

Procesamiento de señales, fundamentos

.....

Maestría en sistemas embebidos

Universidad de Buenos Aires

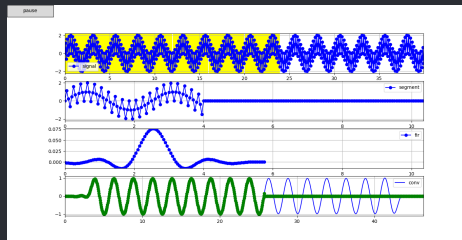
MSE 5Co2020

Clase 6 - Filtrado con DFT

Ing. Pablo Slavkin

slavkin.pablo@gmail.com

wapp:011-62433453



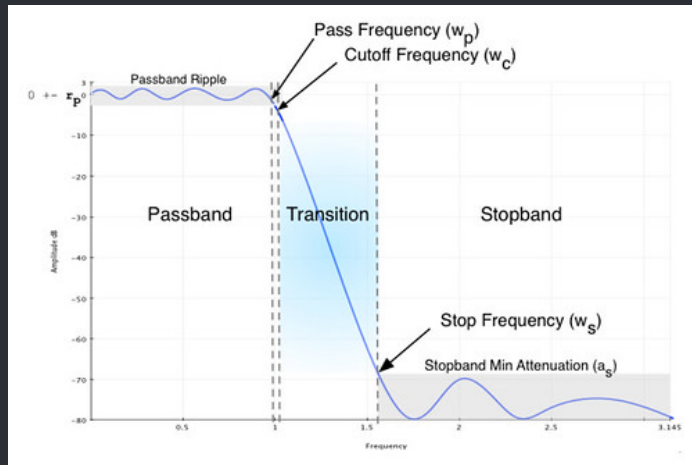
Filtrado

Definición



07m25s

- Plantilla de diseño de un filtro
- En el ejemplo se aprecia un pasabajos pero se destacan las zonas de interés y los niveles de la banda de paso y de rechazo
- Cuanto mas exigente se la plantilla del filtro, mas puntos tendrá nuestra $h(n)$ y mas lenta y compleja su convolución
- El objetivo es llegar a un compromiso entre los requisitos y la performance



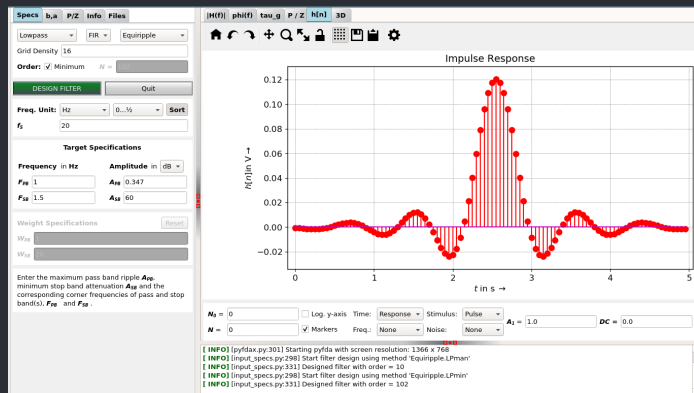
Filtrado

PyFDA `/opt/anaconda3/bin/pyfdax`



37m52s

- Uso de PyFDA como herramienta para diseño de filtros
- Inicialmente solo nos concentramos en la $H(f)$ para visualizar de manera practica las zonas de paso y de rechazo



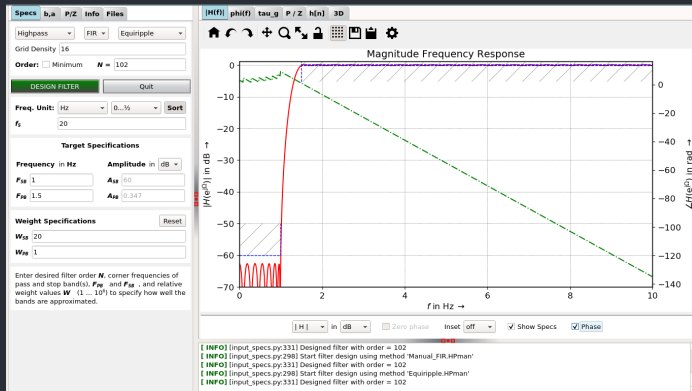
Filtrado

Pyfda `/opt/anaconda3/bin/pyfdax`



37m52s

- Uso de PyFDA como herramienta para diseño de filtros
- Inicialmente solo nos concentramos en la $H(f)$ para visualizar de manera practica las zonas de paso y de rechazo
- Notar la variedad de opciones disponibles y la respuesta en fase en esta imagen



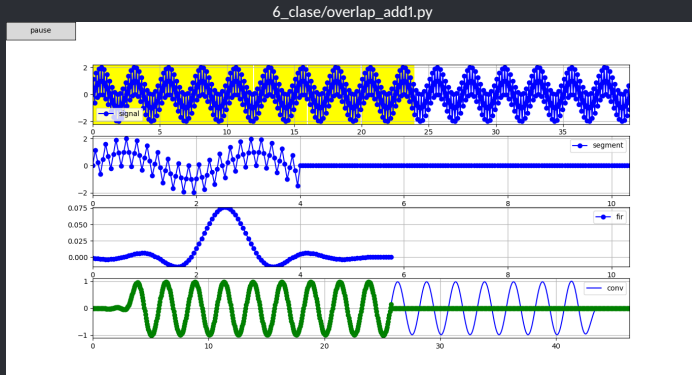
Convolución

Superponer y sumar en t



40m30s

- Ejemplo de como particionar la $x(n)$ en tramos y filtrar usando convolucion en tiempo
- Se convolciona $x(n)$ con $h(n)$ ambas con zero padding hata que cada una tenga $N+M-1$ puntos
- Notar el efecto del inicio del filtrado en cada tramo como se corrige con el siguiente
- Esto es equivalente a tomar la $x(n)$ completa y convolucionar con la $h(n)$
- Si $x(n)$ es una señal muy larga o en tiempo real, convolucionar de una sola vez se vuelve impracticable
- Con este metodo podemos mostrar cada cierto tiempo como va evolucionando la salida



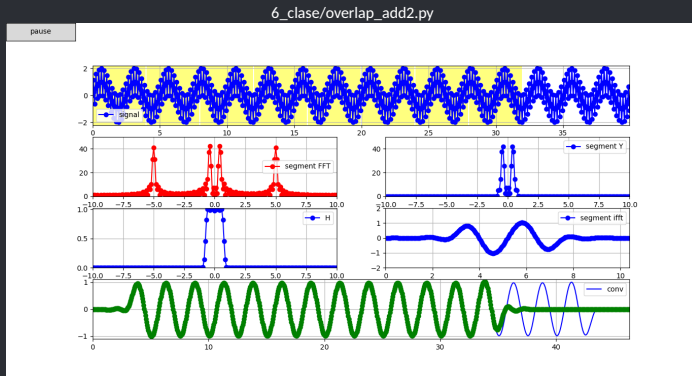
Convolución con FFT

Superponer y sumar en f



45m25s

- Ejemplo de como particionar la $x(n)$ en tramos y filtrar usando $\text{IFFT}(\text{fft}(x\text{-padd}) * \text{fft}(h\text{-padd}))$
- Se hace zero padding a $x(n)$ y $h(n)$ hasta que ambas tengan $N+M-1$ datos
- Se denera elegir N de modo tal que $N+M-1$ sea potencia de 2 para que la eficiencia del algoritmo FFT sea maxima
- Se calcula $\text{DFT}(x)=X$ y $\text{DFT}(h)=H$
- Se calcular $Y=X$ multiplicado H
- Se antitransforma Y y se obtiene $y(n)$



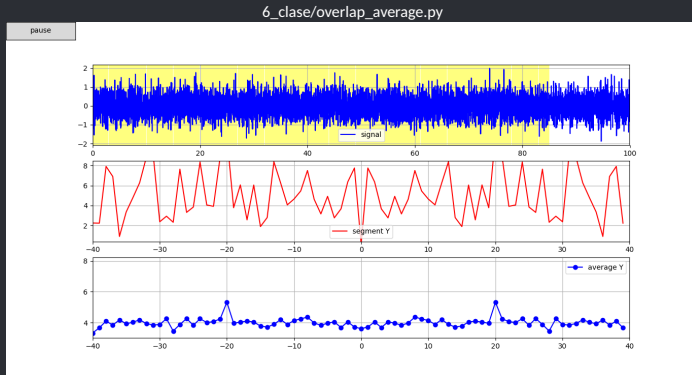
Promedio en magnitud de FFT

Superponer y promediar en f



49m00s

- Ejemplo de como particionar la $x(n)$ en tramos luego calcular $\text{fft}(x)$ y promediar
- Se hace un promedio de la magnitud de los espectros de cada segmento
- Se logra promediar el ruido y destacar las componentes con informacion incluso sumergidas en ruido



Filtrado con CIAA

Conversor PyFDA a fir.h para C

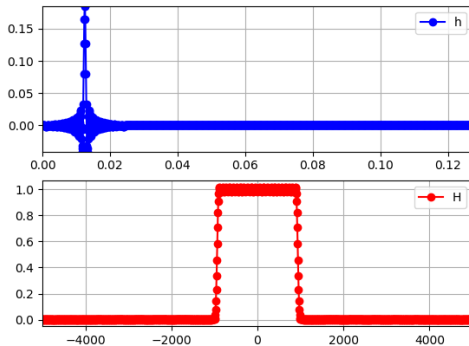


0h29m00s



- Código en Python para convertir los coeficientes del fir extendidos en Q1.15 en C
- Util para pasar de PyFda a los codigos de C

6_clase/fir_to_c.py



Filtrado con CIAA

Convolución en tiempo con padding



59m10s



```
#include "sapi.h"
#include "arm_math.h"
#include "arm_const_structs.h"
#include "fir.h"
#define MAX_FFT_LENGTH 2048
int16_t fftLength = 512;
int16_t hLength = h_LENGTH;
int16_t adc [ MAX_FFT_LENGTH];
q15_t x [ MAX_FFT_LENGTH] ;
q15_t fftOut [ ( MAX_FFT_LENGTH)*2 ] ;
q15_t fftMag [ ( MAX_FFT_LENGTH)/2+1 ] ;

uint32_t maxIndex = 0;
q15_t maxValue = 0;
arm_rfft_instance_q15 S;
uint16_t convLength = 0;
uint16_t sample = 0;

int calcFftLength(int N,int M) {
    int convLength=N-M+1;
    for(i=MAX_FFT_LENGTH;i>=convLength;i++)
        ;
    return i<=1;
}

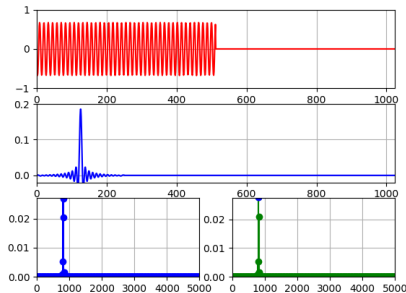
int sendStr(char A[],int N) { uartWriteByteArray (
    UART_USB ,A ,N ); }

int sendBlock(q15_t A[],int N) { uartWriteByteArray
    ( UART_USB ,(uint8_t*) A ,2*N ); }

int main ( void ) {
    boardConfig (
        uartConfig ( UART_USB, 460800
    );
}
```

```
cyclesCounterInit ( EDU_CIAA_NXP_CLOCK_SPEED );
while(1) {
    convLength=calcFftLength(fftLength,hLength);
    for(sample=0;sample<fftLength;sample++) {
        cyclesCounterReset();
        adc[sample] = ((int16_t )adcRead(CH1)-512)
            <<6;
        gpioToggle( LEDB);
        while(cyclesCounterRead())< 20400)
            ;
    }
    for(sample=fftLength;sample<convLength;sample
        ++){
        adc[sample]=0;
        sendStr ( "header" ,6 );
        sendBlock ( &fftLength ,1 );
        sendBlock ( &convLength ,1 );
        sendBlock ( adc ,convLength );
        sendBlock ( h ,convLength );
        arm_conv_fast_q15 ( adc,fftLength,h,
            convLength-fftLength+1,x );
        arm_rfft_init_q15 ( &S ,convLength ,0 ,1 );
        arm_rfft_q15 ( &S ,x ,fftOut );
        arm_cmplx_mag_squared_q15 ( fftOut ,fftMag ,
            convLength/2+1 );
        arm_max_q15 ( fftMag ,
            convLength/2+1 ,&maxValue ,&maxIndex
        );
        sendBlock ( fftOut ,convLength );
        sendBlock ( &maxValue ,1 );
        sendBlock ( (q15_t*)&maxIndex ,1 );
        gpioToggle( LEDR);
    }
}
```

6_clase/ciaa/psf1/src/conv1.c



Filtrado con CIAA

Multiplicacion en frecuencia con padding



1h18m38s



```
#include "sapi.h"
#include "arm_math.h"
#include "arm_const_structs.h"
#include "fir.h"

#define MAX_FFT_LENGTH 1024
int16_t fftLength = 128;
int16_t hLength = h_LENGTH;
int16_t adc [ MAX_FFT_LENGTH ];
q15_t x [ MAX_FFT_LENGTH ];
q15_t fftOut[ 2* MAX_FFT_LENGTH ];
q15_t H [ 2* MAX_FFT_LENGTH ];
q15_t hTemp [ 2* MAX_FFT_LENGTH ];

uint32_t maxIndex = 0;
q15_t maxValue = 0;
arm_rfft_instance_q15 S;
uint16_t convLength = 0;
uint16_t sample = 0;

int calcFftLength(int N,int M) {
    int convLength=N*M-1,i;
    for(i=MAX_FFT_LENGTH;i=convLength;i++)
        ;
    return i<1;
}

int sendStr(char A[],int N) { uartWriteByteArray ( UART_USB ,A
,N ); }

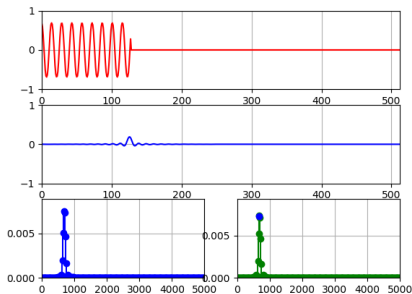
int sendBlock(q15_t A[],int N) { uartWriteByteArray ( UART_USB
,(uint8_t*) A ,2*N ); }

int main ( void ) {
    boardConfig (
        UART_USB , 460800 );
    uartConfig (
        ADC_ENABLE );
    cyclesCounterInit ( EDU_CIAA_NXP_CLOCK_SPEED );
```

```
while(1) {
    convLength=calcFftLength(fftLength,hLength);
    for(sample=0;sample<fftLength;sample++) {
        cyclesCounterReset();
        adc[sample] = ((int16_t) adcRead(CH1)-512)<<6;
        gpioToggle( LEDB);
        while(cyclesCounterRead()< 20480)
            ;
    }
    for(sample=fftLength;sample<convLength;sample++)
        adc[sample]=0;

    sendStr ( "header" ,6 );
    sendBlock ( &fftLength ,1 );
    sendBlock ( &convLength ,1 );
    sendBlock ( adc ,convLength );
    sendBlock ( h ,convLength );
    arm_rfft_init_q15 ( &S ,convLength ,0 ,1 );
    arm_rfft_q15 ( &S ,adc ,fftOut );
    for(int i=0;i<convLength;i++)
        hTemp[i]=h[i];
    arm_rfft_init_q15 ( &S ,convLength ,0 ,1 );
    arm_rfft_q15 ( &S ,hTemp ,H );
    for(int i=0;i<convLength;i++)
        H[i]=H[i]*convLength;
    arm_cmplx_mult_cmplx_q15(fftOut,H,H,convLength);
    sendBlock ( H ,convLength );
    arm_cmplx_mag_squared_q15 ( H ,H ,convLength/2+1
        );
    arm_max_q15 ( H ,convLength/2+1 ,&maxValue
        ,&maxIndex );
    sendBlock ( &maxValue ,1 );
    sendBlock ( (q15_t*) &maxIndex ,1 );
    gpioToggle( LEDR);
}
```

6_clase/ciaa/psf2/src/conv1.c



Enuestas

Encuesta anónima clase a clase

Propiciamos este espacio para compartir sus sugerencias, críticas constructivas, oportunidades de mejora y cualquier tipo de comentario relacionado a la clase.

Encuesta anónima



<https://forms.gle/1j5dDTQ7qjVfRwYo8>

Link al material de la material



https://drive.google.com/drive/u/1/folders/1TIR2cgDPchL_4v7DxdpS7pZHtjKq38CK

Bibliografía

Libros, links y otro material

[1] *ARM CMSIS DSP.*

https://arm-software.github.io/CMSIS_5/DSP/html/index.html

[2] Steven W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. Second Edition, 1999.

[3] *Wikipedia.*

https://en.wikipedia.org/wiki/Convolution_theorem