

CAPÍTULO 4: PLAN DE PRUEBAS

Este capítulo describe inicialmente la metodología de pruebas seguida en el desarrollo del proyecto para la verificación de su funcionamiento. Seguidamente se detallan las pruebas individuales realizadas a nivel *software*.

4.1. ESTRATEGIA DE PRUEBAS

Para comprobar el correcto funcionamiento del simulador del DSP TMS320C30 EVM, se procede a realizar una serie de pruebas software y prácticas de laboratorio. Como ya se conocen los resultados de las prácticas realizadas sobre el DSP, se verificarán las formas de onda obtenidas a la salida con las que se obtienen en el TMS320C30 EVM.

4.2. PRUEBA 1: SEÑAL RETRASADA EN EL TIEMPO

Se trata de implementar una señal con un retraso τ y una atenuación α respecto a la señal de entrada.

El programa ensamblador que implementa los objetivos de la práctica es el siguiente:

```
bufent1          .sect    "buf_e1"
                  .space 100

delay            .word    10
input            .word    bufent1
p0_addr          .word    808040h

                .text
start:           ldi      @input, ar1
                  ldi      @delay, bk
wait_intr:       idle
                  br       wait_intr
receive0:        ldi      @p0_addr, ar0
                  ldi      *+ar0(12), r0
                  lsh      16, r0
                  ash      -18, r0
                  sti      r0, *ar1++%
                  ldi      *ar1, r1
                  float    r1, r1
                  mpyf     0.5, r1
                  fix      r1, r1
                  lsh      2, r1
                  sti      r1, *+ar0(8)
```

```
reti
```

```
.end
```

El funcionamiento del programa es el siguiente: la señal de entrada se saca a la salida con un retardo (10 muestras del *buffer* circular). La señal de salida es justamente la mitad de la señal de entrada (`mpyf 0.5, r1`).

Este programa en ensamblador produce bajo una ejecución continua del simulador, un fichero de muestras de enteros de 14 bits (como se ha comentado en el capítulo 3). Como fichero de muestras de entrada se ha utilizado una señal de 500 Hz muestreada a la máxima frecuencia del DAC (17361 Hz). El tiempo de ejecución elegido es de 0.01 segundos, que equivale a $0.01/(1/500) = 5$ períodos de señal de entrada. Como ya se comentó en el capítulo 3, se ha realizado un escalado de la señal de salida suponiendo que el AIC tiene un rango de voltaje de salida de $\pm 3V$.

En cuanto al retardo de 10 muestras, si se tiene un muestreo de 17361 muestras por segundo, equivaldrá a $10 \cdot 1/17361 = 5.76 \cdot 10^{-4} = 0.000576$ segundos, como puede verse en la figura 4.1.

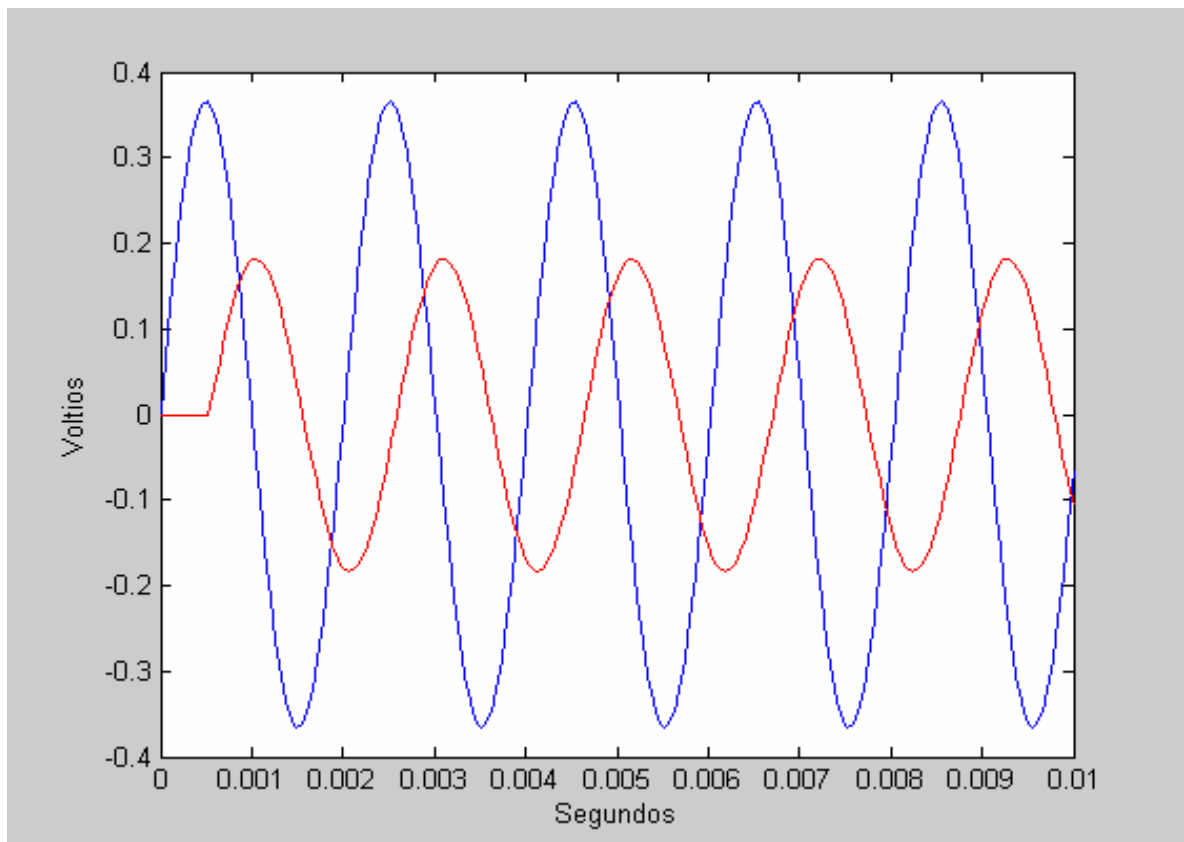


Figura 4.1. Señal de entrada (azul) a 500 Hz de frecuencia y señal de salida a 500 Hz retardada utilizando un muestreo de 17361 Hz.

Para la misma señal de entrada, si elegimos un intervalo de muestreo menor para el mismo tiempo de ejecución, obtendremos lo siguiente:

- Muestreo a 11574 Hz: se observa que la señal de salida empieza a perder calidad respecto a la entrada., debido a que el número de muestras que se utiliza es menor para el mismo intervalo de tiempo: $N = 11574/500 = 23.148$ muestras por período, sobre las 35 que se tienen con el muestreo máximo. Como el número de períodos de la señal de entrada es cinco, se tienen $5 \times 23.148 \approx 116$ muestras. Como se ha supuesto siempre que la frecuencia de conversión del ADC es la misma que el DAC, se debe tener en cuenta ahora que el número de muestras del retardo también se verá afectado proporcionalmente. El resultado es el que se muestra en la siguiente figura:

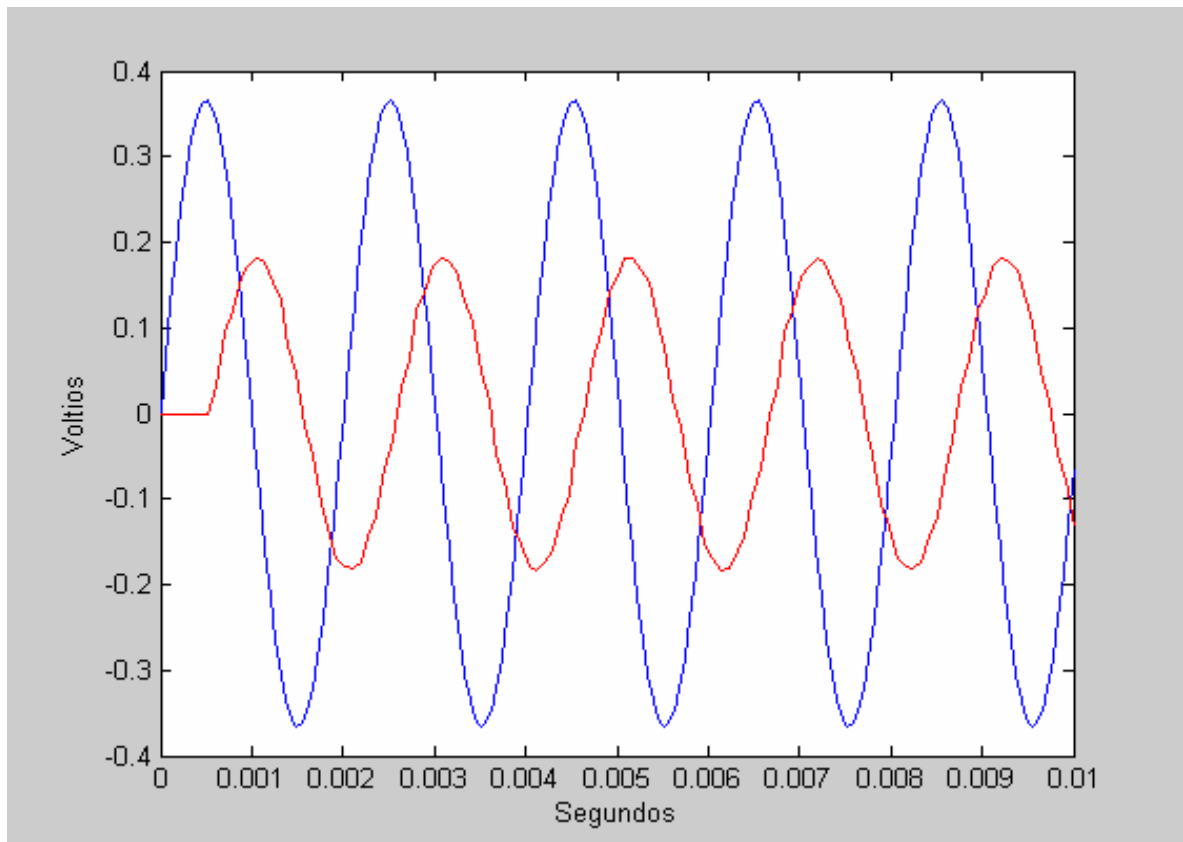


Figura 4.2. Señal de entrada (azul) a 500 Hz de frecuencia y señal de salida a 500 Hz retardada utilizando un muestreo de 11574 Hz.

- Muestreo a 3360Hz: a medida que se vaya disminuyendo la frecuencia de muestreo del AIC, disminuye también la calidad de la señal de salida. En el caso de un muestreo a 3360 Hz, es la peor calidad de las posibles. El número de

muestras que se cogen es menor para el mismo intervalo de tiempo: $N = 3360/500 = 6.72$ muestras por período, sobre las 35 que se tienen con el muestreo máximo. Como hay cinco períodos de la señal de entrada, el número de muestras de la señal de salida en ese intervalo es de aproximadamente 34. El resultado obtenido es el que muestra la siguiente figura:

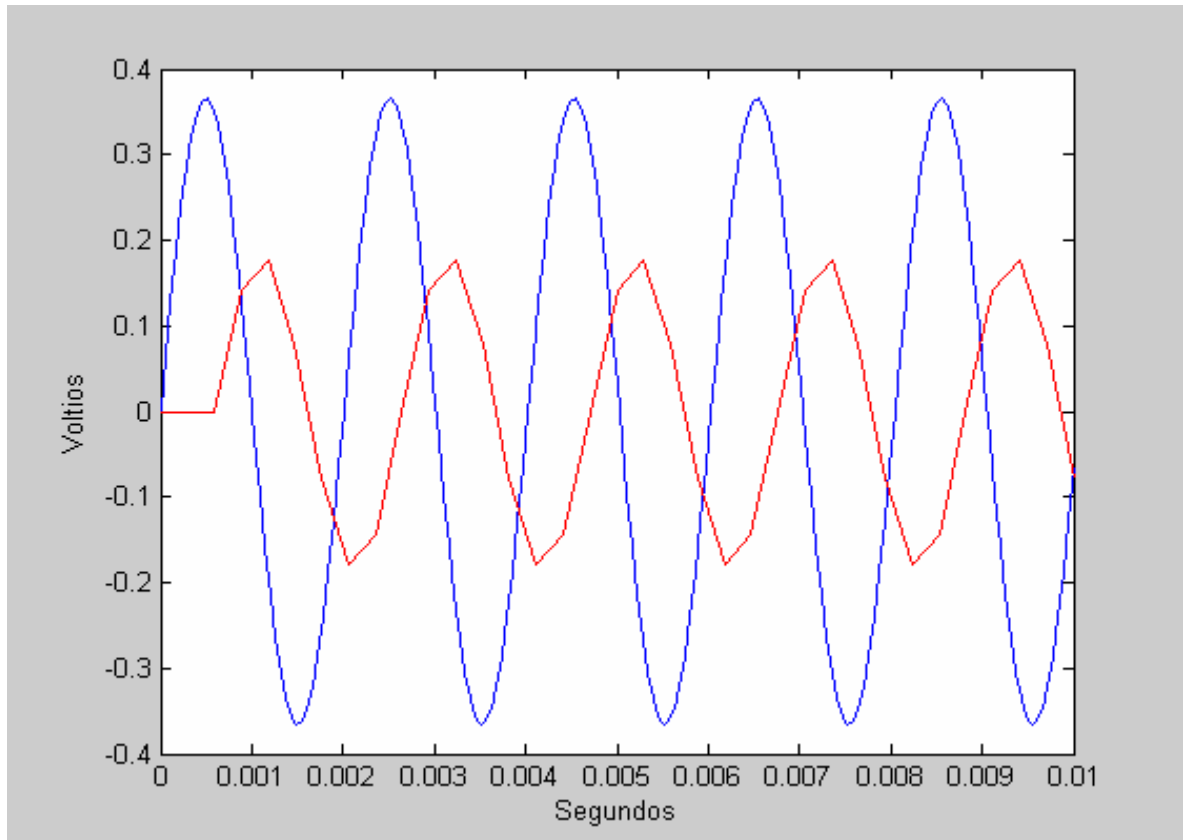


Figura 4.3. Señal de entrada (azul) a 500 Hz de frecuencia y señal de salida a 500 Hz retardada utilizando un muestreo de 3360 Hz.

4.3. PRUEBA 2: GENERADOR DE ONDA CUADRADA

Se pretende generar una onda cuadrada con el DSP TMS320C30 EVM sin necesidad de tener un fichero de entrada.

El programa ensamblador del DSP es el siguiente:

```
buffent    .sect "buf1"
           .space 5
```

```
p0_addr    .word    808040h

.text

start:      ldi      50,r1
            ldi      10000,r3

wait_intr:  idle
            br       wait_intr

receive0:   ldi      @p0_addr,ar0
            ldi      *+ar0(12),r6
            subi     1,r1
            BNZ      aqui
            negi     r3,r3
            ldi      100,r1

aqui:       ldi      r3, r0
            lsh      2,r0
            sti      r0, *+ar0(8)
            reti

.end
```

El funcionamiento del programa es el siguiente: cada 100 muestras, el valor de salida del DSP cambia entre un valor positivo y otro negativo de igual magnitud. El ancho temporal de la señal de salida depende del número de muestras de intervalo que se elija. En este caso se ha elegido un período de 200 muestras. El número de muestras en 0.1 segundos muestreando a 17361 Hz es: 1736.1 muestras. Por tanto el número de períodos en 0.1 segundos es $1736.1/200 = 8.6805$, lo que se puede comprobar en la Figura 4.1. El período de la señal será por tanto $0.1 / 8.6805 = 11.52$ mseg.

El resultado para una ejecución de 0.1 segundos es el que se muestra en la figura 4.4.

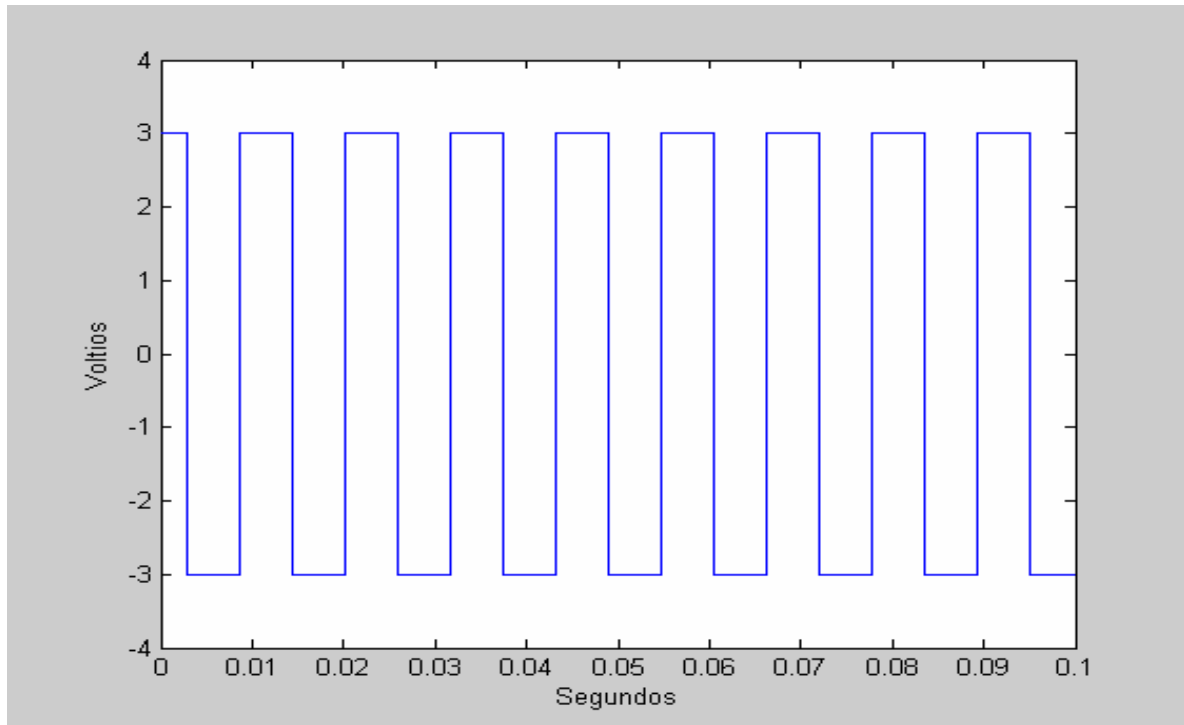


Figura 4.4. Onda cuadrada de período 11.52 mseg

4.4. PRUEBA 3: GENERADOR DE ONDA SENOIDAL

La siguiente ecuación diferencial de segundo orden produce una señal senoidal:

$$y[n] = B_1 * A_1 * y[n-1] + A_0 * y[n-2]$$

Ecuación 4.1.

Con $B_1=1$, $A_0=-1$, $A_1=2*\cos(\theta)$.

$$x[n] = \begin{cases} 1 & \text{si } n = 0 \\ 0 & \text{en otro caso} \end{cases}$$

La frecuencia de la señal generada es:

$$F = (q / 2 * \pi) * F_s = \frac{F_s}{2 * \pi} \arccos(A_1 / 2)$$

Ecuación 4.2.

Siendo F_s la frecuencia de muestreo. De esta ecuación se deduce que para cualquier frecuencia de muestreo, A_1 se debe tomar entre -2 y 2. La amplitud de la oscilación es $1/\sin(\theta)$ veces el impulso inicial $x[n]$.

Se trata de implementar un algoritmo que genere una señal senoidal en la salida del DSP.

El programa ensamblador del DSP es el siguiente:

```

buf_addr      .sect    "inp"
buffent       .sect "buf1"
               .space 5
p0_addr       .word    808040h
input         .word    809800h
B1            .word    1
A0            .word    -1
A1            .word    1
amp           .word    1000
buffent2      .sect "buf2"
               .space 5

.text

start:        ldi        @input, ar1
               ldi        2h,bk
               ldi        @A1,r6
               ldi        @A0,r7
               ldi        1,r5

wait_intr:    idle
               br         wait_intr

receive0:     ldi        @p0_addr,ar0
               cmpi       0,r5
               bz         sigue
               subi       1,r5
               ldi        @amp,r0
               sti        r0,*ar1
               sti        r0,*+ar0(8)
               reti

sigue:        ldi        *ar1++,r1
               ldi        *ar1,r2
               mpyi3      r6,r1,r3
               mpyi3      r7,r2,r4
               addi3      r3,r4,r0
               sti        r1,*ar1++%
               sti        r0,*ar1
               sti        r0,*+ar0(8)
               reti

               .end

```

Se puede apreciar que la señal senoidal tiene 6 muestras por periodo. Suponiendo que el DAC este configurado a una velocidad de muestreo de 17361 Hz, en 0.005 segundos se convierten 86.805 muestras digitales. Esto quiere decir que en 0.005 segundos tendremos 14.4675 períodos, como se puede ver en la figura 4.5. Por tanto, la señal tiene un periodo de 0.3456 mseg.

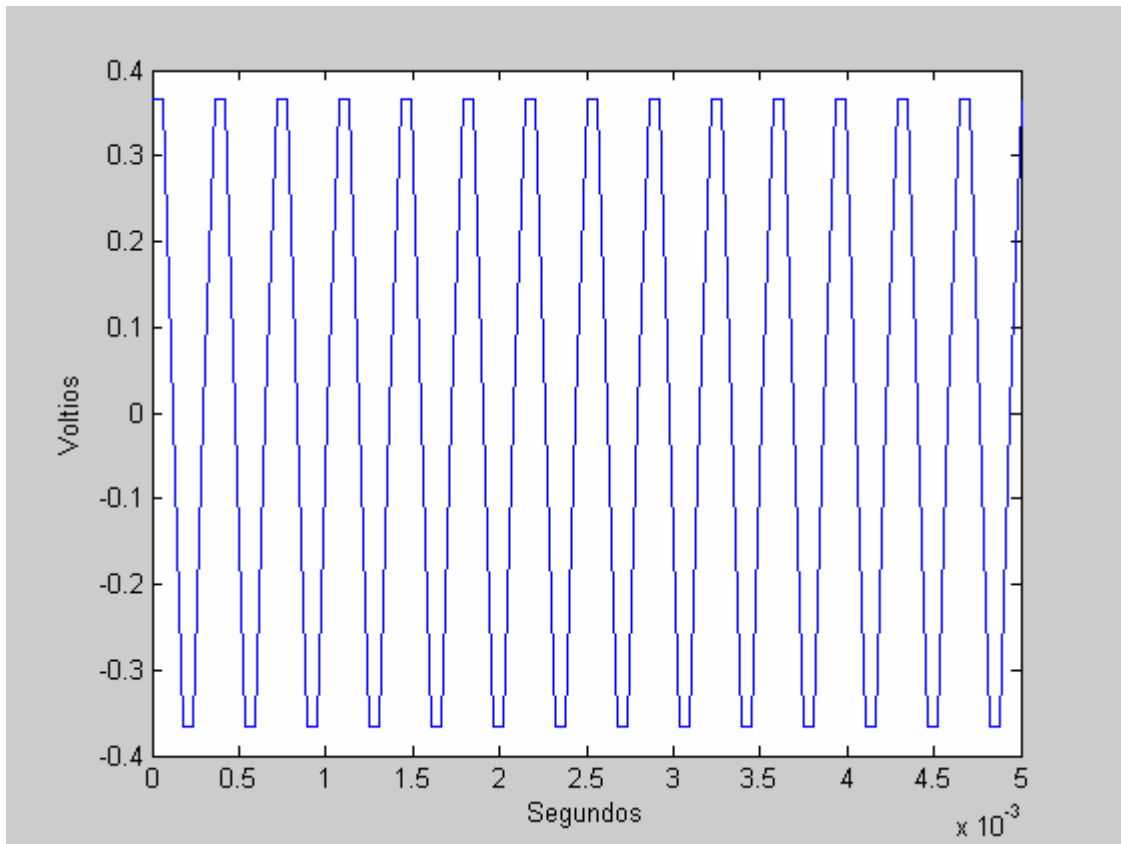


Figura 4.5. Onda cuadrada de período 11.52 mseg.

Una vez que pasen las muestras por el DAC, la forma de la señal de salida tendrá más similitud con la forma senoidal.

4.5. PRUEBA 4: GENERADOR DE RUIDO

Para generar un número aleatorio pseudo uniforme se utilizará el método de la congruencia lineal, que usa la siguiente recursión:

$$x[n+1] = 65539 * x[n] \bmod 2^{31}$$

Ecuación 4.3.

Si como valor de $x[0]$, el valor inicial de $x[n]$, se elige cualquier número entero impar con nueve o menos cifras, se generan todos los 2^{31} posibles enteros antes de que se repita la secuencia. La secuencia, por tanto, aproxima a una variable uniformemente distribuida entre 0 y $2^{31}-1$. Un número aleatorio en punto flotante, X , uniformemente distribuido, en el rango 0 a 1.0 se puede obtener escalando el número aleatorio entero con la ecuación:

$$X = x[n+1] * 4.656612875245797 * 10^{-10}$$

Ecuación 4.4

Se puede encontrar una aproximación a una distribución gaussiana para el número aleatorio Y calculando la secuencia de números aleatorios uniformes mediante la fórmula:

$$Y = \frac{X_1 + X_2 + \dots + X_K - \frac{K}{2}}{\sqrt{\frac{K}{12}}}$$

Ecuación 4.5.

Donde:

- X_i son números aleatorios uniformemente distribuidos obtenidos de la ecuación 4.4.
- K es el número de valores de X_i usados en la suma.
- Y se aproxima a una distribución gaussiana asintótica conforme K se aproxima a infinito.

Para obtener la mediana y la desviación estándar deseada, hay que escalar la variable aleatoria Y:

$$Y' = Y * \sigma + \mu.$$

Ecuación 4.6.

Donde Y' es la variable aleatoria con distribución gaussiana deseada con desviación estándar σ y mediana μ .

Usando las ecuaciones 4.3, 4.4, 4.5 y 4.6, se implementa un generador de ruido usando $K=12$, $\sigma=800$ y $\mu=0$.

El programa implementado con las condiciones anteriormente descritas es el siguiente:

```

buf_xmay          .sect   "xmay"
                   .space  16

p0_addr           .word   808040h
xmayus            .word   buf_xmay
lbuffer           .word   12
kentredos         .word   6
cuatro            .float  4.656612875245797e-10
nimpar            .word   12345
modulo            .word   7FFFFFFFh
tita              .word   800

.text
start:            ldi      @xmayus, ar1
                   ldi      @lbuffer, bk
                   ldi      @nimpar, r4
                   ldf      0.0, r2
                   rpts     11
    
```

```

                                stf                r2,*ar1++%

wait_intr:                    idle

                                br                wait_intr

receive0:                     ldi                @p0_addr,ar0
                                ldi                *+ar0(12),r0
                                ldi                r4,r0
                                ldi                r0,r1
                                lsh                16,r0
                                mpyi               3, r1
                                addi               r1,r0
                                and                @modulo,r0
                                ldi                r0,r4
                                float              r0,r0
                                mpyf               @cuatro,r0
                                stf                r0,*ar1++%
                                rpts               11
                                addf               *ar1++%,r3
                                fix                r3,r3
                                subi               @kentredos,r3
                                mpyi               @tita,r3
                                lsh                2,r3
                                sti                r3,*+ar0(8)
                                reti

                                .end

```

El resultado obtenido ejecutando este programa ensamblador en el simulador es el mostrado en la figura 4.6, para un tiempo de simulación de 0.1 segundos.

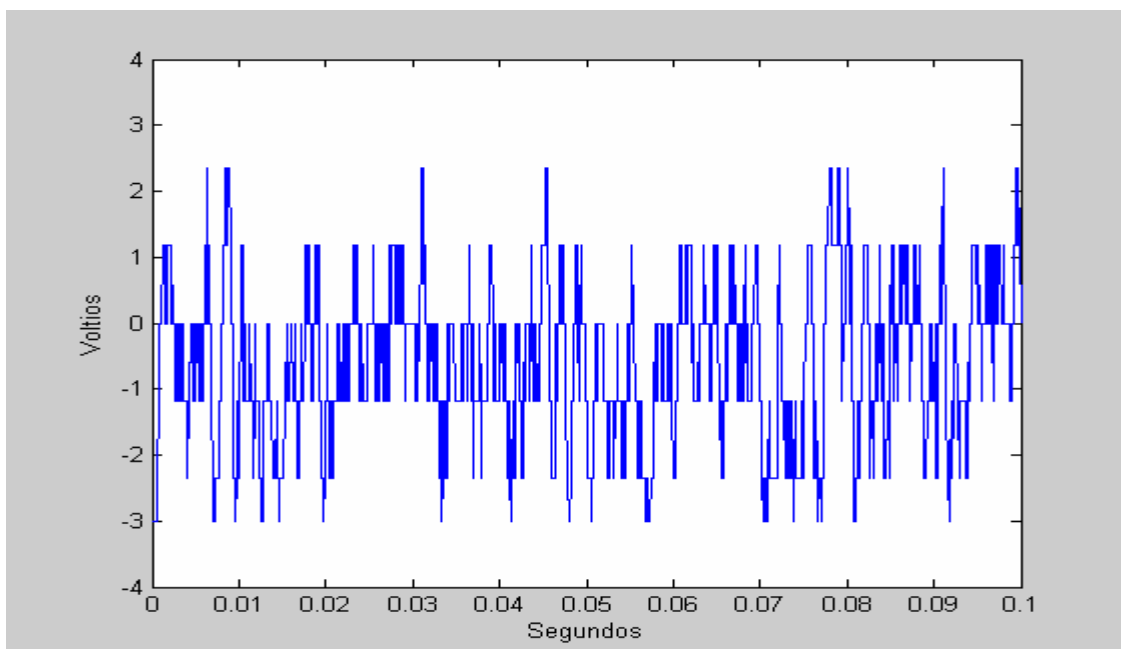


Figura 4.6. Señal ruidosa

Como puede verse en la figura 4.6. es la señal pseudo aleatoria que se buscaba.

4.6. PRUEBA 5: FILTRO FIR

En esta prueba se implementa un filtro FIR paso banda de longitud 7. La banda de paso se especifica desde 0 a 1.2 kHz.

Como ya se ha comprobado en la prueba 1, para una señal de entrada con ruido de frecuencia 500 Hz, en 0.01 segundos de simulación hay 5 periodos de señal, que corresponde con 173 muestras.

La señal de entrada senoidal con ruido es la que muestra la figura 4.7.:

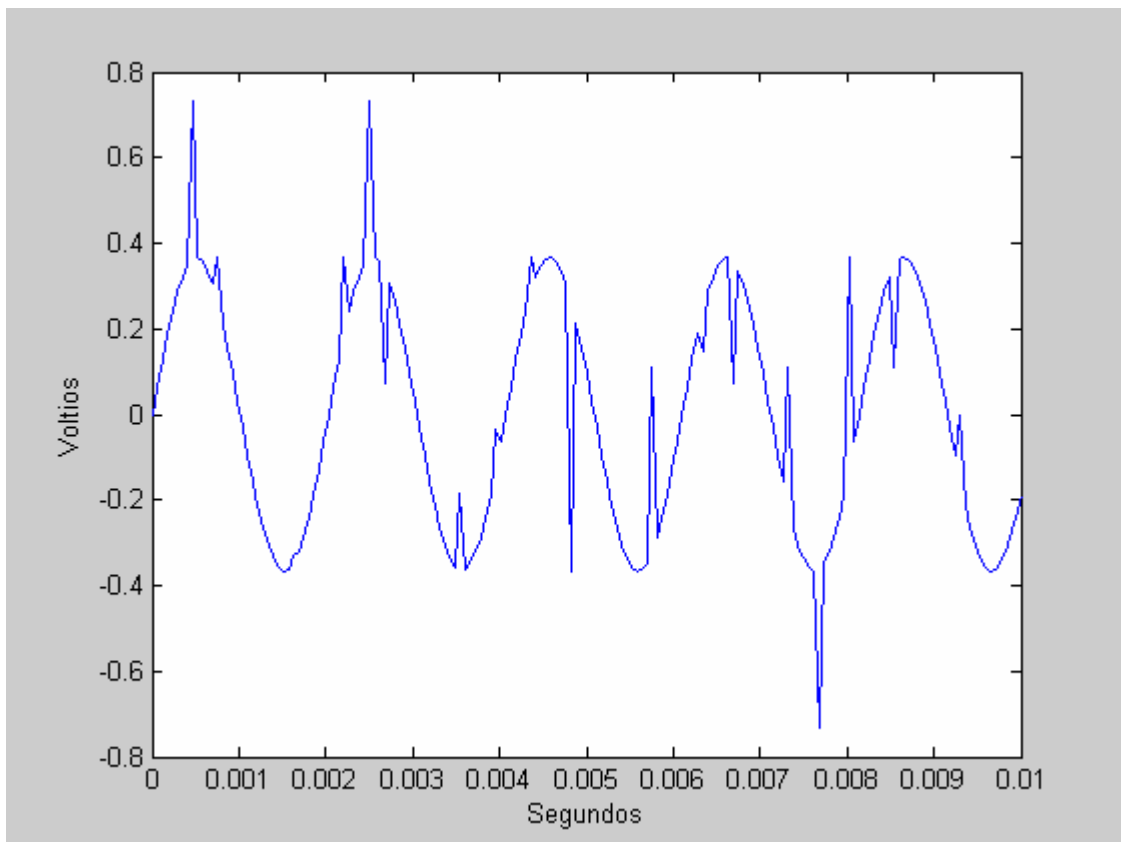


Figura 4.7. Onda senoidal ruidosa de 500 Hz.

El programa ensamblador que implementa el filtrado FIR es el siguiente:

```
bufent1      .sect    "buf_e1"  
             .space 9
```

Implementación de un simulador del DSP TMS320C30 EVM

```

bufcoef      .float      2.770772250334e-2, 1.279134021017e-1,
                        2.176192777938e-1, 2.535191952023e-1,
                        2.176192777938e-1,1.279134021017e-1,
                        2.770772250334e-2

p0_addr      .word      808040h

cont          .word      7

entrada      .word      bufent1
coef         .word      bufcoef

        .text
start:       ldi         @entrada, ar1
              ldi         @coef, ar2
              ldi         7h,bk

wait_intr:   idle
              br          wait_intr


receive0:    ldi         @p0_addr,ar0
              ldi         *+ar0(12),r0
              lsh         16,r0
              ash         -18,r0
              sti         r0, *ar1++%
              ldi         0,r0
              ldi         @cont,r4

fir:         ldf         *ar2++%,r2
              ldi         *ar1++%,r6
              float       r6,r6
              mpyf3       r2,r6,r1
              fix         r1,r1
              addi        r1,r0
              subi        1,r4
              bnz         fir
              lsh         2,r0
              sti         r0,*+ar0(8)
              reti
        .end

```

Los resultados obtenidos para una simulación de 0.01 segundos se muestran en la figura 4.8:

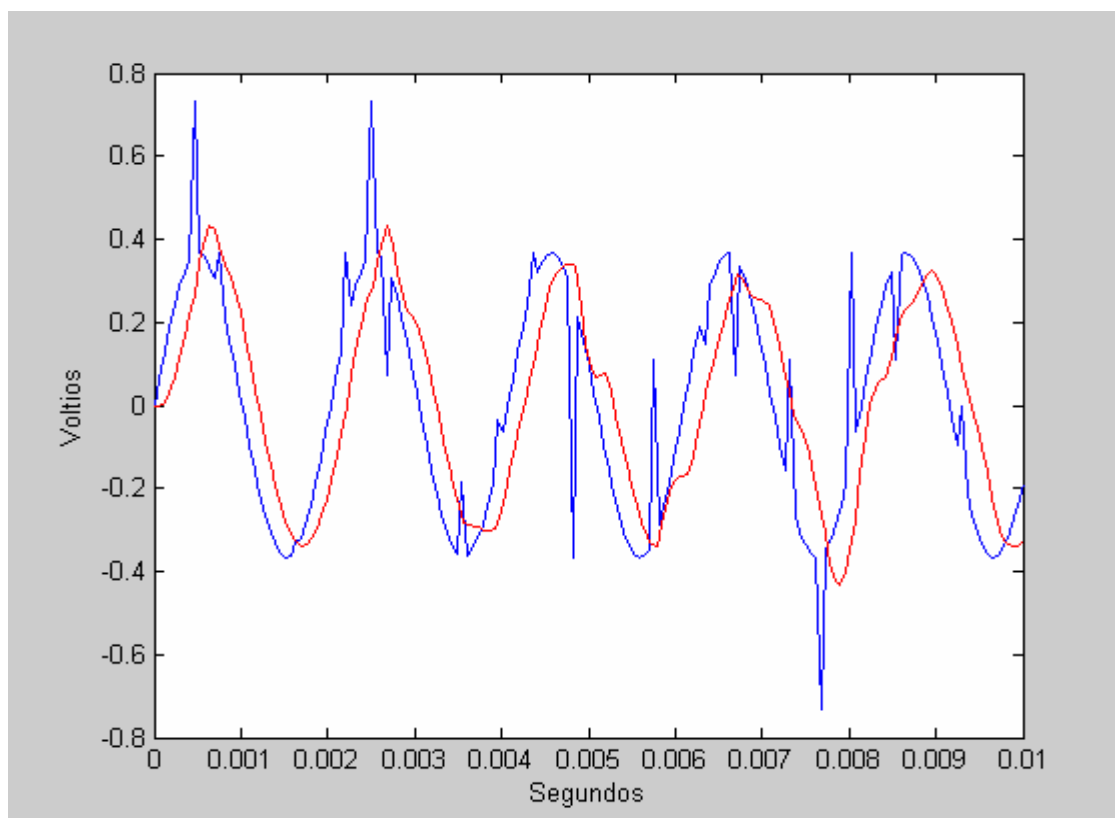


Figura 4.8. Onda senoidal ruidosa de 500 Hz (azul) y señal filtrada (rojo).

Puede verse claramente como la señal ruidosa ha sido filtrada (rojo). El retardo de la señal filtrada con respecto a la de entrada se debe al uso del *buffer* circular, que retarda la salida de muestras.

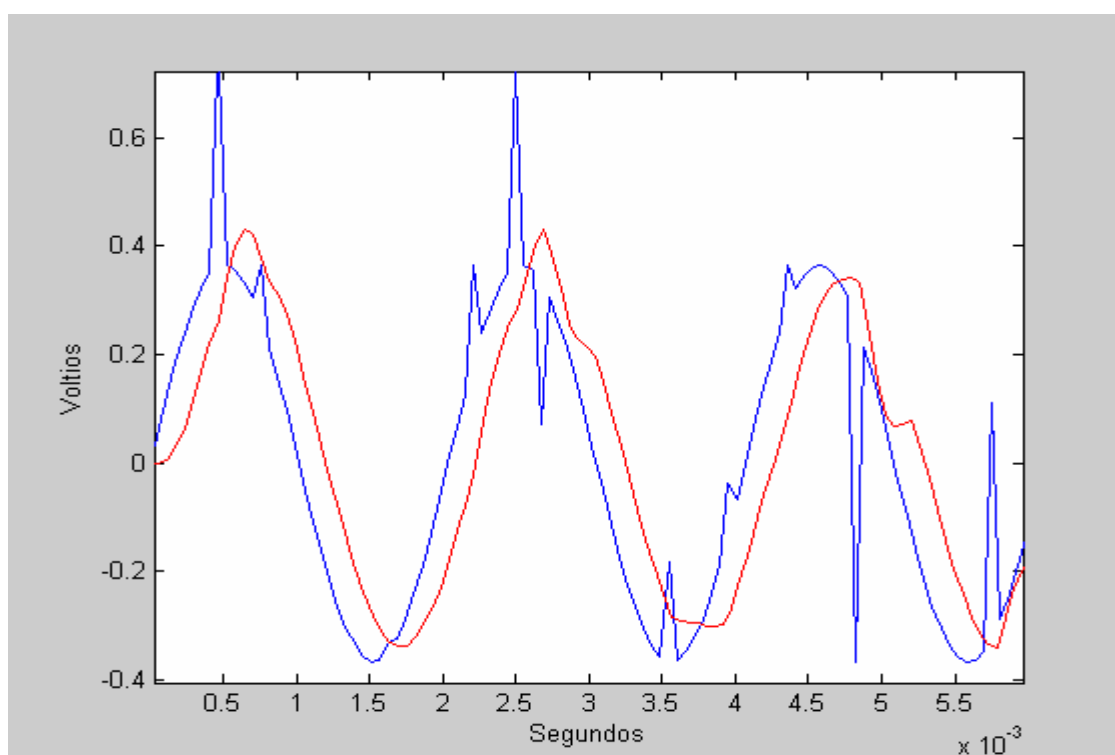


Figura 4.9. Ampliación de la onda senoidal ruidosa de 500 Hz (azul) y señal filtrada (rojo).

4.7. PRUEBA 6: FILTRO IIR

A continuación se implementa un filtro IIR de orden 4, con una banda de paso desde 0 a 1.2 kHz y una frecuencia de muestreo de 8kHz dado por la función:

$$H(z) = \frac{z^4 + 4z^3 + 6z^2 + 4z + 1}{z^4 - 2.5907z^3 + 3.1119z^2 - 1.9198z + 0.5166}$$

Como la señal que se utiliza es de 500 Hz, esta frecuencia se encuentra dentro de la banda de paso.

El programa ensamblador que realiza esta prueba es el siguiente:

```
bufent1      .sect      "buf1"
              .space    5
bufent3      .sect      "buf3"
              .space    4
buf2         .word      1,4,6,4,1
buf4         .float     -0.5166e0,0.191987e1,-0.31119e1,0.25907e1
gan          .float     0.74e-2
input1       .word      bufent1
input2       .word      buf2
input3       .word      bufent3
input4       .word      buf4
contnum      .word      5
contden      .word      4

p0_addr      .word      808040h

.text
start:       ldi        @input1, ar1
              ldi        @input2, ar2
              ldi        @input3, ar3
              ldi        @input4, ar4
              ldi        5,bk

wait_intr:   idle
              br         wait_intr

receive0:    ldi        @p0_addr, ar0
              ldi        *+ar0(12), r0
              lsh        16, r0
              ash        -18, r0
              sti        r0, *ar1++%
              ldi        0, r0
              ldi        @contnum, r4
bpx:         mpyi3      *ar2++%, *ar1++%, r1
              addi       r1, r0
```

```

subi    1,r4
bnz     bpx

ldi     @contden,r4
apy:    ldf     *ar4++,r2
fix     r2,r2
mpyi3   r2,*ar3++,r1
addi    r1,r0
subi    1,r4
bnz     apy
ldf     *ar4++,r2
float   r0,r0
mpyf    @gan,r0
fix     r0,r0
sti     r0,*ar3++
lsh     2, r0
sti     r0, *+ar0(8)
reti

.end

```

Los resultados obtenidos para una señal de entrada de 500 Hz ruidosa y un tiempo de ejecución de 0.01 segundos son los siguientes:

- Para una frecuencia de muestreo de 17361Hz:

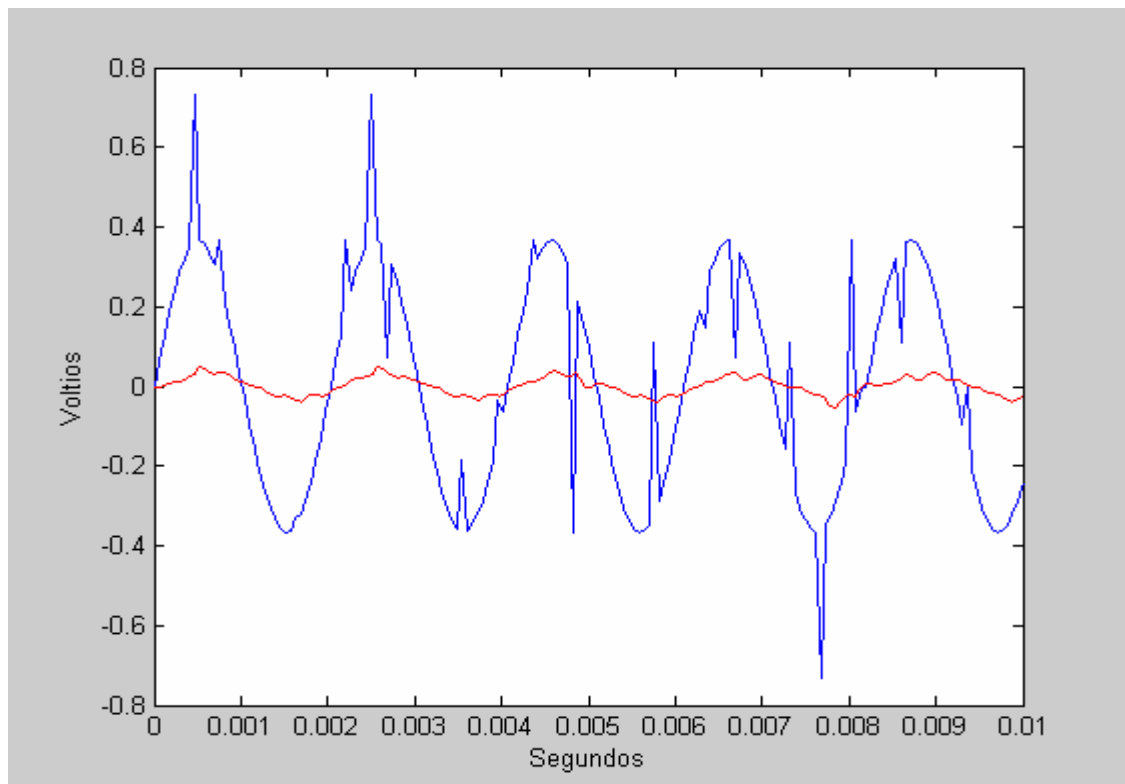


Figura 4.10. Onda senoidal ruidosa de 500 Hz (azul) y señal filtrada (rojo) con un muestreo de 17361Hz.

- Para una frecuencia de muestreo de 8 KHz, el resultado será:

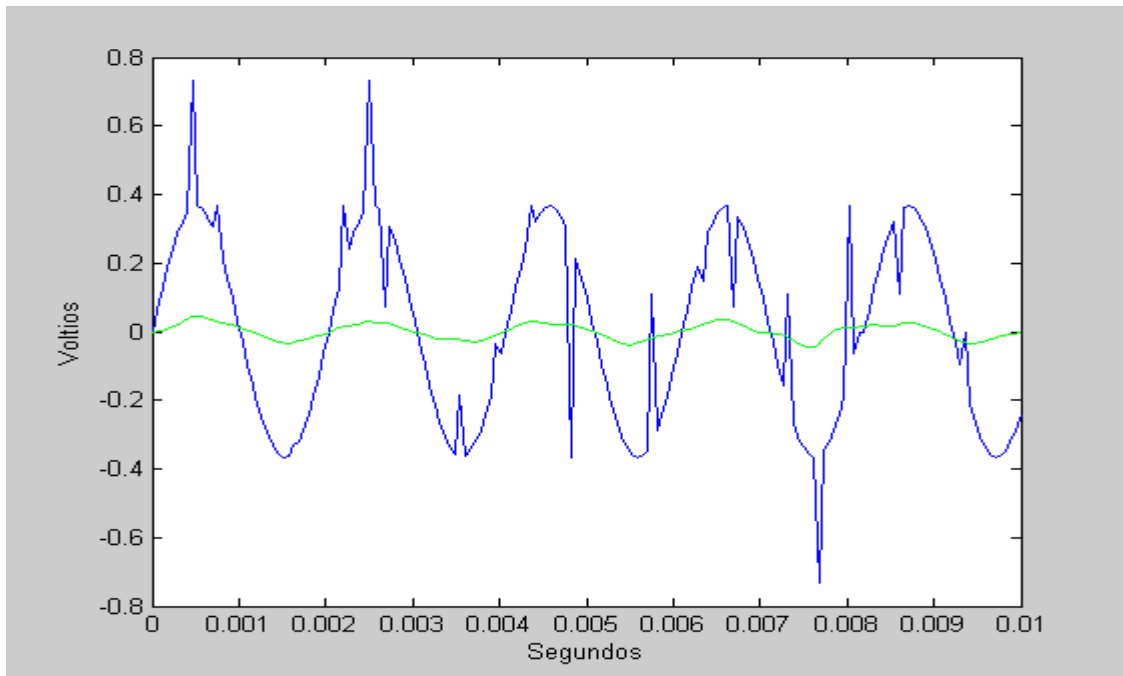


Figura 4.11. Onda senoidal ruidosa de 500 Hz (azul) y señal filtrada (verde) con un muestreo de 8kHz.

Para una frecuencia de muestreo de 17361 Hz, se muestra un rango más amplio de tiempos donde se aprecia mejor el filtrado:

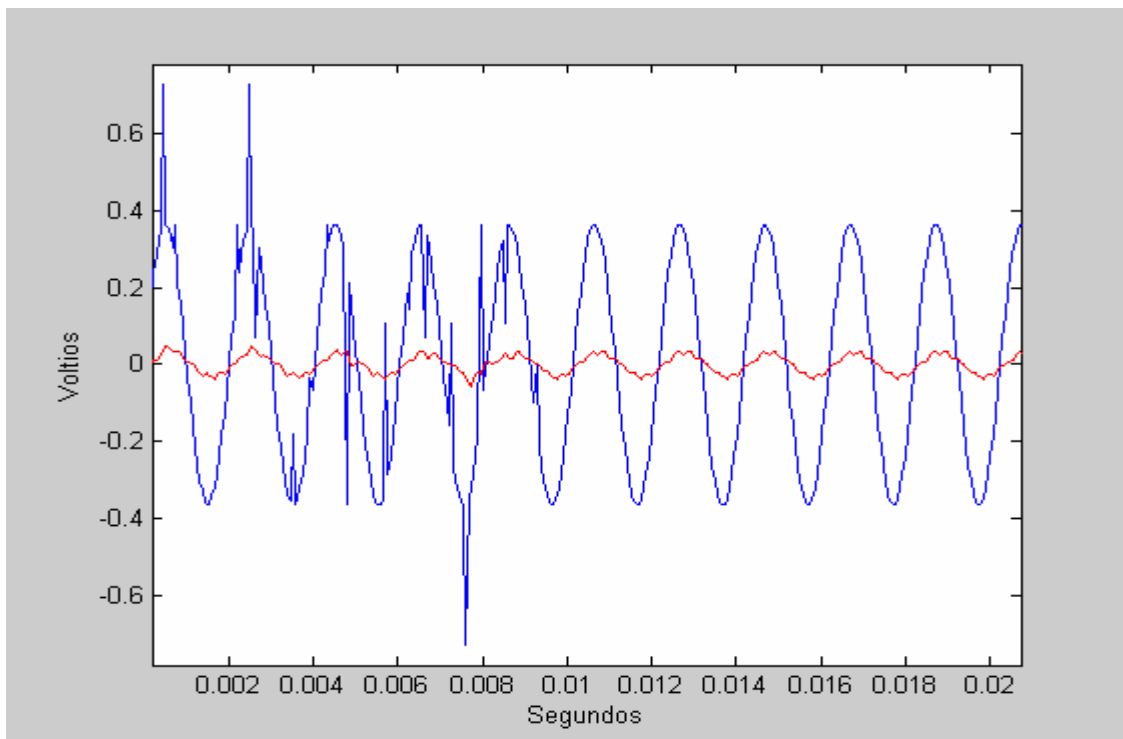


Figura 4.12. Onda senoidal ruidosa de 500 Hz (azul) y señal filtrada (rojo) con un muestreo de 17361Hz

Con esto se da por concluido el plan de pruebas, una vez demostrado el correcto funcionamiento del simulador.