

Informe

Proyecto Reproductor de Voz por Comando

Alumno: Omar Cabrera

Índice:

1	Introducción.....	3
2	Descripción del Proyecto:.....	6
2.1	Sección <i>MSP-EXP430F5438</i> :.....	6
2.1.1	LCD:	6
2.1.2	Memoria Flash:.....	6
2.1.3	Botones:	10
2.1.4	PWM:.....	11
2.1.5	UART:.....	11
2.1.6	ADC:.....	11
2.1.7	DMA:.....	11
2.2	Programa del <i>MSP-EXP430F5438</i> :.....	11
2.2.1	Sección Inicio:.....	12
2.2.2	Sección Menú:	15
2.2.3	Sección Desarrollo aplicación seleccionada	15
2.3	Sección <i>PC</i> :	29
2.3.1	Programa Matlab:	29
2.3.2	Programa Visual Basic 2008:	29
3	Requerimientos:.....	35

1 Introducción.

En el proyecto a presentar se desarrolló un reproductor de voz. El mismo cumple con los requerimientos solicitados en la materia. El dispositivo consiste en la utilización del módulo *MSP-EXP430F5438 Experimenter Board* como hardware (Ver Figura 1), con el cual se podrán reproducir las pistas de audio a través de comandos ingresados por botones.

La configuración y lectura del contenido de este módulo (cargar bibliotecas, ver bibliotecas cargadas, ver memoria ocupada, etc.) se puede ejecutar mediante una aplicación de software utilizada en la computadora, que es muy amigable para el usuario. Esta aplicación está compuesta de dos programas. El primero, fue desarrollado en Matlab, su funcionalidad se basa en grabar las pistas de audio con la menor distorsión posible, eliminando todos los efectos de ruido de la grabación, ya sea: propio de la voz o del ambiente.

El segundo programa es una aplicación desarrollada en la plataforma de Visual Basic. Hace hincapié exclusivamente en: realizar la interface entre la PC y el kit de desarrollo. Esto implica lo siguiente: cargar librerías de las pistas de audio, elegir los comandos con las que se las reproducirá dentro del dispositivo, divisar la cantidad de memoria ocupada, lectura de la información ya cargada en el kit y borrado de la memoria del módulo.

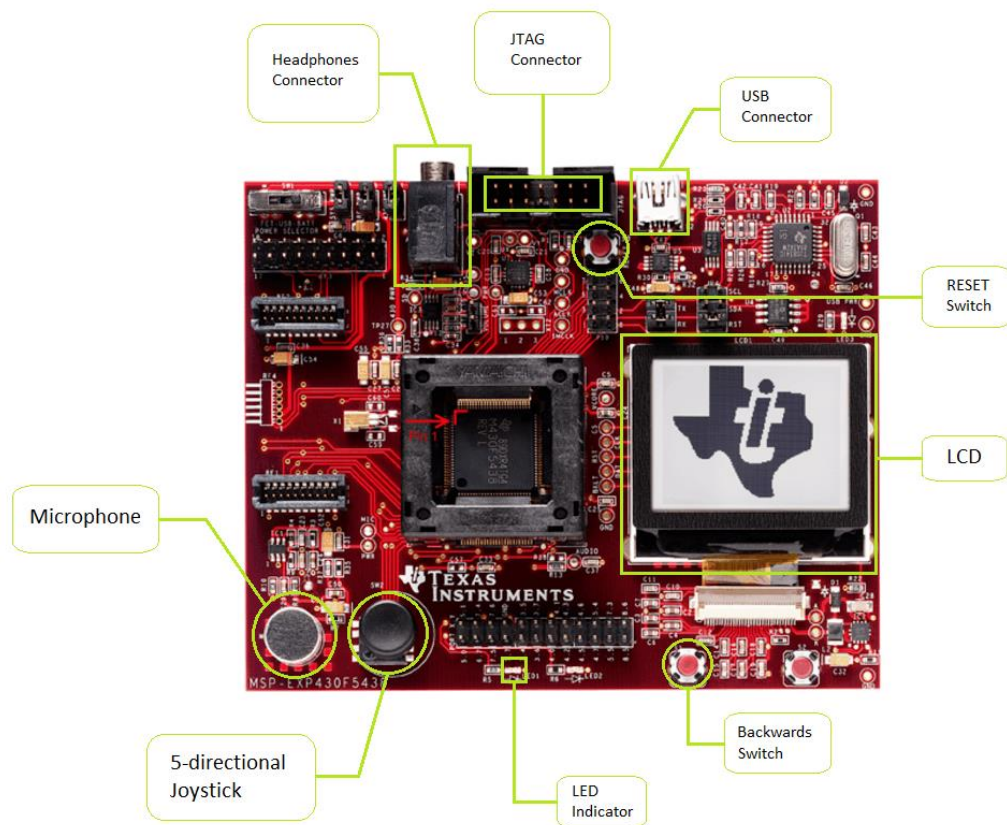


Figura 1: MSP-EXP430F5438 Experimenter Board.

En la siguiente figura (Figura 2) se encuentra un esquema explicativo que ayuda a la comprensión de la implementación del proyecto realizado.

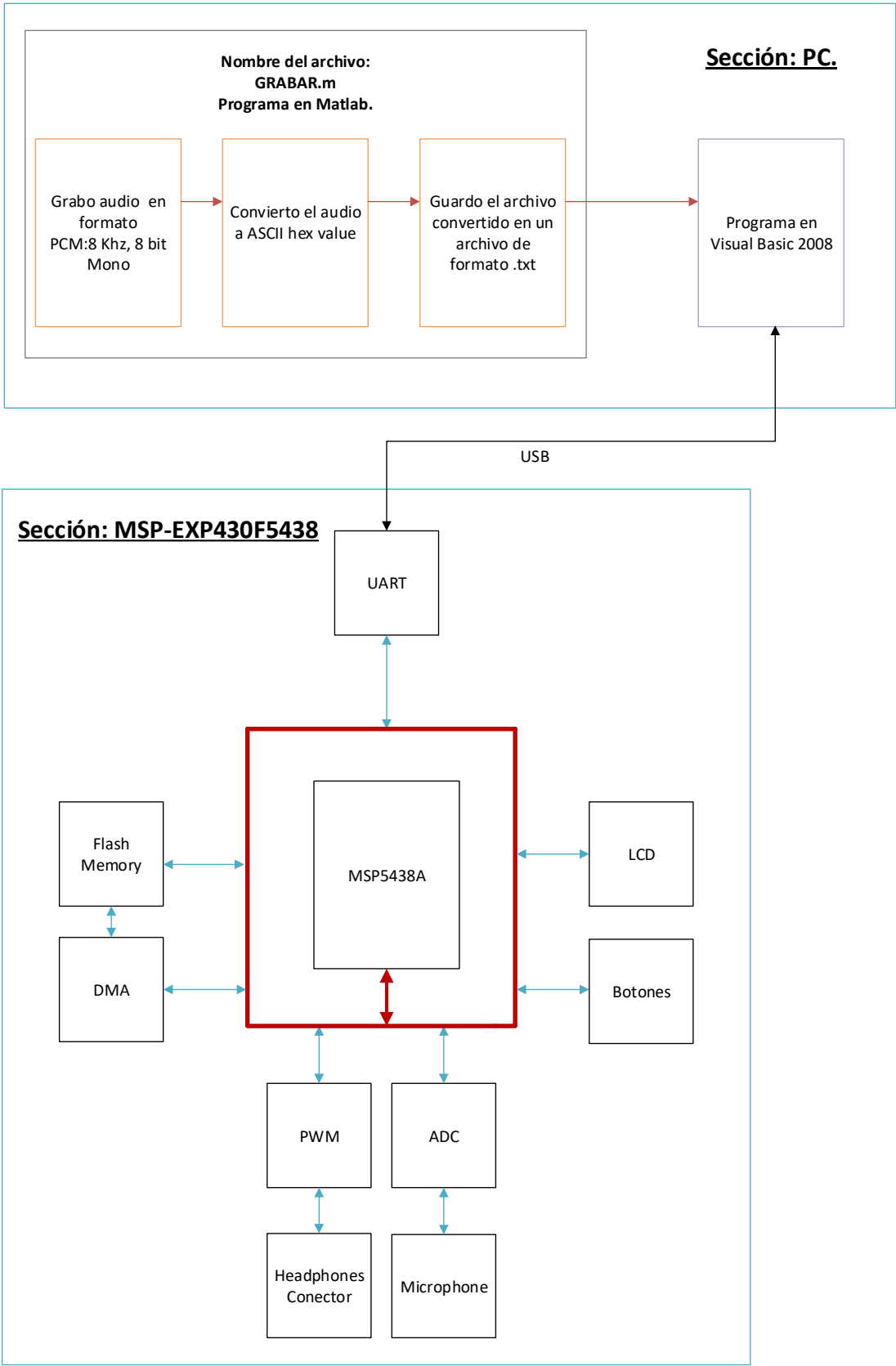


Figura 2: Descripción de la implementación del proyecto.

2 Descripción del Proyecto:

2.1 Sección *MSP-EXP430F5438*:

2.1.1 LCD:

El LCD (*HITACHI HD66753*) es utilizado como interface entre el usuario y el dispositivo. En éste el beneficiario puede ver el menú con las diferentes opciones a elegir.

Para la configuración del LCD se necesita: “*Hitachi HD66753 LCD user's guide*”, la cual, no se puede hallar. Por lo tanto, para la implementación de las librerías y funciones propias del LCD, se utilizaron los ejemplos que venían en la placa de desarrollo.

2.1.2 Memoria Flash:

La memoria flash puede ser direccionada de a byte, word o long-word y esta particionada en segmentos de 512 bytes. En ella se puede escribir de a un solo bit, byte o word y el borrado tiene un tamaño mínimo de un segmento.

Para realizar una escritura en memoria flash es necesario desbloquearla antes de guardar el dato. Una vez grabado el dato, se realiza nuevamente el bloqueo de la memoria.

Para ejecutar el desbloqueo y bloqueo de la memoria basta con modificar los registros FCTL3 y FCTL1 como lo indica la Figura 3 y Figura 4.

Caso Bloqueo:

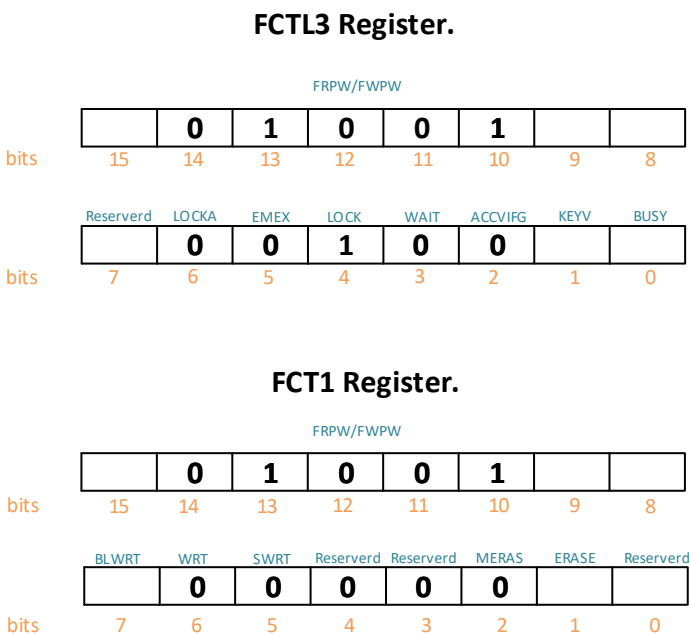
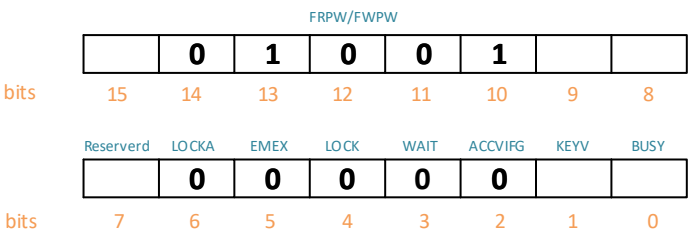


Figura 3: Disposición de registros, para bloqueo.

Desbloqueo:

FCTL3 Register



FCT1 Register

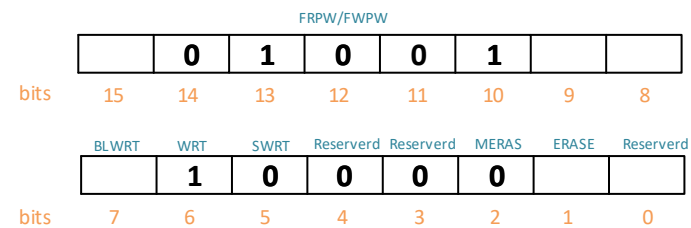


Figura 4: Disposición de registros, para desbloqueo.

Para realizar borrado de memoria se utiliza la función “FlashErase”, la cual se encuentra en el archivo *flashUtils.c*. Esta función pide como parámetro de entrada la dirección de inicio y fin del rango de memoria que se desea borrar. A continuación, se detallará el funcionamiento de la función:

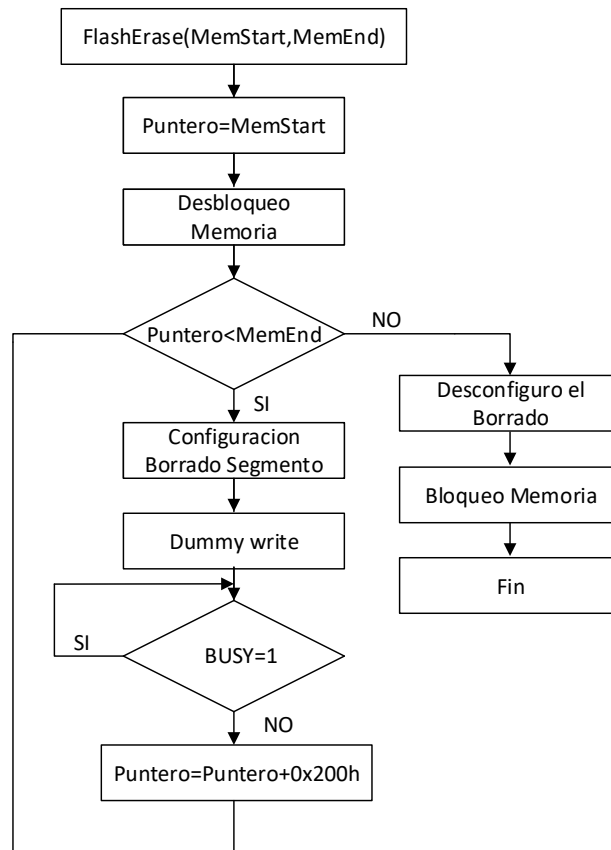


Figura 5: Diagrama de flujo de la función Flash Erase.

El mapa de la memoria flash tendrá la siguiente disposición:

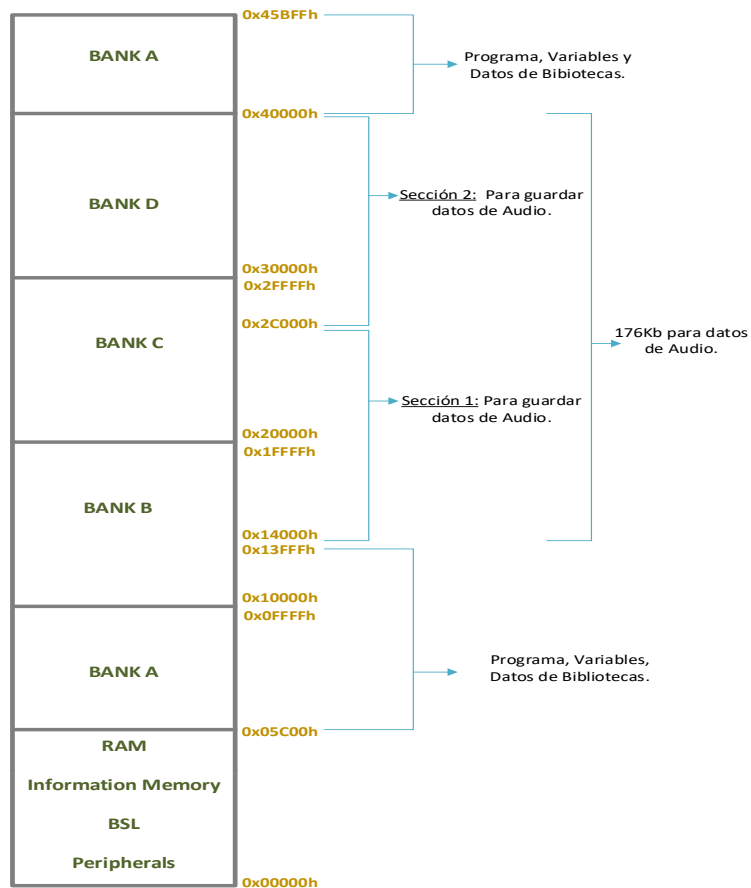


Figura 6: Mapeo de Memoria Flash.

Datos de Biblioteca:

Para poder guardar toda la información de las bibliotecas se creó una estructura llamada “Biblioteca”, la misma está compuesta por los siguientes elementos:

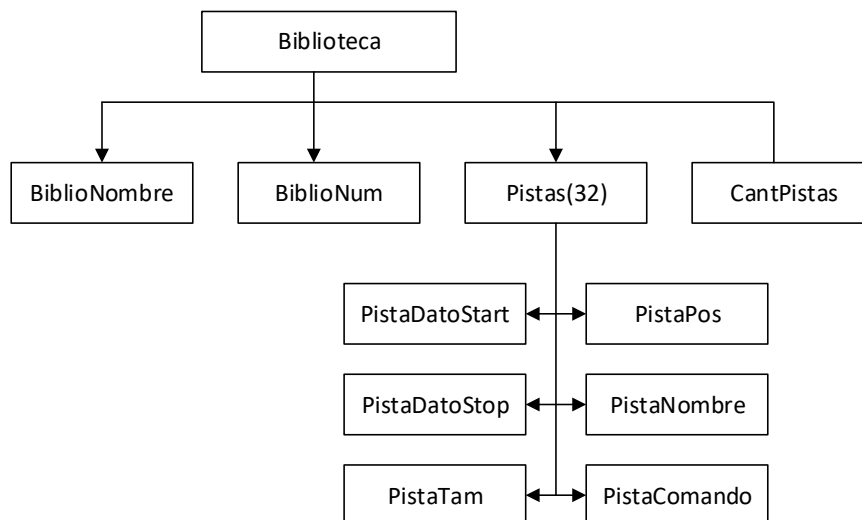


Figura 7: Estructura de la Biblioteca de las pistas.

BiblioNombre: campo que contiene el nombre de la biblioteca.

BiblioNum: campo que contiene el número de la biblioteca.

Pista[32]: vector que contiene las 32 posibles pistas dentro de una biblioteca.

PistaPos: posición que ocupa la pista dentro de una biblioteca.

PistaNombre: nombre de la pista.

PistComando: comando de la pista.

PistaTam: tamaño en bytes que ocupa la pista de audio en memoria.

PistaDatoStart: dirección de inicio de memoria flash donde esta almacenada al pista.

PistaDatoEnd: dirección de fin de memoria flash donde esta almacenada al pista.

CantPistas: cantidad de pistas que contiene la Biblioteca.

El requisito que se pide es administrar por lo menos 2 bibliotecas, por este motivo, se creó un vector de tamaño dos, del tipo “biblioteca”.

Criterios para guardar pistas en memoria:

Las pistas de ambas bibliotecas se guardan una a continuación de la otra, siempre y cuando las mismas entren en la memoria disponible desde la dirección 0x14000h hasta 0x40000. Siempre que se desee guardar una pista se debe corroborar que exista espacio en la memoria; se puede confirmar mediante la siguiente suposición: $PistaMemStart + PistaTam$ no ha de superar la dirección 0x40000. Así mismo, si esta pista es guardada, se le agrega la posición que ocupa dicha pista dentro de la biblioteca en un vector de ensuciado (BitModBiblioteca1 para la biblioteca 1 y BitModBiblioteca2 para la biblioteca 2). Este sistema de categorización hará más fácil la búsqueda de las mismas en un futuro.

Cuando se decide grabar una pista utilizando el micrófono del dispositivo, ésta se graba en el sector 2, previamente, eliminando cualquier pista que se encuentre en el sector.

2.1.3 Botones:

Se utiliza un botón tipo joystick de 5 movimientos para el ingreso de los comandos. Aquí, es necesario hacer la siguiente salvedad: se utilizan los 5 movimientos tanto para seleccionar las pistas de audio a reproducir, así como también, para seleccionar acciones relacionadas con la reproducción de las mismas; por ejemplo: parar, subir volumen, bajar volumen, etc. Además, se cuenta con un botón simple que se utiliza para volver al menú anterior.

La secuencia de posiciones será detallada más adelante en la sección: “Configuración del módulo”.

2.1.4 PWM:

Mediante el uso del Timer B del MSP430F5438A se generara una señal de PWM para reproducir las diferentes pistas almacenadas en la memoria flash. La configuración y el uso del mismo serán detallados en la sección de “Aplicaciones” dentro del programa del módulo.

2.1.5 UART:

La conexión entre la PC y el dispositivo se realizara atreves de la USCI (Universal Serial Communication Interface) en modo UART. La configuración y el protocolo usado serán detallados en la sección de “Aplicaciones” dentro del programa del módulo.

2.1.6 ADC:

Este se utiliza para convertir los valores analógicos obtenidos por el micrófono. Más específicamente, se hace uso del ADC12 del MSP430F5438A. La configuración y el uso del mismo serán detallados en la sección de “Aplicaciones” dentro del programa del módulo.

2.1.7 DMA:

Se encarga básicamente de mover datos de una determinada posición de memoria hacia otra. Este tiene esta forma de funcionamiento particular, ya que todos los periféricos están mapeados a una determinada dirección de memoria. Se usa principalmente, para llevar los datos obtenidos por el ADC12 a la sección 2 de memoria para datos de audio, indicado en la Figura 6.

2.2 Programa del *MSP-EXP430F5438*:

A continuación, se nombran los archivos que componen el proyecto de acuerdo a su sección y su correspondiente función.

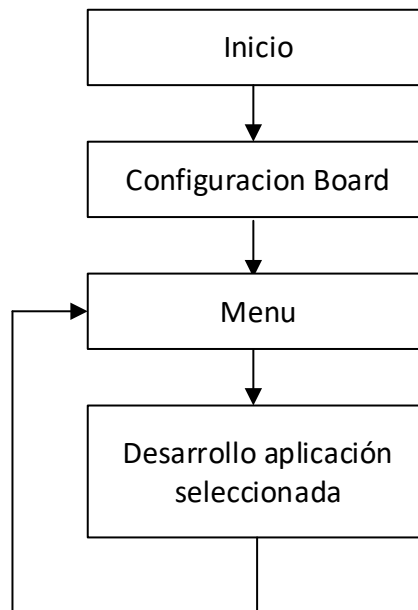


Figura 8: Diagrama de flujo del programa.

2.2.1 Sección Inicio:

Esta sección comienza en el archivo Main.c, en el cual: se deshabilita el watch dog, se habilitan las interrupciones globales y se invocan a las funciones boardconfig (Configuración del módulo) y menú_start(Inicio del menú).

Sección Configuración del Módulo:

Es desarrollada por la función boardconfig(), la cual, se encuentra en el archivo "BoardConfig.c" y tiene el siguiente funcionamiento:

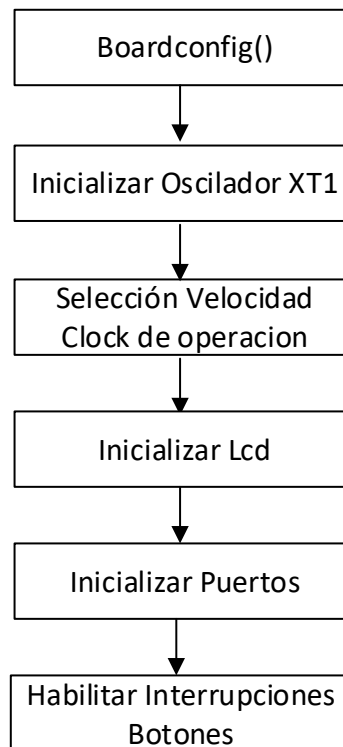


Figura 9: Diagrama de flujo de la configuración de la placa de desarrollo.

Inicializar oscilador XT1:

Se inicializa el oscilador XT1 utilizando la función `BoardStartXT1()`. En ella primeramente, se modifica el bit 7 y el bit 6 del registro `P7sel`, poniéndolos en 1. Así, sus pines correspondientes (`P7.0` y `P7.1`) los configuramos como periférico, es decir, como oscilador XT1.

Luego, se selecciona el capacitor a utilizar con un valor de 12 pF y se configura la “drive strength” necesaria para la frecuencias de operación entre 8 y 16 Mhz.

Selección velocidad de operación:

Se selecciona la frecuencia del clock de operación (MCLK) mediante la función `BoardSetSystemClock()`; la función pide como parámetro de entrada la velocidad deseada. Para lograr esto, primero se realiza la configuración del “Digital controlled oscillator” (DCO), consiste en modificar los bits: `DCORSEL`, `FLLD`, `FLLN`, `SELREF`, para obtener la frecuencia deseada ($f_{DCOCLKDIV}$).

$$f_{DCOCLKDIV} = (N + 1) \times (f_{FLLREFCLK} \div n)$$

Dónde:

$f_{FLLREFCLK}$: es la fuente utilizada para el FLL. Esta es la que viene por defecto, más precisamente, XT1 (32768Hz).

$(N + 1)$: esta constante de multiplicación es definida mediante los bits de `FLLN`.

n : determina la cantidad de veces que se divide la frecuencia de referencia, es fijado mediante los bits `FLLD`.

DCORSEL: selecciona el rango de frecuencia, para el rango de 8 a 16 Mhz.

DCORSEL = 0x04 (dato obtenido de las hojas de datos del microcontrolador MSP430F538A, sección 5.20 figura 5-10:Typical DCO frequency).

Para el caso de una frecuencia deseada de 8Mhz se utiliza: $(N+1) = 245$, $n=1$.

Obtenemos como resultado:

$$f_{DCOCLKDIV} = 245 \times 32768Hz = 8.02MHz.$$

Una vez configurado el DCO, se lo debe seleccionar como fuente del MCLK utilizando los bits SELM del registro UCSCTL4.

Inicializar LCD:

Se realiza mediante la función halLcdInit(), la cual configura la comunicación SPI2C con el módulo y el procedimiento de inicialización según Hitachi.

Luego se establece el "Backligh" y el "Constraste" utilizando las funciones halSetBackLight() y halSetConstrast(). Posteriormente se limpia la pantalla con la función halLcdClearScreen()

Inicialización de Puertos:

Se configuran los puertos de los botones (Arriba=P2.4, Abajo=P2.5, Derecha=P2.2, Izquierda=P2.1, Enter=P2.3, S1=P2.6, S2=P2.7) como entrada en modalidad pullup, para esto se utiliza la función ButtonsInit(). Esto se logra modificando los bits P2OUT,P2DIR,P2REN,P2SEL.

Primero se modifica P2SEL, con sus bits en 0 haciendo que se comporte como la función de pin y no la de periférico. Luego, se selecciona cada bit de: P2OUT,P2DIR,P2REN según la siguiente tabla:

PxDIR	PxREN	PxOUT	I/O Configuration
0	0	x	Input
0	1	0	Input with pulldown resistor
0	1	1	Input with pullup resistor
1	x	x	Output

Habilitación de Interrupciones:

Se habilitan las interrupciones de los botones y se limpian los flags de las mismas, mediante la función ButtonsInterruptEnable(), utilizando los bits: P2IES,P2IFG,P2IE.

2.2.2 Sección Menú:

Esta sección es la encargada de realizar la interface con el usuario a través del LCD. Mediante este menú se seleccionan las distintas aplicaciones. El funcionamiento del mismo es el siguiente:

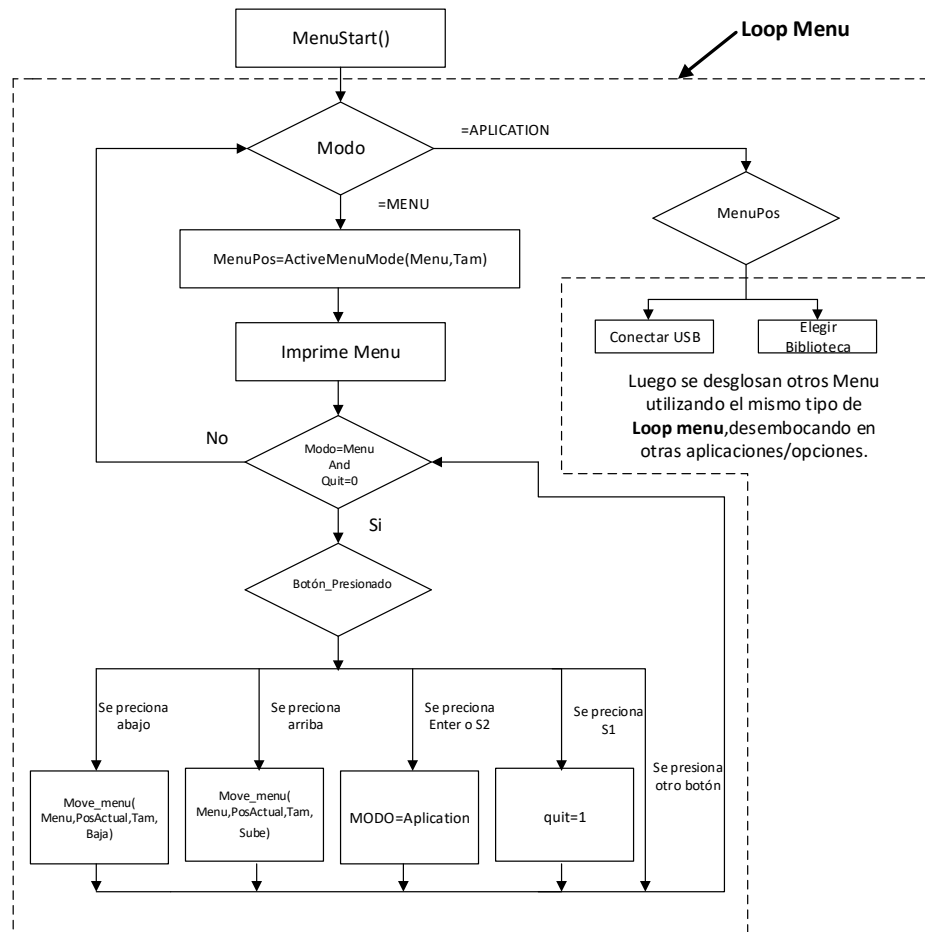


Figura 10: Diagrama de flujo del funcionamiento del programa menú.

2.2.3 Sección Desarrollo aplicación seleccionada

Dentro del menú las aplicaciones seleccionadas se pueden encontrar: “Conectar USB” o “Pistas y Comandos”. En el siguiente diagrama se muestra el flujo de programa:

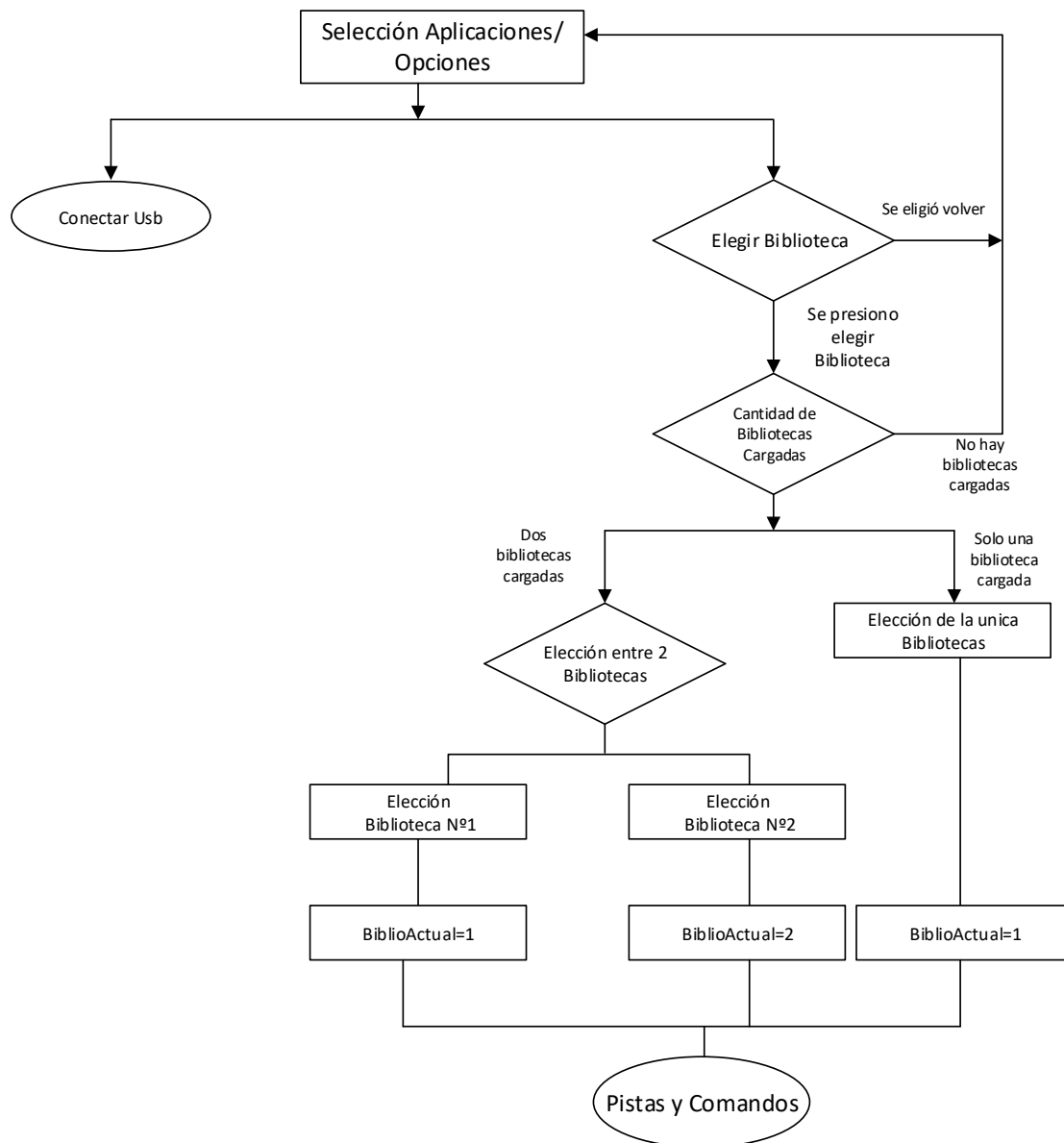


Figura 11: Diagrama de flujo de selección de aplicación.

Aplicación “Conectar USB”:

El objetivo de esta aplicación es: realizar la carga de bibliotecas, lectura de la información de mismas, borrado de memoria o lectura de memoria entre el dispositivo y la PC. Para ello, se deben realizar los siguientes pasos:

- Realizar la conexión física correspondiente.
- En el dispositivo se debe seleccionar la opción ‘Conectar USB’.

- Una vez finalizada la carga de bibliotecas desde la PC al dispositivo o la lectura de la información de las mismas se volverá al menú principal.

Descripción de la conexión UART:

La transmisión de datos bidireccional se realizara atreves de una conexión uart entre el dispositivo y la PC. Está se basa en el siguiente protocolo de comunicación:

- BaudRate:57600
- Data Bit: 8
- No Parity
- Un solo Bit de stop
- No HANDSHAKE
- No CheckSum(lo agregaría para mitigar posibles errores)
- No TimeOut(lo agregaría para mitigar posibles errores)

Comandos/Funciones	N° de comando
Cargar Nombre Biblioteca	'Z' 90
Cargar Numero Biblioteca	'L' 76
Cargar Numero Pista	'P' 80
Cargar Nombre Pista	'N' 78
Cargar Comando Pista	'I' 73
Cargar Tamaño Pista	'T' 84
Cargar Audio Pista	'M' 77
Fin de Carga	'Q' 81
Fin de Transmisión	'U' 85
Pc pregunta cantidad Memoria	'O' 79
Pc borra memoria dispositivo	'H' 72

Funcionamiento:

Las funciones necesarias para llevar a cabo la aplicación "Conectar USB" se encuentran en el archivo USB.c .

El siguiente diagrama de flujo explica cómo es la carga de Bibliotecas desde la PC al Dispositivo:

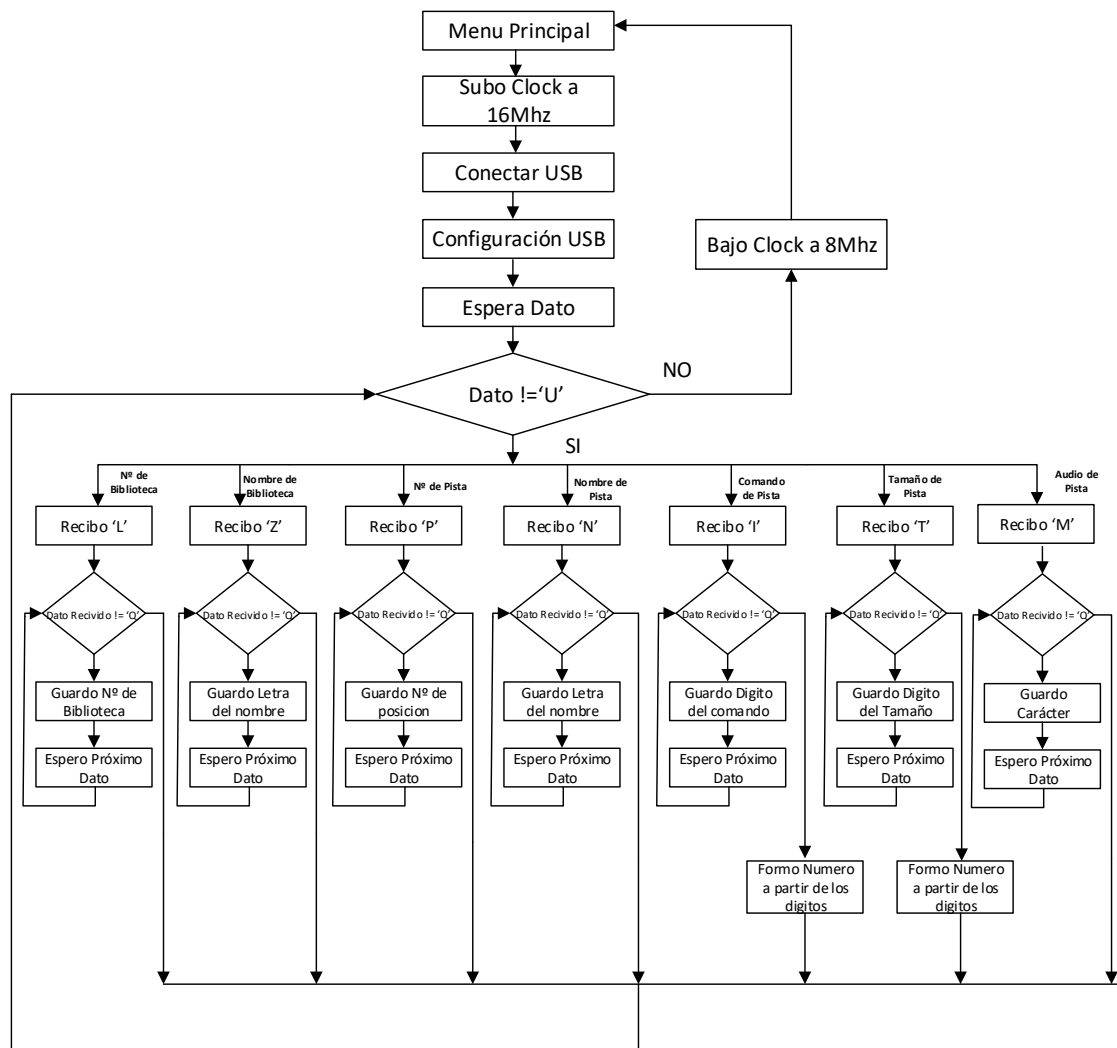


Figura 12: Diagrama de flujo Cargado de bibliotecas.

Configuración USB:

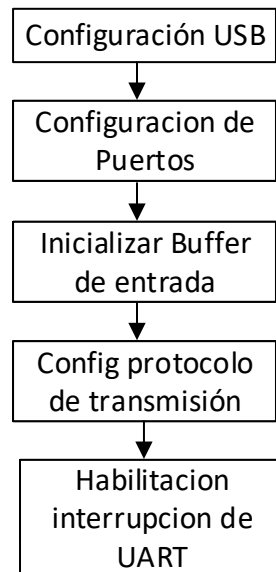


Figura 13: Diagrama de flujo, configuración USB.

La configuración USB es realizada por la función InitUsb(), en ella se configura los puertos de la siguiente manera:

Configuración de puertos:

Primero se configuran los puertos P5.7 y P5.6 como periférico, es decir como transmisor y

receptor UART. Esto se logra modificando en "1" sus correspondientes bits de P5Sel (Bit 7 y Bit 6). Luego se selecciona a P5.7 (Tx) como puerto de salida y P5.6 (Rx) como entrada

Inicialización del Buffer de recepción:

Además se crea un vector llamado UsbReceiveBuffer de tipo char con un tamaño de 255 posiciones para guardar los datos recibidos durante la transmisión. Este vector se comporta como un buffer circular, luego de grabar en la posición 254 la siguiente posición donde guardar será la 0. Este vector será inicializado con un cero en todas sus posiciones.

Configuración del protocolo de transmisión:

Para lograr las características de transmisión definidas se deberán modificar los registros UCA1CTL0, UCA1CTL1, UAC1BR0, UAC1BR1.:

Modificando los siguientes bit de UCA1CTL0 s :

UCPEN=0b: Transmisión sin paridad.

UCMSB=0b: LBS se transmite primero.

UCSPB=0b: un bit de parada.

UCMODE=00b: Modo UART.

UCSYNC=0: Transmisión asincrónica.

Los registros UAC1BR1, UAC1BR0 en conjunto forman un número (UAC1BR1 es la parte alta y UAC1BR0 la parte baja del número) que representa la relación entre la frecuencia de operación y la baudrate requerida. En este caso la frecuencia de operación es de 16Mhz y el baudrate deseado es de 57000, entonces:

$$Relacion = \frac{16 \times 10^6}{57000} = 277$$

Por lo tanto UAC1BR1=1 y UAC1BR0=21 dando un total de: 256+21=277.

Por ultimo modificando el registro UCA1CTL1=10b se selecciona el SMCLK como "clock source" del USCI(universal serial communication interface).

Habilitación Interrupción UART:


Para habilitar las interrupciones por recepción de datos basta con modificar el bit UCR1IE=1 del registro UCA1IE


Aplicación "Comandos y Pistas":

En esta aplicación se está a la espera de comandos ingresados a través del "5-directional joystick". Para salir de la espera y volver al menú anterior sólo basta con presionar el "backwards switch".

La secuencia de botones presionados son guardados en una FIFO llamada "FifoBotones" hasta que se presione la tecla ENTER (esta es la posición central del 5-directional joystick). Una vez presionada, la FIFO se vacía en un vector llamado "comandos"; los valores guardados en este vector son decodificados utilizando un sistema en base 5, en donde:

ENTER = 0

 **= 1**

 **= 2**

 **= 3**

 **= 4**

Los comandos que se pueden formar son los siguientes:

$Play = ENTER \Rightarrow 5^0 \times 0 = 0$

$Stop = \uparrow + ENTER \Rightarrow 5^0 \times 1 + 5^1 \times 0 = 1$

$Repetir = \uparrow + \leftarrow + ENTER \Rightarrow 5^0 \times 1 + 5^1 \times 3 + 5^2 \times 0 = 16$

$SubirVol = \uparrow + \uparrow + ENTER \Rightarrow 5^0 \times 1 + 5^1 \times 1 + 5^2 \times 0 = 6$

$BajarVol = \uparrow + \downarrow + ENTER \Rightarrow 5^0 \times 1 + 5^1 \times 2 + 5^2 \times 0 = 11$

$Borrar = \uparrow + \downarrow + \uparrow + ENTER \Rightarrow 5^0 \times 1 + 5^1 \times 2 + 5^2 \times 1 + 5^3 \times 0 = 36$

$Grabar = \uparrow + \leftarrow + \rightarrow + ENTER \Rightarrow 5^0 \times 1 + 5^1 \times 3 + 5^2 \times 4 + 5^3 \times 0 = 116$

$CambiarComand = \uparrow + \leftarrow + \uparrow + ENTER \Rightarrow 5^0 \times 1 + 5^1 \times 3 + 5^2 \times 1 + 5^3 \times 0 = 41$

$Cambiar Biblioteca = \uparrow + \uparrow + \downarrow + ENTER \Rightarrow 5^0 \times 1 + 5^1 \times 1 + 5^2 \times 2 + 5^3 \times 0 = 56$

Si el comando ingresado es alguno de estos se guardara en la FIFO “FifoAcciones”, en cambio si es diferente, se trata de una pista y se guarda en la FIFO “FifoPistas”

El funcionamiento del programa desarrollado dentro del kit es el siguiente:

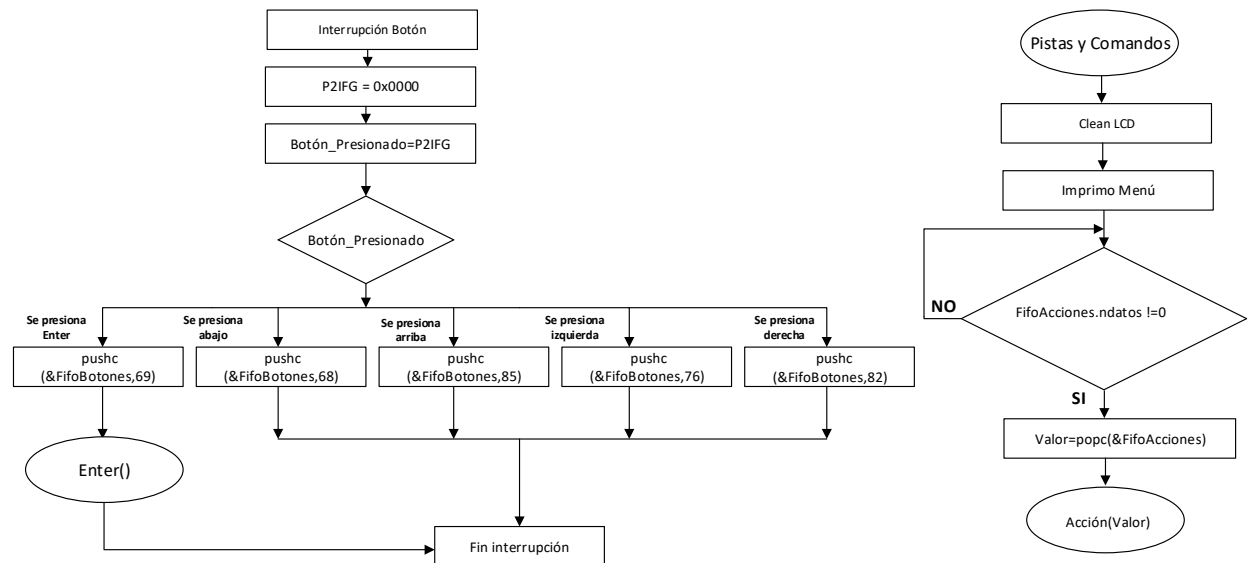


Figura 13.1: Diagrama de flujo, Comandos y pistas.

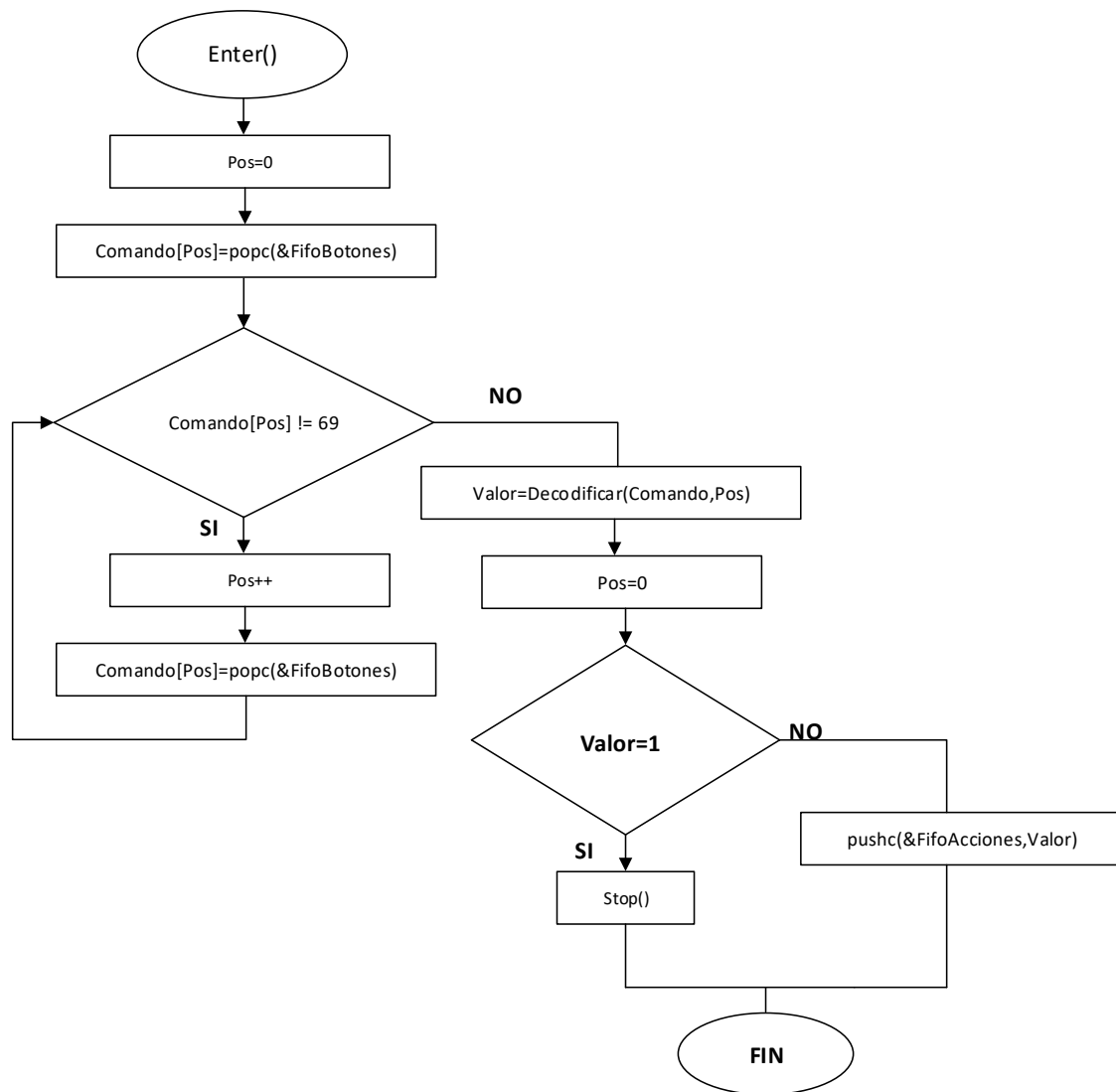


Figura 13.2: Diagrama de flujo, Comandos y pistas.

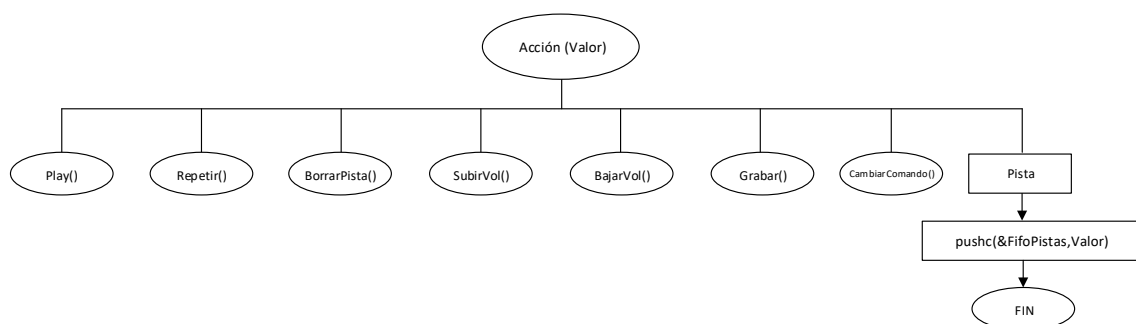


Figura 13.3: Diagrama de flujo, Comandos y pistas.

Comando "Play":

Este comando requiere que previamente se seleccione una pista a reproducir. Primero se busca si la pista seleccionada existe en memoria dentro de su correspondiente biblioteca. Para ello se ve si la pista está en el vector de ensuciado (mencionado en la sección de Memoria Flash). Si la búsqueda resulta fallida se imprime "Pista inexistente" en el LCD, en cambio si la búsqueda resulta exitosa se pasará a la reproducción de la pista de la siguiente manera:

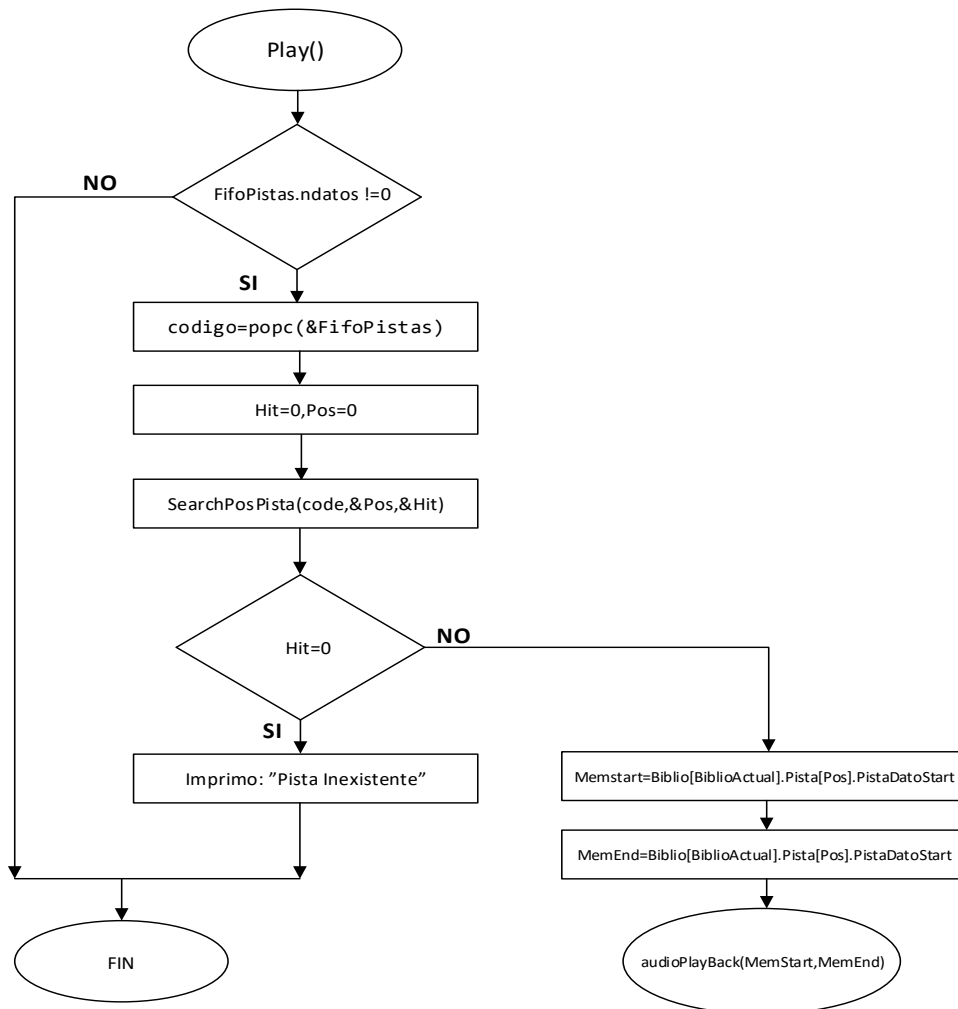


Figura 14.1: Diagrama de flujo, comando Play.

La función SearchPosPista se encarga de buscar si existe la pista ingresada en la biblioteca previamente seleccionada en el menú "elegir biblioteca". El resultado de la búsqueda es exitoso si la pista ingresada se encuentra dentro del vector de ensuciado, obteniendo como dato la posición de dicha pista (Pos) en la biblioteca. Con la posición de la pista se puede saber dónde están guardados los datos de audio para la reproducción de la misma de la siguiente manera:

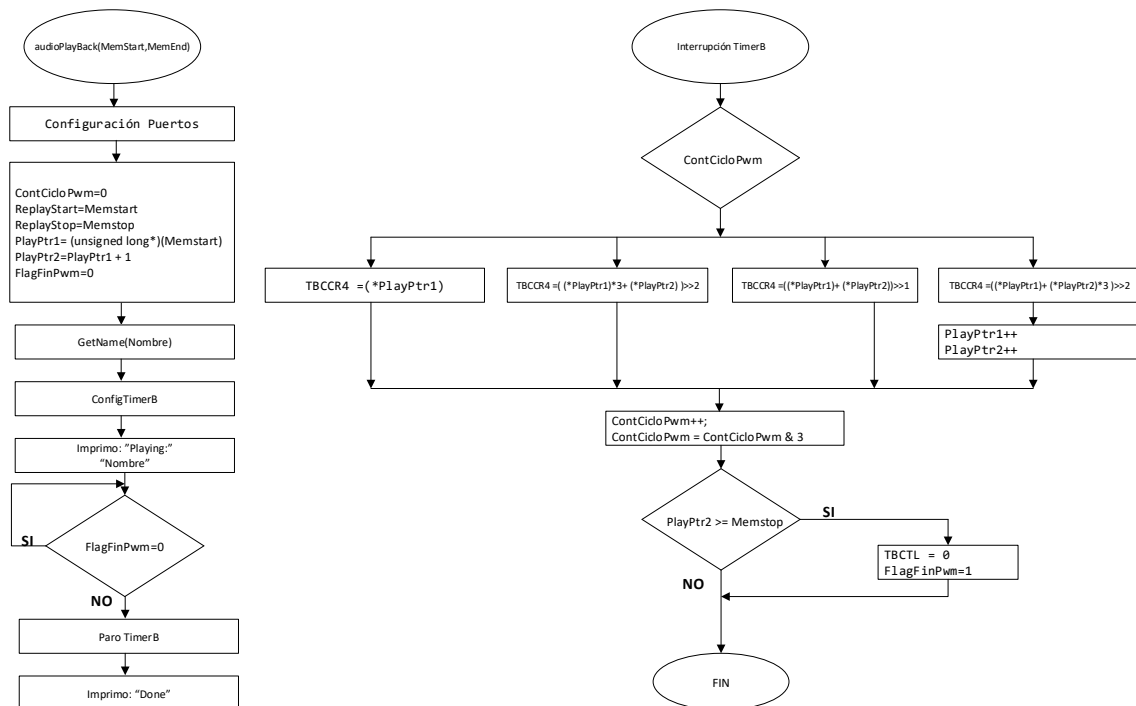


Figura 14.2: Diagrama de flujo, comando Play.

Se comienza configurando el puerto P4.4 como periférico, por este saldrá la señal de PWM que reproducirá el audio. Luego el puerto P6.6 como salida para alimentar el amplificador de audio y por último se enciende el led indicador.

Luego se inicializan las variables que tienen las siguientes funciones:

- ContCicloPwm: contador de ciclos de PWM.
- ReplayStart y ReplayEnd: se guarda la última dirección de comienzo y final para luego utilizarlas en la función Replay.
- PlayPtr1 y PlayPtr2: punteros a la dirección de memoria en donde se encuentra el dato de audio.
- FlagFinPwm: indica cuando se termina de reproducir toda la pista.

A la hora de reproducir la una pista es un requerimiento mostrar el nombre de la misma, por ello se utiliza la función Getname(nombre) para obtener el nombre de la pista para su posterior muestra en el lcd.

Para reproducir las pistas de audio se utilizara una señal de PWM de 32Khz. Esta será generada utilizando el timer B que es configurado de la siguiente manera:

- Se selecciona SMCLK(8Mhz) como clock del timer B utilizando los bits TBSSEL=10b del registro TBOCTL.
- Se habilita la interrupciones del mismo utilizando el bit TBIE=1 del registro TBOCTL.

- Se configura la OUTPUT UNIT, que se encarga de generar señales de salida en modo Reset/Seg. Este modo funciona de la siguiente forma:

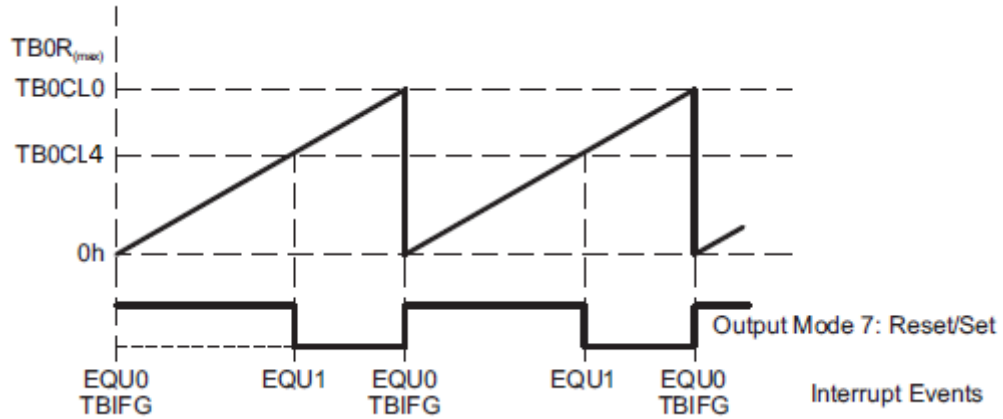


Figura 15: Timer B modo Reset/Set.

El pin P4.4 se mantiene en “1” mientras se cuenta desde 0h hasta TB0CL4, luego cambia a cero mientras su contador se va incrementando de TB0CL4 hasta TB0CL0. Al llegar hasta TB0CL0 se genera la interrupción de timer B en la cual se realiza el procedimiento mostrado en la Figura 14.2.

Para generar un PWM de 32Khz con una frecuencia de clock de 8Mhz el valor de TB0CL0 debe ser iguala a:

$$TB0CL0 = \frac{Frec_{clock}}{Frec_{pwm}} = \frac{8\text{ Mhz}}{32\text{ Khz}} = 250$$

Además como las pistas de audios están muestreadas a una tasa de 8 Ksps a cada muestra de audio le corresponden 4 ciclos de PWM, por lo que se usó un promediado móvil como se muestra en la siguiente figura:

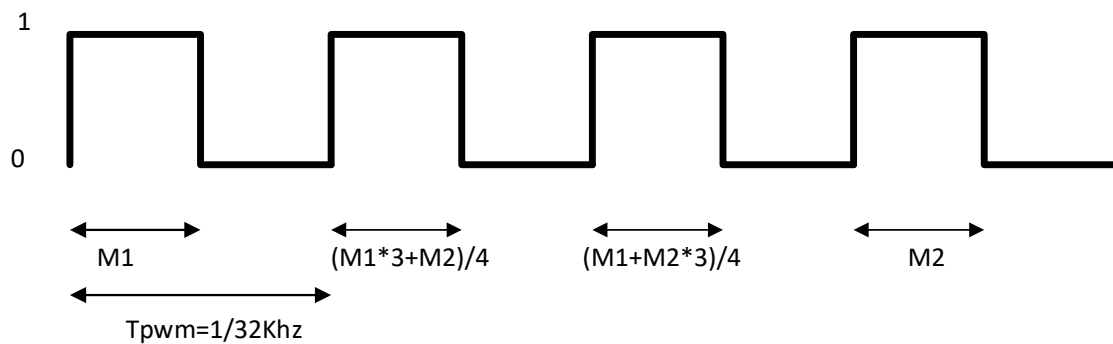


Figura 15: Promediado Móvil.

En donde M1 representa una muestra cualquiera y M2 la siguiente.

Comando “Stop”:

Este comando detiene la reproducción de la pista en curso, apagando el timer B y seteando el FlagEndPwm=1. Para apagar el timer B solo se necesita modificar el bit MC=0 del registro TB0CTL.

Comando “BorrarPista”:

El funcionamiento de este comando es el siguiente:

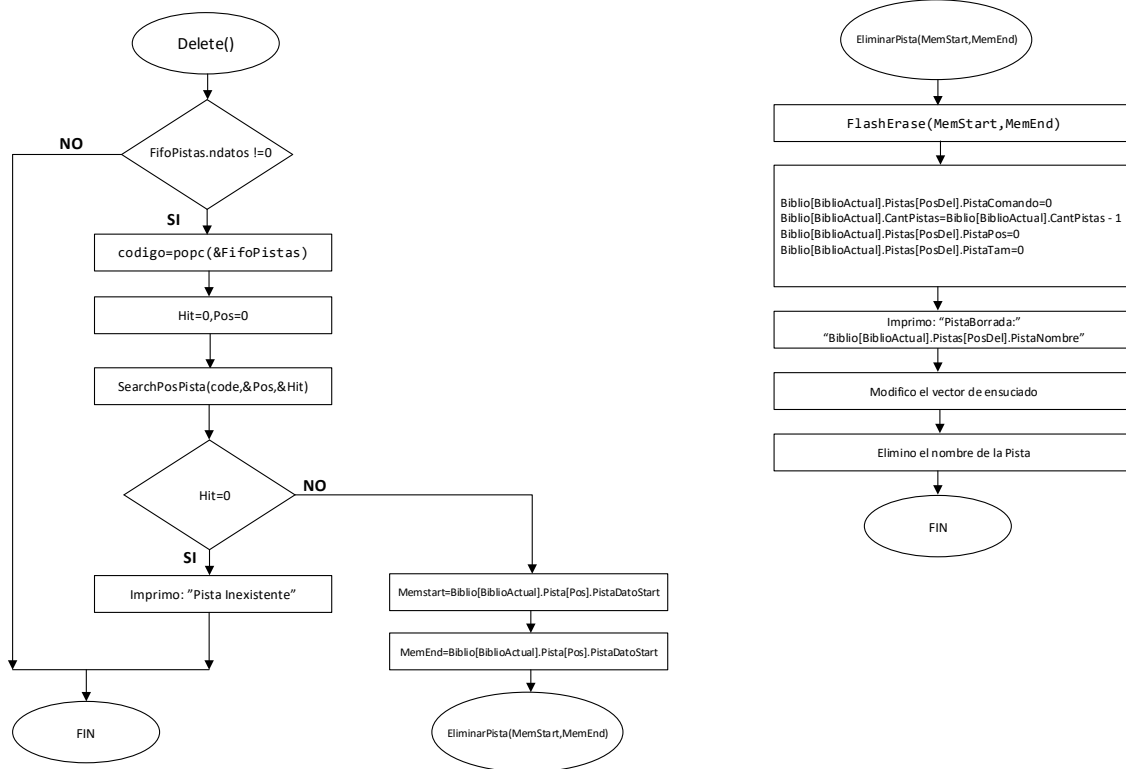


Figura 15: Diagrama de flujo, comando BorrarPista.

Comando “Repetir”:

Se utiliza las variables ReplayStart y ReplayEnd como dato de entrada para la función audioPlayBack.

Comando “CambiarBiblioteca”:

_Si se está en la biblioteca 1 se pasa a la 2 y viceversa

Comando “Cambiar Comando”:

Este comando requiere que previamente se ingresen dos pistas, primero la pista a cambiar y luego el nuevo comando de la misma. El funcionamiento del programa es el siguiente:

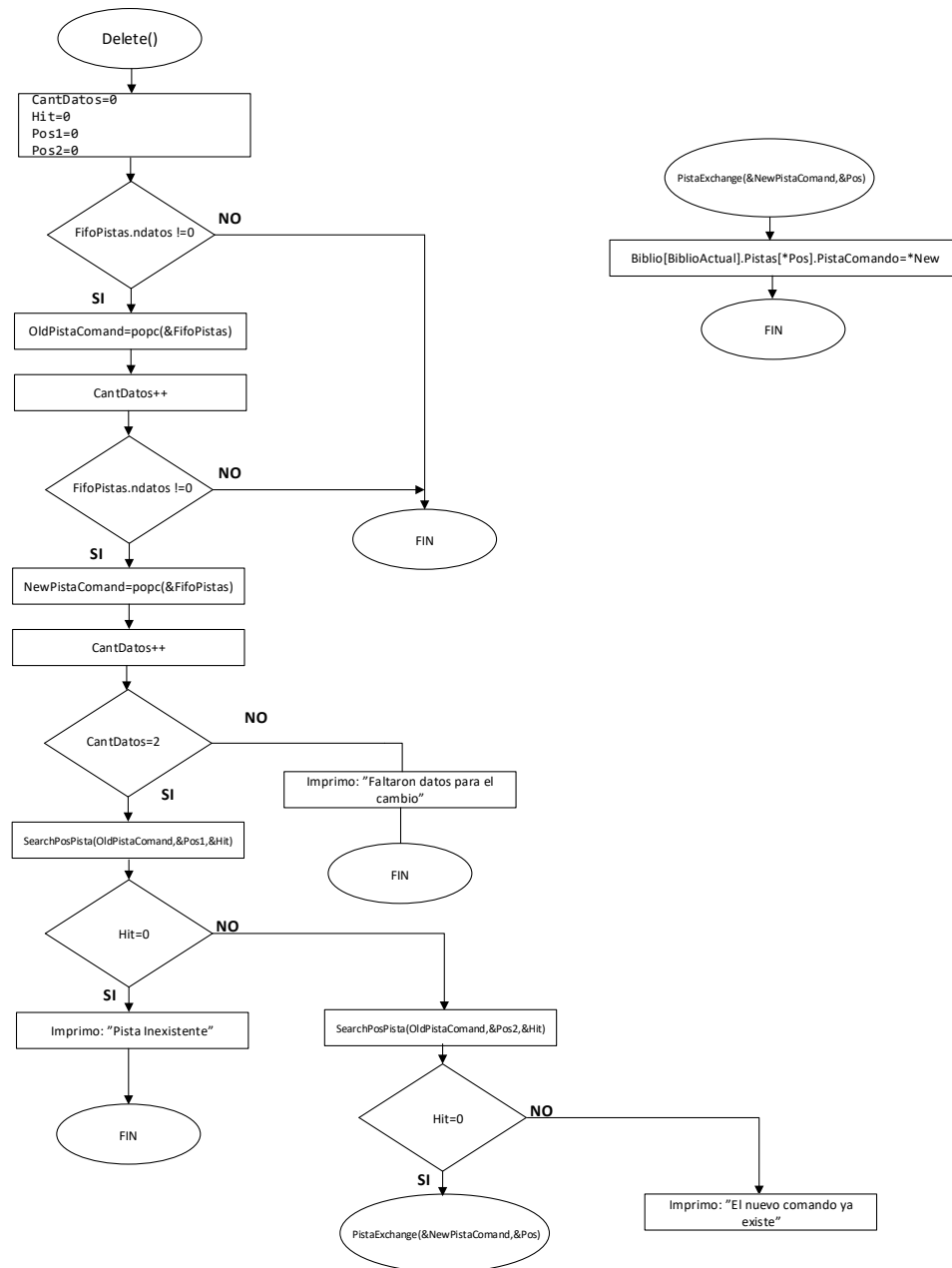


Figura 15: Diagrama de flujo, Cambiar comando.

Comando “Grabar”:

Antes de comenzar con el grabado, se borra cualquier pista que se encuentre en la sección 2 de datos de audios.

Este comando consiste en utilizar el ADC12 para captar el audio que ingresa por el micrófono durante 10 segundos. Se utilizó una resolución de 8 bits y una tasa de muestreo de 8Ksps, para tener coherencia con lo realizado hasta el momento (8 Kbyte 1 segundo de audio).

El tiempo entre muestras viene dado por: $t_{sts} = t_{sample} + t_{convert}$ (despreciando t_{sync}) como se puede apreciar en la siguiente figura:

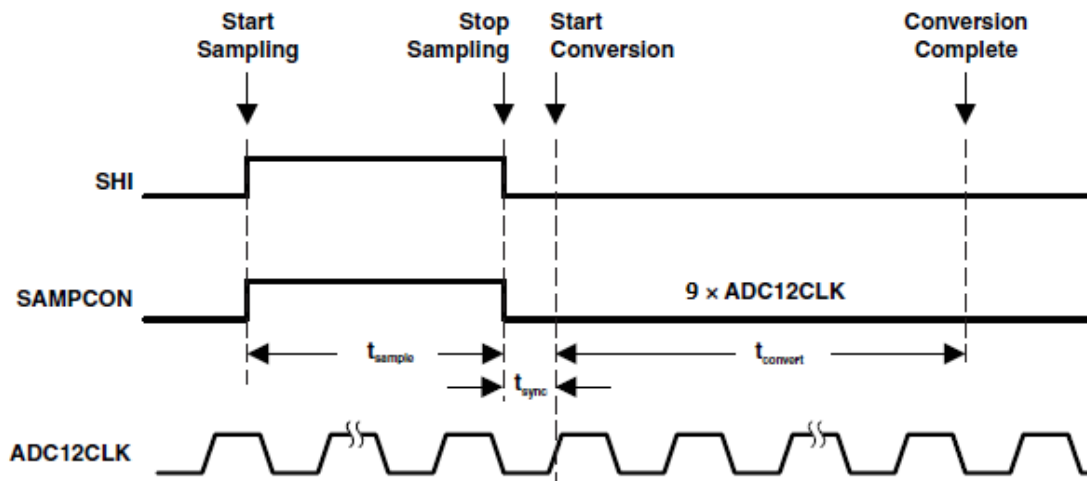


Figura 16: Tiempo de muestreo ADC12.

El ADC12CLK utiliza el clock SMCLK como fuente, teniendo un valor de 8Mhz. Para el ADC12 con una resolución de 8 bit el $t_{convert}$ es de 9 ciclos.

El tiempo de duración de muestreo es fijado utilizando el timer b en modo Reset/set (como en el caso del PWM). En este modo el t_{sample} es equivalente a 991 ciclos de clock que sumados a los 9 ciclos del $t_{convert}$ da un total de 1000 ciclos. Dando un total de 8000 muestras por segundo.

Los valores obtenidos por el ADC12 son llevados a la sección 2 de datos de audio por medio del DMA hasta que la sección 2 este llena. Los parámetros a configurar del DMA son los siguientes:

- Dirección de memoria a donde ir a buscar el dato (Source Address), esta se configura en el registro DMA0SA. Para este caso es la dirección de memoria del ADC12(0x720)
- Dirección de memoria a donde depositar el dato (Destination Address), esta se configura en el registro DMA0DA.
- Tamaño del dato de "Source Address", en este caso un byte. Se configura modificando el bit DMASRCBYTE=1 del registro DMA0CTL.
- Tamaño del dato de "Destination Address", en este caso un byte. Se configura modificando el bit DMADSTBYTE=1 del registro DMA0CTL.
- La dirección de destino se incrementa después de cada transferencia. Se configura modificando el bit DMADSTINCR=1 del registro DMA0CTL.
- Se fija la cantidad de transferencias de un byte que se van a realizar. Se configura modificando el registro DMA0SZ.

- Se habilita el DMA con el bit DMAEN=1 del registro DMA0CTL.
- Por último se habilitan la interrupción del mismo con el bit DMAIE=1 del registro DMA0CTL.

A través del LCD se indicara cuando se está borrando la sección dos, cuando se está grabando y cuando se termina la misma.

La pista grabada de esta manera ocupara la posición 31 de la biblioteca que se esté usando en el momento. Además tendrá un comando de pista reservado= ↑ + ↑ + ↑ +ENTER.

2.3 Sección PC:

2.3.1 Programa Matlab:

Mediante este programa se graban las pistas de audio a cargar en el dispositivo. Estas son grabadas a una tasa de 8 Kbps, en 8 bit mono, y luego se guarda el audio en un archivo txt.

Me faltaría realizar una aplicación en GUI, para que Matlab no será un requisito.

2.3.2 Programa Visual Basic 2008:

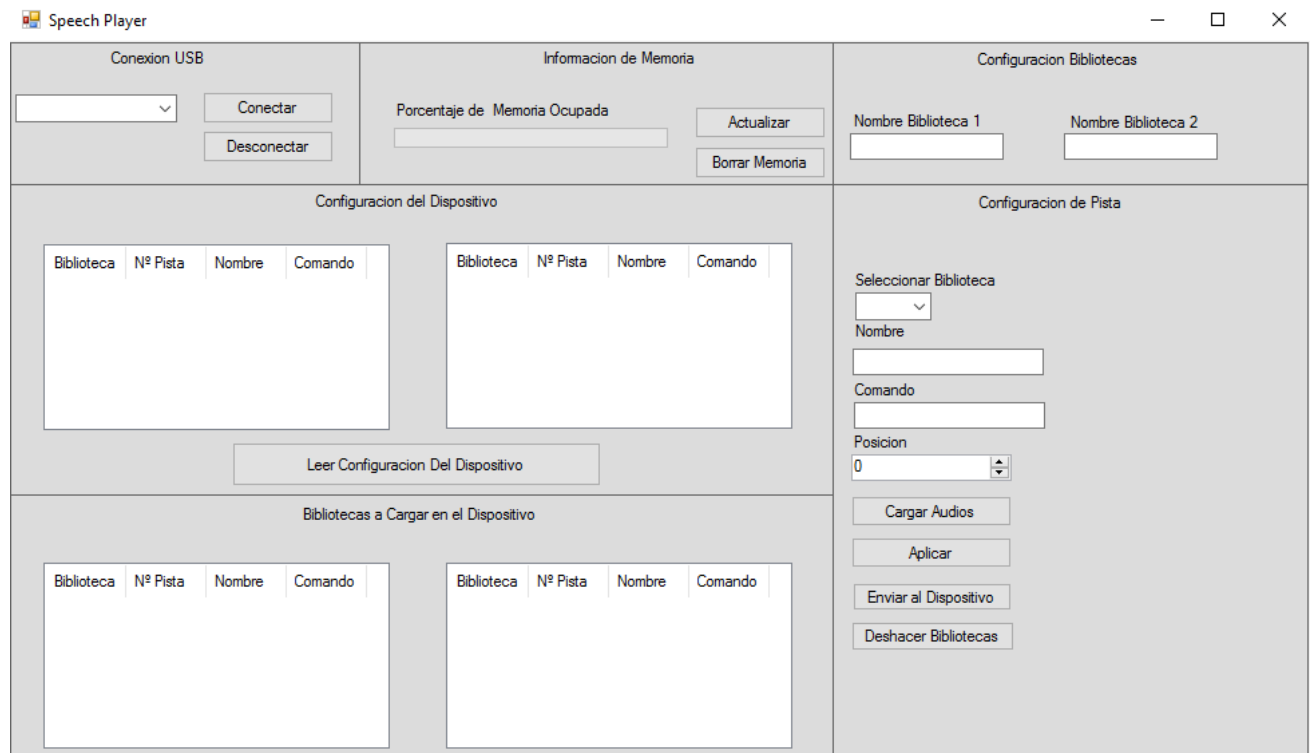


Figura 17: Software PC.

Dentro del software PC se pueden distinguir varias secciones: conexión USB, información de memoria, configuración bibliotecas y configuración del dispositivo.

Conexión USB:

Primero se selecciona el puerto COM, en donde está conectado el dispositivo, a través de un ComboBox. Luego se oprime el Botón Conectar con el cual se realiza la conexión usb con las siguientes características:

- Puerto COM: El seleccionado por el ComboBox
- BaudRate:57600
- Data Bit: 8
- No Parity
- Un solo Bit de stop
- No HANDSHAKE
- No CheckSum(lo agregaría para mitigar posibles errores)
- No TimeOut(lo agregaría para mitigar posibles errores)

Si la conexión resulta exitosa se imprimiera un cartel “Conexión exitosa” caso contrario se imprimirá un cartel “El puerto x no existe”.

Una vez establecida la conexión, las acciones a poder realizar por el Sw son las siguientes:

- Carga Bibliotecas al dispositivo
- Lectura de Bibliotecas en el dispositivo
- Lectura de Memoria ocupada
- Borrado de Memoria

Carga de Bibliotecas al Dispositivo:

Primero se debe cargar la información de las Bibliotecas en la sección “configuración Bibliotecas”, en la cual se carga los nombres de las bibliotecas. Estos datos son adquiridos atreves de dos TextBox y luego guardados en sus correspondientes Bibliotecas al oprimir el botón Aplicar.

Las Bibliotecas son unas estructuras compuestas de la siguiente manera:

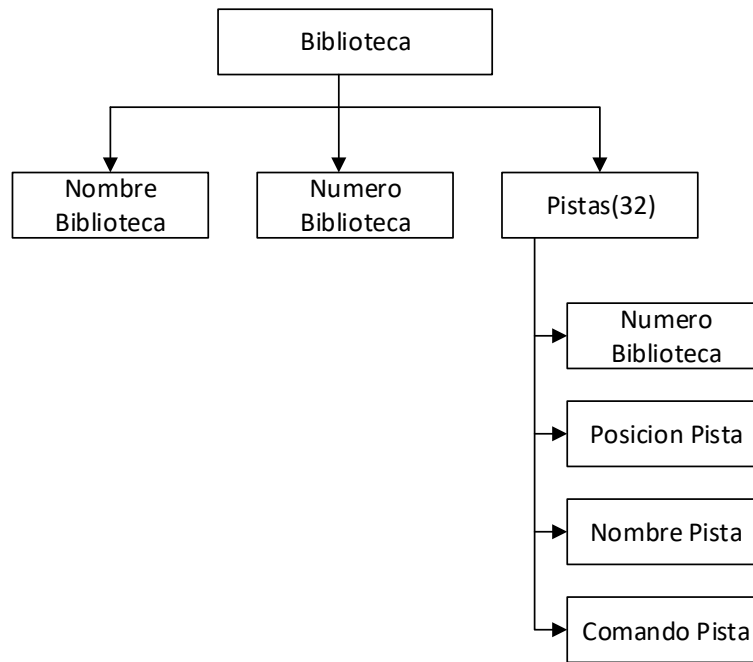


Figura 18: Estructura Biblioteca en Software PC.

Por lo tanto cada nombre será guardado en el campo “Nombre Biblioteca” de cada estructura.

De la misma manera se deberán rellenar los datos en el sector de Configuración de Pista: Número de Biblioteca, Nombre, Comando, Posición. Luego se debe cargar el directorio en donde se encuentra el archivo de texto de dicha pista utilizando el botón “Cargar Audio”.

Cada vez que se quiera cargar un pista se deberán completar todos los campos de la sección configurar pista y luego presionar el botón aplicar. Una vez presionado el botón se guardaran todos los datos en los campos correspondientes de las bibliotecas y se mostraran dichos datos en la sección “Bibliotecas a cargar en dispositivo”.

Para finalizar el cargado de Bibliotecas al dispositivo, una vez que el dispositivo se encuentre dentro de la aplicación “Conectar USB” y las bibliotecas/pistas se encuentren cargadas, se oprimirá el botón Enviar al dispositivo.

La secuencia de transmisión de datos desde la PC hacia el dispositivo es la siguiente:

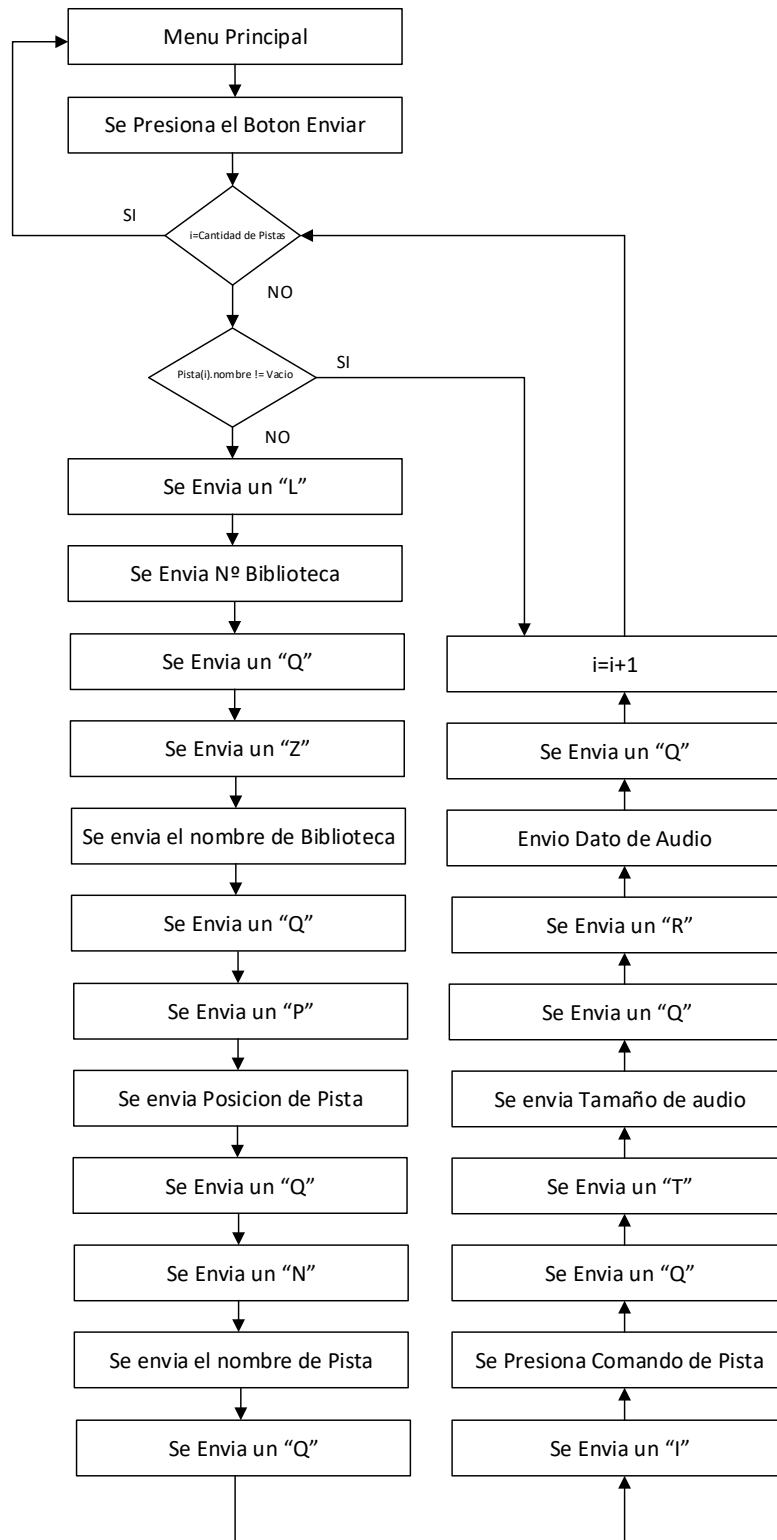


Figura 19: Carga de bibliotecas desde PC a dispositivo.

El llenado de campos de bibliotecas y pistas tienen las siguientes restricciones:

- No se aceptan letras Mayúsculas, para que estas no interfieran con el protocolo.

- Los comandos de las pistas no deben coincidir con los comandos de acciones (Reproducir, grabar, etc.).
- No repetir comandos de Pistas.
- Los audios deben tener una duración menor a 10 segundos.

Si se comente algún error durante la carga de datos antes de enviar se puede presionar el botón “Deshacer Bibliotecas” para reiniciar toda la carga de datos.

Lectura de Bibliotecas cargadas en el Dispositivo:

Para poder leer las bibliotecas cargadas es necesario que el dispositivo se encuentre en la aplicación “Conectar USB” y que se haya realizado la conexión USB. Cumplidos estos dos requerimientos solo bastaría con oprimir el botón “Leer Configuración del Dispositivo”.

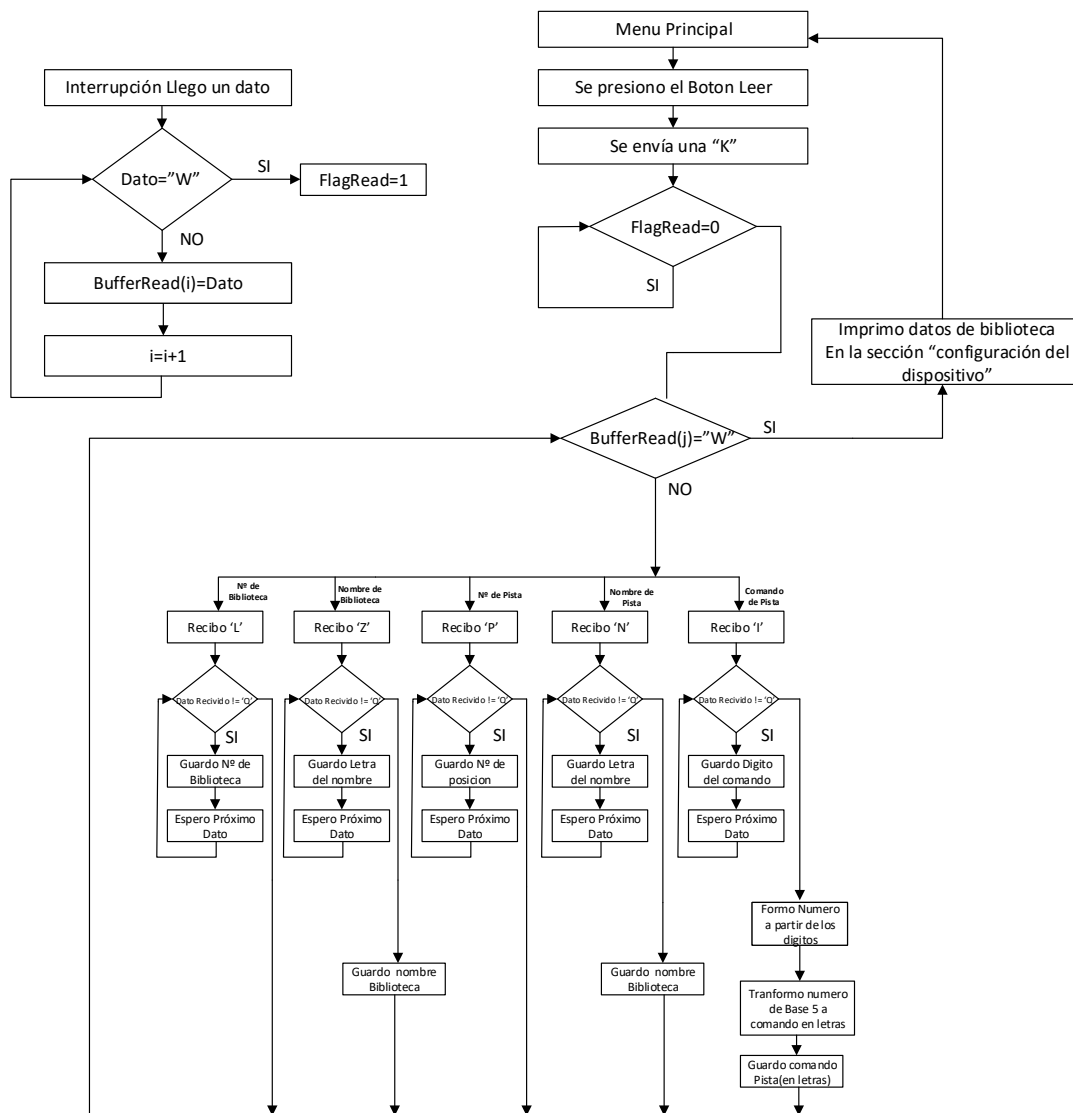


Figura 20: Lectura de bibliotecas del dispositivo.

Lectura de Memoria ocupada:

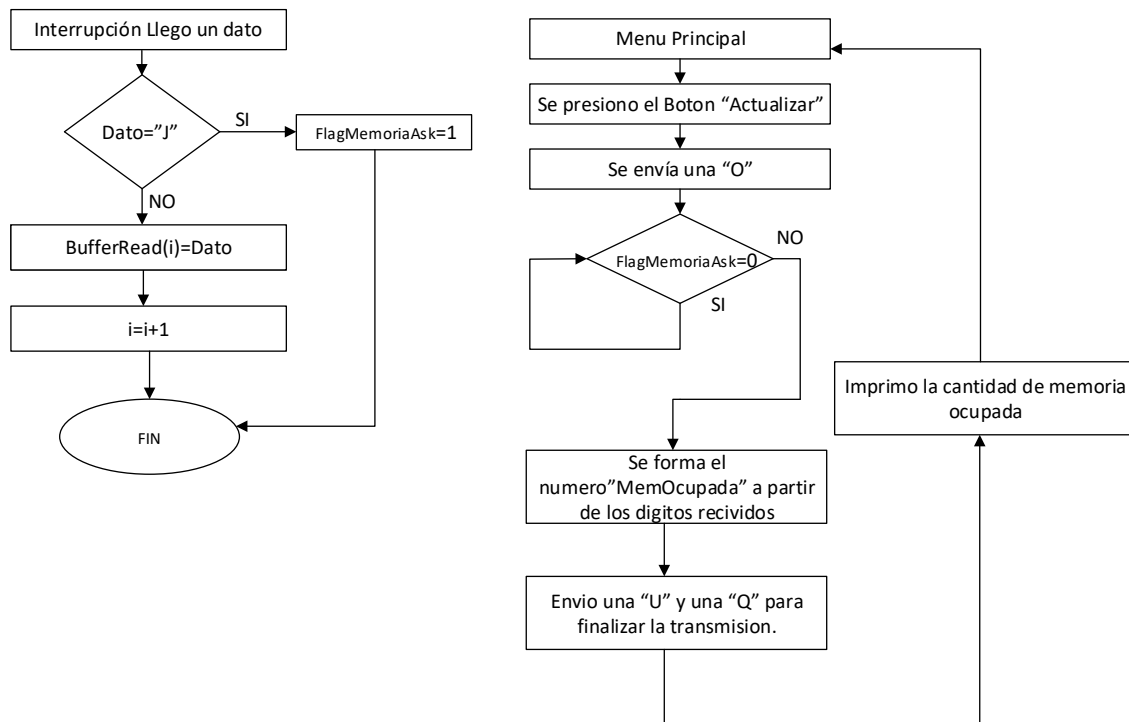


Figura 21: Lectura de memoria ocupada en el dispositivo.

Borrado de Memoria:

Al oprimir el botón "Borrar Memoria" se envía una "H" para dar inicio al borrado de memoria en el dispositivo. Luego una "U" y una "Q" para finalizar la transición.

Para borrar la memoria en el dispositivo solamente se utiliza 2 veces la FlashErase, una para borrar la información de las bibliotecas y otra para borrar los audios.

3 Requerimientos:

Req.	Descripción	Obs.
1.0	Generales	
1.1	El dispositivo debe ser capaz de reproducir pistas de audio almacenadas en el mismo.	Cumple
1.2	El dispositivo debe brindar una interfaz mecánica para el ingreso de comandos.	Cumple
2.0	Plataforma	
2.1	El diseño debe implementarse sobre un único dispositivo de procesamiento (microcontrolador, microprocesador, DSP o FPGA), no pudiéndose distribuir el procesamiento en múltiples dispositivos.	Cumple
2.2	La alimentación de la plataforma debe ser por baterías.	Cumple
3.0	Interfaces	
3.1	Brindar una interfaz para la conexión de un altavoz estándar. A través de dicha interfaz se reproducirán las pistas de audio almacenadas.	Cumple
3.2	Brindar una interfaz para la conexión de un micrófono estándar. A través de dicha interfaz se podrán grabar pistas de audio para su posterior reproducción.	Cumple
3.3	El dispositivo debe brindar una interfaz tipo joystick para el ingreso de comandos de operación y configuración del dispositivo.	Cumple
3.4	La interfaz tipo joystick debe poseer como mínimo los botones "Arriba" (U), "Abajo" (D), "Derecha" (R), "Izquierda" (L) y "Enter" (E).	Cumple
3.5	El dispositivo debe brindar un botón de reset para el usuario.	Cumple
3.6	El dispositivo debe brindar una interfaz para que el usuario pueda identificar la pista de audio seleccionada para alguna operación. La interfaz no debe ser audible.	Cumple
3.7	El dispositivo debe brindar una interfaz para que el usuario pueda identificar que una pista se encuentra en reproducción. La interfaz no debe ser audible.	Cumple
3.8	El dispositivo debe brindar una interfaz para la conexión a una PC, preferentemente en formato USB.	Cumple
4.0	Funcionamiento	
4.1	El dispositivo debe tener la capacidad de almacenar y reproducir pistas de audio de hasta 10 segundos de duración.	Cumple

4.2	Las pistas de audio deben al menos tener un ancho de banda (audio) como mínimo de 3 kHz.	Cumple. La voz fue digitalizada a 8 Kbps.

4.3	El dispositivo debe ser capaz de administrar colecciones de pistas de audio como bibliotecas seleccionables.	Cumple
4.4	Debe administrar al menos dos bibliotecas de 64 pistas máximo por cada una.	Utilice dos bibliotecas de 32 pistas cada una no estipula un mínimo el requerimiento
4.5	Cada pista de audio debe tener una posición estática dentro de la biblioteca.	Cumple
4.6	Cada pista de audio debe tener un código secuencial de hasta 4 valores a ser ingresado por la interfaz de comandos. Dicho código debe poder ser modificado por el usuario pero no puede estar repetido.	Cumple
4.7	Cada pista de audio debe tener propiedades asociadas. Como mínimo: Título, Posición.	Cumple
4.8	El dispositivo debe aceptar los siguientes comandos: <ul style="list-style-type: none"> - Reproducir una pista. Por ejemplo: U+D+D+R + E. - Repetir la última pista reproducida. - Frenar una pista. Por ejemplo: U+D+D+R + E (...) + E. - Grabar una pista. Por ejemplo: U+D+D+R + E(3 seg) + [Audio] + E. - Borrar una pista. - Cambiar el código de una pista. - Cambiar una biblioteca. - Subir/Bajar Volumen del audio. 	Me faltaría: Subir/Bajar volumen. Intenté hacer el mismo modificando el duty cycle de la señal pero no funciona.
4.9	El usuario debe ser capaz de encolar reproducciones mientras una pista se encuentra reproduciéndose. Al menos se espera poder encolar 3 pistas.	Cumple, además las pistas se pueden encolar todas al principio, siendo más eficaz cuando las pistas tienen poca duración de tiempo.
4.10	Las configuraciones del dispositivo, bibliotecas y pistas de audio deben realizarse a través de la PC usando el SW asociado.	Cumple
4.11	En el caso de proveer una pantalla gráfica o alfa-numérica, mostrar el nombre de la pista a reproducir o en reproducción.	Cumple

5.0 SW (PC)		
5.1	Proveer de un SW que corra en una PC que sea capaz de comunicarse con el dispositivo.	Cumple, tiene ciertos bugs dependiendo de cómo el usuario ingrese la información.
5.2	El SW de PC debe permitir la escritura, recuperación y borrado de las pistas almacenadas.	Cumple
5.3	El SW de PC debe permitir al usuario cambiar todos los parámetros de configuración del dispositivo, de sus bibliotecas y pistas.	Cumple
5.4	El SW de PC debe permitir al usuario saber qué configuración tiene cargada el dispositivo, sus bibliotecas y pistas.	Lee todo menos el audio de cada pista. Si esto es necesario puedo agregarlo
5.5	El SW de PC debe permitir al usuario saber qué porcentaje de memoria interna del dispositivo se encuentra ocupada.	Cumple