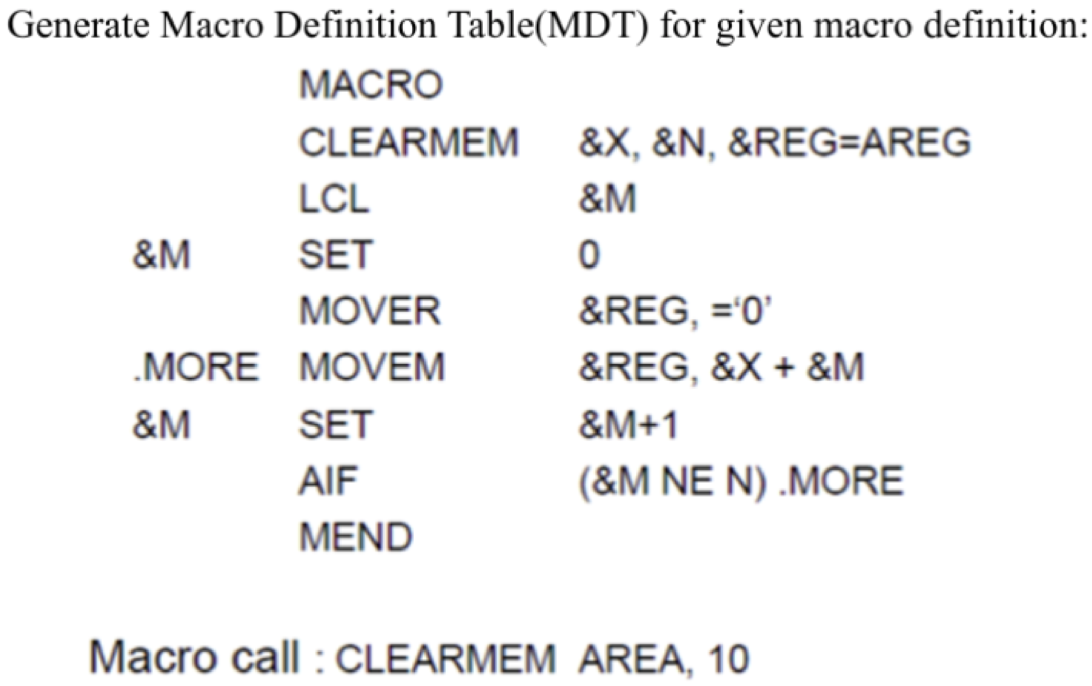
# SS LAB 5

## SAHIL BONDRE: U18CO021



Generate Macro Definition Table(MDT) for given macro definition:

```
        MACRO
        CLEARMEM    &X, &N, &REG=AREG
        LCL         &M
&M      SET         0
        MOVER       &REG, =‘0’
.MORE   MOVEM       &REG, &X + &M
&M      SET         &M+1
        AIF         (&M NE N) .MORE
        MEND
```

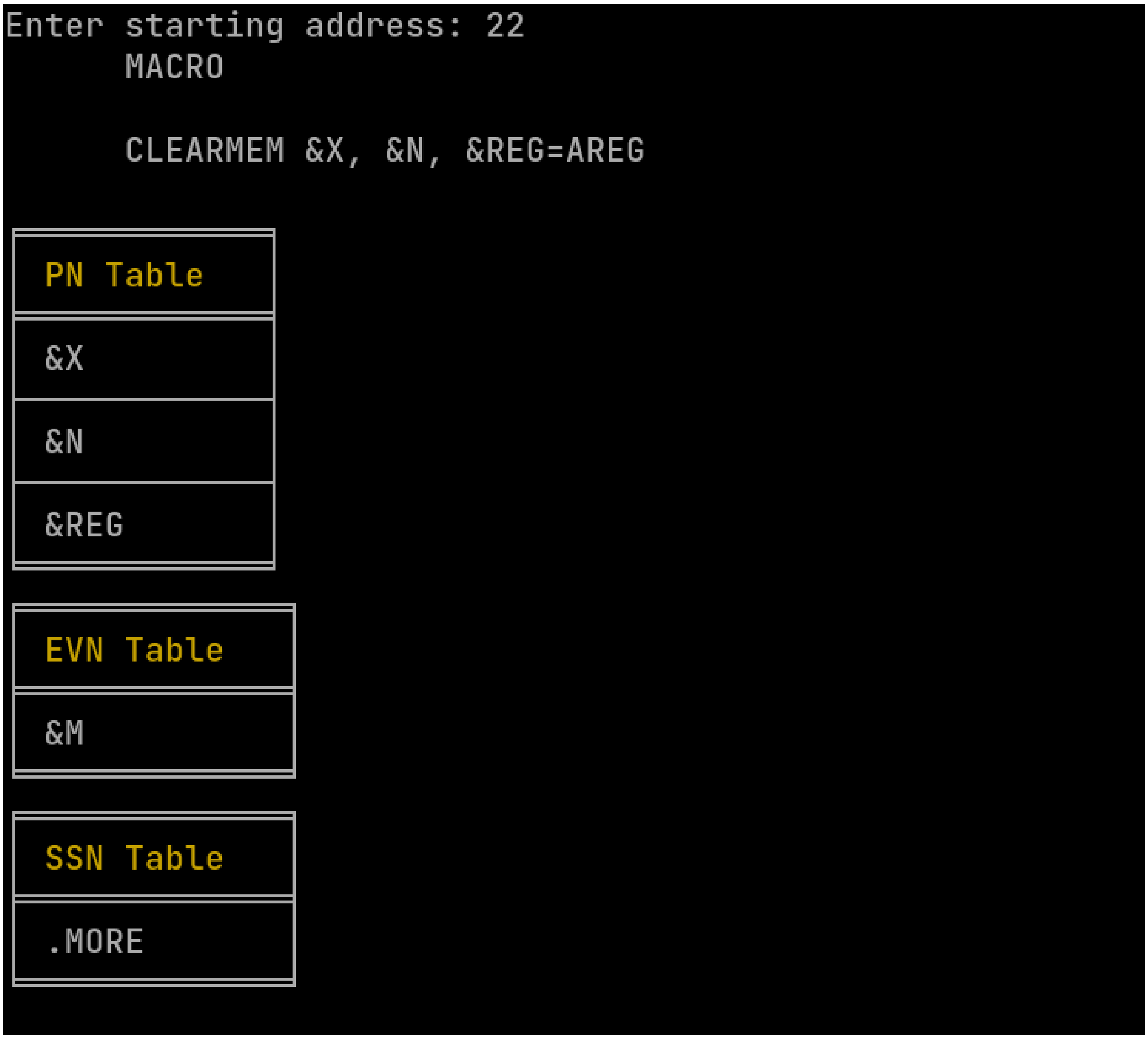
Macro call : CLEARMEM AREA, 10

**file.asm:**

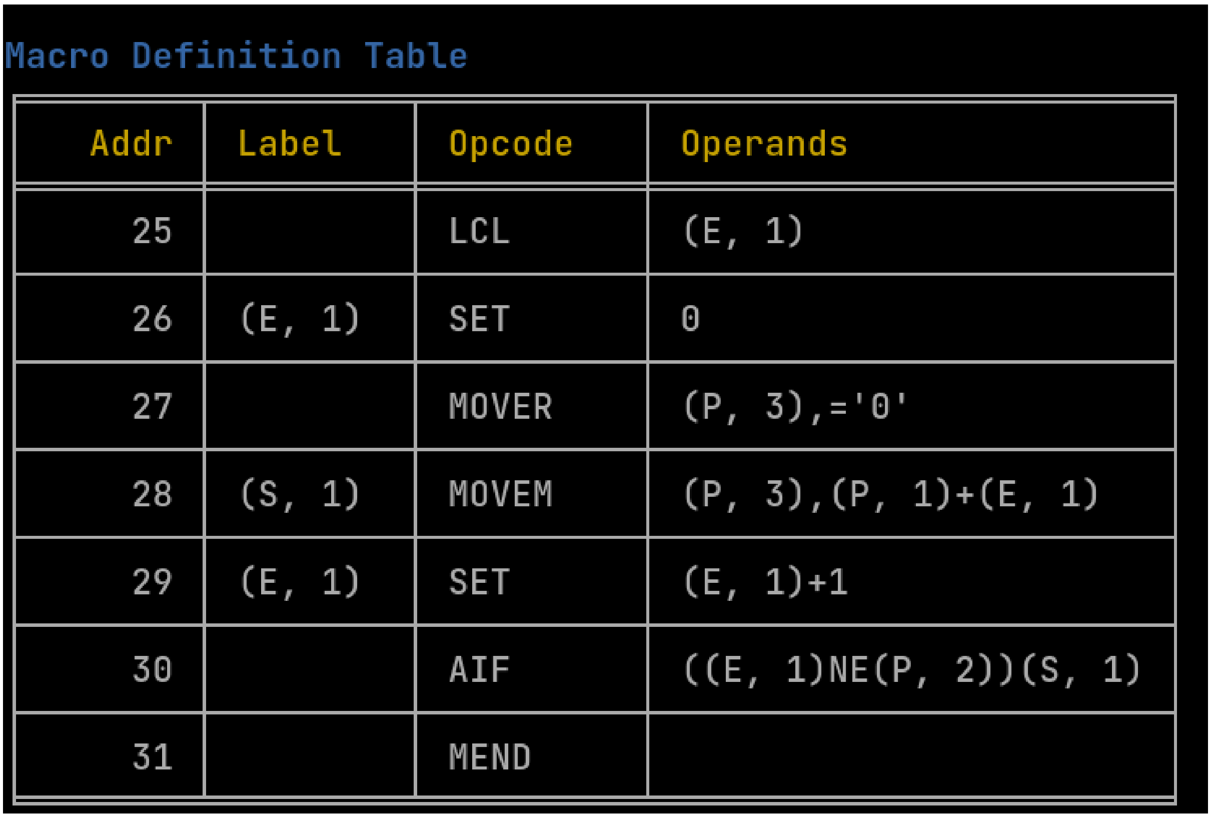
```
      MACRO
      CLEARMEM &X, &N, &REG=AREG
      LCL &M
&M    SET 0
      MOVER &REG, ='0'
.MORE MOVEM &REG, &X + &M
&M    SET &M + 1
      AIF ( &M NE &N ) .MORE
      MEND
```

**index.py:**

```
import sys
import re
from termcolor import colored
from tabulate import tabulate

if len(sys.argv) != 2:
    print("Usage: python index.py <file-name>")
    exit(1)

file_name = sys.argv[1]

ssn_tab = []
evn_tab = []
pn_tab = []
mdt = []  # label, opcode, operands

address = int(input("Enter starting address: "))

with open(file_name) as f:
    for num, line in enumerate(f, 1):
        tokens = line.split()
        is_label_def = not bool(re.match(r'\s', line))
        if num != 1 and num != 2 and is_label_def:
            label = tokens[0]
            if label[0] == '&' and label not in evn_tab:
                evn_tab.append(label)
            elif label[0] == '.' and label not in ssn_tab:
                ssn_tab.append(label)

with open(file_name) as f:
    for num, line in enumerate(f, 1):
        address += 1
        tokens = line.split()
        is_label_def = not bool(re.match(r'\s', line))
        if num == 1:
            print(line)
        elif num == 2:
            pn_tab = [x.split(',')[0].split('=')[0] for x in tokens[1:]]
            print(line)
        else:
            # fill mdt:
            row = ['', '', '']
            if is_label_def:
                label = tokens[0]
                tokens = tokens[1:]
                if label in evn_tab:
                    row[0] += f"(E, {evn_tab.index(label) + 1})"
                elif label in ssn_tab:
                    row[0] += f"(S, {ssn_tab.index(label) + 1})"

            row[1] = tokens[0]
            tokens = tokens[1:]

            res = ''
            for token in tokens:
                temp = token.split(',')[0]
                has_comma = (temp != token)
                token = temp
                if token in evn_tab:
                    res += f"(E, {evn_tab.index(token) + 1})"
                elif token in ssn_tab:
                    res += f"(S, {ssn_tab.index(token) + 1})"
                elif token in pn_tab:
                    res += f"(P, {pn_tab.index(token) + 1})"
                else:
                    res += token

                if has_comma:
                    res += ','

            row[2] = res
            row.insert(0, address)
            mdt.append(row)


print(tabulate([[x] for x in pn_tab], headers=[
      colored("PN Table", color="yellow")], tablefmt="fancy_grid"))

print(tabulate([[x] for x in evn_tab], headers=[
    colored("EVN Table", color="yellow")], tablefmt="fancy_grid"))

print(tabulate([[x] for x in ssn_tab], headers=[
    colored("SSN Table", color="yellow")], tablefmt="fancy_grid"))

print()
print(colored("Macro Definition Table", attrs=["bold"], color="blue"))
print(tabulate(mdt, headers=[colored("Addr", color="yellow"),
      colored("Label", color="yellow"), colored("Opcode",
color="yellow"), colored("Operands", color="yellow")],
tablefmt="fancy_grid"))
```



```
Enter starting address: 22
      MACRO

      CLEARMEM &X, &N, &REG=AREG

╒════════════╕
│ PN  Table  │
╞════════════╡
│ &X         │
├────────────┤
│ &N         │
├────────────┤
│ &REG       │
╘════════════╛
╒═════════════╕
│ EVN  Table  │
╞═════════════╡
│ &M          │
╘═════════════╛
╒═════════════╕
│ SSN  Table  │
╞═════════════╡
│ .MORE       │
╘═════════════╛
```



```
Macro Definition Table
╒══════╤════════╤════════╤═════════════════════════╕
│ Addr │ Label  │ Opcode │ Operands                │
╞══════╪════════╪════════╪═════════════════════════╡
│   25 │        │ LCL    │ (E, 1)                  │
├──────┼────────┼────────┼─────────────────────────┤
│   26 │ (E, 1) │ SET    │ 0                       │
├──────┼────────┼────────┼─────────────────────────┤
│   27 │        │ MOVER  │ (P, 3),='0'             │
├──────┼────────┼────────┼─────────────────────────┤
│   28 │ (S, 1) │ MOVEM  │ (P, 3),(P, 1)+(E, 1)    │
├──────┼────────┼────────┼─────────────────────────┤
│   29 │ (E, 1) │ SET    │ (E, 1)+1                │
├──────┼────────┼────────┼─────────────────────────┤
│   30 │        │ AIF    │ ((E, 1)NE(P, 2))(S, 1)  │
├──────┼────────┼────────┼─────────────────────────┤
│   31 │        │ MEND   │                         │
╘══════╧════════╧════════╧═════════════════════════╛
```