1. converting decimal to binary using C

#include<stdio.h>

#include<stdlib.h>

int main(){

int a[10],n,i;

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

scanf("%d",&n);

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

{

a[i]=n%2;

n=n/2;

}

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

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

{

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

}

return 0;

2. convert decimal to octal

#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;

//Storing remainders until number is equal to zero

while (quotient != 0)

{

octalNumber[i++] = quotient % 8;

quotient = quotient / 8;

}

//Converting stored remainder values in corresponding octal number

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;

}

3. convert decimal to Hexadecimal

#include<stdio.h>

int main() {

long int decimalNumber,remainder,quotient;

int i=1,j,temp;

char hexadecimalNumber[100];

printf("Enter any decimal number: ");

scanf("%ld",&decimalNumber);

quotient = decimalNumber;

while(quotient!=0) {

temp = quotient % 16;

//To convert integer into character

if( temp < 10)

temp =temp + 48; else

temp = temp + 55;

hexadecimalNumber[i++]= temp;

quotient = quotient / 16;

}

printf("Equivalent hexadecimal value of decimal number %d: ",decimalNumber);

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

printf("%c",hexadecimalNumber[j]);

return 0;

}

4. Convert Binary to Decimal

#include <stdio.h>

void main()

{

// declaration of variables

int num, binary\_num, decimal\_num = 0, base = 1, rem;

printf (" Enter a binary number with the combination of 0s and 1s \n");

scanf (" %d", &num); // accept the binary number (0s and 1s)

binary\_num = num; // assign the binary number to the binary\_num variable

while ( num > 0)

{

rem = num % 10; /\* divide the binary number by 10 and store the remainder in rem variable. \*/

decimal\_num = decimal\_num + rem \* base;

num = num / 10; // divide the number with quotient

base = base \* 2;

}

printf ( " The binary number is %d \t", binary\_num); // print the binary number

printf (" \n The decimal number is %d \t", decimal\_num); // print the decimal

}

5. Convert Octal to Binary

#include<stdio.h>

#include<math.h>

int OctalToDecimal(int n) {

int p = 0, decimal = 0, r;

while(n>0){

// retrieving the right-most digit of n

r = n % 10;

// dividing n by 10 to remove the

// right-most digits since it is already

// scanned in previous step

n = n / 10;

decimal = decimal + r \* pow( 8 , p );

++p;

}

return decimal;

}

int main() {

int n, i, k;

printf("Enter Octal: ");

scanf("%d", &n);

printf("\nDecimal of Octal Number %d is : %d", n, OctalToDecimal(n));

return 0;

}

6. Convert Hexadecimal to Decimal

#include <stdio.h>

#include <math.h>

#include <string.h>

int main()

{

char hex[17];

long long decimal, place;

int i = 0, val, len;

decimal = 0;

place = 1;

/\* Input hexadecimal number from user \*/

printf("Enter any hexadecimal number: ");

gets(hex);

/\* Find the length of total number of hex digit \*/

len = strlen(hex);

len--;

/\*

\* Iterate over each hex digit

\*/

for(i=0; hex[i]!='\0'; i++)

{

/\* Find the decimal representation of hex[i] \*/

if(hex[i]>='0' && hex[i]<='9')

{

val = hex[i] - 48;

}

else if(hex[i]>='a' && hex[i]<='f')

{

val = hex[i] - 97 + 10;

}

else if(hex[i]>='A' && hex[i]<='F')

{

val = hex[i] - 65 + 10;

}

decimal += val \* pow(16, len);

len--;

}

printf("Hexadecimal number = %s\n", hex);

printf("Decimal number = %lld", decimal);

return 0;

}

**CPU PERFORMANCE**

#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 Instrcution 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;

}

Enter the number of processors:3

Enter the Cycles per Instrcution of processor:1.5

Enter the clockrate in GHz:3

The CPU time is: 500.000000

Enter the Cycles per Instrcution of processor:1

Enter the clockrate in GHz:2.2

The CPU time is: 454.545441

Enter the Cycles per Instrcution of processor:2

Enter the clockrate in GHz:4

The CPU time is: 500.000000

The processor has lowest Execution time is: 454.545441

**PIPELINING:**

**// Two stage pipeline**

**#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:");**

**scanf("%d",&ins);**

**int performance\_measure = ins/counter;**

**printf("The performance measure is:%d\n",performance\_measure);**

**return 0;**

**}**

**4) // 4 stage pipeline**

**#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);**

**}**

**BOOTH ALGOTITHM:**

#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;

a1 = 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;

}

}

//part for two's complementing

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];

}

//in case of negative inputs

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;

c = 0;

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(){//for arithmetic shift right

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

for (i = 1; i < 5 ; i++){//shift the MSB of product

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

}

pro[4] = temp;

for (i = 1; i < 5 ; i++){//shift the LSB of product

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

}

anumcp[4] = temp2;

printf("\nAR-SHIFT: ");//display together

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");

//simulating for two numbers each below 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){//just shift for 00 or 11

printf("\n-->");

arshift();

q = anum[i];

}

else if(anum[i] == 1 && q == 0){//subtract and shift for 10

printf("\n-->");

printf("\nSUB B: ");

add(bcomp);//add two's complement to implement subtraction

arshift();

q = anum[i];

}

else{//add ans shift for 01

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]);

}

}

BOOTH ALGORITHM

#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;

}

}

}

INTEGER RESTORATION DIVISION

Enter the Number :8

3

Quoient : 0010

Remainder : 00010

#include <stdio.h>

void printBinary(int n, int i)

{

int k;

for (k = i - 1; k >= 0; k--) {

if ((n >> k) & 1)

printf("1");

else

printf("0");

}

}

typedef union {

float f;

struct

{

unsigned int mantissa : 23;

unsigned int exponent : 8;

unsigned int sign : 1;

} raw;

} myfloat;

void printIEEE(myfloat var)

{

printf("%d | ", var.raw.sign);

printBinary(var.raw.exponent, 8);

printf(" | ");

printBinary(var.raw.mantissa, 23);

printf("\n");

}

int main()

{

myfloat var;

var.f = 1259.125;

printf("IEEE 754 representation of %f is : \n",

var.f);

printIEEE(var);

return 0;

}

IEEE 754 representation of 1259.125000 is :

0 | 10001001 | 00111010110010000000000

SINGLE PRECISION FORMAT