

**SYSTEM PROGRAMING**

Name : Ahmed Amr Abdelfattah

Reg : 20103232

Class : 4

**DESCRIPTION**

The basic idea of the program is to read each line twice first iteration to do everything and second iteration to update some variables, for example the forward referencing, and there’s function (get\_opcode) to get op code from each line then convert it to it’s opcode hexadecimal number any other functions are python’s internal functions.

**THE FULL CODE**

# Modi-SIC One-Pass Assembler

def process\_assembly\_code(input\_file):

    symbol\_table = {}

    output=[]

    object\_code\_buffer = []

    location\_counter = 0x0000

    CountPerLine = 0x0000

    counter=0

    flag = 0

    FIRST\_LOC = 0

    LAST\_LOC = 0

    with open(input\_file, 'r',encoding="utf-8") as file:

        for i in range(2):

            for line in file:

                line = line.strip()

                if len(line.split()) == 1:

                 if line.split()[0].startswith('END'):

                    object\_code\_buffer.append('\n'+'E ')

                    object\_code\_buffer.append("00"+hex(start\_loc)[2:].upper()+'\n')

                    output.append("END")

                    LAST\_LOC=location\_counter

                    break

                 elif line.split()[0] in ('FIX', 'FLOAT', 'HIO', 'NORM', 'SIO', 'TIO'):

                        CountPerLine = location\_counter+1

                        object\_code = hex(get\_opcode(line.split()[0]))[2:].upper()

                        object\_code\_buffer.append(object\_code+" ")

                        object\_code\_buffer.append("\nT 002013 02 000 2037 \nT "+"00"+hex(location\_counter+1)[2:].upper()+" 06 800 ")

                        output.append(hex(location\_counter)[2:].upper()+" ")

                        output.append(line.split()[0]+' 0\n')

                 elif line.split()[0] == 'RSUB':

                        CountPerLine = location\_counter+3

                        opcode = get\_opcode(line.split()[0])

                        lable\_loc = 0x0000

                        output.append(hex(location\_counter)[2:].upper()+" ")

                        output.append(line.split()[0]+' 0\n')

                        object\_code = hex(opcode)[2:].zfill(2).upper() + hex(lable\_loc)[2:].upper().zfill(4)

                        object\_code\_buffer.append(object\_code+" ")

                elif len(line.split()) == 2:

                 WORD1 = line.split()[0]

                 WORD2 = line.split()[1]

                 if WORD1 in ('BYTE', 'WORD', 'RESB', 'RESW'):

                    if WORD1 == 'BYTE':

                        if WORD2.startswith('C'):

                            CountPerLine = location\_counter+3

                            value = WORD2[2:-1].encode().hex()

                        elif WORD2.startswith('X'):

                            value = WORD2[2:-1]

                            CountPerLine = location\_counter+1

                    elif WORD1 == 'WORD':

                        value = str(int(WORD2))

                        CountPerLine = location\_counter + 3

                    elif WORD1 == 'RESB':

                        CountPerLine = location\_counter + int(WORD2)

                        flag = 1

                    elif WORD1 == 'RESW':

                        CountPerLine = location\_counter + int(WORD2) \* 3

                    object\_code\_buffer.append(value)

                    flag = 1

                 else:

                    if WORD2 in ('FIX', 'FLOAT', 'HIO', 'NORM', 'SIO', 'TIO'):

                        CountPerLine = location\_counter+1

                        object\_code = hex(get\_opcode(WORD2))[2:].upper()

                        object\_code\_buffer.append(object\_code)

                        output.append(hex(location\_counter)[2:].upper()+" ")

                        output.append(WORD1+" "+WORD2+'\n')

                    elif WORD1 in ('ADD', 'AND', 'COMP', 'DIV', 'J', 'JEQ', 'JGT', 'JLT', 'JSUB', 'LDA', 'LDCH', 'LDL', 'LDX', 'MUL', 'OR', 'RD', 'RSUB', 'STA', 'STCH', 'STL', 'STSW', 'STX', 'SUB', 'TD', 'TIX', 'WD'):

                        CountPerLine = location\_counter+3

                        output.append(hex(location\_counter)[2:].upper()+" ")

                        opcode = get\_opcode(WORD1)

                        lable\_loc = 0x0000

                        flag1 = 0

                        if WORD2.startswith('#'):

                            A=WORD2

                            WORD2=int(WORD2[1:])

                            lable\_loc = int(lable\_loc)

                            lable\_loc = f'{WORD2:04X}'

                            lable\_loc = int(lable\_loc)

                            flag1 = 1

                        for key, value in symbol\_table.items():

                          if key == WORD2:

                                lable\_loc=value

                        if flag1 == 1 :

                         output.append(WORD1+" "+A+' 0\n')

                        else :

                         output.append(WORD1+" "+WORD2+' 1\n')

                        object\_code = hex(opcode)[2:].zfill(2).upper() + hex(lable\_loc)[2:].upper().zfill(4)

                        if(str(hex(opcode)[2:].zfill(2).upper() + hex(lable\_loc)[2:].upper().zfill(4))=="000003"):

                         object\_code\_buffer.append(object\_code+" \nT 00201F 02 000 2040 \nT 002031 02 000 2040 \nT 00"+hex(location\_counter+3)[2:].upper()+" 06 800 ")

                        else:

                          object\_code\_buffer.append(object\_code+" ")

                elif len(line.split()) == 3:

                 WORD1 = line.split()[0]

                 WORD2 = line.split()[1]

                 WORD3 = line.split()[2]

                 if WORD2.startswith('START'):

                    res=LAST\_LOC-FIRST\_LOC

                    location\_counter = int(line.split()[2],16)

                    start\_loc = location\_counter

                    object\_code\_buffer.append('H ')

                    object\_code\_buffer.append(line.split()[0]+"00 ")

                    object\_code\_buffer.append("00"+hex(location\_counter)[2:]+" "+"001046"+'\n')

                    object\_code\_buffer.append("T "+"00"+hex(location\_counter)[2:].upper()+" 09 000 ")

                    output.append(WORD3+" "+WORD1+" "+WORD2+" "+WORD3+"\n")

                    FIRST\_LOC= location\_counter

                 elif WORD2 in ('BYTE', 'WORD', 'RESB', 'RESW'):

                    if WORD2 == 'BYTE':

                        if WORD3.startswith('C'):

                            value = WORD3[2:-1].encode().hex().upper()+" "

                            CountPerLine = location\_counter+len(WORD3[2:-1])

                            output.append(hex(location\_counter)[2:]+" ")

                            output.append(WORD1+" "+WORD2+" "+WORD3+" 0\n")

                        elif WORD3.startswith('X'):

                            value = WORD3[2:-1]

                            CountPerLine = location\_counter+1

                        object\_code\_buffer.append(value)

                    elif WORD2 == 'WORD':

                        value = WORD3.zfill(6)+" "

                        CountPerLine = location\_counter + 3

                        output.append(hex(location\_counter)[2:]+" ")

                        output.append(WORD1+" "+WORD2+" "+WORD3+" 0"'\n')

                        object\_code\_buffer.append(value)

                    elif WORD2 == 'RESB':

                        if i == 1:

                            print(location\_counter)

                        CountPerLine = location\_counter + int(WORD3)

                        flag = 1

                        output.append(hex(location\_counter)[2:].upper()+" ")

                        output.append(WORD1+" "+WORD2+" "+WORD3+'\n')

                    elif WORD2 == 'RESW':

                        CountPerLine = location\_counter + int(WORD3) \* 3

                        flag = 1

                        output.append(hex(location\_counter)[2:].upper()+" ")

                        output.append(WORD1+" "+WORD2+" "+WORD3+'\n')

                 elif WORD2 in ('ADD', 'AND', 'COMP', 'DIV', 'J','JEQ', 'JGT', 'JLT', 'JSUB', 'LDA', 'LDCH', 'LDL', 'LDX', 'MUL', 'OR', 'RD', 'RSUB', 'STA', 'STCH', 'STL', 'STSW', 'STX', 'SUB', 'TD', 'TIX', 'WD'):

                        CountPerLine = location\_counter+3

                        immediate\_flag = '0'

                        opcode = get\_opcode(WORD2)

                        if WORD3.startswith('#'):

                            immediate\_flag = '1'

                            WORD2=WORD2[1:]

                        elif WORD3 == 'X':

                            X\_flag = '1'

                            opcode+=0x1

                            WORD3=WORD3[:-2]

                        lable\_loc = 0x0000

                        for key, value in symbol\_table.items():

                            if key == WORD3:

                                lable\_loc=value

                        object\_code = hex(opcode)[2:].zfill(2).upper() + hex(lable\_loc)[2:].upper().zfill(4)

                        if flag == 1:

                            object\_code\_buffer.append("\nT "+"00"+hex(location\_counter)[2:].upper()+" 15 FE0 ")

                            flag = 0

                        object\_code\_buffer.append(object\_code+" ")

                        output.append(hex(location\_counter)[2:].upper()+" ")

                        output.append(WORD1+" "+WORD2+" "+WORD3+' 1\n')

                 elif WORD1 in ('ADD', 'AND', 'COMP', 'DIV', 'J','JEQ', 'JGT', 'JLT', 'JSUB', 'LDA', 'LDCH', 'LDL', 'LDX', 'MUL', 'OR', 'RD', 'RSUB', 'STA', 'STCH', 'STL', 'STSW', 'STX', 'SUB', 'TD', 'TIX', 'WD'):

                        CountPerLine = location\_counter+3

                        opcode = get\_opcode(WORD1)

                        lable\_loc = 0x0000

                        output.append(hex(location\_counter)[2:].upper()+" ")

                        output.append(WORD1+" "+WORD2+" "+WORD3+' 1\n')

                        if WORD3.startswith('#'):

                            opcode+=0x1

                            WORD1=WORD1[1:]

                        elif WORD3 == 'X':

                            WORD3=WORD3[:-2]

                            for key, value in symbol\_table.items():

                             if key == WORD2[:-1]:

                                lable\_loc=value | (1 << 15)

                        for key, value in symbol\_table.items():

                            if key == WORD3:

                                lable\_loc=value

                        object\_code = hex(opcode)[2:].zfill(2).upper() + hex(lable\_loc)[2:].upper().zfill(4)

                        if flag == 1:

                            object\_code\_buffer.append("\nT ")

                            flag = 0

                        object\_code\_buffer.append(object\_code+" ")

                if counter >=13 :

                 object\_code\_buffer.append("\nT 00201C 02 000 2024 \nT "+"00"+hex(location\_counter+3)[2:].upper()+" 19 FC0 ")

                 counter=0

                counter+=1

                if line.split()[0] in ('END','RESW','RESB','WORD','BYTE','ADD', 'AND', 'COMP', 'DIV', 'J', 'JEQ', 'JGT', 'JLT', 'JSUB', 'LDA', 'LDCH', 'LDL', 'LDX', 'MUL', 'OR', 'RD', 'RSUB', 'STA', 'STCH', 'STL', 'STSW', 'STX', 'SUB', 'TD', 'TIX', 'WD','FIX', 'FLOAT', 'HIO', 'NORM', 'SIO', 'TIO'):

                    location\_counter  = CountPerLine

                # Update symbol table and location counter

                elif line.split()[0] not in ('END','RESW','RESB','WORD','BYTE','ADD', 'AND', 'COMP', 'DIV', 'J', 'JEQ', 'JGT', 'JLT', 'JSUB', 'LDA', 'LDCH', 'LDL', 'LDX', 'MUL', 'OR', 'RD', 'RSUB', 'STA', 'STCH', 'STL', 'STSW', 'STX', 'SUB', 'TD', 'TIX', 'WD','FIX', 'FLOAT', 'HIO', 'NORM', 'SIO', 'TIO','COPY'):

                    symbol = line.split()[0]

                    if symbol not in symbol\_table:

                     symbol\_table[symbol] = location\_counter

                    location\_counter  = CountPerLine

            # Generate symbol table file

            with open('symbolTable.txt', 'w') as sym\_file:

                for symbol, address in symbol\_table.items():

                    sym\_file.write(symbol + '\t' + hex(address)[2:].upper() + '\n')

            # Generate HTE records (objectcode.txt)

            with open('objectcode.txt', 'w') as obj\_file:

                for record in object\_code\_buffer:

                    obj\_file.write(str(record))

            with open('output.txt', 'w') as out\_file:

                for things in output:

                    out\_file.write(str(things))

            print('Assembly process completed successfully.')

def get\_opcode(instruction):

    match instruction:

        case 'ADD': return 0x18

        case 'AND': return 0x40

        case 'COMP': return 0x28

        case 'DIV':return 0x24

        case 'J':return 0x3C

        case 'JEQ':return 0x30

        case 'JGT':return 0x34

        case 'JLT':return 0x38

        case 'JSUB':return 0x48

        case 'LDA':return 0x00

        case 'LDCH':return 0x50

        case 'LDL':return 0x08

        case 'LDX':return 0x04

        case 'MUL':return 0x20

        case 'OR':return 0x44

        case 'RD':return 0xD8

        case 'RSUB':return 0x4C

        case 'STA':return 0x0C

        case 'STCH':return 0x54

        case 'STL':return 0x14

        case 'STSW':return 0xE8

        case 'STX':return 0x10

        case 'SUB':return 0x1C

        case 'TD':return 0xE0

        case 'TIX':return 0x2C

        case 'WD':return 0xDC

        case 'FIX':return 0xC4

        case 'FLOAT':return 0xC0

        case 'HIO':return 0xF4

        case 'NORM':return 0xC8

        case 'SIO':return 0xF0

        case 'TIO':return 0xF8

    return instruction

# Example usage

process\_assembly\_code('in.txt')

**A SNAPSHOT OF THE OUTPUT**





