ASSIGNMENT 01

Q:1 Convert the following pairs of decimal numbers to 5 bit signed 2's complement binary numbers and add them. State whether or not overflow occurs in each case.

Ans: i) 5 and 10 :

```
 00101        = +5
+01010        = +10
 01111          +15
```

Result : there is no overflow.

ii) 14 and 11 :

```
 01110        = +14
+01011        = +11
 11001          +25
```

Result : there is no overflow.

iii) -5 and 7 :

```
 11011.       = 2's complement  = -5
+00111                          = +7
 100010                           +2
 discard
```

Result : there is an overflow, so its discarded.

iv) -10 and -73 :

```
 10110        = 2's complement  = -10
+10011        = 2's complement  = -23
 101001                           -2
 discard
```

Result : there is an overflow, so its discarded.

Q:2 Multiply each of the following pairs of signed 2's complement number using Booth's Multiplication Algorithm:
A = Multiplicand.
B = Multiplier

Ans: i) A = 010111 , B = 110110

• 1 1 0 1 1 0 [0]
0 -1 +1 0 -1 0 Recoded Multiplier

• A = 2's complement = 101001

```
      0 1 0 1 1 1
    * 0-1+1 0-1 0
  ---------------
  0 0 0 0 0 0 0 0 0 0 0 0
  1 1 1 1 1 1 0 1 0 0 1 x
  0 0 0 0 0 0 0 0 0 0 x x
  0 0 0 0 1 0 1 1 1 x x x
  1 1 1 0 1 0 0 1 x x x x
  0 0 0 0 0 0 0 0 x x x x
  ---------------------------
  1 1 1 1 0 0 0 1 1 0 1 0
  ↑
discarded
```

Answer: 1111 0001 1010

ii) A = 110011 , B = 101100

• 1 0 1 1 0 0 [0]
-1 +1 0 -1 0 0
↓
Recoded Multiplier

• A = 2's complement = 001101

```
          110011
        * -1+10-100
  000000000000
  00000000000x
  000000110 1xx
  0000000000xx
  1111 0011xxxx
  0001 101xxx x
 10001 000001 00
 I
```

discarded

Answer = 000100000100

iii) A = 110101 , B = 011011

- 0 1 1 0 1 1 [0]
- +1 0 -1 +1 0 -1

Recoded Multiplier

- A 2's complement = 001011

```
          110101
        * +1 0 -1 +1 0 -1
  000000 0 0 1 011
  000 0000 0 0 0 0x
  111 1 1 1 0 1 0 1xx
  00000 1 0 1 1xxx
  00000 000 0 xxxx
  1111 01 01 xxxxx
 1 1111 01 10 1 0 111
 I
```

discarded

Answer = 111011010111

iv) A = 001111 , B = 001111

• 0 0 1 1 1 1 [0]
  0 +1 0 0 0 -1

A's 2's complement = 110001

```
    001111
  * 0+1000-1
  ------------
  111111110001
  000000000000 x
  000000000000 xx
  000000000000 xxx
  000011110000 xxxx
  000000000000 xxxxx
  ------------------
 1000011110001
 |
 discarded
```

Answer = 000011110001

Q:3 Perform division operation on the following unsigned numbers using the restoring method:

Ans: DIVIDEND = 10101 , DIVISOR = 00100

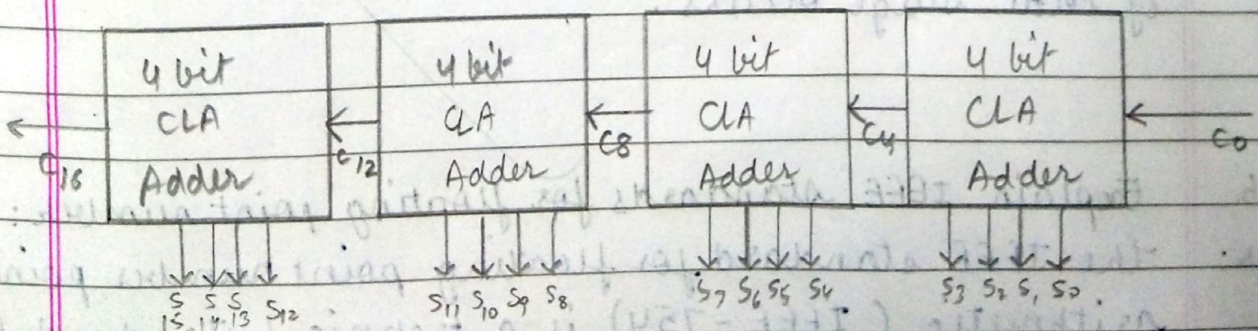
Divisor = 00100 , B = 000100 = 111100

OPERATION	A	B	COUNT
Initialization	000000	10101	= 5
Left Shift	000001	0101 [ ]	
A = A - B	111100	↓	
	111101	01010	
A = A + B	000100		
	000001	01010	

	A	Q	Count
Left Shift	000010	1010□	= 4
A = A − B	111100		
	111110	10100	
A = A + B	000100		
	000010	10100	
Left Shift	000101	0100□	= 3
A = A − B	111100		
	000001	01001	
~~A = A + B~~			
Left Shift	000010	1001□	= 2
A = A − B	111100		
	111110	10010	
A = A + B	000100		
	000010	10010	
Left Shift	000101	0010□	= 1
A = A − B	111100		
	000001	00101	= 0

Q.4 Design a 16-bit Carry look Ahead using 4 bit adder. Also write the expression for c_{i+1}.

Ans:

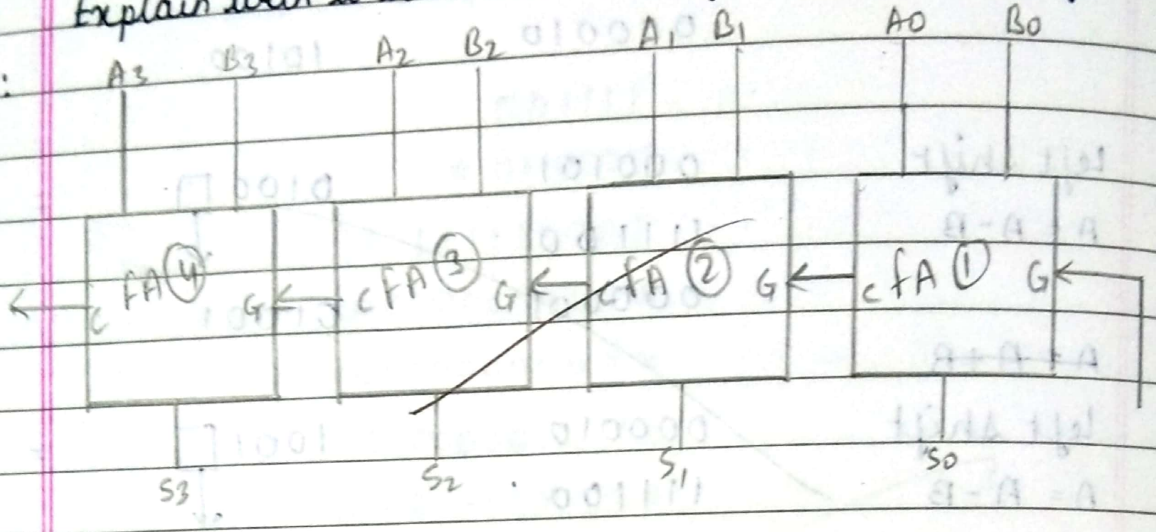


$C_{i+1} = P_i C_i + G_i$

$= A_i \oplus B_i C_i + A_i \cdot B_i$

Q:5 Explain with a neat block diagram ripple carry adder.

Ans:



4-Bit Ripple Carry Adder

Multiple full adders can be cascaded in parallel to add an n-bit number. For an n-bit parallel adder, there must be n numbers of full adder circuit. A Ripple carry adder is a logic circuit in which the carry out of each full adder is the carry in of the succeeding next most significant full adder. It is called a riple carry adder because each carry bit gets rippled into the next stage. In a ripple carry adder the sum and carry out bits of any half adder stage is not valid until the carry is of that stage occurs.

Q:6 Explain IEEE standards for floating point number:

Ans: The IEEE standard for floating point number point arithmetic (IEEE-754) is a technical standard for floating arithmetic established in 1985 by the institute of electrical and electronic engineers.

the standard addressed many problems found in the diverse floating point implementation that made many hardware difficult to use reliably and portably. Many hardware floating point units use the IEEE 754 standards.

- FORMULA for floating point numbers in IEEE 754 is :

⇒ $(-1)^s \times (1.M) \times 2^{e-bias}$

ASSIGNMENT-02

DATE : / /
PAGE NO. : 08

Q:1 Consider a fully associative mapped cache of size 16 KB with block size 256 bytes. the size of main memory is 128 KB. Find :

I) Number of bits in tags.

II) Tag Bits

KB = 2^{10}

Ans:
- Cache size = 16 KB = 2^{14}
- Block Size = 256 bytes = 2^{8}
- Main Memory Size = 128 KB = 2^{17}

- TAG BITS = BLOCK INDEX

- No. of Blocks = $\frac{\text{Total Main Memory}}{\text{Single Block Size}} = \frac{2^{17}}{2^{8}} = 2^{9}$
- Block Index = 9 Bits

* TAG BITS = 9 Bits

Q:2 Consider a fully associative mapped cache of size is 512 KB with Block size 1 KB. There are 17 bits in the tag. Find :

I) Size of main Memory.

Ans:
- Cache Memory Size = 512 KB = 2^{19}
- Block size = 1 KB = 2^{10}
- Tag Bits = 17 Bits

- TAG BITS = Block Index.

- BLOCK INDEX = 17 Bits
- No. of Blocks = 2^{17}

* NO. of blocks = Total Main Memory / Single Block size

⇒ $2^{17} = \frac{Q}{2^{10}}$

⇒ $2^{27} = Q$

* ⇒ Main Memory Size = 2^{27}

Q:3 Consider a fully associative mapped cache with line size of 4 KB. The size of Main Memory is 16 GB. Find:

1) Tag Bits

Ans:
- Block Size = 4 KB = 2^{12} = Line Size
- Main Memory size = 2^{34}
- NO. of Blocks = Main Memory Total size / Single Block Size

⇒ $\frac{2^{34}}{2^{12}} = 2^{22}$

- Block Index = 22 Bits

TAG BITS = BLOCK INDEX

* Tag Bits = 22 Bits

Q:4 A 64 KB cache has 16 bytes blocks. If addresses are 32 bits how many bits have used the tag, index and offset in the cache?

I) For if cache is direct Mapped

II) if cache is 4-way set associative

III) for fully associative cache

Ans:
- S Block size = Single Line size
  ⇒ S Block Size = 16 = 2^4
- Cache Memory Size
  ⇒ 64 KB = 2^{16}
- Physical Address of Main Memory = 32 Bits
- Main Memory Size = 2^{32}

I) DIRECT MAPPING :

- No. of Blocks = $\frac{\text{Total Main Memory}}{\text{single Block size}}$

$$= \frac{2^{32}}{2^4} = 2^{28}$$

- No. of lines = $\frac{\text{Total Cache Memory}}{\text{single line size}}$

$$= \frac{2^{16}}{2^4} = 2^{12}$$

- Block Index = 28 Bits
- Cache Index = 12 Bits

* Block Offset = 4 Bits *

* TAG BITS = Block Index - Cache Index

= 28 - 12

⇒ 16 Bits *

II) SET ASSOCIATIVE MAPPING :

- K = 4
- No. of sets = $\frac{\text{No. of lines}}{K} = \frac{2^{12}}{2^2} = 2^{10}$
- Set Index = 10 Bits
- Block Index = 28 Bits

* TAG BITS = Block Index - Set Index

= 28 - 10

= 18 Bits *

III) ASSOCIATIVE MAPPING :

- TAG BITS = Block Index

∴ Tag Bits = 28 Bits *

DATE : / /
PAGE NO. : 11

Q:5 Consider a 2-way set associative mapped cache of size 16 KB with line size of 256 bytes. The size of main memory is 128 KB. Find :

I) Cache Index

II) Tag Bits

Ans:
- K = 2
- Block Size = Line Size
- Total Cache Memory = 16 KB = 2^{14}
- Single Line Size = 256 bytes = 2^8
- Total Main Memory = 128 KB = 2^{17}
- NO. of blocks = $\frac{2^{17}}{2^8} = 2^9$

- NO. of lines = $\frac{\text{Total Cache Memory}}{\text{Single Line Size}} = \frac{2^{14}}{2^8} = 2^6$

* Cache INDEX : 6 bits
- Block Index = 9 bits

- NO. of sets = $\frac{\text{NO. of lines}}{k} = \frac{2^6}{2^1} = 2^5$
- Set Index = 5 bits

* TAG BITS = Block Index - Set Index
= 9 - 5 = 4 bits *

Q:6 Consider a 8-way set associative mapped cache of size of 512 KB with block size 1KB. There are 7 bits in the tag. Find:

I) Size of Main Memory

Ans:
- K = 8
- Total Cache Size = 512 KB = 2^{19}
- Single Block Size = single line size = 1KB = 2^{10}
- Tag Bits = 7 bits

- NO. of lines = $\frac{\text{Total Cache Memory}}{\text{Single Line Size}} = \frac{2^{19}}{2^{10}} = 2^9$

• No. of sets = No. of lines / K ⇒ $2^9 / 2^3$ ⇒ 2^6

• Set Index = 6 bits

• Tag Bits = Block Index - Set Index
7 = BI - 6

• Block Index = Tag Bits + Set Index
BI = 7 + 6
Block Index = 13

• No. of Blocks : 2^{13}

* Main Memory Total Size = No. of Blocks × Single Block Size
$= 2^{13} \times 2^{10}$
$= 2^{23}$ *

Q:7 Consider a 4-way set associative mapped cache with block size 4 KB. The size of main memory is 16 GB and there are 10 bits in the tag. Find:
1) Size of Cache Memory.

Ans:
• K = 4
• Block Size = Single Line size = 2^{12}
• Total Main Memory = 2^{34}
• Tag Bits = 10 bits
• No. of Blocks = $2^{34} / 2^{12} = 2^{22}$
• No. of sets = No. of lines / K
• Block Index = 22 bits
• Tag Bits = Block Index - Set Index
10 = BI - SI ⇒ 10 = 22 - SI
= Set Index = 12 bits
• No. of sets = 2^{12}
• No. of sets = No. of lines / K ⇒ $2^{12} = \text{No. of lines} / 2^2$ = No. of lines = 2^{14}

* Total Cache Memory = No. of lines × Single Line size
$= 2^{14} \times 2^{12}$
$= \underline{\underline{2^{26}}}$ *

Q:8 A block-set associative cache memory consists of 128 blocks divided into four block sets. The main memory consists of 16,384 blocks and each block contains 256 eight bit words.
i) How many bits are required for addressing the main memory?
ii) How many bits are needed to represent the TAG, SET, and WORD fields?

Ans: • Main Memory = 16384 bytes × 2^8 bytes
$= 2^{14} \times 2^8$
$= \underline{\underline{2^{22}}}$ bytes

→ * • Physical Address = $\underline{\underline{22}}$ bits

• NO. of lines = $\underline{\underline{2^7}}$

• NO. of sets = 4 = $\underline{\underline{2^2}}$ *

• Set Index = $\underline{\underline{2}}$ bits

• Block Index = $\underline{\underline{8}}$ bits

→ * • TAG BIT = Block Index - Set Index
$= 8 - 2 \Rightarrow \underline{\underline{6 \text{ bits}}}$ *

Q:9 Consider a 8-way set associative mapped cache. The size of cache memory is 512 KB and there are 10 bits in the Tag. Find the size of Main Memory.

Ans:
- Total Cache Memory = 512 KB = 2^{19}
- Cache Add = 19
- Tag Bits = Physical Address - Cache Address.
- 10 = Physical Address - 19
  - ⇒ Physical Address = 10 + 19
  - = 29

* • Main Memory Total size
  = 2^{29} *

Q:10 Consider a 4-way set associative mapped cache. The size of Main Memory is 64 MB and there are 10 bits in the tag. Find the size of cache Memory.

Ans:
- K = 4
- Total Main Memory : 64 MB ⇒ 2^{26}
- Physical Address = 26 bits
- TAG BITS ⇒ Physical Address - Cache Address
  - 10 ⇒ Physical Address - Cache Address
  - 10 ⇒ 26 - Cache Address

Cache Address ⇒ 26 - 10
⇒ 16

* Cache Memory Total ⇒ 2^{16} *

ASSIGNMENT - 03

DATE : / /
PAGE NO. : 15

Q:1 What do you mean by signals, logic operators and gates?

Ans: **Signals :**

In electronics, a signal is an electrical or electromagnetic current this is used for carrying data from one device or network to another

Logic operators :

A logical operator is a special symbol or word which connects two or more phases and phrases of information. In computing, logical operations are necessary because they can be used to model the way that information flows through electrical circuits.

Logic Gates :

A logic gate is an idealized or physical device implementing a Boolean function; that is, it performs a logical operation on one or more binary inputs and produces a single binary output.

Q:2 Simplify the following BOOLEAN EXPRESSIONS :

i) $F(A,B,C) = \bar{A}B + B\bar{C} + BC + A\bar{B}\bar{C}$

$= B(\bar{A} + \bar{C} + C) + A\bar{B}\bar{C}$

$= B(\bar{A} + 1) + A\bar{B}\bar{C}$

$= B(1) + A\bar{B}\bar{C}$

$= B + A\bar{B}\bar{C}$

$* = B + A\bar{C}$

ii) $F(A,B,C) = (A+B)(B+C)$

$= AB + AC + B + BC$

$= B(A + 1 + C) + AC$

$= B(1) + AC$

$* = B + AC$

iii) $F(A,B,C) = \bar{A}BC + \bar{A}B\bar{C} + A\bar{B}\bar{C} + A\bar{B}C$

$= \bar{A}B(C+\bar{C}) + A\bar{B}(\bar{C}+C)$

$= \bar{A}B(1) + A\bar{B}(1)$

$* = \bar{A}B + A\bar{B}$

iv) $AB + A(B+C) + B(B+C)$

$= AB + AB + AC + B + BC$

$= AB + AC + B + BC$

$= B(A+1+C) + AC$

$= B(1) + AC$

$* = B + AC$

Q:3 What do you mean by COMBINATIONAL LOGIC CIRCUITS?

Ans: In digital circuit theory, combinational logic circuits (sometimes also referred as time independent logic circuits) are type of digital logic circuits which are implemented by Boolean circuits, where the output is a pure function of the present inputs only. Combinational circuits have no memory storage.

Q:4 What do you mean by SEQUENTIAL CIRCUITS?

Ans: In digital circuit theory, sequential logic circuits is a type of logic circuit whose output not only depends on the present values of its inputs but also depends on the previous values of its outputs.
Sequential Circuit make use of clock and also contains Memory storage.

Q:5 How to convert one flip flop to other flip flop ? Give example.

Ans: **JK FLIPFLOP to SR FLIPFLOP :**

Available Flipflop = JK Flipflop

Required Flipflop = SR Flipflop

* **REQUIREMENTS :**

- Characteristic Table of SR Flipflop and
- Excitation Table of JK Flipflop.

* **Characteristic Table of SR :**

Q	S	R	Q_n
0	0	0	0
1	0	0	1
0	0	1	0
1	0	1	0
0	1	0	1
1	1	0	1
0	1	1	*
1	1	1	*

Excitation Table of JK :

Q	Q_n	J	K
0	0	0	*
0	1	1	*
1	0	*	1
1	1	*	0

* **EXCITATION TABLE**

from characteristic table of SR ←

Q_n	Q	J	K	Outputs
0	0	0	*	M_0
1	1	*	0	M_1
0	0	0	*	M_2
0	1	*	1	M_3
1	0	1	*	M_4
1	1	*	0	M_5
*	0	*	*	M_6
*	1	*	*	M_7

* K-MAPPING :

■ K-Map for J :

Q \ SR	00	01	11	10
0		*	*	
	0	1	3	2
1	1	*	*	*
	4	5	7	6

- $Q_n \Rightarrow Q\bar{R}$

■ K-Map for K :

Q \ SR	00	01	11	10
0	*		1	*
	0	1	3	2
1	*		*	*
	4	5	6	7

$\Rightarrow S$

ASSIGNMENT - 04

DATE : / /
PAGE NO. : 19

Q:1 What do you mean by Computer Architecture ?

Ans: In Computer Engineering, Computer Architecture is a set of rules and methods that describe the functionally, organisation and implementation of computer systems.
It is a specification detailing how a set of software and hardware technology standard interact to form a computer system or platform. In short, it refers to how a computer system is designed an what technologies it is compatible with.

Q:2 Differentiate between Application and System softwares :

Ans:

APPLICATION SOFTWARES :

- Application software are used by users to perform some specific tasks.
- Application softwares are installed according to user's requirements.
- In general, the user interacts with application softwares.
- Application software can't run independently. They can't run without the presence of system software.
- some examples of Application softwares are: Word Processor, Web Browser, Media Player etc.

SYSTEM SOFTWARES :

- System softwares are used for operating computer hardware.
- System softwares are installed on the computer when operating system is used and installed.
- In general, the user does not interact with system softwar

because it works in the background.

- System softwares can run independently. It provides a platform for running application softwares.
- Some example of system softwares are : Compiler, Assembler, debugger, driver etc.

Q:3 If a computer is executing 50 instructions, and one instruction is taking 3 clock cycles, and one cycle takes 20 seconds. what is the total CPU time taken by the computer?

Ans: FORMULA FOR TOTAL CPU TIME Calculation :

NO. of instructions to be executed × Clock Cycles × seconds (time)

- NO. of instructions = 50
- Clock Cycles = 3
- Time (seconds) = 20

⇒ Putting the values in the formula :

⇒ 50 × 3 × 20

⇒ 3000 seconds *

Q:4 Explain AMDAHL's LAW with Example :

Ans: In Computer Architecture, AMDAHL's LAW is a formula which gives the theoritical speedup in latency of the execution of a task at fixed workload that can be expected of a system whose resources are improved.

EXAMPLE :

Q: suppose that a task makes extensive use of fixed point operations with 40% of the time consumed by fixed point operation with a new hardware design. If the floating

point module is speed up by factor 4 then what is the overall speedup.

FORMULA : $\frac{1}{(1-P) + \frac{P}{n}}$ → Parallel Fraction
↓ → Factor.
sequential fraction

⇒ P = 40% ⇒ 40/100 ⇒ 0.4
N = 4

$$* \quad \frac{1}{(1-0.4) + \frac{0.4}{4}} = \frac{1}{0.6 + 0.1} = \frac{1}{0.7} = 1.428$$

• total speedup is 1.428 times.

0:5 A processor spends 30% of its time on flp addition, 25% on flp mult and 10% on flp division. Evaluate the following enhancements, each costing the same to implement:
a) Redesign of the flp adder to make it twice as fast.
b) Redesign of the flp mult to make it three times as fast.
c) Redesign the flp divider to make it 10 times fast.

Ans: FORMULA : $\frac{1}{(1-P) + \frac{P}{n}}$

P = for flp adder ⇒ 30% = 30/100 = 0.3
for flp mult = 25% = 25/100 = 0.25
for flp division = 10% = 10/100 = 0.1

N = for flp adder = 2
for flp mult = 3
for flp division = 10

① * for flp adder = $\frac{1}{(1-P) + \frac{P}{n}}$

$\Rightarrow \frac{1}{(1-0.3) + \frac{0.3}{2}} = \frac{1}{0.7 + 0.15} = \frac{1}{0.85} = 1.17$

- total speedup = 1.17 times

② * for flp mult = $\frac{1}{(1-P) + \frac{P}{n}}$

$= \frac{1}{(1-0.25) + \frac{0.25}{3}} = \frac{1}{0.75 + 0.083} \Rightarrow \frac{1}{0.83} = 1.20$

- total speedup = 1.20 times

③ * for flp division = $\frac{1}{(1-P) + \frac{P}{n}}$

$= \frac{1}{(1-0.1) + \frac{0.1}{10}} = \frac{1}{0.9 + 0.01} = \frac{1}{0.91} = 1.09$

- total speedup = 1.09 times