

CPSC 141 – Homework #7

Due: Friday, 27 Nov 2020

William Friend

All of the below problems are from the Chapter 4 exercises in the Text, starting on page 289. Be extra sure to use the correct edition of the Null/Lubor text (5th edition). If you do not have the correct edition, please see me or photocopy the pages from one of your classmates.

Be sure to use the MARIE simulator software for the last two questions.

Problem	Percentage	Instructions
11	15%	
14	20%	
16	10%	Hint: Look at #15 (a, b)
18	15%	To get credit here, you must provide a credible explanation for your answer
24	10%	Use the Marie Simulator. For parts a & b, you may include the assembler list file.
33	30%	Submit the MAS file. Be sure to test the MAS file with the Marie simulator.

11. Redo exercise 10 assuming a $16M \times 16$ memory built using $512K \times 8$ RAM chips.

a) How many RAM chips are necessary?

64(32 rows, 2 columns)

b) If we were accessing one full word, how many chips would be involved?

2

c) How many address bits are needed for each RAM chip?

$512k = 2^9(512) \times 2^{10}(1024) = 2^{19}$, or 19 address bits.

d) How many banks will this memory have?

32

e) How many address bits are needed for all memory?

$16M = 2^4(16) \times 2^{20}(1048576) = 2^{24}$, or 24 address bits

f) If high-order interleaving is used, where would address 14 (which is E in hex) be located?

Bank 0 (000)

g) Repeat exercise 9f for low-order interleaving.

Bank 14 (110)

CPSC 141 – Homework #7

Due: Friday, 27 Nov 2020

14. A digital computer has a memory unit with 32 bits per word. The instruction set consists of 110 different operations. All instructions have an operation code part (opcode) and two address fields: one for a memory address and one for a register address. This particular system includes eight general-purpose, user-addressable registers. Registers may be loaded directly from memory, and memory may be updated directly from the registers. Direct memory-to-memory data movement operations are not supported. Each instruction is stored in one word of memory.

a) How many bits are needed for the opcode?

$$7, 2^7=128 \ 110 < 128$$

b) How many bits are needed to specify the register?

$$3, 2^3=8$$

c) How many bits are left for the memory address part of the instruction?

$$22 \text{ bits, } 32 - 7 - 3 = 22$$

d) What is the maximum allowable size for memory?

$$2^{22}$$

e) What is the largest unsigned binary number that can be accommodated in one word of memory?

$$2^{22}-1$$

16. Suppose the RAM for a certain computer has 256M words, where each word is 16 bits long

a) What is the capacity of this memory expressed in bytes?

The capacity of the memory of 256M words expressed in bytes is shown below:

In this case, since the memory is byte-addressable, which means that a word is 8 bit long. The number of 8-bit words this memory holds is calculated as

$$= 256 \text{M} \times 16 \text{bits}$$

$$= 256 \times 2^{20} \times 16 \text{bits}$$

$$= 2^8 \times 2^{20} \times 2^4 \text{ bits}$$

$$= 2^{32} \text{bit}$$

$$= 4294967296 \text{ bits}$$

CPSC 141 – Homework #7

Due: Friday, 27 Nov 2020

Divide by 8 to get bytes

= 536870912 bytes

Divide 536870912 bytes by 1024

So, the capacity of memory is 524288 byte items

b) If this RAM is byte addressable, how many bits must an address contain?

The number of bits required to address the memory if the RAM is byte addressable is calculated as follows,

Length of RAM x (width of RAM / byte size in bits)

= 256M x (16/8)

= $2^8 \times 2^{20} \times 2^1$

= 2^{29}

Hence, if the RAM is byte addressable, an address must contain 29 bits.

c) If this RAM is word addressable, how many bits must an address contain?

The number of bits required to address the memory if the RAM is word addressable is calculated as follows:

Length of RAM x (width of RAM/ byte size in bits)

= 256M x (16/16)

= 256M

= 256×2^{20}

= $2^8 \times 2^{20}$

= 2^{28}

Hence, if the RAM is word addressable, an address must contain 28 bits

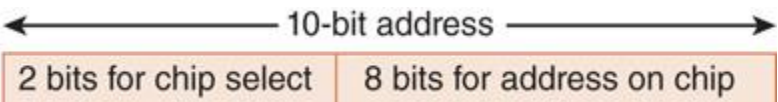

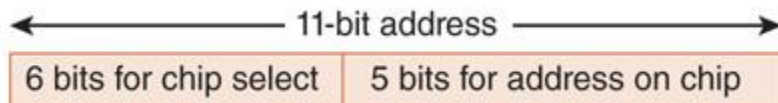
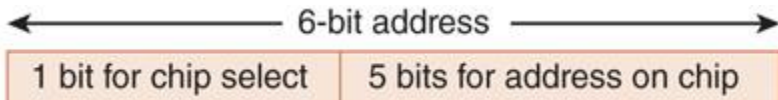
18. Given a memory of 2048 bytes consisting of several 64×8 RAM chips, and assuming byte-addressable memory, which of the following seven diagrams indicates the correct way to use the address bits? Explain your answer.

Answer:

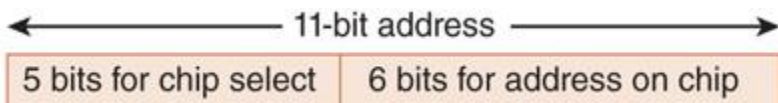
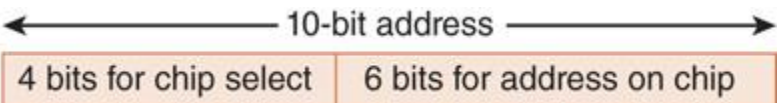
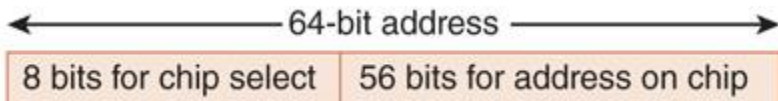
The

CPSC 141 – Homework #7

Due: Friday, 27 Nov 2020

- (a)  10-bit address
2 bits for chip select | 8 bits for address on chip
- (b)  64-bit address
16 bits for chip select | 48 bits for address on chip
- (c)  11-bit address
6 bits for chip select | 5 bits for address on chip
- (d)  6-bit address
1 bit for chip select | 5 bits for address on chip

e.jpg correct, least possible address codes needed, while giving optimal space to allocate opcodes.

- (e)  11-bit address
5 bits for chip select | 6 bits for address on chip
- (f)  10-bit address
4 bits for chip select | 6 bits for address on chip
- (g)  64-bit address
8 bits for chip select | 56 bits for address on chip

Since memory is 2048 bytes built from 64x8 Ram chips, and since the memory is byte addressable.

Since we are given 64 x 8 RAM chips, we have 64 rows of 8 bits per row, or arranged into 64 8 bit bytes.

Since byte addressable memory, the rule that with N address bits we can address 2^N bytes, so in our case with 64 x 8 bit bytes we can either realize $2^6 = 64$, or $\log_2(64) = 6$, either way it tells use well need a 6-bit address, and each address corresponds to it's own unique byte on the RAM chip.

Next,

Since we were given 2048-byte total, we'd need $2048/64 = 32$ RAM chips.

And since 32 also $2^5 = 32$ and $\log_2(32) = 5$ we know we need 5 bits in the chip select

So with five to select the chip, we stilll need the other 6 bit to address the right byte on the right chip.

So we get 11 total bits, 5 chip select, and 6 address, and e is the only one that fits this.

So the answer is e.

CPSC 141 – Homework #7

Due: Friday, 27 Nov 2020

24. Consider the MARIE program below.

Hex Address	Label	Instruction
100	Start,	LOAD A
101		ADD B
102		STORE D
103		CLEAR
104		OUTPUT
105		ADDI D
106		STORE B
107		HALT
108	A,	HEX 00FC
109	B,	DEC 14
10A	C,	HEX 0108
10B	D,	HEX 0000

Friend_William_hw_7/33/24.mex

Friend_William_hw_7/33/24.mas

For me (need the reference):

0	Jns X	Store the PC at address X and jump to X+1
1	Load X	Load contents of address x into AC
2	Store X	Store the contents of AC at address X
3	Add X	Add the contents of address X to AC
4	Subt X	Subtract the contents of address X from AC
5	Input	Input a value from the keyboard into AC
6	Output	Output the value in AC to the display
7	Halt	Terminate program
8	Skipcond	Skip next instruction on condition
9	Jump X	Load the value of X into PC
A	Clear	Put all zeros in AC
B	AddI X	Add indirect: Use the value at X as the actual address of the data operand to add to AC

CPSC 141 – Homework #7

Due: Friday, 27 Nov 2020

C	JumpI X	Use the value at X as the address to jump to
D	LoadI X	Load indirect: Use the value at X as the address of the value to load
E	StoreI X	Store indirect: Use X the value at X as the address of where to store the value.
F		

a) List the hexadecimal code for each instruction

Following is the solution with explanation for above given MARIE program,

a) Given below is the list of hexadecimal code for each instruction,

Answer:

LOAD A 1108
ADD B 3109
STORE D 210B
CLEAR A000
OUTPUT 6000
ADDI D B10B
STORE B 2109
HALT 7000
(A) HEX 00FC 00FC
(B) DEC 14 000E
(C) HEX 0108 0108
(D) HEX 0000 0000

Explanation:

To figure out the hex value, which is just an easier to read version of the binary value, each instruction of the MARIE program is 16 bits long in which the opcode field occupies first 4 bits of the 16 bits of a

CPSC 141 – Homework #7

Due: Friday, 27 Nov 2020

MARIE instruction. LOAD instruction has defined opcode value 1 in hexadecimal, ADD instruction has opcode value 3 in hexadecimal etc.

The 16 bits instruction can be written in hexadecimal code of length 4 because 2^4 is 16 so each of 4 binary bits can be grouped to one hexadecimal value. Hence 16 binary bits of instruction can be grouped into 4 bits of hexadecimal code. The remaining 12 bits ($16-4=12$) depicts the address field, so we need to understand this when determining our hex values.

Counter	Label	Opcode	Operand	Hexadecimal code
100	Start	LOAD	A	1108
101		ADD	B	3109
102		STORE	D	210B
103		CLEAR		A000
104		OUTPUT		6000
105		ADDI	D	B10B
106		STORE	B	2109
107		HALT		7000
108	A	HEX	00FC	00FC
109	B	DEC	14	000E
10A	C	HEX	0108	0108
10B	D	HEX	0000	0000

b) Draw the symbol table.

Symbol	Location
A	108
B	109
C	10A
D	10B
Start	100

The symbol table is made up of symbols used in MARIE program and the address corresponding to where these symbols are stored. In program, symbols, or labels, A, B, C and D have been used and addresses are come the Hex column given in question for each of these symbols.

c) What is the value stored in the AC when the program terminates?

Counter	Label	Opcode	Operand (human	Hexadecimal code	ACC Value
---------	-------	--------	----------------	------------------	-----------

CPSC 141 – Homework #7

Due: Friday, 27 Nov 2020

			readable for me)		
100	Start	LOAD	A, (00FC or 252)	1108	00FC
101		ADD	B, (+ E or 14)	3109	010A
102		STORE	D, (010A or 266)	210B	010A
103		CLEAR	(000)	A000	0000
104		OUTPUT	(000)	6000	0000
105		ADDI	D(use value in d, 010A, as address and put that value into ac, or 0000, not 010A)	B10B	
106		STORE	B(Store 010A in B)	2109	0000
107		HALT		7000	0000
108	A	HEX	00FC	00FC	
109	B	DEC	14	000E	
10A	C	HEX	0108	0108	
10B	D	HEX	0000	0000	

So, the value stored in AC when the program terminates

First, we Load A, giving the value 00FC that was stored in A to the ac

Next, we add B or 14, to value in ac, giving us 010A in ac

Then we Store the ac value, 010A, in D

Then we Clear the ac setting it to 0000

We then output the same ac value as previous, 0000

Then we access the address pointed too in D, so not D itself but we use the value in D as address, or 010A, and take value at 0104 into accumulator or 0000

We then take 0000 and store it in B

And then we Halt which stops the program and we still have 0000 in the ac so that should be the answer.

33. Write the following code segment in MARIE assembly language

CPSC 141 – Homework #7

Due: Friday, 27 Nov 2020

```
X = 1;  
while X < 10 do  
    X = X + 1;  
endwhile;
```

This is basically just a while loop that counts from 0 to 9 and terminates at 10.

I named the files in the zip:

Friend_William_hw_7/33/while.mas

Friend_William_hw_7/33/while.mex