Ch-3 - Programming the Basic Computer

1 Define Program and Categories of Program.

=> Program is a specific set of ordered operation for computer to perform.

Program is a set instruction that a computer follows.

Program contains a one-at-a time sequence of instruction that computer follows.

Program is a set instruction that process input, manipulate data and output a result.

- Categories of Program:

1 Binary Program:

Binary Program can only write with two number 0 and 1.

In this program, instruction can write in the form of 0 and 1 number combination.

2 Symbolic OP-code Program:

Symbolic codes includes the specify an operation code for machine instruction.

In this program, instruction can write in the form of mnemonic code.

3 Hexa Program:

Hexa Program can write in the hexadecimal number system.

In this program instruction can write in the form of Hexa number.

4 Assembly-Language Program:

Assembly Language is a programming Language that has a machine code instruction.

In this Program, A Instruction can write in the form of machine code.

5 Fortan Program:

Fortan Programming languages designed for numeric computation and scientific computing.

2 Explain Assembly Language and also state the rules of Language.

=> Assembly Language is a programming Language that has a machine code instruction.

Each processor have its own machine code instruction.

It requires less memory and less execution time.

- Assembly Language Program can be divided into three Section.

1) Data Section

Page No.
Date : / /

2) Bss Section
3) Text Section

1 Data Section :

The data section is used for declaring intialized data

Syntax : section .data

2 Bss Section :

The bss section is used for declaring variables.

Syntax : section .bss

3 Text Section :

The text section is used for writing the actual code of Program.

Syntax : section .text
global main
main:

- Assembly Program Language statements mainly divided into three part.

1) label
2) Instruction Field
3) Comment Field

Syntax:

label instruction ; comment

1 Label : Specify a symbolic address consists of up to 3 characters.

2 Instruction : Specify a machine or pseudo instruction.

3 Comment : text For user understand.

Page No.
Date : / /

3 Explain Pseudo Instruction:

=> A pseudo operation commanly called a pseudo-op.

A pseudo-op is an instruction to the assembler.

A pseudo-op code does not generated any type of machine code.

A pseudo-op code is directly execute by the assembler.

A pseudo-op instruction are directives for the assembler.

This are the basic Pseudo instruction.

1 ORG : Defines the absolute address of a section.

Syntax: ORG N

N is hexadecima number

2 END : defines the end of the the program.

Syntax : END

3 DEC : Convert decimal number to the Binary number

Syntax : DEC N

N = Hexadecimal number.

4 HEX : Convert hexadecimal number to the binary number.

Syntax : HEX N

N = Hexadecimal number.

Page No.
Date : / /

4 Define Assembler and explain First pass an assembler with Flow chart.

=> Assembler:

Assembler is convert low level assembly code into the machine code.

=> First Pass Assembler.

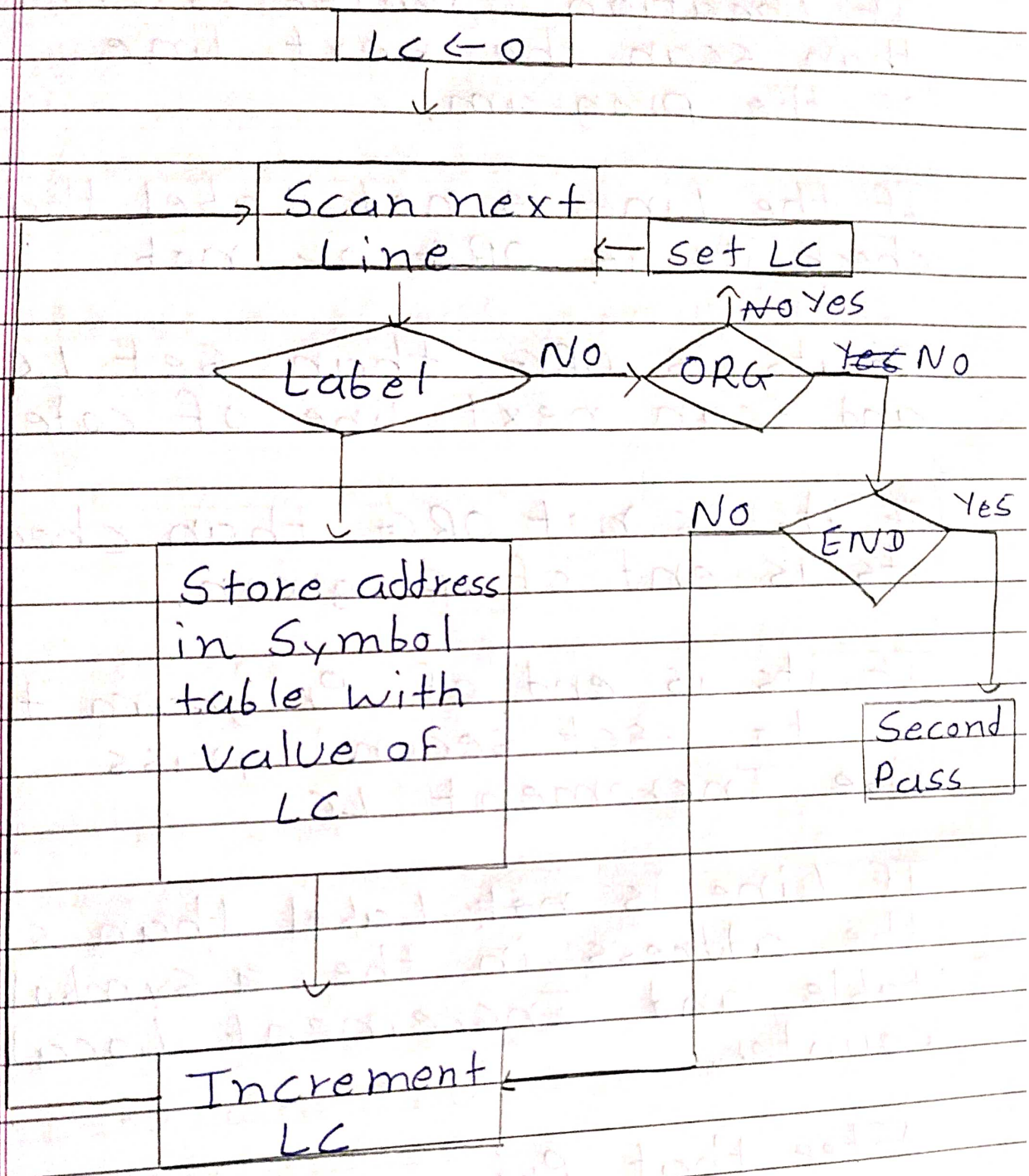
First Pass Assembler is use to shiffted to the Symbol.

Symbol which are used into a program are shiffted to the symbol table.

In this Pass, Symbol are store in the symbol table with its location.

- Flowchart:

First Pass started with Location counter is zero.


LC ← 0
Scan next Line
set LC
Label
NO
ORG
NO
Yes
Yes
NO
NO
END
Yes
Store address in Symbol table with value of LC
Second Pass
Increment LC

If Location Counter is zero than scan the next line of the program.

If the Line is not Label than check its is ORG or not.

If its is ORG, than set LC and scan next line of code.

If its is not ORG, than check its is end of program.

If its is end of Program than Go to ~~sect~~ second pass else Increment LC.

If Line is ~~not~~ Label than store the address in the a symbol table and Increment Location counter.

After that, Program execute next line.

5 Only draw Flowchart of First Pass Assembler.

=> (* Draw Que - 4 Flowchart)

6 Draw only Second Pass assembler Flowchart.

=> (* Draw Que - 7 Flowchart)

7 Explain the working of second pass assembler with its flowchart

=> After the First Pass assembler second Pass assembler start the working.

First set Location counter zero and scan next line of Program.

After that check this instruction is Pseudo or not.

If its a Pseudo instruction than check its is ORG or not. If its ORG than set

LC, else check its END or not. If it is END than END Process else convert the operand into binary code and store Location on LC and Increment LC.

If it is not Pseudo instruction than ek check it is MRI or not MRI instruction.

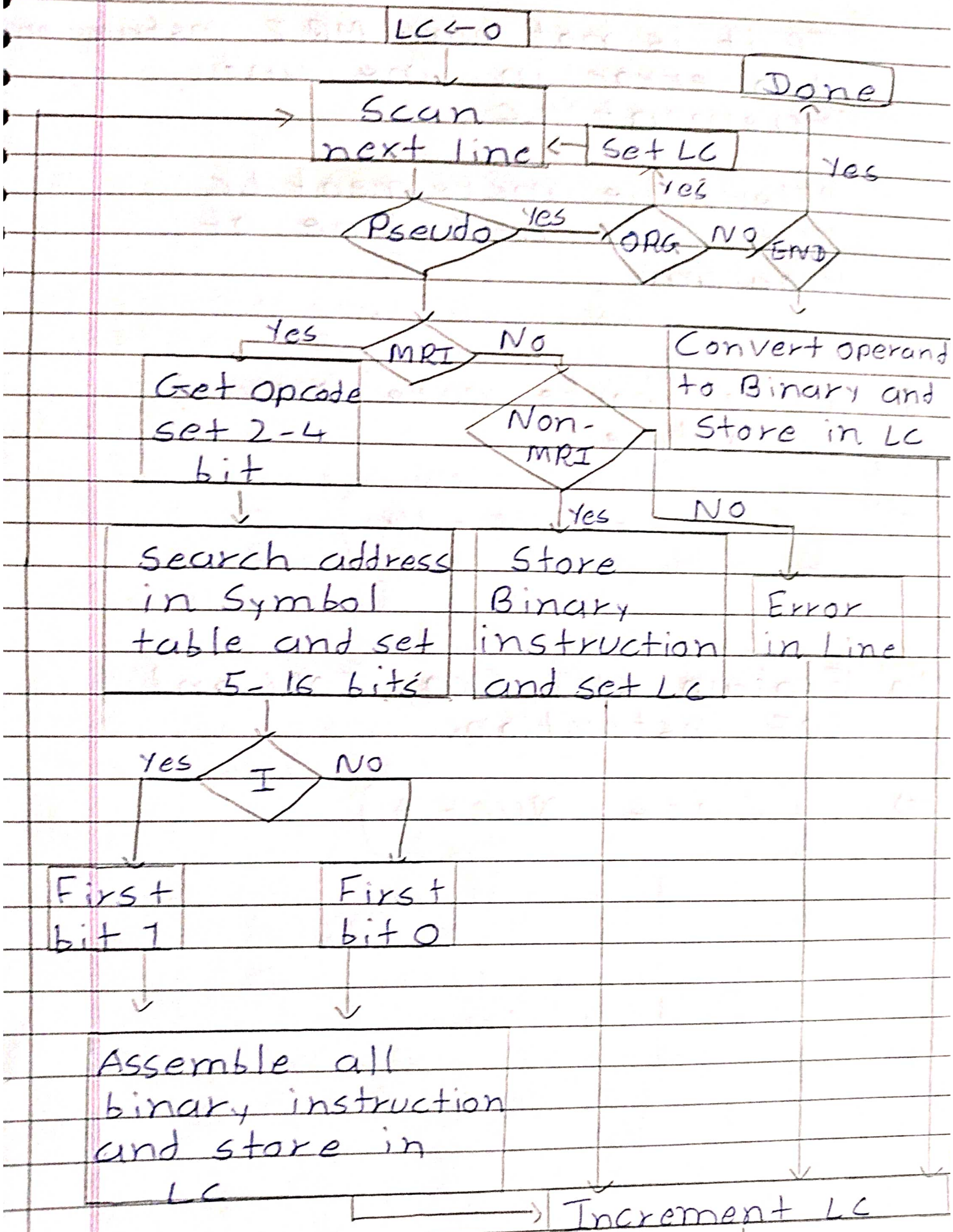
If it is MRI instruction than Get op code and set 2-4 bits and search address in Symbol table and set 5-16 bits.

After that check its I value.

If it is I than set first bit 1 and assemble instruction and store in LC and increment LC. else set first bit zero and assemble instruction and store in LC and increment LC.

If it is not MRI instruction than check it is non-MRI instruction than Store binary instruction in location by LC and Increment LC.

Date : / /


LC ← 0
Scan next line
Set LC
Done
Yes
Yes
Pseudo
Yes
ORG
NO
END
Yes
MRI
No
Get Opcode set 2-4 bit
Non-MRI
Convert operand to Binary and Store in LC
Yes
NO
Search address in Symbol table and set 5-16 bits
Store Binary instruction and set LC
Error in Line
Yes
I
NO
First bit 1
First bit 0
Assemble all binary instruction and store in LC
Increment LC

If it is not non-MRI instruction than error in line and increment LC.

After the increment LC scan the next line of Program.

8 Write Shorte note on Subroutine

=> (Ch-2-Que-14)

10 Explain SKI, SKO, ION and IOF instruction.

=> (Ch-2-Que-9)