DPU

Assignment No 1

* Problem Statement :- Design suitable data structures and implement pass - I of a two pass Assembler for pseudo machine in Java / C++ using object oriented features. Implementation should consist of a instructions from each category and few assembler directives.

* Objectives :-

1 To study the design and implementation of 1st pass of two pass assembler.

2 To study the categorized instruction set of assembler

3 To study the data structure used in assembler implementation.

* Algorithms (Procedure)

Pass I

- Initialize location counter, entries of all tables as Zero.
- Read statements from input file one by one
- While next statement is not END statement.

I Tokenize or Separate out input statement as label, mnemonic, operand 1, operand 2

II If label is present insert label into symbol table

III If the statement is LTORG statement processes it by making it's entry into literal table, pool table, and allocate memory.

IV If statement is START or ORIGEN Process locatio counter accordingly.

V If an EQU statement, assign value to symbol by correcting entry in symbol table.

VI For declarating statement update code, size and location counter.

VII Generate intermediate code.

VIII Pass this intermediate code to pass-2

* Input :-

Source code of Assembly Language.

```
    SAMPLE        START          100
            USING                     *,15
             L                        1,FOUR
             A                        1,=F'3'
            ST                        1,RESULT
            SR                        1,2
            LTORG
             L                        2,FIVE
             A                        2,=F'5'
             A                        2,=F'7'
    FIVE          DC       F'5'
    FOUR          DC       F'4'
    RESULT        DC       1F
          END
```

* Output :-

```
100    SAMPLE    START    100
100              USING    *,15
100              L        1,FOUR
104              A        1,=F'3'
108              ST       1,RESULT
112              SR       1,2
114              LTORG
124              L        2,FIVE
128              A        2,=F'5'
132              A        2,=F'7'
136    FIVE      DC       F'5'
140    FOUR      DC       F'4'
144    RESULT    DS       IF
152              5
156              7
160              END
```

× Machine Opcode Table (MOT)

Mnemonic	Hex/Binary code	Length (Bytes)	Format
L	5A	4	RX
A	1B	4	RX
ST	50	4	RX
SR	18	2	RR

* Pseudo Opcode Table (POT)

Pseudo OP	Address / Name of Procedure to implement pseudo operation.
START	PSTART
USING	PUSING
DC	PDC
DS	PDS
LTORG	PLTORG
END	PEND

* Symbol Table (ST)

Sr. No.	Symbol Name	Address	Value	Length	Relocation
1	SAMPLE	100	–	160	R
2	FIVE	136	5	4	R
3	FOUR	140	4	4	R
4	RESULT	144	–	4	R

* Literal Table (LT)

Sr. No	Literal	Address	Length
1	3	120	4
2	5	152	4
3	7	156	4

* Test Cases :-

1 Check Syntax of instruction (correct and wrong)
2 Symbol not found.
3 Wrong instruction.
4 Duplicate Symbol declaration.
5 Test the output of program by changing value of START pseudo opcode.
6 Test the output of program by changing position of LTORG pseudo-op.

* Software Requirement :-

1 Fedora
2 Eclipse
3 JDK

* Conclusion :- Input assembly language program is processed by applying Pass-I algorithms of assembler and intermediate data structures, Symbol Table, Literal Table, MOT, POT, BT, etc generated.