UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

GUÍAS DE PRÁCTICAS

PRÁCTICA N.º 5

TEMA: Códigos de Transmisión Binaria UNRZ(h), URZ(h), PNRZ(h) y BRZ(h)

I. OBJETIVOS

• Experimentar la generación y representación de la codificación Unipolar NRZ utilizando la función UNRZ(h).

• Explorar el comportamiento de la codificación Unipolar RZ mediante la función URZ(h).

• Analizar la forma de onda Polar NRZ implementada con la función PNRZ(h).

• Investigar la codificación Bipolar RZ (AMI) a través de la función BRZ(h).

II. INSTRUCCIONES

• Constituir grupos de trabajo de 2 a 3 integrantes.

• Revisar la teoría de los códigos de línea binarios (UNRZ, URZ, PNRZ, BRZ).

• Implementar cada función en MATLAB y ejecutar las simulaciones.

• Interpretar y comparar las señales resultantes.

III. EQUIPOS, MATERIALES Y RECURSOS

• Computadora con MATLAB instalado.

• Calculadora científica.

• Apuntes o bibliografía sobre codificación digital.

IV. PROCEDIMIENTO

Experimento #1: Función UNRZ(h)

1. Definir un vector de bits de prueba h, por ejemplo: h=[1 0 1 1 0].

2. Copiar y pegar la función UNRZ(h) en el editor de MATLAB:

```matlab
function UNRZ(h)
   clf;
   n=1; l=length(h);
   h(end+1)=1;
   while n<=length(h)-1
```

```
        t=n-1:0.001:n;
        if h(n)==0
            y = (t>n);
        else
            y = (t<n) + (t==n);
        end
        plot(t,y,LineWidth,2.5);
        hold on; grid on;
        axis([0 l-1 -1.5 1.5]);
        n=n+1;
    end
end
```

3. Ejecutar UNRZ(h) y observar la señal unipolar NRZ.

Experimento #2: Función URZ(h)

1. Utilizar el mismo vector h y añadir la función URZ(h) en MATLAB:

```matlab
function URZ(h)
    clf;
    n=1; l=length(h);
    h(end+1)=1;
    while n<=length(h)-1
        t=n-1:0.001:n;
        if h(n)==0
            y = (t>n);
        else
            y = (t<n-0.5) + (t==n);
        end
        plot(t,y,LineWidth,2.5);
        hold on; grid on;
        axis([0 l-1 -1.5 1.5]);
        n=n+1;
    end
end
```

2. Ejecutar URZ(h) y analizar la señal unipolar RZ.

Experimento #3: Función PNRZ(h)

1. Copiar la función PNRZ(h) y pegar en MATLAB:

```matlab
function PNRZ(h)
```

```
    clf;
    n=1; l=length(h);
    h(end+1)=1;
    while n<=length(h)-1
        t=n-1:0.001:n;
        if h(n)==0
            y = -(t<n) - (t==n);
        else
            y =  (t<n) + (t==n);
        end
        plot(t,y,LineWidth,2.5);
        hold on; grid on;
        axis([0 l-1 -1.5 1.5]);
        n=n+1;
    end
end
```

2. Ejecutar PNRZ(h) para generar la señal polar NRZ.

Experimento #4: Función BRZ(h)

1. Incluir la función BRZ(h) en el script de MATLAB:

```matlab
function BRZ(h)
    clf;
    n=1; l=length(h);
    h(end+1)=1;
    while n<=length(h)-1
        t=n-1:0.001:n;
        if h(n)==0
            y = -(t<n-0.5)-(t==n);
        else
            y =  (t<n-0.5)+(t==n);
        end
        plot(t,y,LineWidth,2.5);
        hold on; grid on;
        axis([0 l-1 -1.5 1.5]);
        n=n+1;
    end
end
```

2. Ejecutar BRZ(h) y evaluar la codificación bipolar RZ.

V. RESULTADOS OBTENIDOS

• La simulación en MATLAB muestra la forma de onda específica de cada código de línea.

• Se pueden contrastar las características de ancho de banda y nivel de sincronización entre los distintos métodos.

VI. CONCLUSIONES

1. Cada variante de codificación afecta el ancho de banda y la eficiencia espectral de la señal.

2. Los códigos con retorno a cero facilitan la sincronización del receptor.

3. La codificación bipolar reduce la componente de continua y mejora la inmunidad a errores.

VII. RECOMENDACIONES

1. Comparar las formas de onda con diferentes tasas de bits para observar efectos en frecuencia.

2. Evaluar el impacto de ruido en la calidad de la sincronización de cada código.

3. Investigar códigos multivaluados como MLT-3 para aplicaciones de mayor eficiencia.

VALIDACIÓN DE LAS GUÍAS DE PRÁCTICAS