

Controlador USB Dongle TDA

1.-NÚMERO:	2.-FECHA:
1	10/08/2017

3. DIRECCIÓN	4. INVESTIGADOR RESPONSABLE:
Electrónica de Comunicaciones	Edgar Gómez

5. LUGAR.

CENDIT. Complejo Tecnológico Simón Rodríguez, Base Aérea Generalísimo Francisco de Miranda, La Carlota, Caracas.

6. Proyecto

Desarrollo de un controlador USB para la comunicación entre software decodificador de paquetes TS y Dispositivo USB receptor de TDA bajo el sistema operativo GNU/Linux.

ÍNDICE

ÍNDICE.....	2
ÍNDICE DE FIGURAS.....	3
ÍNDICE DE TABLAS.....	6
METODOLOGÍA.....	6
6.1 Utilidades de Software.....	7
6.1.1 Esquema de Software.....	8
7.1 La API de Samsung.....	10
7.1.1 Funciones del demodulador SemcoTC90527.....	10
7.1.2 Funciones del sintonizador SemcoSTV4100.....	17
7.1.3 Funciones definidas por el desarrollador.....	19
7.1.4 Procesos de inicialización y sintonización.....	19
8.1 GNU/Linux y el Dongel USB de TDA.....	25
9.1 La API Libusb.....	29
9.1.1 Transferencias asincrónicas.....	42
9.1.2 Rutinas USB.....	44
9.1.2.1 Rutinas de Conexión en Caliente.....	44
9.1.2.1 Rutinas de Transferencias de Control.....	48
9.1.2.1 Rutinas de Transferencias de Interrupción.....	54
10.1 La API Libvlc.....	60
10.1.1 Rutinas Multimedia.....	67
11.1 Interfaz Gráfica.....	73
11.1.1 Empaquetamiento gráfico.....	73
11.1.1 Funciones de Enlace entorno Gráfico.....	79
12.1 Instalador del Programa.....	84
12.1.1 Paquetes y software requerido.....	86
12.1.2 Proceso de compilación.....	87
12.1.3 Reglas Udev para el Dongle USB.....	88
12.1.4 Acceso directo.....	89
12.1.5 Script de instalación del programa.....	90
12.1.6 Script de desinstalación del programa.....	92
13.1 Documentación.....	92
14.1 Ejecución del instalador del programa.....	99
14.2 Montaje físico de los equipos.....	106
14.3 Pruebas de la aplicación desarrollada.....	111
14.4 Ejecución del script de desinstalación del programa.....	117
CONCLUSIONES Y RECOMENDACIONES.....	119
REFERENCIAS BIBLIOGRÁFICAS.....	122
ANEXOS.....	125

ÍNDICE DE FIGURAS

Figura 6.1.1: Pirámide de Dependencia.....	7
Figura 7.1.1: Procedimiento de inicialización del demodulador y sintonizador..	18
Figura 7.1.2: Procedimiento de sintonización de una frecuencia.....	18
Figura 8.1.1: Salida de la ejecución del comando lsusb.....	23
Figura 8.1.2: Salida de la ejecución del comando lusb-v.....	27
Figura 11.1. 1: Distribución de la rejilla, coordenadas de distribución pares de puntos (x, y).....	77
Figura 11.1.2: Ventana del Reproductor.....	77
Figura 11.1.3: Botones de Control Multimedia.....	78
Figura 11.1.4: Botones Indicadores.....	78
Figura 11.1.5: Botones de Programación.....	79
Figura 11.1.6: Ventana de Desplazamiento.....	80
Figura 11.1.7: Interfaz gráfica de la aplicación.....	84
Figura 11.1.8: Widget del botón reproducir y pausar, estado reproducir.....	86
Figura 11.1.9: Widget del botón reproducir y pausar, estado pausar.....	86
Figura 11.2.1: Widget del botón detener.....	86
Figura 11.2.2: Widget del botón ajuste de volumen.....	87
Figura 11.2.3: Widget del botón pantalla completa.....	87
Figura 11.2.4: Widget estado conexión USB, desconectado.....	87
Figura 11.2.5: Widget intensidad de la señal, nivel de intensidad 0.....	88
Figura 11.2.6: Widget intensidad de la señal, nivel de intensidad 1.....	88
Figura 11.2.7: Widget intensidad de la señal, nivel de intensidad 2.....	88
Figura 11.2.8: Widget intensidad de la señal, nivel de intensidad 3.....	88
Figura 11.2.9: Widget intensidad de la señal, nivel de intensidad 4.....	88
Figura 11.2.10: Widget estado conexión USB, conectado.....	88
Figura 11.2.11: Widget intensidad de la señal, nivel de intensidad 5.....	88
Figura 11.2.12: Widget intensidad de la señal, nivel de intensidad 6.....	88
Figura 11.2.13: Widget intensidad de la señal, nivel de intensidad 7.....	88
Figura 11.2.14: Widget intensidad de la señal, nivel de intensidad 8.....	88
Figura 11.2.15: Widget intensidad de la señal, nivel de intensidad 9.....	88
Figura 11.2.16: Widget intensidad de la señal, nivel de intensidad 10.....	89
Figura 12.1.1: Estructura de archivos del instalador.....	92
Figura 13.1.1: Documentación Doxygen.....	107
Figura 13.1.2: Documentación Doxygen, enlaces pseudocódigos.....	108
Figura 13.1.3: Documentación Doxygen, ejemplo pseudocódigo.....	109

Figura 13.1.4: Documentación Doxygen, ejemplo documentación de variables.	109
Figura 13.1.5: Documentación Doxygen, ejemplo documentación de funciones.	110
Figura 13.1.6: Manual de instalación muestra número uno.....	111
Figura 13.1.7: Manual de instalación muestra número dos.....	111
Figura 14.1.1: Ejecución del comando de instalación del programa.....	112
Figura 14.1.2: Importar repositorios de software para VLC en el proceso de instalación.....	113
Figura 14.1.3: Instalación de librerías y software necesario en el proceso de instalación.....	114
Figura 14.1.4: Verificar huella del repositorio importado en el proceso de instalación.....	115
Figura 14.1.5: Creación de directorios, compilación y reglas Udev.....	116
Figura 14.1.6: Estructura de archivos de reglas udev.....	116
Figura 14.1.7: Estructura de archivos del directorio de instalación.....	117
Figura 14.1.8: Lanzador de la aplicación.....	118
Figura 14.1.9: Ejecución de la aplicación.....	118
Figura 14.2.1: Diagrama de conexiones montaje físico.....	124
Figura 14.2.2: Montaje en físico de las pruebas.....	125
Figura 14.2.3: Conexión en físico del dispositivo USB de TDA.....	126
Figura 14.3.1: Dispositivo USB de TDA conectado.....	128
Figura 14.3.2: Prueba de sintonización y reproducción del programa 123 TV..	129
Figura 14.3.3: Prueba de sintonización y reproducción del programa SIBCHID.....	129
Figura 14.3.4: Prueba de sintonización y reproducción del programa Avila TV Móvil.....	130
Figura 14.3.5: Prueba de sintonización y reproducción del programa Meridiano, vista del escritorio.....	131
Figura 14.3.6: Prueba de sintonización y reproducción del programa Meridiano, aplicación maximizada.....	132
Figura 14.3.7: Prueba de sintonización y reproducción del programa Meridiano, modo pantalla completa.....	133
Figura 14.3.8: Prueba de sintonización y reproducción del programa CCTV, control de volumen.....	133
Figura 14.3.9: Montaje físico prueba de sintonización y reproducción del programa 123 TV.....	134
Figura 14.4.1: Ejecución del comando de desinstalación del programa.....	128
Figura 14.4.2: Mensaje de desinstalación del programa.....	129
Figura 14.4.3: Verificación de desinstalación.....	129

ÍNDICE DE TABLAS

Tabla 7.1.1: Funciones de la API Samsung, demodulador. [10].....	10
Tabla 7.1.2: Funciones de la API Samsung, sintonizador. [10].....	17
Tabla 7.1.3: Funciones API Samsung definidos por el desarrollador. [10].....	19
Tabla 7.1.4: Funciones desarrolladas en base a la API de Samsung y la librería estándar de C++.....	22
Tabla 9.1.1: Funciones de la API Libusb. [11].....	30
Tabla 9.1.2: Funciones desarrolladas en base de la API Libusb para proporcionar el soporte de conexión en caliente.....	46
Tabla 9.1.3: Funciones desarrolladas en base de la API Libusb para las transferencias USB de tipo control pertenecientes al demodulador.....	51
Tabla 9.1.4: Funciones desarrolladas en base de la API Libusb para las transferencias USB de tipo control pertenecientes al sintonizador.....	53
Tabla 9.1.5: Funciones desarrolladas en base de la API Libusb para las transferencias USB de tipo interrupción.....	58
Tabla 9.1.6: Distribución de las posiciones de memoria pertenecientes al buffer de 4,1 Mb.....	63
Tabla 10.1.1: Funciones de la API Libvlc. [13].....	65
Tabla 10.1.2: Funciones desarrolladas en base a la API Libvlc.....	72
Tabla 11.1.1: Empaquetamiento de widgets.....	79
Tabla 11.1.2: Empaquetamiento en la rejilla donde el valor de la coordenada X es 0.....	84
Tabla 11.1.3: Empaquetamiento en la rejilla donde el valor de la coordenada X es 1.....	84
Tabla 11.1.4: Widgets, descripción y funciones de enlace.....	88
Tabla 12.1.1: Código de Regla Udev para el dispositivo USB de TDA.....	98
Tabla 12.1.2: Código del script de instalación.....	101
Tabla 12.1.3: Código del script de desinstalación.....	104

METODOLOGÍA

6.1 Utilidades de Software.

Las rutinas de software desarrolladas en este trabajo de grado tienen como finalidad otorgar la capacidad reconocer al dispositivo USB de TDA cuando este se conecte o desconecte desde un computador (De escritorio o laptop) bajo un sistema operativo GNU/Linux, realizar acciones de control sobre el dispositivo USB, controlar los buses de datos, reproducir el contenido multimedia obtenido a través de los buses de datos del dispositivo y por último deben documentarse las rutinas desarrolladas y crear un manual de instalación del software desarrollado.

Se realizó un proceso de investigación sobre las herramientas existentes útiles con la finalidad de seleccionar aquellas que permitieran alcanzar los objetivos planteados.

En cuanto a la interacción USB anfitrión-dispositivo se optó por una librería de nombre Libusb que ofrece diversas soluciones en el campo de la comunicación USB, la librería escrita en lenguaje C posee un código muy sencillo de usar e interpretar, además el fabricante Cypress hace uso de esta librería para manejar sus dispositivos USB en un ambiente GNU/Linux.

Cabe destacar que Samsung el fabricante del dispositivo Tunner NIM proporciona una API para el control del dispositivo, la cual fue empleada en este proyecto.

Para el proceso de control multimedia ya el paquete de software VLC proporciona una API llamada Libvlc para utilizar el núcleo de este reproductor multimedia en software desarrollados por terceros, esta fue la librería usada para interactuar con el reproductor VLC.

En cuanto a la interfaz gráfica se utilizaron las bibliotecas de software de GTK debido al gran soporte que posee y a su abstracción a la hora de programar. Para la elaboración de la documentación se utilizó el software Doxygen, una poderosa herramienta destinada solo a la

documentación de software con muchas funcionalidades para este fin, la creación del manual de instalación se hizo utilizando LibreOffice un popular software libre de ofimática.

Como el proyecto de software fue desarrollado bajo un ambiente GNU/Linux se uso la distribución de nombre Fedora en su versión 24, ya que Fedora es excelente como sistema operativo de desarrollo de software debido a que maneja los repositorios de software más actualizados de las versiones de los programas y librerías usadas para el desarrollo de software.

El lenguaje de programación usado fue C++ debido a que las librerías Libusb, Libvlc, la Api de Samsung y el entorno gráfico GTK están escritos en C. C++ logra un nivel de abstracción superior a C facilitando el desarrollo y disminuyendo el tiempo del mismo sin perder la compatibilidad con rutinas escritas en C.

Como C++ fue seleccionado como lenguaje de programación para desarrollar por completo todo el proyecto se escogió como IDE de desarrollo el software CodeBlocks que permite desarrollar proyectos escritos en C++ haciendo uso del compilador g++, siendo este software muy popular en los entornos de desarrolladores en GNU/Linux.

Todas estas herramientas fueron usadas para alcanzar los objetivos planteados en este trabajo de grado y se describe su implementación en los puntos posteriores.

6.1.1 Esquema de Software.

El esquema de software empleado se basa en un modelo piramidal siendo las rutinas USB la base de la pirámide seguidas de las rutinas de la API de Samsung y luego las rutinas multimedia de VLC y por último el empaquetado gráfico. Como se observa en la **Figura 6.1.1** la base de la pirámide son las rutinas USB ya que sin ellas ninguna de las rutinas posteriores pueden llevarse a cabo.

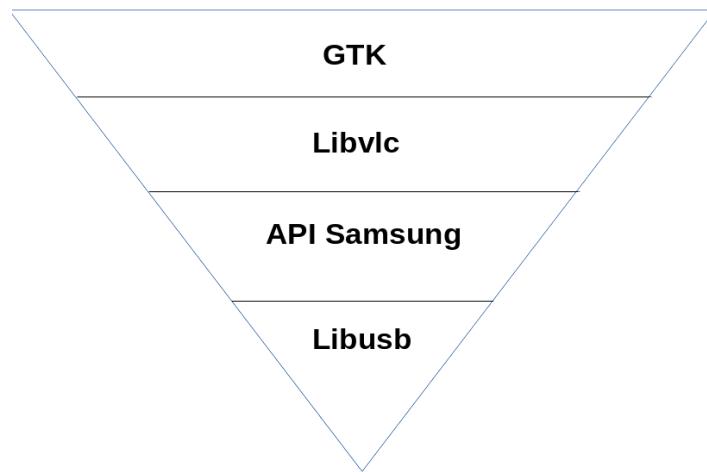


Figura 6.1.1: Pirámide de Dependencia.

Fue necesaria la utilización de hilos de programación, un hilo es simplemente una tarea que puede ser ejecutada al mismo tiempo que otra tarea en paralelo. Esto debido a que si el software estuviera solamente enfocado en un solo proceso como por ejemplo la obtención constante de paquetes BTS del dispositivo TDA este no podría manejar el reproductor multimedia para reproducir estos paquetes lo que hace necesario que sean empleado un hilo diferente para cada proceso haciendo posible que ambos procesos se ejecuten en paralelo.

El software desarrollado emplea tres grandes hilos de procesos que se describen a continuación.

- **Hilo principal:** Este representa al hilo principal de ejecución del programa, en este hilo se ejecutan las tareas de las funciones de la API de Samsung, se ejecutan las actividades del reproductor multimedia VLC y los procesos de la interfaz gráfica.
- **Hilo de Conexión en Caliente:** Este hilo se encarga del monitoreo constantemente de la conexión en caliente del dispositivo USB.
- **Hilo de adquisición de paquetes BTS:** Este hilo se encarga del proceso de obtención de los paquetes BTS desde el dispositivo USB.

7.1 La API de Samsung

Samsung ofrece una API para el manejo del dispositivo Tuner NIM DNOD22QXV104A definiendo funciones para el manejo del demodulador y sintonizador.

7.1.1 Funciones del demodulador SemcoTC90527.

En la **Tabla 7.1.1** se muestran el nombre de las funciones pertenecientes a la API Samsung para el manejo del demodulador y su descripción.

Tabla 7.1.1: Funciones de la API Samsung, demodulador. [10]

SemcoTC90527Init()	Descripción: Esta función inicializa el demodulador.
	Parámetros: Ninguno.
	Retorna: <ul style="list-style-type: none"> • 0

	<p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoTC90527SoftReset()	<p>Descripción: Esta función reinicia el demodulador.</p> <p>Parámetros: Ninguno.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoTC90527RegReset()	<p>Descripción: Esta función reiniciar los registros del demodulador.</p> <p>Parámetros: Ninguno.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoTC90527SleepOn()	<p>Descripción: Esta función coloca al demodulador en modo suspensión.</p> <p>Parámetros: Ninguno.</p>

	<p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoTC90527SleepOff()	<p>Descripción: Esta función saca del modo de suspensión al demodulador.</p> <p>Parámetros: Ninguno.</p>
	<p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoTC90527MasterLock()	<p>Descripción: Esta función retorna el estado de bloqueo del demodulador.</p> <p>Parámetros: Ninguno.</p>
	<p>Retorna:</p> <ul style="list-style-type: none"> • 1 <p>Dispositivo bloqueado.</p> <ul style="list-style-type: none"> • 0 <p>Dispositivo desbloqueado.</p>
SemcoTC90527_ISDBT_SNRResultCheck()	<p>Descripción: Esta función retorna la relación señal/ruido.</p> <p>Parámetros: Ninguno.</p>

	<p>Retorna:</p> <ul style="list-style-type: none"> • Punto flotante <p>Relación señal/ruido.</p>
SemcoTC90527_GetSignalQuality()	<p>Descripción: Esta función retorna escala de calidad de la señal.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • Estructura <p>Esta estructura aloja los siguientes valores:</p> <p>bLock, estado de bloqueo.</p> <p>SNR, relación señal/ruido.</p> <p>pre_BER, (Bit Error Rate) tasa de error binario, de la señal entrante.</p> <p>post_BER, tasa de error binario, de la señal corregida.</p> <p>mode_90527, modo de demodulación 0:DQPSK, 1:QPSK, 2:16QAM, 3:64QAM.</p> <p>code_rate, 0:1/2, 1:2/3, 2:3/4, 3:5/6, 4:7/8.</p> <p>hab_error, cantidad errores detectados en la señal corregida.</p>
	<p>Retorna:</p> <ul style="list-style-type: none"> • entero <p>Con el valor de 1, 2, 3, 4, 5, 6, 7, 8, 9 o 10. Mayor escala mejor calidad de la señal.</p>

SemcoTC90527_Set_TS_Output()	<p>Descripción: Esta función determinar el tipo la salida de los paquetes BTS;</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • Entero <p>El valor 0 designa serial, otro calor diferente designa paralelo.</p>
	<p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoTC90527_GetDatas()	<p>Descripción: Esta función retorna el estado de bloqueo, la relación señal/ruido, tasa de error binario, tipo de demodulación y errores a través del llenado de una estructura.</p>

	<p>Parámetros:</p> <ul style="list-style-type: none"> • Estructura <p>Esta estructura aloja los siguientes valores:</p> <p>bLock, estado de bloqueo.</p> <p>SNR, relación señal/ruido.</p> <p>pre_BER, (Bit Error Rate) tasa de error binario, de la señal entrante.</p> <p>post_BER, tasa de error binario, de la señal corregida.</p> <p>mode_90527, modo de demodulación 0:DQPSK, 1:QPSK, 2:16QAM, 3:64QAM.</p> <p>code_rate, 0:1/2, 1:2/3, 2:3/4, 3:5/6, 4:7/8.</p> <p>hab_error, cantidad errores detectados en la señal corregida.</p>
	<p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoTC90527_GetIfAgcLevel()	<p>Descripción: Esta función retorna el nivel de la intensidad de la señal.</p> <p>Parámetros: Ninguno.</p>

	<p>Retorna:</p> <ul style="list-style-type: none">• Punto flotante <p>Intensidad de la señal.</p>
SemcoTC90527_GetSSI()	<p>Descripción: Esta función retorna la escala de la intensidad de señal.</p> <p>Parámetros: Ninguno.</p> <p>Retorna:</p> <ul style="list-style-type: none">• entero <p>Con el valor de 1, 2, 3, 4, 5, 6, 7, 8, 9 o 10. Mayor escala mayor intensidad de la señal.</p>

7.1.2 Funciones del sintonizador SemcoSTV4100.

En la **Tabla 7.1.2** se muestran el nombre de las funciones pertenecientes a la API Samsung para el manejo del sintonizador y su descripción.

Tabla 7.1.2: Funciones de la API Samsung, sintonizador. [10]	
SemcoSTV4100_Initialize()	<p>Descripción: Esta función inicializa el sintonizador.</p> <p>Parámetros: Ninguno.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoSTV4100_SetFrequency()	<p>Descripción: Esta función sintoniza una frecuencia deseada.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • entero <p>Indica el valor de la frecuencia en Hz a sintonizar.</p> <ul style="list-style-type: none"> • entero <p>Indica el ancho de bando, por defecto el de Venezuela 6 MHz.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p>

	<ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoSTV4100_GetLockStatus()	<p>Descripción: Esta función retorna el estado de bloqueo del sintonizador.</p> <p>Parámetros: Ninguno.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 1 <p>Dispositivo bloqueado.</p> <ul style="list-style-type: none"> • 0 <p>Dispositivo desbloqueado.</p>
SemcoSTV4100_On()	<p>Descripción: Esta función coloca al sintonizador en modo de suspensión.</p> <p>Parámetros: Ninguno.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
SemcoSTV4100_StandBy()	<p>Descripción: Esta función saca del modo de suspensión al sintonizador.</p> <p>Parámetros: Ninguno.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso exitoso.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>

7.1.3 Funciones definidas por el desarrollador

La API no posee funciones para la comunicación USB, estas deben ser creadas por el desarrollador al igual que la función de retardo, estas se especifican en la **Tabla 7.1.3**.

Las funciones de comunicación USB fueron desarrollados con la API Libusb y la función de retardo utiliza una función de la biblioteca estándar del lenguaje C++.

Tabla 7.1.3: Funciones API Samsung definidos por el desarrollador. [10]

Funciones	Descripción
STV4100_I2C_Write()	Se encarga de realizar la transferencia de control en modo de escritura del sintonizador.
STV4100_I2C_Read()	Se encarga de realizar la transferencia de control en modo de lectura del sintonizador.
TC90527_I2cWrite()	Se encarga de realizar la transferencia de control en modo de escritura del demodulador.
TC90527_I2cRead()	Se encarga de realizar la transferencia de control en modo de lectura del demodulador.
procedimiento_retardo_milisegundos()	Realiza retardos en mili segundos.

7.1.4 Procesos de inicialización y sintonización

Para el procedimiento de inicialización del demodulador y sintonizador se realiza el siguiente llamado de funciones de la API como se muestra en la **Figura 7.1.1**.

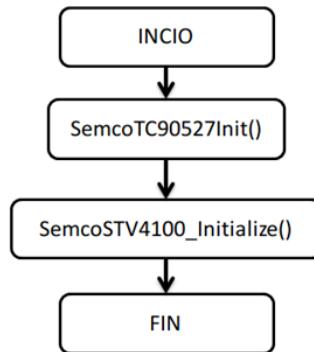


Figura 7.1.1: Procedimiento de inicialización del demodulador y sintonizador.

El procedimiento de sintonización se describe a continuación en la

Figura 7.1.2.

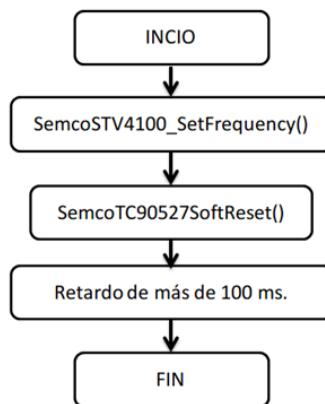


Figura 7.1.2: Procedimiento de sintonización de una frecuencia.

En la **Tabla 7.1.4** se describen las funciones desarrolladas que tiene como base las funciones de la API de Samsung y de la librería estándar de C++.

Tabla 7.1.4: Funciones desarrolladas en base a la API de Samsung y la librería estándar de C++.

procedimiento_retardo_milisegundo_s()	<p>Descripción: Esta función es la encargada de realizar retardos en mili segundos.</p> <p>Variables: Ninguno.</p> <p>Funciones de la librería estándar de C++:</p> <ul style="list-style-type: none"> • usleep() <p>tiempo_ms, parámetro entero que representa los mili segundos de retardo.</p>
procedimiento_inicializar_demodulador_sintonizador()	<p>Descripción: Esta función se encarga de inicializar tanto al demodulador como al sintonizador.</p> <p>Variables: Ninguno.</p> <p>Funciones de la API Samsung:</p> <ul style="list-style-type: none"> • SemcoTC90527Init() • SemcoSTV4100_Initialize() <p>Funciones de la librería estándar de C++:</p> <ul style="list-style-type: none"> • procedimiento_retardo_milisegundos() <p>tiempo_ms, parámetro entero que representa los mili segundos de retardo.</p>
procedimiento_sintonizar_frecuencia()	<p>Descripción: Esta función realiza la función de sintonizar una frecuencia deseada en el sintonizador. Recibe un entero como parámetro con la frecuencia a sintonizar en Hz.</p> <p>Variables: Ninguno.</p>

	<p>Funciones de la API Samsung:</p> <ul style="list-style-type: none"> • SemcoSTV4100_SetFrequency() <p>nFrequency_KHz, parámetro entero indica el valor de la frecuencia en Hz a sintonizar.</p> <p>BW, parámetro entero indica el ancho de bando, por defecto el de Venezuela 6 MHz.</p> <p>Funciones de la librería estándar de C++:</p> <ul style="list-style-type: none"> • procedimiento_retardo_milisegundos() <p>tiempo_ms, parámetro que representa los milisegundos de retardo.</p>
procedimiento_programa_521000_5 7856() procedimiento_programa_521000_5 7857() procedimiento_programa_521000_5 7858() procedimiento_programa_521000_5 7859() procedimiento_programa_521000_5 7860() procedimiento_programa_521000_5 7880() procedimiento_programa_533000_5 7920() procedimiento_programa_533000_5 7921() procedimiento_programa_533000_5 7922()	<p>Descripción: Este grupo de funciones representan a cada uno de los programas de la TDA venezolana las funciones tienen el nombre de la forma procedimiento_programa_NUMEROx_NUMEROy(). NUMEROx representa la frecuencia en Hertz del grupo de programas y NUMEROy representa el ID de un programa en específico.</p> <p>Estas funciones son las encargadas de sintonizar una frecuencia específica, seleccionar el ID del programa a sintonizar y de establecer en alto una bandera que señala que una frecuencia a sido establecida.</p>

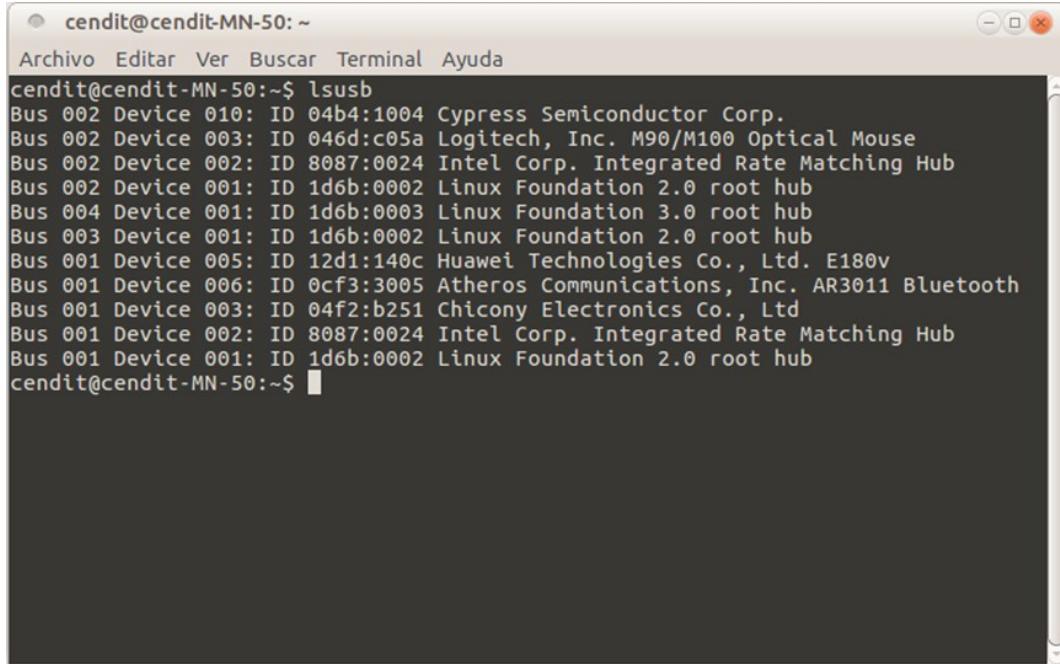
<pre> procedimiento_programa_533000_5 7944() procedimiento_programa_539000_5 7952() procedimiento_programa_539000_5 7953() procedimiento_programa_539000_5 7954() procedimiento_programa_539000_5 7955() procedimiento_programa_539000_5 7956(); procedimiento_programa_539000_5 7976() procedimiento_programa_527000_5 7888() procedimiento_programa_527000_5 7889() procedimiento_programa_527000_5 7890() procedimiento_programa_527000_5 7891() procedimiento_programa_527000_5 7892() procedimiento_programa_527000_5 7912() </pre>	<p>Variables:</p> <ul style="list-style-type: none"> • id <p>Esta variable aloja una cadena de caracteres que especifica el ID del programa.</p> <ul style="list-style-type: none"> • estado_frecuencia <p>Esta variable de tipo boolena toma valor de verdadero cuando una frecuencia fue establecida y toma el valor de falso cuando una frecuencia nunca ha sido establecida.</p> <p>Funciones de la API Samsung:</p> <ul style="list-style-type: none"> • SemcoTC90527Init() • SemcoSTV4100_Initialize() • SemcoSTV4100_SetFrequency() <p>nFrequency_KHz, parámetro entero indica el valor de la frecuencia en Hz a sintonizar.</p> <p>BW, parámetro entero indica el ancho de banda, por defecto el de Venezuela 6 MHz.</p> <p>Funciones de la librería estándar de C++:</p> <ul style="list-style-type: none"> • procedimiento_retardo_milisegundos() <p>tiempo_ms, parámetro que representa los milisegundos de retardo.</p>
--	---

funcion_leer_id()	<p>Descripción: Esta función retorna la cadena de caracteres que representa un ID de programa.</p> <p>Variables:</p> <ul style="list-style-type: none">• id <p>Esta variable aloja una cadena de caracteres que especifica el ID del programa.</p>
-------------------	--

8.1 GNU/Linux y el Dongel USB de TDA

Al momento de la inserción del dispositivo TDA en el puerto USB del anfitrión o PC, se realiza el proceso de enumeración en el cual se le asigna una dirección al dispositivo y se obtiene la descripción de este dispositivo (configuraciones, interfaces, endpoints, etc). El núcleo de Linux proporciona comandos que nos brindan información sobre los dispositivos USB.

El comando **lsusb** ejecutado en una terminal nos proporciona una lista de los dispositivos USB conectados al anfitrión, en este caso la PC, en la **Figura 8.1.1** se muestra la salida arrojada por el comando.



```
cendit@cendit-MN-50: ~
Archivo Editar Ver Buscar Terminal Ayuda
cendit@cendit-MN-50:~$ lsusb
Bus 002 Device 010: ID 04b4:1004 Cypress Semiconductor Corp.
Bus 002 Device 003: ID 046d:c05a Logitech, Inc. M90/M100 Optical Mouse
Bus 002 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 004 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
Bus 003 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 001 Device 005: ID 12d1:140c Huawei Technologies Co., Ltd. E180v
Bus 001 Device 006: ID 0cf3:3005 Atheros Communications, Inc. AR3011 Bluetooth
Bus 001 Device 003: ID 04f2:b251 Chicony Electronics Co., Ltd
Bus 001 Device 002: ID 8087:0024 Intel Corp. Integrated Rate Matching Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
cendit@cendit-MN-50:~$
```

Figura 8.1.1: Salida de la ejecución del comando lsusb.

Como se observa en la **Figura 8.1.1** el dispositivo USB de TDA se identifica como Cypress Semiconductor Corp. Con el VendorID: (0x4b4)h y con el ProductID: (0x1004)h, utiliza el BUS 02 y su número de dispositivo es 010, el número de dispositivo es asignado por el anfitrión.

Para obtener el descriptor del dispositivo USB Cypress. Se utilizó el comando **lsusb -v**, el cual brinda información sobre los descriptores de los dispositivos conectados al equipo. La **Figura 8.1.2** muestra la información obtenida por el sistema operativo del descriptor del dispositivo Cypress, se observa que el dispositivo posee una sola configuración que contiene una sola interfaz, un solo endpoint especificado, ninguna clase en especial asignada, los ID Vendor y Product del dispositivo, su fabricante es señalado como Cypress, dentro de la configuración se observa que la alimentación

del dispositivo está definida para que el dispositivo obtenga su alimentación a través del puerto físico del anfitrión, en la única interfaz identificada como interfaz0 se encuentra especificado el endpoint (0x82)h como endpoint de tipo IN, entrada hacia el anfitrión, tipo de transferencia Interrupción, tipo de sincronización ninguna, el tipo del endpoint se especifica para datos, número máximo de paquetes 1 x 1024 bytes, cada paquete en un mircoframe, cada 125 micro segundos. El endpoint (0x82)h es el buffer en el dispositivo USB encargado de alojar la información de los paquetes BTS de 1024 bytes de longitud que serán enviados al anfitrión cada 125 micro segundos.

Esta información es de utilidad a la hora de programar un controlador en espacio de usuario usando la API Libusb para definir los valores y parámetros de las funciones que realizan las tareas de control y transferencias de datos ya que estos datos otorgados a través de los comandos **lsusb** y **lsusb-v** definen el comportamiento interno del dispositivo USB de TDA y el modo de interacción con el anfitrión al momento de comunicarse.

```
cendit@cendit-MN-50: ~
cendit@cendit-MN-50:~$ lsusb -v

Bus 002 Device 017: ID 04b4:1004 Cypress Semiconductor Corp.
Device Descriptor:
  bLength          18
  bDescriptorType   1
  bcdUSB         2.00
  bDeviceClass      0 (Defined at Interface level)
  bDeviceSubClass    0
  bDeviceProtocol     0
  bMaxPacketSize0     64
  idVendor        0x04b4 Cypress Semiconductor Corp.
  idProduct        0x1004
  bcdDevice        0.00
  iManufacturer       1 Cypress
  iProduct          2 DMB-TV
  iSerial           0
  bNumConfigurations  1
Configuration Descriptor:
  bLength          9
  bDescriptorType   2
  wTotalLength     25
  bNumInterfaces     1
  bConfigurationValue  1
  iConfiguration      0
  bmAttributes       0x80
    (Bus Powered)
  MaxPower         100mA
Interface Descriptor:
  bLength          9
  bDescriptorType   4
  bInterfaceNumber    0
  bAlternateSetting   0
  bNumEndpoints      1
  bInterfaceClass     255 Vendor Specific Class
  bInterfaceSubClass  0
  bInterfaceProtocol  0
  iInterface          0
Endpoint Descriptor:
  bLength          7
  bDescriptorType   5
  bEndpointAddress  0x82 EP 2 IN
  bmAttributes       3
    Transfer Type      Interrupt
    Sync Type          None
    Usage Type          Data
  wMaxPacketSize     0x0400 1x 1024 bytes
  bInterval          1
Device Qualifier (for other device speed):
  bLength          10
  bDescriptorType   6
  bcdUSB         2.00
  bDeviceClass      0 (Defined at Interface level)
  bDeviceSubClass    0
  bDeviceProtocol     0
  bMaxPacketSize0     64
  bNumConfigurations  1
Device Status: 0x0000
  (Bus Powered)
```

Figura 8.1.2: Salida de la ejecución del comando `lusb-v`.

9.1 La API Libusb

Para el desarrollo del controlador USB, se decidió usar un controlador en espacio de usuario, usando la librería Libusb, que es una API que comunica los procesos USB creados en el espacio de usuario con la interfaz de llamadas al sistema y estas a su vez se comunican con el núcleo GNU/Linux que maneja el hardware de los procesos USB.

Libusb es una biblioteca escrita en lenguaje C que proporciona acceso genérico a dispositivos USB. Está destinada a ser utilizado por los desarrolladores para facilitar la producción de aplicaciones que se comunican con hardware USB.

Es portátil, utilizando una única API multiplataforma, proporciona acceso a dispositivos USB en Linux, OS X, Windows, Android, OpenBSD, etc.

A continuación el **Tabla 9.1.1** se listan las funciones usadas de la librería conjuntamente con una descripción.

Tabla 9.1.1: Funciones de la API Libusb. [11]

libusb_init()	Descripción: Crea una sesión de la librería.
	Parámetros: <ul style="list-style-type: none"> • libusb_context Representa una sesión de la librería.
	Retorna: <ul style="list-style-type: none"> • 0 En caso de éxito o un código <ul style="list-style-type: none"> • LIBUSB_ERROR

	<p>En caso de error.</p>
libusb_exit()	<p>Descripción: Cierra una sesión de la librería.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_context <p>Representa una sesión de la librería.</p> <p>Retorna: Nada.</p>
libusb_set_debug()	<p>Descripción: Establece el nivel de depuración.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_context <p>Representa una sesión de la librería.</p> <ul style="list-style-type: none"> • nivelDeDebug <p>Variable de tipo entero que define el nivel de depuración de la librería.</p> <p>LIBUSB_LOG_LEVEL_NONE (nivelDeDebug=0): ningún mensaje impreso por la biblioteca (predeterminado).</p> <p>LIBUSB_LOG_LEVEL_ERROR (nivelDeDebug =1): los mensajes de error se imprimen en stderr (Standard error stream).</p> <p>LIBUSB_LOG_LEVEL_WARNING (nivelDeDebug =2): mensajes de advertencia y de error se imprimen en stderr.</p> <p>LIBUSB_LOG_LEVEL_INFO (nivelDeDebug =3): los mensajes de</p>

	<p>información se imprimen en stdout (Standard output), los mensajes de advertencia y de error se imprimen en stderr.</p> <p>LIBUSB_LOG_LEVEL_DEBUG (nivelDeDebug =4): se imprimen mensajes de depuración e información a stdout, advertencias y errores a stderr.</p> <p>Retorna: Nada.</p>
libusb_open()	<p>Descripción: Establece un manejador a nivel de software de un dispositivo USB detectado por el sistema.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_device <p>Es una variable que representa a un dispositivo USB detectado por el sistema.</p> <ul style="list-style-type: none"> • libusb_device_handle <p>Es una variable que permite el manejo del dispositivo a nivel de software.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso de éxito.</p> <ul style="list-style-type: none"> • LIBUSB_ERROR_NO_MEM <p>En el fallo de asignación de memoria.</p> <ul style="list-style-type: none"> • LIBUSB_ERROR_ACCESS

	<p>Si el usuario tiene permisos insuficientes.</p> <ul style="list-style-type: none"> • LIBUSB_ERROR_NO_DEVICE <p>Si el dispositivo ha sido desconectado.</p> <ul style="list-style-type: none"> • LIBUSB_ERROR <p>En otro error.</p>
libusb_close()	<p>Descripción: Cierra un manejador a nivel de software de un dispositivo USB.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_device_handle <p>Es una variable que permite el manejo del dispositivo a nivel de software.</p> <p>Retorna: Nada.</p>
libusb_hotplug_register_callback ()	<p>Descripción: Función que proporciona una solución a la conexión en caliente (hotplug) para dispositivos USB. Esta función filtra los dispositivos USB para detectar cuando un dispositivo es despachado de el sistema .</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_context <p>Representa una sesión de la librería.</p> <ul style="list-style-type: none"> • libusb_hotplug_event

	<p>Selecciona que eventos filtrar para un dispositivo USB en el ámbito de conexión en caliente, LIBUSB maneja dos eventos:</p> <p>LIBUSB_HOTPLUG_EVENT_DEVICE_ARRIVED: Un dispositivo ha arribado y está listo para usar.</p> <p>LIBUSB_HOTPLUG_EVENT_DEVICE_LEFT: Un dispositivo se ha despachado y no se encuentra disponible.</p> <ul style="list-style-type: none"> • <code>libusb_hotplug_flag</code> <p>Son valores enteros que se usan de bandera para armar la función de llamada:</p> <p>LIBUSB_HOTPLUG_NO_FLAGS: valor por defecto cuando no se utiliza ninguna bandera.</p> <p>LIBUSB_HOTPLUG_ENUMERATE: Arma a la función <code>libusb_hotplug_callback_fn</code> para que se dispare si algún dispositivo USB es detectado con las características especificadas.</p> <ul style="list-style-type: none"> • <code>vendor_id</code> <p>Valor entero para filtrar dispositivos por Vendor ID.</p> <ul style="list-style-type: none"> • <code>product_id</code> <p>Valor entero para filtrar dispositivos por Product ID.</p> <ul style="list-style-type: none"> • <code>dev_class</code>
--	---

	<p>Indica la clase de dispositivo USB a detectar.</p> <ul style="list-style-type: none"> • <code>libusb_hotplug_callback_fn</code> <p>Es una apuntador a una función de llamada que es invocado cuando una coincidencia del dispositivo USB ocurre sea en un evento de arribo o despacho.</p> <ul style="list-style-type: none"> • <code>user_data</code> <p>Es un apuntador a cualquier dato que se desea pasar a la función.</p> <ul style="list-style-type: none"> • <code>libusb_hotplug_callback_handle</code> • <p>Variable que permite manejar el dispositivo USB.</p>
	<p>Retorna:</p> <ul style="list-style-type: none"> • <code>LIBUSB_SUCCESS</code> <p>En caso de éxito o un código</p> <ul style="list-style-type: none"> • <code>LIBUSB_ERROR</code> <p>Para denotar un error.</p>
<code>libusb_hotplug_callback_fn()</code>	<p>Descripción: Es la función de llamada que se ejecuta cuando el dispositivo USB específico es detectado.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • <code>libusb_context</code> <p>Comparte la misma información de</p>

	<p>parámetro con la función libusb_hotplug_register_callback</p> <ul style="list-style-type: none"> • libusb_device <p>Es una variable que representa a un dispositivo USB detectado por el sistema. En este caso el dispositivo detectado.</p> <ul style="list-style-type: none"> • libusb_hotplug_event <p>Comparte la misma información de parámetro con la función libusb_hotplug_register_callback</p> <ul style="list-style-type: none"> • user_data <p>Comparte la misma información de parámetro con la función libusb_hotplug_register_callback</p> <p>Retorna: Booleano si esta devolución de llamada ha terminado de procesar eventos. Devolver 1 hará que esta devolución de llamada se cancele.</p>
libusb_control_transfer()	<p>Descripción: Esa función permite realizar y ejecutar transferencia de control de entrada y salida desde el anfitrión hacia el dispositivo USB.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_device_handle <p>Variable LIBUSB para manejar el dispositivo USB.</p> <ul style="list-style-type: none"> • bmRequestType <p>El campo tipo de solicitud para el paquete de configuración de una</p>

	<p>transferencia de control.</p> <ul style="list-style-type: none"> • bRequest <p>El campo de solicitud para el paquete de configuración de una transferencia de control.</p> <ul style="list-style-type: none"> • wValue <p>El campo de valor para el paquete de configuración de una transferencia de control.</p> <ul style="list-style-type: none"> • wIndex <p>El campo de dirección para el paquete de configuración de una transferencia de control.</p> <ul style="list-style-type: none"> • data <p>Apuntador al buffer de datos.</p> <ul style="list-style-type: none"> • wLength <p>Indica la longitud del buffer de datos.</p> <ul style="list-style-type: none"> • timeout <p>Tiempo límite para recibir una respuesta en milisegundos.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • En caso de éxito <p>El número de bytes transferidos.</p> <ul style="list-style-type: none"> • LIBUSB_ERROR_TIMEOUT <p>Si la transferencia se ha agotado.</p>
--	--

	<ul style="list-style-type: none"> • LIBUSB_ERROR_PIPE <p>Si la solicitud de control no era compatible con el dispositivo.</p>
libusb_alloc_transfer()	<p>Descripción: Esta función asigna espacio para una transferencia USB de tipo masivo, interrupción o isócrona.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • iso_packets <p>Es una variable de tipo entero que representa el número de paquetes isócronos.</p> <p>Retorna:</p> <p>libusb_transfer, es una variable de tipo estructura que contiene los siguientes campos y representa a una transferencia USB:</p> <ul style="list-style-type: none"> • libusb_device_handle <p>Variable Libusb para manejar el dispositivo USB.</p> <ul style="list-style-type: none"> • Flags <p>Son valores enteros que se usan para denotar las acciones a realizar dependiendo del valor de la bandera:</p> <p>LIBUSB_TRANSFER_SHORT_NO_T_OK (0), reporta transferencias incompletas cortas como errores.</p> <p>LIBUSB_TRANSFER_FREE_BUFFER(1), libera automáticamente el buffer de transferencia durante el llamado de la función</p>

	<p>libusb_free_transfer().</p> <p>LIBUSB_TRANSFER_FREE_TRANSFER (2), automáticamente llama a la función libusb_free_transfer() después que la función de llamada retorna.</p> <p>LIBUSB_TRANSFER_ADD_ZERO_PACKET (3), finaliza las transferencias del endpoint con un paquete de longitud cero adicional.</p> <ul style="list-style-type: none"> • endpoint <p>Dirección del endpoint en el que se enviará esta transferencia.</p> <ul style="list-style-type: none"> • Type <p>Tipo de endpoint, interrupción, masivo o isócrono.</p> <ul style="list-style-type: none"> • Timeout <p>tiempo límite para esta transferencia en milisegundos.</p> <ul style="list-style-type: none"> • libusb_transfer_status <p>son valores enteros que se usan para denotar los estados de las transferencias:</p> <p>LIBUSB_TRANSFER_COMPLETE_D (0): La transferencia se completó sin errores.</p> <p>LIBUSB_TRANSFER_ERROR (1): La transferencia falló.</p> <p>LIBUSB_TRANSFER_TIMED_OUT</p>
--	--

	<p>(2): Tiempo de realización de transferencia finalizado.</p> <p>LIBUSB_TRANSFER_CANCELLED</p> <p>(3): La transferencia fue cancelada.</p> <p>LIBUSB_TRANSFER_STALL (4): Condición de halt (condición de parada) detectada en el endpoint.</p> <p>LIBUSB_TRANSFER_NO_DEVICE</p> <p>(5): El dispositivo USB fue desconectado.</p> <p>LIBUSB_TRANSFER_OVERFLOW</p> <p>(6): El dispositivo envió más datos que los solicitados.</p> <ul style="list-style-type: none"> • length <p>Longitud del buffer.</p> <ul style="list-style-type: none"> • actual_length <p>Longitud de los datos que fueron transferidos.</p> <ul style="list-style-type: none"> • libusb_transfer_cb_fn <p>Apuntador a la función de llamada que se ejecuta cuando finaliza una transferencia de datos.</p> <ul style="list-style-type: none"> • user_data <p>Apuntador de datos pasado a la función de llamada.</p> <ul style="list-style-type: none"> • Buffer <p>apuntador al buffer de datos.</p>
--	--

	<ul style="list-style-type: none"> • num_iso_packets <p>Número de paquetes isócronos en caso de transferencias isócronas.</p> <ul style="list-style-type: none"> • libusb_iso_packet_descriptor <p>Estructura para manejar datos isócronos.</p>
libusb_fill_interrupt_transfer()	<p>Descripción: Llena la instancia libusb_transfer con información sobre la transferencia que desea realizar.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_device_handle <p>Variable Libusb para manejar el dispositivo USB.</p> <ul style="list-style-type: none"> • endpoint <p>Dirección del endpoint en el que se enviará esta transferencia.</p> <ul style="list-style-type: none"> • buffer <p>Apuntador al buffer de datos.</p> <ul style="list-style-type: none"> • length <p>Longitud del buffer.</p> <ul style="list-style-type: none"> • callback <p>Apuntador a la función de llamada que se ejecuta cuando finaliza una transferencia de datos.</p> <ul style="list-style-type: none"> • user_data

	<p>Apuntador de datos pasado a la función de llamada.</p> <ul style="list-style-type: none"> • timeout <p>Tiempo límite para esta transferencia en milisegundos.</p> <p>Retorna: Nada</p>
libusb_submit_transfer()	<p>Descripción: Esta función ejecuta una transferencia USB de tipo masivo, interrupción o isócrona.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_transfer <p>Representa una transferencia USB.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>Si fue exitosa la ejecución.</p> <ul style="list-style-type: none"> • LIBUSB_ERROR_NO_DEVICE <p>Si el dispositivo ha sido desconectado.</p> <ul style="list-style-type: none"> • LIBUSB_ERROR_BUSY <p>Si la transferencia ya ha sido enviada.</p> <ul style="list-style-type: none"> • LIBUSB_ERROR_NOT_SUPPORTED <p>Si los indicadores de transferencia no son compatibles con el sistema operativo.</p>

	<ul style="list-style-type: none"> • LIBUSB_ERROR <p>Para denotar otro error.</p>
libusb_free_transfer()	<p>Descripción: Esta función permite desasignar las transferencias USB.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_transfer <p>Representa una transferencia USB.</p> <p>Retorna: Nada</p>
libusb_handle_events()	<p>Descripción: Esta función permite manejar los eventos ocurridos en una transferencia USB.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libusb_context <p>Representa una sesión de la librería.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>En caso de éxito o un código</p> <ul style="list-style-type: none"> • LIBUSB_ERROR <p>En caso de error.</p>

9.1.1 Transferencias asincrónicas

Para las transferencias de tipo interrupción que reciben la información de audio y video en el anfitrión se utilizó la interfaz asincrónica para transferencias de interrupción de Libusb, una interfaz asincrónica se refiere a la ocurrencia de eventos de forma independiente de la corriente

principal del programa. El proceso para iniciar y realizar una transferencia de interrupción asincrónica contiene los siguientes pasos:

1. **Asignación:** consiste en asignar una variable de tipo **libusb_transfer** para la transferencia, este paso implica asignar memoria para una transferencia USB. En esta etapa, la transferencia está "en blanco" sin detalles sobre qué tipo de Entrada/ Salida se utilizará para la transferencia.
La asignación se realiza con la función **libusb_alloc_transfer(iso_packets)**, se debe utilizar esta función en lugar de asignar las transferencias de manera propia, la función retorna una variable **libusb_transfer** y recibe una variable denominada **iso_packets** de tipo entero con el número de paquetes isócronos, para transferencia de tipo interrupción y masivas la variable debe tener un valor de 0.
2. **Llenado:** Llena la instancia **libusb_transfer** con información sobre la transferencia que desea realizar, este paso es donde se toma una transferencia previamente asignada y se completa con información para determinar el tipo y la dirección del mensaje, el buffer de datos, la función de devolución de llamada, etc.
Puede llenar los campos requeridos se utilizan las funciones de ayuda proporcionados por la librería debido a que la transferencia de contenido multimedia es de tipo interrupción se utilizó la función **libusb_fill_interrupt_transfer()**.
3. **Tratamiento de la terminación:** se examinan los resultados de la transferencia en la estructura:
libusb_transfer->libusb_transfer_status.
4. **Desasignación:** limpiar los recursos, cuando se ha realizado una transferencia (es decir, se ha invocado la función de devolución de

llamada), se aconseja liberar la transferencia (a menos que desee volver a enviarla). Las transferencias se desasignan a través de la función **libusb_free_transfer(libusb_transfer)**. [11]

9.1.2 Rutinas USB

A continuación se especifican las rutinas USB creadas que permiten la conexión en caliente del dispositivo el manejo de los buses de entrada y salida y el control del dispositivo.

9.1.2.1 Rutinas de Conexión en Caliente

En la **Tabla 9.1.2** se describen las funciones desarrolladas para la conexión en caliente para el dispositivo USB de TDA y las funciones de la API Libusb que son llamados junto los parámetros utilizados.

Tabla 9.1.2: Funciones desarrolladas en base de la API Libusb para proporcionar el soporte de conexión en caliente.

procedimiento_hotplug()	<p>Descripción: Esta función se ejecuta en un hilo secundario del programa, es ejecutado al inicio del programa y se mantiene en ejecución dentro de un ciclo infinito hasta que el programa finalice, se encarga de iniciar una sesión de la librería Libusb, libusb_init(), y de finalizarla, libusb_exit(), también maneja el filtrado y registro el dispositivo USB si este arriba o se despacha del puerto USB, libusb_hotplug_register_callback().</p>
	<p>Variables:</p> <ul style="list-style-type: none"> • estado_hotplug

	<p>Para manejar los estados de conexión en caliente se definió una variable de tipo entero llamado estado_hotplug la cual puede tomar los siguientes valores:</p> <p>Si estado_hotplug igual 0, dispositivo no conectado.</p> <p>Si estado_hotplug igual 1, dispositivo conectado.</p> <p>Si estado_hotplug igual 2, dispositivo desconectado.</p>
	<p>Funciones Libusb:</p> <ul style="list-style-type: none"> • libusb_init() <p>contexto_usb, para indicar la sesión de Libusb usada.</p> <ul style="list-style-type: none"> • libusb_exit() <p>contexto_usb, para indicar la sesión de Libusb usada.</p> <ul style="list-style-type: none"> • libusb_hotplug_register_callback() <p>contexto_usb, para indicar la sesión de Libusb usada.</p> <p>(libusb_hotplug_event), banderas que indican los eventos LIBUSB a tomar en cuenta, en este caso arribo y despacho del dispositivo USB.</p> <p>LIBUSB_HOTPLUG_ENUMERATE, arma la función de llamada</p> <p>libusb_hotplug_callback_fn()</p>

	<p>para que se dispare si algún dispositivo USB es detectado con las características especificadas.</p> <p>1204, IDVendor del dispositivo a detectar, en este caso el del dispositivo USB de Cypress.</p> <p>4100, IDProduct del dispositivo a detectar, en este caso el del dispositivo USB de Cypress.</p> <p>LIBUSB_HOTPLUG_MATCH_ANY, bandera que indica cualquier clase de dispositivo USB a detectar.</p> <p>funcion_de_llamada_hotplug, apuntador a la función de llamada que se ejecuta cuando el dispositivo USB especificado es detectado.</p> <p>NULL, ningún dato es pasado a la función de llamada.</p> <p>manejador_dispositivo_usb_hotplug, variable Libusb para el manejo del dispositivo USB.</p>
funcion_de_llamada_hotplug()	<p>Descripción: Función de llamada que se ejecuta luego de que un dispositivo coincida con las características del dispositivo especificado para la conexión en caliente, se encarga de abrir el dispositivo usando la función libusb_open() y de cerrarlo usando la función libusb_close() y cambiar el valor del variable estado_hotplug.</p> <p>Variables:</p> <ul style="list-style-type: none"> • manejador_dispositivo_usb

	<p>Esta función es la encargada de crear o eliminar una variable global que define un manejador a nivel de software de un dispositivo USB.</p> <p>Funciones Libusb:</p> <ul style="list-style-type: none"> • libusb_open() <p>dispositivo_usb_hotplug_funcion_llamada, Variable Libusb para manejar el dispositivo USB.</p> <p>manejador_dispositivo_usb, Es una variable que representa a un dispositivo USB detectado por el sistema.</p> <ul style="list-style-type: none"> • libusb_close() <p>manejador_dispositivo_usb, Variable Libusb para manejar el dispositivo USB.</p>
funcion_leer_estado_hotplug()	<p>Descripción: Esta función retorna el valor de la variable estado_hotplug. Para que otras funciones tengan acceso al estado de conexión en caliente del dispositivo USB.</p> <p>Variables:</p> <ul style="list-style-type: none"> • estado_hotplug <p>Esta variable representa el estado de conexión en caliente de un dispositivo USB.</p> <p>Funciones Libusb: Ninguna.</p>

9.1.2.1 Rutinas de Transferencias de Control

Estas rutinas son las que especifica la API de Samsung que deben ser creadas por el desarrollador, las cuales se encargan de realizar las transferencias de control USB. Los datos recibidos o enviados a través de estas transferencias de control luego viajan a través del protocolo I2C desde o hacia el chip de USB Cypress, que luego se comunica con el demodulador o el sintonizador del dispositivo Samsung Tuner NIM DNOD22QXV104A lo que permite realizar acciones de control en el dispositivo.

Las funciones de transferencias de control son cuatro, dos pertenecientes al sintonizador y los dos restantes al demodulador, cada dispositivo tienen una función de lectura y uno de escritura desde el anfitrión estas están basadas en la función **libusb_control_transfer()** de la API LIBUSB, las transferencias de control se dirigen en el caso del dispositivo de USB de TDA al endpoint 0. En la **Tabla 9.1.3** se describen las funciones de control USB pertenecientes al demodulador.

Tabla 9.1.3: Funciones desarrolladas en base de la API Libusb para las transferencias USB de tipo control pertenecientes al demodulador.

Tipo de operación	Función USB del Demodulador	Parámetros libusb_control_transfer()
Lectura	<p>TC90527_I2cRead():</p> <p>Esta función recibe como parámetro la dirección del registro a leer en el demodulador, retorna el dato leído.</p>	<ul style="list-style-type: none"> • libusb_device_handle <p>Recibe una variable Libusb para manejar el dispositivo USB.</p> <ul style="list-style-type: none"> • bmRequestType <p>Es el tipo de operación que se desea realizar, recibe el valor de (0xC0)h para indicar que es una operación de lectura.</p> <ul style="list-style-type: none"> • brequest <p>Este byte es usado para indicar que función de transferencia I2C se ejecutará en el firmware, el valor para lectura desde el demodulador está definido por (0xA3)h.</p> <ul style="list-style-type: none"> • wValue <p>Es la información a pasar a través de la transferencia de control, en este caso consiste en el byte que representa el registro a leer en el demodulador.</p> <ul style="list-style-type: none"> • wIndex <p>Recibe el número de endpoint, en este caso 0.</p> <ul style="list-style-type: none"> • data <p>Recibe un apuntador al buffer donde se guarda el resultado de la operación I2C, en este caso el buffer es de dos</p>

		<p>dimensiones en la primera dimensión se guarda el resultado de la operación; 0 exitosa, otro valor en caso contrario, y en la segunda dimensión el dato leído.</p> <ul style="list-style-type: none"> • wLength <p>La longitud del buffer en este caso 2.</p> <ul style="list-style-type: none"> • timeout <p>Tiempo de espera a que se realice la operación, 20 ms.</p>
Escritura	TC90527_I2cWrite():	<ul style="list-style-type: none"> • libusb_device_handle <p>Esta función recibe como parámetro la dirección del registro a escribir en el demodulador y la data a ser escrita, retorna 0 o otro valor en caso contrario.</p> <p>Recibe una variable Libusb para manejar el dispositivo USB.</p> <ul style="list-style-type: none"> • bmRequestType <p>Es el tipo de operación que se desea realizar, recibe el valor de (0x40)h para operación de escritura indicar que es una operación de escritura.</p> <ul style="list-style-type: none"> • bRequest <p>Este byte es usado para indicar que función de transferencia I2C se ejecutará en el firmware, el valor para escritura hacia el demodulador está definido por (0xA2)h.</p> <ul style="list-style-type: none"> • wValue <p>Es la información a pasar a través de la transferencia de control, en este caso consiste en la combinación de los bytes de dato a escribir y dirección de registro a escribir para formar un dato de dos bytes, esta operación la realiza la función funcion_combinar_bytes()</p>

	<p>antes de que la información sea mandada al dispositivo.</p> <ul style="list-style-type: none"> • wIndex <p>Recibe el número de endpoint, en este caso 0.</p> <ul style="list-style-type: none"> • data <p>Recibe un apuntador al buffer donde se guarda el resultado de la operación I2C, 0 exitosa, otro valor distinto de 0 si no lo fue.</p> <ul style="list-style-type: none"> • wLength <p>La longitud del buffer en este caso 1.</p> <ul style="list-style-type: none"> • timeout <p>Tiempo de espera a que se realice la operación, 20 ms.</p>
--	--

A continuación en la **Tabla 9.1.4** se describen las funciones de control USB pertenecientes al sintonizador.

Tabla 9.1. 4: Funciones desarrolladas en base de la API Libusb para las transferencias USB de tipo control pertenecientes al sintonizador.

Tipo de operación	Función USB del Sintonizador	Parámetros libusb_control_transfer()
Lectura	<p>STV4100_I2C_Read():</p> <p>Esta función recibe como parámetro la dirección del registro a leer en el</p>	<ul style="list-style-type: none"> • libusb_device_handle <p>Recibe una variable Libusb para manejar el dispositivo USB.</p> <ul style="list-style-type: none"> • bmRequestType

	<p>sintonizador y un apuntador donde se retorna el dato leído.</p>	<p>Es el tipo de operación que se desea realizar, recibe el valor de (0xC0)h para indicar que es una operación de lectura.</p> <ul style="list-style-type: none"> • bRequest <p>Este byte es usado para indicar que función de transferencia I2C se ejecutará en el firmware, el valor para lectura desde el sintonizador está definido por (0xA5)h.</p> <ul style="list-style-type: none"> • wValue <p>Es la información a pasar a través de la transferencia de control, en este caso consiste en el byte que representa el registro a leer en el sintonizador.</p> <ul style="list-style-type: none"> • wIndex <p>Recibe el número de endpoint, en este caso 0.</p> <ul style="list-style-type: none"> • data <p>Recibe un apuntador al buffer donde se guarda el resultado de la operación I2C, en este caso el buffer es de dos dimensiones en la primera dimensión se guarda el resultado de la operación; 0 exitosa, otro valor en caso contrario, y en la segunda dimensión el dato leído.</p> <ul style="list-style-type: none"> • wLength <p>La longitud del buffer en este caso 2.</p> <ul style="list-style-type: none"> • timeout
--	--	--

		Tiempo de espera a que se realice la operación, 20 ms.
Escritura	<p>STV4100_I2C_Write():</p> <p>Esta función recibe como parámetro la dirección del registro a escribir en el sintonizador y la data a ser escrita, retorna 0 en caso de la operación de escritura haya sido exitosa u otro valor en caso contrario.</p>	<ul style="list-style-type: none"> • libusb_device_handle <p>Recibe una variable Libusb para manejar el dispositivo USB.</p> <ul style="list-style-type: none"> • bmRequestType <p>Es el tipo de operación que se desea realizar, recibe el valor de (0x40)h para indicar que es una operación de escritura.</p> <ul style="list-style-type: none"> • bRequest <p>Este byte es usado para indicar que función de transferencia I2C se ejecutará en el firmware, el valor para escritura hacia el sintonizador está definido por (0xA4)h.</p> <ul style="list-style-type: none"> • wValue <p>Es la información a pasar a través de la transferencia de control, en este caso consiste en la combinación de los bytes de dato a escribir y dirección de registro a escribir para formar un dato de dos bytes, esta operación la realiza la función funcion_combinar_bytes() antes de que la información sea mandada al dispositivo.</p> <ul style="list-style-type: none"> • wIndex <p>Recibe el número de endpoint, en este caso 0.</p> <ul style="list-style-type: none"> • data

		<p>Recibe un apuntador al buffer donde se guarda el resultado de la operación I2C, 0 exitosa, otro valor distinto de 0 si no lo fue.</p> <ul style="list-style-type: none">• wLength <p>La longitud del buffer en este caso 1.</p> <ul style="list-style-type: none">• timeout <p>Tiempo de espera a que se realice la operación, 20 ms.</p>
--	--	--

9.1.2.1 Rutinas de Transferencias de Interrupción

Las transferencias de interrupción son las encargadas de obtener el flujo de audio y video en paquetes BTS, para ello se utilizó la interfaz asincrónica de la API Libusb para este tipo de transferencias, de este modo se puede obtener un flujo constante de video evitando perdidas de paquetes de información entre llamado y llamado de una transferencia de interrupción.

La técnica consiste en colocar una cierta cantidad de transferencias en cola, en este caso se utilizaron 64 transferencias de interrupción, con buffers de información de tamaño considerable, de 240Kb, donde en cada transferencias se espera a llenar en totalidad el buffer con paquetes de datos BTS. [12]

La API Libusb es un puente entre el espacio de usuario y el espacio del núcleo Linux, los procesos en el espacio del núcleo ocurren con mayor rapidez que en el espacio de usuario, es allí en el espacio del núcleo donde el buffer de datos es llenado, entonces para buffers más grandes significa menos viajes hacia el espacio de usuario para realizar una nueva transferencia, una vez que una transferencia finaliza otra transferencia está

esperando a ser realizada y la transferencia finalizada es puesta al final de la cola de transferencias para ser llamada nuevamente.

Las funciones encargadas de la obtención de los paquetes BTS se muestran en la **Tabla 9.1.5**.

Tabla 9.1.5: Funciones desarrolladas en base de la API Libusb para las transferencias USB de tipo interrupción.

procedimiento_ciclo_repcion_BTS() 	<p>Descripción: Esta función se ejecuta en un hilo secundario del programa. Si el dispositivo se encuentra conectado, se encarga de crear 64 transferencias de interrupción haciendo uso de la función funcion_crear_transferencia_usb() y luego ejecutando cada una de las transferencias con la función libusb_submit_transfer(). Al finalizar la ejecución de las transferencias entra en un ciclo de manejo de eventos de las transferencias que se están realizando y usando la función funcion_leer_estado_hotplug() chequea el estado de conexión en caliente, si el dispositivo es desconectado de manera abrupta en un proceso de recepción de datos se liberan las transferencias y el hilo terminar.</p> <p>Esta función retorna 0 para indicar el fin del proceso ejecutado en el hilo.</p>
	<p>Variables:</p> <ul style="list-style-type: none"> • transferencia_usb <p>Variable Libusb para manejar la transferencia.</p>

	<p>Funciones Libusb:</p> <ul style="list-style-type: none"> • libusb_submit_transfer() <p>transferencia_usb, variable Libusb para manejar la transferencia.</p> <ul style="list-style-type: none"> • libusb_handle_events() <p>contexto_usb, representa una sesión de la librería Libusb.</p> <ul style="list-style-type: none"> • libusb_free_transfer() <p>transferencia_usb, variable Libusb para manejar la transferencia.</p>
funcion_crear_transferencia_usb()	<p>Descripción: Esta función se encarga de alojar, crear y devolver una variable Libusb del tipo libusb_transfer. Se usa la función libusb_alloc_transfer() para alojar la transferencia, para la especificación de los datos de la transferencia se hace uso de la función libusb_fill_interrupt_transfer().</p> <p>Variables:</p> <ul style="list-style-type: none"> • transferencia_usb_2 <p>Variable Libusb para manejar la transferencia.</p>

	<p>Funciones Libusb:</p> <ul style="list-style-type: none"> • <code>libusb_alloc_transfer()</code> <p>0, para alojar la transferencia recibe como parámetro 0 ya que es una transferencia de tipo interrupción.</p> <p>transferencia_usb_2, variable Libusb para manejar la transferencia.</p> <ul style="list-style-type: none"> • <code>libusb_fill_interrupt_transfer()</code> <p>transferencia_usb_2, variable Libusb para manejar la transferencia.</p> <p>manejador_dispositivo_usb, variable Libusb para el manejo del dispositivo USB.</p> <p>0x82, dirección del Endpoint 2.</p> <p>MPEG2_TS, buffer de 240 Kb donde se reciben los paquetes BTS.</p> <p>tamanio_buffer, tamaño del buffer en bytes, en este caso 245760 bytes.</p> <p>procedimiento_de_llamada_leer_transferencia_usb, función de llamada cuando finaliza una transferencia.</p> <p>NULL, ningún dato es pasado a la función de llamada.</p> <p>20000, tiempo de espera para realizar la transferencia en milisegundos.</p>
--	--

procedimiento_de_llamada_leer_transferencia_usb()	<p>Descripción: Esta función se ejecuta de manera asincrónica cada vez que una transferencia finaliza, lee si el estado de la transferencia fue exitosa o no, si fue exitosa se procede a copiar la información del buffer de 240Kb a aun buffer de mayor tamaño utilizado por VLC para la reproducción y luego esa transferencia es puesta en cola nuevamente.</p>
	<p>Funciones Libusb:</p> <ul style="list-style-type: none">• libusb_submit_transfer() <p>transferencia_usb_2, variable Libusb para manejar la transferencia.</p>

Para la reproducción de audio y video de los datos obtenidos en las transferencias que se encuentran en el buffer de 240Kb, estos datos deben ser copiados en un buffer de mayor tamaño para que el reproductor VLC cuente con la cantidad necesarias de datos para la reproducción, por lo cual se creó un buffer de 4,1 Mb. El procedimiento de copiado usa apuntadores para ir copiando de manera sucesiva los bloques de información que contienen paquetes BTS al buffer utilizado por VLC.

El buffer de 4,1 Mb se define como 20 bloques de posiciones de memoria con una longitud de 240Kb para cada bloque y se representa a continuación en la **Tabla 9.1.6.**

Tabla 9.1.6: Distribución de las posiciones de memoria pertenecientes al buffer de 4,1 Mb.

0	245760	491520	737280	983040
1228800	1474560	1720320	1966080	2211840
2457600	2703360	2949120	3194880	3440640
3686400	3932160	4177920	4423680	4669440

En la primera transferencia recibida el apuntador apunta a la posición cero del buffer utilizado por VLC y copia los 240Kb de datos obtenidos en la transferencia de interrupción, luego el apuntador se incrementa en 245760 posiciones, 240Kb, y realiza el proceso de copia nuevamente en la siguiente posición hasta alcanzar finalmente el bloque número 20, luego el apuntador se inicializa nuevamente en la posición 0 y se realiza el proceso nuevamente.

10.1 La API Libvlc

Libvlc es una librería que puede integrarse en una aplicación para dotarla de capacidades multimedia, debido a que VLC se basa en Libvlc es posible acceder a la mayoría de las características que VLC Media Player posee, las rutinas desarrolladas hacen uso de la versión 3.0.0 de VLC.

En la **Tabla 10.1.1** se listan las funciones usadas de la librería Libvlc conjuntamente con una descripción.

Tabla 10.1.1: Funciones de la API Libvlc. [13]

libvlc_new()	<p>Descripción: Crea una sesión de la librería.</p>
	<p>Parámetros:</p> <ul style="list-style-type: none"> • argc <p>Argumento tipo entero que denota la cantidad de apuntadores pasados a la función.</p> <ul style="list-style-type: none"> • argv <p>Arreglo de apuntadores pasados a la función.</p>
	<p>Retorna:</p> <ul style="list-style-type: none"> • libvlc_instance_t <p>Variable que representa una instancia de la librería.</p>
libvlc_release()	<p>Descripción: Cierra una sesión de la librería.</p>
	<p>Parámetros:</p> <ul style="list-style-type: none"> • libvlc_instance_t <p>Variable que representa una instancia de la librería.</p>
	<p>Retorna: Nada</p>
libvlc_media_player_pause()	<p>Descripción: Esta función coloca en estado la reproducción del reproductor.</p>
	<p>Parámetros:</p>
	<ul style="list-style-type: none"> • libvlc_media_player_t
	<p>Estructura que representa a un</p>

	<p>reproductor de la librería.</p> <p>Retorna: Nada.</p>
libvlc_media_player_stop()	<p>Descripción: Esta función detiene la reproducción del reproductor.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libvlc_media_player_t <p>Estructura que representa a un reproductor de la librería.</p> <p>Retorna: Nada.</p>
libvlc_media_player_is_playing()	<p>Descripción: Esta función retorna el estado de reproducción del reproductor.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libvlc_media_player_t <p>Estructura que representa a un reproductor de la librería.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • 1 <p>Si el reproductor esta reproduciendo.</p> <ul style="list-style-type: none"> • 0 <p>Si el reproductor no está reproduciendo.</p>
libvlc_audio_set_volume()	<p>Descripción: Esta función permite fijar el nivel de dB del audio del reproductor.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libvlc_media_player_t <p>Estructura que representa a un</p>

	<p>reproductor de la librería.</p> <ul style="list-style-type: none"> • entero <p>Representa el nivel de dB del audio su valor debe estar entre 0 y 100.</p> <p>Retorna: Nada.</p>
libvlc_media_player_set_media()	<p>Descripción: Esta función permite seleccionar un programa en específico dentro de archivos multimedia que contengan varios programas.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libvlc_media_t <p>Variable que representa al contenido multimedia.</p> <ul style="list-style-type: none"> • psz_options <p>Es una cadena de caracteres que contiene el ID del programa a reproducir.</p> <p>Retorna: Nada.</p>
libvlc_media_player_set_xwindow()	<p>Descripción: Esta función se encarga de embeber una interfaz de video en una interfaz gráfica.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libvlc_media_player_t <p>Estructura que representa a un reproductor de la librería.</p> <ul style="list-style-type: none"> • entero <p>Representa el ID de la ventana o widget de la interfaz gráfica donde incrustar el reproductor.</p>

	<p>Retorna: Nada.</p>
libvlc_media_new_callbacks()	<p>Descripción: Esta función crea un medio de contenido multimedia con devoluciones de funciones llamada personalizadas que leen porciones de datos multimedia.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • libvlc_instance_t <p>Variabile que representa una instancia de la librería.</p> <ul style="list-style-type: none"> • libvlc_media_open_cb <p>Apuntador a la función de llamada.</p> <ul style="list-style-type: none"> • libvlc_media_read_cb <p>Apuntador a la función de llamada.</p> <ul style="list-style-type: none"> • libvlc_media_seek_cb <p>Apuntador a la función de llamada.</p> <ul style="list-style-type: none"> • libvlc_media_close_cb <p>Apuntador a la función de llamada.</p> <ul style="list-style-type: none"> • data <p>Apuntador de datos a pasar a la función.</p> <p>Retorna:</p> <ul style="list-style-type: none"> • libvlc_media_t <p>Variabile que representa al contenido multimedia.</p>

libvlc_media_open_cb()	<p>Descripción: Esta función se encarga de abrir el medio de datos de entrada. Es una función de llamada de libvlc_media_new_callbacks().</p>
	<p>Parámetros:</p> <ul style="list-style-type: none"> • opaque <p>Es el apuntador pasado a la función libvlc_media_new_callbacks().</p> <ul style="list-style-type: none"> • datap <p>Define el espacio de almacenamiento para un puntero de datos privado.</p> <ul style="list-style-type: none"> • sizep <p>Longitud del flujo de datos.</p>
	<p>Retorna:</p> <ul style="list-style-type: none"> • 0 <p>Si la operación fue exitosa.</p> <ul style="list-style-type: none"> • Otro valor <p>En caso de error.</p>
libvlc_media_read_cb()	<p>Descripción: Esta función se encarga de leer el medio de datos de entrada. Es una función de llamada de libvlc_media_new_callbacks().</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • opaque <p>Es el apuntador pasado a la función libvlc_media_new_callbacks().</p>

	<ul style="list-style-type: none"> • <code>buf</code> Dirección de inicio del buffer para leer datos. • <code>len</code> Longitud del buffer. <p>Retorna:</p> <ul style="list-style-type: none"> • entero Número de bytes leídos. • 0 Si llega al final del flujo de datos. • -1 Si ocurre un error.
<code>libvlc_media_seek_cb()</code>	<p>Descripción: Esta función se encarga de buscar una posición en el medio de datos de entrada. Es una función de llamada de <code>libvlc_media_new_callbacks()</code>.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • <code>opaque</code> Es el apuntador pasado a la función <code>libvlc_media_new_callbacks()</code>. • <code>offset</code> Desplazamiento de bytes absolutos para buscar. <p>Retorna:</p>

	<ul style="list-style-type: none"> • 0 Si la operación fue exitosa. • -1 En caso de error.
libvlc_media_close_cb()	<p>Descripción: Esta función se encarga de cerrar el medio de datos de entrada. Es una función de llamada de <code>libvlc_media_new_callbacks()</code>.</p> <p>Parámetros:</p> <ul style="list-style-type: none"> • opaque <p>Es el apuntador pasado a la función <code>libvlc_media_new_callbacks()</code>.</p> <p>Retorna: Nada</p>

10.1.1 Rutinas Multimedia

Para la obtención del contenido multimedia se creó una función que se encarga de obtener la información multimedia del buffer de paquetes BTS que posee 4,1 Mb de longitud, para ello la función se basa en llamadas asincrónicas que obtienen tramos de información del buffer y luego estos datos son pasados como información multimedia al reproductor.

Los tramos de información están divididos en paquetes de 240Kb para un total de 20 tramos, dando un total de 4,1 Mb el tamaño total del buffer, en la primera lectura un apuntador apunta a la posición inicial del buffer y se copian los primeros 240Kb, que equivalen a 245760 posiciones, a la variable de tipo **`libvlc_media_t`**, luego el apuntador se incrementa 245760 posiciones y procede a realizar la misma operación con el siguiente bloque de datos hasta realizar la última operación correspondiente al bloque

20, después de esta operación el apuntador se inicializa nuevamente en la posición inicial y el proceso se repite nuevamente.

Hay que hacer notar que el buffer de 4,1 Mb es un recurso compartido con la función que se ejecuta en otro hilo del programa encargada de llenarlo con el flujo de paquetes BTS procedente de la transferencias USB de tipo interrupción, por lo cual mientras se obtiene el contenido multimedia del buffer este a su vez es refrescado con nueva información obtenida a través de transferencias USB desde el dispositivo de TDA.

A continuación en la **Tabla 10.1.2** se especifican las funciones desarrolladas para el control multimedia.

Tabla 10.1.2: Funciones desarrolladas en base a la API Libvlc.

procedimiento_media_callbacks()	<p>Descripción: Esta función se encarga de obtener la información multimedia del buffer de paquetes BTS con la función libvlc_media_new_callbacks(), añadir al reproductor el contenido multimedia a producir usando la función libvlc_media_player_set_media(), seleccionar el programa multimedia a reproducir utilizando la función libvlc_media_add_option() y finalmente comenzar la reproducción con la función libvlc_media_player_play().</p> <p>Variables:</p> <ul style="list-style-type: none">• media <p>Variable que representa al contenido multimedia.</p>
---------------------------------	---

	<p>Funciones Libvlc:</p> <ul style="list-style-type: none"> • libvlc_media_new_callbacks() <p>instancia_vlc, es una variable del tipo libvlc_instance_t, que representa una sesión de la librería.</p> <p>abrir_media, es el apuntador a la función de llamada <code>abrir_media()</code>.</p> <p>leer_media, es el apuntador a la función de llamada <code>leer_media()</code>.</p> <p>posicionar_media, es el apuntador a la función de llamada <code>posicionar_media()</code>.</p> <p>NULL, es el apuntador a la función llamada cerrar datos, nulo no se utilizó.</p> <p>apuntador_al_buffer_repcion_BTS, es un apuntador al buffer de 4,1 Mb que contiene los paquetes BTS.</p> <ul style="list-style-type: none"> • libvlc_media_add_option() <p>media, es una variable del tipo libvlc_media_t, que representa al contenido multimedia.</p> <p>funcion_leer_id(), función que retorna una cadena de caracteres con el ID del programa.</p> <ul style="list-style-type: none"> • libvlc_media_player_set_media() <p>reproductor, es una variable del</p>
--	--

	<p>tipo libvlc_media_player_t, que representa a un reproductor.</p> <p>Media, es una variable del tipo libvlc_media_t, que representa al contenido multimedia.</p> <ul style="list-style-type: none"> • libvlc_media_player_play() <p>reproductor, es una variable del tipo libvlc_media_player_t, que representa a un reproductor.</p>
abrir_media()	<p>Descripción: Esta función abre el recurso de donde se extrae el contenido multimedia, el buffer de BTS de 4,1 Mb. Esta es una función de llamada de libvlc_media_new_callbacks().</p> <p>Funciones Libusb:</p> <ul style="list-style-type: none"> • libvlc_media_open_cb() <p>opaque, es el apuntador pasado a la función libvlc_media_new_callbacks() del buffer de datos a leer.</p> <p>datap, dirección de datos interna de la función.</p> <p>sizep, longitud del buffer.</p> <p>Retorna 0 como operación exitosa.</p>
leer_media()	<p>Descripción: Esta función lee la información multimedia del buffer de BTS de 4,1 Mb en porciones de 240Kb en cada llamada. Esta es una función de llamada de libvlc_media_new_callbacks().</p>

	<p>Funciones Libusb:</p> <ul style="list-style-type: none"> • <code>libvlc_media_read_cb()</code> <p>opaque, es el apuntador pasado a la función <code>libvlc_media_new_callbacks()</code> del buffer de datos a leer.</p> <p>buffer_de_llamada, es el buffer interno de la función que guarda los datos leídos.</p> <p>longitud_de_llamada, representa la cantidad de datos leídos.</p> <p>Retorna la cantidad de datos leídos.</p>
posicionar_media()	<p>Descripción: Esta función se posiciona en el recurso multimedia para determinar la posición de recolección de datos. Esta es una función de llamada de <code>libvlc_media_new_callbacks()</code>.</p> <p>Esta función no es utilizada ya que el apuntador de bloques al buffer de BTS en la función <code>leer_media()</code> realiza la función de offset, pero debe ser implementada para que las llamadas de obtención de datos multimedia de manera asincrónica funcionen.</p>
	<p>Funciones Libusb:</p> <ul style="list-style-type: none"> • <code>libvlc_media_seek_cb()</code>

opaque, es el apuntador pasado a la función **libvlc_media_new_callbacks()** del buffer de datos a leer.

offset, ultima posición de datos leídos.

Retorna 0.

11.1 Interfaz Gráfica

Existen muchas soluciones para la realización y manejo de interfaces gráficas en los sistemas operativos basados en GNU/Linux, entre ellas se tomaron en cuenta a la hora de realizar el proyecto QT, GTK y WxWidgets por el soporte y códigos de ejemplo de la API Libvlc sobre estas interfaces, estos dados por la VideoLAN Organization empresa sin fines de lucro responsable por el desarrollo y mantenimiento del reproductor VLC Media Player, resultando seleccionada la utilidad GTK debido a su facilidad a la hora de embeber el reproductor en la interfaz gráfica y la sencillez del código para implementar esta interfaz, además de que GTK es la interfaz con la cual está escrito el escrito GNOME para sistemas Linux y muchas aplicaciones hoy en día usan este tipo de interfaz lo cual augura un soporte a largo tiempo de esta interfaz, la documentación de GTK es extensa y existen muchas comunidades a las cuales pedir soporte sin costo alguno, QT fue dejado de lado en parte por su complejidad y la obsolescencia de la versión de QT usada por VLC, aunque esta interfaz gráfica posee gran soporte y documentación fue descartada, también fue descartada la interfaz WxWidgets por la falta de soporte, documentación y la complejidad del código usado para generar los mismo resultados que la interfaz GTK.

11.1.1 Empaquetamiento gráfico.

La aplicación desarrollada posee una sola ventana, la distribución gráfica se realizó usando una rejilla que es embebida a la ventana de la aplicación, la rejilla es una utilidad que ofrece GTK para distribuir los diferentes layaout, un layout es una cuadrícula imaginaria que divide en espacios o campos la página que se diseña para facilitar la distribución de elementos como textos o gráficos, GTK define diferentes tipos de layaout como contenedores verticales y horizontales, los cuales pueden contener

widgets, los widgets dan fácil acceso a funciones frecuentemente usadas y proveen de información visual como botones, imágenes, etiquetas, ventanas, etc.

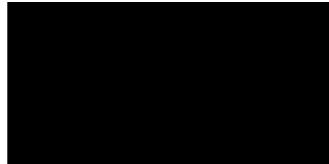
En la **Figura 11.1.1** se observa la distribución de la rejilla, donde las coordenadas de distribución se muestran como pares de puntos (x, y).

0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9	0,10	0,11
1,0	1,1	1,2	1,3	1,4	1,5	1,6	1,7	1,8	1,9	1,10	1,11

Figura 11.1. 1: Distribución de la rejilla, coordenadas de distribución pares de puntos (x, y).

A continuación en **Tabla 11.1.1** se muestra el empaquetamiento de los distintos widgets que componen la interfaz gráfica.

Tabla 11.1.1: Empaquetamiento de widgets.

Widgets	Empaquetamiento
	La ventana del reproductor multimedia está representada por el widget mostrado en la Figura 11.1.2 , este widget se encarga de dibujar un área en la aplicación donde se muestra el contenido del reproductor, el widget está insertado directamente a la rejilla.
	Los widgets que realizan las acciones de reproducción, pausar, detener, control de volumen del contenido multimedia y además modo de reproducción a pantalla

	<p>completa, se encuentran empaquetados en el layout Botones de Control Multimedia que contiene a sus hijos en un formato de caja horizontal, como se muestra en la Figura 11.1.3.</p>
<p>Conexión USB:  Señal: </p> <p>Figura 11.1.4: Botones Indicadores.</p>	<p>Los widgets que tienen como función entregar información visual acerca del estado de conexión del dispositivo USB así como la intensidad de la señal recibida por el dispositivo están agrupados en un layout de nombre Botones Indicadores, que contiene a sus hijos en un formato de caja horizontal, como se muestra en la Figura 11.1.4.</p>
 <p>Figura 11.1.5: Botones de Programación.</p>	<p>Los widgets de botones que representan a los distintos programas de la TDA venezolana se encuentran agrupados en un layout que contiene a sus hijos en un formato de caja vertical caja vertical el cual se muestra en la Figura 11.1.5.</p>



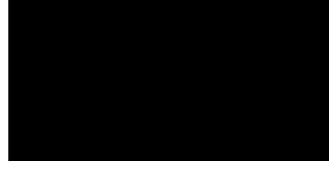
Figura 11.1.6:
Ventana de
Desplazamiento.

El widget de una ventana con una barra de desplazamiento vertical contiene al layout de Caja Vertical Botones de Programación como se muestra en la **Figura 11.1.6**.

El modo de empaquetamiento de los widgets y layouts en la rejilla se realizó de la siguiente manera:

En la **Tabla 11.1.2** se muestran las coordenadas ocupadas por los elementos donde la coordenada X de la rejilla posee un valor fijo de 0.

Tabla 11.1.2: Empaqueamiento en la rejilla donde el valor de la coordenada X es 0.

Coordenadas rejilla X/Y	0..9	10..11
0	 Figura 11.1.2: Ventana del Reproductor.	 Figura 11.1.6: Ventana de Desplazamiento.

En la **Tabla 11.1.3** se muestran las coordenadas ocupadas por los elementos donde la coordenada X de la rejilla posee un valor fijo de 1.

Tabla 11.1.3: Empaquetamiento en la rejilla donde el valor de la coordenada X es 1.

Coordenadas rejilla X/Y	0..1	9..10
1	<p>Figura 11.1.3: Botones de Control Multimedia.</p>	<p>Figura 11.1.4: Botones Indicadores.</p>

Las acciones de empaquetamiento realizado dan como resultado la interfaz gráfica de la **Figura 11.1.7.**

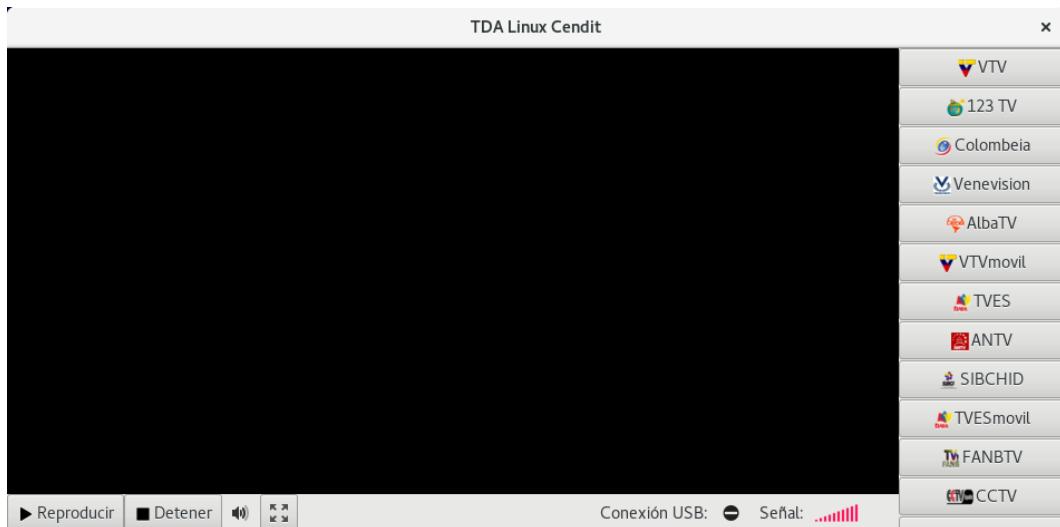


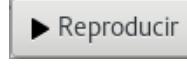
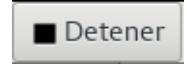
Figura 11.1.7: Interfaz gráfica de la aplicación.

11.1.1 Funciones de Enlace entorno Gráfico.

Las rutinas de enlace del entorno gráfico son las funciones encargados de hacer la conexión entre el front-end y back-end de la aplicación, el front-end es la parte del software que interactúa con el usuario, en este caso la interfaz gráfica desarrollada en GTK, y el back-end es la parte que procesa la entrada desde el front-end, son las funciones encargadas de los procesos USB y la reproducción de video. La separación del sistema en front-ends y back-ends es un tipo de abstracción que ayuda a mantener las diferentes partes del sistema separadas.

En la Tabla se muestran los widgets, el nombre de los métodos que actúan de conexión entre el front-end y el back-end conjuntamente con una descripción de los procesos que ejecutan.

Tabla 11.1.4: Widgets, descripción y funciones de enlace.

 <p>Figura 11.1.8: Widget del botón reproducir y pausar, estado reproducir.</p>  <p>Figura 11.1.9: Widget del botón reproducir y pausar, estado pausar.</p>	<p>Descripción: Esto widget se encarga de reproducir o pausar el contenido multimedia, si ningún programa ha sido seleccionado anteriormente se sintonizara y reproducirá por defecto el Programa VTV al momento de hacer click sobre el botón reproducir.</p> <p>Funciones:</p> <ul style="list-style-type: none"> • procedimiento_de_llamada_reproducir_pausar() • procedimiento_de_llamada_hilo_recepcion()
 <p>Figura 11.2.1: Widget del botón detener.</p>	<p>Descripción: Este widget al hacer click sobre él detiene el contenido multimedia que se esta reproduciendo, sino hay cometido en reproducción no causa efecto alguno.</p> <p>Funciones:</p> <ul style="list-style-type: none"> • procedimiento_de_llamada_detener()

	<p>Descripción: Este widget permite controlar el nivel del audio en dB del reproductor.</p> <p>Funciones:</p> <ul style="list-style-type: none"> • procedimiento_de_llamada_volumen()
	<p>Descripción: Este widget al hacer click sobre él permite colocar en modo pantalla completa la ventana del reproductor.</p> <p>Funciones:</p> <ul style="list-style-type: none"> • procedimiento_de_llamada_maxima_pantalla()
Widgets del Layout Botones Indicadores	
<p>Descripción: Los widgets indicadores de estado muestran cuando el dispositivo USB está conectado como en la Figura 11.2.5 o desconectado como se muestra en la Figura 11.2.4, los otros widgets muestran la intensidad de la señal recibida entre una escala de 0 y 10 siendo 10 la máxima intensidad de señal.</p>	
<p>Funciones:</p> <ul style="list-style-type: none"> • procedimiento_imagenes_caja_horizontal_indicadores() 	
<p>Widgets:</p>	
 Figura 11.2.4: Widget estado conexión USB, desconectado.	 Figura 11.2.10: Widget estado conexión USB, conectado.

Señal:

Figura 11.2.5:
Widget intensidad
de la señal, nivel
de intensidad 0.

Señal:

Figura 11.2.6:
Widget
intensidad de la
señal, nivel de
intensidad 1.

Señal:

Figura 11.2.7:
Widget
intensidad de la
señal, nivel de
intensidad 2.

Señal:

Figura 11.2.8:
Widget
intensidad de la
señal, nivel de
intensidad 3.

Señal:

Figura 11.2.9:
Widget
intensidad de la
señal, nivel de
intensidad 4.

Señal:

Figura 11.2.11:
Widget
intensidad de la
señal, nivel de
intensidad 5.

Señal:

Figura 11.2.12:
Widget
intensidad de la
señal, nivel de
intensidad 6.

Señal:

Figura 11.2.13:
Widget
intensidad de la
señal, nivel de
intensidad 7.

Señal:

Figura 11.2.14:
Widget intensidad
de la señal, nivel
de intensidad 8.

Señal:

Figura 11.2.15:
Widget
intensidad de la
señal, nivel de
intensidad 9.

Señal:

Figura 11.2.16:
Widget

Se añadieron otras características que interactúa con el usuario con la creación de una función de nombre **procedimiento_de_llamada_tecla_presionada()** la cual habilita un grupo de teclas calientes para un cierto número de acciones de control de la interfaz de usuario. A continuación se definen las teclas calientes programadas y su función.

- TECLA_MÁS: Incrementar volumen.
- TECLA_MENOS: Decremento de volumen.
- TECLA_A: Reproducir o pausar contenido multimedia.
- TECLA_a: Reproducir o pausar contenido multimedia.
- TECLA_D: Detener contenido multimedia.
- TECLA_d: Detener contenido multimedia.
- TECLA_S: Entrar en modo pantalla completa.
- TECLA_s: Entrar en modo pantalla completa.
- TECLA_ESCAPE: Salir del modo pantalla completa.

12.1 Instalador del Programa

El sistema operativo para el cual se crearon rutinas de instalación y desinstalación del software desarrollado es el sistema operativo **GNU/Linux Fedora 24** y por lo tanto es en el cual se realizaron las pruebas de funcionamiento de las rutinas.

El programa desarrollado para su compilación depende de ciertas librerías y de la definición de algunas banderas de compilación que enlacen el proceso de generar el archivo binario con las librerías necesarias.

Las distribuciones que usan el núcleo GNU/Linux cuenta con un directorio ubicado en la raíz de archivos **/opt** cuya existencia está definida para la instalación de paquetes de software de aplicación complementaria, en este directorio es donde se ubicará el archivo binario ejecutable de la aplicación y la imagen del ícono de acceso directo.

La estructura de archivos del instalador se observa en la **Figura 12.1.1** ejecutando el comando **tree** en la terminal:

```
[cendit@cendit-machine ~]$ tree instalador_fedora
instalador_fedora
├── 40_usb.rules
├── imagenes
│   ├── canales
│   │   ├── alba_imagen.h
│   │   ├── antv_imagen.h
│   │   ├── avilatv_imagen.h
│   │   ├── cctv_imagen.h
│   │   ├── colombeia_imagen.h
│   │   ├── conciencia_imagen.h
│   │   ├── fanbtv_imagen.h
│   │   ├── meridiano_imagen.h
│   │   ├── pdvsatv_imagen.h
│   │   ├── russiatoday_imagen.h
│   │   ├── sibci_imagen.h
│   │   ├── telesur_imagen.h
│   │   ├── televen_imagen.h
│   │   ├── tv123_imagen.h
│   │   ├── tves_imagen.h
│   │   ├── venevision_imagen.h
│   │   ├── vive_imagen.h
│   │   └── vtv_imagen.h
│   └── iconos
│       ├── detener_imagen.h
│       ├── icon_imagen.h
│       ├── pantalla_completa_imagen.h
│       ├── pausar_imagen.h
│       ├── reproducir_imagen.h
│       ├── senial_00_imagen.h
│       ├── senial_01_imagen.h
│       ├── senial_02_imagen.h
│       ├── senial_03_imagen.h
│       ├── senial_04_imagen.h
│       ├── senial_05_imagen.h
│       ├── senial_06_imagen.h
│       ├── senial_07_imagen.h
│       ├── senial_08_imagen.h
│       ├── senial_09_imagen.h
│       ├── senial_10_imagen.h
│       └── usb_conectado_imagen.h
│           └── usb_desconectado_imagen.h
└── include
    ├── calls.h
    ├── initialize.h
    ├── samsung.h
    ├── thread.h
    └── usb.h
    ├── install
    ├── logo.png
    ├── main.cpp
    └── uninstall
    ├── vlc-3.0.0-0.26snap.20170601git.fc24.x86_64.rpm
    ├── vlc-core-3.0.0-0.26snap.20170601git.fc24.x86_64.rpm
    └── vlc-devel-3.0.0-0.26snap.20170601git.fc24.x86_64.rpm

```

4 directories, 49 files

Figura 12.1.1: Estructura de archivos del instalador.

El instalador se encuentra dividido en dos archivos un archivo que ejecuta la instalación con el nombre **install** y otro que ejecuta la desinstalación llamado **unisntall**.

La carpeta **imagenes** contiene las imágenes usadas por los widgets en formato binario, las cuales se integran al programa como cabeceras de C.

El archivo **40_usb.rules**, es un archivo que añade reglas para los dispositivos USB, en este caso solo para el dispositivo USB de la compañía Cypress, las cuales permiten procesos de lectura y escritura sin necesidad de ser un usuario root del sistema.

El archivo **main.cpp**, es el archivo principal del programa que enlaza a todos los demás archivo, el directorio **include** contiene los archivos cabeceras con las funciones desarrolladas para el programa.

El archivo **logo.png** es el icono utilizado por el lanzador de la aplicación.

Los paquetes **vlc** son los paquetes de instalación del software multimedia VLC.

12.1.1 Paquetes y software requerido.

Los paquetes de software utilizados para el desarrollo y compilación del software fueron:

- libusb-1.0-0, versión: 2:1.0.20-1;
- vlc, versión: 3.0.0.26; vlc-core, versión: 3.0.0.26.
- libgtk-3-0, versión: 3.18.9.
- g++, versión: 4:5.3.1.
- libusb-1.0-0-devel, versión: 2:1.0.20-1.
- libvlc-devel, versión: 3.0.0.
- libgtk-3-devel, versión: 3.18.9.

12.1.2 Proceso de compilación

El proceso de compilación se realiza con el compilador g++, mediante la siguiente línea de comando:

```
g++ main.cpp -std=c++11 -lusb-1.0 -lvlc -pthread -rdynamic  
-lx11 -s `pkg-config --cflags gtk+-3.0` `pkg-config --libs gtk+-3.0` -o  
tda_executable -w
```

El comando especifica el compilador a usar **g++**, el nombre del archivo a compilar en este caso el **main.cpp**, que es archivo principal del programa, el compilador se encarga de enlazar y compilar los otros archivos que estén enlazados dentro del archivo principal para generar el ejecutable binario, las banderas de compilación. Definición de las banderas:

- **-std=c++11**, esta bandera establece la versión del lenguaje c++ 2011.
- **-lusb-1.0**, esta bandera enlaza la librería LIBUSB.
- **-lvlc**, esta bandera enlaza la librería LIBVLC.
- **-pthread**, esta bandera enlaza la librería para poder usar hilos.
- **-rdynamic**, esta bandera permite obtener el rastreo de pila completo con los nombres de función.
- **-lx11**, esta bandera enlaza con la librería Xlib que es usada por VLC.
- **-s**, esta bandera elimina toda la tabla de símbolos y la información de reubicación del archivo ejecutable.
- **`pkg-config --cflags gtk+-3.0`**, **`pkg-config --libs gtk+-3.0`** define la versión y banderas de GTK3.

- **-o tda_executable**, esta bandera establece el nombre del archivo binario compilado resultante.
- **-w** esta bandera es para ignorar las advertencias de compilación.

12.1.3 Reglas Udev para el Dongle USB.

Udev es el gestor de dispositivos que usa el Kernel Linux. Su función es controlar los ficheros de dispositivo en el directorio **/dev**, en **/dev** hay nodos de dispositivo creados para cada dispositivo conocido, también es posible definir reglas para dispositivos individuales gestionados por Udev estas reglas son añadidas como archivos en el directorio **/etc/udev/rules.d/**, la regla definida como **40_usb.rules** para el dispositivo USB de TDA se observa en la **Tabla 12.1.1**.

Tabla 12.1.1: Código de Regla Udev para el dispositivo USB de TDA.

```
DRIVER=="usb", SUBSYSTEMS=="usb", ATTR{idVendor}=="04b4",  
ATTR{idProduct}=="1004", GROUP="video", MODE="0666"
```

Esta regla entrega permisos de lectura y escritura a un dispositivo USB que se conecta en caliente con un IDVendor = (0x04b4)h y un IDProduct = (0x1004)h. Los usuarios pertenecientes al grupo **video** con el uso de esta regla podrán acceder al dispositivo sin necesidad de ser usuarios root del sistema GNU/Linux.

12.1.4 Acceso directo

La creación del acceso directo en GNU/Linux se basa en un archivo de texto plano con extensión **.desktop**, este archivo crea un lanzador de la aplicación, los accesos directos poseen el siguiente formato:

[Desktop Entry]

Version=Version del programa

Name=Nombre del programa

Comment=Comentario sobre el programa

Exec=Dirección del ejecutable

Icon=Dirección del ícono

Terminal=Si es lanzada en la terminal o no la aplicación

Type=Tipo de Aplicación.

Categories=Categoría de la aplicación.

Llenado los campos para el acceso directo de la aplicación desarrollada se obtuvo el siguiente formato de acceso directo.

[DesktopEntry]

Version=0.9

Name=TDA_LINUX

Comment=Televisión Digital Abierta Venezolana

Exec=/opt/TDA_LINUX/tda_executable

Icon=/opt/TDA_LINUX/logo.png

Terminal=false

Type=Application

Categories= Video;AudioVideo;

12.1.5 Script de instalación del programa

El script de instalación del programa son líneas de comandos que se ejecutan en la terminal del sistema operativo y realizan los procesos de obtención e instalación de paquetes necesarios para las rutinas desarrolladas, ejecuta el comando de compilación de la aplicación, crea el acceso directo de la aplicación, crea el directorio de instalación y modifica las reglas Udev para dar los permisos necesarios al dispositivo USB, este script debe ejecutarse con permisos elevados de usuario, utilizando el comando **sudo ./install** para la instalación del software.

A continuación se muestran las operaciones que realiza la rutina de instalación en la **Tabla 12.1.2**.

Tabla 12.1.2: Código del script de instalación.

```

#!/bin/bash

echo "Para la instalación TDA LINUX debe ser super usuario." #Mostrar mensaje

echo "Instalacion de dependencias necesarias...\n"      #Mostrar mensaje

#Añade repositorios de VLC
dnf install https://download1.rpmfusion.org/free/fedora/rpmfusion-free-release-$(rpm -E %fedora).noarch.rpm

#Se instalan las dependencias de paquetes necesarios para la compilación y ejecución de la aplicación.
sudo dnf install gcc-c++ gtkmm30-devel gstreamermm-devel cluttermm-devel webkitgtk3-devel libgdammm-devel vlc vlc-devel libusb-devel

#Se instala la versión de VLC del instalador
rpm -e --nodeps vlc-core vlc vlc-devel
rpm -ivh *.rpm

echo "Creando carpeta en la ruta de instalación...\n"      #Mostrar mensaje
    
```

```

mkdir -p /opt/TDA_LINUX
#Crea la carpeta TDA_LINUX

echo "Compilando...\n"
#Mostrar mensaje

g++ main.cpp -std=c++11 -lusb-1.0 -lvlc -pthread -rdynamic -lX11 -s `pkg-config --cflags gtk+-3.0` `pkg-config --libs gtk+-3.0` -o tda_executable -w
#Compila el código fuente y crea el ejecutable

mv tda_executable /opt/TDA_LINUX #mueve el ejecutable a la carpeta TDA_LINUX

cp logo.png /opt/TDA_LINUX #mueve el ejecutable a la carpeta TDA_LINUX

echo "Creando acceso directo...\n"
#Mostrar mensaje

touch /opt/TDA_LINUX/TDA.desktop
#crea el archivo del lanzador en la carpeta TDA_LINUX llamado TDA

shopt -s xpg_echo
echo "[Desktop
Entry]\nVersion=0.1\nName=TDA_LINUX\nComment=Televisión Digital Abierta
Venezolana\nExec=/opt/TDA_LINUX/tda_executable\nIcon=/opt/TDA_LINUX/logo.png\nTerminal=false\nType=Application\nCategories=Video;Audio;Video;" > /opt/TDA_LINUX/TDA.desktop

chmod 777 -R /opt/TDA_LINUX/
#Permisos de ejecución del lanzador

mv /opt/TDA_LINUX/TDA.desktop /usr/share/applications/ #Mover a aplicaciones

echo "Copiando regla udev...\n"

cp -R 40_usb.rules /etc/udev/rules.d/ #Mover las reglas Udev al sistema
output=$(whoami) #Obtiene el nombre del usuario
sudo usermod -a -G video $output #Añade al usuario al grupo de

```

video

```
echo "Reiniciando reglas udev...\n"           #Mostrar mensaje
sudo udevadm control --reload-rules && udevadm trigger #Reiniciar las
reglas Udev
```

12.1.6 Script de desinstalación del programa

El script de desinstalación del programa debe ejecutarse con permisos elevados de usuario, utilizando el comando **sudo ./uninstall** para la desinstalación del software y ejecuta los procesos de eliminación del directorio y los archivos de la aplicación también se carga de eliminar la regla Udev para el dispositivo USB añadida en el proceso de instalación.

A continuación se muestran las operaciones que realiza la rutina de desinstalación en la **Tabla 12.1.3**.

Tabla 12.1.3: Código del script de desinstalación.

```
#!/bin/bash
rm -R /opt/TDA_LINUX          #eliminar la carpeta TDA_LINUX
rm /etc/udev/rules.d/40_usb.rules #eliminar reglas Udev del dispositivo
TDA
rm /usr/share/applications/TDA.desktop #eliminar el lanzador
sudo udevadm control --reload-rules && udevadm trigger #Reiniciar las
reglas Udev
echo "Desisntalación finalizada.\n"      #Mostrar mensaje
```

13.1 Documentación.

La documentación de las rutinas desarrolladas lo que incluye el pseudocódigo se realizó utilizando el software de documentación **Doxygen** versión 1.8.13. Es la herramienta estándar de facto para generar documentación a partir de fuentes C++, pero también es compatible con otros lenguajes de programación populares como C, Objective-C, C#, PHP, Java, Python, Fortran, VHDL, Tcl, y hasta cierto punto D.

Puede generar una de documentación en línea para ver desde un navegador (en HTML) y/o un manual de referencia de un conjunto de archivos fuente documentados. También soporta salida en formato RTF (MS-Word), PostScript, PDF con hiperenlaces, HTML comprimido y páginas de manual Unix. La documentación se extrae directamente de las fuentes del código, lo que hace que sea mucho más fácil mantener la documentación coherente con el código fuente. Doxygen también puede visualizar las relaciones entre los distintos elementos mediante gráficos de dependencia de inclusión, diagramas de herencia y diagramas de colaboración, que se generan automáticamente. Doxygen está desarrollado bajo Mac OS X y Linux, pero está configurado para ser altamente portátil. Como resultado, funciona en la mayoría de otros sabores de Unix también. Además, los ejecutables para Windows están disponibles.

Doxygen fue configurado para generar la documentación del código fuente en formato HTML para ser visualizada desde un navegador, esta documentación incluye descripción de todas las funciones desarrolladas, pseudocódigo de todos los archivos del código fuente, diagramas de dependencia y herencia de las funciones.

En las **Figuras 13.1.1, 13.1.2, 13.1.3, 13.1.4, 13.1.5, 13.7.6 y 13.1.7** se muestra la documentación generada para el grupo de archivos del

código fuente de la aplicación desarrollada. Esta documentación puede ser encontrada en los anexos del presente trabajo de grado.



Figura 13.1.1: Documentación Doxygen.

TDA_LINUX: Página principal - Mozilla Firefox

TDA_LINUX: Página principal +

file:///home/cendit/edgar/doc_version_3/html/index.html Buscar

Rutina USB del Dongle USB de TDA version 0.9

Documentación de la rutina de control USB para el Dongle de TDA en el sistema operativo GNU/Linux

Página principal Archivos VLC.

Enlaces de interés:

- [VLC versión 3.0: VLC 3.0](#)
- [Documentación de la API LIBVLC: LIBVLC](#)

GTK3

Es un conjunto de bibliotecas multiplataforma para desarrollar interfaces gráficas de usuario (GUI), principalmente para los entornos gráficos GNOME, XFCE y ROX aunque también se puede usar en el escritorio de Windows, Mac OS y otros.

Enlaces de interés:

- [Documentación GTK: GTK](#)

Pseudocódigo

A continuación se encuentran los enlaces de los pseudocódigos para cada archivo fuente del programa:

- [Pseudocódigo main.cpp](#)
- [Pseudocódigo calls.h](#)
- [Pseudocódigo usb.h](#)
- [Pseudocódigo thread.h](#)
- [Pseudocódigo initialize.h](#)
- [Pseudocódigo samsung.h](#)

Generado por [doxygen](#) 1.8.11

Figura 13.1.2: Documentación Doxygen, enlaces pseudocódigos.

TDA Linux: Página principal - Mozilla Firefox

TDA_LINUX: Página principal TDA Linux: Página principal +

file:///home/cendit/edgar/doc_version_3/html/pseudocodigo_usb.h.html | C | Buscar | Star | Back | Forward | Stop | Home | Refresh | Print | E-mail | Help | Search | Window

TDA Linux 1.0

Rutina de control USB y reproducción

Página principal Archivos

Peseudocódigo-usb.h

```

INCLUIR libusb-1.0/libusb.h
INCLUIR iostream
INCLUIR unistd.h

ESTRUCTURA LIBUSB contexto_usb <- NULL
ESTRUCTURA LIBUSB manejador_dispositivo_usb
ENTERO estado_hotplug <- 0
BOOLEANO estado_frecuencia <- FALSE

FUNCION RETORNA ENTERO funcion_de_llamada_hotplug(RECIBE ESTRUCTURA LIBUSB contexto_usb hotplug_funcion_llamada,
RECIBE ESTRUCTURA LIBUSB dispositivo_usb hotplug_funcion_llamada,
RECIBE ENTERO evento_usb hotplug_funcion_llamada,
RECIBE ENTERO data_usuario_hotplug_funcion_llamada)

FUNCION libusb_get_device_descriptor(RECIBE ESTRUCTURA LIBUSB dispositivo_usb hotplug_funcion_llamada, RECIBE ESTRUCTURA LIBUSB descriptor_usb hotplug_funcion_llamada)

SI (LIBUSB_HOTPLUG_EVENT_DEVICE_ARRIVED IGUAL QUE evento_usb hotplug_funcion_llamada) ENTONCES
  ENTERO respuesta <- FUNCION RETORNA ENTERO libusb_open(RECIBE ESTRUCTURA LIBUSB dispositivo_usb hotplug_funcion_llamada, RECIBE ESTRUCTURA LIBUSB manejador_dispositivo_usb hotplug_funcion_llamada)
  SI (LIBUSB_SUCCESS NO ES IGUAL QUE respuesta) ENTONCES
    IMPRIMIR "Error: No pudo abrirse el dispositivo USB"
  SI NO SI (LIBUSB_HOTPLUG_EVENT_DEVICE_LEFT IGUAL QUE evento_usb hotplug_funcion_llamada) ENTONCES
    SI (manejador_dispositivo_usb hotplug_funcion_llamada) ENTONCES
      manejador_dispositivo_usb_close(RECIBE manejador_dispositivo_usb hotplug_funcion_llamada)

```

Generated by doxygen 1.8.11

Figura 13.1.3: Documentación Doxygen, ejemplo pesudocódigo.

TDA: Referencia del Archivo /home/cendit/Escritorio/version_5/include/thread.h - Mozilla Firefox

TDA_LINUX: Página principal TDA: Referencia del Archivo +

file:///home/cendit/edgar/doc_version_3/html/thread_8h.html | C | Buscar | Star | Back | Forward | Stop | Home | Refresh | Print | E-mail | Help | Search | Window

TDA

Página principal Clases Archivos

Lista de archivos Miembros de los ficheros

Variables

```

const int tamano_buffer_MPEG_TS = 245760
Constante entera tamaño del buffer = 245760. Más...
static unsigned char MPEG2_TS [tamano_buffer_MPEG_TS]
Buffer para alojar los paquetes BTS. Más...
const int tamano_buffer = 4915200
Constante entera tamaño del buffer = 4915200. Más...
static unsigned char buffer [tamano_buffer]
Buffer rotativo para alojar los paquetes BTS. Más...
static int posicion =0
Variable para incrementar el apuntor del buffer y posicionarse en varias posiciones de memoria dentro del buffer. Posiciones= {0,245760,491520,737280,983040,1228800,1474560,1720320,1966080,2211840,2457600,2703360,2949120,3194880,3440640,3686400, Más...
static int contador_evitar_basura = 0
Variable para evitar basura de las primeras transmisiones. Más...
static bool estado_detener = false
Variable estado_detener para detectar si el reproductor esta detenido. Más...
unsigned char * apuntador_al_buffer_repcion_BTS =buffer
Varibale que apunta al buffer de transferencias USB. Más...

```

version_5 > include > thread.h

Generated by doxygen 1.8.11

Figura 13.1.4: Documentación Doxygen, ejemplo documentación de variables.

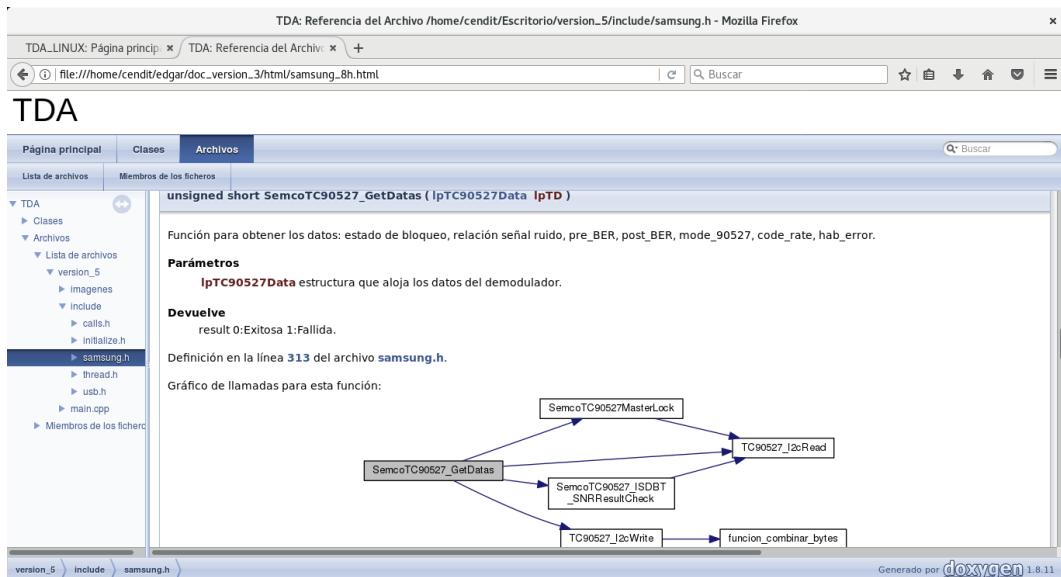


Figura 13.1.5: Documentación Doxygen, ejemplo documentación de funciones.

Para la creación del manual de instalación de la aplicación desarrollada se utilizó el software de ofimática LibreOffice versión 5.3.4.2 el cual es software libre, posee un aplicación llamada LibreOffice Writer la cual es el procesador de texto, esta funciona de manera cada vez más similar a las aplicaciones Microsoft Word y WordPerfect, permite exportar archivos de texto a distintos formatos como pueden ser PDF y HTML sin software adicional, utiliza la funcionalidad WYSIWYG lo que permite escribir un documento viendo directamente el resultado final, también puede utilizarse como un simple editor de textos.

El manual de instalación desarrollado con LibreOffice Writer consta de una serie de pasos detallados los cuales contienen imágenes que pretenden facilitar la realización de las acciones de instalación de la aplicación desarrollada de manera exitosa. El manual se encuentra en formato PDF esto debido a que este formato puede ser visto usando diferentes tipos de software como aplicaciones ofimáticas, visor de



documentos o navegadores web. En las **Figura 13.1.6 y 13.1.7** se puede apreciar muestras del manual de instalación dirigido al usuario, este manual puede ser encontrado en los anexos del presente trabajo de grado.

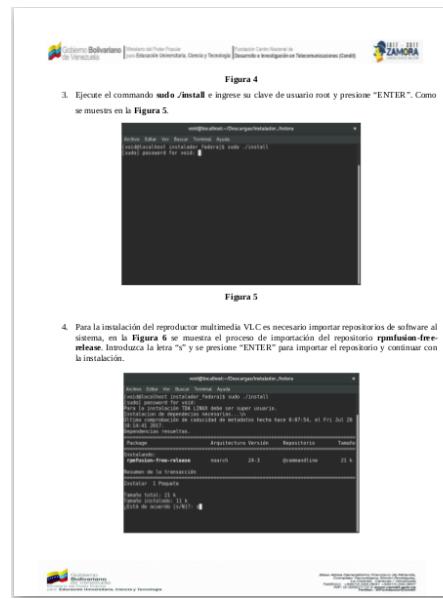
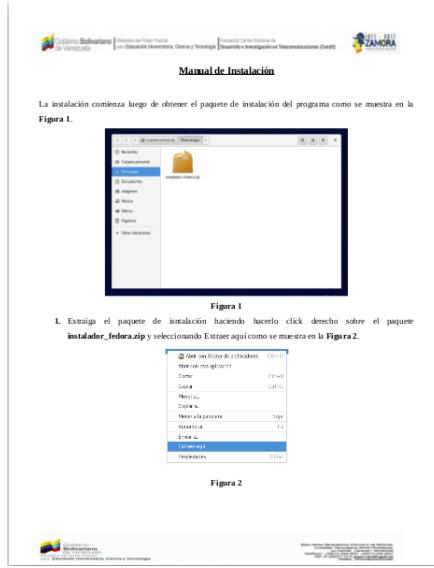


Figura 13.1.6: Manual de instalación muestra número uno.

Figura 13.1.7: Manual de instalación muestra número dos.

14.1 Ejecución del instalador del programa.

La ejecución del script de instalación se le realizó mediante la terminal usando el comando **sudo ./install** como se muestra en la **Figura 14.1.1** posteriormente se introdujo la contraseña de usuario root y se presionó “ENTER”.

```
void@localhost:~/Descargas/installador_fedora
Archivo Editar Ver Buscar Terminal Ayuda
[void@localhost instalador fedora]$ sudo ./install
[sudo] password for void: █
```

Figura 14.1.1: Ejecución del comando de instalación del programa.

Para la instalación del reproductor multimedia VLC es necesario importar repositorios de software al sistema, en la **Figura 14.1.2** se muestra el proceso de importación del repositorio **rpmfusion-free-release**. Se introdujo la letra “s” y se presionó “ENTER” para importar el repositorio y continuar con la instalación.

```

void@localhost:~/Descargas/instalador_fedora
Archivo Editar Ver Buscar Terminal Ayuda
[void@localhost instalador_fedora]$ sudo ./install
[sudo] password for void:
Para la instalación TDA LINUX debe ser super usuario.
Instalacion de dependencias necesarias...\\n
Última comprobación de caducidad de metadatos hecha hace 0:07:54, el Fri Jul 28
10:14:41 2017.
Dependencias resueltas.
=====
Package          Arquitectura Versión     Repositorio      Tamaño
=====
Instalando:
  rpmfusion-free-release      noarch        24-3           @commandline    21 k
Resumen de la transacción
=====
Instalar 1 Paquete
Tamaño total: 21 k
Tamaño instalado: 11 k
¿Está de acuerdo [s/N]?:

```

Figura 14.1.2: Importar repositorios de software para VLC en el proceso de instalación.

Luego de la importación de los repositorios para VLC se procede a la instalación de las librerías y software necesario para la compilación y posterior ejecución del programa desarrollado, como se muestra en la **Figura 14.1.3** el instalador muestra los paquetes a instalar. Se introdujo la letra “s” y se presionó “ENTER” para la instalación de estos paquetes y para continuar con la instalación.

```
void@localhost:~/Descargas/instalador_fedora
Archivo Editar Ver Buscar Terminal Ayuda
s
x265-libs           x86_64 1.9-1.fc24          rpmfusion-free 574 k
xorg-x11proto-devel noarch 7.7-19.fc24        fedora       556 k
xvidcore            x86_64 1.3.4-2.fc24        rpmfusion-free 287 k
xz-devel             x86_64 5.2.2-2.fc24        fedora       262 k
zlib-devel           x86_64 1.2.8-10.fc24       fedora      60 k
zvbi                x86_64 0.2.35-1.fc24       fedora      55 k
Actualizando:
cpp                 x86_64 6.3.1-1.fc24       updates     414 k
gcc                 x86_64 6.3.1-1.fc24       updates     9.0 M
gcc-gdb-plugin      x86_64 6.3.1-1.fc24       updates     20 M
libgcc              x86_64 6.3.1-1.fc24       updates     85 k
libgomp              x86_64 6.3.1-1.fc24       updates     89 k
libstdc++            x86_64 6.3.1-1.fc24       updates     191 k
webkitgtk3          x86_64 2.4.11-2.fc24       updates     451 k
Resumen de la transacción
=====
Instalar   149 Paquetes
Actualizar   7 Paquetes

Tamaño total de la descarga: 116 M
¿Está de acuerdo [s/N]?: s
```

Figura 14.1.3: Instalación de librerías y software necesario en el proceso de instalación.

Después de acceder a la instalación de los paquetes necesarios el instalador pide verificar el repositorio importado como método de confianza en el software externo que se instala desde el repositorio externo **rpmfusion-free-release** como se muestra en la **Figura 14.1.4**. Se introdujo la letra “s” y se presionó “ENTER” para verificar el repositorio y continuar con la instalación.

```
void@localhost:~/Descargas/instalador_fedora

Archivo Editar Ver Buscar Terminal Ayuda
(151/156): subunit-devel-1.2.0-4.fc24.x86_64.rpm 51 kB/s | 12 kB 00:00
(152/156): dbus-devel-1.11.2-1.fc24.x86_64.rpm 222 kB/s | 60 kB 00:00
(153/156): liba52-0.7.4-25.fc24.x86_64.rpm 155 kB/s | 44 kB 00:00
(154/156): libicu-devel-56.1-4.fc24.x86_64.rpm 351 kB/s | 750 kB 00:02
(155/156): cpp-6.3.1-1.fc24.x86_64.rpm 325 kB/s | 9.0 MB 00:28
[MIRROR] gcc-6.3.1-1.fc24.x86_64.rpm: Status code: 421 for http://mirror.cedia.org.ec/fedora/updates/24/x86_64/g/gcc-6.3.1-1.fc24.x86_64.rpm
(156/156): gcc-6.3.1-1.fc24.x86_64.rpm 171 kB/s | 20 MB 02:00
[DRPM] libstdc++-6.1.1-2.fc24_6.3.1-1.fc24.x86_64.rpm: hecho
[DRPM] libgomp-6.1.1-2.fc24_6.3.1-1.fc24.x86_64.rpm: hecho
[DRPM] gcc-gdb-plugin-6.1.1-2.fc24_6.3.1-1.fc24.x86_64.rpm: hecho
[DRPM] webkitgtk3-2.4.11-1.fc24_2.4.11-2.fc24.x86_64.rpm: hecho
-----
Total 439 kB/s | 108 MB 04:13
Delta RPMs redujo 115.8 MB de actualizaciones a 108.4 MB (6.1% de ahorro)
advertencia:/var/cache/dnf/rpmfusion-free-updates-1e475027b1818d49/packages/vlc-3.0.0-0.28.git20170622.fc24.x86_64.rpm: EncabezadoV4 RSA/SHA1 Signature, ID de clave b7546f06: NOKEY
Importando llave GPG 0xB7546F06:
ID usuario: "RPM Fusion free repository for Fedora (24) <rpmfusion-buildsys@lists.rpmfusion.org>"
Huella: 55E7 903B 6087 98E4 EC78 64CD 9F63 8721 B754 6F06
Desde: /etc/pki/rpm-gpg/RPM-GPG-KEY-rpmfusion-free-fedora-24
¿Está de acuerdo [s/N]? : s
```

Figura 14.1.4: Verificar huella del repositorio importado en el proceso de instalación.

Finalmente se mostraron los mensajes de las acciones de: Creando carpeta en la ruta de instalación, Compilando, Creando acceso directo, Creando regla udev y Reiniciando regla udev, como se muestra en la **Figura 14.1.5.**

```
void@localhost:~/Descargas/instalador_fedora
Archivo Editar Ver Buscar Terminal Ayuda
webkitgtk3-devel.x86_64 2.4.11-2.fc24
x264-libs.x86_64 0.148-13.20160924git86b7198.fc24
x265-libs.x86_64 1.9-1.fc24
xorg-x11-proto-devel.noarch 7.7-19.fc24
xvidcore.x86_64 1.3.4-2.fc24
xz-devel.x86_64 5.2.2-2.fc24
zlib-devel.x86_64 1.2.8-10.fc24
zvbi.x86_64 0.2.35-1.fc24

Actualizado:
cpp.x86_64 6.3.1-1.fc24          gcc.x86_64 6.3.1-1.fc24
gcc-gdb-plugin.x86_64 6.3.1-1.fc24 libgcc.x86_64 6.3.1-1.fc24
libgomp.x86_64 6.3.1-1.fc24      libstdc++.x86_64 6.3.1-1.fc24
webkitgtk3.x86_64 2.4.11-2.fc24

¡Listo!
Creando carpeta en la ruta de instalacion...\\n
Compilando...\\n
Creando acceso directo...\\n
Copiando regla udev...

Reiniciando reglas udev...

[void@localhost instalador_fedora]$
```

Figura 14.1.5: Creación de directorios, compilación y reglas Udev.

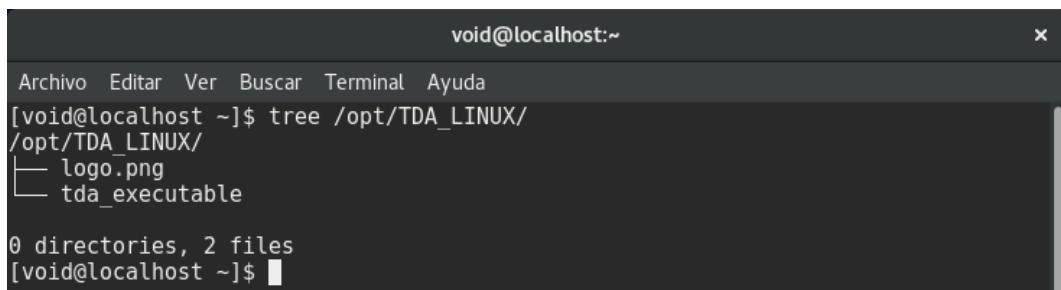
En la **Figura 14.1.6** se observa usando el comando **tree /etc/udev/rules.d** que fue añadida la nueva regla llamada **40_usb.rules** para el dispositivo USB de TDA.

```
void@localhost:~
Archivo Editar Ver Buscar Terminal Ayuda
[void@localhost ~]$ tree /etc/udev/rules.d/
/etc/udev/rules.d/
└── 40_usb.rules

0 directories, 1 file
[void@localhost ~]$
```

Figura 14.1.6: Estructura de archivos de reglas udev.

En la **Figura 14.1.7** se observa usando el comando **tree** /opt/TDA_LINUX que fue creado el directorio que contiene el ejecutable de la aplicación desarrollada y el logo del lanzador de la aplicación.



```
void@localhost:~$ tree /opt/TDA_LINUX/
/opt/TDA_LINUX/
└── logo.png
    └── tda_executable

0 directories, 2 files
[void@localhost ~]$
```

Figura 14.1.7: Estructura de archivos del directorio de instalación.

Ya finalizado el proceso de instalación fue creado un lanzador a la aplicación como se muestra en la **Figura 14.1.8** con el nombre **TDA_LINUX**.

Al lanzar la aplicación se abrió la aplicación desarrollada como se muestra en la Figura **14.1.9**.

El proceso de instalación engloba a todas las rutinas desarrolladas en este trabajo de grado menos la rutina de desinstalación, el script de instalación compila a las rutinas del software para crear el ejecutable de la aplicación, crea el lanzador de la aplicación y añade la regla USB para el dispositivo de TDA, durante este proceso de compilación no se evidenciaron errores por lo cual el proceso de compilación fue exitoso también pudo comprobarse la inserción de la nueva regla udev, la creación del lanzador y la correcta ejecución del software.

En las imágenes mostradas se observa que se logró satisfactoriamente el proceso de ejecución e instalación del software controlador USB y reproductor multimedia desarrollado bajo el sistema operativo **GNU/Linux Fedora 24**.

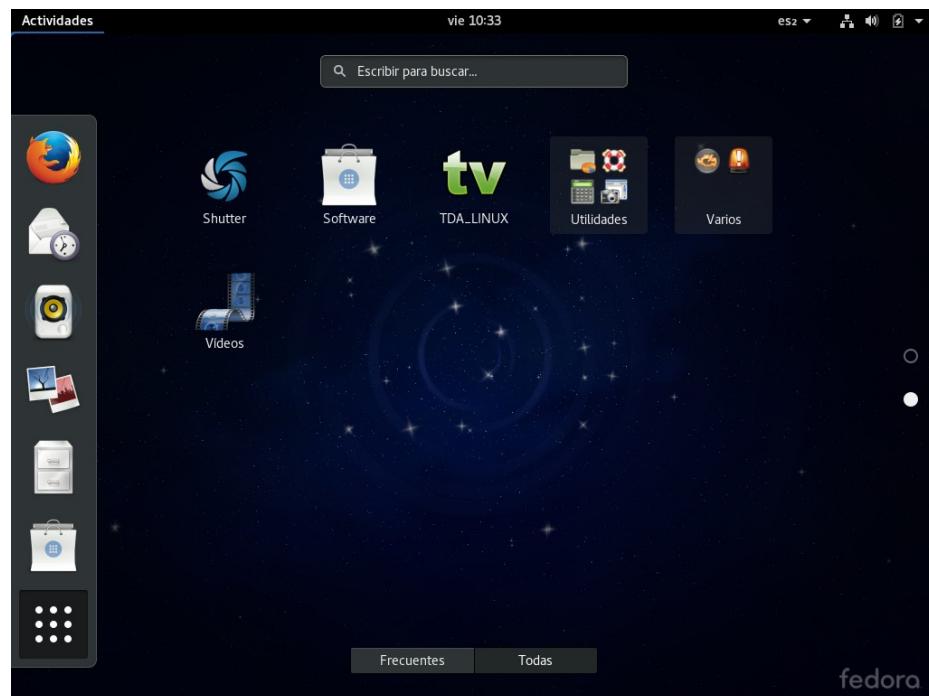


Figura 14.1.8: Lanzador de la aplicación.

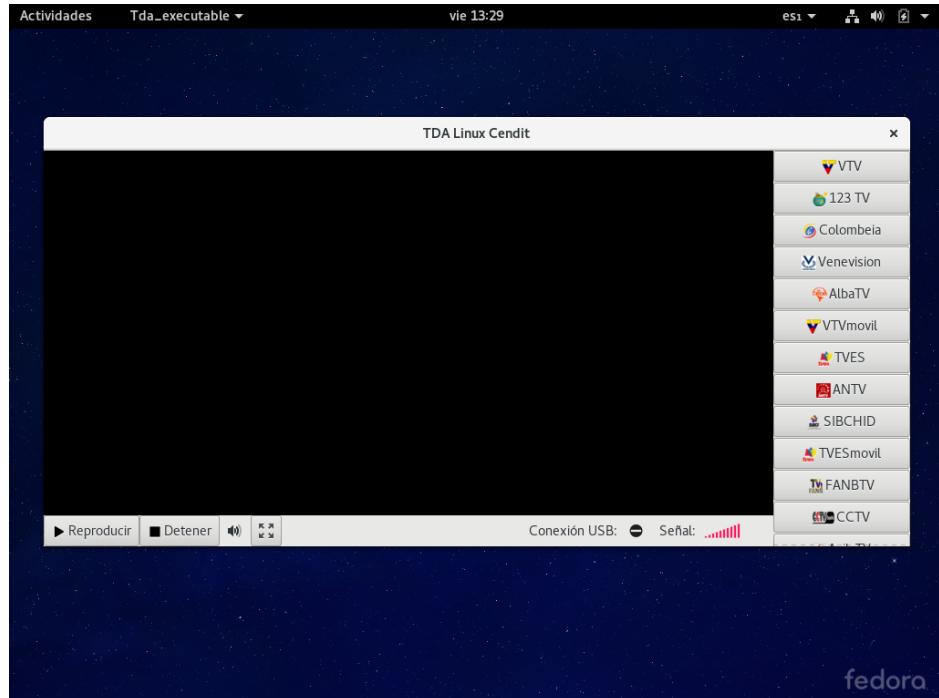


Figura 14.1.9: Ejecución de la aplicación.

14.2 Montaje físico de los equipos.

Para las pruebas del software desarrollado se utilizó el prototipo de pruebas de USB de TDA [Error: no se encontró el origen de la referencia], el siguiente diagrama de conexiones mostrado en la **Figura 14.2.1** muestra las conexiones y equipos utilizados para la realización de las pruebas.

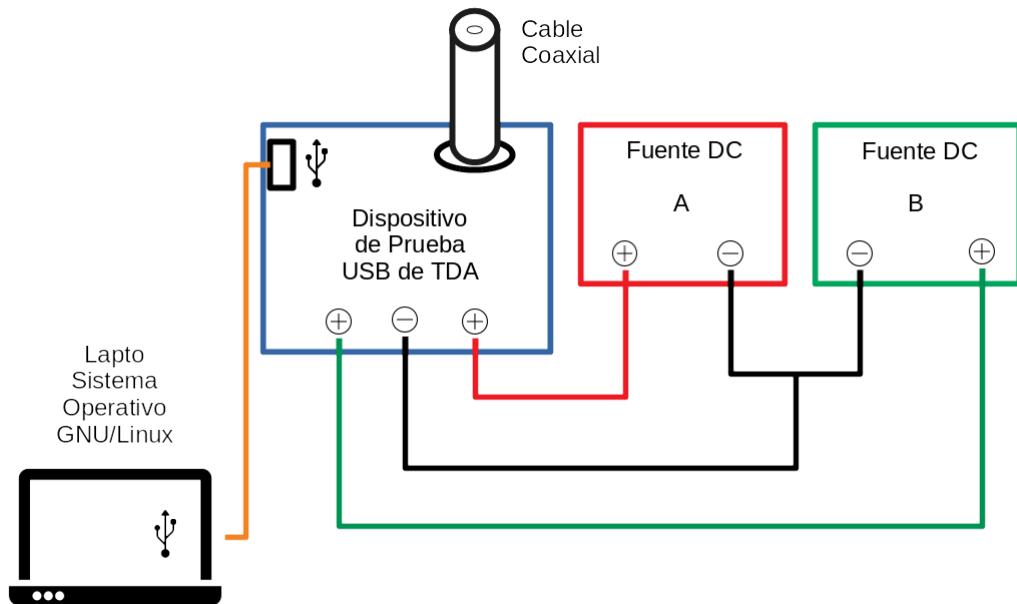


Figura 14.2.1: Diagrama de conexiones montaje físico.

La fuente DC A fue calibrada a (5.1 ± 0.1) V y la fuente DC B fue calibrada a (1.2 ± 0.1) V, estas fuentes entregan el voltaje necesario al dispositivo de prueba para su funcionamiento.

El dispositivo de prueba recibe la señal de TDA a través de un cable coaxial que es conectado al terminal del dispositivo Tuner NIM de Samsung y el puerto USB de la placa de desarrollo es conectado a la PC.

A continuación se muestra el montaje en físico realizado en la **Figura 14.2.2.**

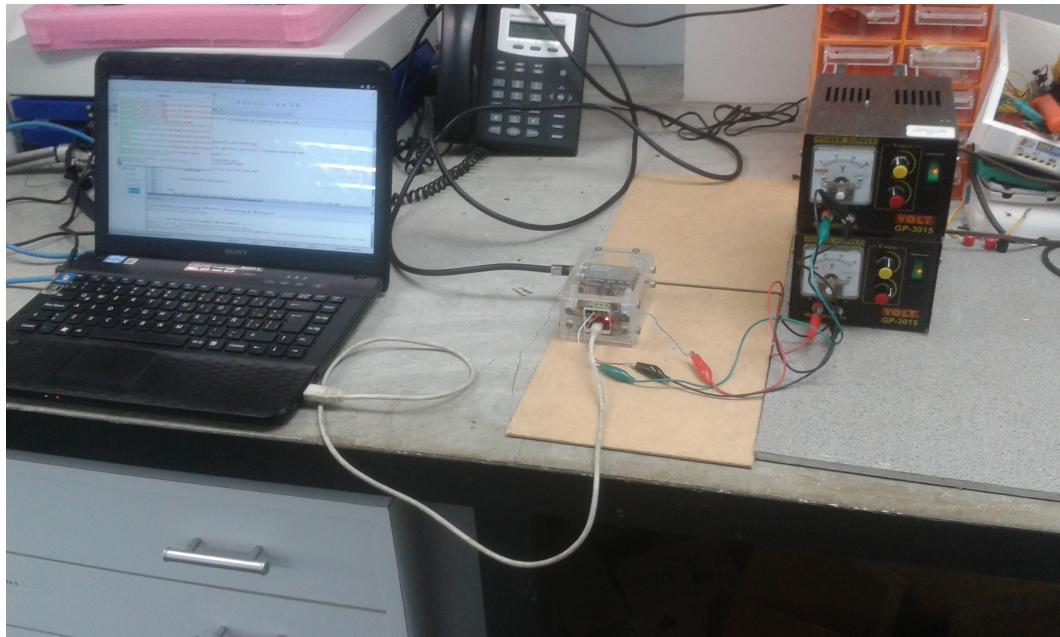


Figura 14.2.2: Montaje en físico de las pruebas.

En la **Figura 14.2.3** se muestra detalladamente la conexión física del dispositivo de prueba USB de TDA.

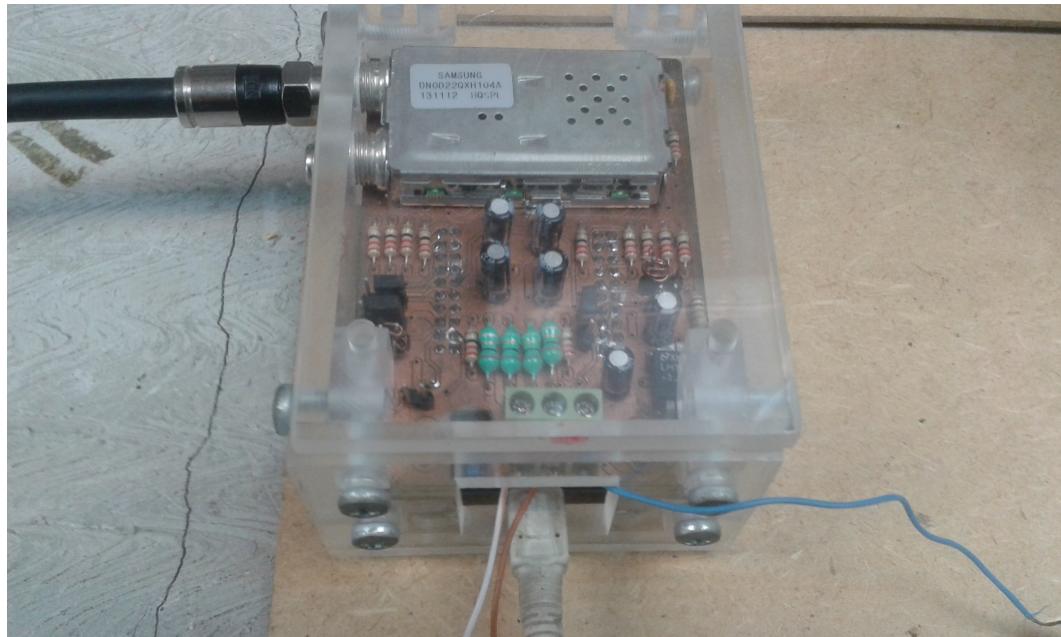


Figura 14.2.3: Conexión en físico del dispositivo USB de TDA.

Lista de equipos utilizados para la realización de las pruebas:

- Lpto Siragon Modelo MN-50.
- Lpto Sony Vaio Modelo VPCCA.
- 2 Fuentes DC Modelo GP-3015.
- 3 cables tipo 16 con conectores caimanes en ambos extremos.
- Cable Coaxial de 1 metro de largo conectores hembras de tipo f.
- Cable de conexión microusb.
- Multímetro Fluke Modelo 179, para calibrar la salida deseada de las fuentes DC.

14.3 Pruebas de la aplicación desarrollada.

Al ejecutar la aplicación esta inició el proceso de detección en caliente para detectar al dispositivo USB de Cypress al ser conectado o desconectado del puerto USB de la PC para luego continuar con las operaciones de recepción y reproducción del contenido multimedia.

Luego de haber insertado el dispositivo de USB a la PC el ícono en la barra de indicadores con la etiqueta “Conexión USB” cambio para mostrar un signo de más indicando que el dispositivo fue reconocido y esta listo para ser usado, como se muestra en la **Figura 14.3.1**.

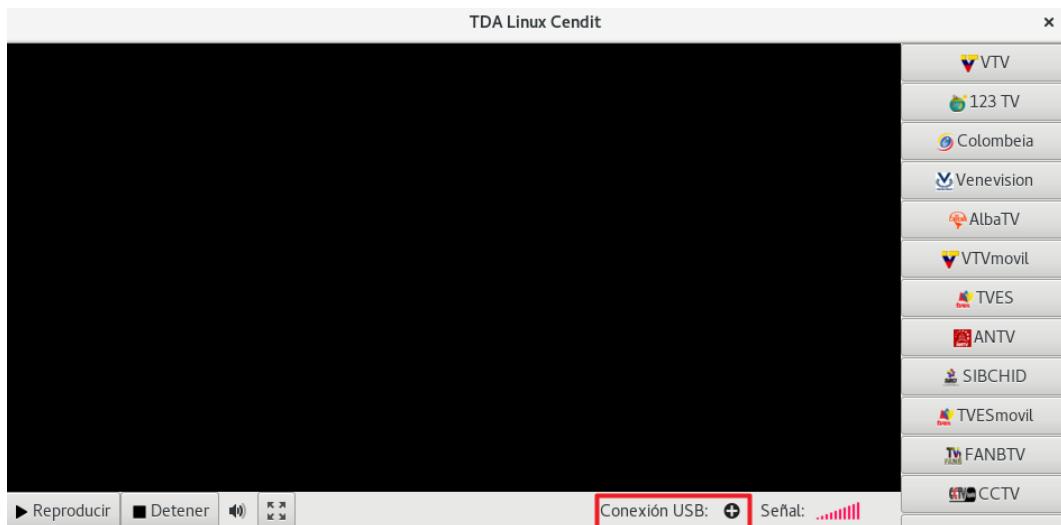


Figura 14.3.1: Dispositivo USB de TDA conectado.

Luego de conectar el dispositivo se procedió a dar click en los programas **123 TV**, **SIBCHID** y **AvilaTV Móvil** para su sintonización y reproducción como se muestra en las **Figuras 14.3.1**, **14.3.2** y **14.3.3**.

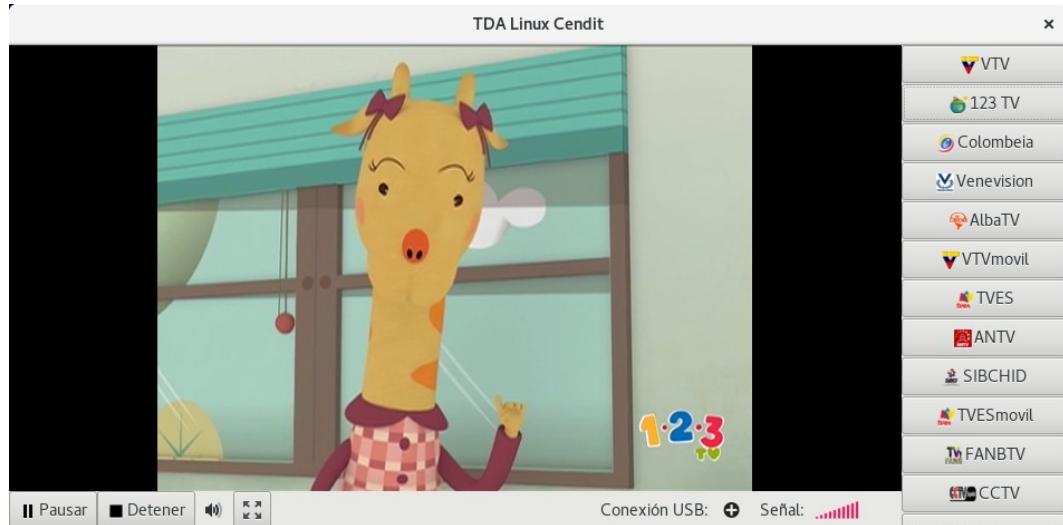


Figura 14.3.2: Prueba de sintonización y reproducción del programa 123 TV.

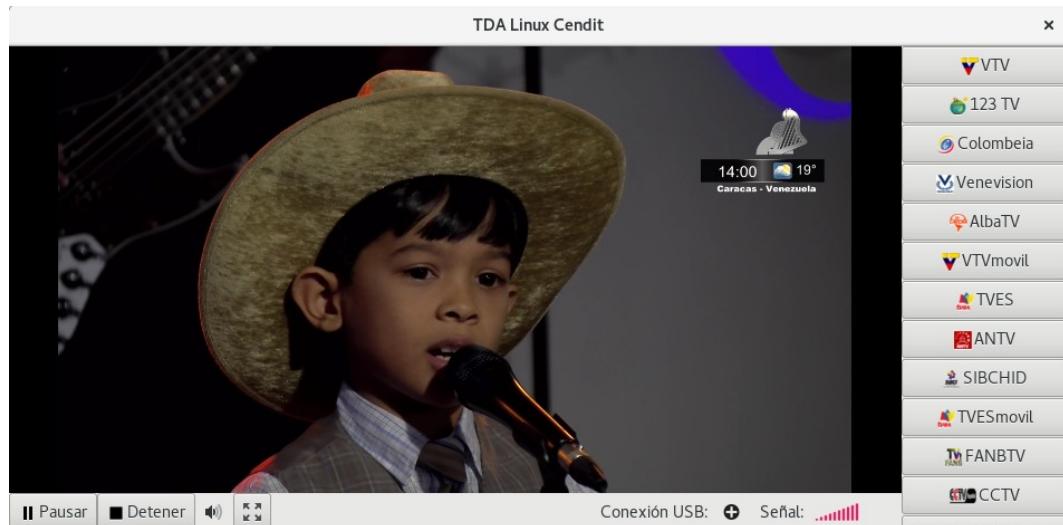


Figura 14.3.3: Prueba de sintonización y reproducción del programa SIBCHID.



Figura 14.3.4: Prueba de sintonización y reproducción del programa Avila TV Móvil.

A continuación se muestran imágenes de la aplicación sintonizando y reproduciendo el programa de TDA Meridiano en el entorno de escritorio, como se muestra en la **Figura 14.3.5** luego la aplicación maximizada en la **Figura 14.3.6** y la aplicación en pantalla completa en la **Figura 14.3.7**.

En la **Figura 14.3.8** se observa la ejecución del control de volumen de la al momento de sintonizar el programa CCTV.



Gobierno Bolivariano
de Venezuela

Ministerio del Poder Popular
para Educación Universitaria, Ciencia y Tecnología

Fundación Centro Nacional de
Desarrollo e Investigación en Telecomunicaciones (Cendit)

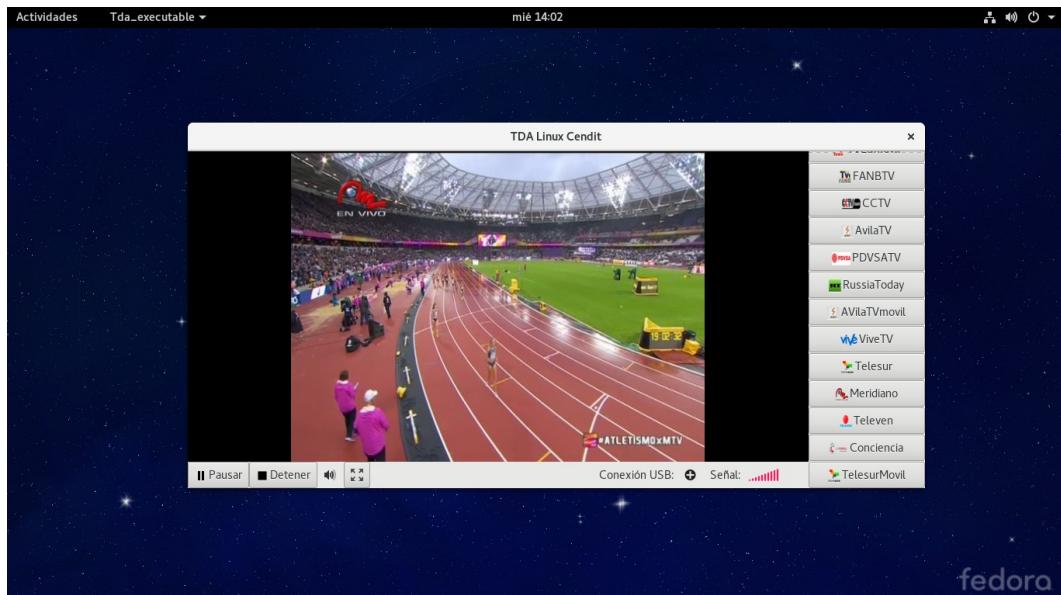


Figura 14.3.5: Prueba de sintonización y reproducción del programa Meridiano, vista del escritorio.

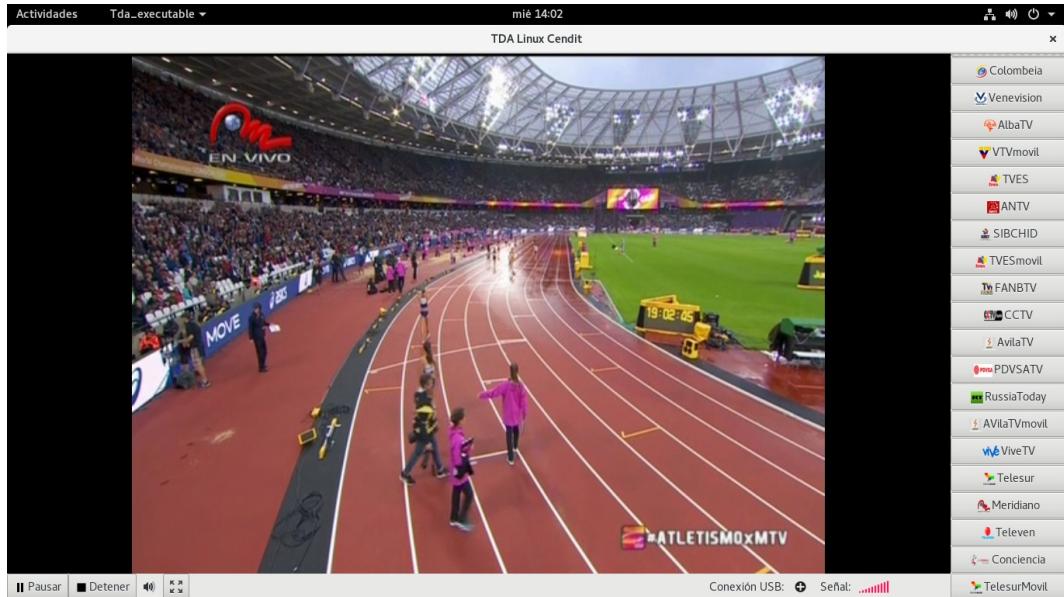


Figura 14.3.6: Prueba de sintonización y reproducción del programa Meridiano, aplicación maximizada.



Gobierno Bolivariano
de Venezuela

Ministerio del Poder Popular
para Educación Universitaria, Ciencia y Tecnología

Fundación Centro Nacional de
Desarrollo e Investigación en Telecomunicaciones (Cendit)



Figura 14.3.7: Prueba de sintonización y reproducción del programa Meridiano, modo pantalla completa.



Figura 14.3.8: Prueba de sintonización y reproducción del programa CCTV, control de volumen.

En la imagen de la **Figura 14.3.9** se muestra el montaje en físico conjuntamente con la aplicación desarrollada ejecutándose y reproduciendo el programa 123 TV de TDA.

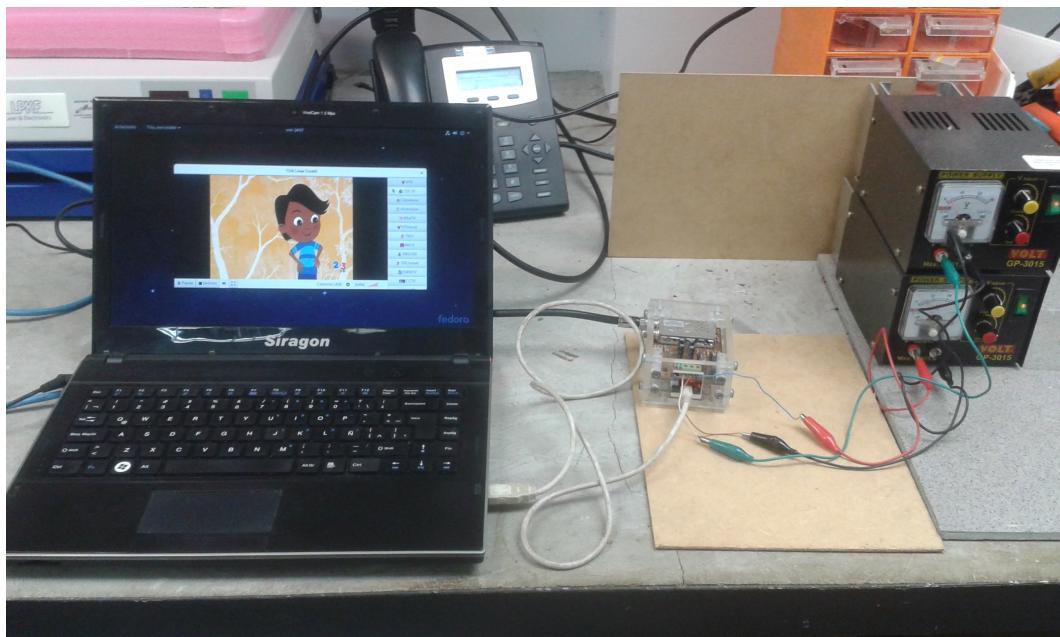


Figura 14.3.9: Montaje físico prueba de sintonización y reproducción del programa 123 TV.

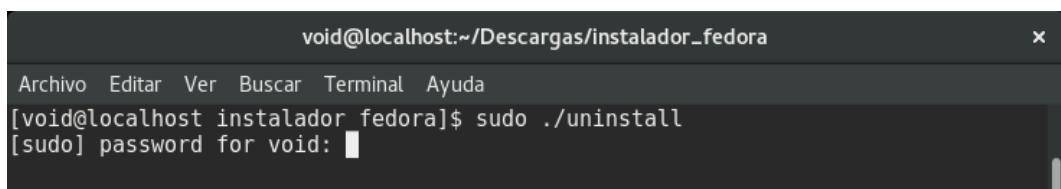
En las imágenes mostradas se evidencia que las rutinas desarrolladas fueron capaces de realizar la detección en caliente del dispositivo a través del puerto USB, realizar acciones de control sobre el dispositivo USB de TDA a la hora de inicializar el demodulador y sintonizador del dispositivo Samsung, ejecutar procesos de sintonización de las frecuencias deseadas de grupos de programas, el manejo y control de los buses de datos al realizar la transferencia de datos del contenido multimedia y transferencias de control, la reproducción y selección de contenido multimedia específico pertenecientes a los paquetes de

frecuencias de la TDA venezolana además de acciones de control sobre el contenido multimedia.

Un inconveniente fue la obtención de la intensidad de la señal la cual no pudo obtenerse utilizando la función de la API de Samsung diseñada para tal proceso, aun así se decidió dejar la imagen de la intensidad de la señal en la aplicación desarrollada para su uso cuando el defecto con la aplicación de la API de Samsung se ha resuelto.

14.4 Ejecución del script de desinstalación del programa.

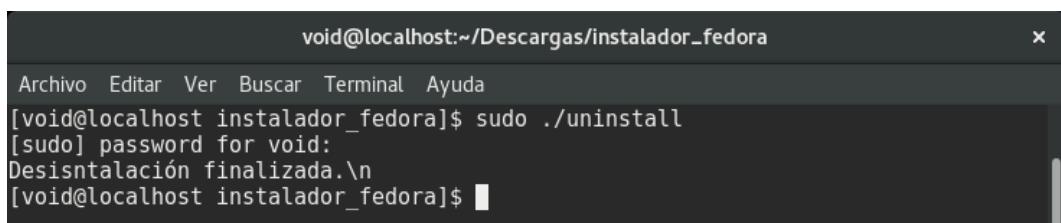
La ejecución del script de desinstalación se le realizó mediante la terminal usando el comando **sudo ./uninstall** como se muestra en la **Figura 14.4.1** posteriormente se introdujo la contraseña de usuario root y se presionó “ENTER”.



```
void@localhost:~/Descargas/instalador_fedora
Archivo Editar Ver Buscar Terminal Ayuda
[void@localhost instalador_fedora]$ sudo ./uninstall
[sudo] password for void:
```

Figura 14.4.1: Ejecución del comando de desinstalación del programa.

Luego de finalizada la desinstalación se mostró el siguiente mensaje de la **Figura 14.4.2**.

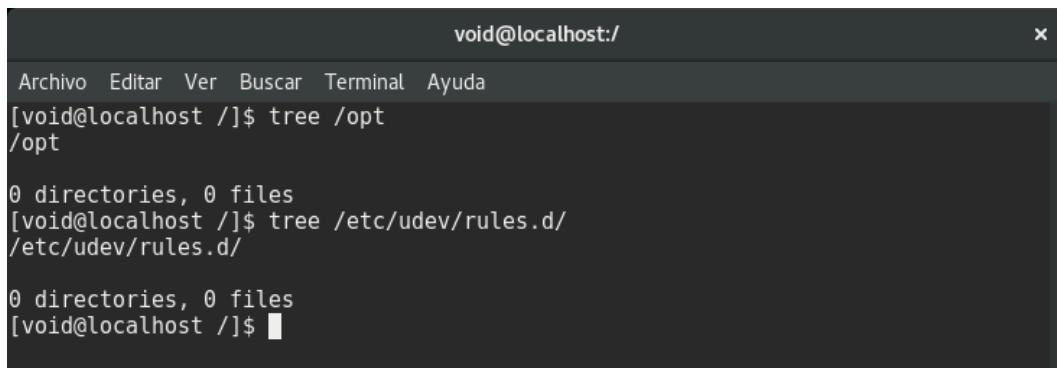


```
void@localhost:~/Descargas/instalador_fedora
Archivo Editar Ver Buscar Terminal Ayuda
[void@localhost instalador_fedora]$ sudo ./uninstall
[sudo] password for void:
Desisntalación finalizada.\n
[void@localhost instalador_fedora]$
```

Figura 14.4.2: Mensaje de desinstalación del programa.



La desinstalación del programa se verificó la eliminación del directorio de programa mediante el comando **tree /opt** y mediante el comando **tree /etc/udev/rules.d/** la eliminación de la regla udev para el dispositivo USB de TDA como se muestra en la **Figura 14.4.3.**



```
void@localhost:/
Archivo Editar Ver Buscar Terminal Ayuda
[void@localhost /]$ tree /opt
/opt
0 directories, 0 files
[void@localhost /]$ tree /etc/udev/rules.d/
/etc/udev/rules.d/
0 directories, 0 files
[void@localhost /]$ █
```

Figura 14.4.3: Verificación de desinstalación.

Al finalizar el script de desinstalación se comprobó la completa eliminación del software desarrollado del sistema operativo GNU/Linux Fedora 24 en el cual fue instalado como se muestra en la **Figura 14.3.3.**

CONCLUSIONES Y RECOMENDACIONES

GNU/Linux es un sistema operativo que cuenta con él apoya de una cantidad enorme de desarrolladores, el sistema operativo cuenta con miles de repositorios de programas y librerías que facilitan las tareas de desarrollo de software sin verse el desarrollador en la necesidad de empezar desde de cero, el apoyo de la comunidad alrededor de este sistema operativo es muy bueno siempre que surja una duda o dificultad puedes dirigirte directamente a la comunidad preguntar o pedir soporte sobre un tema, es software libre de costos y es accesible para todo aquel que quiera descargarlo. Uno de los repositorios usados para el manejo de dispositivos USB fue la librería Libusb que permite una abstracción a nivel de software para el manejo de dispositivos USB lo cual fue de gran ayuda para realizar un controlador USB en el espacio de usuario para dispositivo de TDA, también la librería Libvlc permitió la creación de rutinas para el manejo de la información multimedia usando el nucleo del reproductor VLC y la interfaz gráfica fue creada en base a las librerías de GTK, todas las herramientas usadas fueron creadas y son mantenidas por un gran número de usuarios que aportan su tiempo y conocimiento en estos proyectos, en este trabajo de grado se juntaron piezas de software de distintos proyectos con la finalidad de hacerlas trabajar en conjunto para lograr los objetivos planteados.

El software desarrollado en este trabajo de grado busca brindar una alternativa al software privativo y dar acceso libre a las tecnologías de la información basándose en la Ley de Infogobierno del Estado Venezolano, la Fundación CENDIT siendo un ente público e impulsor de este proyecto debe velar porque la ley de Infogobierno se cumpla y siempre optar en lo

que sea posible a soluciones de software libre por lo cual este trabajo de grado se llevó a la realización.

En cuanto a las recomendaciones, existen varias recomendaciones que hacer sobre el software desarrollado, las cuales se listan a continuación:

- En cuanto a la obtención de la intensidad de la señal debería contarse con soporte de parte de Samsung sobre el manejo de la API del Tuner NIM ya que hay una falta de información sobre a cuales registros apunta la función a la hora de obtener datos.
- Implementar una rutina de instalación más amistosa para el usuario, podría hacerse mediante una aplicación con interfaz gráfica o un empaquetado de software como RPM p DEB.
- Una rutina de sintonización automática de programas de TDA que logre listar los programas de TDA disponibles para su visualización sin necesidad de modificar el software cada vez que un nuevo programa es agregado.
- Otras funciones extras como capacidad de grabar el contenido en reproducción, adelantar y retroceder el contenido multimedia son funciones que el usuario puede encontrar provechosas.

REFERENCIAS BIBLIOGRÁFICAS

- [1] J. Manjares (2015), *Diseño y construcción de prototipo industrializable de dispositivo portátil de sintonización full-seg de TDA, compatible con la norma ISDB-Tb, para recepción en un computador a través del puerto USB*, CENDIT, Caracas.
- [2] N. Pisciotta, C. Liendo, R. Lauro, *TRANSMISIÓN DE TELEVISIÓN DIGITAL TERRESTRE EN LA NO RMA ISDB-Tb*, 2007.
- [3] *Televisión digital terrestre en Venezuela* (2017). En Wikipedia. Recuperado el 05 de Mayo de 2017 de https://es.wikipedia.org/wiki/Televisi%C3%B3n_digital_terrestre_en_Venezuela
- [4] Axelson, *USB Complete The Developer Guide*, 2015.
- [5] J. Axelson, *Serial Port Complete COM Ports, USB Virtual COM Ports, and Ports for Embedded Systems*, 2007.
- [6] *USB in a NutShell, Making sense of the USB standard* (2010). Recuperado el 05 de Mayo de 2017 de <http://www.beyondlogic.org/usbnutshell/usb1.shtml>
- [7] J. Corbet, A. Rubini, G. Kroah-Hartman, *Linux Device Drivers*, 2007.

[8] D. Córdoba (2014). GNU/Linux: Arquitectura básica del sistema. España. Recuperado de <https://juncotic.com/gnulinux-arquitectura-basica-del-sistema/>

[9] *GNU/Linux* (2017). En Wikipedia. Recuperado el 15 de Mayo de 2017 de <https://es.wikipedia.org/wiki/GNU/Linux>

[10] KJK (2011). *S/W Porting Guide, SDB-T NIM* : *DNOD22QXV104A*, 2011.

[11] *libusb-1.0 API Reference* (2015). Recuperado el 07 de Febrero de 2017 de <http://libusb.sourceforge.net/api-1.0/>

[12] A. Ott, (2014). USB and the Real World. Estados Unidos.: Signal11. Recuperado de <http://www.signal11.us/>

[13] *VLC 3.0.0-git* (2014). Recuperado el 11 de Marzo de 2017 de https://www.videolan.org/developers/vlc/doc/doxygen/html/group__libvlc.html

[14] K .Opasiak, (2015). Understand USB (in Linux). Polonia: linuxfoundation. Recuperado de <http://events.linuxfoundation.org/>

[15] B. Kernighan, D. Ritchie, *The C programming Language*, 1988.

[16] Doxygen(2017) Recuperado el 15 de Marzo de 2017 de <http://www.stack.nl/~dimitri/doxygen/>

[17] Asamblea Nacional de la República Bolivariana de Venezuela.
(10 de Octubre de 2013). Artículo 34 [Título I]. *Ley de Infogobierno.* [Ley 402-14 de 2013]. Gaceta: 40.217

[18] Asamblea Nacional de la República Bolivariana de Venezuela.
(10 de Octubre de 2013). Artículo 35 [Título I]. *Ley de Infogobierno.* [Ley 402-14 de 2013]. Gaceta: 40.217

ANEXOS

Repositorio del código fuente

https://github.com/aurquiel/Open_Digital_Television_Linux_USB_VLC/tree/master/src

Repositorio de la documentación

https://github.com/aurquiel/Open_Digital_Television_Linux_USB_VLC/tree/master/Documentation

Repositorio del manual de uso

https://github.com/aurquiel/Open_Digital_Television_Linux_USB_VLC/tree/master/Documentation