**EXPERIMENT NO 1**

8 BIT ADDITION

LDA 8500

MOV B,A

LDA 8501

ADD B

STA 8502

RST 1

**EXPERIMENT NO 2**

8 – BIT SUBTRACTION

LDA 8500

MOV B,A

LDA 8501

SUB B

STA 8502

RST 1

**EXPERIMENT NO 3**

8- BIT MULTIPLICATION

LDA 8500

MOV B,A

LDA 8501

MOV C,A

CPI 00

JZ LOOP

XRA A

LOOP1: ADD B

DCR C

JZ LOOP

JMP LOOP 1

LOOP:STA 8502

RST 1

**EXPERIMENT NO 4**

8-BIT DIVISION

LDA 8501

MOV B,A

LDA 8500

MVI C,00

LOOP:CMP B

JC LOOP

SUB B

INR C

JMP LOOP

STA 8503

MOV A,C

STA 8502

RST 1

**EXPERIMENT NO 5**

16-BIT ADDITION

LDA 8500

MOV B,A

LDA 8501

MOV C,A

LDA 8502

MOV D,A

LDA 8503

ADD C

STA 8505

MOV A,D

ADD B

STA 8504

RST 1

**EXPERIMENT NO 6**

16-BIT SUBTRACTION

LDA 8500

MOV B,A

LDA 8501

MOV C,A

LDA 8502

MOV D,A

LDA 8503

ADD C

STA 8505

MOV A,D

ADD B

STA 8504

RST 1

**EXPERIMENT NO 7**

16-BIT MULTIPLICATION

LHLD 8500

MOV D,H

MOV E,L

LDA 8502

MOV C,A

CPI 00

JZ LOOP1

LXI H,0000

LOOP: DAD D

DCR C

JZ LOOP 1

JMP LOOP

LOOP1:SHLD 8503

HLT

**EXPERIMENT NO 8**

16-BIT DIVISION

MOV DX,0000

MOV AX,0006H

MOV CX,0004H

DIV CX

MOV DI,1300H

MOV[DI],AH

INC DI

MOV[DI],AH

INC DI

MOV[DI],DX

HLT

**EXPERIMENT NO 9**

FACTORIAL OF A NUMBER

LXI H,8500H

MOV B,H

MVI D,01H

CALL MULTIPLY

DCR B

JNZ FACTORIAL

INX H

MOV M,D

HLT

MOV E,B

MVI A,00H

ADD D

DCR E

JNZ MULTIPLY LOOP

MOV D,A

RET

**EXPERIMENT 12**

LXI H,8000 MOV B,M MVI D,01

FACT: CALL MUL DCR B

JNZ FACT INX H MOV M,D HLT

MUL: MOV E,B XRA A

ML: ADD D DCR E

JNZ ML MOV D,A RET

**EXPERIMENT 13:**

LXI H,8000 MOV C,M INX H

MOV B,M DCR C LOOP: INX H MOV A,M CMP B

JC SKIP MOV B,A SKIP: DCR C JNZ LOOP LXI H,8010 MOV M,B HLT

**EXPERIMENT :14**

#include<stdio.h>

int main()

{

int counter =1,a,b,choice,res,ins; printf("Enter number 1:"); scanf("%d",&a);

counter = counter+1; printf("Enter number 2:"); scanf("%d",&b);

counter = counter +1;

printf("1-Addition:\n2-Subtraction:\n3-Multiplication:\n4-Division:");

scanf("%d",&choice);

switch(choice)

{

case 1: printf("Performing addition\n");

res = a+b;

counter = counter+1;

break;

case 2: printf("Performing subtraction\n"); res = a-b;

counter = counter+1;

break;

case 3: printf("Performing Multiplication\n"); res = a\*b;

counter = counter+1;

break;

case 4: printf("Performing Division\n");

res = a/b;

counter = counter+1;

break;

default: printf("Wrong input");

break;

}

printf("The cycle value is:%d\n",counter); printf("Enter the number of instructions:");

**EXPERIMENT 15:**

#include<stdio.h>

int main( )

{

float a,b,counter=1,res,INS;

float performance\_measure;

printf("Enter the number 1: ");

scanf("%f",&a);

printf("Enter the number 2: ");

scanf("%f",&b);

counter =counter+1;

res=a || b;

counter=counter+2;

printf("enter no.of instruction:");

scanf("%f",&INS);

performance\_measure=INS/counter;

printf("performance\_measure:%f ",performance\_measure);

return 0;

}

#include<stdio.h>

int main( )

{

float a,b,counter=1,res,INS;

float performance\_measure;

printf("Enter the number 1: ");

scanf("%f",&a);

printf("Enter the number 2: ");

scanf("%f",&b);

counter =counter+1;

res=a&&b;

counter=counter+2;

printf("enter no.of instruction:");

scanf("%f",&INS);

performance\_measure=INS/counter;

printf("performance\_measure:%f ",performance\_measure);

return 0;

}

**EXPERIMENT:16**

#include<stdio.h>

void main(){

int counter=0;

int input;

int num1,num2;

int op;

int res;

int ins;

int performance\_measure=0;

printf("\n Enter 1st value: ");

scanf("%d",&num1);

counter+=1;

printf("\n Enter the 2nd value: ");

scanf("%d",&num2);

counter+=1;

printf("\n Enter the option: \n1)Addition\n2)Subraction\n3)Multiplication\n4)Division");

scanf("%d",&op);

switch(op){

case 1:

printf("Performing addition operation");

res=num1+num2;

counter+=1;

break;

case 2:

printf("Performing subraction operation");

res=num1-num2;

counter+=1;

break;

case 3:

printf("Performing multiplication operation");

res=num1\*num2;

counter+=1;

break;

case 4:

if(num2==0){

printf("\n Denominator can't be zero");

}

else{

printf("Performing division operation");

res=num1/num2;

counter+=1;

break;

}

default:

printf("Invalid case...");

counter+=3;

break;

}

printf("\n CYCLE VALUE IS : %d",counter);

printf("Enter the no.instruction");

scanf("%d",&ins);

performance\_measure=ins/counter;

printf("\n Performance Measure is: %d",performance\_measure);

}

**EXPERIMENT 17:**

#include <stdio.h>

#include <math.h>

int a = 0,b = 0, c = 0, a1 = 0, b1 = 0, com[5] = { 1, 0, 0, 0, 0};

int anum[5] = {0}, anumcp[5] = {0}, bnum[5] = {0};

int acomp[5] = {0}, bcomp[5] = {0}, pro[5] = {0}, res[5] = {0};

void binary(){

a1 = fabs(a);

b1 = fabs(b);

int r, r2, i, temp;

for (i = 0; i < 5; i++){

r = a1 % 2;

r2 = b1 % 2;

b1 = b1 / 2;

anum[i] = r;

anumcp[i] = r;

bnum[i] = r2;

if(r2 == 0){

bcomp[i] = 1;

}

if(r == 0){

acomp[i] =1;

}

}

c = 0;

for ( i = 0; i < 5; i++){

res[i] = com[i]+ bcomp[i] + c;

if(res[i] >= 2){

c = 1;

}

else

c = 0;

res[i] = res[i] % 2;

}

for (i = 4; i >= 0; i--){

bcomp[i] = res[i];

}

if (a < 0){

c = 0;

for (i = 4; i >= 0; i--){

res[i] = 0;

}

for ( i = 0; i < 5; i++){

res[i] = com[i] + acomp[i] + c;

if (res[i] >= 2){

c = 1;

}

else

c = 0;

res[i] = res[i]%2;

}

for (i = 4; i >= 0; i--){

anum[i] = res[i];

anumcp[i] = res[i];

}

}

if(b < 0){

for (i = 0; i < 5; i++){

temp = bnum[i];

bnum[i] = bcomp[i];

bcomp[i] = temp;

}

}

}

void add(int num[]){

int i;

for ( i = 0; i < 5; i++){

res[i] = pro[i] + num[i] + c;

if (res[i] >= 2){

c = 1;

}

else{

c = 0;

}

res[i] = res[i]%2;

}

for (i = 4; i >= 0; i--){

pro[i] = res[i];

printf("%d",pro[i]);

}

printf(":");

for (i = 4; i >= 0; i--){

printf("%d", anumcp[i]);

}

}

void arshift(){

int temp = pro[4], temp2 = pro[0], i;

for (i = 1; i < 5 ; i++){

pro[i-1] = pro[i];

}

pro[4] = temp;

for (i = 1; i < 5 ; i++){

anumcp[i-1] = anumcp[i];

}

anumcp[4] = temp2;

printf("\nAR-SHIFT: ");

for (i = 4; i >= 0; i--){

printf("%d",pro[i]);

}

printf(":");

for(i = 4; i >= 0; i--){

printf("%d", anumcp[i]);

}

}

void main(){

int i, q = 0;

printf("\t\tBOOTH'S MULTIPLICATION ALGORITHM");

printf("\nEnter two numbers to multiply: ");

printf("\nBoth must be less than 16");

do{

printf("\nEnter A: ");

scanf("%d",&a);

printf("Enter B: ");

scanf("%d", &b);

}while(a >=16 || b >=16);

printf("\nExpected product = %d", a \* b);

binary();

printf("\n\nBinary Equivalents are: ");

printf("\nA = ");

for (i = 4; i >= 0; i--){

printf("%d", anum[i]);

}

printf("\nB = ");

for (i = 4; i >= 0; i--){

printf("%d", bnum[i]);

}

printf("\nB'+ 1 = ");

for (i = 4; i >= 0; i--){

printf("%d", bcomp[i]);

}

printf("\n\n");

for (i = 0;i < 5; i++){

if (anum[i] == q){

printf("\n-->");

arshift();

q = anum[i];

}

else if(anum[i] == 1 && q == 0){

printf("\n-->");

printf("\nSUB B: ");

add(bcomp);*//*

arshift();

q = anum[i];

}

else{

printf("\n-->");

printf("\nADD B: ");

add(bnum);

arshift();

q = anum[i];

}

}

printf("\nProduct is = ");

for (i = 4; i >= 0; i--){

printf("%d", pro[i]);

}

for (i = 4; i >= 0; i--){

printf("%d", anumcp[i]);

}

}

**EXPERIMENT :18**

#include<stdlib.h>

#include<stdio.h>

int acum[100]={0} ;

void add(int acum[],int b[],int n);

int q[100],b[100];

int main()

{

int x,y;

printf("Enter the Number :");

scanf("%d%d",&x,&y);

int i=0;

while(x>0||y>0)

{

if(x>0)

{

q[i]=x%2;

x=x/2;

}

else

{

q[i]=0;

}

if(y>0)

{

b[i]=y%2;

y=y/2;

}

else

{

b[i]=0;

}

i++;

}

int n=i;

int bc[50];

printf("\n");

for(i=0;i<n;i++)

{

if(b[i]==0)

{

bc[i]=1;

}

else

{

bc[i]=0;

}

}

bc[n]=1;

for(i=0;i<=n;i++)

{

if(bc[i]==0)

{

bc[i]=1;

i=n+2;

}

else

{

bc[i]=0;

}

}

int l;

b[n]=0;

int k=n;

int n1=n+n-1;

int j,mi=n-1;

for(i=n;i!=0;i--)

{

for(j=n;j>0;j--)

{

acum[j]=acum[j-1];

}

acum[0]=q[n-1];

for(j=n-1;j>0;j--)

{

q[j]=q[j-1];

}

add(acum,bc,n+1);

if(acum[n]==1)

{

q[0]=0;

add(acum,b,n+1);

}

else

{

q[0]=1;

}

}

printf("\nQuoient : ");

for( l=n-1;l>=0;l--)

{

printf("%d",q[l]);

}

printf("\nRemainder : ");

for( l=n;l>=0;l--)

{

printf("%d",acum[l]);

}

return 0;

}

void add(int acum[],int bo[],int n)

{

int i=0,temp=0,sum=0;

for(i=0;i<n;i++)

{

sum=0;

sum=acum[i]+bo[i]+temp;

if(sum==0)

{

acum[i]=0;

temp=0;

}

else if (sum==2)

{

acum[i]=0;

temp=1;

}

else if(sum==1)

{

acum[i]=1;

temp=0;

}

else if(sum==3)

{

acum[i]=1;

temp=1;

}

}

}

**EXPERIMENT :20**

LDA 8000

CMA

STA 8010

INR A

STA 8011

HLT

EXPERIMENT:21

1. #include<stdio.h>

2. #include<stdlib.h>

3. int main(){

4. int a[10],n,i;

5. system ("cls");

6. printf("Enter the number to convert: ");

7. scanf("%d",&n);

8. for(i=0;n>0;i++)

9. {

10. a[i]=n%2;

11. n=n/2;

12. }

13. printf("\nBinary of Given Number is=");

14. for(i=i-1;i>=0;i--)

15. {

16. printf("%d",a[i]);

17. }

18. return 0;

19. }

**EXPERIMENT :22**

#include <stdio.h>

int main()

{

long decimalnum, remainder, quotient,octalnum=0;

int octalNumber[100], i = 1, j;

printf("Enter the decimal number: ");

scanf("%ld", &decimalnum);

quotient = decimalnum;

while (quotient != 0)

{

octalNumber[i++] = quotient % 8;

quotient = quotient / 8;

}

for (j = i - 1; j > 0; j--)

octalnum = octalnum\*10 + octalNumber[j];

printf("Equivalent octal value of decimal no %d is: %d ", decimalnum,octalnum);

return 0;

}

**EXPERIMENT:23**

#include <stdio.h>

#include <math.h>

int convert(long long);

int main() {

long long n;

printf("Enter a binary number: ");

scanf("%lld", &n);

printf("%lld in binary = %d in decimal", n, convert(n));

return 0;

}

int convert(long long n) {

int dec = 0, i = 0, rem;

while (n!=0) {

rem = n % 10;

n /= 10;

dec += rem \* pow(2, i);

++i;

}

return dec;

}

**EXPERIMENT:24**

#include<stdio.h>

int main()

{

float cr;

int p,p1,i;

float cpu[5];

float cpi,ct,max;

int n=1000;

for(i=0;i<=4;i++)

{

Cpu[5]=0;

}

Printf(“\n enter the number of processors:”);

Scanf(“%d”,&p);

P1=p;

for(i=0;i<p;i++)

{

printf(“\n enter the cycles per instruction of processor:”);

scanf(“%f”,&cpi);

printf(“\n enter the clockrate in GHZ:”);

scanf(“%f”,cr);

ct=1000\*cpi/cr;

printf(“the cpu time is:%f”,ct);

cpu[i]=ct;

}

max=cpu[0];

//printf(“%f”,max);

for(i=0;i<p1;i++)

{

if(cpu[i]<=max)

max=cpu[i];

}

Printf(“\n the processor has lowest execution time is:%f”,max);

return 0;

}

**EXPERIMENT:25**

LDA 0000H

MOV B,A

LDA 0001H

STA 0000H

MOV A,B

STA 0001H

HLT