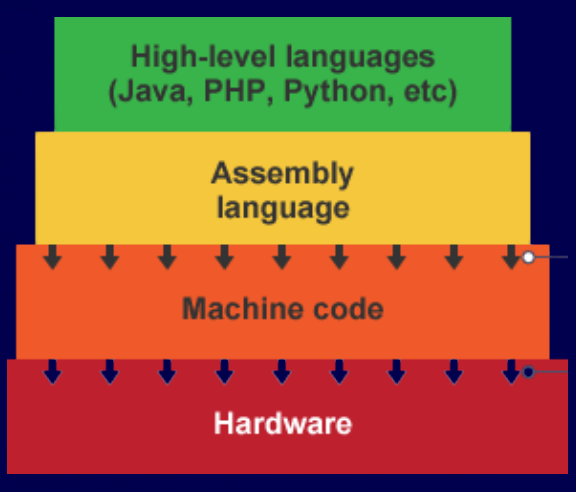
C034-Reneeka Nadkarni
C044-Amishi Desai
C049-Chahel Gupta
C065-Kirtan Thakkar

ASSEMBLERS

An assembler is a program that takes basic computer instructions and converts them into a pattern of bits that the computer's processor can use to perform its basic operations. Some people call these instructions assembler language and others use the term assembly language.



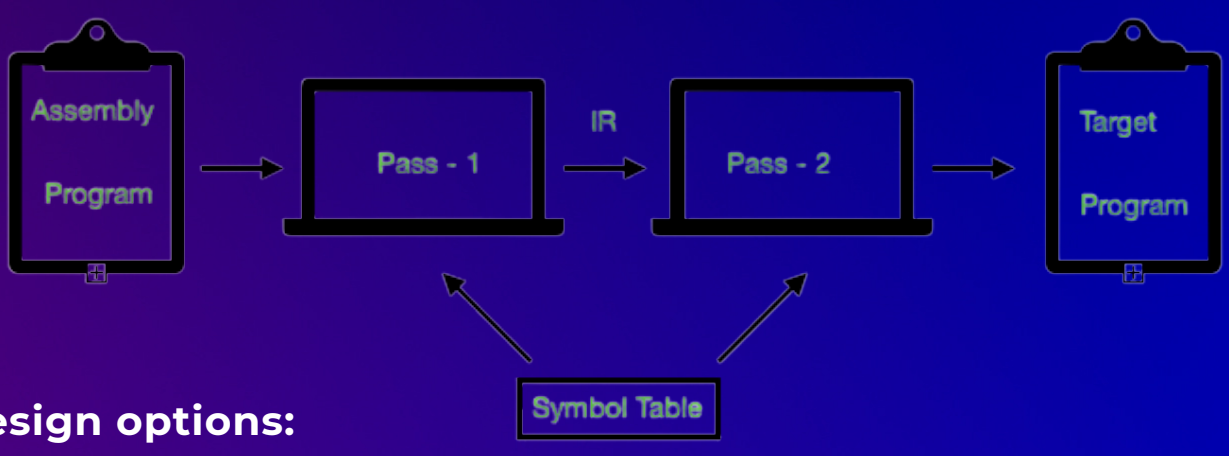
In assembly language, there are two types of instructions:

1. Pseudo-Op: does not generate any machine code. It is resolved during assembly, unlike machine instructions, which are resolved only at runtime. Eg: START, USING, L, A, ST, DC ,DS
2. Machine-op: represents a machine instruction to the assembler. Eg: BR

Design of assemblers performs:

– Scanning (tokenizing)
– Parsing (validating the instructions)
– Creating the symbol table
– Resolving the forward references
– Converting into the machine language

• The design comprises of:
– mnemonic operation codes ---> their machine language equivalents
– symbolic operands ---> their equivalent machine addresses
– data constants ---> internal machine representations
– writing the object program and assembly listing



Assemblers design options:

- One pass Assemblers
- 2 pass assemblers
- Multi pass assembler

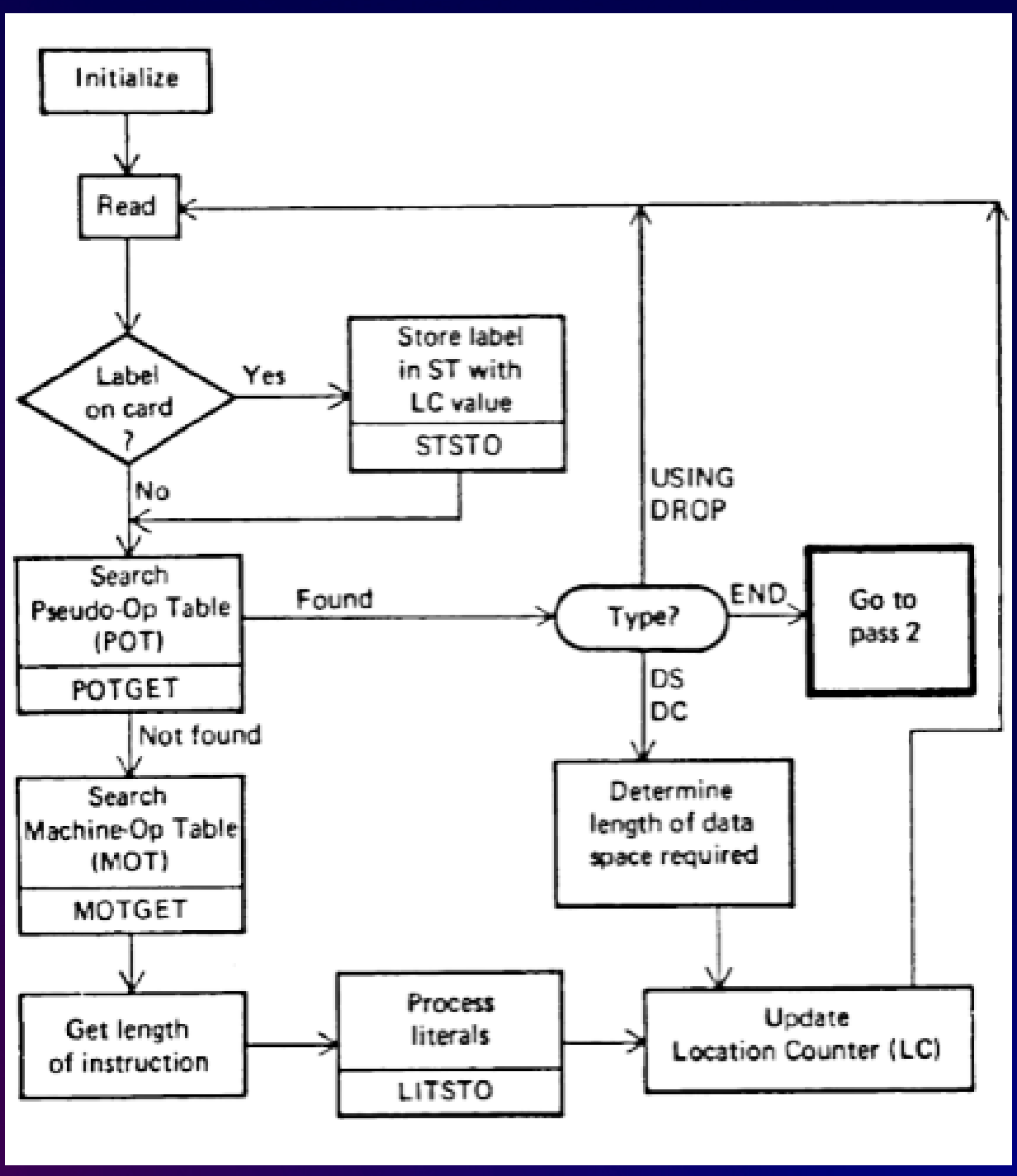
Data Structures for design of assemblers

LC	Location Counter	Pass 1 + Pass 2
MOT	Machine Operation Table	Pass 1 + Pass 2
POT	Pseudo Operation Table	Pass 1 + Pass 2
ST	Symbol Table	Pass 1 + Pass 2
LT	Literal Table	Pass 1 + Pass 2
BT	Base Table	Pass 2

Working of assemblers

- Checks the syntax of the instruction or directive. It faults if there is an error in the syntax, for example if a label is specified on a directive that does not accept one.
- Determines the size of the instruction and data being assembled and reserves space.
- Determines offsets of labels within sections.
- Creates a symbol table containing label definitions and their memory addresses.
- In the second pass, the assembler:
- Faults if an undefined reference is specified in an instruction operand or directive.
- Encodes the instructions using the label offsets from pass 1, where applicable.
- Generates relocations.
- Generates debug information if requested.
- Outputs the object file.

Pass-1



Pass-2