# Project 3 Report

Name:Zhexin Zhang

Student ID: ipk693

1. Program list:

|  |
| --- |
| #include<iostream>  #include<cstdlib>  #include<ctime>  #include<vector>  using namespace std;  int CCA(int operandSize,int numA[33],int numB[33])  {  int realoperandSize=operandSize+1;  int carryIn1[realoperandSize], carryIn0[realoperandSize], carryOut0[realoperandSize], carryOut1[realoperandSize],sum[33];  int i, j, cycle=0,temp1,delayCCA;  bool done[realoperandSize], CC=false;  delayCCA=0;  for (j = 0; j < realoperandSize; j++)/\*generate the two operands and initialize other outputs to zero\*/  {  carryIn0[j] = 0;  carryIn1[j] = 0;  carryOut0[j] = 0;  carryOut1[j] = 0;  sum[j]=0;    }    /\*at the beginning of each test set,carryin0 on the first bit is always 1,that means the first bit has no carry in.\*/  carryIn0[0] = 1;  carryIn1[0] = 0;  while (CC == false)//start the simulation  {  if (cycle != 0)  {  CC = true;  temp1 = 1;  while ((CC == true)&&(temp1<realoperandSize))/\*every 2d delay,check if the carries has been completed\*/  {  for (j = 0; j<realoperandSize; j++)//set all the done for each bits in this cycle  {  done[j] = ((carryOut0[j] || carryOut1[j]) ? true : false);  }  for (j = 0; j<realoperandSize; j++)//if one bit has not been finalized,set CC to false,and break the loop  {  if (done[j] == false)  {  CC = false;  break;  }  }  temp1++;  }  }    if (CC == true) break;  /\*generate new sum,carryout0,carryout1 in this cycle\*/  for (j = 0; j<realoperandSize; j++)  {  carryOut1[j] = ((numA[j] && numB[j]) || (carryIn1[j] && (numA[j] ^ numB[j])));  carryOut0[j] = ((!numA[j] && !numB[j]) || (carryIn0[j] && (numA[j] ^ numB[j])));  sum[j] = numA[j] ^ numB[j] ^ carryIn1[j];  }    for (j = 1; j<realoperandSize; j++)  {  carryIn0[j] = carryOut0[j - 1];  carryIn1[j] = carryOut1[j - 1];  }  cycle++;//need 1 more cycle  }  delayCCA=cycle \* 2 + 2;  //cout<<"delayCCA ="<<delayCCA<<endl;  //cout<<"sum=";  for(int k=realoperandSize-1;0<=k;k--)  {  numB[k]=sum[k];  //cout<<numB[k];  }  return delayCCA;    }  int twosComp(int bits,int a[33],int A2[33])  {  int one[33]={0},carrya[33]={0},A[33]={0};  one[0]=1;  int newBits=bits+1;  for(int k=0;k<newBits;k++) A[k]=!a[k];  //cout<<"1's compliment a=";  //for (int k=bits;0<=k;k--) cout<<A[k];  for(int k=0;k<newBits;k++)  {  if(k!=0)  {  carrya[k]=(A[k]\*one[k])||(A[k]\*carrya[k-1])||(carrya[k-1]\*one[k]);  A2[k]=A[k]^one[k]^carrya[k-1];  }  else if (k==0)  {  carrya[k]=A[k]\*one[k];  A2[k]=(A[k]^one[k]^0);  }  }  //cout<<"\n2's compliment a=";  //for (int k=bits;0<=k;k--) cout<<A2[k];  //cout<<"\n";  }  int main()  {  const int bits=32,testsets=10000;  int n=16,N[64]={0},D[32]={0},Q[33]={0},R[32]={0},newD[32]={0},Cp[32]={0},Cn[32]={0};  int i,j,k,cn[32]={0},cp[32]={0};  double count,c[32]={0};  srand((unsigned)time(NULL));  for(;n<=bits;n=n+2)  {  for(k=0;k<testsets;k++)  {  count=0.0;  for(i=0;i<n;i++)  {  //generate N and D  N[i]=rand()%2;  N[i+n]=rand()%2;  D[i]=rand()%2;  Q[i]=0;  R[i]=0;  newD[i]=0;  }  //normalize N and D  N[2\*n-1]=0;  D[n-1]=0;D[n-2]=1;  Cp[n-1]=0;Cp[n-2]=1;//set comparator 01000...  Cn[n-1]=1;Cn[n-2]=1;//set comparator 11000...  twosComp(n,D,newD);//generate 2's compliment of D for subtract  twosComp(n,Cp,cp);  twosComp(n,Cn,cn);  for(int i=n-1;0<=i;i--) R[i]=N[i+n];  for(int i=n;0<=i;i--)  {    if ((R[n-1]==0&&R[n-2]==0))  {    Q[i]=0;    if(i==0)  {    break;  }  for(int j=n-1;0<=j;j--) R[j]=R[j-1];R[0]=N[i-1];    }  else if((R[n-1]==1&&R[n-2]==1))  {    Q[i]=1;    if(i==0)  {    break;  }  for(int j=n-1;0<=j;j--) R[j]=R[j-1];R[0]=N[i-1];    }  else  {    if(R[n-1]==0)  {  CCA(n,newD,R);  Q[i]=1;  }  else  {  CCA(n,D,R);  Q[i]=0;  }    count+=1;  if(i==0)  {  break;  }  for(int j=n-1;0<=j;j--) R[j]=R[j-1];R[0]=N[i-1];    }  }  c[n]=c[n]+count;  }  cout<<n<<"bits,the average efficiency rate="<<(n+1)\*testsets/c[n]<<endl;  }  } |

1. Plot of average number of bits per addition/subtraction cycle:



1. Test result:

