Carrera de Especialización / Maestría de Sistemas Embebidos
Universidad de Buenos Aires
Arquitectura de Microprocesadores

Guía de Ejercicios

Se pide realizar los ejercicios primeros en C y luego en assembler, verificando posteriormente que el algoritmo funcione correctamente y tomando nota del tiempo de ejecución en ciclos de cada función y su diferencia.

1) Realizar una función que inicialice un vector con ceros. La función debe tener el siguiente prototipo:

void zeros (*uint32_t* * vector, *uint32_t* longitud);

2) Realizar una función que realice el producto de un vector y un escalar (por ejemplo, podría servir para cambiar el nivel de amplitud de una señal).

void productoEscalar32 (*uint32_t* * vectorIn, *uint32_t* * vectorOut *uint32_t* longitud, *uint32_t* escalar);

3) Adapte la función del ejercicio 2 para realizar operaciones sobre vectores de 16 bits:

void productoEscalar16 (*uint16_t* * vectorIn, *uint16_t* * vectorOut, *uint32_t* longitud, *uint16_t* escalar);

4) Adapte la función del ejercicio 3 para saturar el resultado del producto a 12 bits:

void productoEscalar12 (*uint16_t* * vectorIn, *uint16_t* * vectorOut, *uint32_t* longitud, *uint16_t* escalar);

5) Realice una función que implemente un filtro de ventana móvil de 10 valores sobre un vector de muestras.

void filtroVentana10(*uint16_t* * vectorIn, *uint16_t* * vectorOut, *uint32_t* longitudVectorIn);

6) Realizar una función que reciba un vector de números signados de 32 bits y los “empaquete” en otro vector de 16 bits. La función deberá adecuar los valores de entrada a la nueva precisión.

void pack32to16 (*int32_t* * vectorIn, int16_t *vectorOut, *uint32_t* longitud);

7) Realizar una función que reciba un vector de números signados de 32 bits y devuelva la posición del máximo del vector.

int32_t max (*int32_t* * vectorIn, *uint32_t* longitud);

8) Realizar una función que reciba un vector de muestras signadas de 32 bits y lo decime descartando una cada N muestras.

void downsampleM (*int32_t* * vectorIn, int32_t * vectorOut, *uint32_t* longitud, uint32_t N);

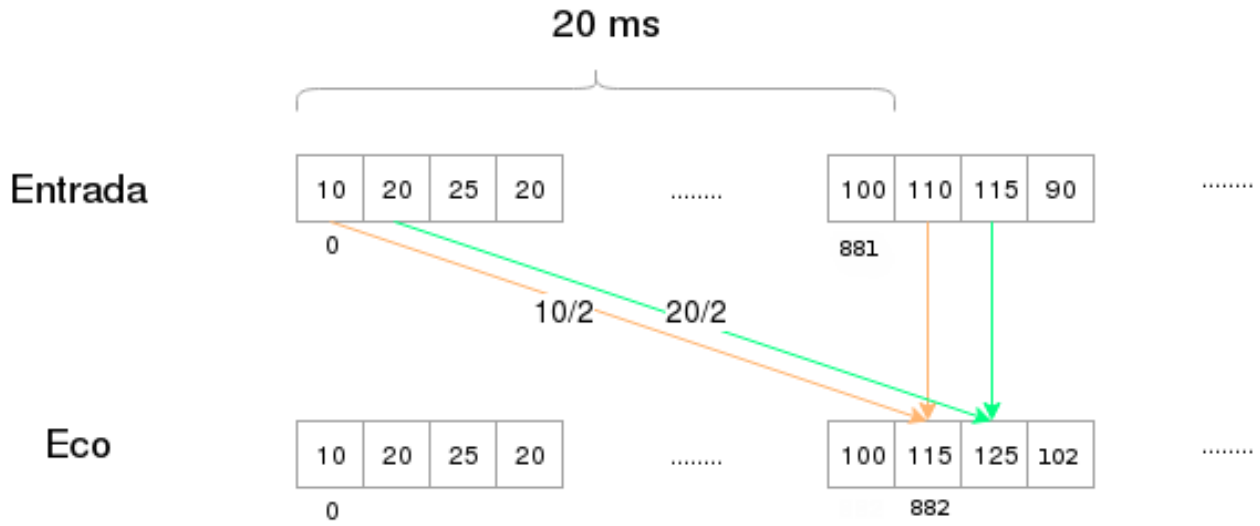
9) Realizar una función que reciba un vector de muestras no signadas de 16 bits e invierta su orden.

void invertir (uint16_t * vector, *uint32_t* longitud);

Autor: Ing. Facundo S. Larosa – Lic. Danilo Zecchin

Carrera de Especialización / Maestría de Sistemas Embebidos
Universidad de Buenos Aires
Arquitectura de Microprocesadores

10) Realizar una función que recibe un vector de 4096 valores de 16 bits (signados), que corresponden a muestras de audio tomadas a una tasa de muestreo de 44.100 muestras/s. La función debe introducir un “eco” de la mitad de la amplitud de la muestra original a los 20ms de comenzada la grabación.
Nota: El eco consiste en adicionar a la señal original, la propia señal original dividida por dos y atrasada en 20ms, como muestra la ilustración debajo. Por qué la cantidad de muestras copiadas iguales es de 882?
Variante: Una vez implementada la función, programar una nueva versión que utilice las instrucciones SIMD para optimizar la ejecución del algoritmo.



11) La correlación cruzada (del inglés *cross correlation*) es una operación usada para estudiar el grado de similitud de dos señales y su fase relativa, aún en presencia de ruido. La correlación de dos funciones da como resultado una nueva función llamada función de correlación. La fórmula de cálculo es:

$$corr[l] = \sum_{n=0}^{N} x[n]\,y[n-l]$$

Realizar una función que calcule la correlación entre dos vectores. Luego optimice utilizando instrucciones SIMD.

*void corr (int16_t *vectorX, int16_t * vector Y, int16_t vectorCorr, uint32_t longitud)*

Autor: Ing. Facundo S. Larosa – Lic. Danilo Zecchin