Muestreo y Reconstrucción

Ejemplos

Ejemplos de cálculo

1. Muestreo.

1.1. Una señal analógica es representada por x(t)=sen(450πt)+3sen(1450πt).

1.1.1. Determinar la frecuencia de muestreo mínima apropiada para esta señal.

1.1.2. Determinar la frecuencia analógica de corte del filtro muestreador.

1.1.3. Calcular las frecuencias digitales de las componentes de la señal muestreada si se muestrea a 500 Hz.

1.1.4. Calcular las frecuencias digitales de las réplicas.

1.1.5. Determinar cuáles de dichas componentes quedan submuestreadas.

1.1.6. Calcular las frecuencias analógicas de la señal reconstruida si se reconstruye a la misma frecuencia con la que se muestreó y con un filtro reconstructor de igual ancho de banda que el muestreador.

Solución:

1.1.1. Para que no ocurra submuestreo se debe cumplir el teorema de Nyquist. Como la señal es x(t)=sen(2*225πt)+3sen(2*725πt) debería muestrearse a

$$F_m > 2F_{máx} = 2 * 725Hz = 1450Hz$$

1.1.2. El filtro muestreador debería ser un pasabajos con una frecuencia de corte superior igual a la frecuencia máxima permitida, que es la mitad de la frecuencia de muestreo

$$F_c > \frac{F_m}{2} = 725Hz$$

1.1.3. La frecuencia digital resultante del muestreo de una componente senoidal es

$$f = \frac{F}{F_m} \quad \text{donde} \quad f = \frac{\omega}{2\pi}$$

Para la componente de menor frecuencia, el resultado del muestreo da una frecuencia digital menor que ½, pero en la otra componente ocurre lo contrario

$$f_{min} = \frac{F_{min}}{F_m} = \frac{225Hz}{500Hz} = \frac{9}{20} \qquad f_{max} = \frac{F_{max}}{F_m} = \frac{725Hz}{500Hz} = \frac{29}{20}$$

Esto puede verse también desarrollando la expresión de la secuencia de la señal muestreada, agrupando apropiadamente dentro del corchete

$$x[n] = sen\left[\frac{450}{500}\pi n\right] + 3sen\left[\frac{1000+450}{500}\pi n\right] = sen\left[\frac{450}{500}\pi n\right] + 3sen\left[2\pi n + \frac{450}{500}\pi n\right] =$$

$$= 4sen\left[2\frac{9}{20}\pi n\right]$$

1.1.4. Una componente de frecuencia analógica F_0, al ser muestreada, tiene un espectro periódico (de período 1 para $f=F/F_s$ o 2π para $\omega=2\pi\Omega/\Omega_s$). La k-ésima repetición de una componente de frecuencia muestreada es f_k

$$f_k = \pm\frac{F_0}{F_s} \pm k \quad ; \quad k = 0;1;2;3;...$$

El cálculo de las réplicas de ambas componentes, en este caso, conduce a valores de frecuencias digitales coincidentes

$$f_{min} = \frac{F_{min}}{F_m} \pm k = \frac{9}{20} \pm k \quad ; \quad k = 0;1;2;3;...$$

$$f_{max} = \frac{F_{max}}{F_m} \pm k = \frac{29}{20} \pm k = \left(\frac{9}{20} + \frac{\overset{1}{\not 20}}{\underset{1}{\not 20}}\right) \pm k = \frac{9}{20} + 1 \pm k = \frac{9}{20} \pm c \quad ; \quad c = 0;1;2;3;...$$

Las repeticiones periódicas de la primer componente son

$$\rightarrow f_0 = \left\{\left\langle +\frac{9}{20}\right\rangle; \left\langle -\frac{9}{20}\right\rangle\right\}; \quad f_1 = \begin{cases} +\frac{9}{20}+1 = +\frac{29}{20} \\ -\frac{9}{20}+1 = +\frac{11}{20}, \\ +\frac{9}{20}-1 = -\frac{11}{20} \\ -\frac{9}{20}-1 = -\frac{29}{20} \end{cases} \quad f_2 = \begin{cases} +\frac{9}{20}+2 = +\frac{49}{20} \\ -\frac{9}{20}+2 = +\frac{31}{20}, \\ +\frac{9}{20}-2 = -\frac{31}{20} \\ -\frac{9}{20}-2 = -\frac{49}{20} \end{cases} \quad f_3 = \begin{cases} +\frac{9}{20}+3 = +\frac{69}{20} \\ -\frac{9}{20}+3 = +\frac{51}{20}, \\ +\frac{9}{20}-3 = -\frac{51}{20} \\ -\frac{9}{20}-3 = -\frac{69}{20} \end{cases} \quad ...$$

Para la segunda componente

$$\rightarrow f_0 = \begin{cases} +\frac{29}{20}; \\ -\frac{29}{20} \end{cases} \quad f_1 = \begin{cases} +\frac{29}{20}+1 = +\frac{49}{20} \\ -\frac{29}{20}+1 = \left\langle -\frac{9}{20} \right\rangle \\ +\frac{29}{20}-1 = \left\langle +\frac{9}{20} \right\rangle \\ -\frac{29}{20}-1 = -\frac{49}{20} \end{cases}; \quad f_2 = \begin{cases} +\frac{29}{20}+2 = +\frac{69}{20} \\ -\frac{29}{20}+2 = +\frac{11}{20}; \\ +\frac{29}{20}-2 = -\frac{11}{20} \\ -\frac{29}{20}-2 = -\frac{69}{20} \end{cases} \quad f_3 = \begin{cases} +\frac{29}{20}+3 = +\frac{89}{20} \\ -\frac{29}{20}+3 = +\frac{31}{20}; \\ +\frac{29}{20}-3 = -\frac{31}{20} \\ -\frac{29}{20}-3 = -\frac{89}{20} \end{cases} \quad \ldots$$

1.1.5. El incumplimiento del teorema de muestreo se verifica cuando la frecuencia digital de una componente muestreada es mayor que ½ porque queda fuera de la banda pasante del filtro reconstructor. Quien queda adentro de dicha banda es una réplica, que es interpretada como una componente real. La componente de F=725Hz queda submuestreada mientras que la primer repetición f_1 es interpretada como una frecuencia original. En el cálculo de las réplicas se han marcado con corchetes quebrados las frecuencias digitales que serán reconstruidas.

1.1.6. Con la reconstrucción de la señal se obtiene

$$F = fF_m = \frac{9}{20}500\text{Hz} = 225\text{Hz} \therefore x_r(t) = 4\text{sen}(2\pi 225t)$$

Ejemplos de simulación

1. Muestreo.

1.1. Simular el muestreo de una señal cosenoidal pura y determinar su espectro.

Solución:

```
%Ejemplo de simulacion 4.1.
%Muestreo de senal cosenoidal pura
clc,clear,close all

%Especificaciones
N=200; %numero de muestras
Fs=1000; %frecuencia de muestreo
k=1:1:N; %rango de indice de orden de muestras

%Rangos de tiempo y frecuencia
n=0:1:N-1; %rango de tiempo discreto
f=-1/2:1/N:1/2-1/N; %rango de frecuencia digital
w=-pi:2*pi/N:2*pi*(1/2-1/N); %rango de pulsacion digital
F=-Fs/2:Fs/N:Fs*(1/2-1/N); %rango de frecuencia analogica
W=-pi*Fs:2*pi*Fs/N:2*pi*Fs*(1/2-1/N); %rango pulsacion analogica

%Senal y espectro
Fa=200; %frecuencia analogica
xn=cos(2*pi*Fa*n/Fs); %senal muestreada
Xw=freqz(xn,1,w); %TFTD de senal muestreada

%Representacion grafica
%Senal muestreada
figure(1)
stem(n,xn,'k.')
xlabel('n')
ylabel('x[n]')
title('Senal muestreada')

figure(2)
%Espectro de la senal muestreada en funcion de w
subplot(2,2,1)
plot(w,abs(Xw),'k-')
xlabel('w,rad')
ylabel('abs{X(e^(j*\omega))}')
title('Modulo del espectro de la senal muestreada')
axis([-pi pi 0 100])
subplot(2,2,2)
plot(w,angle(Xw),'k-')
xlabel('w,rad')
ylabel('ang{X(e^(j*\omega))}')
title('Fase del espectro de la senal muestreada')
axis([-pi pi -pi pi])

%Espectro de la senal muestreada en funcion de F
subplot(2,2,3)
plot(F,abs(Xw),'k-')
xlabel('F,Hz')
ylabel('abs{X(j*2*pi*F)}')
title('Modulo del espectro de la senal')
axis([-Fs/2 Fs/2 0 100])
subplot(2,2,4)
plot(F,angle(Xw),'k-')
xlabel('F,Hz')
ylabel('ang{X(j*2*pi*F)}')
title('Fase del espectro de la senal')
axis([-Fs/2 Fs/2 -pi pi])
```

Ejemplos de laboratorio

1. Filtro anti-solapamiento, muestreo y reconstrucción.

★ Escribir un programa que permita ingresar una señal en la entrada del conversor A/D, para luego filtrarla con un filtro anti-solapamiento y muestrearla con una frecuencia ajustable, multiplicarla por una ganancia, y reconstruirla con un filtro reconstructor también con una frecuencia ajustable.

Solución:

```
;-------------------------------------------------------------------------
; muestreo.asm
; Programa para sacar por el DAC lo que entra por el ADC
; Guardando tambien en memoria n valores para analizarlos
;-------------------------------------------------------------------------
;
               .start  "AICTEST",0x809802; inicio de ensamblado
               .sect   "AICTEST"
;
;-------------------------------------------------------------------------
; Definicion constantes usadas en el programa
;-------------------------------------------------------------------------
; Direcciones de los registros de control del TIMER0.
;
; El TIMER0 es usado para generar el Master Clock.
;
; El Master Clock es usado por el AIC para generar
; las frecuencias de  conversion del ADC y del DAC
; y para las frecuencias de corte
; de los filtros de capacidades conmutadas
;
T0_ctrl         .set    0x808020; TIM0 gl control
T0_count        .set    0x808024; TIM0 count
T0_prd          .set    0x808028; TIM0 prd
;
; Direcciones de los registros de la puerta serie
;
S0_gctrl        .set    0x808040; SP0 global control
S0_xctrl        .set    0x808042; SP0 FSX/DX/CLKX port ctl
S0_rctrl        .set    0x808043; SP0 FSR/DR/CLKR port ctl
S0_xdata        .set    0x808048; SP0 data transmit
S0_rdata        .set    0x80804C; SP0 data receive
;
; Valores de los registros del AIC que definen las
; frecuencias de conversion del DAC (TB) y del ADC (RB)
; frec. de conversion = frec. del SCF / contador B
; frec. del SCF = Master Clock / (2 * Contador A)
; Master Clock =  Timer cero del DSP
;
TA              .set    11
TB              .set    15
RA              .set    11
RB              .set    15
;
; Constante utilizada para poner en 1 el bit GIE
; del registro ST que habilita las interrupciones
;
GIE             .set    0x2000
;
;-------------------------------------------------------------------------
; Definicion de algunas constantes como datos
;-------------------------------------------------------------------------
; Si la primer palabra que se envia al AIC
; tiene los dos bits menos significativos en uno
; indica que la proxima es una palabra secundaria.
;
; Luego, de acuerdo a los dos bits menos significativos
; de la palabra secundaria se realiza una tarea diferente
; (cf manual USER'S GUIDE STARTER KIT pag 4-17).
;
; Se acomodan los valores de los registros TA, RA, TB y RB
; de acuerdo al formato de la palabra secundaria (cf. manual citado)
;
A_REG           .word   (TA<<9)+(RA<<2)+0; registro A
B_REG           .word   (TB<<9)+(RB<<2)+2; registro B
C_REG           .word   00000111b; registro de control
ultimo          .word   TABLA; dir ult dato tabla
veces           .word   100
;
; Valores que se cargaran en los registros
; de control de la puerta serie 0
;
S0_gctrl_val    .word   0x0E970300
S0_xctrl_val    .word   0x00000111
S0_rctrl_val    .word   0x00000111
;
ADC_last        .word   0; ultimo valor ADC recibido
;
```

```
;************************************************************************
; El programa principal es un lazo
;************************************************************************
main            or      GIE,ST; activa interrupciones
                ldi     0xF4,IE; habilita XINT/RINT/INT2
                b       main; repeticion
;------------------------------------------------------------------------
DAC2            push    ST; rutina de servicio
                        ; interrupcion DAC
                push    R3
                ldi     @ADC_last,R3
                lsh     2,R3
                cmpi    32767,R3
                ldigt   32767,R3
                cmpi    -32768,R3
                ldilt   -32768,R3
                andn    3,R3
                sti     R3,@S0_xdata; pone a la salida
                        ; el nuevo valor DAC
                pop     R3
                pop     ST
                reti
;------------------------------------------------------------------------
ADC2            push    ST
                push    R3
                push    AR0
                push    R1
                ldi     @S0_rdata,R3
                lsh     16,R3
                ash     -18,R3
; La tabla se carga con 100  valores y luego se saltea este codigo
                ldi     @veces,R1
                cmpi    0,R1
                bz      seguir2
                subi    1,R1
                sti     R1,@veces
                ldi     @ultimo,AR0
                sti     R3,*AR0++
                sti     AR0,@ultimo
seguir          sti     R3,@ADC_last
                pop     R1
                pop     AR0
                pop     R3
                pop     ST
                reti
seguir2         b       seguir
```

```
;************************************************************************
; Esta parte del programa se usa solamente durante la inicializacion
; y se puede sobreescribir sin problemas por el stack o datos
;************************************************************************
                .entry  ST_STUB; inicio de la depuracion
;
; Carga del puntero pagina de datos DP con los 8 bits mas significativos
; del T0_ctrl (=808020h) o sea 80, definiendo asi que se use el DP
; y el stack del Kernel
ST_STUB         ldp     T0_ctrl
;
; Carga del SP con la direccion del lugar asignado
; con la directiva que se usa mas adelante,
; que indica que el stack comienza ahí
; Se programa el TIMER0 que se usa para generar el Master clock.
; El TIMER0 se ha programado en modo pulso con el periodo en 2.
; La frecuencia de salida es fsp=(50mhz/4)/2=12.5MHz/2=6.25MHz
;
                ldi     @stack,SP
                ldi     0,R0; Halt TIM0 & TIM1
                sti     R0,@T0_ctrl
                sti     R0,@T0_count; Ajusta la cuenta a 0
                ldi     2,R0; Ajusta los periodos a 2
                sti     R0,@T0_prd
;
; GO=1 hace el reset y comienzo de la cuenta y HLDneg=1 quita el hold
; (en la palabra de control del TIMER0=T0_ctrl)
;
                ldi     0x2C1,R0; Reinicializa ambos temp
                sti     R0,@T0_ctrl
;
;------------------------------------------------------------------------
; Programacion del registro del control de la transmision
                ldi     @S0_xctrl_val,R0
; Control transmision: Con FSX/DX/CLKX en modo de transmision
; de puerta serie (podrian usarse como simples pines de E/S).
                sti     R0,@S0_xctrl
;
; Programacion del registro de control de la recepcion
; definiendo los pines FSR/DR/CLKR para comunicacion serie (no E/S)
                ldi     @S0_rctrl_val,R0
                sti     R0,@S0_rctrl; control de recepcion
                ldi     0,R0
                sti     R0,@S0_xdata; DXR data value
;
```

```
; Programa el Global Control Register para la puerta serie.
; El 'C31 le envia al conversor (AIC) los datos en serie.
; Se le carga una palabra definida al inicio del programa (0x0E970300)
; y se define fuente externa de reloj para la Tx y Rx
; (en este caso la genera el AIC).
                ldi     @S0_gctrl_val,R0; Setup serial port
                sti     R0,@S0_gctrl    ; Global control
;
;=============================================================================
; Esta seccion inicializa el  AIC
;=============================================================================
;
AIC_INIT        LDI     0x10,IE; Habilita solo XINT int
                andn    0x34,IF; Deshabilita int, int2,
                        ; xint0, rint0
                ldi     0,R0; transmite un cero
                sti     R0,@S0_xdata
;
; Por el pin XF0 (controlado por el registro IOF)
; saca un cero y luego un uno,
; escalon necesario para resetear el AIC
; y luego ponerlo en RUN
                rpts    0x040
                ldi     2,IOF; XF0=0 resets AIC
                rpts    0x40
                ldi     6,IOF; XF0=1 runs AIC
;-----------------------------------------------------------------------------
                ldi     @C_REG,R0; Setup control register
                call    prog_AIC
                ldi     0xfffc,R0; Program AIC
                        ; to be real slow
                call    prog_AIC
                ldi     0xfffc|2,R0
                call    prog_AIC
                ldi     @B_REG,R0; Bump up the Fs
                        ; to final rate
                call    prog_AIC; (smallest divisor
                        ; should be last)
                ldi     @A_REG,R0
                call    prog_AIC
                b       main
;-----------------------------------------------------------------------------
prog_AIC        ldi     @S0_xdata,R1; Use original DXR data
                        ; during 2 ndy
                sti     R1,@S0_xdata
                idle
                ldi     @S0_xdata,R1; Use original DXR data
                        ; during 2 ndy
                or      3,R1; Request 2 ndy XMIT
                sti     R1,@S0_xdata
                idle
                sti     R0,@S0_xdata; Send register value
                idle
                andn    3,R1
                sti     R1,@S0_xdata; Leave with original safe
                        ; value in DXR
;-----------------------------------------------------------------------------
                ldi     @S0_rdata,R0; Fix the receiver
                        ; underrun by reading
                rets    ; the DRR before
                        ; going to the main loop
TABLA           .space  120
stack           .word   $; Put stack here
;
;*****************************************************************************
; Install the XINT/RINT ISR handler directly                                 *
; into the vector RAM location it will be used for                           *
;*****************************************************************************
                .start  "SP0VECTS",0x809FC5
                .sect   "SP0VECTS"
                B       DAC2; XINT0
                B       ADC2; RINT
```