**Solutions of Problems based on Instruction Format**

An instruction format defines the different component of an instruction. The main components of an instruction are opcode (which instruction to be executed) and operands (data on which instruction to be executed). Here are the different terms related to instruction format:

* **Instruction set size –** It tells the total number of instructions defined in the processor.
* **Opcode size –** It is the number of bits occupied by the opcode which is calculated by taking log of instruction set size.
* **Operand size –** It is the number of bits occupied by the operand.
* **Instruction size –** It is calculated as sum of bits occupied by opcode and operands.

**Que-1.** Consider a processor with 64 registers and an instruction set of size twelve. Each instruction has five distinct fields, namely, opcode, two source register identifiers, one destination register identifier, and a twelve-bit immediate value. Each instruction must be stored in memory in a byte-aligned fashion. If a program has 100 instructions, the amount of memory (in bytes) consumed by the program text is \_\_\_\_\_\_\_\_\_\_\_\_.   
(A) 100   
(B) 200   
(C) 400   
(D) 500

Answer: (D)

Explanation: One instruction is divided into five parts,

* (1): The opcode- As we have instruction set of size 12, an instruction opcode can be identified by 4 bits, as 2^4=16 and we cannot go any less.
* (2) & (3): Two source register identifiers- As there are total 64 registers, they can be identified by 6 bits. As they are two i.e. 6 bit + 6 bit.
* (4): One destination register identifier- Again it will be 6 bits.
* (5): A twelve-bit immediate value- 12 bit.

Adding them all we get,

= 4 + 6 + 6 + 6 + 12

= 34 bit

= 34/8 byte

= 4.25 byte

As given Each instruction must be stored in memory in a byte-aligned fashion,4.25 is not byte alignment, memory address should be 0,1,2,3,4,5,6,7……... so it should be 5 bytes.

As there are 100 instructions, we have a size of 5\*100= 500 bytes.

**Que-2.** A processor has 40 distinct instructions and 24 general purpose registers. A 32-bit instruction word has an opcode, two registers operands and an immediate operand. The number of bits available for the immediate operand field is\_\_\_\_\_\_\_.

Answer: (A)  
  
Explanation: 6 bits are needed for 40 distinct instructions (because, 32 < 40 < 64 ) 5 bits are needed for 24 general purpose registers( because, 16< 24 < 32) 32-bit instruction word has an opcode(6 bit), two register operands(total 10 bits) and an immediate operand (x bits). The number of bits available for the immediate operand field => x = 32 – (6 + 10) = 16 bits.

**Que-3.** A machine has a 32-bit architecture, with 1-word long instructions. It has 64 registers, each of which is 32 bits long. It needs to support 45 instructions, which have an immediate operand in addition to two register operands. Assuming that the immediate operand is an unsigned integer, the maximum value of the immediate operand is \_\_\_\_\_\_\_\_\_\_\_.

**Answer:** **16383**  
  
Explanation:

1 Word = 32 bits

Each instruction has 32 bits. To support 45 instructions, opcode must contain 6-bits

Register operand1 requires 6 bits, since the total registers are 64. Register operand 2 also requires 6 bits. So total 18 bits for all 45 instruction and two register operands. So

32-18 = 14

14-bits are left over for immediate Operand Using 14-bits, now

2^14 - 1 = 16383

We can give maximum 16383.

**Que-4.** A processor has 16 integer registers (R0, R1, …, R15) and 64 floating point registers (F0, F1, …, F63). It uses a 2-byte instruction format. There are four categories of instructions: Type-1, Type-2, Type-3, and Type 4. Type-1 category consists of four instructions, each with 3 integer register operands (3Rs). Type-2 category consists of eight instructions, each with 2 floating point register operands (2Fs). Type-3 category consists of fourteen instructions, each with one integer register operand and one floating point register operand (1R+1F). Type-4 category consists of N instructions, each with a floating-point register operand (1F).

The maximum value of N is \_\_\_\_\_\_\_\_

**Answer:** **32**  
  
**Explanation:** Given, size of instruction format is 2 byte (= 16 bits), therefore number of instructions encoding = 216  
Also, total number of bits in integer operand = log2(16 integer registers) = 4  
Total number of bits in floating point operand = log2(64 floating point registers) = 6

So, number of encodings consumed:

By type 1 instructions = 4×23×4 = 214  
By type 2 instructions = 8×22×6 = 215  
By type 3 instructions = 14×2(4+6) = 14336

Now, number of encoding left for type 4 instructions = 216 − (214 + 215 + 14336) = 2048  
Therefore, total number of different instructions of type 4 instructions = 2048 /64 = 32

Please note that there is difference between number of different instructions and number of different encodings, a single instruction can have different encodings when the address part differs.

So, answer is 32.

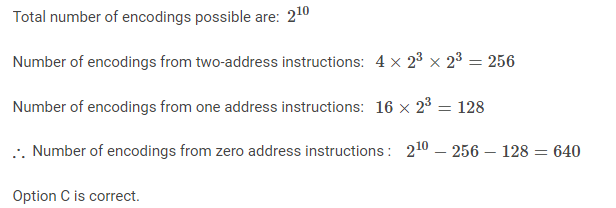
Que-5 Top of Form

Bottom of Form

Top of Form

Que-5 In a 10-bit computer instruction format, the size of address field is 33-bits. The computer uses expanding OP code technique and has 44 two-address instructions and 16 one-address instructions. The number of zero address instructions it can support is

1. 256
2. 356
3. 640
4. 756



Que-6 Match list I with List II and select the correct answer using the codes given below the lists.Table

Description automatically generated

Ans: (d)

Que-7 A stack organized computer has which of the following instructions?

1. zero-address
2. one-address
3. two-address
4. three-address

Ans: (a)

Que-8 For computer based on three-address instruction formats, each address field can be used to specify which of the following:  
(S1) A memory operand  
(S2) A processor register  
(S3) An implied accumulator register

1. Either S1 or S2
2. Either S2 or S3
3. Only S2 and S3
4. All of S1, S2 and S3

Ans: (b)

Que-9. The instruction ADD 3030 is of

* 1. 3-address instruction format
  2. 2-address instruction format
  3. 1-address instruction format
  4. 0-address instruction form

Ans: (c)

Q10. For a 0-address instruction format, what would be the top element of the stack following sequences of instructions?

PUSH 20;

PUSH 5;

PUSH 5;

ADD;

SUB;

PUSH 20;

MUL.

1. 100
2. 200
3. 10
4. 5

Ans: (b)