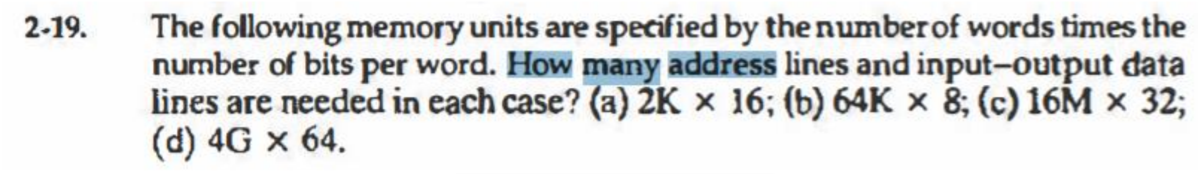
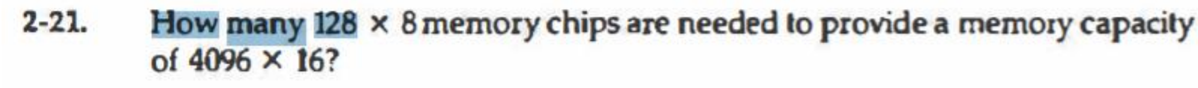


2-19. The following memory units are specified by the number of words times the number of bits per word. How many address lines and input–output data lines are needed in each case? (a) 2K × 16; (b) 64K × 8; (c) 16M × 32; (d) 4G × 64.

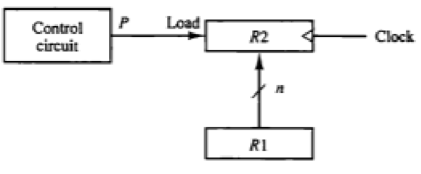
2.19

		Address lines	Data lines
(a)	2K × 16 = $2^{11} \times 16$	11	16
(b)	64K × 8 = $2^{16} \times 16$	16	8
(c)	16M × 32 = $2^{24} \times 32$	24	32
(d)	4G × 64 = $2^{32} \times 64$	32	64



2-21. How many 128 × 8 memory chips are needed to provide a memory capacity of 4096 × 16?

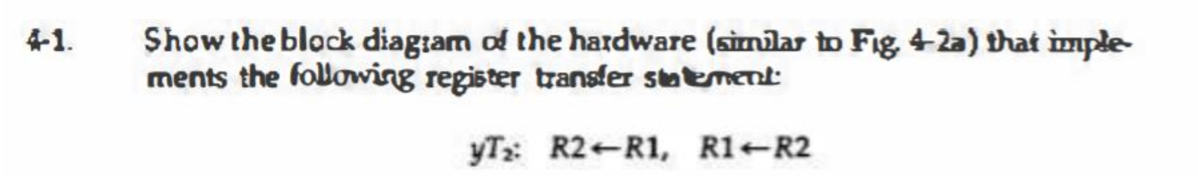
Figure 4-2 Transfer from R1 to R2 when P = 1.



(a) Block diagram

2.21

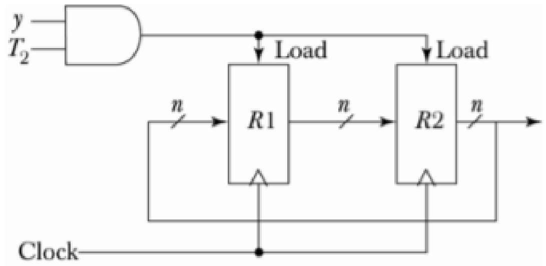
$$\frac{4096\times16}{128\times8}=\frac{2^{12}\times2^{4}}{2^{7}\times2^{3}}=2^{6}=64 \text{ chips}$$



4-1. Show the block diagram of the hardware (similar to Fig. 4-2a) that implements the following register transfer statement:

$$yT_2\text{:}\quad R2 \leftarrow R1,\quad R1 \leftarrow R2$$

4.1



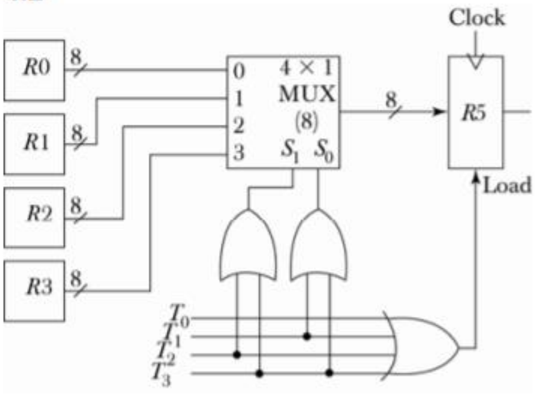
\

4-2. The outputs of four registers, $R0$, $R1$, $R2$, and $R3$, are connected through 4-to-1-line multiplexers to the inputs of a fifth register, $R5$. Each register is eight bits long. The required transfers are dictated by four timing variables T_0 through T_3 as follows:

T_0: $R5 \leftarrow R0$
T_1: $R5 \leftarrow R1$
T_2: $R5 \leftarrow R2$
T_3: $R5 \leftarrow R3$

The timing variables are mutually exclusive, which means that only one variable is equal to 1 at any given time, while the other three are equal to 0. Draw a block diagram showing the hardware implementation of the register transfers. Include the connections necessary from the four timing variables to the selection inputs of the multiplexers and to the load input of register $R5$.

4.2



$T_0T_1T_2T_3$	S_1 S_0 R_3 load
0 0 0 0	X X 0
1 0 0 0	0 0 1
0 1 0 0	0 1 1
0 0 1 0	1 0 1
0 0 0 1	1 1 1

$S_1 = T_2 + T_3$
$S_0 = T_1 + T_3$
$\text{load} = T_0 + T_1 + T_2 + T_3$

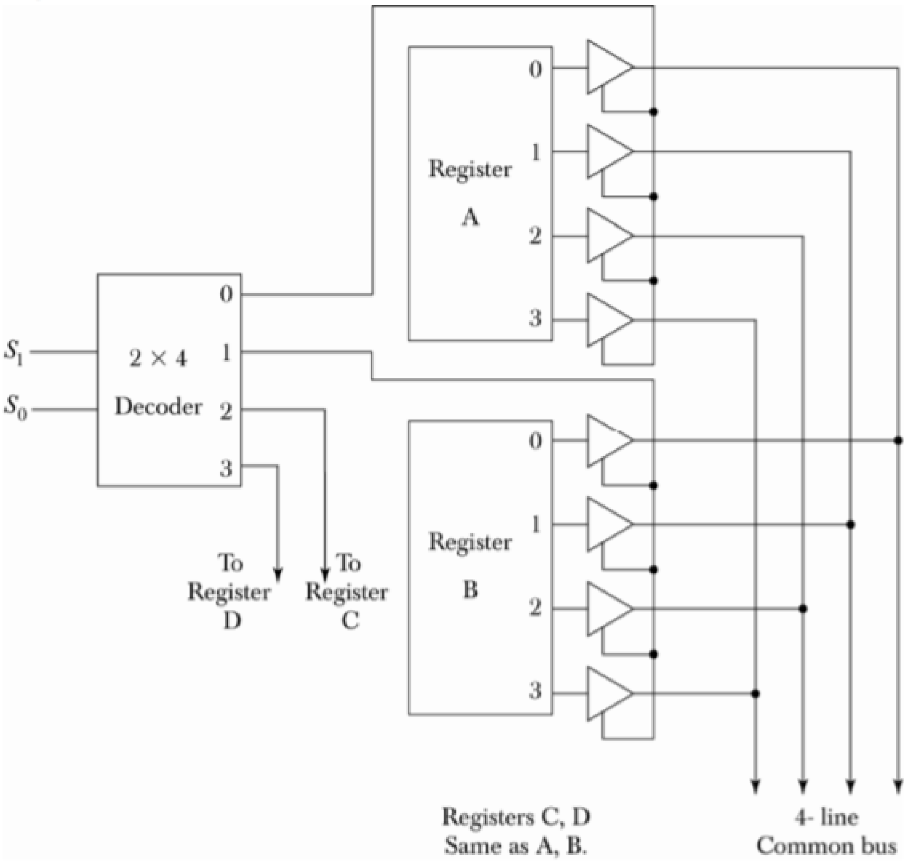
4-3. Represent the following conditional control statement by two register transfer statements with control functions.

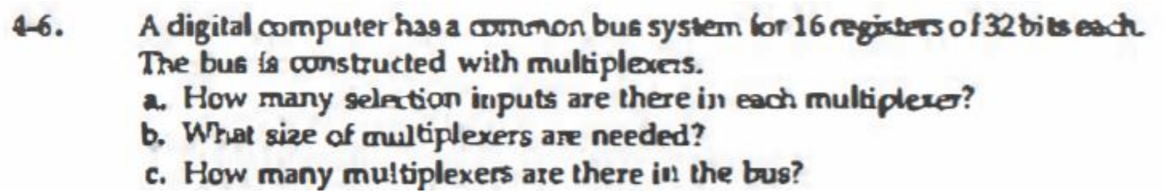
If $(P = 1)$ then $(R1 \leftarrow R2)$ else if $(Q = 1)$ then $(R1 \leftarrow R3)$

4.3
P: R1 ← R2
P'Q: R1 ← R3

4-5. Draw a diagram of a bus system similar to the one shown in Fig. 4-3, but use three-state buffers and a decoder instead of the multiplexers.

4.5





4-6. A digital computer has a common bus system for 16 registers of 32 bits each. The bus is constructed with multiplexers.

a. How many selection inputs are there in each multiplexer?

b. What size of multiplexers are needed?

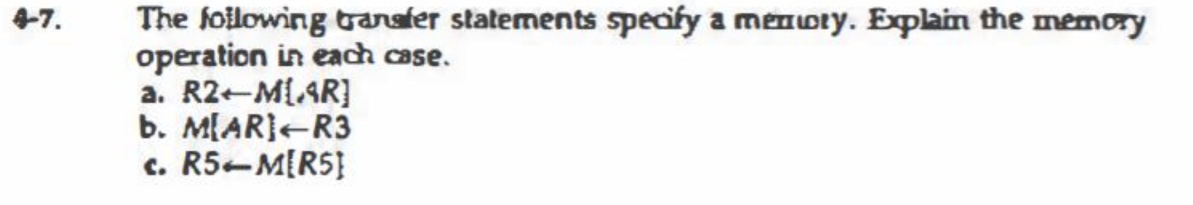
c. How many multiplexers are there in the bus?

4.6

(a) 4 selection lines to select one of 16 registers.

(b) 16 × 1 multiplexers.

(c) 32 multiplexers, one for each bit of the registers.



4-7. The following transfer statements specify a memory. Explain the memory operation in each case.

a. $R2 \leftarrow M[AR]$

b. $M[AR] \leftarrow R3$

c. $R5 \leftarrow M[R5]$

4.7

(a) Read memory word specified by the address in AR into register R2.

(b) Write content of register R3 into the memory word specified by the address in AR.

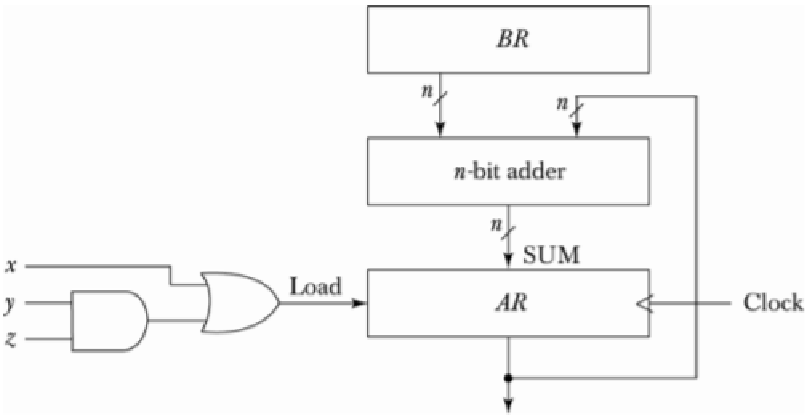
(c) Read memory word specified by the address in R5 and transfer content to R5 (destroys previous value)

4-8. Draw the block diagram for the hardware that implements the following statements:

$$x + yz: \quad AR \leftarrow AR + BR$$

where AR and BR are two n-bit registers and x, y, and z are control variables. Include the logic gates for the control function. (Remember that the symbol + designates an OR operation in a control or Boolean function but that it represents an arithmetic plus in a microoperation.)

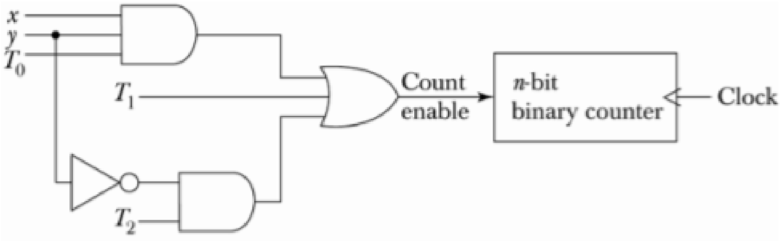
4.8



4-9. Show the hardware that implements the following statement. Include the logic gates for the control function and a block diagram for the binary counter with a count enable input.

$$xyT_0 + T_1 + y'T_2: \quad AR \leftarrow AR + 1$$

4.9

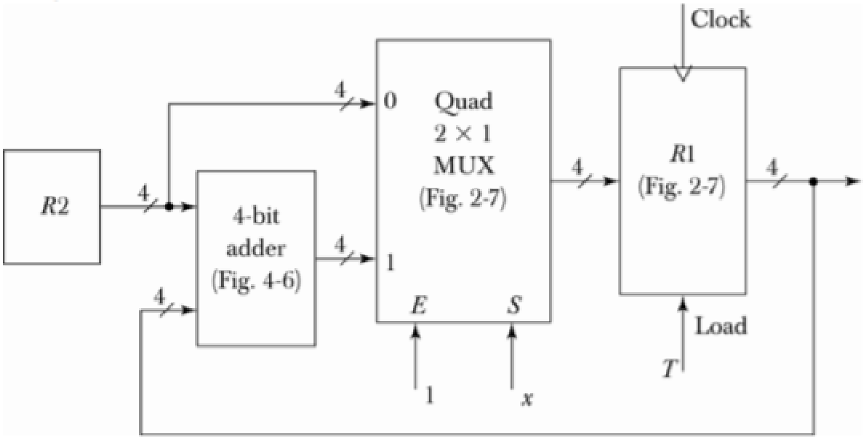


4-10. Consider the following register transfer statements for two 4-bit registers $R1$ and $R2$.

$$\begin{aligned} xT&: \quad R1 \leftarrow R1 + R2 \\ x'T&: \quad R1 \leftarrow R2 \end{aligned}$$

Every time that variable $T = 1$, either the content of $R2$ is added to the content of $R1$ if $x = 1$, or the content of $R2$ is transferred to $R1$ if $x = 0$. Draw a diagram showing the hardware implementation of the two statements. Use block diagrams for the two 4-bit registers, a 4-bit adder, and a quadruple 2-to-1-line multiplexer that selects the inputs to $R1$. In the diagram, show how the control variables x and T select the inputs of the multiplexer and the load input of register $R1$.

4.10



where $R1$ is a counter with parallel load and $R2$ is a 4-bit register.

4-12. The adder-subtractor circuit of Fig. 4-7 has the following values for input mode M and data inputs A and B. In each case, determine the values of the outputs: S_3, S_2, S_1, S_0, and C_4.

	M	A	B
a.	0	0111	0110
b.	0	1000	1001
c.	1	1100	1000
d.	1	0101	1010
e.	1	0000	0001

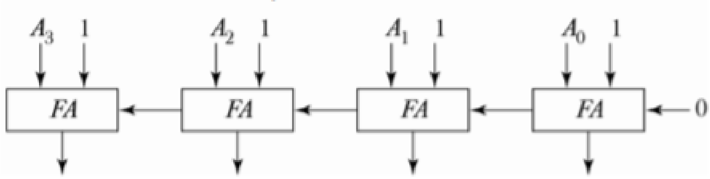
$'y \quad 'x \quad ' = 0$

4.12

M	A		B	Sum	Cu	
0	0111	+	0110	1101	0	7 + 6 = 13
0	1000	+	1001	0001	1	8 + 9 = 16 + 1
1	1100	–	1000	0100	1	12 – 8 = 4
1	0101	–	1010	1011	0	5 – 10 = – 5(in 2's comp.)
1	0000	–	0001	1111	0	0 –1 = –1 (in 2's comp.)

4-13. Design a 4-bit combinational circuit decrementer using four full-adder circuits.

4.13 A – 1 = A + 2's complement of 1 = A + 1111

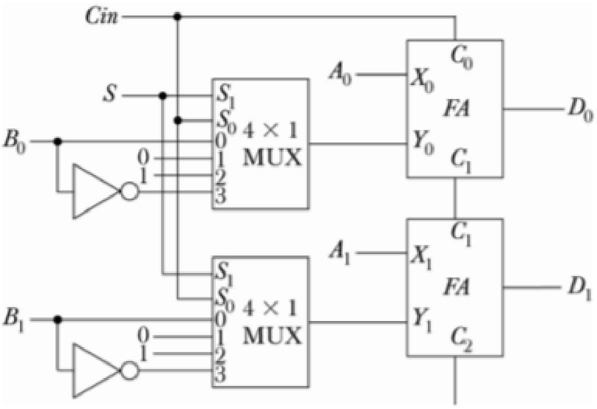


4-15. Design an arithmetic circuit with one selection variable *S* and two *n*-bit data inputs *A* and *B*. The circuit generates the following four arithmetic operations in conjunction with the input carry C_{in}. Draw the logic diagram for the first two stages.

S	$C_{in} = 0$	$C_{in} = 1$
0	$D = A + B$ (add)	$D = A + 1$ (increment)
1	$D = A - 1$ (decrement)	$D = A + \overline{B} + 1$ (subtract)

4.15

S	Cin	X	Y	
0	0	A	B	(A + B)
0	1	A	0	(A + 1)
1	0	A	1	(A –1)
1	1	A	$\overline{B}$	(A – B)



4-18. Register *A* holds the 8-bit binary 11011001. Determine the *B* operand and the logic microoperation to be performed in order to change the value in *A* to:

a. 01101101

b. 11111101

4.18

(a) A = 11011001 $\oplus$
B = 10110100
A ← A⊕B 01101101

A = 11011001
B = 11111101 (OR)
11111101 A ← AVB

4-19. The 8-bit registers *AR*, *BR*, *CR*, and *DR* initially have the following values:

$$\begin{aligned} AR &= 11110010 \\ BR &= 11111111 \\ CR &= 10111001 \\ DR &= 11101010 \end{aligned}$$

Determine the 8-bit values in each register after the execution of the following sequence of microoperations.

$AR \leftarrow AR + BR$	Add *BR* to *AR*
$CR \leftarrow CR \wedge DR$, $BR \leftarrow BR + 1$	AND *DR* to *CR*, increment *BR*
$AR \leftarrow AR - CR$	Subtract *CR* from *AR*

4.19

(a) AR = 11110010
BR = $\underline{11111111(+)}$
AR = 11110001 BR = 11111111 CR = 10111001 DR= 1110
1010

(b) CR = 10111001
$\underline{\text{DR = 11101010}}^{(AND)}$
CR = 10101000
BR = 1111 1111
$\underline{\quad +1}$
BR = 0000 0000 AR = 1111 0001 DR = 11101010

(c) AR = 11110001 $_{(-1)}$
CR = $\underline{10101000}$
AR = 01001001; BR = 00000000; CR = 10101000; DR = 11101010

4-20. An 8-bit register contains the binary value 10011100. What is the register value after an arithmetic shift right? Starting from the initial number 10011100, determine the register value after an arithmetic shift left, and state whether there is an overflow.

4.20

R = 10011100
Arithmetic shift right: 11001110
Arithmetic shift left: 00111000 overflow because a negative number changed to positive.

4-21. Starting from an initial value of *R* = 11011101, determine the sequence of binary values in *R* after a logical shift-left, followed by a circular shift-right, followed by a logical shift-right and a circular shift-left.

4.21

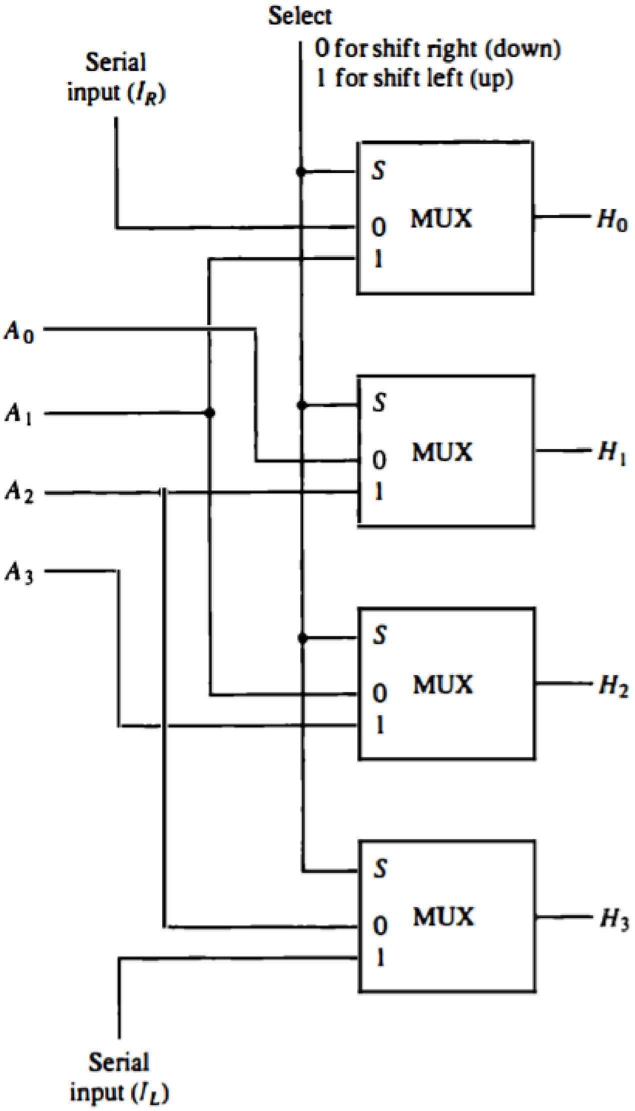
R = 11011101

Logical shift left: 10111010

Circular shift right: 01011101

Logical shift right: 00101110

Circular shift left: 01011100



Function table

Select	Output			
S	H_0	H_1	H_2	H_3
0	I_R	A_0	A_1	A_2
1	A_1	A_2	A_3	I_L

Figure 4-12 4-bit combinational circuit shifter.

4-22. What is the value of output H in Fig. 4-12 if input A is 1001, $S = 1$, $I_R = 1$, and $I_L = 0$?

4.22

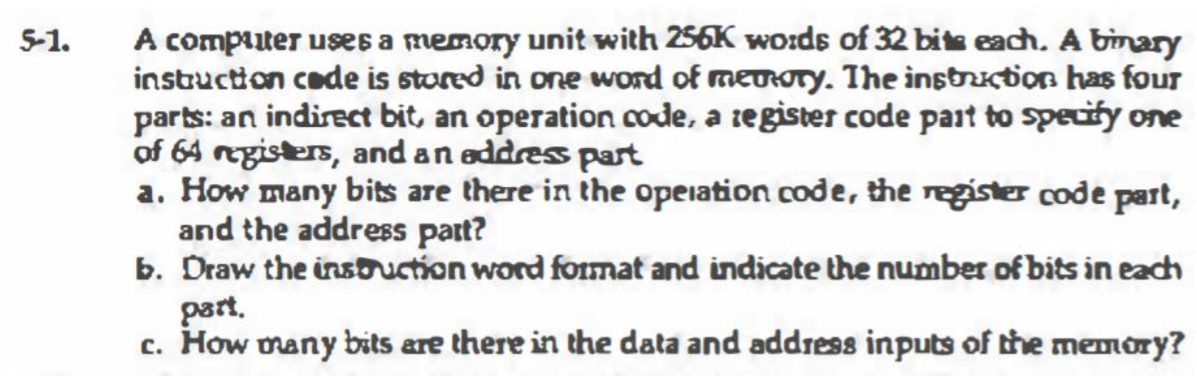
S = 1 Shift left

$A_0\ A_1\ A_2\ A_3\ I_L$

1 0 0 1 0

↙↙↙↙

H = 0 0 1 0 shift left


5-1. A computer uses a memory unit with 256K words of 32 bits each. A binary instruction code is stored in one word of memory. The instruction has four parts: an indirect bit, an operation code, a register code part to specify one of 64 registers, and an address part.

a. How many bits are there in the operation code, the register code part, and the address part?

b. Draw the instruction word format and indicate the number of bits in each part.

c. How many bits are there in the data and address inputs of the memory?

5.1

$256\text{ K} = 2^8 \times 2^{10} = 2^{18}$

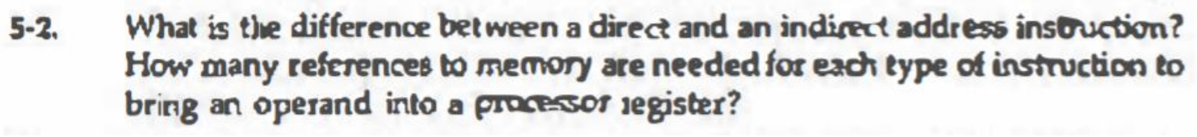
$64 = 2^6$

(a)	Address:		18 bits
	Register code:		6 bits
	Indirect bit:	1	bit
		25	32 – 25 = 7 bits for opcode.

(b)

1	7	6	18	= 32 bits
I	opcode	Register	Address	

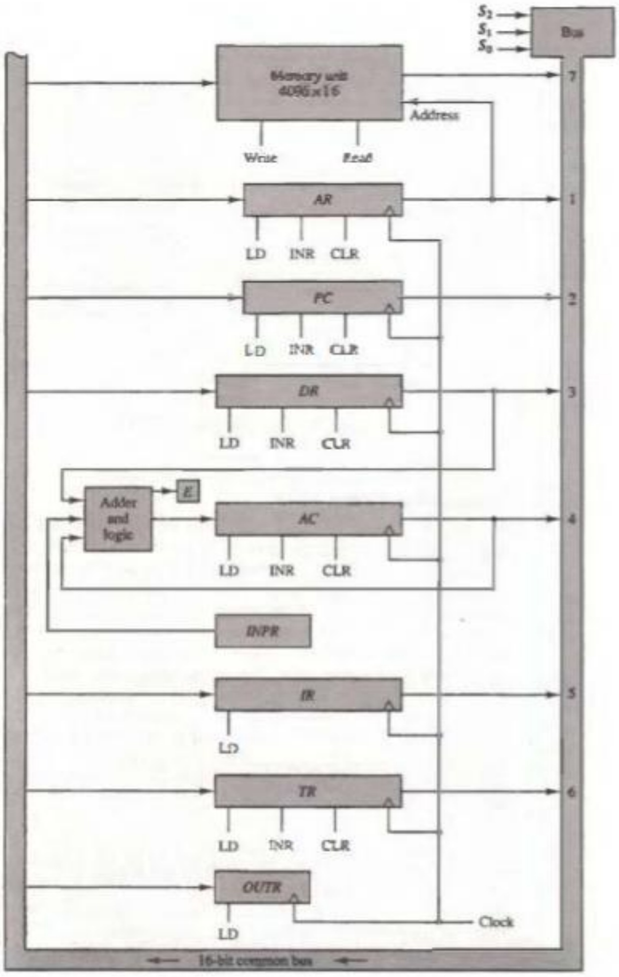
(c) Data; 32 bits; address: 18 bits.


5-2. What is the difference between a direct and an indirect address instruction? How many references to memory are needed for each type of instruction to bring an operand into a processor register?

5.2

A direct address instruction needs two references to memory: (1) Read instruction; (2) Read operand.

An indirect address instruction needs three references to memory: (1) Read instruction; (2) Read effective address; (3) Read operand.



5-3. The following control inputs are active in the bus system shown in Fig. 5.4. For each case, specify the register transfer that will be executed during the next clock transition.

	S_2	S_1	S_0	LD of register	Memory	Adder
a.	1	1	1	*IR*	Read	—
b.	1	1	0	*PC*	—	—
c.	1	0	0	*DR*	Write	—
d.	0	0	0	*AC*	—	Add

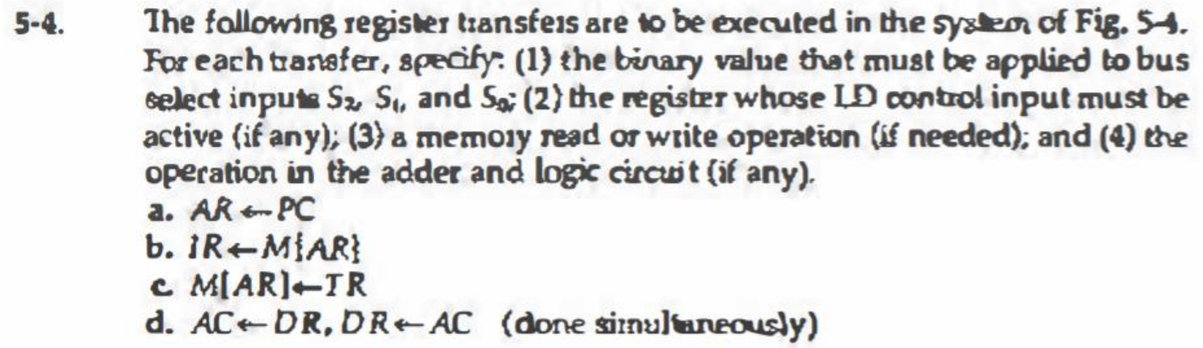
5.3

(a) Memory read to bus and load to IR: IR ← M[AR]

(b) TR to bus and load to PC: PC ← TR

(c) AC to bus, write to memory, and load to DR:
DR ← AC, M[AR]← AC

(d) Add DR (or INPR) to AC: AC ← AC + DR



5-4. The following register transfers are to be executed in the system of Fig. 5-4. For each transfer, specify: (1) the binary value that must be applied to bus select inputs S_2, S_1, and S_0; (2) the register whose LD control input must be active (if any); (3) a memory read or write operation (if needed); and (4) the operation in the adder and logic circuit (if any).

a. AR ← PC

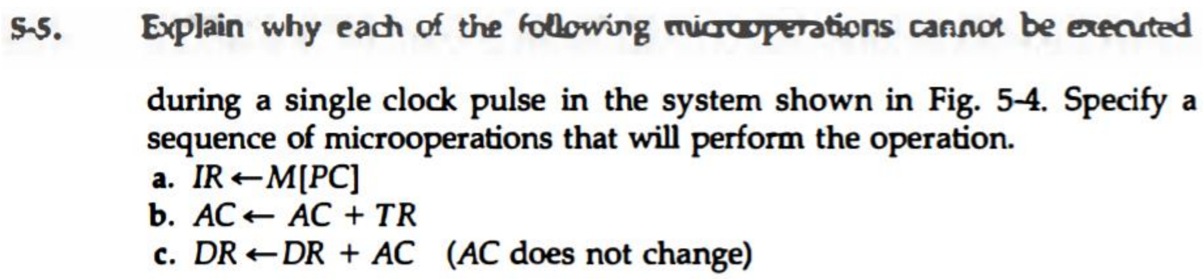
b. IR ← M[AR]

c. M[AR] ← TR

d. AC ← DR, DR ← AC (done simultaneously)

5.4

		(1) $S_2S_1S_0$	(2) Load(LD)	(3) Memory	(4) Adder
(a)	AR ← PC	010 (PC)	AR	—	—
(b)	IR ← M[AR]	111 (M)	IR	Read	—
(c)	M[AR] ← TR	110 (TR)	—	Write	—
(d)	DR ← AC AC ← DR	100 (AC)	DR and AC	—	Transfer DR to AC



5-5. Explain why each of the following microoperations cannot be executed during a single clock pulse in the system shown in Fig. 5-4. Specify a sequence of microoperations that will perform the operation.

a. *IR* ← *M*[*PC*]

b. *AC* ← *AC* + *TR*

c. *DR* ← *DR* + *AC* (*AC* does not change)

5.5

(a) IR ← M[PC] PC cannot provide address to memory. Address must be transferred to AR first

AR← PC

IR ← M[AR]

(b) AC ← AC + TR Add operation must be done with DR. Transfer TR to DR first.

DR ← TR

AC ← AC + DR

(c) DR ← DR + AC Result of addition is transferred to AC (not DR). To save value of AC its content must be stored temporary in DR (or TR).

AC ← DR, DR ← AC (See answer to Problem 5.4(d))

AC ← AC + DR

AC ← DR, DR ← AC

5-6. Consider the instruction formats of the basic computer shown in Fig. 5-5 and the list of instructions given in Table 5-2. For each of the following 16-bit instructions, give the equivalent four-digit hexadecimal code and explain in your own words what it is that the instruction is going to perform.

a. 0001 0000 0010 0100
b. 1011 0001 0010 0100
c. 0111 0000 0010 0000

5.6

(a) 0001 0000 0010 0010 = $(1024)_{16}$
ADD $(024)_{16}$
ADD content of M[024] to AC ADD 024

(b) 1 011 0001 0010 0100 = $(B124)_{16}$
I STA $(124)_6$
Store AC in M[M[124]] STA I 124

(c) 0111 0000 0010 0000 = $(7020)_{16}$
Register Increment AC INC

5-7. What are the two instructions needed in the basic computer in order to set the *E* flip-flop to 1?

5.7

CLE Clear E
CME Complement E

5-9. The content of *AC* in the basic computer is hexadecimal A937 and the initial value of *E* is 1. Determine the contents of *AC, E, PC, AR,* and *IR* in hexadecimal after the execution of the CLA instruction. Repeat 11 more times, starting from each one of the register-reference instructions. The initial value of *PC* is hexadecimal 021.

5.9

	E	AC	PC	AR	IR
Initial	1	A937	021	—	—
CLA	1	0000	022	800	7800
CLE	0	A937	022	400	7400
CMA	1	56C8	022	200	7200
CME	0	A937	022	100	7100
CIR	1	D49B	022	080	7080
CIL	1	526F	022	040	7040
INC	1	A938	022	020	7020
SPA	1	A937	022	010	7010
SNA	1	A937	023	008	7008
SZA	1	A937	022	004	7004
SZE	1	A937	022	002	7002
HLT	1	A937	022	001	7001

5-10. An instruction at address 021 in the basic computer has *I* = 0, an operation code of the AND instruction, and an address part equal to 083 (all numbers are in hexadecimal). The memory word at address 083 contains the operand B8F2 and the content of *AC* is A937. Go over the instruction cycle and determine the contents of the following registers at the end of the execute phase: *PC*, *AR*, *DR*, *AC*, and *IR*. Repeat the problem six more times starting with an operation code of another memory-reference instruction.

5.10

	PC	AR	DR	AC	IR
Initial	021	—	—	A937	—
AND	022	083	B8F2	A832	0083
ADD	022	083	B8F2	6229	1083
LDA	022	083	B8F2	B8F2	2083
STA	022	083	—	A937	3083
BUN	083	083	—	A937	4083
BSA	084	084	—	A937	5083
ISZ	022	083	B8F3	A937	6083

5-11. Show the contents in hexadecimal of registers *PC*, *AR*, *DR*, *IR*, and *SC* of the basic computer when an ISZ indirect instruction is fetched from memory and executed. The initial content of *PC* is 7FF. The content of memory at address 7FF is EA9F. The content of memory at address A9F is 0C35. The content of memory at address C35 is FFFF. Give the answer in a table with five columns, one for each register and a row for each timing signal. Show the contents of the registers after the positive transition of each clock pulse.

5.11

	PC	AR	DR	IR	SC
Initial	7FF	—	—	—	0
T_0	7FF	7FF	—	—	1
T_1	800	7FF	—	EA9F	2
T_2	800	A9F	—	EA9F	3
T_3	800	C35	—	EA9F	4
T_4	800	C35	FFFF	EA9F	5
T_5	800	C35	0000	EA9F	6
T_6	801	C35	0000	EA9F	0

5-12. The content of *PC* in the basic computer is 3AF (all numbers are in hexadecimal). The content of *AC* is 7EC3. The content of memory at address 3AF is 932E. The content of memory at address 32E is 09AC. The content of memory at address 9AC is 8B9F.

a. What is the instruction that will be fetched and executed next?

b. Show the binary operation that will be performed in the *AC* when the instruction is executed.

c. Give the contents of registers *PC*, *AR*, *DR*, *AC*, and *IR* in hexadecimal and the values of *E*, *I*, and the sequence counter *SC* in binary at the end of the instruction cycle.

5.12

(a) 9 = (1001)

$\underset{I=1}{1}\,\underset{\text{ADD}}{\underbrace{001}}$ ADD I 32E

Memory

3AF	932E
32E	09AC
9AC	8B9F

AC = 7EC3

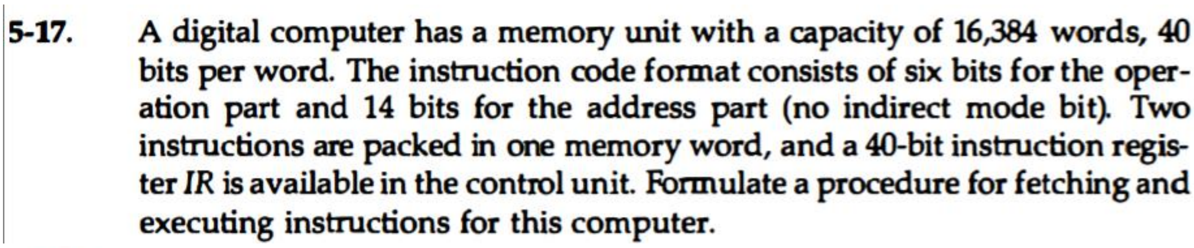
(b)

AC = 7EC3 (ADD)
DR = <u>8B9F</u>
0A62

E=1

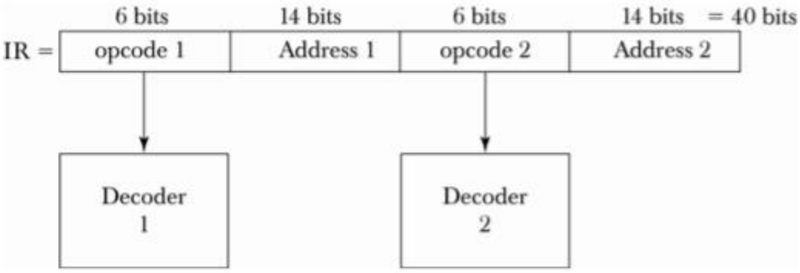
(c)

PC = 3AF + 1 = 3BO IR = 932E
AR = 7AC E = 1
DR = 8B9F I = 1
AC = 0A62 SC = 0000



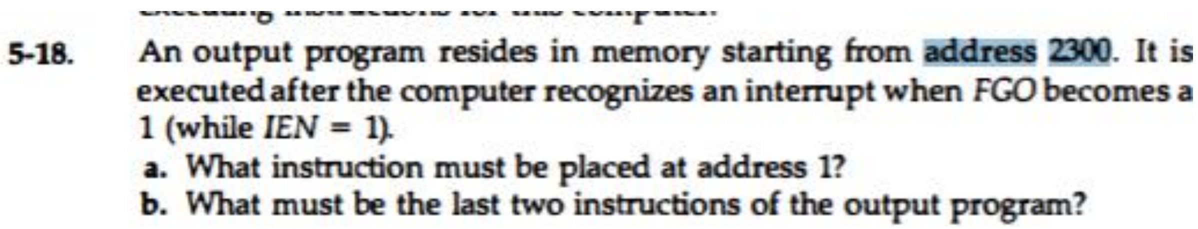
5-17. A digital computer has a memory unit with a capacity of 16,384 words, 40 bits per word. The instruction code format consists of six bits for the operation part and 14 bits for the address part (no indirect mode bit). Two instructions are packed in one memory word, and a 40-bit instruction register *IR* is available in the control unit. Formulate a procedure for fetching and executing instructions for this computer.

5.17



1. Read 40-bit double instruction from memory to IR and then increment PC.
2. Decode opcode 1.
3. Execute instruction 1 using address 1.
4. Decode opcode 2.
5. Execute instruction 2 using address 2.
6. Go back to step 1.

executing instructions for this computer.



5-18. An output program resides in memory starting from address 2300. It is executed after the computer recognizes an interrupt when *FGO* becomes a 1 (while *IEN* = 1).

a. What instruction must be placed at address 1?

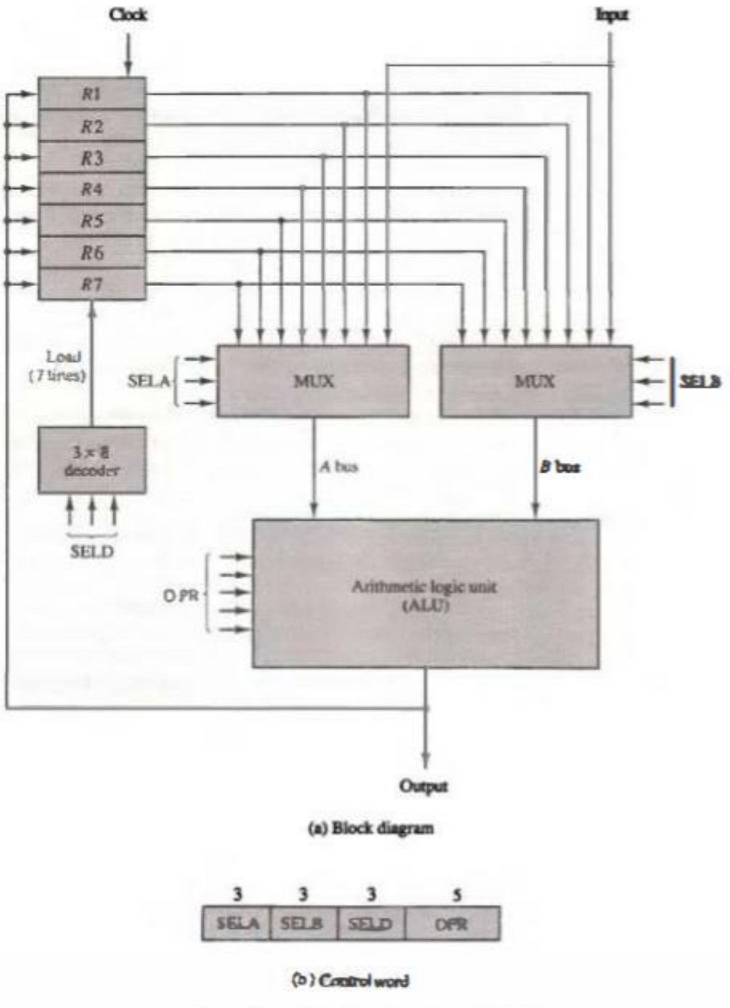
b. What must be the last two instructions of the output program?

5.18

(a) BUN 2300

(b) ION

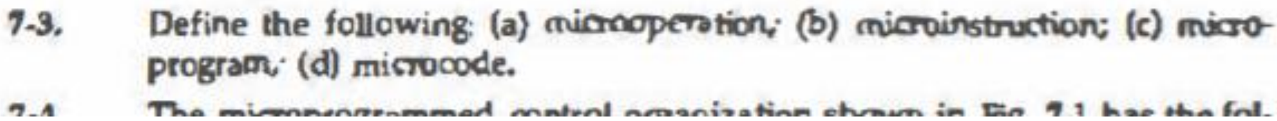
BUN 0 I (Branch indirect with address 0)



(a) Block diagram

(b) Control word

Figure 8-2 Register set with common ALU.

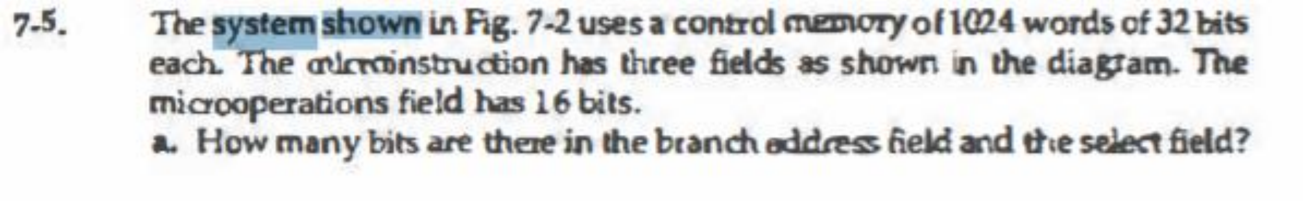


7-3. Define the following: (a) microoperation; (b) microinstruction; (c) microprogram; (d) microcode.

7-4 The microprogrammed control organization shown in Fig. 7-1 has the fol-

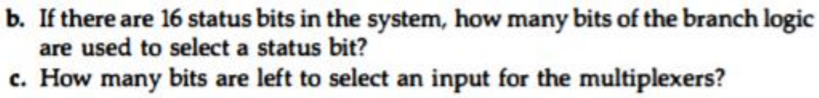
7.3

Micro operation - an elementary digital computer operation.
Micro instruction - an instruction stored in control memory.
Micro program - a sequence of microinstructions.
Micro code - same as microprogram.



7-5. The system shown in Fig. 7-2 uses a control memory of 1024 words of 32 bits each. The microinstruction has three fields as shown in the diagram. The microoperations field has 16 bits.

a. How many bits are there in the branch address field and the select field?



b. If there are 16 status bits in the system, how many bits of the branch logic are used to select a status bit?

c. How many bits are left to select an input for the multiplexers?

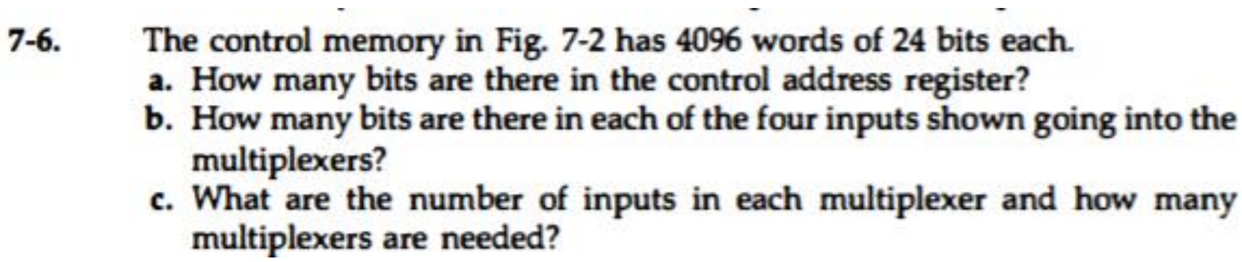
7.5

Control memory = $2^{10} \times 32$

(a)

6	10	16	=	32 bits
Select	Address	Micro operations		

(b) 4 bits
(c) 2 bits



7-6. The control memory in Fig. 7-2 has 4096 words of 24 bits each.

a. How many bits are there in the control address register?

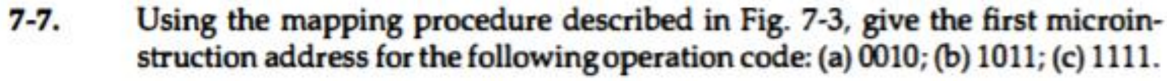
b. How many bits are there in each of the four inputs shown going into the multiplexers?

c. What are the number of inputs in each multiplexer and how many multiplexers are needed?

7.6

Control memory = $2^{12} \times 24$

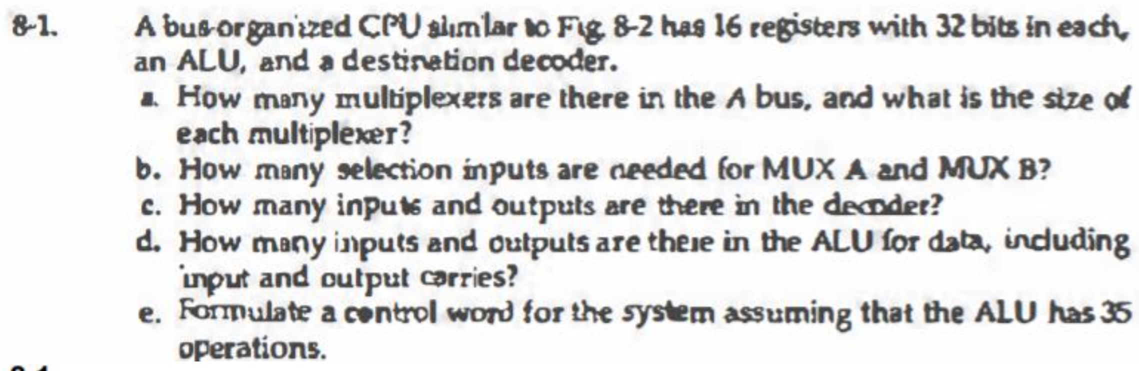
(a) 12 bits
(b) 12 bits
(c) 12 multiplexers, each of size 4-to-1 line.



7-7. Using the mapping procedure described in Fig. 7-3, give the first microinstruction address for the following operation code: (a) 0010; (b) 1011; (c) 1111.

7.7

(a) 0001000 = 8
(b) 0101100 = 44
(c) 0111100 = 60



8-1. A bus-organized CPU similar to Fig. 8-2 has 16 registers with 32 bits in each, an ALU, and a destination decoder.

a. How many multiplexers are there in the *A* bus, and what is the size of each multiplexer?

b. How many selection inputs are needed for MUX A and MUX B?

c. How many inputs and outputs are there in the decoder?

d. How many inputs and outputs are there in the ALU for data, including input and output carries?

e. Formulate a control word for the system assuming that the ALU has 35 operations.

8.1

(a) 32 multiplexers, each of size 16 × 1.

(b) 4 inputs each, to select one of 16 registers.

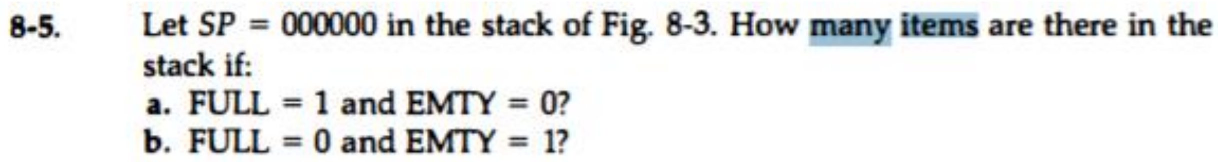
(c) 4-to-16 – line decoder

(d) 32 + 32 + 1 = 65 data input lines

32 + 1 = 33 data output lines.

(e)

4	4	4	6	= 18 bits
SELA	SELB	SELD	OPR	



8-5. Let *SP* = 000000 in the stack of Fig. 8-3. How many items are there in the stack if:

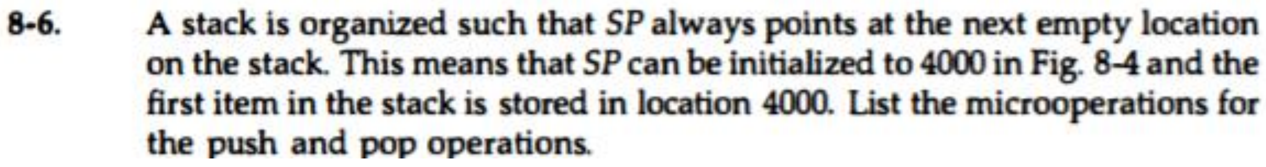
a. FULL = 1 and EMTY = 0?

b. FULL = 0 and EMTY = 1?

8.5

(a) Stack full with 64 items.

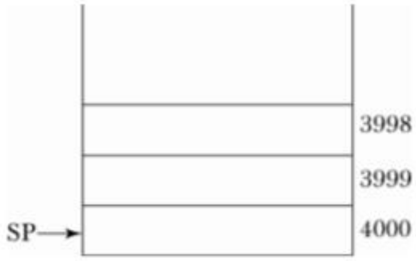
(b) stack empty



8-6. A stack is organized such that *SP* always points at the next empty location on the stack. This means that *SP* can be initialized to 4000 in Fig. 8-4 and the first item in the stack is stored in location 4000. List the microoperations for the push and pop operations.

(a) Stack full with 64 items.

(b) stack empty



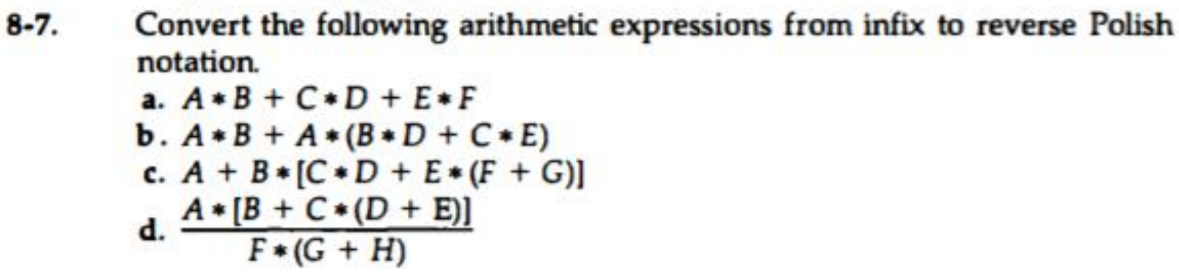
8.6

PUSH : $M[SP] \leftarrow DR$

$SP \leftarrow SP - 1$

POP : $SP \leftarrow SP + 1$

$DR \leftarrow M[SP]$



8-7. Convert the following arithmetic expressions from infix to reverse Polish notation.

a. $A * B + C * D + E * F$

b. $A * B + A * (B * D + C * E)$

c. $A + B * [C * D + E * (F + G)]$

d. $\dfrac{A * [B + C * (D + E)]}{F * (G + H)}$

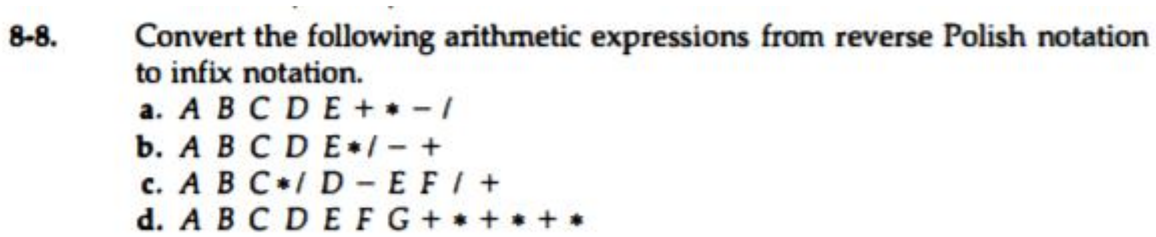
8.7

(a) AB * CD * EF * + +

(b) AB * ABD * CE * + * +

(c) FG + E * CD * + B * A +

(d) ABCDE + * + * FGH + */



8-8. Convert the following arithmetic expressions from reverse Polish notation to infix notation.

a. *A B C D E* + ∗ − /

b. *A B C D E*∗/ − +

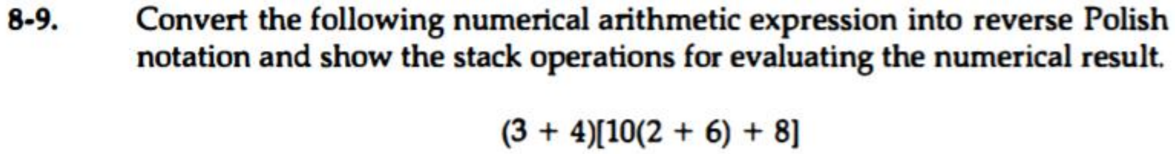
c. *A B C*∗/ *D* − *E F* / +

d. *A B C D E F G* + ∗ + ∗ + ∗

8.8

(a) $\frac{A}{B-(D+E)*C}$ (b) $A+B-\frac{C}{D*E}$

(c) $\frac{A}{B*C}-D+\frac{E}{F}$ (d) (((F + G) * E + D) * C + B) * A



8-9. Convert the following numerical arithmetic expression into reverse Polish notation and show the stack operations for evaluating the numerical result.

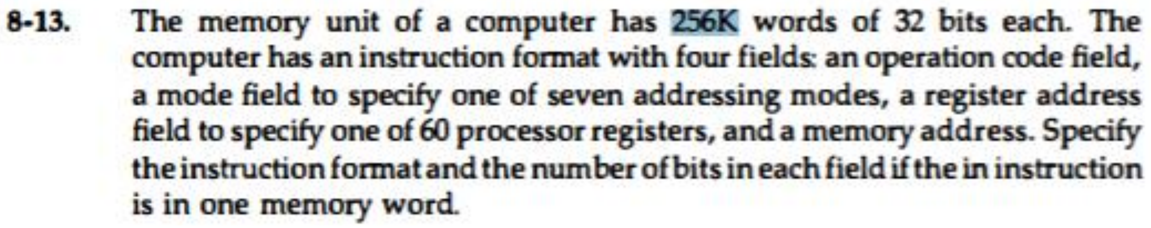
$$(3 + 4)[10(2 + 6) + 8]$$

8.9

(3 + 4) [10 (2 + 6) + 8] = 616

RPN : 34 + 2 6 + 10 * 8 + *

				6		10		8		
	4		2	2	8	8	80	80	88	
3	3	7	7	7	7	7	7	7	7	616
3	4	+	2	6	+	10	*	8	+	*

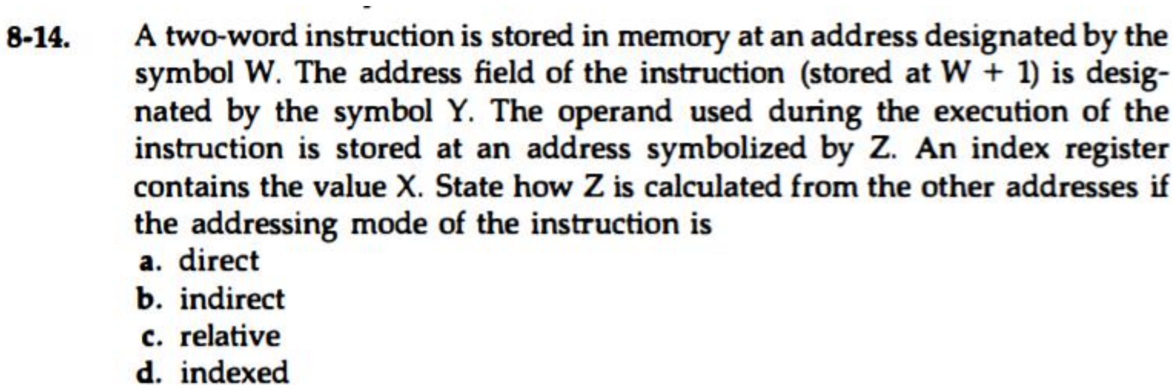


8-13. The memory unit of a computer has 256K words of 32 bits each. The computer has an instruction format with four fields: an operation code field, a mode field to specify one of seven addressing modes, a register address field to specify one of 60 processor registers, and a memory address. Specify the instruction format and the number of bits in each field if the in instruction is in one memory word.

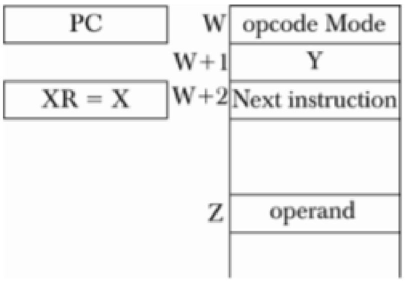
8.13
256 K = $2^8 \times 2^{10} = 2^{18}$

op code	Mode	Register	Address		
5	3	6	18	=	32

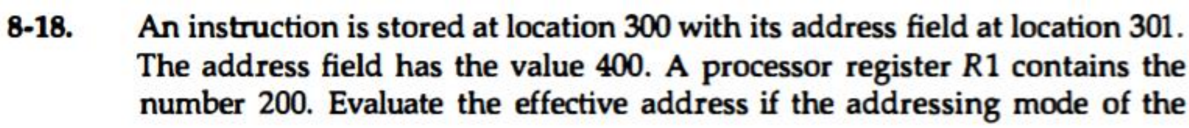
Address = 18 bits
Mode = 3 "
Register = 6 "
27 bits
op code 5
32 bits


8-14. A two-word instruction is stored in memory at an address designated by the symbol W. The address field of the instruction (stored at W + 1) is designated by the symbol Y. The operand used during the execution of the instruction is stored at an address symbolized by Z. An index register contains the value X. State how Z is calculated from the other addresses if the addressing mode of the instruction is
a. direct
b. indirect
c. relative
d. indexed

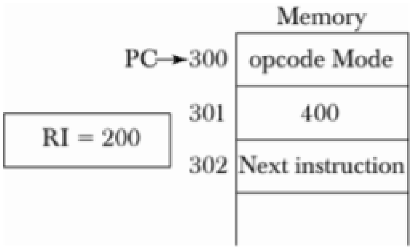
8.14
Z = Effective address
(a) Direct: Z = Y
(b) Indirect: Z = M[Y]
(c) Relative: Z = Y + W + 2
(d) Indexed: Z = Y + X

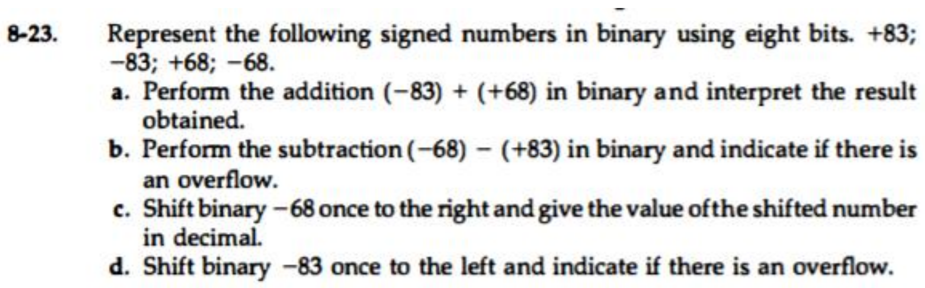


8.15
[illegible]


8-18. An instruction is stored at location 300 with its address field at location 301. The address field has the value 400. A processor register R1 contains the number 200. Evaluate the effective address if the addressing mode of the

8.18
Effective address
(a) Direct: 400
(b) Immediate: 301
(c) Relative: 302 + 400 = 702
(d) Reg. Indirect: 200
(e) Indexed: 200 + 400 = 600





8-23. Represent the following signed numbers in binary using eight bits. +83; −83; +68; −68.

a. Perform the addition (−83) + (+68) in binary and interpret the result obtained.

b. Perform the subtraction (−68) − (+83) in binary and indicate if there is an overflow.

c. Shift binary −68 once to the right and give the value of the shifted number in decimal.

d. Shift binary −83 once to the left and indicate if there is an overflow.

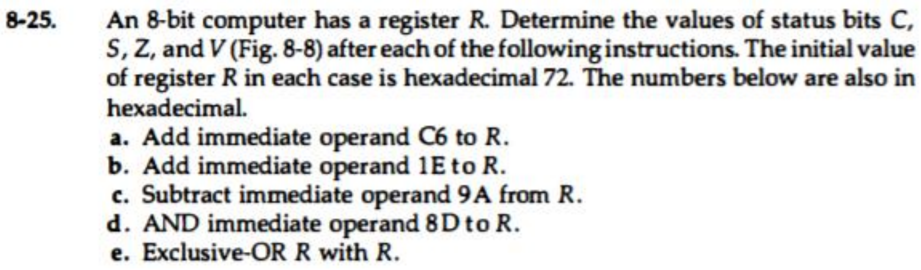
8.23

```
+ 83 = 01010011     – 83 = 10101101
+ 68 = 01000100     – 68 = 10111100
```

```
(a) – 83   10101101
    + 68  +01000100
    – 15   11110001
          (in 2's complement)
```

```
(b)          1 0 carries
    – 68     10111100
    – 83    +10101101
    –151     01101001
      ^
    –128  (over flow)
```

```
(c) – 68 = 10111100
    – 34 = 11011110
             ⊕ = 1
(d)  – 83 = 10101101
    –166 ≠ 01011010
             Over flow
```



8-25. An 8-bit computer has a register *R*. Determine the values of status bits *C*, *S*, *Z*, and *V* (Fig. 8-8) after each of the following instructions. The initial value of register *R* in each case is hexadecimal 72. The numbers below are also in hexadecimal.

a. Add immediate operand C6 to *R*.

b. Add immediate operand 1E to *R*.

c. Subtract immediate operand 9A from *R*.

d. AND immediate operand 8D to *R*.

e. Exclusive-OR *R* with *R*.

8.25

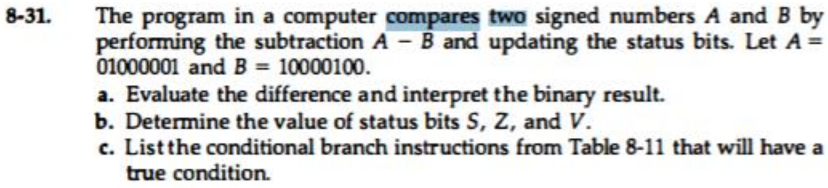
```
           1 1
(a) 72     01110010
    C6     11000110
    138    00111000
    C = 1  S = 0  Z = 0  V = 0

(b)        0 1
    72     01110010
    1E     00011110
    90     10010000
C = 0      S = 1  Z = 0  V = 1

(c) 9A =   10011010 \ 2's comp.
           01100110<
    72     01110010
    D8     11011000
C = 0  S = 1  Z = 0  V = 1
    (Borrow = 1)

(d) 72 =   01110010
    8D     10001100
    00     00000000

C = 0  S = 0  Z = 1  V = 0
(e) C = 0  S = 0  Z = 1  V = 0
```



8-31. The program in a computer compares two signed numbers A and B by performing the subtraction $A - B$ and updating the status bits. Let $A =$ 01000001 and $B =$ 10000100.

a. Evaluate the difference and interpret the binary result.

b. Determine the value of status bits S, Z, and V.

c. List the conditional branch instructions from Table 8-11 that will have a true condition.

8.31

(a)

$$\begin{aligned} A &= 01000001 = +\ 65 \\ \underline{B} &\underline{= 10000100 = -124} \quad + \\ A - B &= 10111101 \quad +189 \quad = \underbrace{010111101}_{9\text{ bits}} \end{aligned}$$

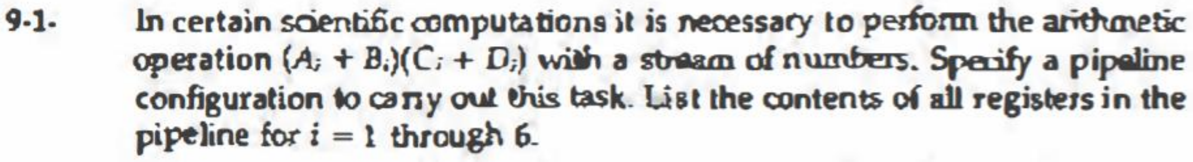
(b) S = 1 (sign reveral) +189 > 127

Z = 0

V = 1 (over flow) 65 > –124

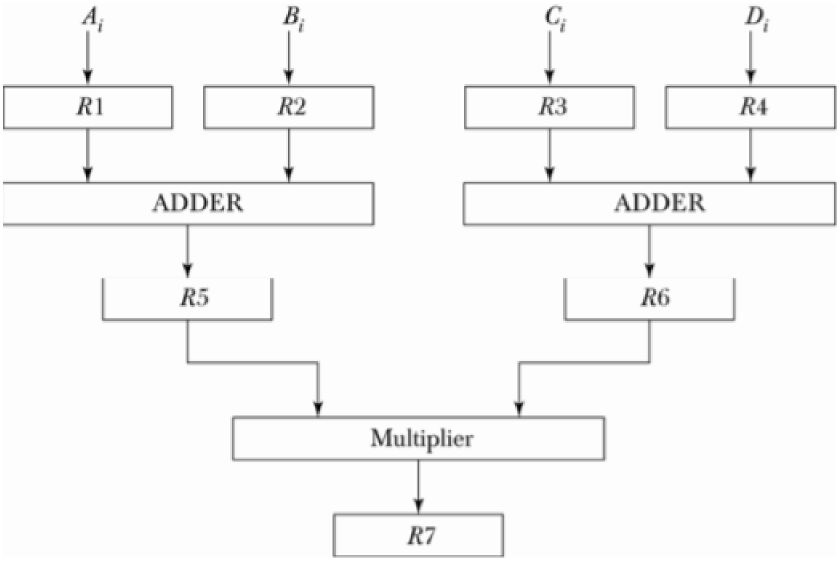
A > B

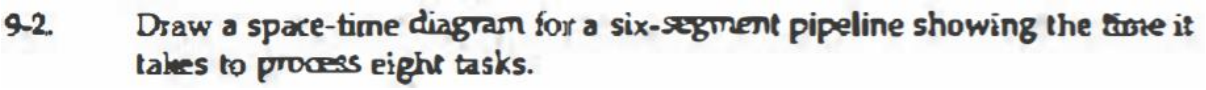
(c) BGT, BGE, BNE



9-1. In certain scientific computations it is necessary to perform the arithmetic operation $(A_i + B_i)(C_i + D_i)$ with a stream of numbers. Specify a pipeline configuration to carry out this task. List the contents of all registers in the pipeline for $i = 1$ through 6.

9.1





9-2. Draw a space-time diagram for a six-segment pipeline showing the time it takes to process eight tasks.

9-2

Segment	1	2	3	4	5	6	7	8	9	10	11	12	13
1	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8					
2		T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8				
3			T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8			
4				T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8		
5					T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	
6						T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8

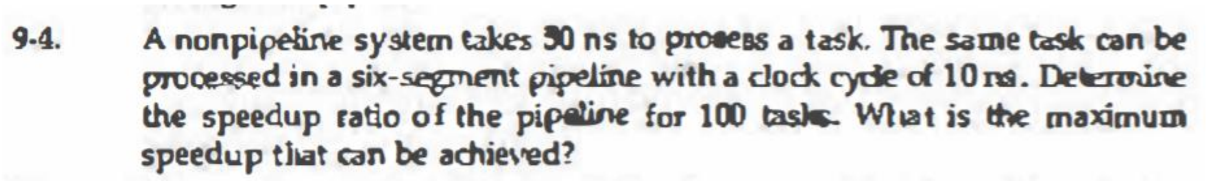
$(k + n - 1)t_p = 6 + 8 - 1 = 13$ cycles

9-3. Determine the number of clock cycles that it takes to process 200 tasks in a six-segment pipeline.

9.3

k = 6 segments

n = 200 tasks $(k + n - 1) = 6 + 200 - 1 = 205$ cycles



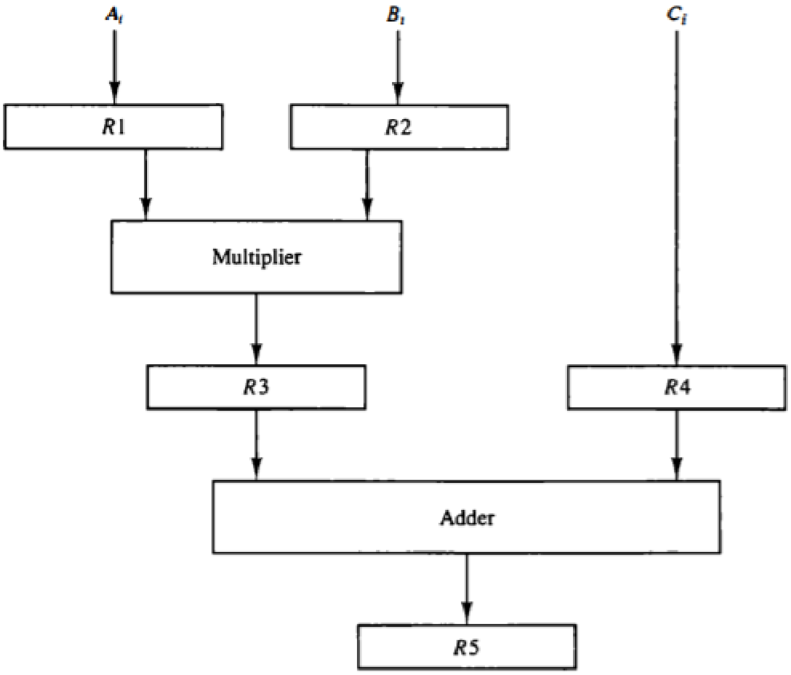
9-4. A nonpipeline system takes 30 ns to process a task. The same task can be processed in a six-segment pipeline with a clock cycle of 10 ns. Determine the speedup ratio of the pipeline for 100 tasks. What is the maximum speedup that can be achieved?

9.4
t_n = 50 ns
k = 6
t_p = 10 ns
n = 100

$$S = \frac{nt_n}{(k+n-1)t_p} = \frac{100 \times 50}{(6-99) \times 10} = 4.76$$

$$S_{max} = \frac{t_n}{t_p} = \frac{50}{10} = 5$$

Figure 9-2 Example of pipeline processing.



9-5. The pipeline of Fig. 9-2 has the following propagation times: 40 ns for the operands to be read from memory into registers $R1$ and $R2$, 45 ns for the signal to propagate through the multiplier, 5 ns for the transfer into $R3$, and 15 ns to add the two numbers into $R5$.

a. What is the minimum clock cycle time that can be used?
b. A nonpipeline system can perform the same operation by removing $R3$ and $R4$. How long will it take to multiply and add the operands without using the pipeline?
c. Calculate the speedup of the pipeline for 10 tasks and again for 100 tasks.
d. What is the maximum speedup that can be achieved?

9.5

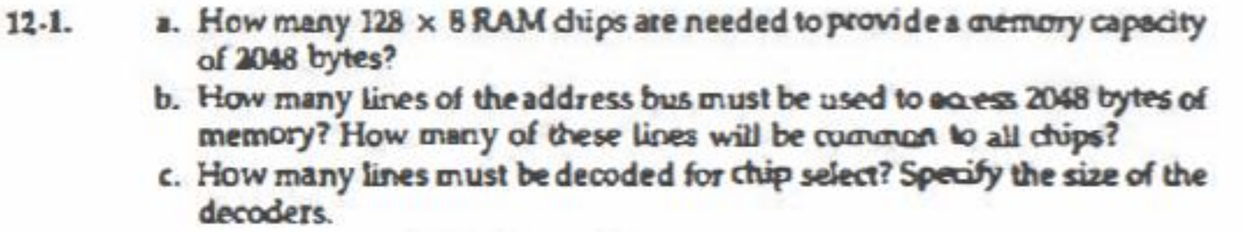
(a) $t_p = 45 + 5 = 50$ ns $\qquad$ k = 3

(b) $t_n = 40 + 45 + 15 = 100$ ns

(c) $$S = \frac{nt_n}{(k + n\text{-}1)\, t_p} = \frac{10 \times 100}{(3+9)50} = 1.67 \qquad \text{for n = 10}$$

$$= \frac{100 \times 100}{(3+99)50} = 1.96 \qquad \text{for n = 100}$$

(d) $$S_{max} = \frac{t_n}{t_p} = \frac{100}{50} = 2$$



12-1. a. How many 128 × 8 RAM chips are needed to provide a memory capacity of 2048 bytes?

b. How many lines of the address bus must be used to access 2048 bytes of memory? How many of these lines will be common to all chips?

c. How many lines must be decoded for chip select? Specify the size of the decoders.

12.1

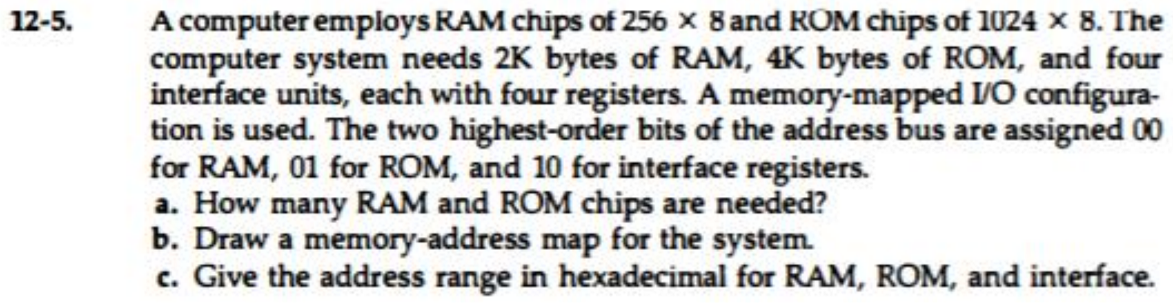
(a) $\frac{2048}{128} = 16\,\text{chips}$

(b) $2048 = 2^{11}$ $\qquad$ 11 lines to address 2078 bytes.

$128 = 2^7$ $\qquad$ _7_ lines to address each chip

4 lines to decoder for selecting 16 chips

(c) 4 × 16 decoder

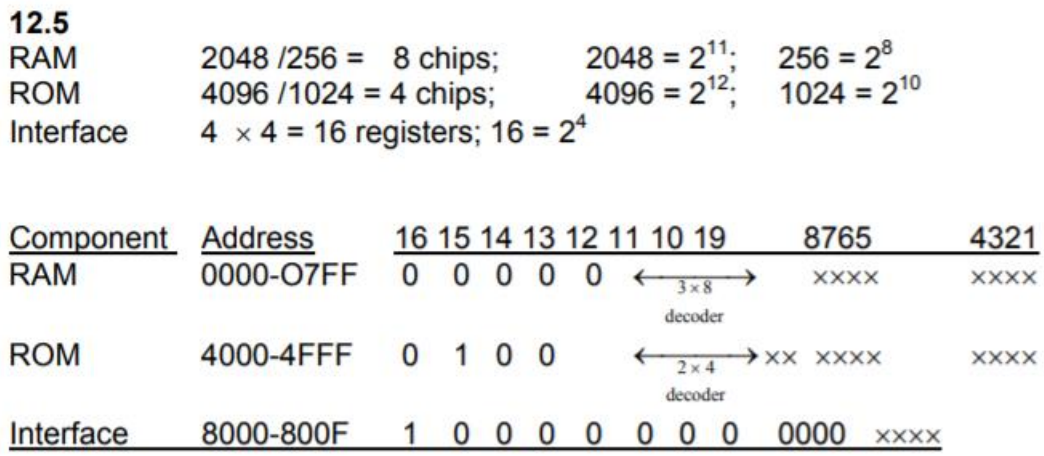


12-5. A computer employs RAM chips of 256 × 8 and ROM chips of 1024 × 8. The computer system needs 2K bytes of RAM, 4K bytes of ROM, and four interface units, each with four registers. A memory-mapped I/O configuration is used. The two highest-order bits of the address bus are assigned 00 for RAM, 01 for ROM, and 10 for interface registers.

a. How many RAM and ROM chips are needed?

b. Draw a memory-address map for the system.

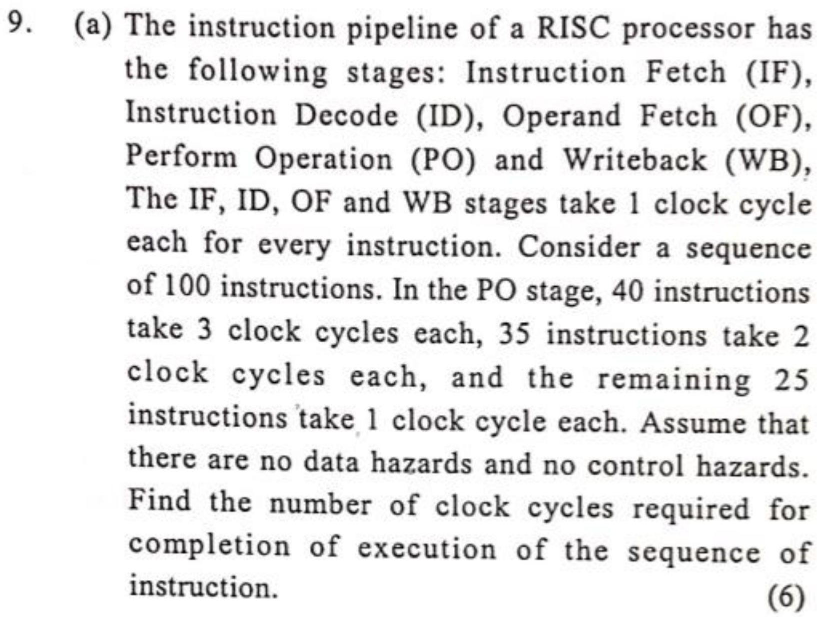
c. Give the address range in hexadecimal for RAM, ROM, and interface.



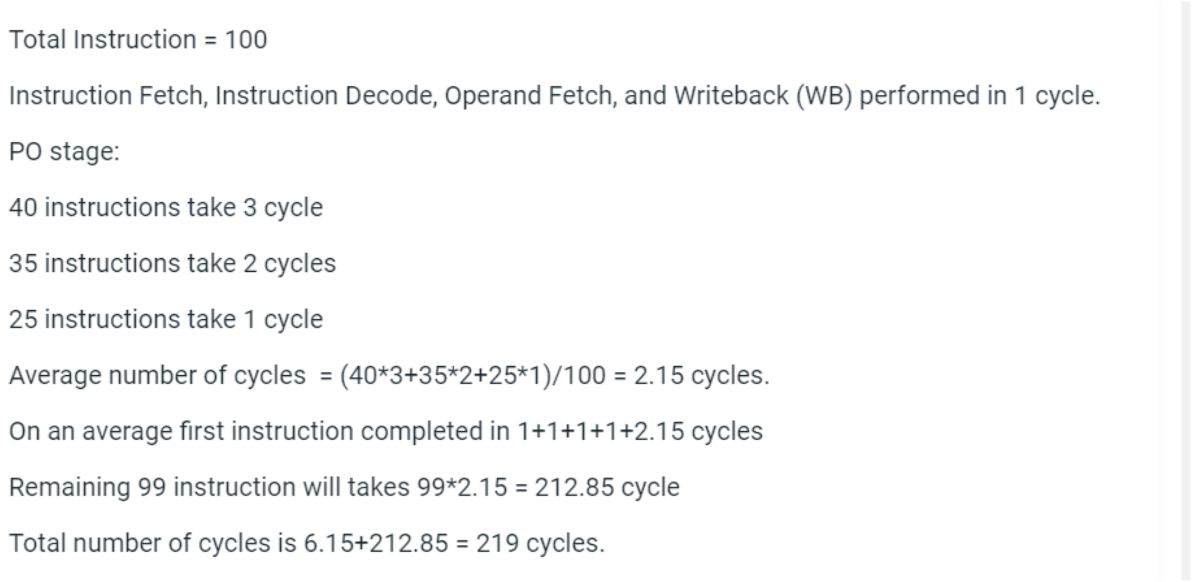
12.5

RAM	2048 /256 = 8 chips;	$2048 = 2^{11}$;	$256 = 2^{8}$
ROM	4096 /1024 = 4 chips;	$4096 = 2^{12}$;	$1024 = 2^{10}$
Interface	4×4 = 16 registers; $16 = 2^{4}$		

Component	Address	16 15 14 13 12 11 10 19	8765	4321
RAM	0000-O7FF	0 0 0 0 0 ←3×8 decoder→	xxxx	xxxx
ROM	4000-4FFF	0 1 0 0 ←2×4 decoder→ xx	xxxx	xxxx
Interface	8000-800F	1 0 0 0 0 0 0 0	0000	xxxx



9. (a) The instruction pipeline of a RISC processor has the following stages: Instruction Fetch (IF), Instruction Decode (ID), Operand Fetch (OF), Perform Operation (PO) and Writeback (WB), The IF, ID, OF and WB stages take 1 clock cycle each for every instruction. Consider a sequence of 100 instructions. In the PO stage, 40 instructions take 3 clock cycles each, 35 instructions take 2 clock cycles each, and the remaining 25 instructions take 1 clock cycle each. Assume that there are no data hazards and no control hazards. Find the number of clock cycles required for completion of execution of the sequence of instruction. (6)



Total Instruction = 100

Instruction Fetch, Instruction Decode, Operand Fetch, and Writeback (WB) performed in 1 cycle.

PO stage:

40 instructions take 3 cycle

35 instructions take 2 cycles

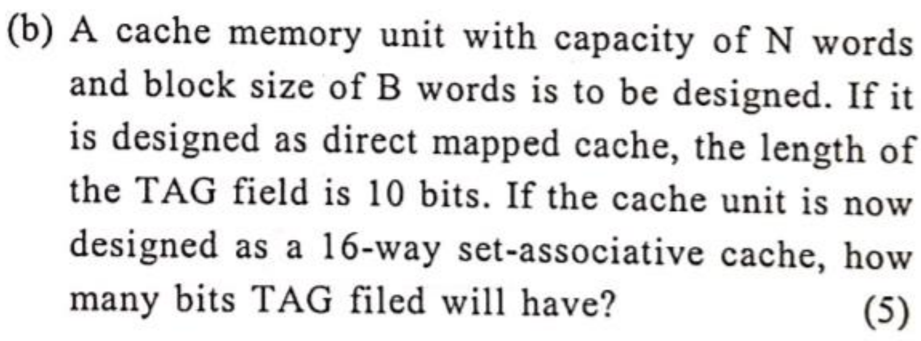
25 instructions take 1 cycle

Average number of cycles = (40*3+35*2+25*1)/100 = 2.15 cycles.

On an average first instruction completed in 1+1+1+1+2.15 cycles

Remaining 99 instruction will takes 99*2.15 = 212.85 cycle

Total number of cycles is 6.15+212.85 = 219 cycles.



(b) A cache memory unit with capacity of N words and block size of B words is to be designed. If it is designed as direct mapped cache, the length of the TAG field is 10 bits. If the cache unit is now designed as a 16-way set-associative cache, how many bits TAG filed will have? (5)

Here, direct mapped cache is compared with set associative cache, but in both the cases block offset is same so neglect it.

Calculation:

$10 + \log_2 (\frac{N}{B}) = x + \log_2 (\frac{N}{B}) / 16)$

$10 + \log_2 (\frac{N}{B}) = x + \log_2 (\frac{N}{B}) - \log_2 16$

$10 = x - 4$

$X = 14$

Tag bits for 16-bit set associative cache are 14.