LIDAOQI 5101728 daoqi001

Problem 1:



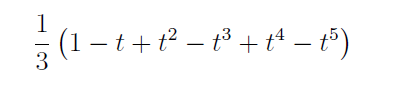
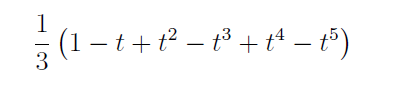




Figure 1. Circuit Diagram



|  |  |  |
| --- | --- | --- |
| X | (targetFunction) | Circuit Output |
| 0 | 0.3333 | 0.3333 |
| 0.25 | 0.2666 | 0.2666 |
| 0.5 | 0.2188 | 0.2187 |
| 0.75 | 0.1566 | 0.1566 |
| 1 | 0 | 0 |

This circuit work for all the given X values.

Code:

function [f] = targetFunction (t)

f=1/3\*(1-t+t.^2-t.^3+t.^4-t.^5);

end

>> X=linspace(0,1,5)

X =

0 0.2500 0.5000 0.7500 1.0000

>> targetFunction(X)

ans =

0.3333 0.2666 0.2188 0.1566 0

function [f] = P1SI(X)

CKT=[0.0000

0.2000

0.2000

0.2333

0.2667

0.3333];

n=length(CKT);

s=length(X);

f=zeros(1,s);

for j=1:s

for i=1:n

f(j)=f(j)+CKT(n+1-i)\*nchoosek(n-1,i-1)\*X(j).^(i-1)\*(1-X(j)).^(n-i);

end

end

end

>> P1SI(X)

ans =

0.3333 0.2666 0.2187 0.1566 0

%circuit design code

function [M] = binmatrix(n)

%UNTITLED2 Summary of this function goes here

% Detailed explanation goes here

M=zeros(n+1,n+1);

MSC=0;

thisRow=zeros(1,n+1);

for i=0:n

MSC=nchoosek(n,n-i);%main stream coeffient

mainstream=1;

for j=1:i

mainstream=conv(mainstream,[1 -1]);

end

mainstream=mainstream\*MSC;

thisRow=[zeros(1:n-i) mainstream];

M(i+1,:)=thisRow;

end

end

>> B5=binmatrix(5)

B5 =

0 0 0 0 0 1

0 0 0 0 5 -5

0 0 0 10 -20 10

0 0 10 -30 30 -10

0 5 -20 30 -20 5

1 -5 10 -10 5 -1

>> f=1/3\*[1 -1 1 -1 1 -1]';

>> CKT=inv(B5)\*f

CKT =

-0.0000

0.2000

0.2000

0.2333

0.2667

0.3333

Problem 2:

Code:

function [M] = convmatrix(n)

%UNTITLED2 Summary of this function goes here

% Detailed explanation goes here

M=zeros(n+1,n+1);

thisRow=zeros(1,n+1);

for i=0:n

mainstream=1;

for j=1:i

mainstream=conv(mainstream,[1 -1]);

end

thisRow=[zeros(1,n-i) mainstream];

M(i+1,:)=thisRow;

end

end

function [] = problem2Print()

%UNTITLED2 Summary of this function goes here

% Detailed explanation goes here

x=[0 0 0 0 1 1 1 1];

y=[0 0 1 1 0 0 1 1];

z=[0 1 0 1 0 1 0 1];

g=zeros(5,8);

g(1,:)=and(x,y);

g(2,:)=xor(x,y);

g(3,:)=and(z,g(2,:));

g(4,:)=or(g(1,:),g(3,:));

g(5,:)=not(xor(z,g(2,:)));

gs1='and(x,y)';

gs2='xor(x,y)';

gs3='and(z,g(2,:))';

gs4='or(g(1,:),g(3,:))';

gs5='xor(z,g(2,:))';

c\_std=or(and(x,y),and(z,xor(x,y)));

s\_std=xor(z,xor(x,y));

c=zeros(32,8);

s=zeros(32,8);

numFaultGates=zeros(1,32);

for i=0:31

rem=i;

thisCase=zeros(1,5);

for j=1:5

thisCase(6-j)=mod(rem,2);

rem=(rem-thisCase(6-j))/2;

end

numFaultGates(i+1)=sum(thisCase);

for j=1:5

thisAct=eval(strcat('gs',num2str(j)));

if thisCase(j)

thisAct=strcat('not(',thisAct,');');

end

g(j,:)=eval(thisAct);

end

c(i+1,:)=g(4,:);

s(i+1,:)=g(5,:);

end

cc=zeros(32,8);

sc=zeros(32,8);

for i=1:32

cc(i,:)=xor(c(i,:),c\_std);

sc(i,:)=xor(s(i,:),s\_std);

end

cF=zeros(6,8);

sF=zeros(6,8);

for i=1:32

for j=1:8

cF(numFaultGates(i)+1,j)=cF(numFaultGates(i)+1,j)+cc(i,j);

sF(numFaultGates(i)+1,j)=sF(numFaultGates(i)+1,j)+sc(i,j);

end

end

C5=convmatrix(5);

rC5=zeros(6);

for i=1:6

rC5(i,:)=C5(7-i,:);

end

rcF=rC5'\*cF;

rsF=rC5'\*sF;

fprintf('x\ty\tz\tPc\n')

for i=1:8

start=1;

fprintf('%d\t%d\t%d\t',x(i),y(i),z(i))

for j=1:6

if rcF(j,i)>0

if start==1

fprintf('%d\*E^%d',rcF(j,i),j-1)

start=0;

else

fprintf('+%d\*E^%d',rcF(j,i),j-1)

end

else

if rcF(j,i)<0

fprintf('%d\*E^%d',rcF(j,i),j-1)

end

end

end

%for j=1:6

% fprintf('%dE^%d+',rsF(j,i),j-1)

%end

fprintf('\n')

end

fprintf('x\ty\tz\tPs\n')

for i=1:8

start=1;

fprintf('%d\t%d\t%d\t',x(i),y(i),z(i))

for j=1:6

if rsF(j,i)>0

if start==1

fprintf('%d\*E^%d',rsF(j,i),j-1)

start=0;

else

fprintf('+%d\*E^%d',rsF(j,i),j-1)

end

else

if rsF(j,i)<0

fprintf('%d\*E^%d',rsF(j,i),j-1)

end

end

end

fprintf('\n')

end

end

Problem 2:

x y z Pc

0 0 0 3\*E^1-5\*E^2+2\*E^3

0 0 1 4\*E^1-10\*E^2+10\*E^3-4\*E^4

0 1 0 3\*E^1-5\*E^2+2\*E^3

0 1 1 3\*E^1-8\*E^2+10\*E^3-4\*E^4

1 0 0 3\*E^1-5\*E^2+2\*E^3

1 0 1 3\*E^1-8\*E^2+10\*E^3-4\*E^4

1 1 0 2\*E^1-3\*E^2+2\*E^3

1 1 1 2\*E^1-4\*E^2+6\*E^3-4\*E^4

x y z Ps

0 0 0 2\*E^1-2\*E^2

0 0 1 2\*E^1-2\*E^2

0 1 0 2\*E^1-2\*E^2

0 1 1 2\*E^1-2\*E^2

1 0 0 2\*E^1-2\*E^2

1 0 1 2\*E^1-2\*E^2

1 1 0 2\*E^1-2\*E^2

1 1 1 2\*E^1-2\*E^2

Problem 3:

function [ ] = syn0405dec(num,den)

tOri=num/den;

action=[];

actionIndex=1;

t=num;

shift=log(den)/log(10);

while 1

if or((t==4),(t==5))

break;

end

if (t/10^shift)>0.5

t=10^shift-t;

action(actionIndex)=0;

actionIndex=actionIndex+1;

else

thisDigit=rem(t,10);

if rem(thisDigit,2)

t=t\*2;

action(actionIndex)=2;

actionIndex=actionIndex+1;

else

if (t\*2.5)<10^shift

t=t\*2.5;

action(actionIndex)=1;

actionIndex=actionIndex+1;

else

t=t\*2;

action(actionIndex)=2;

actionIndex=actionIndex+1;

end

end

end

while 1

if rem(t,10)==0

t=t/10;

shift=shift-1;

else

break;

end

end

end

t=tOri;

nextT=0;

fprintf('t = %1.5f\n',t);

for i=1:actionIndex-1

if action(i)==0

nextT=1-t;

fprintf('%1.4f = 1-%1.4f\n',t,nextT);

elseif action(i)==1

nextT=t\*2.5;

fprintf('%1.4f = 0.4 \* %1.4f\n',t,nextT);

elseif action(i)==2

nextT=t\*2;

fprintf('%1.4f = 0.5 \* %1.4f\n',t,nextT);

end

t=nextT;

end

end

>> syn0405dec(6555,10000)

t = 0.65550

0.6555 = 1-0.3445

0.3445 = 0.5 \* 0.6890

0.6890 = 1-0.3110

0.3110 = 0.5 \* 0.6220

0.6220 = 1-0.3780

0.3780 = 0.4 \* 0.9450

0.9450 = 1-0.0550

0.0550 = 0.5 \* 0.1100

0.1100 = 0.5 \* 0.2200

0.2200 = 0.4 \* 0.5500

0.5500 = 1-0.4500

0.4500 = 0.5 \* 0.9000

0.9000 = 1-0.1000

0.1000 = 0.5 \* 0.2000

0.2000 = 0.4 \* 0.5000

>> syn0405dec(6666,10000)

t = 0.66660

0.6666 = 1-0.3334

0.3334 = 0.4 \* 0.8335

0.8335 = 1-0.1665

0.1665 = 0.5 \* 0.3330

0.3330 = 0.5 \* 0.6660

0.6660 = 1-0.3340

0.3340 = 0.4 \* 0.8350

0.8350 = 1-0.1650

0.1650 = 0.5 \* 0.3300

0.3300 = 0.5 \* 0.6600

0.6600 = 1-0.3400

0.3400 = 0.4 \* 0.8500

0.8500 = 1-0.1500

0.1500 = 0.5 \* 0.3000

0.3000 = 0.5 \* 0.6000

0.6000 = 1-0.4000

>> syn0405dec(1111,10000)

t = 0.11110

0.1111 = 0.5 \* 0.2222

0.2222 = 0.4 \* 0.5555

0.5555 = 1-0.4445

0.4445 = 0.5 \* 0.8890

0.8890 = 1-0.1110

0.1110 = 0.5 \* 0.2220

0.2220 = 0.4 \* 0.5550

0.5550 = 1-0.4450

0.4450 = 0.5 \* 0.8900

0.8900 = 1-0.1100

0.1100 = 0.5 \* 0.2200

0.2200 = 0.4 \* 0.5500

0.5500 = 1-0.4500

0.4500 = 0.5 \* 0.9000

0.9000 = 1-0.1000

0.1000 = 0.5 \* 0.2000

0.2000 = 0.4 \* 0.5000

>>

Problem 3B

>> syn0405(1/4)

t = 0.25000

0.2500 = 0.5 \* 0.5000

>> syn0405(3/4)

t = 0.75000

0.7500 = 1-0.2500

0.2500 = 0.5 \* 0.5000

>> syn0405(5/16)

t = 0.31250

0.3125 = 0.5 \* 0.6250

0.6250 = 1-0.3750

0.3750 = 0.5 \* 0.7500

0.7500 = 1-0.2500

0.2500 = 0.5 \* 0.5000

>> syn0405(11/16)

t = 0.68750

0.6875 = 1-0.3125

0.3125 = 0.5 \* 0.6250

0.6250 = 1-0.3750

0.3750 = 0.5 \* 0.7500

0.7500 = 1-0.2500

0.2500 = 0.5 \* 0.5000

>> syn0405(27/64)

t = 0.42188

0.4219 = 0.5 \* 0.8438

0.8438 = 1-0.1563

0.1563 = 0.5 \* 0.3125

0.3125 = 0.5 \* 0.6250

0.6250 = 1-0.3750

0.3750 = 0.5 \* 0.7500

0.7500 = 1-0.2500

0.2500 = 0.5 \* 0.5000

Problem 3C

Describe a general method for implementing probabilities, given

in binary, starting from 1/2

Suppose the number is N

Step 0: Truncate all the trailing zeros

Step 1: If the number N is larger than 0.5 then do N=1-N;

Step 2: Do N=2\*N;

Step 3: redo Step 1 until N=0.5

Step 4: Backing tracking this path will able to get this probability for 0.5

Suppose we have a number N represent in binary decimal



Figure 2 Digits moving diagram.

If first digit is 1 then it means the number is larger than 0.5 right now, we do N=1-N

At this point, every digit from first digit (include) to last digit (not include) will flip.

First digit will change to 0 last digit will

Afterwards we do N=N\*2 this action will corresponding to a left shift. No digit will move across binary point. And the last digit will move one position towards binary point.

Repeat this step we will get to the point that last digit move to the position right next to binary point, this is N=0.5

Backing tracking this path will able to get this probability for 0.5