**BÁO CÁO THỰC HÀNH LAB 2**

**Môn: Thực hành truyền thông số và dữ liệu**

Họ và tên: Lê Hoàng Nam – MSSV: 21207246

**Câu 1: Phân tích\* đoạn chương trình mẫu ở mục 1.1.3 ?**

|  |  |
| --- | --- |
| clear;clc  bitstream = [0 1 0 0 1 1 0 0 0 1 0];  pulse\_high = 5;  pulse\_low = 0;  for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.001:bit;  if bitstream(bit) == 0  % low level pulse  y = (bt<bit)\*pulse\_low;  else  % high level pulse  y = (bt<bit) \* pulse\_high;  end    try  if bitstream(bit+1) == 1  y(end) = pulse\_high;  end  catch e  % assume next bit is 1  y(end) = pulse\_high;  end    % draw pulse and label  figure(1)  plot(bt, y, 'LineWidth', 2);  text(bit-0.5,pulse\_high+0.5, num2str(bitstream(bit)), 'FontWeight', 'bold');  hold on;  end  % draw grid  grid on;  axis([0 length(bitstream) pulse\_low-1 pulse\_high+1]);  set(gca,'YTick', [pulse\_low pulse\_high])  set(gca,'XTick', 1:length(bitstream))  title('Unipolar Non-Return-to-Zero Level') |  |

**Câu 2: Thay đổi chuỗi dữ liệu toàn bit “0” quan sát và vẽ tín hiệu sau điều biến?**

|  |  |
| --- | --- |
| clear;clc  bitstream = [ 0 0 0 0 0 0 0 0 0 0 0];  pulse\_high = 5;  pulse\_low = 0;  for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.001:bit;  if bitstream(bit) == 0  % low level pulse  y = (bt<bit)\*pulse\_low;  else  % high level pulse  y = (bt<bit) \* pulse\_high;  end    try  if bitstream(bit+1) == 1  y(end) = pulse\_high;  end  catch e  % assume next bit is 1  y(end) = pulse\_high;  end    % draw pulse and label  plot(bt, y, 'LineWidth', 2);  text(bit-0.5,pulse\_high+0.5, num2str(bitstream(bit)), 'FontWeight', 'bold');  hold on;  end  % draw grid  grid on;  axis([0 length(bitstream) pulse\_low-1 pulse\_high+1]);  set(gca,'YTick', [pulse\_low pulse\_high])  set(gca,'XTick', 1:length(bitstream))  title('Unipolar Non-Return-to-Zero Level') |  |

**Câu 3: Thay đổi chuỗi dữ liệu toàn bit “1” quan sát và vẽ tín hiệu sau điều biến?**

|  |  |
| --- | --- |
| clear;clc  bitstream = [ 1 1 1 1 1 1 1 1 1 1];  pulse\_high = 5;  pulse\_low = 0;  for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.001:bit;  if bitstream(bit) == 0  % low level pulse  y = (bt<bit)\*pulse\_low;  else  % high level pulse  y = (bt<bit) \* pulse\_high;  end    try  if bitstream(bit+1) == 1  y(end) = pulse\_high;  end  catch e  % assume next bit is 1  y(end) = pulse\_high;  end    % draw pulse and label  plot(bt, y, 'LineWidth', 2);  text(bit-0.5,pulse\_high+0.5, num2str(bitstream(bit)), 'FontWeight', 'bold');  hold on;  end  % draw grid  grid on;  axis([0 length(bitstream) pulse\_low-1 pulse\_high+1]);  set(gca,'YTick', [pulse\_low pulse\_high])  set(gca,'XTick', 1:length(bitstream))  title('Unipolar Non-Return-to-Zero Level') |  |

**Câu 4: Từ chương trình mẫu 1.1.3, thay đổi “pulse\_low = -5”, quan sát và vẽ tín hiệu sau mã hóa?**

|  |  |
| --- | --- |
| clear;clc  bitstream = [ 0 1 0 0 1 1 0 0 0 1 0];  pulse\_high = 5;  pulse\_low = -5;  for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.001:bit;  if bitstream(bit) == 0  % low level pulse  y = (bt<bit)\*pulse\_low;  else  % high level pulse  y = (bt<bit) \* pulse\_high;  end  try  if bitstream(bit+1) == 1  y(end) = pulse\_high;  else  y(end) = pulse\_low;  end  catch e  % assume next bit is 1  y(end) = pulse\_high;  end  % draw pulse and label  plot(bt, y, 'LineWidth', 2);  text(bit-0.5,pulse\_high+0.5, num2str(bitstream(bit)), 'FontWeight', 'bold');  hold on;  end  % draw grid  grid on;  axis([0 length(bitstream) pulse\_low-1 pulse\_high+1]);  set(gca,'YTick', [pulse\_low pulse\_high])  set(gca,'XTick', 1:length(bitstream))  title('Polar Non-Return-to-Zero Level') |  |

**Câu 5: Phân tích\*\* đoạn chương trình mẫu ở mục 1.1.4?**

|  |  |
| --- | --- |
| clear ; clc  bitstream = randi([0 1], 1, 10000);  fb=100;  pulse\_high = 1;  pulse\_low = -1;  yy=[];  for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.25:(bit-0.25);  if bitstream(bit) == 0  % low level pulse  y = (bt<bit)\*pulse\_low;  else  % high level pulse  y = (bt<bit) \* pulse\_high;  end  yy=[yy y];  end  k=zeros(100,400);  for j=1:1:100  k(j,:)=yy(400\*j-399:400\*j);  sep(j,:)=fft(k(j,:),128);  end  n=size(sep,2);  fs=4\*fb;  f = (0:n-1)\*(fs/n); % frequency range  m\_sep=mean((abs(sep).^2),1)/fs;  figure(1);  plot(f,m\_sep);  % draw grid  grid on;  ylabel('Mean square voltage');  xlabel('Frequency');  title('Polar Non-Return-to-Zero Level'); |  |

**Câu 7: Hoàn thiện chương trình thực hiện mã hóa Bipolar-AMI?**

|  |  |
| --- | --- |
| %% Part 3  clear;clc  bitstream = [ 0 1 0 0 1 1 0 0 0 1 0];  % pulse height  pulse = 5;  % assume that current pulse level is a "low" pulse (binary 1)  % this is the pulse level for the bit before given bitstream  current\_level = -pulse;  for bit = 1:length(bitstream)  % set bit time  bt=bit-1:0.001:bit;  if bitstream(bit) == 0  % binary 0, set to zero  y = (bt<bit)\*0;  else  % each binary 1 has the opposite pulse level from the previous  current\_level = -current\_level;  y = (bt<bit)\*current\_level;  end  % assign last pulse point by inspecting the following bit  try  % we care only about ones as they use alternate levels  if bitstream(bit+1) == 1  y(end) = -current\_level;  end  catch e  % bitstream end; assume next bit is 0  y(end) = -current\_level;  end  % draw pulse and label  plot(bt, y, 'LineWidth', 2);  text(bit-0.5,pulse+0.5, num2str(bitstream(bit)), 'FontWeight', 'bold');  hold on;  end  % draw grid  grid on;  axis([0 length(bitstream) -5 6]);  set(gca,'YTick', [-pulse pulse])  set(gca,'XTick', 1:length(bitstream))  title('Bipolar - AMI') |  |

**Câu 8: Vẽ phổ tín hiệu mã hóa Bipolar-AMI?**

|  |  |
| --- | --- |
| %% Part 4: spectrum of AMI  clear ; clc  bitstream = randi([0 1], 1, 10000);  fb=100;  yy=[];  pulse = 5;  current\_level = -pulse;  for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.25:(bit-0.25);  if bitstream(bit) == 0  % binary 0, set to zero  y = (bt<bit)\*0;  else  % each binary 1 has the opposite pulse level from the previous  current\_level = -current\_level;  y = (bt<bit)\*current\_level;  end  % assign last pulse point by inspecting the following bit  try  % we care only about ones as they use alternate levels  if bitstream(bit+1) == 1  y(end) = -current\_level;  end  catch e  % bitstream end; assume next bit is 0  y(end) = -current\_level;  end  yy=[yy y];  end  k=zeros(100,400);  for j=1:1:100  k(j,:)=yy(400\*j-399:400\*j);  sep(j,:)=fft(k(j,:),128);  end  n=size(sep,2);  fs=4\*fb;  f = (0:n-1)\*(fs/n); % frequency range  m\_sep=mean((abs(sep).^2),1)/fs;  figure(1);  plot(f,m\_sep);  grid on;  xlabel('Frequency')  title('Bipolar - AMI') |  |

**Câu 9: Hoàn thiện chương trình thực hiện mã hóa Manchester?**

|  |  |
| --- | --- |
| %% Manchester  clear;clc  bitstream = [ 0 1 0 0 1 1 0 0 0 1 0];  % pulse height  pulse = 1;  yy=[];  for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.01:(bit);  if bitstream(bit) == 1  % low -> high  y = (bt<bit) \* pulse - 2\*pulse \* (bt < bit - 0.5);  % set last pulse point to high level  current\_level = pulse;  else  % high -> low  y = -(bt<bit) \* pulse + 2\*pulse\*(bt < bit - 0.5);  % set last pulse point to low level  current\_level = -pulse;  end  try  % if the next bit is the same as this one  %change the level  if bitstream(bit+1) == bitstream(bit)  y(end) = -current\_level;  else  y(end) = current\_level;  end  catch e  % assume next bit is the same as the last one  y(end) = current\_level;  end  % draw pulse and label  plot(bt, y, 'LineWidth', 2);  text(bit-0.5,pulse+0.25, num2str(bitstream(bit)), 'FontWeight', 'bold');  hold on;  end  % draw grid  grid on;  axis([0 length(bitstream) -1 1.5]);  set(gca,'YTick', [-pulse pulse])  set(gca,'XTick', 1:length(bitstream))  title('Manchester') |  |

**Câu 10: Vẽ phổ tín hiệu mã hóa Manchester?**

|  |  |
| --- | --- |
| %% Spectrum of Manchester  clear ; clc  bitstream = randi([0 1], 1, 10000);  fb=100;  yy=[];  pulse = 1;  yy=[];  for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.25:(bit-0.25);  if bitstream(bit) == 1  % low -> high  y = (bt<bit) \* pulse - 2\*pulse \* (bt < bit - 0.5);  % set last pulse point to high level  current\_level = pulse;  else  % high -> low  y = -(bt<bit) \* pulse + 2\*pulse\*(bt < bit - 0.5);  % set last pulse point to low level  current\_level = -pulse;  end  try  % if the next bit is the same as this one  %change the level  if bitstream(bit+1) == bitstream(bit)  y(end) = -current\_level;  else  y(end) = current\_level;  end  catch e  % assume next bit is the same as the last one  y(end) = current\_level;  end  yy=[yy y];  end  k=zeros(100,400);  for j=1:1:100  k(j,:)=yy(400\*j-399:400\*j);  sep(j,:)=fft(k(j,:),128);  end  n=size(sep,2);  fs=4\*fb;  f = (0:n-1)\*(fs/n); % frequency range  m\_sep=mean((abs(sep).^2),1)/fs;  figure(1);  plot(f,m\_sep);  grid on;  xlabel('Frequency')  title('Manchester') |  |

**\*Phân tích câu 1:**

|  |  |
| --- | --- |
| bitstream = [0 1 0 0 1 1 0 0 0 1 0];  pulse\_high = 5;  pulse\_low = 0; | * Tạo chuỗi gồm 11 bit * Thiết lập hai mức điện áp cao và thấp (5V và 0V). |
| for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.001:bit; | * Xét từng bit trong chuỗi bit vừa tạo * “bit” là số thứ tự của các bit trong chuỗi. * Lấy mẫu tín hiệu từng bit với tần số lấy mẫu là 1000 (1000 mẫu/bit) |
| if bitstream(bit) == 0  % low level pulse  y = (bt<bit)\*pulse\_low;  else  % high level pulse  y = (bt<bit) \* pulse\_high;  end | * Nếu là bit 0: y nhận giá trị là 0. * Ngược lại: y nhận giá trị là 5. * Biểu thức điều kiện (bt<bit) trả về 1 nếu đúng, ngược lại trả về 0 |
| try  if bitstream(bit+1) == 1  y(end) = pulse\_high;  end  catch e  % assume next bit is 1  y(end) = pulse\_high;  end | * Nếu bit tiếp theo là 1, giá trị y cuối cùng là 5V (mức cao). * Nếu không còn giá trị bit sau, giá trị y cuối cùng ở mức cao. |

**\*\*Phân tích câu 5:**

|  |  |
| --- | --- |
| bitstream = randi([0 1], 1, 10000);  fb=100;  pulse\_high = 1;  pulse\_low = -1;  yy=[]; | * Tạo chuỗi ngẫu nhiên 10000 bits. * Tốc độ bit: 100bps * Thiết lập hai mức logic tương ứng 1V và -1V. * Mảng “yy” lưu các giá trị của biến y. |
| for bit = 1:length(bitstream)  % set bit time  bt = bit-1:0.25:(bit-0.25);  if bitstream(bit) == 0  % low level pulse  y = (bt<bit)\*pulse\_low;  else  % high level pulse  y = (bt<bit) \* pulse\_high;  end  yy=[yy y];  end | * Tốc độ lấy mẫu: 4 mẫu/bit * Mã hóa NRZ-L. * Mảng “yy” lưu lại giá trị của y để dùng phân tích phổ. |
| k=zeros(100,400);  for j=1:1:100  k(j,:)=yy(400\*j-399:400\*j);  sep(j,:)=fft(k(j,:),128);  end | * Có tổng cộng 40000 mẫu cho chuỗi tín hiệu này. * Mảng k gồm 100 hàng 400 cột. * Chia mảng yy thành mảng k với 100 hàng và 400 cột. * Phân tích FFT 128 điểm với từng hàng của k. |
| n=size(sep,2);  fs=4\*fb;  f = (0:n-1)\*(fs/n); % frequency range | * Tạo khoảng tần số của tín hiệu. * Tần số lấy mẫu gấp 4 lần bit rate |
| m\_sep=mean((abs(sep).^2),1)/fs; | * Tính trung bình 100 hàng của mỗi 128 điểm FFT. * Kết quả là biên độ của phổ tín hiệu. |