Batch 1
PA 17 Ketaki Patil

MIT-WPU
Dr. Vishwanath Karad
MIT WORLD PEACE UNIVERSITY | PUNE

SSC Lab Assignment no. 1
Design of Pass 1 of Two Pass Assembler

* **Aim:** Design suitable data structure and implement pass I of Two pass Assembler pseudo machine.

* **Objective:** Design suitable data structure and implement pass 1 of Two pass Assembler pseudo machine. Subset should consist of a few instructions from each category and few assembler directive.

* **Theory:**

① **Assembler:** Assembler is a program for converting instructions written in low assembly code into relocatable machine code and generating along information for loader.

Assembly code → [Assembler] → Machine code

It generates instructions by evaluating the mnemonics in operation field and find the value of symbol and literals to produce machine code. Now if assembler do all this work in one scan then it is called single pass assembler, Otherwise called multipass assembler. Here it divides task in two passes:

Pass I

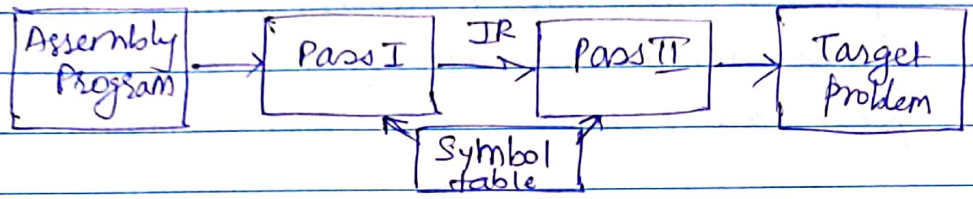
ⓐ Define symbols and literals and remember them in symbol table and literal table.

ⓑ Keep track of location counter.

(c) Process pseudo operations.

Pass 2

(a) Generate object code by converting symbolic opcode into respective numeric opcode.
(b) Generate data for literals and look for values of symbols.



(2) Design Specification of an assembler
(a) Analysis phase
(b) Synthesis phase
→ (a) Identify the information necessary to perform a task.
(b) Design a suitable data structure to record the information.
(c) Determine processing necessary to obtain and maintain the information.
(d) Determine the processing necessary to perform the task.

1. Analysis Phase

Known as front end of compiler, the analysis phase of compiler reads the src program, divides it into core parts and then checks for lexical, grammar and syntax errors. It generates intermediate code and symbol table, which is given as input to synthesis phase. Primary function is to generate table. Concept of memory allocated is

done using location counter. LC is always made to contain address of next memory word in target program. It is initialized to constant specified in START. To update it analysis phase needs to know the length of different instructions.

2. Synthesis Phase

Known as backend of compiler, the synthesis phase generates the target program with help of intermediate code and symbol table.

eg) MOVER BREG, ONE

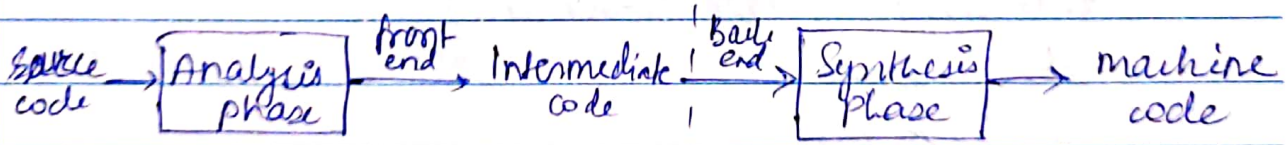
Address of memory word with which name ONE is associated depends on src program, so it must be made available by analysis phase.

Machine opcode corresponding to mnemonic MOVER not depends on src program, it depends on assembly language.

It uses two data structures:-

(i) Symbol table - built by analysis phases

(ii) Mnemonic table



* (3) Algorithm

Algorithm for pass 1

1. loccntr := 0; (default value)
2. while next statement is not an END statement.

(a) If label is present then
this-label := symbol in label field;
Enter (this-label, loccntr) in SYMTAB
(b) If a START or ORIGIN statement then
loccntr := value specified in operand field;
(c) If an EQU statement then
(i) this-addr := value of <address spec>;
(ii) Correct the symtab entry for this-label to (this-label, this-addr).
(d) If a declaration statement then
(i) code := code of the declaration statement;
(ii) size := size of memory area required by DC/DS
(iii) loccntr := loccntr + size;
(iv) Generate IC'(DL, code).
(f) If an imperative statement then
(i) code := machine opcode from OPTAB;
(ii) loccntr := loccntr + instruction length from OPTAB;
(iii) If operand is a symbol then
this-entry := SYMTAB entry number of operands;
Generate IC'(IS, code) (S, this-entry)';
3. (Processing of END statement)
(b) Generate IC
(c) Go to Pass II

* Input: Assembly language program / intermediate code generated by pass I.

★ Listing and errors handling:

→ Syntax error - Missing commas or parantheses.
Semantic error - Duplicate definition of symbols
Reference to undefined variables.

Eg) MOVER BREG,A
Error - Invalid opcode
A DC '5'
Error - Duplicate definition of sym A

★ Output:

for Batch 1 i/p:

```
START 100
MOVER AREG, A
L1 ADD BREG, A
MOVER BREG, B
ORIGIN L1
MOVER BREG, A
A DS 5
B DC 5
END
```

• Intermediate code. (IC) :

LC	opcode	operand1	operand2
-	(AD,01)	-	
100	(IS,04)	1	(S,1)
101	(IS,01)	2	(S,1)
102	(IS,04)	2	(S,3)
-	(AD,03)	-	
101	(IS,04)	2	(S,1)
102	(DL,02)	-	(C,5)
107	(DL,01)	-	(C,1)
-	(AD,02)	-	

- Symbol Table

Symbol_id	Symbolname	Address	Length
1.	A	102	5
2.	L1	101	1
3.	B	107	1

- Opcode table

Mnemonic	Op-code	class
START	01	AD
MOVER	04	IS
ADD	01	IS
MOVER	04	IS
ORIGIN	03	AD
MOVER	04	IS
DS	02	DL
DC	01	DL
END	02	AD

* Conclusion : The function of Pass 1 in assembler are studied along with errors coming in each pass.

* Platform : Windows (Java)