*Index*

|  |  |
| --- | --- |
| Topic | Page No. |
| Index | 1 |
| How to use ? | 2 |
| Instruction Format summary | 3 |
| General Architecture | 4 |
| Assembly instruction to binary encoding | 5 |
| Operation codes | 6 |
| Error Handling | 7 |
| Instruction Set Examples | 8 |

*How to use?*

Easy !

Run the ‘assemblerMainCode.py’ after placing the ‘sourceCode.txt’ containing the source Assembly program in the same directory. Then check the new files created for the output.

*Instruction Formats*

Type 1: no operands required

[opcode]

Example:

CLA

STP

Type 2: requires 1 operand

[opcode] [operand]

Example:

MUL R1

INP R7

Note: The operand can be a register, an immediate value, symbol or a literal.

In case of branch statements, it should be a label.

Example:

BRP loop

BRZ here

Pseudo Opcodes

*General Architecture*

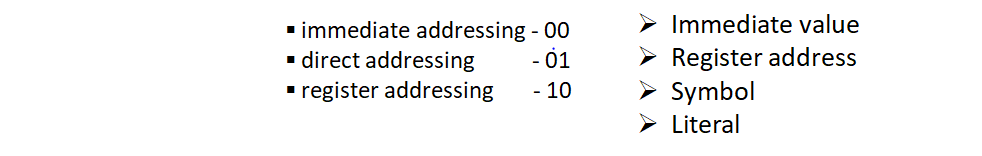
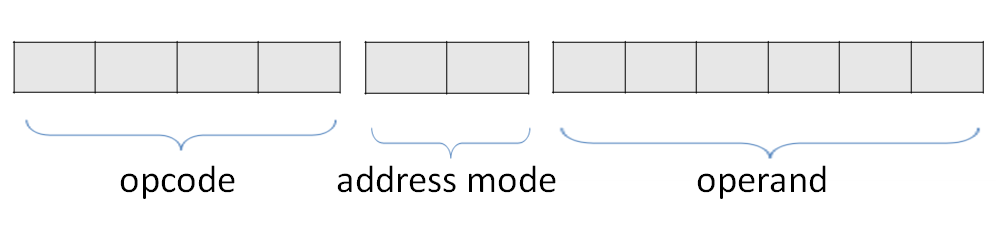
The assembler is built for a 12 bit accumulator architecture consisting of 16 registers and 64 memory locations.

The 16 registers are named as follows-

'R0', 'R1', 'R2', 'R3', 'R4', 'R5', 'R6', 'R7', 'R8', 'R9', 'R10', 'R11', 'R12', 'R13', 'R14', 'R15'.

Each memory location has length 1 word where 1 word = 12 bits.

*Assembly instruction to Binary Encoding*



*Operation Codes*

|  |  |  |
| --- | --- | --- |
| Operation Code | Instruction | Mnemonic |
| 0000 | Clear accumulator | CLA |
| 0001 | Load into accumulator from address | LAC |
| 0010 | Store accumulator contents into address | SAC |
| 0011 | Add address contents to accumulator contents | ADD |
| 0100 | Subtract address contents from accumulator contents | SUB |
| 0101 | Branch to address if accumulator contains zero | BRZ |
| 0110 | Branch to address if accumulator contains negative value | BRN |
| 0111 | Branch to address if accumulator contains positive value | BRP |
| 1000 | Read from terminal and put in address | INP |
| 1001 | Display value in address on terminal | DSP |
| 1010 | Multiply accumulator and address contents | MUL |
| 1011 | Divide accumulator contents by address content. Quotient in R1 and remainder in R2 | DIV |
| 1100 | Stop execution | STP |

*Error Handling*

Note: As soon as the assembler encounters an error, it will report the error on the python IDE along with the line number and will leave a blank line in the machine code output.

Zero Division Error:

When programmer sends 0 to the accumulator as immediate value with DIV opcode.

*Instruction set Examples*

T