**EXP NO:5(a) DESIGN OF MASM MACRO PROCESSOR**

1. **MACRO DEFENITION**

**AIM:**

To write a C program to implement the macro definition in the design of MASM macro processor.

**ALGORITHM:**

**STEP 1:** SSNTAB\_ptr :=1;

PNTAB\_ptr :=1;

**STEP 2: Process the macro prototype statement and form the MNT entry**

**(a)** name: =macro name;

**(b)** For each positional parameter

1. Enter parameter name in PNTAB[PNTAB\_ptr].
2. PNTAB\_ptr: =PNTAB\_ptr + 1;
3. #PP: = #PP + 1;

**(c)**KPDTP: = KPDTAB\_ptr;

**(d)**For each keyword parameter

1. Enter parameter name and default value of the KPDTAB[KPDTAB\_ptr].
2. Enter parameter name in PNTAB[PNTAB\_ptr];
3. KPDTAB\_ptr: = KPDTAB\_ptr + 1;
4. PNTAB\_ptr: = PNTAB\_ptr + 1;
5. #KP: = #KP +1;

**(e)** MDTP: = MDT\_ptr;

**(f)**#EV: = 0;

**(g)**SSTP: = SSTAB\_ptr;

**STEP 3: While not a MEND statement**

1. If an LCL statement then
2. Enter expansion time variable name in EVNTAB.
3. #EV: = # EV + 1;

**(b)**If model statement then

1. If label field contains a sequencing symbol then

If symbol is present in SSNTAB then

q: = entry number in SSNTAB;

else

enter symbol in SSNTAB [SSNTAB\_ptr];

q: = SSNTAB\_ptr;

SSNTAB\_ptr: = SSNTAB\_ptr +1;

SSTAB[SSTAB+q-1]: =MDT\_ptr;

1. For a parameter, generate the specification (P, #A).
2. For an expansion variable, generate the specification (E, #m)
3. Record the LC in MDT [MDT\_ptr];
4. MDT\_ptr: = MDT\_ptr+ 1;

**(c)**if a preprocessor statement then

1. If a SET statement

Search each expansion time variable name used in the statement.

EVNTAB and generate the spec (E, #m).

1. If an AIF or AGO statement then

If sequencing symbol used in the statement is present in SSNTAB then

q: = entry number in SSNTAB;

q: =SSNTAB;

Replace the symbol by (S, SSTP+q-1).

1. Record the LC in NOT[NOT\_ptr].
2. MDT\_ptr: = MDT\_ptr +1;

**STEP 4: (MEND Statement)**

If SSNTAB\_ptr: =1(i.e., SSNTAB is empty) then

SSTP! = 0;

Else

SSTAB\_ptr: = SSTAB\_ptr+SSNTAB\_ptr -1;

If #KP =0 then KPDTP =0;

**CODE:**

#include<stdio.h>

#include<string.h>

char pntab [4][5], evtab [3][3], kpdtab [3][5], ssntab [2][5];

int pn, pn1, ev, ssn;

int checkpn (char var [])

{

int j;

for(j=1;j<=5;j++)

{

// printf("%s\n",var);

if(strcmp(pntab[j],var)==0)

return j;

}

return 0;

}

int checkev(char var[])

{

int j;

for(j=1;j<=3;j++)

{

strtok(var,"\n");

//printf("%s\n",var);

if(strcmp(evtab[j],var)==0)

return j;

}

return 0;

}

int checkssn(char var[])

{

int j;

for(j=1;j<=3;j++)

{

strtok(var,"\n");

//printf("%s\n",var);

if(strcmp(ssntab[j],var)==0)

return j;

}

return 0;

}

main()

{

char var[256],var1[7][10],mnt[10];

char delim[2]=" ";

char \*ptr;

int i,m=1,n=1,pp=0,kp=0,ev2=0,mdtp,kpdtp,sstp,l=0;

FILE \*f1,\*f2,\*f3,\*f4;

f1=fopen("macro def.txt","r");

f2=fopen("macro ip.txt","w");

f3=fopen("macro ip2.txt","w");

f4=fopen("kpdtab.txt","w");

while(!feof(f1))

{

fgets(var,sizeof(var),f1);

ptr=strtok(var,delim);

i=0;

while(ptr!=NULL)

{

strcpy(\*(var1+i),ptr);

i++;

ptr=strtok(NULL,delim);

}

if(strcmp(var1+0,"MACRO\n")==0)

{

}

else if(strcmp(var1+0,"LCL")==0)

{

m=1;

ev2++;

strtok(var1+1,"\n");

ptr=strtok(var1+1,"&");

strcpy(\*(evtab+m),ptr);

ev=checkev(ptr);

printf("%s E,%d\n",var1+0,ev);

fprintf(f2,"%s E,%d\n",var1+0,ev);

l++;

mdtp=l;

}

else if(strcmp(var1+1,"SET")==0)

{

ptr=strtok(var1+0,"&");

ev=checkev(ptr);

//printf("%s\n",ptr);

if(strcmp(var1+2,"0\n")==0)

{

printf("E,%d SET 0\n",ev,var1+0);

fprintf(f2,"E,%d SET 0\n",ev,var1+0);

}

else

{

printf("E,%d %s E,%d %s %s",ev,var1+1,ev,var1+3,var1+4);

fprintf(f2,"E,%d %s E,%d %s %s",ev,var1+1,ev,var1+3,var1+4);

}

l++;

}

else if(strcmp(var1+0,"MOVER")==0)

{

ptr=strtok(var1+1,"&");

pn=checkpn(ptr);

ptr=strtok(var1+3,"&");

ev=checkev(ptr);

printf("%s P,%d , E,%d\n",var1+0,pn,ev);

fprintf(f2,"%s P,%d , E,%d\n",var1+0,pn,ev);

l++;

}

else if(strcmp(var1+1,"MOVEM")==0)

{

m=1;

ptr=strtok(var1+0,".");

strcpy(\*(ssntab+m),ptr);

ptr=strtok(var1+2,"&");

pn=checkpn(ptr);

ptr=strtok(var1+4,"&");

pn1=checkpn(ptr);

ptr=strtok(var1+6,"&");

ev=checkev(ptr);

printf("%s P,%d , P,%d %s E,%d\n",var1+1,pn,pn1,var1+5,ev);

fprintf(f2,"%s P,%d , P,%d %s E,%d\n",var1+1,pn,pn1,var1+5,ev);

l++;

sstp=l;

}

else if(strcmp(var1+0,"AIF")==0)

{

ptr=strtok(var1+1,"&");

ev=checkev(ptr);

ptr=strtok(var1+3,"&");

pn=checkpn(ptr);

strtok(var1+4,"\n");

ptr=strtok(var1+4,".");

ssn=checkssn(ptr);

printf("%s E,%d %s P,%d S,%d\n",var1+0,ev,var1+2,pn,ssn);

fprintf(f2,"%s E,%d %s P,%d S,%d\n",var1+0,ev,var1+2,pn,ssn);

l++;

}

else if(strcmp(var1+0,"MEND")==0)

{

printf("%s\n",var1+0);

fprintf(f2,"%s",var1+0);

l++;

}

else if(strcmp(var1+0,"MACRO")!=0)

{

strcpy(mnt,var1+0);

for(i=1;i<7;i=i+1)

{

if(strcmp(var1+i,",")==0)

continue;

if(strcmp(var1+i,"=")==0)

{

ptr=strtok(var1+(i-1),"&");

strcpy(\*(kpdtab+n),ptr);

//printf("%s\n",kpdtab+n);

n++;

strcpy(\*(kpdtab+n),var1+(i+1));

kp++;

kpdtp=kp;

//printf("%s\n",kpdtab+n);

continue;

}

ptr=strtok(var1+i,"&");

strcpy(\*(pntab+m),ptr);

//

m++;

pp++;

}

pp=pp-kp;

//printf("%d %d\n",pp,kp);

}

}

printf("pp:%d kp:%d ev:%d kpdtp:%d sstp:%d\n",pp,kp,ev,mdtp,sstp);

fprintf(f3,"%s 00\npp %d\nkp %d\nev %d\nmdtp %d\nkpdtp %d\nsstp %d\n",mnt,pp,kp,ev,mdtp,kpdtp,sstp);

fprintf(f4,"%s %s",kpdtab[1],kpdtab[2]);

}

**INPUT:**

MACRO

CLEARMEM &X , &N , &REG = AREG

LCL &M

&M SET 0

MOVER &REG , &M

.MORE MOVEM &REG , &X + &M

&M SET &M + 1

AIF &M NE &N .MORE

MEND

**OUTPUT:**

LCL (E,1)

(E,1) SET 0

MOVER (P,3) = 0

MOVEM (P,3) (P,1) + (E,1)

(E,1) SET (E,1) + 1

AIF (E,1) NE (P,2) (S,1)

**RESULT:**

Thus, the given C program on the MACRO definition in the execution of MASM macro processor has been executed and obtained the required result.

**EXP NO :5(b) DESIGN OF MASM MACRO PROCESSOR**

1. **MACRO EXPANSION**

**AIM:**

To write a C program to implement the macro expansion in the design of MASM macro processor.

**ALGORITHM:**

**STEP 1:** Performance initialization for the expansion of macro

**(a)**MEC := MDTP field of the MNT entry

(**b)**Create EVTAB with #EV entries and set EVTAB\_ptr;

**(c)**Create APTAB with #PP + # KP entries

**(d)**Copy keyword parameter defaults from the entries KPDTAB[KPDTAB]… KPDTAB[KPDTAB +#KP -1] into APTAB[#PP +1 ]…APTAB[#PP +# KP]

**(e)**Process positional parameter**s** in the actual parameter list and copy them into APTAB[1]…APTAB[#PP].

(**f)**For keyword parameters in the actual parameter list

Search the keyword name in the parameter name field in KPDTAB[KPDTP]…[KPDTP+#KP-1], Let KPDTAB[q] contains a matching entry.Enter value of the keyword parameter in the call (if any) in APTAB[#PP + q – KPDTP + 1].

**STEP 2:** While statement pointed by MEC is not MEND statement

**(a)**If a model statement then

1. Replace operands of the form (P,#n) and (E,#m) by values in APTAB[n] and EVTAB[m] respectively.
2. Output the generated statement.
3. MEC = MEC +1 ;

**(b)**If a SET statement with a specification ( E,#m ) in the label field then

1. Evaluate the expression in the operand field and set an appropriate value in EVTAB[m].
2. MEC:= MEC+1;

**(c)**If an AGO statement with (S,#S) in operation field then MEC:=SSTAB[SSTAB+S-1];

**(d)**If an AIF statement with (S,#S) in operand field then ,

If condition in the AIF statement is true then MEC:=SSTAB[SSTP +S-1];

**STEP 3:** Exit of MACRO expansion.

**CODE:**

#include<stdio.h>

#include<string.h>

char str1[10][7],str2[10][7];

int checkssn(char var[])

{

int j;

for(j=1;j<=9;j++)

{

if(strcmp(str1[j],var)==0)

{

strtok(str2[j],"\n");

j=atoi(str2[j]);

return j;

}

}

return 0;

}

int main()

{

char ssntab[4][10],var[10][256],var1[4][10],str[10][50],mnt1[10],mnt2[8],mnt3[10];

char kp[3][5];

char delim[2]=" ";

char \*ptr;

int sstab[5],mec[50];

int i,j=1,m=0,n=0,x=0,l=1,flag=0,flag1=0,ssn,end=0,b;

FILE \*f1,\*f2,\*f3,\*f4;

f1=fopen("macro ip.txt","r");

f2=fopen("macro ip2.txt","r");

f3=fopen("kpdtab.txt","r");

f4=fopen("macro call.txt","r");

fscanf(f4,"%s%s%d",mnt1,mnt2,&b);

for(l=1;l<9;l++)

{

fscanf(f2,"%s%s",str1[l],str2[l]);

}

for(l=1;l<2;l++)

{

fscanf(f3,"%s%s",kp[l],kp[l+1]);

//printf("%s\t%s\n",kp[l],kp[l+1]);

}

if(strcmp(str1[1],mnt1)!=0)

{

while(!feof(f1))

{

static int j=1;

fgets(var,sizeof(var),f1);

strcpy(str[j],var);

j++;

}

j=1;

while(j<8)

{

i=0;

strcpy(var,str[j]);

ptr=strtok(var,delim);

while(ptr!=NULL)

{

strcpy(\*(var1+i),ptr);

i++;

ptr=strtok(NULL,delim);

}

if(strcmp(var1+0,"LCL")==0)

{

mec[x]=1;

x++;

}

if(strcmp(var1+1,"SET")==0)

{

mec[x]=2;

if(strcmp(var1+2,"0")==0)

m=0;

else if(strcmp(var1+4,"1\n")==0)

{

mec[x]=5;

m=m+1;

//printf("%d\n",m);

}

x++;

}

if(strcmp(var1+0,"MOVER")==0)

{

mec[x]=3;

x++;

ptr=strtok(var1+1,",");

ptr=strtok(NULL," ");

if(strcmp(ptr,"3")==0)

{

printf("+MOVER %s,%d\n",kp[2],m);

}

}

if(strcmp(var1+0,"MOVEM")==0)

{

mec[x]=4;

x++;

ptr=strtok(var1+1,",");

ptr=strtok(NULL," ");

if(strcmp(ptr,"3")==0)

{

ptr=strtok(var1+3,",");

ptr=strtok(NULL," ");

if(strcmp(ptr,"1")==0)

{

ptr=strtok(var1+5,",");

if(strcmp(ptr,"E")==0)

printf("+MOVEM %s,%s+%d\n",kp[2],mnt2,m);

}

}

}

if(strcmp(var1+0,"AIF")==0)

{

mec[x]=6;

x++;

ptr=strtok(var1+3,",");

ptr=strtok(NULL," ");

if(strcmp(ptr,"2")==0)

{

if(m<b)

{

end=1;

l++;

//break;

}

else

{

end=0;

}

}

}

//printf("%d\n",end);

if(end==0)

j++;

else if(end==1)

{

ssn=checkssn("sstp");

j=ssn;

end=0;

}

}

}

else

return 0;

//printf("%s",var[1]);

}

**INPUT:**

LCL (E,1)

(E,1) SET 0

MOVER (P,3) = ‘0’

MOVEM (P,3) (P,1) + (E,1)

(E,1) SET (E,1) + 1

AIF (E,1) NE (P,2) (S,1)

MEND

**Function call:** CLEARMEM AREA, 10, REG= BREG

**OUTPUT:**

+MOVER BREG = 0

+MOVER BREG AREA + 0

+MOVER BREG AREA + 1

+MOVER BREG AREA + 2

+MOVER BREG AREA + 3

+MOVER BREG AREA + 4

+MOVER BREG AREA + 5

+MOVER BREG AREA + 6

+MOVER BREG AREA + 7

+MOVER BREG AREA + 8

+MOVER BREG AREA + 9

APTAB:

AREA 10 BREG

EVTAB:

10

**RESULT:**

Thus, the given C program on the MACRO expansion in the execution of MASM macro processor has been executed and obtained the required result.