Group A

SPOS : Assignment 1st

Nikita Bhosale.
Roll NO: 17

GURUKUL | Page No. | Date / /

Aim: To implement Pass I assembler

PB statement :- Design suitable DS and implement Pass I of two Pass assembler for Pseudo mlc i.e Java OO Return. Implementation should consist of few instruction for each category and few assembler directive.

Theory :-

① Assembly language:
It is low level Programming lang for computer and other programable device in which there is a very strong correspondence betn the lang. and architecture mlc code instruction.

② Assembler :
As lang is converted into executable mlc code by utility Prog referred to as an assembler.
The conversion process is refered to as assembly or assembling code
Assembler is a translater that translates Assembler Prog into conventional mlc lang

③ Assembler directives :-
- This are Pseudo instruction.
- not translated into mlc instruction
- only Provide instruction / direction / info to assembler directive.

Basic assembler directive.
START :- Name & starting address of source.
END :- Indicate end of source Prog

GURUKUL | Page No. | Date

④ Instruction format :-

① Direct addressing mode
addr of operand in instruction

② Register addressing mode :
one operand is general purpose reg

③ Register indirect addressing mode.
addr of operand is given by reg pair

④ Immediate addressing mode
operand is specified in instruction

⑤ Implict addressing mode :-
operation operate on contents of accumulat

Data Structure

⑤ Program Relocation :-
An object prog that contain type of modification info necessary to perform modifica is called as re-locable prog

⑥ literal :-
It is convenient for programes to be able to write the value of constant operand as a part of the instruction that user it.
- such operand is called as literal.

⑦ One pass assembler :-
Passes over the source file exactly once in the same pass collecting the labels resolving the future ref and doing the actual assembly

GURUKUL | Page No. | Date

① Algorithem for pass I assemblee :-

```
begin
    if starting address is given
        LOCCTR = starting address.
    else
        LOCCTR = 0
    while OPCODE := END do ; // OR EOF
    beging
        read a line from the code
        If there is a label
            if this lable in SYMTAB, then error
            eles. insert (lable, LOCCTR) into
                SYMTAB.
        Search OPTAB for OPCODE
        if found.
            LOCCTR += N ; (N lenof instruction.
        else if this is an assembly directive.
            update LOCCTR as directed
            else error.
            write time to intermediate rule
            end
        Program size = LOCCTR - starting address
    end.
```

conclusion :

Thus we have implemented Pass I assemblee using OO Reteres.