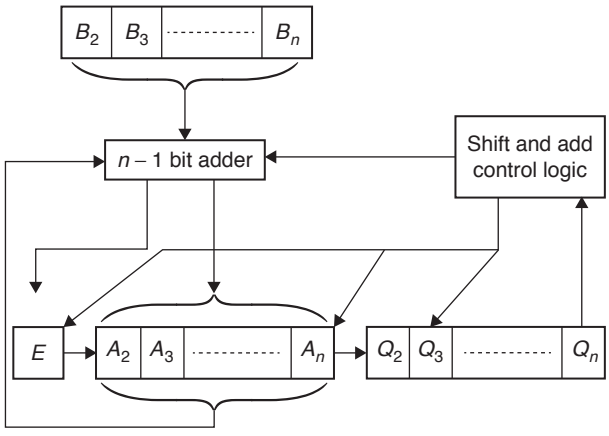
By excluding sign-bits, the multiplicand, $B = 0111$ and multiplier $Q = 1000$. Initially $A = 0000$, SC is sequence counter contains number of bits in multiplier magnitude.

Here SC = 4

Multiplicand B = 0111	***E***	***A***	***Q***	**SC**
Multiplier in *Q*	0	0000	1000	4
Last bit of *Q*, $Q_n = 0 \Rightarrow$ Shr *EAQ*	0	0000	0100	3
$Q_n = 0 \Rightarrow$ Shr *EAQ*	0	0000	0010	2
$Q_n = 0 \Rightarrow$ Shr *EAQ*	0	0000	0001	1
$Q_n = 1 \Rightarrow$ Add *B* to *A*	0	0000 0111 0111	0001	
Shr *EAQ*	0	0011	1000	0

$B \times Q = 00111000 = 56$
Sign = $Q_s \oplus B_s = 1 \oplus 0 = 1$
∴ Result = –56

Hardware for signed magnitude data multiplication: B_1, A_1, Q_1 represent the respective signs of the registers B, A, Q. Final result will be in AQ, which consist of $2n$-bits. (Here each register has n-bits).



Multiplication of Singed 2's complement data The straight forward multiplication will not work if either the multiplicand or the multiplier is negative. There are number of ways to perform multiplication of signed 2's complement data. One such a technique is *Booth's multiplication algorithm*. The following flowchart depicts about Booth's algorithm (see figure 7).

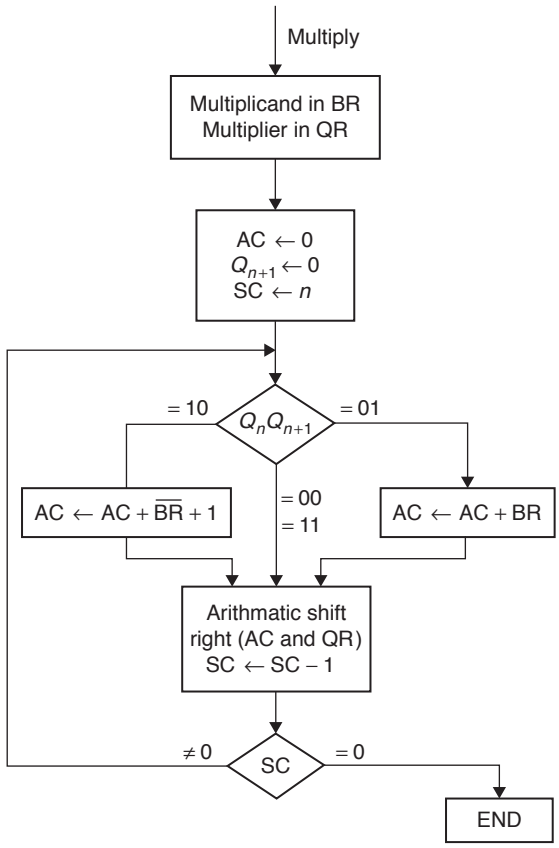


Figure 7 Booth's multiplication algorithm

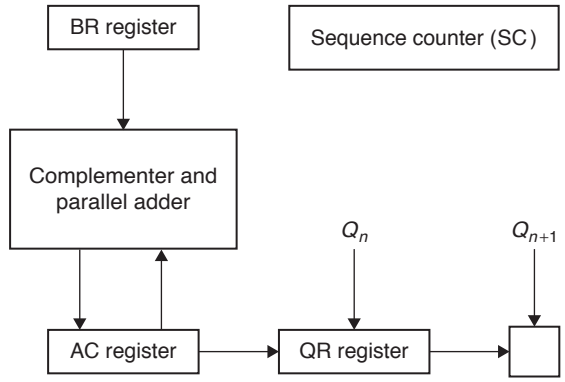
An additional 1-bit register placed logically to the right of the LSB(Q_n) of Q register designated Q_{n+1}.

Example 2: Multiply the two numbers −7 and +8 using booth's algorithm, using 5 bits.
BR = −7 = 11001
QR = +8 = 01000
Initially AC = 00000, $Q_{n+1} = 0$, SC = 5

$Q_n\, Q_{n+1}$	**BR = 11001** $\overline{\text{BR}} + 1 = 00111$	**AC**	**QR**	Q_{n+1}	**SC**
	Initial	00000	01000	0	5
00	ashr(AC and QR)	00000	00100	0	4
00	ashr(AC and QR)	00000	00010	0	3
00	ashr(AC and QR)	00000	00001	0	2
10	subtract BR ashr(AC and QR)	00000 00111 00111 00011			
			10000	1	1
01	add BR ashr(AC and QR)	00011 11001 11100 11110	10000 01000	1 0	1 0

∴ Result = 1111001000 = –56

Hardware implementation for Booth's algorithm:



Note: These two multiplication algorithms are sequential but we can also do the operation by means of a combinational circuit that forms the product bits all at once. The circuit consist of AND gates and adders.

Division algorithms

Division of signed magnitude data: Division of signed magnitude data is a process of successive compare, shift and subtract operations.

Example:

Dividend = 0111000000
Divisor = 10001

```
10001) 0111000000 (11010
       –10001
        010110
        –10001
         0010100
          –10001
           000110
```

Hardware implementation:

- The hardware implementation of division is same as multiplication, instead of shifting the divisor to the right, the dividend or partial remainder is shifted to the left, thus leaving the two numbers in the required relative position.
- The divisor is stored in the B register and the double-length dividend is stored in registers A and Q. The dividend is shifted to the left and the divisor is subtracted by adding its 2's complement value. The information about the relative magnitude is available in E.
- If $E = 1$, it signifies that $A \geq B$. A quotient bit 1 is inserted into Q_n and the partial remainder is shifted to the left to repeat the process.
- If $E = 0$, it signifies that $A < B$ so the quotient is Q_n remains a 0. The value of B is added to restore the partial remainder in A to its previous value. The partial remainder is shifted to the left and the process is repeated again until all quotient bits are formed.
- Finally, the quotient is in Q and remainder is in A. This method is called *restoring method*.

Divide overflow:

- A divide overflow condition occurs if the high-order half bits of the dividend constitute a number greater than or equal to the divisor.
- A division by zero must be avoided.

Other algorithms for division: Two other methods are available for dividing numbers:

Comparison method: To divide the two numbers A and B in comparison method, they are compared prior to the subtraction operation. If $A \geq B$, B is subtracted from A. If $A < B$ nothing is done. The partial remainder is shifted left and the numbers are compared again.

Non-restoring method: To divide two numbers A and B is non-restoring method, B is not added if the difference is negative but instead, the negative difference is shifted left and then B is added.

Floating-point Arithmetic Operations

Floating-point representation

Fixed-point representation allows representation of numbers with fractional component as well. But this approach has limitations. It is not possible to represent very large numbers and very small numbers in fixed point representation.

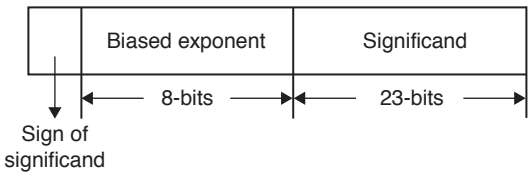
In floating-point representation, the numbers can be represented in the form,

$\pm S \times B^{\pm E}$

The three fields are
Sign: plus or minus
Significand: S
Exponent: E
are stored in a binary word.

The base B is implicit and need not be stored because it is same for all the numbers. It is assumed that the radix point is to the right of the left most or most significant bit of the significand, i.e., there is one bit to the left of the radix point.

32-bit floating-point format: The left most bit stores the sign of the number. The exponent value is stored in next 8-bits. This is represented in biased representation.



Biased representation: In biased representation, a fixed value, called the bias, is subtracted from the exponent field to get the true exponent value.

Bias = $(2^{k-1} - 1)$, where k = number of bits in binary exponent. In IEEE 32-bit floating point representation, bias = $2^7 - 1 = 127$.

And the range of true exponents is −127 to +128.

The advantage of biased representation is that non-negative floating point numbers can be treated as integers for comparison purposes.

The last portion of word is the siginificand.

Normalized numbers:

- To simplify operation on floating point numbers, it is typically required that they must be normalized.
- A normalized number is one in which the most significant digit of significand is non-zero.
- For base-2 representation, a normalized number is therefore one in which the MSB of the significand is one.
- Normalized non-zero number is one in the form $\pm 1.bbb\cdots b \times 2^{\pm E}$, where b is either binary digit 0 or 1.
- As the MSB is always one, it is unnecessary to store this bit. Thus the 23-bit field is used to store a 24-bit significand with a value in the half open internal (1, 2).
- A number may be normalized by shifting the radix point to the right of the leftmost 1 bit and adjusting the exponent accordingly.

Example: In 32-bit floating representation of $-1.6328125 \times 2^{-20}$,

Sign = 1 (as the number is negative)

$(.6328125)_{10} = (.1010001\ldots)_2$

Exponent = −20

Biased exponent = 127 − 20 = 107

= 1101011

$\therefore -1.6328125 \times 2^{-20}$

= 1 01101011 10100010000000000000000

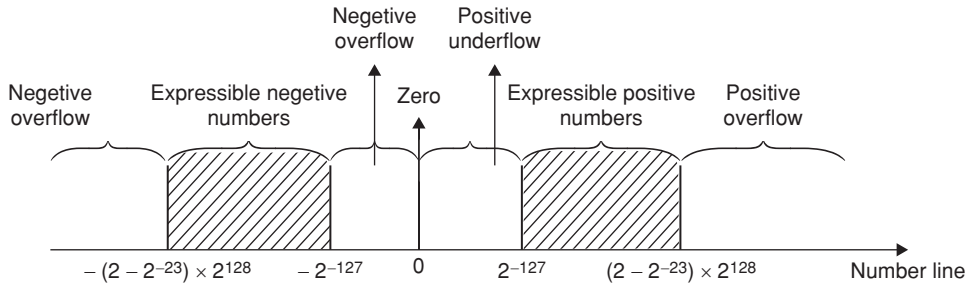
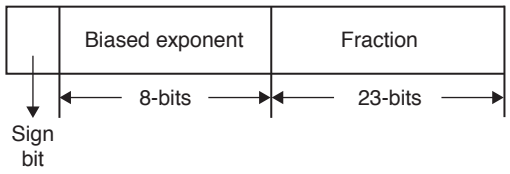
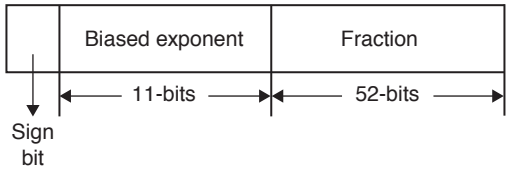


Figure 8 Range of expressible numbers in a 32-bit floating point format

IEEE standard for binary floating-point representation



Single precision format



Double precision Format

Floating-point arithmetic

(i) Addition and subtraction: The algorithm consists the following phases:

1. Check for zeros
2. Align the significands/mantissas
3. Add or subtract the significands
4. Normalize the result

Example:

$(123 \times 10^0) + (234 \times 10^{-2})$

$=123 \times 10^0 + 2.34 \times 10^0 = 125.34 \times 10^0$

(ii) Multiplication: The steps to multiply two floating point numbers are

1. Check for zeros
2. Add the exponents
3. Multiply the significands
4. Normalize the result

(iii) Division: The steps to divide two floating point numbers are

1. Check for zeros
2. Initialize registers and evaluate the sign
3. Align the dividend
4. Subtract the exponents
5. Divide the significands

Binary-Coded Decimal (BCD) Arithmetic Operations

Computers capable of performing decimal arithmetic must store the data in binary-coded form.

Example: BCD of 239 = 0010 0011 1001

BCD addition

In BCD each digit do not exceed 9, so the sum of two BCD digits cannot be greater than 9 + 9 + 1 = 19, the 1 in the sum being an input carry. When the binary sum of two BCD digits is greater than 1001, we obtain a non-valid BCD representation. The addition of binary 6 (0110) to the binary sum converts it to the correct BCD representation and also produces an output carry as required.

Example:

239 = 0010 0011 1001
426 = 0100 0010 0110
665 0110 0101 1111 >1001
0110
0110 0110 0101
= 665

BCD subtraction

- Perform the subtraction by taking the 9's or 10's complement of the subtrahend and adding it to the minuend.
- The 9's complement of a decimal digit represented in BCD can be obtained by complementing the bits in the coded representation of the digit, provided a correction is included.

There are two possible correction methods:

1. Binary 1010 is added to each complemented digit and the carry discarded after each addition.

Example:

9's complement of 7 = 2
7 in BCD = 0111.
Complement of 7 = 1000
Add 1010 = 1010
1] 0010 = 2
↑
Discard

2. Binary 0110 is added before the digit is complemented.

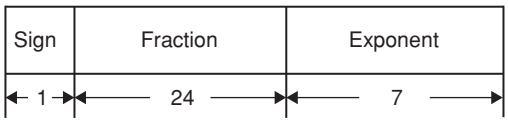
Example:

BCD of 7 = 0111
Add 0110 = 0110
1101
Complement = 0010 = 2

Example 4: Which of the following multiplier bit pattern of Booth's multiplication algorithm gives worst case performance?
(A) 01010101….0101
(B) 000000….0000
(C) 11111111….1111
(D) 011110111110….01110

Solution: (A)
Booth's multiplication algorithm works well with consecutive 0's or 1's. But it gives worst case performance when the multiplier consists of alternative 0's and 1's. (As 01, 10 pattern leads to addition and subtractions).

Example 5: Consider the following 32-bit floating point representation scheme as shown in the format below:



A value is specified by three fields:
Sign field: 1 bit (0 for positive and 1 for negative values)
Fraction: 24-bits (with binary point being at the left end of fraction bits)
Exponent:7-bits (in excess-64 signed integer representation)
The base of exponentiation is 16. The sign bit is in MSB. Then the normalized floating point representation of – 6.5 is

(A) E8000042 (B) E1000012
(C) D8000841 (D) D0000042

Solution: (D)
Here sign = 1 as the number is negative.
$(-6.5)_{10} = (-0110.1)_2$
$= (-1.101 \times 2^2)$
Fraction = 10100000000000000000000
Exponent = Excess – 64 exponent
= 64 + 2 = 66 = 1000010
$\therefore (-6.5)_{10}$
= 1 10100000000000000000001000010
= D0000042.

Data Path

Data path consists of the components of the processor that performs arithmetic operations.

Components of data path: ALU is just one data path building block. Other components are

1. Computational Components, which consist of combinational circuits (output follow inputs)

Example: ALU.

2. State components, which consists of sequential circuits (output changes on clock edge)

Example: Registers.

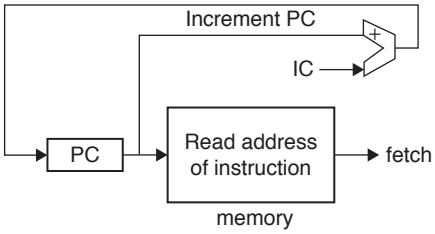
Example: The sequence of steps for the addition of two registers content are

1. $R1_{out}, X_{in}$
2. $R2_{out}$, Choose X, ADDITION, Y_{in}
3. $Y_{out}, R3_{in}$

(Each step executed in a single clock cycle).

- Data path and control unit forms the processing unit of a computer. The Data path includes ALU, multiplexers, all registers (like PC, IR) etc.

Example Data Path Design:

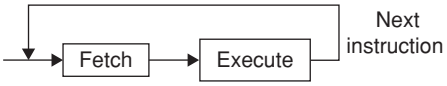


CPU Control Design

Instruction Cycle

A program residing in the memory unit of the computer consists of a sequence of instructions. The program is executed in the computer by going through a cycle for each instruction. Each instruction cycle in turn is subdivided into a sequence of sub cycles. For example, the phases of instruction cycle may be

1. Fetch
2. Decode
3. Read effective address
4. Execute, etc.



Instruction cycle

The cycle will be repeated, till all the instructions are executed.

Each phase is made up of more fundamental operations, called micro-operations.

Example micro-operations: Transfer between registers, simple ALU operation, etc.

Control Unit

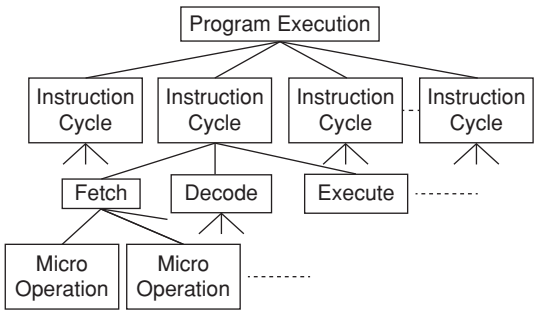
The control unit of a processor performs two tasks:

1. It causes the processor to execute micro-operations in the proper sequence, determined by the program being executed.
2. It generates the control signals that cause each micro-operation to be executed.

We now discuss the micro-operations of various phases of instruction cycle.

Micro-operation

- These are the functional or atomic operations of a processor.



Fetch Cycle: 'Fetch' stage of an instruction occurs at the beginning of each instruction, which causes an instruction to be fetched from memory. The micro-operations involved in fetch phase are

t_1: MAR ← PC (move contents of PC to MAR)
t_2: MBR← memory; PC ← (PC) + I (move contents of MAR location to MBR and increment PC by I)
t_3: IR← (MBR) · (move contents of MBR to IR)

Here *I* is instruction length.
Each micro-operation can be performed within the time of a single time unit.

Execute Cycle: For a machine with *N* different opcodes, there will be *N* different sequence of micro-operations. For the execution of following instruction.
Add R_1, *X*, the micro-operations will be

t_1: MAR← (IR(Address))
t_2: MBR← memory
t_3: R_1← (R_1) + (MBR)

Control of the Processor

(i) Functional requirements of control unit: Let us consider the following concepts to the characterization of a CU.

1. Define the basic elements of the processor.
2. Describe the micro-operations that the processor performs.
3. Determine the functions that the control unit must perform to cause the micro-operations to be performed.

(ii) Basic elements of processor:

- ALU
- Registers
- Internal data path: Used to move data between registers and between register and ALU.
- External data path: Used to link registers to memory and *input–output* modules, often by means of a system bus.
- Control unit: Causes operations to happen within the processor.

(iii) Micro-operations of processor:

- Transfer data from one register to another.
- Transfer data from a register to an external interface.

- Transfer data from an external interface to a register.
- Perform an arithmetic or logic operation, using registers for input and output.

(iv) Control unit tasks:

- **Sequencing:** The control unit causes the processor to step through a series of micro-operations in the proper sequence, based on the program being executed.
- **Execution:** The control unit causes each micro-operation to be executed.

(v) Control signals: For the control unit to perform its function, it must have inputs that allow it to determine the state of the system and outputs that allows it to control the behaviour of the system. These are external specifications of the control unit.

Internally, the control unit must have the logic required to perform its sequencing and execution functions.

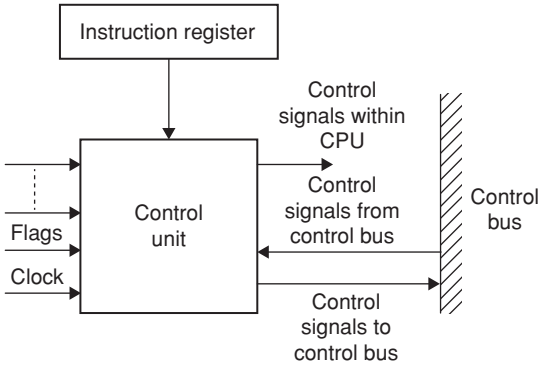


Figure 9 Block diagram of control unit

(a) Clock: This is how the control unit 'keeps time.' The control unit causes one micro-operation to be performed for each clock pulse. This is referred as processor cycle time or clock cycle time.

(b) Instruction registers: The opcode of current instruction is used to determine which micro-operations to perform during the execute cycle.

(c) Flags: Used to determine the status of the processor and outcome of previous ALU operations.

(d) Control signals from control bus: The control bus portion of system bus provides signals to the control unit.

(e) Control signals within the processor:

1. Those that cause data to be moved from register to another.
2. Those that activate specific ALU functions.

(f) Control signals to control bus:

1. Control signals to memory.
2. Control signals to input–output modules.

Totally, there are three types of control signals:

1. Those that activate ALU function.
2. Those that activate a data path.
3. Those that are signals on the external system bus.

Functions of Control Unit

- The control unit directs the entire computer system to carry out stored program instructions.
- The control unit must communicate with both the Arithmetic Logic Unit and Main memory.
- The control unit instructs the arithmetic logic unit by which, logical or arithmetic operation is to be performed.
- The control unit coordinate the activities of the other two units as well as all peripheral and auxiliary storage devices linked to the computer.

Design of Control Unit

Control unit generates control signals using one of the two organizations

(1) Hardwired control unit
(2) Micro-programmed control unit.

Hardwired control unit

- It is implemented as logic circuits (gates, flip-flops, decoders, etc.) in the hardware.
- It is very complicated if we have a large control unit.
- In this organization, if the design has to be modified or changed. It requires changes in wiring among the various components. Thus the modification of all the combinational circuits may be very difficult.

Architecture of hardwired control unit An example hardwired control unit is shown in Figure 9.

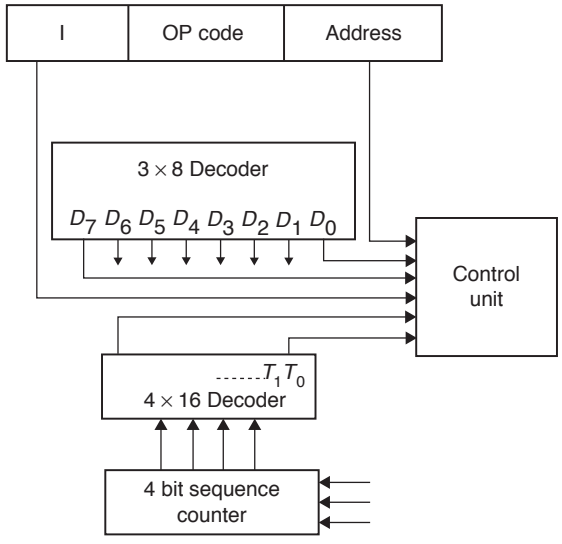


Figure 10 Hardwired control unit

The above control unit consists of:

- Instruction Register
- Number of control logic gates

- Two decoders
- 4-bit sequence counter.
- An instruction read from memory is placed in the instruction register (IR)
- The instruction register is divided into three parts: the *I* bit, operation code and Address part.
- First 12-bits (0-11) to specify an address, next 3-bits specify the operation code (op code) field of the instruction and last left most bit specify the addressing mode *I*.
  $I = 0$ for direct address
  $I = 1$ for indirect address
- First 12-bits are applied to the control logic gates.
- The Opcode bits (12-14) are decoded with 3 × 8 decoder.
- The eight outputs (D_0 through D_7) from a decoder go to the control logic gates to perform specific operation.
- Last bit 15 is transferred to a *I* flip flop designated by symbol *I*.
- The 4-bit sequence counter SC can count in binary from 0 through15.
- The counter output is decoded into 16 timing pulses T_0 through T_{15}.
- The sequence counter can be incremented by INR input or clear by CLR input synchronically.

Advantages:

- Hardwired control unit is fast because control signals are generated by combinational circuits.
- The delay in generation of control signals depends upon the number of gates.

Disadvantages:

- More is the control signal required by CPU, more complex will be the design of control unit.
- Modifications in control signal are very difficult. That means it requires rearranging of wires in the hardware circuit.
- It is difficult to correct mistake in original design or adding new features.

Micro-programming control unit

- A micro-programmed Control unit is implemented using programming approach. A sequence of micro-operations are carried out by executing a program consisting of microinstructions.
- Micro-program, consisting of micro instructions is stored in the control memory of the control unit.
- Execution of micro-instruction is responsible for generation of a set of control signals.

A micro-instruction consists of:

- One or more micro-instructions to be executed.
- Address of next micro-instruction to be executed.

(a) Micro-operations: The operations performed on the Data stored inside the registers are called Micro-operations.

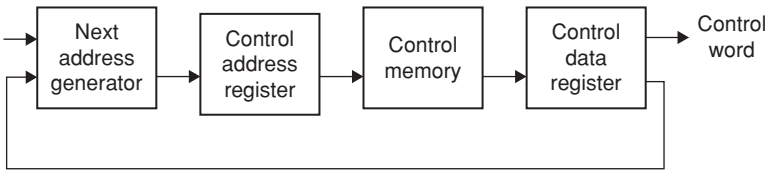
(b) Micro-programs: Micro-programming is the concept for generating control signals using programs. These programs are called Micro-programs.

(c) Micro-instructions: The instructions that make Micro-programs are called micro-instructions.

(d) Micro-code: Micro-program is a group of micro-instructions. Micro-program can also be termed as micro-code.

(e) Control memory: Micro-programs are stored in the read-only memory (ROM). That memory is called control memory.

(f) Architecture of Micro-Programmed Control Unit:



- The address of micro-instruction that is to be executed is stored in the control address register (CAR).
- Micro-instruction corresponding to the address stored in CAR is fetched from control memory and is stored in the control data register (CDR).
- This micro-instruction contains control word to execute one or more micro-operations.
- After the execution of all micro-operations of micro-instructions, the address of next micro-instructions is located.

Advantages:

- The design of micro-program control unit is less complex because micro-programs are implemented using software routines.
- The micro-programmed control unit is more flexible because design modifications, correction and enhancement is easily possible.
- The new or modified instruction set of CPU can be easily implemented by simply rewriting or modifying the contents of control memory.
- The fault can be easily diagnosed in the micro-program control unit using diagnostic tools by maintaining the contents of flags, registers and counters.

Disadvantages:

- The micro-program control unit is slower than hardwired control unit. That means to execute an instruction in micro-program control unit requires more time.

- The micro-program control unit is expensive than hard-wired control unit in case of limited hardware resources.
- The design duration of micro-program control unit is more than hardwired control unit for smaller CPU.

Types of Micro-instructions

Micro-instructions can be classified as

Horizontal micro-instruction

- Individual bits in horizontal micro-instructions correspond to individual control lines.
- These are long and allow maximum parallelism since each bit controls a single control line.
- No decoding needed.

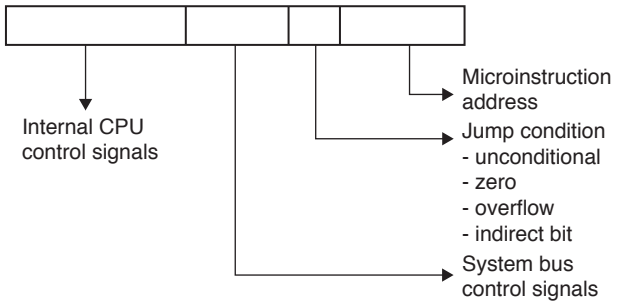


Figure 11 Horizontal micro-instruction format

Vertical micro-instruction

- Here, control lines are coded into specific fields within a micro-instruction.
- Decoders are needed to map a field of k-bits to 2^k possible combinations of control lines.

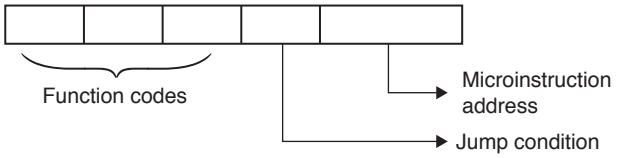


Figure 12 Vertical micro-instruction format

Example: A 3-bit field in a micro-instruction could be used to specify any one of eight possible lines.

- Hence these instructions are much shorter than horizontal ones.
- Control fields encoded in the same field cannot be activated simultaneously. Therefore vertical micro-instructions allow only limited parallelism.
- Decoding is necessary.

Micro-instruction Sequencing

Two concerns are involved in the design of a micro-instruction sequencing technique:

1. The size of micro-instruction: Minimizing size of control memory reduces the cost of that component.
2. The address-generation time:

A desire to execute micro-instructions as fast as possible. In executing a micro program, the address of next micro-instruction to be executed is in one of these categories.

1. Determined by IR
2. Next sequential address
3. Branch

Micro-instructions Execution

The Micro-instruction cycle has two parts:

1. Fetch
2. Execution

The effect of execution of a micro-instruction is to generate control signals. Some of the signals control points internal to the processor. The remaining signals go to the external control bus or other external interface.

Micro-instructions can be classified in a variety of ways.

1. Vertical/horizontal
2. Packed/unpacked
3. Hard/soft micro-programming
4. Direct/indirect encoding.

RISC AND CISC

One of the important aspects of computer architecture is the design of the instruction set for the processor. The instruction set chosen for a particular computer determines the way that machine language programs are constructed. There are two categories of computers based on instructions:

1. Complex instruction set computer (CISC)
2. Reduced instruction set computer (RISC)

CISC: A computer with a large number of instructions is classified as a complex instruction set computer.

RISC: A computer which has fewer instructions with simple constructs, so they can be executed much faster with in the CPU without having to use memory as often. This type of computer is classified as RISC.

CISC characteristics

- CISC provides a single machine instruction for each statement, That is written in a high level language so that compilation process is simplified and the over all computer performance improved.
- It has variable length instruction formats.
- It provides direct manipulation of operands residing in memory.
- Some instructions that perform specialized tasks and are used infrequently.
- A large variety of addressing modes

Drawback of CISC architecture As more instructions and addressing modes are incorporated into a computer, the more hardware logic is needed to implement and support them and hence this causes the computations to slow down.

RISC characteristics

- Reduce execution time by simplifying the instruction set of the computer.
- Fewer numbers of instructions
- Relatively fewer addressing modes
- Memory access is limited to load and store instructions.
- All operations are done with in the register of the CPU.
- Fixed - length, easily decoded instruction format.
- Single-cycle instruction execution.
- Hardwired rather than micro-programmed control.
- Relatively large number of registers.
- Uses overlapped register windows to speed - up procedure call and return.
- Efficient instruction pipeline.
- Efficient translation of high - level language programs into machine language programs by the compiler.

Example 6: An instruction set of a processor has 200 signals which can be divided into 5 groups of mutually exclusive signals as follows.

Group 1: 30 Signals
Group 2: 90 Signals
Group 3: 20 Signals
Group 4: 10 Signals
Group 5: 50 Signals

How many bits of the control words can be saved by using vertical micro-programming over horizontal microprogramming?

(A) 27 (B) 173
(C) 200 (D) 227

Solution: Horizontal micro-programming requires 200 signals. But vertical micro-programming uses encoding. So

Group 1 requires 5-bits ($\because 2^5 = 32$)
Group 2 requires 7-bits ($\because 2^7 = 128$)
Group 3 requires 5-bits ($\because 2^5 = 32$)
Group 4 requires 4-bits ($\because 2^4 = 16$)
Group 5 requires 6-bits ($\because 2^6 = 64$)

$\therefore$ Total bits required using vertical micro programming = 27

$\therefore$ Number of bits saved = 200 – 27 = 173

EXERCISE

Practice Problems I

Directions for questions 1 to 20: Select the correct alternative from the given choices.

1. Using two's complement arithmetic the resultant of 111100001111 – 110011110011 is
(A) 0010 0001 1111
(B) 0011 0000 1100
(C) 0010 0001 1101
(D) 0010 0001 1100

2. IEEE 32-bit floating point format of 384 is
(A) 0 10000111 00000000000000000000000
(B) 0 10000111 10000000000000000000000
(C) 0 00001000 00000000000000000000000
(D) 0 00001000 10000000000000000000000

3. Consider the following IEEE 32-bit floating point number:
0 01111110 10100000000000000000000.
What is the decimal value equivalent to given number?
(A) 0.25 (B) 3.25
(C) 0.8125 (D) 0.9375

4. What would be the bias value for a base-8 exponent in a 7-bit field?
(A) 8 (B) 16
(C) 63 (D) 64

5. The normalized value of the resultant of $8.844 \times 10^{-3} - 2.233 \times 10^{-1}$ is
(A) -2.144×10^{-1} (B) -0.2144
(C) -2×10^{-1} (D) -0.2

6. Which of the following is the correct sequence of micro-operations to add a number to the AC when the operand is a direct address operand and store the final result to AC?
(A) MAR←(IR(address))
MBR ← memory
$R_1 \leftarrow$ (AC) + (MBR)
(B) MAR ← IR(address)
MBR ← MAR
$R_1 \leftarrow$ (MBR)
$R_2 \leftarrow$ (AC) + (R_1)
AC ← R_2
(C) MAR ← (IR(address))
MBR ← Memory(MAR)
$R_1 \leftarrow$ (MBR)
$R_2 \leftarrow$ (AC) + (R_1)
AC ← (R_2)
(D) MAR ← (IR (address))
MBR ← Memory(MAR)
AC ← (AC) + (MAR)

Statement for linked answer questions 7 to 9: Assume that the control memory is 24 bits wide. The control portion of the micro-instruction format is divided into two fields. A micro-operation field of 13-bits specifies the micro-operation to be performed. An address selection field specifies a condition, based on the flags, that will cause a micro-instruction branch. There are eight flags.

7. How many bits are there in address selection field?
(A) 1 (B) 2
(C) 3 (D) 4

8. How many bits are there in address field?
(A) 8 (B) 9
(C) 13 (D) 24

9. What is the size of control memory in bits?
(A) 256 (B) 768
(C) 3328 (D) 6144

10. A simple processor has 3 major phases to its instructions cycle:
1. Fetch
2. Decode
3. Execute

Two 1-bit flags are used to specify the current phase in hardwired implementation. Will these flags required in micro-programming also?
(A) Yes
(B) No
(C) Cannot predict
(D) Depends on clock cycle time

11. In a 3-bus data path, the micro instructions format will be Opcode src1, src2, desti; The number of operations supported are 8 and the src1, src2 and desti require 20, 16 and 20 bits respectively.
The total number of horizontal microinstructions specified will be
(A) 2^{64} (B) 2^{8}
(C) 2^{56} (D) 2^{61}

12. What is the smallest positive normalized number represented using IEEE single precision floating point representation?
(A) 2^{-128} (B) $1 - 2^{-127}$
(C) 2^{-127} (D) 2^{-126}

13. A micro program control unit is required to generate a total of 30 control signals. Assume that during any micro instruction, almost two control signals are active. Minimum number of bits required in the control word to generate the required control signals will be
(A) 2 (B) 2.5
(C) 10 (D) 12

14. What is the fraction field of the single-precision floating point representation of 6.25?
(A) 1110 1000 0000 0000 0000 000
(B) 1001 0000 0000 0000 0000 000
(C) 1100 0000 0000 0000 0000 000
(D) 0110 0100 0000 0000 0000 000

15. Let the total number of control signals generated are n, then what is the number of bits allocated in control field of vertical micro programming?
(A) $n/2$ (B) n
(C) 2^n (D) $\log_2^n$

16. In a micro programmed control unit, a control field of one address control instruction has to support two groups of control signals. In group1 it is required to generate either one or none of the 32 control signals. In group 2 at most 5 from the remaining, what will be the number of bits needed for the control field?
(A) 8 (B) 10
(C) 35 (D) 37

17. Assume that the exponent e is constrained to lie in the range $0 \le e \le x$, with a bias of q, that the base is b and that the significant is P-digits in length.
What is the largest positive value that can be written is normalized floating point?
(A) $b^{x-q}(1 - b^{-p})$ (B) b^{-q-1}
(C) b^{-q-p} (D) $b^{x-q}(b^{-p} - 1)$

18. By using Booth's Multiplication algorithm. Below two numbers are multiplied:
Multiplicand: 0111 0111 1011 1101
Multiplier: 0101 1010 1110 1110
How many additions/subtractions are required for the multiplication of the above two numbers?
(A) 8 (B) 10
(C) 13 (D) 7

19. Let us assume, we are multiplying two positive integers 1101 and 1011. The multiplicand M is 1101 and Multiplier Q is 1011. What is partial product after second cycle?
(A) 0110 1101 (B) 1001 1110
(C) 0100 1111 (D) 1000 1111

20. The decimal representation of the 2's complement number 1101011 is
(A) 21 (B) −21
(C) 219 (D) 91

Practice Problems 2

Directions for questions 1 to 20: Select the correct alternative from the given choices.

1. A microprogrammed control unit
(A) is faster than a hard-wired control unit
(B) facilitates easy implementation of new instructions.
(C) is useful when very small programs are to be run.
(D) usually refers to the control unit of a micro-processor.

2. Micro-program is
(A) the name of a source program in micro computers.
(B) a primitive form of macros used in assembly language programming.
(C) a program of a very small size.
(D) the set of instruction indicating the basic elemental commands which directly control the operation of a system.

3. Programming that actually controls the path of signal or data within the computer is called
(A) System programming
(B) Micro-programming
(C) High-level language programming
(D) Assembly language programming

4. The instruction cycle time in a generic microprocessor is
(A) Longer than the machine cycle time
(B) Shorter than the machine cycle time
(C) Same as the machine cycle time
(D) Double the machine cycle time

5. Microprocessor unit or central processor unit consist of
(A) Control circuitry (B) ALU
(C) Memory (D) All of these

6. The exponent of a floating point number is represented in excess-N code so that
(A) the dynamic range is large
(B) overflow is avoided
(C) the precision is high
(D) the smallest number is represented efficiently

7. Using Booth's algorithm for Multiplication, the Multiplier −14 is coded as
(A) 11110 (B) 01110
(C) 10010 (D) 00010

8. Data Path consists of
(A) Registers (B) ALU
(C) Bus (D) All of these

9. A floating point number that has a '0' in MSB of mantissa is said to have ____
(A) Overflow (B) Underflow
(C) Normalization (D) Positive exponent

10. Let the Binary sum after BCD addition is stored in K, Z_8, Z_4, Z_2, and Z_1 Then the condition for a correction and output carry can be expressed as $C =$
(A) $K + Z_8 Z_4 + Z_8 Z_2$ (B) $K + Z_8 Z_4 + Z_4 Z_2$
(C) $K + Z_8 Z_2 + Z_8 Z_1$ (D) $K + Z_4 Z_2 + Z_2 Z_1$

11. Which of the following is an advantage of biased exponents?
(A) Convenient way to represent exponents
(B) Useful for conversion
(C) Convenient for comparison purposes
(D) All of these

12. Booth multiplication skips over runs of zeros and ones which reduces the number of add and subtract steps needed to multiply two n-bit numbers to n to a variable number whose average value n_{avg} is less than n what will be n_{avg}?
(A) $n/3$ (B) $n/4$
(C) $n/2$ (D) n

13. The sequence of events that happen during a fetch operation is:
(A) PC → memory → IR
(B) PC → MAR → memory → IR
(C) PC → MAR → memory → MDR → IR
(D) PC → memory → MDR → IR

14. Micro-programming is a technique for
(A) Programming input or output routines
(B) Programming the microprocessors
(C) Programming the control steps of a computer
(D) Writing small programs

15. In a micro program ____ specifies the address of Micro-instructions to be executed.
(A) AR (B) PC
(C) SP (D) CAR

16. Which one of the following statements is correct?
(A) Micro-programmed control unit is costlier and slow.
(B) Micro-programmed control unit are cheap and slow.
(C) Micro-programmed control unit is costlier and fast.
(D) Micro-programmed control unit are fast and cheaper.

17. Horizontal micro-instructions have
(A) High degree parallelism, more encoding of control information.
(B) High degree parallelism, little encoding of control information.
(C) Low degree parallelism, more encoding of control information.
(D) Low degree parallelism, little encoding of control information.

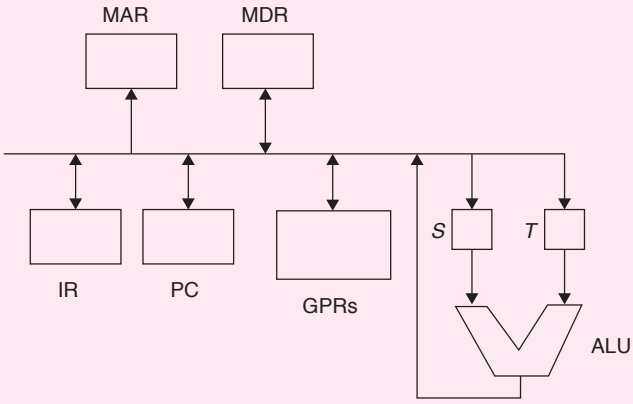
18. A vertical micro-instruction have ____.
(A) Short formats and considerable encoding of control information
(B) Long formats and considerable encoding of control information
(C) Short formats and little encoding of control information
(D) Long formats and little encoding of control information

19. Guard bits are used to
(A) avoid unnecessary loss of MSB
(B) avoid unnecessary loss of LSB
(C) the loss of MSB
(D) the loss of LSB

20. Which of the following is not the essential element of a number represented in floating-point notation?
(A) Exponent (B) Significand
(C) Sign (D) Normalization

Previous Years' Questions

Common data for questions 1 and 2: Consider the following data path of a CPU.



The ALU, the bus and all the registers in the data path are of identical size. All operations including incrementation of the PC and the GPRs are to be carried out in the ALU. Two clock cycles are needed for memory read operation—the first one for loading address in the MAR and the next one for loading data from the memory bus into the MDR.

1. The instruction 'add R_0, R_1' has the register transfer interpretation $R_0 \leftarrow R_0 + R_1$. The minimum number of clock cycles needed for execution cycle of this instruction is

(A) 2 (B) 3
(C) 4 (D) 5

2. The instruction 'call *Rn*, sub' is a two word instruction. Assuming that PC is incremented during the fetch cycle of the first word of the instruction, its register transfer interpretation is

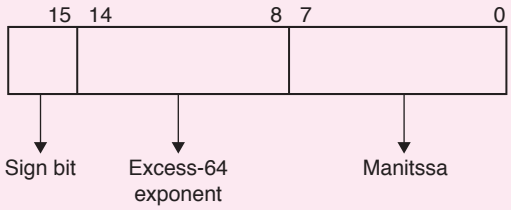
Rn ← PC + 1;

PC ← M[PC]

The minimum number of CPU clock cycles, needed during the execution cycle of this instruction is

(A) 2 (B) 3
(C) 4 (D) 5

Data for question 3: Consider the following floating-point format.



Mantissa is a pure fraction in sign-magnitude form.

3. The normalized representation for the above format is specified as follows. The mantissa has an implicit 1 preceding the binary (radix) point. Assume that only 0 's are padded in while shifting a field.

The normalized representation of the above number (0.239×2^{13}) is: **[2005]**

(A) 0*A* 20 (B) 11 34
(C) 49 *D*0 (D) 4*A E*8

4. In the IEEE floating point representation the hexadecimal value 0*x*00000000 corresponds to **[2008]**

(A) The normalized value 2^{-127}
(B) The normalized value 2^{-126}
(C) The normalized value + 0
(D) The special value + 0

5. *P* is a 16-bit signed integer. The 2's complement representation of *P* is $(F87B)_{16}$. The 2's complement representation of 8**P* is **[2010]**

(A) $(C3D8)_{16}$ (B) $(187B)_{16}$
(C) $(F878)_{16}$ (D) $(987B)_{16}$

6. The decimal value 0.5 in IEEE single precision floating point representation has **[2012]**

(A) fraction bits of 000 … 000 and exponent value of 0
(B) fraction bits of 000…000 and exponent value of –1
(C) fraction bits of 100…000 and exponent value of 0
(D) no exact representation

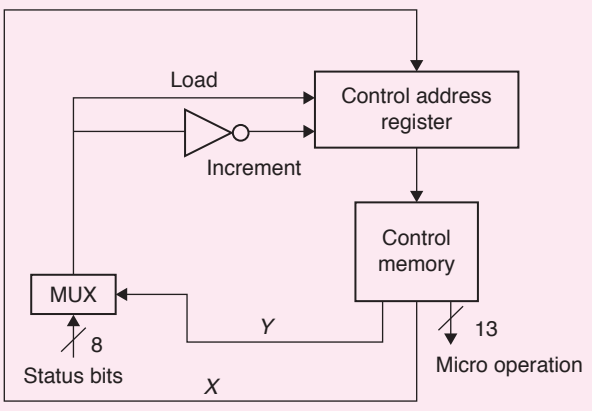
7. The smallest integer that can be represented by an 8-bit number in 2's complement form is **[2013]**

(A) –256 (B) –128
(C) –127 (D) 0

8. Let *A* = 1111 1010 and *B* = 0000 1010 be two 8-bit 2's complement numbers. Their product in 2's complement is **[2004]**

(A) 1100 0100 (B) 1001 1100
(C) 1010 0101 (D) 1101 0101

9. The microinstructions stored in the control memory of a processor have a width of 26 bits. Each microinstruction is divided into three fields, a micro-operation field of 13 bits, a next address field (*X*), and a MUX select field (*Y*), there are 8 status bits in the inputs of the MUX **[2004]**



How many bits are there in the X and Y fields, and what is the size of the control memory in number of words?

(A) 10, 3, 1024 (B) 8, 5, 256
(C) 5, 8, 2048 (D) 10, 3, 512

10. Consider the following sequence of micro-operations.

MBR ← PC
MAR ← X
PC ← Y
Memory ← MBR

Which one of the following is a possible operation performed by this sequence? **[2013]**

(A) Instruction fetch
(B) Operand fetch
(C) Conditional branch
(D) Initiation of interrupt service

11. For computers based on three-address instruction formats, each address field can be used to specify which of the following: **[2015]**

(S_1) A memory operand
(S_2) A processor register
(S_3) An implied accumulator register

(A) Either S_1 or S_2
(B) Either S_2 or S_3
(C) Only S_2 and S_3
(D) All of S_1, S_2 and S_3

12. Let X be the number of distinct 16 - bit integers in 2's complement representation. Let Y be the number of distinct 16 - bit integers in sign magnitude representation. They $x - y$ is _____ . **[2016]**

13. The n-bit fixed-point representation of an unsigned real number X uses f bits for the fraction part. Let $i = n - f$. The range of decimal values for X in this representation is **[2017]**

(A) 2^{-f} to 2^i (B) 2^{-f} to $(2^i - 2^{-f})$
(C) 0 to 2^i (D) 0 to $(2^i - 2^{-f})$

14 Consider the C code fragment given below.

```
typedef struct node  {
  int data;
  node* next;
}   node;
void join (node* m, node* n)  {
  node* p = n;
  while (p - >next != NULL)  {
    p = p - >next;
  }
  p - >next = m;
}
```

Assuming that m and n point to valid NULL-terminated linked lists, invocation of join will **[2017]**

(A) append list m to the end of list n for all inputs.
(B) either cause a null pointer dereference or append list m to the end of list n.
(C) cause a null pointer dereference for all inputs.
(D) append list n to the end of list m for all inputs.

15. The representation of the value of a 16-bit unsigned integer X in hexadecimal number system is BCA9. The representation of the value of X in octal number system is **[2017]**

(A) 571244 (B) 736251
(C) 571247 (D) 136251

16. Consider the following processor design characteristics.

I. Register-to-register arithmetic operations only
II. Fixed-length instruction format
III. Hardwired control unit

Which of the characteristics above are used in the design of a RISC processor? **[2018]**

(A) I and II only (B) II and III only
(C) I and III only (D) I, II and III

17. Consider the unsigned 8-bit fixed point binary number representation below:

$$b_7\, b_6\, b_5\, b_4\, b_3 \cdot b_2\, b_1\, b_0$$

where the position of the binary point is between b_3 and b_2. Assume b_7 is the most significant bit. Some of the decimal numbers listed below cannot be represented exactly in the above representation:

(i) 31.500 (ii) 0.875
(iii) 12.100 (iv) 3.001

Which one of the following statements is true? **[2018]**

(A) None of (i), (ii), (iii), (iv) can be exactly represented
(B) Only (ii) cannot be exactly represented
(C) Only (iii) and (iv) cannot be exactly represented
(D) Only (i) and (ii) cannot be exactly represented

Answer Keys

Exercises

Practice Problems I

1. D	**2.** B	**3.** C	**4.** C	**5.** A	**6.** C	**7.** C	**8.** A	**9.** D	**10.** B
11. A	**12.** D	**13.** C	**14.** B	**15.** D	**16.** B	**17.** A	**18.** B	**19.** B	**20.** B

Practice Problems I

1. B	**2.** D	**3.** B	**4.** C	**5.** D	**6.** D	**7.** C	**8.** D	**9.** B	**10.** A
11. C	**12.** C	**13.** C	**14.** C	**15.** D	**16.** A	**17.** B	**18.** A	**19.** B	**20.** D

Previous Years' Questions

1. B	**2.** B	**3.** D	**4.** D	**5.** A	**6.** B	**7.** B	**8.** A	**9.** A	**10.** D
11. A	**12.** 1	**13.** D	**14.** B	**15.** D	**16.** D	**17.** C			