Codificación de señales digitales

Julio César Ramirez-Pacheco

6/3/2020

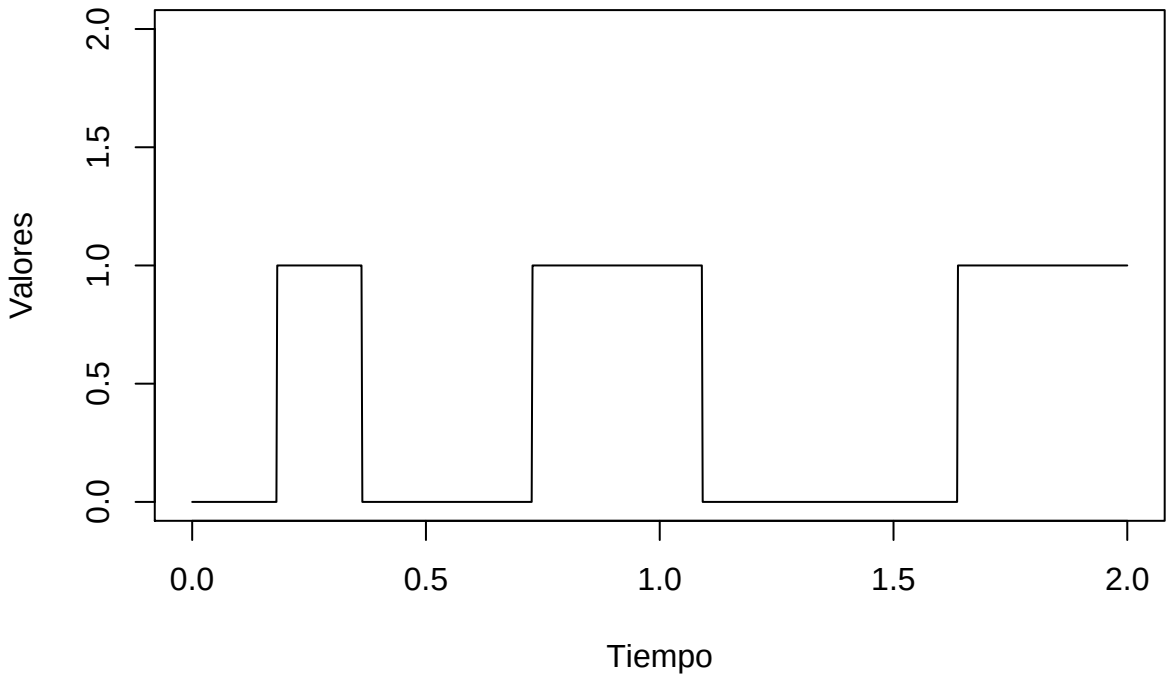
El concepto de codificación

La codificación de señales permite modificar características de la señal de información para adaptarlas al canal de comunicación. La codificación de puede realizar de señales digitales a digitales, de señales digitales a analógicas, y de señales analógicas a analógicas. En esta práctica nos concentraremos en la codificación de señales digitales a digitales y usualmente podemos encontrar los siguientes tipos de codificación:

- NRZ-I.
- NRZ-L.
- AMI bipolar.
- Pseudoternario.
- Manchester.
- Manchester diferencial.
- B8ZS.
- HDB3.

El propósito de la siguiente práctica es generar todas las codificaciones usando el lenguaje de programación R. Iniciaremos generando la señal de información y posteriormente codificar usando NRZ-L. La generación de la señal de información se realiza de la siguiente manera:

```
# Información original 01001100011
# Representación gráfica
xt <- c(0,1,0,0,1,1,0,0,0,1,1)
bits <- rep(xt, each=100)
time <- seq(0,2, length=length(bits))
plot(time, bits, type = "l",
main="Secuencia de datos de información", xlab="Tiempo", ylab="Valores", ylim=c(0,2))
```

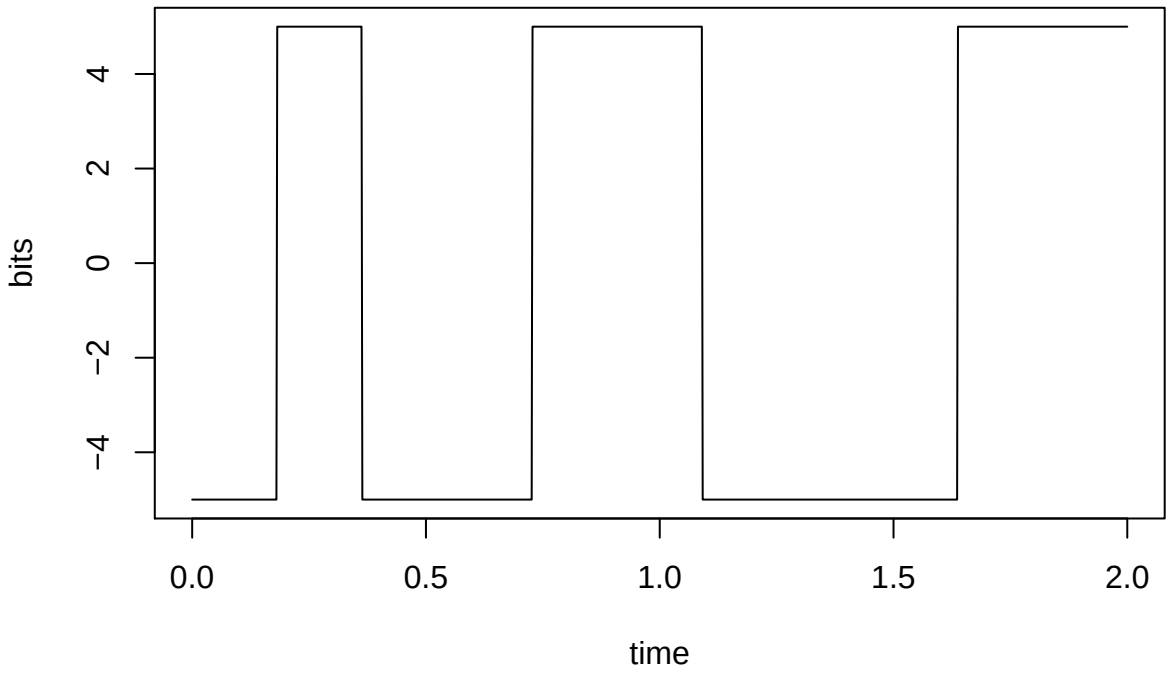


Posteriormente codificamos la anterior señal de comunicación usando las siguientes reglas:

- 0 se codifica con un nivel alto.
- 1 se codifica con un nivel bajo.

Programando la anterior regla, la codificación NRZ-L queda de la siguiente manera:

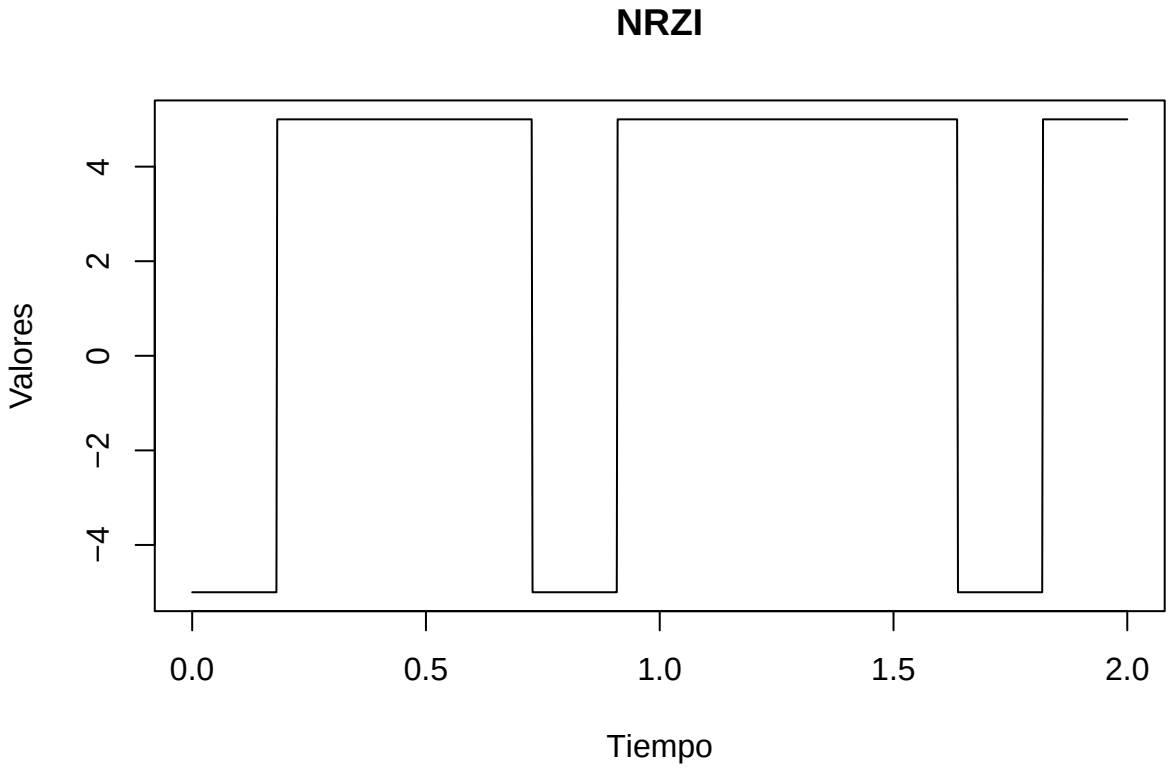
```
xt <- c(0,1,0,0,1,1,0,0,0,1,1)          # Se generan los códigos
# Ahora se codifica de acuerdo a la regla descrita arriba
nrzl <- ifelse(xt>0, 5, -5)
bits <- rep(nrzl, each=100)
time <- seq(0,2, length=length(bits))  # Se genera el tiempo (puede ser cualquiera).
plot(time, bits, type = "l")
```



NRZ-I

Si ahora queremos codificar con NRZ-I, entonces tenemos:

```
xt <- c(0,1,0,0,1,1,0,0,0,1,1)          # Se generan los códigos
# Ahora se codifica de acuerdo a la regla del NRZ-I
nrzi <- NULL
memory <- 0
for(i in 1:length(xt)){
  if(xt[i] == 0 && memory == 0){
    nrzi[i] <- 0
    memory  <- 0
  }
  else if(xt[i]==0 && memory ==1){
    nrzi[i] <- 1
    memory  <- 1
  }
  else if(xt[i] == 1 && memory == 0){
    nrzi[i] <- 1
    memory  <- 1
  }
  else{
    nrzi[i] <- 0
    memory  <- 0
  }
}
nrzi_polar <- ifelse(nrzi==0,-5,5)
bits <- rep(nrzi_polar, each=100)
time <- seq(0,2, length=length(bits))  # Se genera el tiempo (puede ser cualquiera).
plot(time, bits, type= "l", main="NRZI", xlab="Tiempo", ylab="Valores")
```



Tareas

1. Implementar la codificación AMI bipolar.
2. Pseudoternaria.
3. Manchester.
4. Manchester diferencial.
5. B8ZS.
6. HDB3.

Utilizando el lenguaje de programación R. Pueden utilizar la plataforma RStudio Cloud para poder realizarlo. Fecha de entrega: Dominggo 7 de Junio de 2020. Atte. Dr. Julio César Ramírez Pacheco.