**Simulación en MATLAB de Códigos de Línea y Comparación de sus PSD**

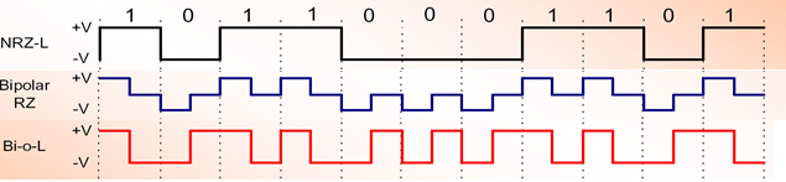
Adapted from: <https://drmoazzam.com/matlab-simulation-of-line-codes-and-their-psd>

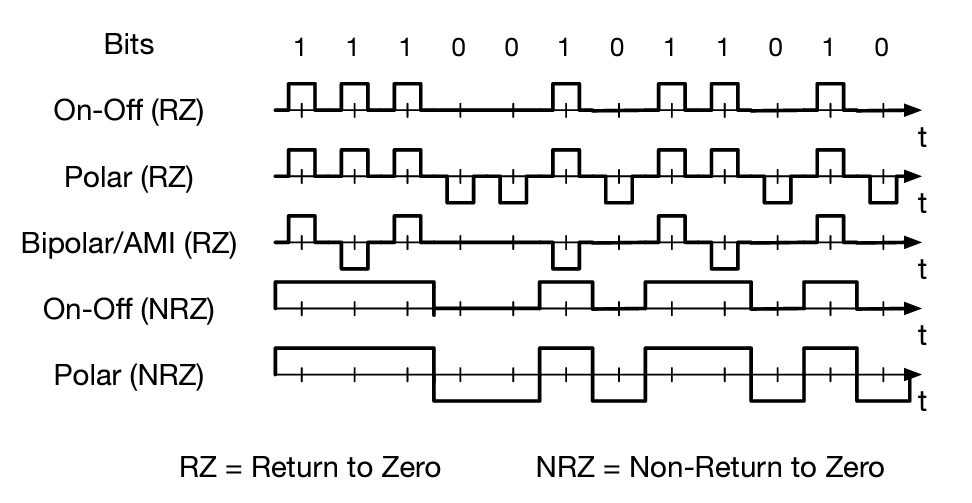
Para transmitir los bits sobre un canal físico, deben transformarse en una forma de onda física. Un codificador de línea o transmisor binario en banda base convierte una secuencia de bits en una forma de onda física adecuada para la transmisión a través de un canal. Los codificadores de línea utilizan el término "marca" para indicar un “1” y "espacio" para indicar “0”. En sistemas de banda base, los datos binarios pueden transmitirse utilizando varios tipos de pulsos. Estos diferentes tipos de formas de onda están diseñados para satisfacer distintos criterios de rendimiento en diversos escenarios. Cada tipo de código de línea tiene sus propias ventajas y desventajas. La elección de la forma de onda de codificación de línea depende de las características operativas de un sistema, tales como:

**Requerimiento de ancho de banda:** Los códigos de línea como NRZ-L requieren menos ancho de banda, pero tienen la desventaja de tener un nivel de DC y una mala sincronización.

**Requerimiento de sincronización:** Un esquema de codificación de línea como Manchester tiene buena sincronización, pero requiere un ancho de banda significativamente mayor.

**Complejidad del receptor:** Los esquemas complejos obviamente requieren un receptor más complejo, lo que aumenta el costo.

La entrada del codificador de línea es la secuencia de bits digitales que se transmitirá por el canal, y la salida es una secuencia de formas de onda en función de los bits de datos de entrada. Aquí vamos a presentar la simulación en MATLAB de los esquemas On-Off NRZ, Bipolar NRZ-L, Bipolar-RZ y Manchester-L. En la siguiente figura, podemos ver cómo una secuencia de bits ‘10110001101’ es codificada en formas de onda físicas por cada uno de estos esquemas. Para NRZ-L, un ‘1’ se representa como voltaje positivo +V durante toda la duración del bit, mientras que un ‘0’ se representa como voltaje negativo -V durante toda la duración del bit. En comparación, para Bipolar-RZ, vemos que, en el medio del bit, +V y -V regresan a cero para ‘1’ y ‘0’, respectivamente. Bi-phase-L es otro nombre para el esquema Manchester-L, donde observamos una transición de +V a -V en el medio del bit para un ‘1’ y una transición de -V a +V para un ‘0’.[](http://drmoazzam.com/wp-content/uploads/2015/06/Line-Codings-schemes.png)



En comunicaciones digitales, la Densidad Espectral de Potencia (PSD, por sus siglas en inglés) es un concepto fundamental para el análisis y diseño de sistemas de transmisión. La PSD describe cómo se distribuye la potencia de una señal en función de la frecuencia, proporcionando una representación del contenido de frecuencia de las formas de onda utilizadas en la transmisión de datos. Conocer la PSD es crucial, ya que permite evaluar el ancho de banda necesario para transmitir una señal y prever posibles interferencias en el canal. Además, la PSD ayuda a determinar la eficiencia espectral de diferentes esquemas de codificación de línea y permite identificar la presencia de componentes de baja frecuencia (DC) que podrían afectar la estabilidad y sincronización de la señal en sistemas de banda base.

Para simular estas formas de onda en MATLAB y comparar sus Densidades Espectrales de Potencia (PSD), necesitamos representar estas formas de onda como valores o muestras discretas. En nuestra simulación, estamos representando la forma de onda correspondiente a cada bit mediante 10 muestras. Por ejemplo, para el código Bipolar RZ-L, si el bit de entrada es ‘1’, generaremos la forma de onda muestreada “V V V V V 0 0 0 0 0” en la salida, y en el caso de que el bit sea ‘0’, la salida será “-V -V -V -V -V 0 0 0 0 0”.

A continuación se presenta el código fuente completo:

clear all;

close all;

%Nb is the number of bits to be transmitted

Nb=10000;

% Generate Nb bits randomly

b=rand(1,Nb)>0.5;

%Rb is the bit rate in bits/second

Rb=4;

%Since each waveform is represented by 10 samples so sampling

%frequency is 10 times the bit rate

fs=10\*Rb;

NRZ\_out=[];

RZ\_out=[];

Manchester\_out=[];

NRZno\_polar=[];

%Vp is the peak voltage +v of the waveform

Vp=5;

%Here we encode input bitstream as On-Off (NRZ) waveform

for index=1:size(b,2)

if b(index)==1

NRZno\_polar=[NRZno\_polar [1 1 1 1 1 1 1 1 1 1]\*Vp];

elseif b(index)==0

NRZno\_polar=[NRZno\_polar [0 0 0 0 0 0 0 0 0 0]\*0];

end

end

%Now we draw the PSD spectrum of On-Off (NRZ) using Welch PSD

%estimation method

h = spectrum.welch;

Hpsd=psd(h,NRZno\_polar,'Fs',fs);

figure;

hold on;

handle1=plot(Hpsd);

set(handle1,'LineWidth',2.5,'Color','g')

%Here we encode input bitstream as Bipolar NRZ-L waveform

for index=1:size(b,2)

if b(index)==1

NRZ\_out=[NRZ\_out [1 1 1 1 1 1 1 1 1 1]\*Vp];

elseif b(index)==0

NRZ\_out=[NRZ\_out [1 1 1 1 1 1 1 1 1 1]\*(-Vp)];

end

end

%Now we draw the PSD spectrum of Bipolar NRZ-L using Welch PSD

%estimation method

h = spectrum.welch;

Hpsd=psd(h,NRZ\_out,'Fs',fs);

handle2=plot(Hpsd)

set(handle2,'LineWidth',2.5,'Color','r')

%Here we encode input bitstream as Bipolar RZ waveform

for index=1:size(b,2)

if b(index)==1

RZ\_out=[RZ\_out [1 1 1 1 1 0 0 0 0 0]\*Vp];

elseif b(index)==0

RZ\_out=[RZ\_out [1 1 1 1 1 0 0 0 0 0]\*(-Vp)];

end

end

%Now we draw the PSD spectrum of Bipolar RZ using Welch PSD

%estimation method

h = spectrum.welch;

Hpsd=psd(h,RZ\_out,'Fs',fs);

handle3=plot(Hpsd);

set(handle3,'LineWidth',2.5,'Color','b')

%Here we encode input bitstream as Manchester-L waveform

for index=1:size(b,2)

if b(index)==1

Manchester\_out=[Manchester\_out [1 1 1 1 1 -1 -1 -1 -1 -1]\*Vp];

elseif b(index)==0

Manchester\_out=[Manchester\_out [1 1 1 1 1 -1 -1 -1 -1 -1]\*(-Vp)];

end

end

%Now we draw the PSD spectrum of Manchester-L using Welch PSD

%estimation method

h = spectrum.welch;

Hpsd=psd(h,Manchester\_out,'Fs',fs);

handle4=plot(Hpsd)

set(handle4,'LineWidth',2.5,'Color','k')

legend('On-Off (NRZ)','Bipolar NRZ-L','Bipolar-RZ','Manchester-L');

hold off;

figure;

plot((1:Nb\*10)/10,NRZno\_polar);

ylim([-Vp\*1.1 Vp\*1.1 ])

xlabel('bits-->');

ylabel('Amplitude (volts)-->');

title('ON OFF NRZ encoded bit stream');

figure;

plot((1:Nb\*10)/10,NRZ\_out);

ylim([-Vp\*1.1 Vp\*1.1 ])

xlabel('bits-->');

ylabel('Amplitude (volts)-->');

title('Bipolar NRZ-L encoded bit stream');

figure;

plot((1:Nb\*10)/10,RZ\_out);

ylim([-Vp\*1.1 Vp\*1.1 ])

xlabel('bits-->');

ylabel('Amplitude (volts)-->');

title('RZ-L encoded bit stream');

figure;

plot((1:Nb\*10)/10,Manchester\_out);

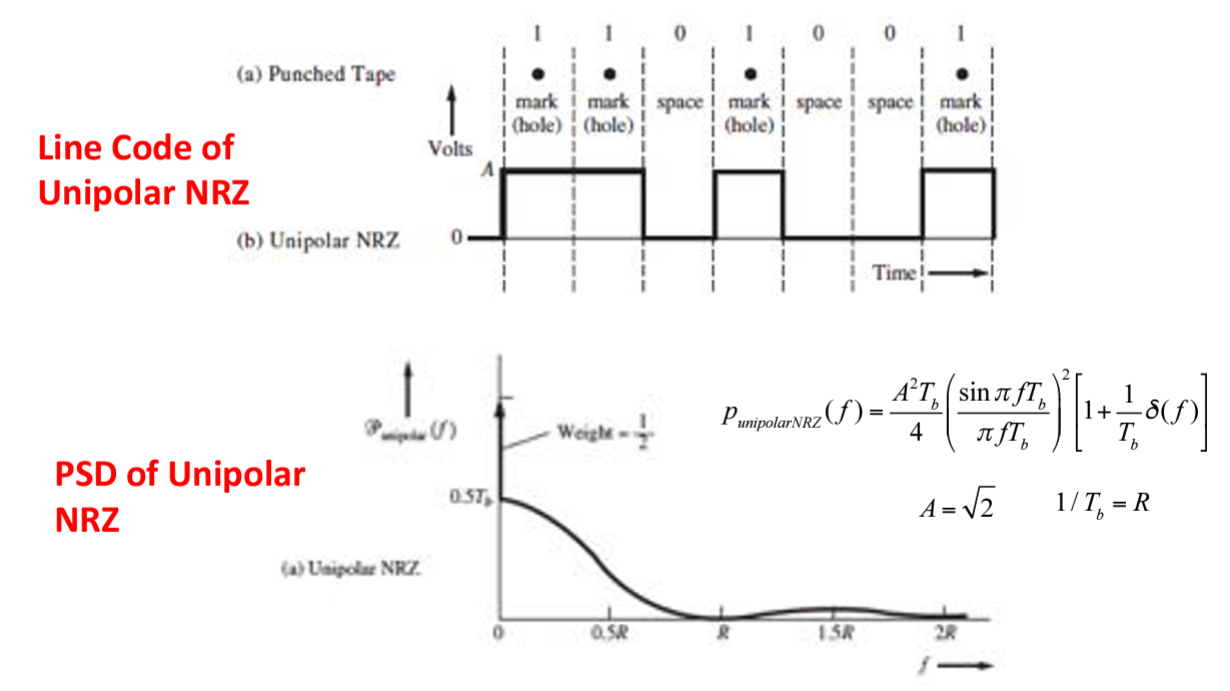
ylim([-Vp\*1.1 Vp\*1.1 ])

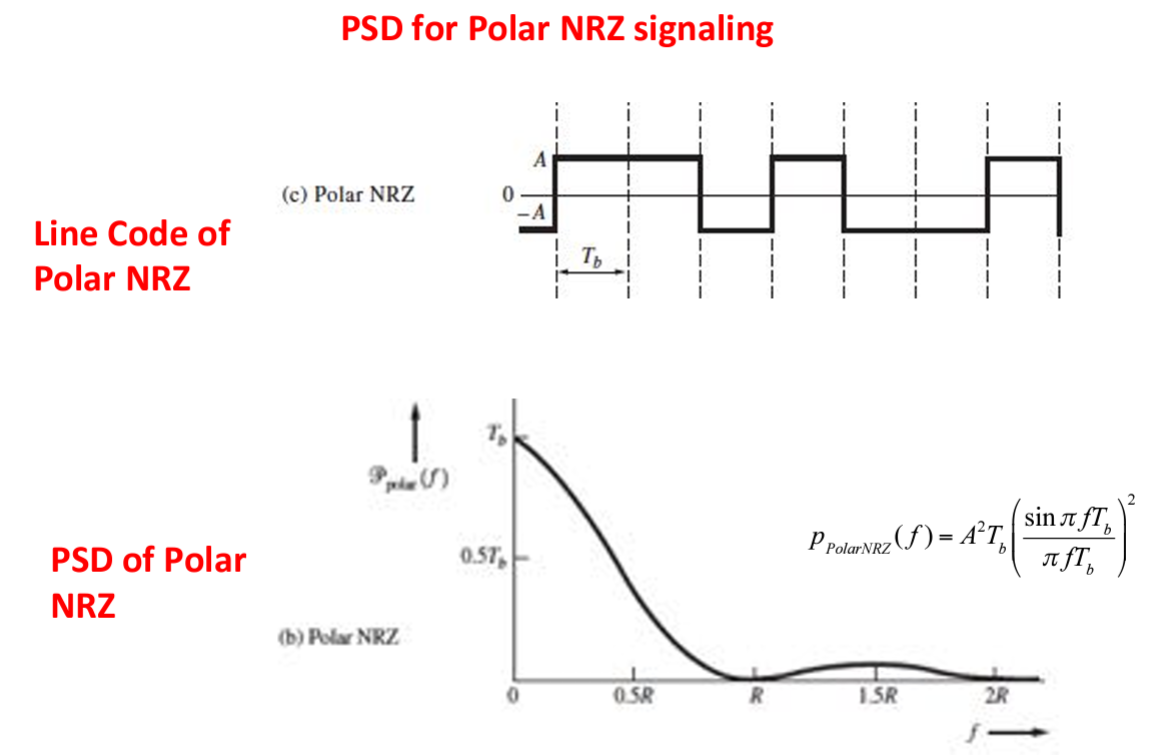
xlabel('bits-->');

ylabel('Amplitude (volts)-->');

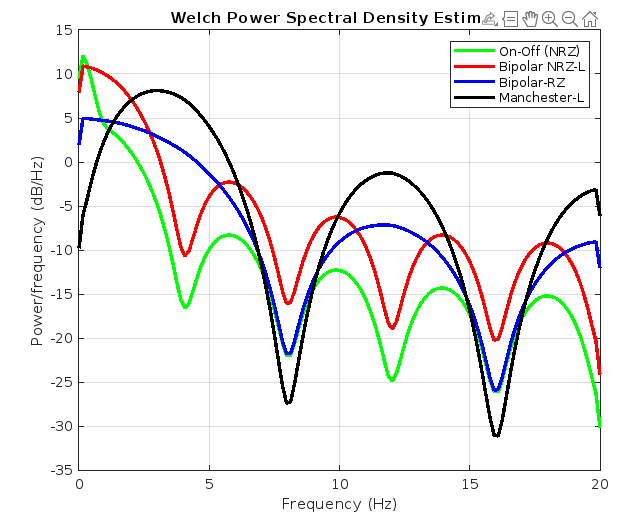
title('Manchester-L encoded bit stream');

Al comparar la PSD de diferentes códigos de línea, los ingenieros Telemáticos pueden seleccionar el esquema de codificación más adecuado para una aplicación específica, optimizando así el rendimiento del sistema en términos de ancho de banda, consumo de energía y complejidad de implementación. En un mundo donde la eficiencia espectral y la robustez de la comunicación son cada vez más demandadas, entender y analizar la PSD se vuelve indispensable para el desarrollo de soluciones de comunicación confiables y eficientes. A continuación se muestra la PSD teórica para dos esquemas de codificación de línea (tomado de <https://www.siue.edu/~yadwang/ECE375_Lec8.pdf>)

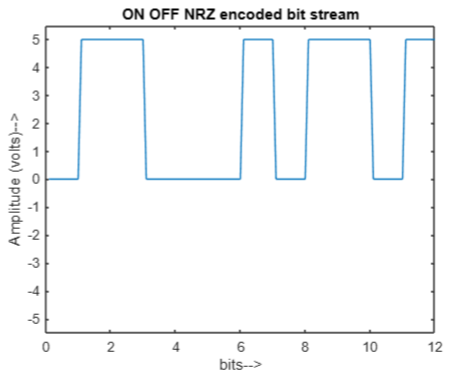
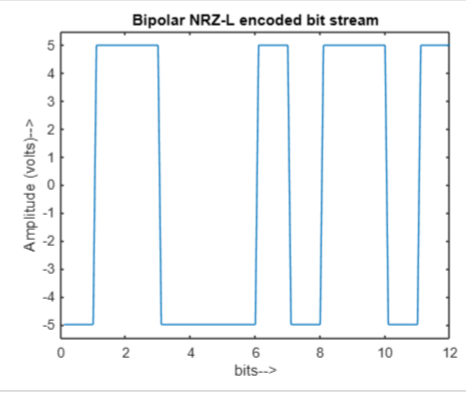


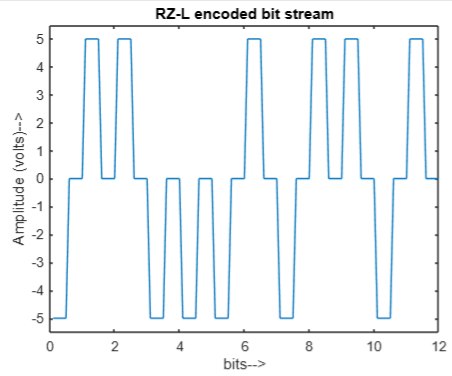
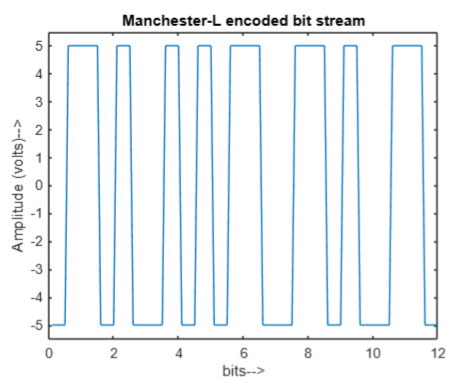


La comparación de la PSD de estos esquemas de codificación de línea es la siguiente:



La salida de los diferentes codificadores de línea para la secuencia de bits 0 1 1 0 0 0 1 0 1 1 0 1 se grafica de la siguiente manera:





Preguntas y Ejercicios Basados en el Código de Simulación:

**Parte 1: Interpretación de Resultados**

Observando las gráficas de PSD (densidad espectral de potencia) para los diferentes esquemas de códigos de línea, ¿qué código de línea tiene el mayor ancho de banda y cuál el menor? ¿Cómo se explica esto con base en la teoría?

Describe las características de sincronización los códigos en la presencia de a una cadena larga de unos o de ceros. Por ejemplo, compara el comportamiento del código Manchester (que tiene una transición en cada bit) con el On-Off NRZ Explique ¿por qué Manchester proporciona mejor sincronización?

**Parte 2: Comparación y Simulación con Comunicaciones (RS-232 y UART)**

1. Considerando los códigos de línea implementados en clase, ¿cuál esquema sería adecuado para la comunicación RS-232? ¿cuál esquema sería adecuado para la comunicación UART?

2. Ajustes para Simular Comunicación RS-232 (9600 8N1)

En RS-232 a 9600 baudios, con un formato de 8 bits de datos, sin paridad y 1 bit de parada (8N1), se pide realizar ajustes sobre el código de Matlab para simular la transmisión de una cadena de 100 caracteres aleatorios. Muestre la gráfica de PSD.

Además, realice la simulación en Matlab de una comunicación UART (9600 8N1) entre dos tarjetas ESP32. En este caso, se le pide simular el flujo de datos de 100 caracteres aleatorios. Muestre la gráfica de PSD.

Luego muestre el bitstream al trasmitir juntos los caracteres A y Z para cada uno de los esquemas de códigos de línea.

Explique que cambios se puede ver en la PSD si para la misma secuencia de bits, se multiplica el Rb por 100?