IIT Jodhpur, II Mid Sem. Examination
Computer Science & Engineering V Sem. 2014
30002:Computer Organization

Max. Marks=30 Time= 1 Hr.

1. Consider a processor in which each instruction is 16-bit size. The following contents appears in main memory, starting at location 200. Each memory location holds one byte.

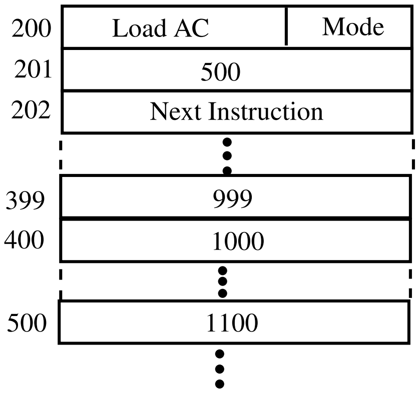

Figure 1: Three locations of main memory.

The first part of the first instruction indicates that this instruction loads a value into an accumulator. The *mode* field specifies addressing mode and, if appropriate a *source register* (depending on the addressing mode used). Assume that when used, the source register is R_1 having a value inside of 400. There is also a *base register* that contains the value 100. The value 500 in location 201 may be part of the address calculation for some instructions. Determine: (1) effective address and (2) value of operand to be loaded into accumulator for the "*Load AC, ...*" instruction when following addressing modes are used: (6)

(a) Direct
(b) Indirect
(c) Immediate
(d) Displacement
(e) Register
(f) Register Indirect

Ans. The current instruction is at location 200-201. The value "500" which is at location 201, can be operand itself, then it is Immediate. If this is direct, then address of operand is 500, and value as 1100. If it is indirect, then value at address 500, i.e., 1100 is address and value of operand is 1700. This can be visualized from the . . . showing continuation of memory. When base register is added into the start address of the program, it is called relative address. But this has net been asked. This is used for relocation of programs. The Displacement is obtained by adding one of the register and the constant given along with the instruction, which is 500. So displacement address is 400+500= 900. The corresponding operand is 1500. So the answers are:

(a) effective address 500, value 1100.
(b) 1100, 1700
(c) 201, 500
(d) 900, 1500
(e) R1, 400
(f) 400, 1000

2. There is a variable-length opcode to allow all of the following to be encoded in a 32-bit instruction, with each memory location of one byte size, and total eight general purpose CPU registers. Suggest a suitable instruction set architecture, addressing modes, and types of instructions: arithmetics, logical, and control instructions, like conditional and unconditional jumps and subroutine calls. (6)

   (a) Instructions category 1, have three operands: two 13-bit addresses and one 3-bit register number.

   (b) instructions category 2, have two operands: one 13-bit address and one 3-bit register number

   (c) Instructions category 3, with no address or registers (i.e., implied address)

Ans. (a) The memory unit is byte, there are 8 GPRs. In this instruction format, there is one register, which can be used as index / base / displacement. One of the two 13-bit values can be added into register to obtain the displacement address. The other 13-bit can be used as source/destination address. The register can be used as counter to move data from source to destination. Such instructions are used for block move between two memory locations. But, not for arithmetics and logic operations.

   (b) The type 2, with 13-bit address and register, can be for all arithmetics and logical operations, with two operands, the memory is source and register is destination. Alternatively the register can be used as index / base / displacement register. The 13-bit value can be immediate operand.

   (c) The last type with no operand is for implied addressing, where accumulator is source and destination. The instructions can be complement, increment, decrement, rotate left, right, etc.

3. Write a PDP-8 Assembly language program to sum an array of 16 elements, and store the result at location 0357_8. Each element is 12-bit length and occupies single memory location. The "program" and "data" of array are stored in page number one of the memory as shown in figure 2. (5)

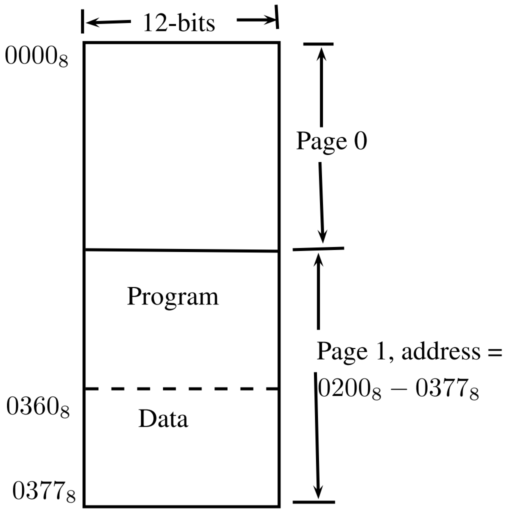

Figure 2: Memory with program and data.

The size of instructions is fixed 12-bits ($D_0 - D_{11}$), with bit D_0 as MSB and bit D_{11} as LSB, the bits are used as follows:

bits $D_0 - D_2$ are 3-bit opcode,

bit $D_3 = 0$ means direct address, and $D_3 = 1$ means indirect address.

bit $D_4 = 0$ means zero page, and $D_4 = 1$ means current page (in which instruction is)

bits $D_5 - D_{11}$ is 7-bit offset for data address (other bits are taken as zeros to form the address)

(a) What should be the values of bits D_3, D_4 to access the data?

(b) Should the program use direct or indirect addressing mode?

(c) Where would you maintain the "data pointer" and "counter"?

(d) Should we up-count or down-count? Justify.

(e) Explain the logic of steps of above assembly language program.

.

address / code in octal/in octal	instruction	Comment
0200 / 7200	CLA	clear accumulator
0201/1755	TAD	twos complement add using indirect address at 0355
0202/3357	DCA	deposit and clear accumulator i.e. update sum at 0357
0203/1355	TAD	move data pointer to accumulator
0204/1354	TAD	add 1 to pointer to point to next data
0205/3355	DCA	update data pointer at 0355
0206/2356	ISZ	increment and skip next instruction on counter zero counter is maintained at 0356
0207/5200	JMP	jump to begin (0200)
0208/7402	HLT	else Halt the program
different	data	follows:
0354/1		1 is added to -ve value of counter every time till it becomes zero
0355/0360		data pointer address is here
0356/-16		-ve value of counter is initial counter
0357/0000		initial sum is this

(1, 1, 2, 1, 3)

Ans. The program in PDP8 assembly language is as shown below. The instructions have octal code as: 1 = TAD, 2 = ISZ, 3 = DCA, 5 = JMP, CLA = 7200.

The remaining answers are:

(a) $D_3 = 1, D_4 = 1$ to access the data?

(b) Indirect addressing

(c) The "data pointer" and "counter" are in memory at 0355, 0356.

(d) Up-count due to ISZ.

(e) There is indirect address pointer, which is increased by 1 every time in loop to point to next data location. For this 1 is added from location 0354.

4. By analyzing the program below, write the expression that is computed using the stack architecture. Assume A, B, C, D, E, F, G, H are the register names to hold values a, b, c, d, e, f, g, h, respectively. (5)

```
PUSH B
PUSH C
PUSH D
MUL
PUSH E
MUL
PUSH F
PUSH G
MUL
PUSH H
ADD
SUB
ADD
POP A
```

Ans. The progressive values in stack (from bottom) are:

b c d
b c*d e
b c*d*e f g
b c*d*e f*g h
b c*d*e (f*g)+h
b c*d*e-(f*g+h)
a=b+cde-(fg+h)