ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TESIS PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTOR: JORDÁN ÁLVAREZ, PATRICIO JAVIER

TEMA: PROTOTIPO DE UN CONTROL REMOTO CON ACCESO VIRTUAL PARA PRUEBAS DE USABILIDAD DE APLICACIONES INTERACTIVAS DE TELEVISION DIGITAL TERRESTRE

DIRECTOR: DR. OLMEDO, GONZALO

CODIRECTOR: ING. ACOSTA, FREDDY

SANGOLQUÍ

2015

UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICADO

Dr. Gonzalo Olmedo e Ing. Freddy Acosta

CERTIFICAN

Que el trabajo titulado **"PROTOTIPO DE UN CONTROL REMOTO CON ACCESO VIRTUAL PARA PRUEBAS DE USABILIDAD DE APLICACIONES INTERACTIVAS DE TELEVISION DIGITAL TERRESTRE"**, realizado por el SR. PATRICIO JAVIER JORDÁN ÁLVAREZ, ha sido guiado y revisado periódicamente y cumple normas estatutarias establecidas por la ESPE, en el Reglamento de Estudiantes de la Universidad de las Fuerzas Armadas - ESPE.

El mencionado trabajo consta de un documento empastado y un disco compacto el cual contiene los archivos en formato portátil de Acrobat (pdf). Autorizan al SR. PATRICIO JAVIER JORDÁN ÁLVAREZ que lo entregue al Ingeniero Paúl Bernal, en su calidad de Coordinador de la Carrera.

Sangolquí, 08 de mayo de 2015

Dr. Gonzalo Olmedo
DIRECTOR

Ing. Freddy Acosta
CODIRECTOR

UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

PATRICIO JAVIER JORDÁN ÁLVAREZ

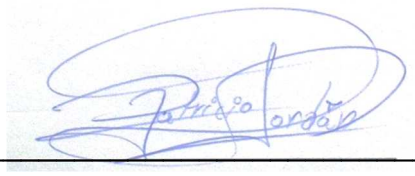
DECLARO QUE:

El proyecto de grado denominado “PROTOTIPO DE UN CONTROL REMOTO CON ACCESO VIRTUAL PARA PRUEBAS DE USABILIDAD DE APLICACIONES INTERACTIVAS DE TELEVISION DIGITAL TERRESTRE” ha sido desarrollado con base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas que constan al pie, de las páginas correspondientes, cuyas fuentes se incorporan en la bibliografía.

Consecuentemente este trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico del proyecto de grado en mención.

Sangolquí, 08 de mayo de 2015



Patricio Javier Jordán Álvarez

1804529004

UNIVERSIDAD DE LAS FUERZAS ARMADAS – ESPE

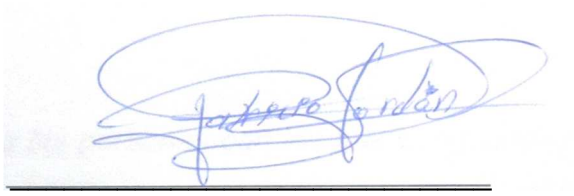
INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

Yo, Patricio Javier Jordán Álvarez

Autorizo a la Universidad de las Fuerzas Armadas- ESPE, la publicación, en la biblioteca virtual de la Institución del trabajo “PROTOTIPO DE UN CONTROL REMOTO CON ACCESO VIRTUAL PARA PRUEBAS DE USABILIDAD DE APLICACIONES INTERACTIVAS DE TELEVISION DIGITAL TERRESTRE”, cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y autoría.

Sangolquí, 08 de mayo de 2015



Patricio Javier Jordán Álvarez

1804529004

DEDICATORIA

A Dios, quien ha sido fuente de amor, y perdón, de quien recibí la sabiduría e inteligencia.

A mis queridos padres, quienes nunca me han negado su apoyo, su amor y comprensión, con el único afán de dejarme como herencia los valores y el respeto hacia los demás, para hacer de mí un hombre fiel a sus principios.

A mi reina, mi esposa, quien ha sido mi alegría desde el primer momento que la vi, mi compañera para toda la vida que cuida con amor y dedicación de nuestro hogar.

A mi hermosa y pequeña hijita, que me hizo conocer el amor en otra de sus dimensiones y es mi fuente de inspiración para seguir buscando y forjando una vida llena de éxitos para compartir con ella muchos juegos y carcajadas.

A mi hermana y hermanos, quienes a pesar de las adversidades han puesto en primer lugar nuestra familia.

Y finalmente a todas las personas con las que pude cultivar una amistad verdadera dentro de la ESPE, a aquellos que me han brindado sus malas noches, su confianza y gratitud y sus consejos.

AGRADECIMIENTO

En primer lugar quiero agradecer al Todo Poderoso por la vida y la salud, por su amor y su perdón, por su palabra entregada a mis padres para que sepan cómo guiarme por el buen camino, por regalarme a mis padres en quienes siempre he podido confiar, a quienes siempre he podido contar todos mis problemas, mis angustias y necesidades, por darme una compañera fiel y una hija hermosa.

En segundo lugar agradezco a mis padres y a mis suegros, por su vivo ejemplo de lucha constante, de entrega admirable y amor hacia sus hijos, de su ejemplo de matrimonio apoyándose en todo y juntos, alcanzar todos los sueños y metas propuestas.

Agradezco también a mi bella esposa, mi amiga, mi confidente, ella me enseñó a quererme primero, para poder amar a los demás, que despertó en mí, hermosos sentimientos de entrega sin esperar nada a cambio, que no se niega a darnos hijos porque es una mujer de gran fuerza y llena de bondad que me da su palabra inquebrantable y me ha hecho madurar.

También debo resaltar un sincero agradecimiento a los docentes que me han ayudado a que esta tesis sea una realidad palpable en beneficio de mi querida Universidad, a mi Director y Codirector, el Dr. Gonzalo Olmedo y el Ing. Freddy Acosta.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El grupo de investigación ESPE TV propuso la creación de una plataforma virtual para pruebas de usabilidad de aplicaciones interactivas y dentro de este proyecto macro, es necesario la construcción del prototipo de un control remoto IR para el STB con el estándar ISDB-Tb embebido, y por este motivo, el presente trabajo de tesis tiene como objetivo diseñar un prototipo de control remoto IR con acceso virtual. En primer lugar se levantarán los procesos de identificación de la codificación de señales IR, apoyándose de MatLab.® se decodificará la señal IR ingresada por el puerto in de audio, aprovechando la capacidad de muestreo y cuantificación de este puerto, segundo la generación o reproducción de las señales IR mediante Arduino™ UNO R3, luego el diseño del acceso virtual, con Netbeans IDE 8.0.1 y mediante HMTL5 presentar la página web con el control IR virtual, después la comunicación serial RS-232 de 9600 baudios en configuración 8N1, entre un servidor web XAMPP con interpretes para PHP y el hardware Arduino™ UNO R3, también la maquetación de los botones del control IR virtual con CSS3 y pixlr de [1]. Para las pruebas de comunicación se realizaran simulaciones en Proteus v8.0 con soporte para Arduino ™, se harán pruebas con el STB del laboratorio y además se revisaran parámetros de usabilidad como retardos, correspondencia, que sea intuitivo y manejable. Finalmente se escribirán las respectivas conclusiones y recomendaciones sobre el prototipo diseñado y la tecnología usada.

PALABRAS CLAVE:

- **ACCESO VIRTUAL.**
- **ARDUINO ™ Y HTML5.**
- **CONTROL REMOTO VIRTUAL.**
- **COMUNICACIÓN SERIAL CON ARDUINO ™.**
- **HTML5 CSS3 Y PHP.**

ABSTRACT

The TV ESPE research group proposed the creation of a virtual platform for usability testing of interactive applications and within this project macro, is necessary the building of a prototype of remote control for STB with embedded ISDB-Tb standard, and for this reason, this thesis is about to design a prototype of a virtual IR remote control access. First, the process of identifying the IR signal encoding will rise, based on MatLab.®, after that signal is entered by the audio port, the IR signal is decoded, taking advantage of this port skills, of sampling and quantization of signals; second, generation or reproduction of IR signals by Arduino™ UNO R3, then, the design of virtual access with Netbeans IDE 8.0.1 and by HMTL5 submitting the website of control virtual IR, after, RS-232 serial communication with a 9600 baud rate 8N1 configuration, between a web server XAMPP with PHP interpreters and Arduino™ UNO R3 hardware, also the layout of IR virtual control buttons with CSS3 and pixlr of (Autodesk, 2014). For communication tests, Proteus v8.0 simulations were carried out with support for Arduino ™, tests will be performed with the STB of the TV´s laboratory, test of usability and further parameters such as delays, correspondence, intuitive and manageable will be performed too. Finally the respective conclusions and recommendations of the prototype design and the technology used is written.

KEY WORDS:

- **ARDUINO ™ AND HTML5.**
- **HTML5 CSS3 AND PHP.**
- **SERIAL COMMUNICATION WITH ARDUINO ™.**
- **VIRTUAL ACCESS.**
- **VIRTUAL REMOTE CONTROL.**

PRÓLOGO

Como egresado de la Universidad de las Fuerzas Armadas ESPE, es mi deber e interés participar en la solución de proyectos de magnitud relevante para la sociedad Ecuatoriana y de la Región, ya que con este tema realizado se colabora para que diseñadores de aplicaciones interactivas que usen el middleware GINGA; es decir, diseñadores de todos los países que adoptaron el estándar ISDB-Tb, tengan disponible una plataforma para probar su aplicaciones en una transmisión en vivo, por lo tanto la necesidad de construir un prototipo de control remoto surgió como primer tema a tratar para dar solución a la interacción del acceso web con el hardware que controle el decodificador (con ISDB-Tb embebido), es así que me interese en participar con el desarrollo de esta solución, para ser parte de esta nueva tecnología de transmisión de televisión digital terrestre. Por tal motivo en mi trabajo de tesis expongo todas las herramientas, sus aplicaciones, el diseño, la simulación, y las pruebas del prototipo del control remoto y el acceso web enlazado al control.

Este es el primer paso a la cooperación de todos los países en desarrollar herramientas que permitan mejorar la calidad de las aplicaciones que pasarán a formar parte de la comunidad y de los pueblos.

Una gran ventaja de estudiar lenguajes de programación del lado del servidor, es que permanece segura la información, y si a esto le añadimos que el lenguaje de marcado HTML en su quinta revisión, añade etiquetas para elementos multimedia como audio y video, evitando el uso de complementos que antes eran necesarios para reproducir multimedia. El poder encargarme de la presentación de los componentes que forman una página web con CSS3 sin haber tenido conocimiento previo me comprometió a investigar y entender además de estos lenguajes de la web, que Internet es una herramienta muy valiosa si uno aprende a sacar provecho de ella, y así como existe software

libre existe también hardware libre como Arduino ™, que quiere decir que son dispositivos cuyas especificaciones, diseños y esquemas son de acceso público, esto no implica gratuidad sino flexibilidad para diseñar a partir de estos dispositivos, al igual que PHP hay clases diseñadas que son de libre acceso como la sugerida en este trabajo.

Finalmente lo que interesa es que el acceso virtual al prototipo de control remoto IR contenga valor agregado, que responda a parámetros de usabilidad, además queda entendido que se puede mejorar la plataforma a partir de la introducción que aquí presento, es decir que su programación sirva como guía para reproducir otras codificaciones IR y también implementar más utilidades de los lenguajes de programación aquí descritos, por ejemplo utilidades como acceso a bases de datos, envío de mensajes vía mail, ventanas de chat, inicio de sesión, etc. Se puede entonces pensar en temas a partir de estas aplicaciones.

GLOSARIO

C

CYTED Programa Iberoamericano de ciencia y tecnología para el desarrollo.

CSS3 Cascading Style Sheets 3.

COM De "*COMunications*", nombre asignado al puerto de comunicación serial.

F

FPS6038 T53 Receptor de señales infrarrojas de 3 pines.

H

HD High Definition.

H.264 Estándar de compresión de vídeo también conocido como MPEG-4 Part 10.

HTML5 HyperText Markup Language versión 5.

Hand-shake Término usado en comunicaciones para representar el acuerdo entre dispositivos para enviar y recibir información.

I

ISDB-Tb Integrated Services Digital Broadcasting – televisión brasileña.

ISDB-T Integrated Services Digital Broadcasting – estándar japonés.

IRLED led infrarrojo.

J

.js extensión de archivos JavaScript.

M

MySQL™ Sistema de gestión de base de datos relacional.

N

NEC Protocolo IR hecho por la compañía Nippon Electric Company.

P

PC Personal Computer.

PCM Pulse Code Modulation.

PHP Hypertext Preprocessor.

Perl Practical Extraction and Report Language.

Proteus v8.0 Simulador de circuitos electrónicos con soporte para Arduino ™.

R

RS-232 Recomended Standard-232, hace referencia también al puerto serial de comunicaciones.

S

STB set-top-box, equipo decodificador.

T

.ts Extensión de archivo transport stream, archivo que contiene paquetes de video, voz y datos.

TVDi Televisión Digital Interactiva.

V

VSPE Virtual Serial Port Emulator, Emulador de Puerto Serial Virtual.

W

W3C World Wide Web Consortium.

CAPÍTULO 1

INTRODUCCIÓN

1.1. ANTECEDENTES

La televisión en su evolución ha llegado a un nuevo formato de transmisión, el digital, y éste permite incluir dentro del carrusel de datos paquetes de Video, Audio y Datos que juntos conforman un archivo de extensión .ts (transport stream).

Ecuador adoptó oficialmente el estándar japonés-brasileño (ISDB-Tb) para la Televisión Digital Terrestre, el 26 de marzo de 2010.

Este nuevo estándar permite alcanzar grandes cambios en la forma de ver televisión, como son: la gran calidad de audio y video, la posibilidad de alta definición (HD), televisión portátil, movilidad y la interactividad. Cabe recalcar que el estándar fue adoptado por 11 países de América Latina, cuya capa física está basada en el estándar Japonés ISDB-T y como aporte Brasileño está la codificación de video H.264 y para interactividad el middleware Ginga, creado por el laboratorio de TELEMIDIA de la Pontificia Universidad Católica de Río de Janeiro y por el Laboratorio LAVID de la Universidad Federal de la Paraíba.

En La ESPE se ha trabajado desde el año 2009 en proyectos de interactividad para televisión digital basada en el middleware Ginga y hoy el grupo de investigación ESPE TV es parte de redes de colaboración a nivel internacional, a través del CYTED.

El grupo ESPE TV se encuentra desarrollando el proyecto de investigación “Plataforma de Análisis de Usabilidad para Aplicaciones Interactivas de Televisión Digital”, dentro del marco de colaboración de la Red temática en aplicaciones y usabilidad de la Televisión Digital Interactiva (REDAUTI).

La interactividad es un mercado de negocios nuevo, poco explotado en Ecuador, y explorado en los últimos años con muchos puntos de vista a nivel mundial, por una parte existen canales de televisión que no desean que el telespectador pierda su atención en el contenido de la programación y por otro lado, el deseo que la interactividad sea parte de la programación o incluso una herramienta para disminuir la brecha digital, donde se incluya información que hasta ahora solo puede ser vista y aprovechada a través de internet.

Es importante que las aplicaciones interactivas desarrolladas pasen por pruebas de calidad, tanto de programación, diseño y usabilidad. Se requiere verificar el comportamiento de las aplicaciones interactivas transmitidas por el aire, tanto en tiempo de transmisión, retardos, tiempo de procesamiento. Además hablando de los países en la Región no se cuenta con suficientes laboratorios de pruebas mucho menos laboratorios dedicados al tema de interactividad específicamente.

En el proyecto antes mencionado se presenta la necesidad de manipular un decodificador (STB) con el estándar ISDB-Tb embebido de forma remota para que un diseñador entrenándose en Ginga pueda tener acceso a la plataforma que pretende poner al alcance un escenario real de TVDi y le permita revisar el funcionamiento de su aplicación.

Para manejar el decodificador se debe reproducir las señales infrarrojas que genera el control remoto del decodificador y comunicar el circuito con la PC, vincular este a una página web de forma que se pueda enviar remotamente las instrucciones requeridas para las pruebas en tiempo real de las aplicaciones.

1.2. ALCANCE DEL PROYECTO

Se investigará la manera de capturar las señales IR del control remoto del decodificador con el estándar ISDB-Tb, luego se analizaran estas señales para decodificarlas y recrearlas mediante un hardware, se va a usar MatLab.® para dicho propósito ya que este software posee herramientas para comunicación serial, con circuitos electrónicos además que soporta varios formatos (extensiones de archivos).

Se programará en Arduino™ ya que también soporta comunicación SERIAL, para comunicarse con la PC y reproducir las señales decodificadas, es una alternativa con la que se puede dar solución a la temática planteada.

Para las pruebas se implementará con programación HTML5 más un soporte para comunicación serial con circuitos electrónicos, que servirá para la plataforma web, que será implementada junto con otros proyectos de grado.

Se implementara el prototipo de control remoto, que será capaz de sustituir al control original que viene con el decodificador, porque clonara las señales en el estándar interpretado y obtenido. En un inicio el análisis se hará con un solo decodificador y se dejara claro como reproducir señales IR de más estándares según el control del decodificador que se vaya a emplear en el proyecto general.

1.3. OBJETIVOS

1.3.1. General

Diseñar e implementar el prototipo de un control remoto con acceso virtual para la Plataforma de Análisis de Usabilidad de Aplicaciones Interactivas de Televisión Digital.

1.3.2. Específicos

Levantar los procesos de comunicación y recepción de las señales IR, vinculadas con la página Web para el decodificador de televisión digital terrestre.

Implementar el hardware interpretador de señales entrantes y codificador de señales IR de control para el STB.

Diseñar el portal Web que de soporte a los requerimientos de acceso virtual para el control remoto.

Analizar los resultados de las pruebas del control remoto con acceso virtual y el portal Web, de acuerdo a parámetros de usabilidad como, retardos, correspondencia, que sea intuitivo y manejable.

1.4. ORGANIZACIÓN DEL DOCUMENTO

En el Capítulo 2, se hace una introducción a todas las herramientas que serán utilizadas para el desarrollo del tema de tesis, primero se revisaran conceptos de lo que es una comunicación infrarroja, cómo se modula y sus tipos de codificación, una vez con este conocimiento, interpretar el protocolo de transmisión infrarroja NEC, asimismo una revisión de HTML5, CSS3 y PHP para el acceso web, con la respectiva comunicación RS-232 y la instalación de un servidor para que de soporte a la página web.

En el Capítulo 3, se diseña el prototipo del control remoto con acceso virtual, comenzando por el circuito que permite ingresar las señales IR del control original para su análisis y decodificación respectivo, después con las señales decodificadas se presenta el código en formato hexadecimal y uint8 para usar esta representación y generar mediante Arduino ™ la señal IR codificada en PCM con el estándar NEC, posteriormente se hace la

estructuración de la página web con HTML5 y CSS3 para el acceso virtual, además de la comunicación serial con el estándar RS-232 entre PHP y Arduino ™

En el Capítulo 4, se harán las simulaciones para probar la comunicación entre la placa Arduino ™ y la página web, entonces dentro de Proteus v8.0 se ubicará una terminal virtual y un osciloscopio para verificar la recepción correcta de los caracteres que se transmitirán desde el servidor, y para observar la señal codificada con NEC que Arduino ™ reproducirá a su salida, esto con el propósito de hacer las pruebas de correspondencia entre botón y señal codificada, después y una vez instalado en el servidor de ESPE TV se realizarán las pruebas finales de funcionamiento en el escenario donde surgió la necesidad de implementar este tema de tesis.

Finalmente el en Capítulo 5, se escribirán las respectivas conclusiones y recomendaciones del trabajo realizado.

CAPÍTULO 2

FUNDAMENTO TEÓRICO

2.1. CONCEPTOS DE COMUNICACIÓN INALÁMBRICA

Los dispositivos de control remoto infrarrojo son abundantes en el actual mundo lleno de gadgets. Desde la televisión, reproductores de video, una puerta de garaje, que se operan remotamente, un control remoto de una u otra forma nunca está lejos de nuestro alcance.

¿Por qué usar luz infrarroja para enviar señales de control remoto? Dos razones en particular se destacan, la primera es que los diodos usados para emitir luz infrarroja son poco costosos y altamente disponibles. La segunda es el hecho de que la luz infrarroja esté a una longitud de onda fuera del espectro de luz visible para el ojo humano, así podemos apuntar y disparar cualquier control remoto y no ser cegados en el proceso.

Entonces ¿cómo exactamente trabajan los controles infrarrojos? A nivel más básico, el control remoto contiene un circuito transmisor, parte del cual será un IRLED. Cuando un botón es presionado en el control, el comando es enviado como una señal IR al dispositivo el cual estas apuntando con el control. El dispositivo siendo controlado tendrá un circuito receptor, parte del cual será un fotodiodo para detectar la señal IR y convertirla en corriente eléctrica.

Esa es una visión muy simplista de comunicaciones IR, sin embargo, cuando se toma en cuenta en fondo el ruido infrarrojo emitido por otros objetos que generan calor y múltiples dispositivos a control remoto, situados en puntos muy cercanos entre sí, las cosas se vuelven notablemente más complicadas.

Con luz infrarroja simple, ahora hay posibilidades de que el comando no llegue al receptor en absoluto, y mucho menos el receptor en el dispositivo previsto [2].

2.2. MODULACIÓN Y MÉTODOS DE CODIFICACIÓN

Para asegurar que una señal IR transmitida alcance su destino correcto, o por el contrario el dispositivo de destino reciba solo la señal que está destinado a recibir, se utiliza modulación. Los sistemas de control remoto IR utilizan modulación por código de pulsos (PCM), donde la frecuencia portadora de modulación típica reside en el rango de 30kHz a 58kHz.

En términos de transmisión, la modulación significa prender y apagar el IRLED rápidamente en ráfagas de la frecuencia portadora. El receptor normalmente se sintonizará a esta frecuencia portadora, asegurando que solo recibe la señal requerida. A continuación utiliza esta frecuencia para demodular la señal.

Cuando el IRLED no emite luz, el transmisor está en el estado OFF, el cual en término de señales se conoce como un espacio. Durante la actividad del IRLED, cuando la luz es emitida en forma de impulsos a la frecuencia portadora, el transmisor está en estado ON, el cual es conocido como un pulso o marca. En el receptor un espacio se envía como un alto, mientras que una marca se emite como un bajo, ver Figura 2.1.

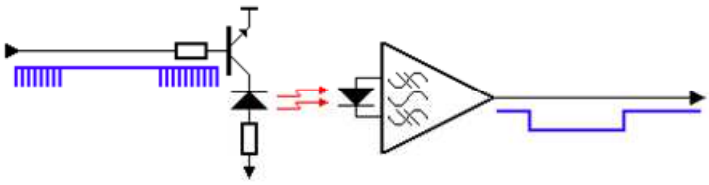


Figura 2.1: Transmisión de ráfagas IR y la recepción de ráfagas IR [3].

Estos espacios no son ‘0’s y las marcas no son ‘1’s del comando que está siendo transmitido, sin embargo el dato real a ser enviado desde el controlador se codifica con estos espacios y marcas. El método de codificación usado determina como se representan los unos o ceros en términos de las marcas y espacios. Los siguientes 3 métodos de codificación son típicamente usados en sistemas de control remoto IR [2].

2.2.1. Codificación por distancia de pulso

En este método de codificación, el largo del disparo del pulso (marcador) es siempre el mismo entre disparos consecutivos diferentes, dependiendo de si un cero lógico o uno lógico está siendo transmitido. El tiempo tomado para transmitir un ‘1’ lógico es más largo (es decir transmitir OFF por largo tiempo después del disparo IR) [2].

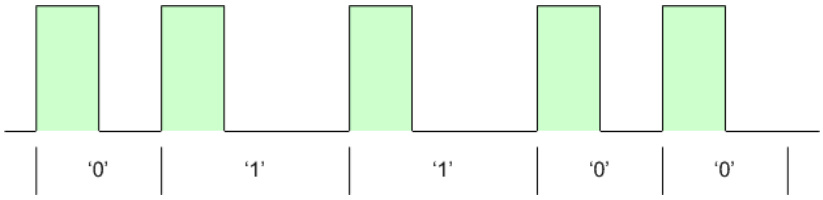


Figura 2.2: Codificación por distancia de pulso [4].

2.2.2. Codificación por longitud de pulso

En este método de codificación, la longitud de la ráfaga de impulsos (marca) es diferente para un '0' lógico y un lógico '1', con '1' lógico que requiere una ráfaga más larga [2].

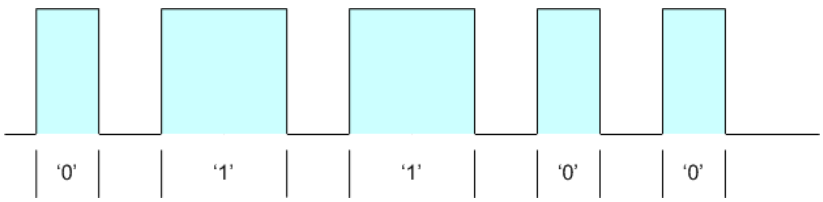


Figura 2.3: Codificación por longitud de pulso [4].

2.2.3. Codificación Manchester

En este método de codificación, todos los bits son de igual longitud, con la mitad del periodo del bit siendo una ráfaga de impulsos (marcas) y la otra mitad siendo un espacio. Un ‘0’ lógico está representado por una ráfaga en la primera mitad del periodo del bit y un espacio en la segunda, dando un periodo de transición medio de alto a bajo. Un ‘1’ lógico está representado por un espacio en la primera mitad del periodo del bit y una ráfaga en el segundo, dando un periodo de transición medio de bajo a alto [2].

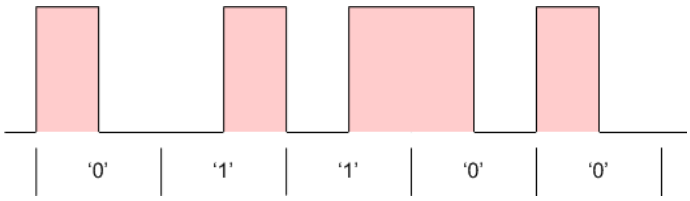


Figura 2.4: Codificación Manchester [4].

2.3. PROTOCOLO DE TRANSMISIÓN INFRARROJA NEC

El protocolo de transmisión infrarroja NEC usa codificación por distancia de pulsos de los bits de mensaje. Cada ráfaga de pulsos (marca - transmisión RC ON) es 562,5µs en longitud, a una frecuencia portadora de 38kHz (26,3µs). Bits lógicos son transmitidos de la siguiente manera:

- ‘0’ lógico – una ráfaga pulsos de 562,5µs seguido por un espacio de 562,5µs, con un tiempo de transmisión total de 1,125ms.
- ‘1’ lógico – una ráfaga pulsos de 562,5µs seguido por un espacio de 1.687,5ms, con un tiempo de transmisión total de 2,25ms.

Cuando un botón es pulsado en el control remoto, el mensaje transmitido consiste de lo siguiente, en orden:

- Una ráfaga de pulsos guía de 9ms (16 veces la longitud de la ráfaga de pulsos usada para un bit de dato lógico)
- Un espacio de 4,5ms.
- La dirección de 8 bits para el dispositivo receptor.
- Los 8 bits lógicos invertidos de la dirección.
- Los 8 bits comando.
- Los 8 bits lógicos invertidos del comando.
- Una ráfaga final de 562,5µs para indicar el final del mensaje de transmisión.

Los cuatro bytes de datos son enviados cada uno el bit menos significativo primero. La siguiente figura ilustra el formato de una trama de transmisión IR NEC, para una dirección de 00h (00000000b) y un comando de ADh (10101101b) [4].

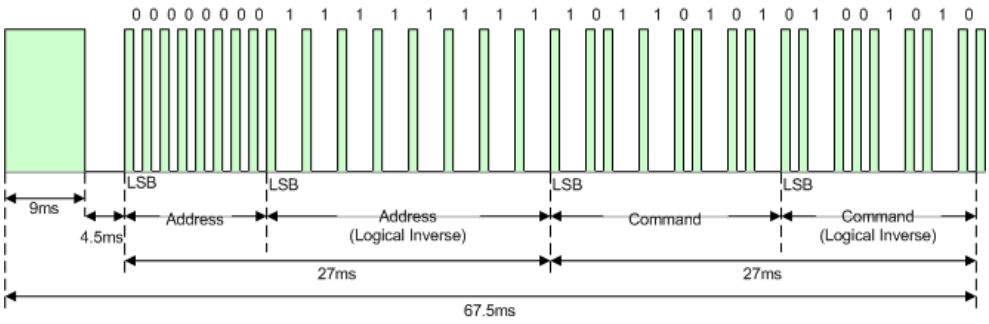


Figura 2.5: Trama completa usando el protocolo NEC [2].

Nótese en la Figura 2.5 que toma:

- 27ms para transmitir ambas partes de 16 bits para la dirección (dirección + su inverso) y 16 bits para el comando (comando + su inverso). Esto viene a ser para cada bloque de 16 bits finalmente contando ocho ‘0’s y ocho ‘1’s – dándonos (8*1,125ms) + (8*2,25ms).

- 67,5ms para transmitir completamente la trama del mensaje (sin tomar en cuenta los 562,5µs del pulso final para indicar que el mensaje acabó). [4]

2.3.1. Códigos repetidos

Si el botón en el control se mantiene presionado, un código repetido será enviado, normalmente cada 40ms después de la ráfaga de pulsos que significan el final del mensaje. Un código repetido será continuamente enviado en intervalos de 108ms, hasta que sea finalmente soltado [4]. El código repetido consiste de lo siguiente, en orden:

- Un pulso guía de 9ms.
- Un espacio de 2,25ms
- Una ráfaga de pulsos de 562,5µs para marcar el final del espacio (así como el final del código repetido).

La siguiente figura ilustra la transmisión de 2 códigos repetidos después de que una trama de mensaje inicial es enviada.

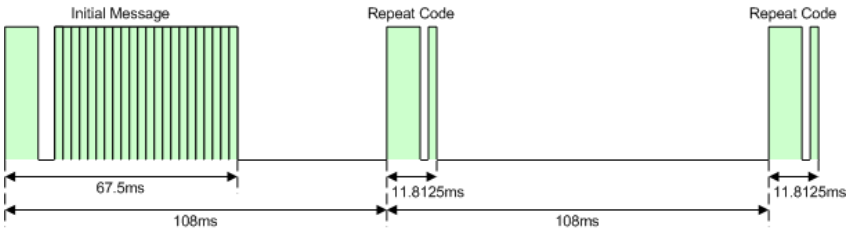


Figura 2.6: Códigos repetidos codificación NEC [2].

2.4. HTML5, CSS3 Y PHP

A continuación se describen los lenguajes que se utilizaran en el diseño, estilización y programación que se usaran para al diseño del acceso virtual, los 2 primero son lenguajes de marcado, entiéndase la descripción como un

lenguaje para que el explorador sepa dónde ubicar los elementos o componentes de una página web, por ejemplo títulos, párrafos botones, enlaces, elementos multimedia (audio/video) etc. El segundo es un lenguaje para dar formato a la apariencia de los componentes, por ejemplo tamaño, color, color de fondo, cambiar de apariencia al pasar el cursor del mouse sobre el elemento, etc. Y el tercero un lenguaje de programación para hacer de las páginas web, páginas más dinámicas capaces de hacer cálculos, subir archivos, abrir puertos de comunicación, etc.

2.4.1. ¿Qué es HTML5?

Es la quinta revisión del lenguaje de marcado HTML usado para estructurar y presentar el contenido de la web. Se trata de un sistema para formatear el *layout* (disposición) de nuestras páginas, así como hacer algunos ajustes a su aspecto, donde tenemos otras posibilidades para explotar más usando menos recursos, ya que esta revisión del lenguaje HTML añade nuevas etiquetas que nos ahorran el uso de otros productos que se usaban para complementar y hacer cosas como reproductores de audio y video. Con HTML5, los navegadores como Firefox, Chrome, Explorer, Safari y más pueden saber cómo mostrar una determinada página web, saber dónde están los elementos, dónde poner las imágenes, dónde ubicar el texto, aunque siga sonando parecido a sus predecesores, la diferencia principal, sin embargo, es el nivel de sofisticación del código que podremos construir usando HTML5 [5].

2.4.1.1. Nuevas etiquetas

Las más importantes de las nuevas etiquetas creadas son:

article: esta etiqueta sirve para definir un artículo, un comentario de usuario o una publicación independiente dentro del sitio.

header, footer: estas etiquetas individuales ahorran tener que insertar IDs para cada uno, como se solía hacer anteriormente. Además, se pueden insertar headers y footers para cada sección, en lugar de tener que hacerlo únicamente en general.

nav: la negación puede ser insertada directamente en el markup, entre estas etiquetas, que nos permitirán hacer que nuestras listas oficien de navegación.

section: con esta etiqueta, una de las más importantes de las novedades, se puede definir todo tipo de secciones dentro de un documento. Por ponerlo de forma sencilla, funciona de una forma similar a la etiqueta div que nos separa también diferentes secciones.

audio y video: estas son las dos más importantes etiquetas de HTML5, dado que nos permiten acceder de forma más simple a contenido multimedia que puede ser reproducido por casi todo tipo de dispositivos; marcan el tipo de contenido que estará en su interior.

embed: con esta etiqueta se puede marcar la presencia de un contenido interactivo o aplicación externa.

canvas: finalmente, esta etiqueta nos permite introducir un “lienzo” dentro de un documento, para poder dibujar gráficos por vectores; será necesario el uso de JavaScript.

2.4.2. ¿Qué es CSS3?

Hojas de estilo en cascada (Cascading Style Sheets) es la tecnología desarrollada por el World Wide Web Consortium (W3C) con el fin de separar la estructura de la presentación, dando facilidad y rapidez a la hora de hacer cambios estéticos dentro de una página, así evitamos hacer a los archivos

demasiado pesados (excluyendo el largo código requerido para las tablas anidadas y el añadido de características gráficas), y definimos el "estilo visual" de un sitio entero sin necesidad de hacerlo etiqueta por etiqueta, para cada una de las páginas, y por otro lado, trabajamos con estándares, y separamos hasta cierto punto la estructura (vale decir, el código) de la presentación, logrando una manera más nítida de trabajar, y lo que es más: en un sencillo documento CSS, definimos lo que bien podría llamarse una "plantilla gráfica" para todo un sitio. Vale decir, que cualquier cambio hecho a un estilo CSS, se reflejará en todos los elementos que sean referidos a éste, automáticamente, con sólo editar un sencillo documento CSS [6].

En la Tabla 2.1 hay algunos de los selectores añadidos en CSS3, útiles para “apuntar” rápidamente sin necesidad de señalar en nivel de profundidad del elemento HTML.

Tabla 2.1.

Selectores añadidos en CSS3 [7].

Selector CSS3/ejemplo	Explicación
element1 ~ element2 p ~ ul	Selecciona todo elemento que este precedido por un elemento <p>
[attribute^=value] a[href^="https"]	Selecciona todo elemento <a> el cual su atributo href comienza con "https"
[attribute*=value] a[href*="w3schools"]	Selecciona todo elemento <a> el cual el valor de su atributo href contiene el substring "w3schools"
:first-of-type p:first-of-type	Selecciona todo elemento <p> que es el primer elemento <p> de su padre
:in-range input:in-range	Selecciona los elementos de entrada con un valor dentro de un rango específico
:last-child p:last-child	Selecciona todo elemento <p> que es el último hijo de su padre
:last-of-type p:last-of-type	Selecciona todo elemento <p> que es el último elemento <p> de su padre

Selector CSS3/ejemplo	Explicación
:not(selector) :not(p)	Selecciona todo elemento que no es un elemento <p>
:nth-child(n) p:nth-child(2)	Selecciona todo elemento <p> que es el segundo hijo de su padre
:nth-last-child(n) p:nth-last-child(2)	Selecciona todo elemento <p> que es el segundo hijo de su padre, contando desde el ultimo hijo
:in-range input:in-range	Selecciona elementos de entrada con un valor dentro de un rango especifico
:out-of-range input:out-of-range	Selecciona elementos de entrada con un valor fuera del rango especifico

2.4.3. ¿Qué es PHP?

PHP (Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML.

Ejemplo:

```
   <!DOCTYPE HTML>
<html>
    <head>
        <title>Ejemplo</title>
    </head>
    <body>
        <?php
            echo "¡Hola, soy un script de PHP!";
        ?>
    </body>
</html>
```

En lugar de usar muchos comandos para mostrar HTML (como en C o en Perl), las páginas de PHP contienen HTML con código incrustado que “hace algo" (en este caso, mostrar "¡Hola, soy un script de PHP!”). El código de PHP está encerrado entre las etiquetas especiales de comienzo y final “<?php” y “?>” que permiten entrar y salir del "modo PHP" [8].

Lo que distingue a PHP de algo del lado del cliente como JavaScript es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente. El cliente recibirá el resultado de ejecutar el script, aunque no se sabrá el código subyacente que era. El servidor web puede ser configurado incluso para que procese todos los ficheros HTML con PHP, por lo que no hay manera de que los usuarios puedan saber qué se tiene debajo de la manga.

Entre sus muchas características y aplicaciones PHP tiene soporte para comunicarse con servicios que usen protocolos para COM en Windows, corre sobre la mayoría de sistemas operativos más conocidos, trabaja con soporte a base de datos e incluso utiliza programación orientada a objetos.

2.5. COMUNICACIÓN SERIAL RS-232

Cuando hablamos de comunicación por códigos Morse estamos tratando con una comunicación tipo serial, ya que los datos o la información enviada se la realiza en forma secuencial, es decir una letra tras otra, lo mismo que la norma RS-232 un bit tras otro.

El protocolo RS-232 es una norma o estándar mundial que rige los parámetros de uno de los modos de comunicación serial. Por medio de este protocolo se estandarizan las velocidades de transferencia de datos, la forma de control que utiliza dicha transferencia, los niveles de voltajes utilizados, el tipo de cable permitido, las distancias entre equipos, los conectores, etc.

Además de las líneas de transmisión (Tx) y recepción (Rx), las comunicaciones seriales poseen otras líneas de control de flujo (Hand-shake), donde su uso es opcional dependiendo del dispositivo a conectar.

A nivel de software, la configuración principal que se debe dar a una conexión a través de puertos seriales. RS-232 es básicamente la selección de la velocidad en baudios (bits por segundo) que va de 300 a 115200 (9600 las más usada), la verificación de datos o paridad (paridad par o paridad impar o sin paridad), los bits de parada luego de cada dato (1 o 2), y la cantidad de bits por dato (7 u 8), que se utiliza para cada símbolo o carácter enviado.

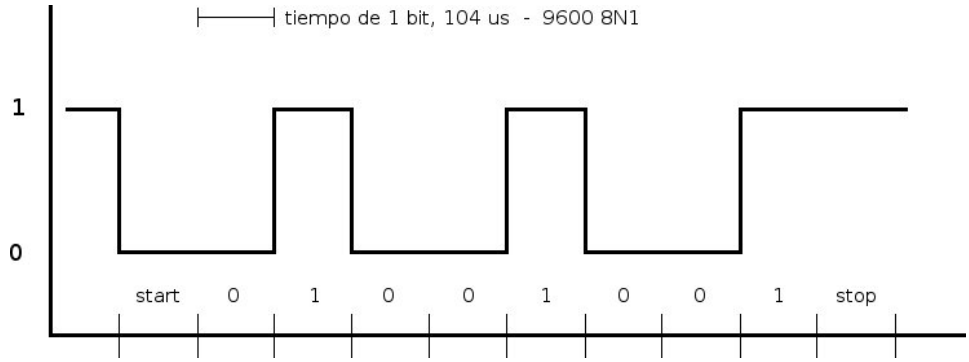


Figura 2.6. Ejemplo de una transmisión con comunicación serial [9].

2.6. INSTALACION DE XAMPP

La instalación del panel de control XAMPP, dará el soporte necesario para levantar el servidor y además al poseer intérpretes de PHP será muy útil para poder correr sobre la página diseñada para el acceso virtual del control remoto IR virtual. Por eso, aquí hay unos tips para la instalación de esta herramienta, la versión aquí instalada es la *xampp-win32-1.8.3-5-VC11-installer.exe*, vale aclarar que al ejecutar la aplicación se puede levantar las páginas web en la red local y cualquier PC dentro de la red podrá abrir la página insertando la dirección IP de la máquina donde este “corriendo” el servidor.

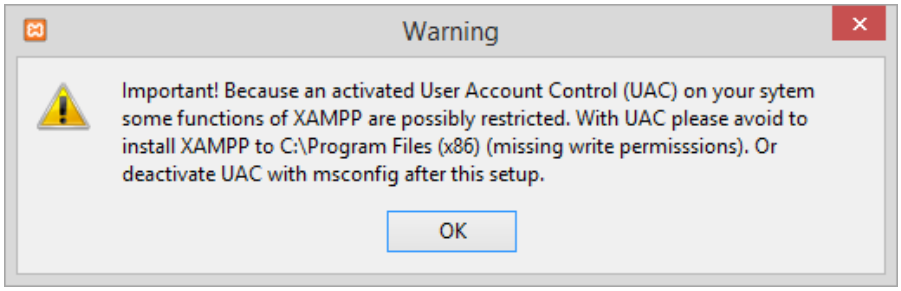


Figura 2.7: Advertencia durante la instalación de XAMPP.

Si al iniciar la instalación se recibe este mensaje, ver Figura 2.7, quiere decir que las carpetas están protegidas contra escritura, así que se debe crear un directorio para uso exclusivo de XAMPP, y dar permisos de lectura y escritura a la carpeta que lo va a contener. Por ejemplo en este trabajo se presenta la opción de cargar archivos desde el cliente, y PHP necesita permisos de escritura para pegar los archivos enviados dentro del servidor.

3023.

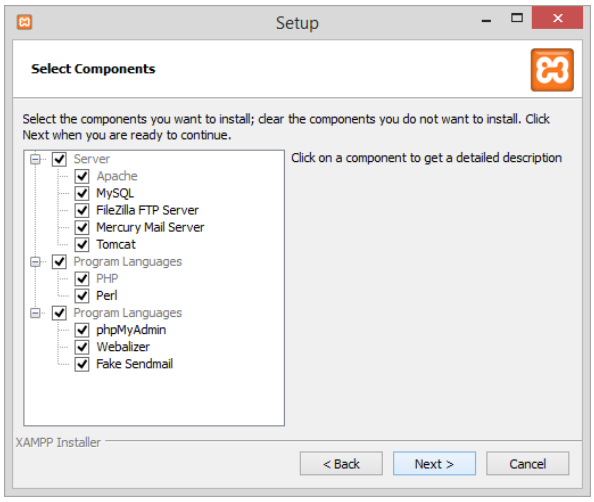


Figura 2.8: Ventana de componentes a instalarse en XAMPP.

Si observamos la Figura 2.8, veremos los componentes que trae el panel de control XAMPP entre los que se distinguen servidores y lenguajes de programación, incluso MySQL™ que es una base de datos que podría ser usada para registrar a los usuario que necesiten utilizar la plataforma de pruebas de aplicaciones interactivas.

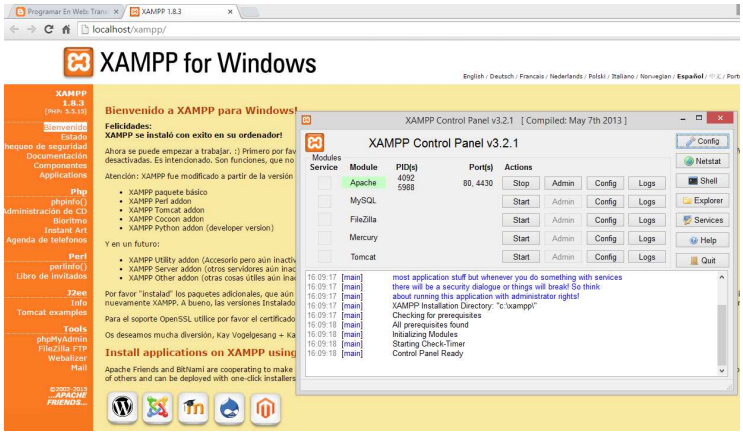


Figura 2.9: Panel de control XAMPP.

CAPÍTULO 3

DISEÑO DEL PROTOTIPO DE CONTROL REMOTO, ACCESO VIRTUAL Y LA COMUNICACIÓN SERIAL ENTRE EL SERVIDOR Y EL HARDWARE

El diseño de este prototipo se enfoca en la programación en Arduino™ para soporte de comunicaciones IR, RS-232, y en la virtualización del control remoto (HTML5 y CSS3), para su acceso mediante cualquier navegador web, de manera que del lado del servidor será necesario implementar las características más funcionales de PHP (comunicación con puertos COM).

3.1. CAPTURA Y ANALISIS DE LAS SEÑALES IR

La captura se hará mediante el ingreso de señales analógicas por el puerto del micrófono y mediante MatLab.® se analizará y decodificará las señales. Como sabemos una señal IR es generada con ráfagas de 38kHz, pero el receptor para este tipo de señales FPS6038 T53 a su salida devuelve 5V+ si no recibe la ráfaga de 38kHz y 0V+ cuando recibe dicha ráfaga [3], entonces, fue necesario invertir la señal después del receptor IR, para ello se utilizó un transistor 2N3906 (pnp), mismo que con la excitación del receptor se logra invertir dicha señal. Tomando en cuenta que los pulsos más pequeños de la señal recibida son 562,5µs es suficiente muestrear con MatLab.® a 8000 muestras por segundo o 125µs (5.22 veces más pequeño que la señal recibida). Se puede usar el comando wavrecord o audiorecorder, para recibir la señal en MatLab.®. La Figura 3.1 muestra el circuito usado para recibir las señales IR provenientes del control remoto.

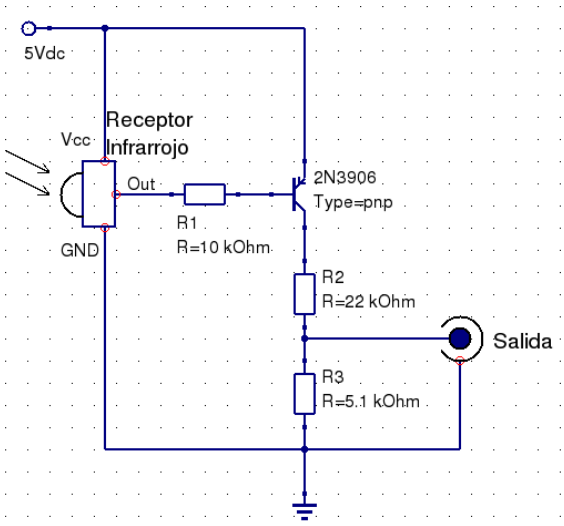


Figura 3.1: Receptor IR para puerto de audio in de una laptop [10].

El circuito receptor se alimenta con 5 voltios y el juego de resistencias en el mismo, limita la corriente que ingresa a la pc, para facilidad de manipulación se puede reemplazar R2 y R3 por un potenciómetro, ya que en ocasiones R3 necesita consumir mayor voltaje, para que se pueda apreciar de manera correcta la señal recibida en MatLab.®.

La Figura 3.2 representa la señal IR que entra por el puerto de audio in (micrófono) usando MatLab.®, con 2 segundos de grabación a una frecuencia de 8000 muestras por segundo, en la que podemos ver que contiene 3 pulsos, de los cuales, el primer pulso viene junto con la información del código IR para el botón pulsado en el control remoto, los 2 pulsos consecutivos indican que el botón se mantiene pulsado y mientras lo haga se repetirán cada 108ms según lo indica la Figura 2.6 que es el uso correcto para códigos repetidos.

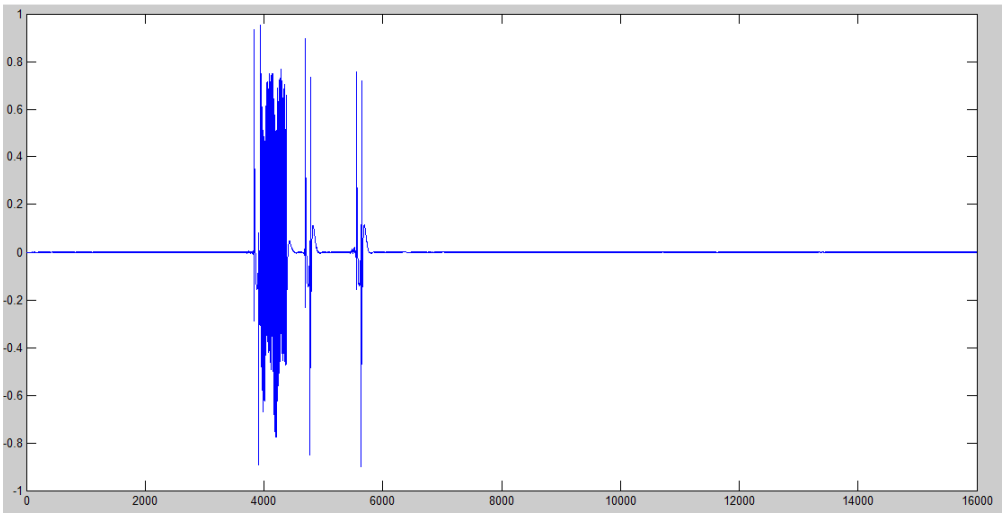


Figura 3.2: Señal recibida por el puerto de audio in usando MatLab.®

Si Ampliamos el primer pulso junto con la información del botón presionado, veremos que a la señal le hace falta tratamiento para verse como pulsos rectangulares y más entendible, como lo indica la Figura 3.3, para corregir esto se hizo un programa en MatLab.® en el cual se carga la señal recibida.

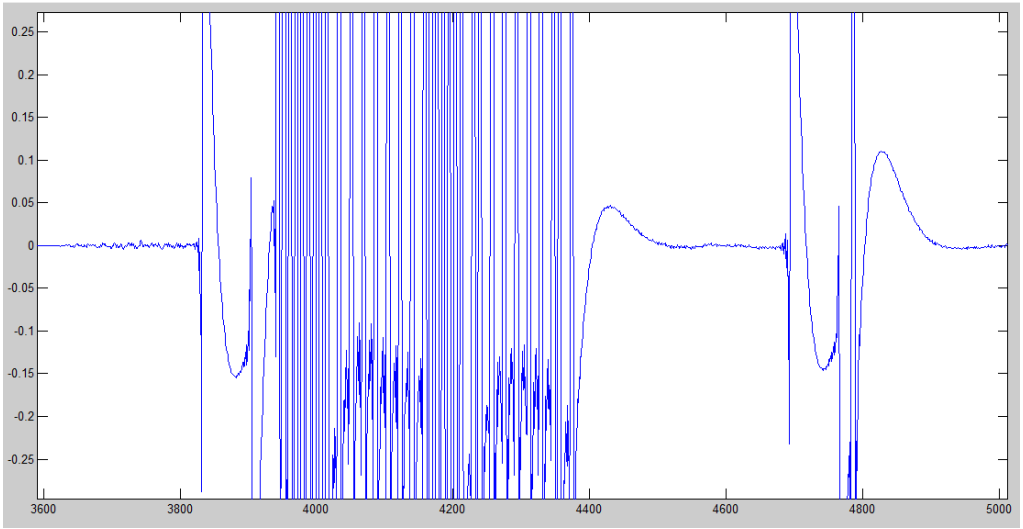
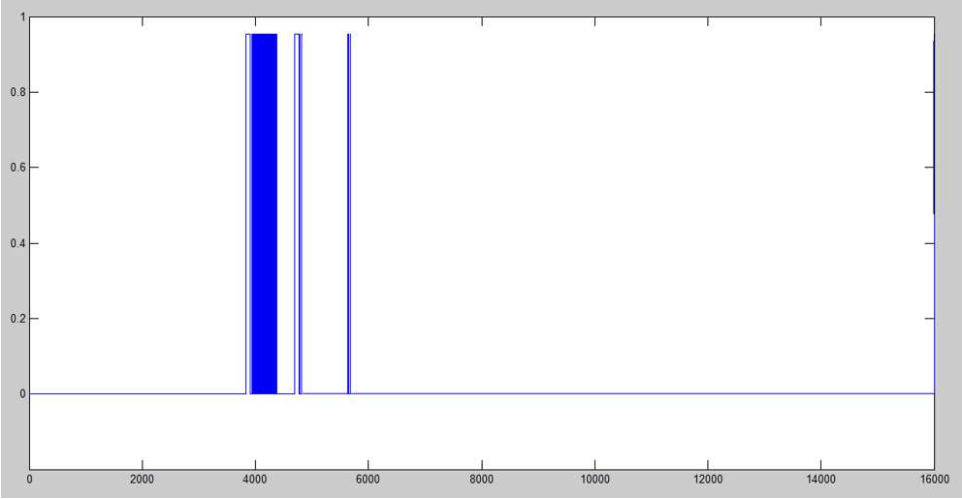


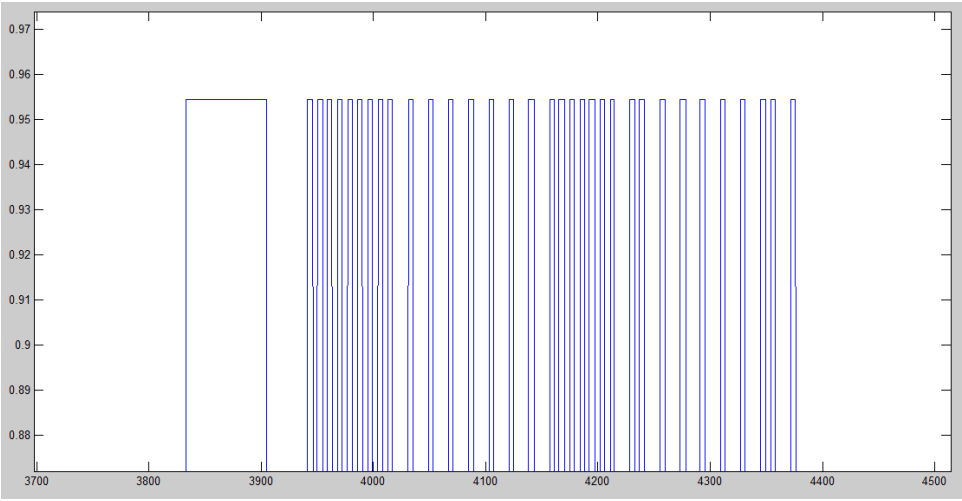
Figura 3.3: Ampliación de la señal IR recibida en MatLab.®

Debido al uso de comandos de MatLab.® para tratamiento de esta señal, y si nos fijamos en la Figura 3.3 la señal está comprendida entre 1 y -1, y que el primer pulso guía de 9ms tiene distorsión hacia valores negativos, se

programó para que compruebe intervalos entre valores positivos y negativos, para de esta manera corregir la señal de entrada, en la Figura 3.4, se puede revisar la corrección de la señal recibida.



(a)



(b)

Figura 3.4: a) señal recibida después de la corrección, b) primer pulso con la información del botón.

Una vez que se consigue la señal “rectangular” necesitamos decodificar la información de cada botón presionado. MatLab.® permite continuar el tratamiento de esta señal. Primero se discrimina el pulso guía para luego identificar los bits con espacios largos como ‘1’s y los bits de con espacios

cortos como ‘0’s, de esta manera conseguiremos en un momento dado agrupar estos bits en grupos de 4 y mostrar el código hexadecimal que representan, ya que el código hexadecimal se representan los códigos de un control remoto IR. MatLab.® dispone de funciones para el cambio del tipo de dato como son: *dec2base*, *bin2dec*, *dec2hex* y aprovechando esta funcionalidad, se decodifica la información de los botones, la Figura 3.6 muestra el código del botón rojo en formato hexadecimal y la Tabla 3.1 contiene los códigos de los botones de interés para la plataforma de pruebas de aplicaciones interactivas de televisión digital terrestre.

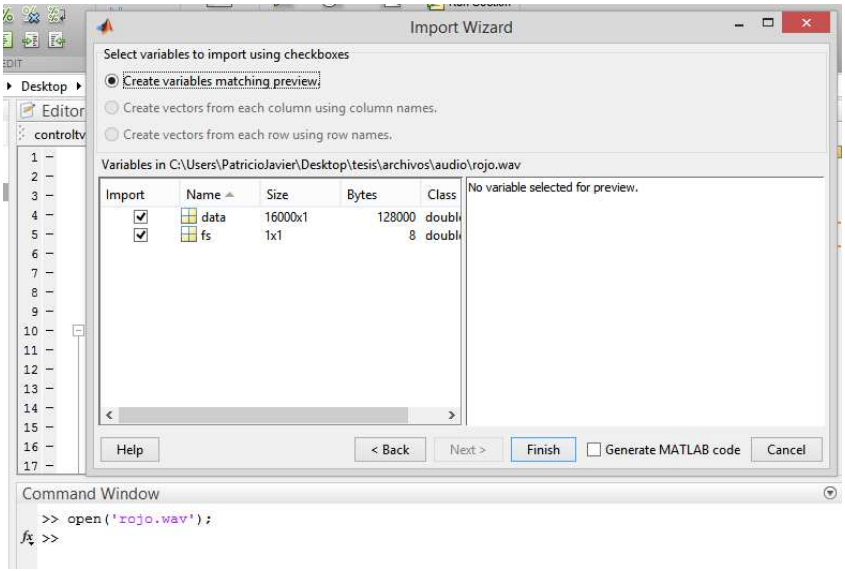


Figura 3.5: Abrir archivo .wav con que contiene la información del botón en pulsos rectangulares.

Luego de abrir el archivo .wav que contiene la información del botón en pulsos rectangulares semejantes a los de la Figura 3.4.a, se puede correr el decodificador. La Figura 3.5 indica los pasos a seguir para abrir un archivo de extensión .wav este comando carga los datos en la variable *data*. En la Figura 3.6 se puede ver que el programa *decodificador.m* solo captura datos hasta acabar los 32 pulsos correspondientes a la información del botón pulsado. Una vez completa los 32bits, termina de contar los demás pulsos, excluyendo

incluso los códigos repetidos que aparecen después del primer pulso guía y de la información del botón.

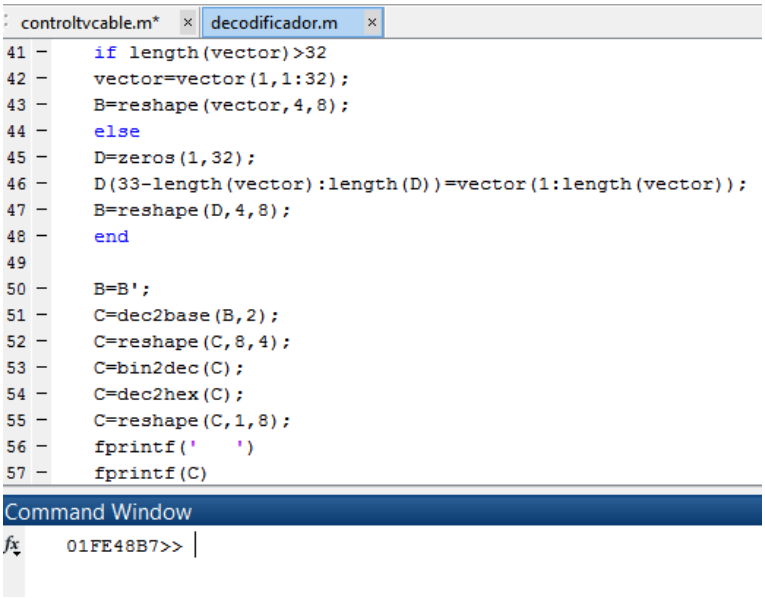


Figura 3.6: Código del botón rojo en formato hexadecimal.

Tabla 3.1.

Botones necesarios para "clonar" el control remoto, con su respectivo código en formato hexadecimal.

Botón	Código hex	Código uint8
ROJO	01FE48B7	1,254,72,183
VERDE	01FE708F	1,254,112,143
AMARILLO	01FEB847	1,254,184,71
AZUL	01FE58A7	1,254,88,167
UP	01FE30CF	1,254,48,207
DOWN	01FE8877	1,254,136,119
LEFT	01FE9867	1,254,152,103
RIGHT	01FE08F7	1,254,8,247
OK	01FEC837	1,254,200,55
EXIT	01FEF00F	1,254,240,15
MENU	01FE7887	1,254,120,135
VOL+	01FE629D	1,254,98,157
VOL-	01FEE21D	1,254,226,29
CH+	01FE12ED	1,254,18,237
CH-	01FE926D	1,254,146,109
INFO	01FE10EF	1,254,16,239
TEXT	01FEE817	1,254, 232, 23

En la Figura 2.2 también se representa los códigos en formato *uint8*, que serán útiles para ser usados en la generación de señales IR en Arduino™.

3.2. REPRODUCCIÓN DE LAS SEÑALES IR MEDIANTE ARDUINO™

Una vez que se consigue los códigos en hexadecimal, dentro de Arduino™ se programa de manera que se pueda disponer de la misma tabla para la transmisión de códigos IR, a la frecuencia indicada por el protocolo NEC. La Tabla 3.1 también contiene los códigos en formato *uint8* para cada par de números en hexadecimal, es un muy buen ejemplo el programa en Arduino™ hecho por [11] y con unas modificaciones y adaptaciones a ese ejemplo, se puede lograr reproducir las mismas señales de control para televisión digital.

A continuación la Figura 3.7 muestra la subrutina *on1* del código en Arduino™ tomado de la fuente [11] y modificado para que responda a la codificación infrarroja NEC, donde:

- period=562,5µs/26,316µs=21,375 (se redondea a 21 para saber cuántos pedacitos de 1/38kHz entran en los 562,5µs).
- wait_time=21,375/2=10,6875 (se redondea a 11 para que la ráfaga emita pulsos de 26,317µs).
- wait_time1=1687 (tiempo que falta para el pulso largo).

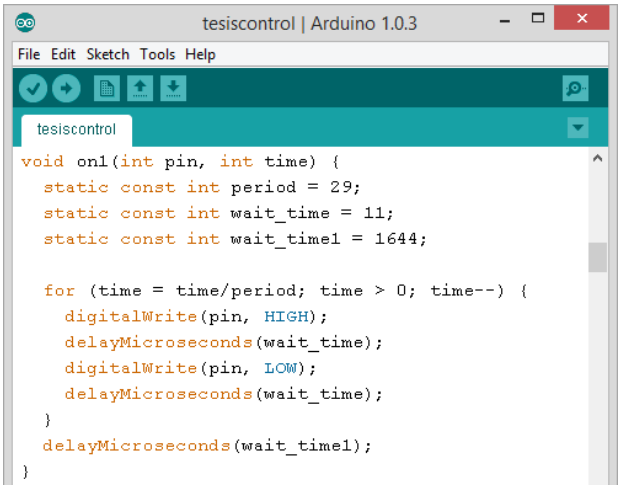


Figura 3.7: Subrutina *on1* para pulso largo modificada de [11].

La Figura 3.8 muestra la subrutina *on0* tomada del mismo código de la fuente [11] y modificado para NEC, donde:

- period=562,5µs/26,316µs=21,375 (se redondea a 21 para saber cuántos pedacitos de 1/38kHz entran en los 562,5µs).
- wait_time=21,375/2=10,6875 (se redondea a 11 para que la ráfaga emita pulsos de 26,317µs).
- wait_time1=562 (tiempo que falta para el pulso corto).

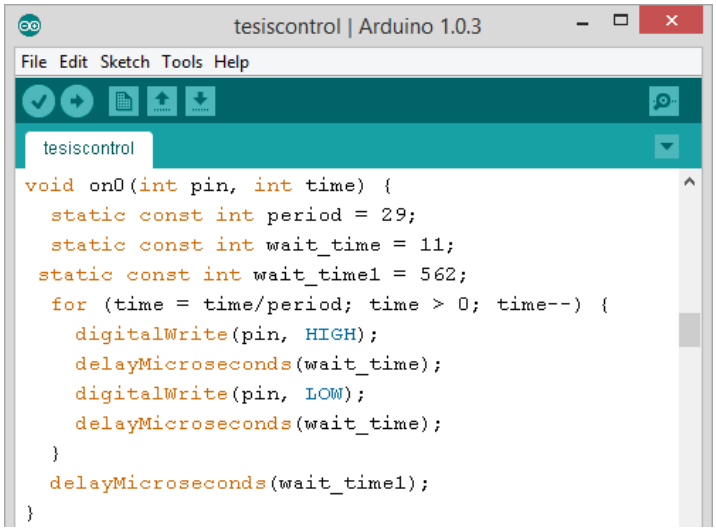


Figura 3.8: Subrutina *on0* para pulso corto, modificada de [11].

La siguiente subrutina mostrada en la Figura 3.9, envía un pulso largo o un pulso corto según el valor del bit dentro de la variable *bits,* si es 1 ejecuta *on1* y si es 0 ejecuta *on0*. La variable PULSE_WIDTH es igual a 562,5µs.

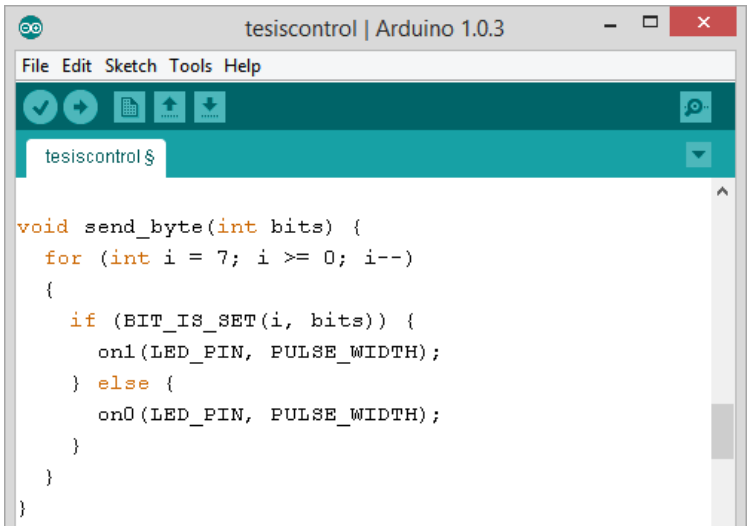


Figura 3.9: Subrutina *send_byte* que envía pulsos largos o cortos [11].

En la Figura 3.10 está la subrutina que envía cada uno de los 4 bytes del comando de cada botón hacia la subrutina send_byte, además de un último pulso corto que representa el final de la trama NEC.

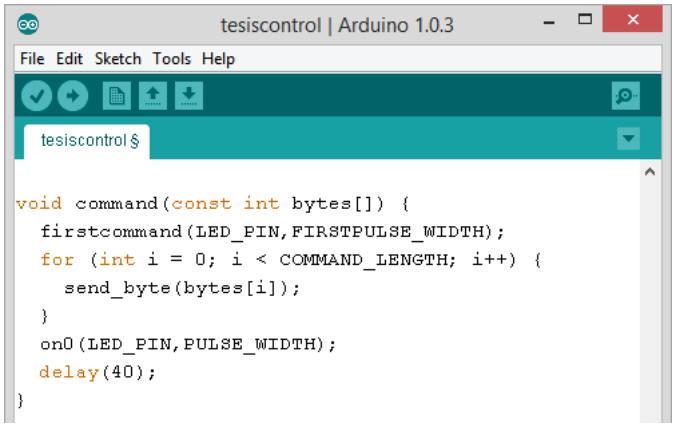


Figura 3.10: Subrutina *command*, envía uno a uno los 4 bytes [11].

La subrutina de la Figura 3.11 transmite el pulso guía necesario para que el dispositivo identifique, que lo que le sigue a este pulso guía es el comando de un botón presionado. Donde el wait_time1=4.5ms.

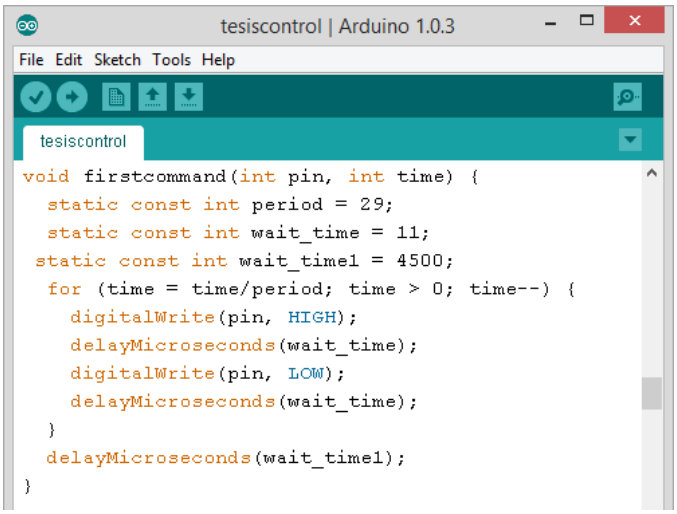


Figura 3.11: Subrutina para el pulso guía [11].

Con todas estas subrutinas queda completa en la generación del protocolo NEC para transmisión de señales IR, en las siguientes secciones se tratará sobre el acceso virtual construyendo PHP sobre HTML5 y la comunicación de este acceso virtual con la placa Arduino™ UNO R3 mediante RS-232.

3.3. COMUNICACIÓN SERIAL ENTRE EL ACCESO VIRTUAL Y EL HARDWARE

La comunicación entre el servidor y el circuito debe garantizar que los bits de información correspondientes a los botones cliqueados en la página web, sean correctamente decodificados en el hardware e interpretar su información para reproducir la señal IR que se le haya asignado. Primero hay que identificar el elemento cliqueado en la página web, enviar esta información a PHP, donde según el elemento cliqueado se enviará una variable tipo carácter (char), hacia el puerto COM3, luego el hardware recibirá una cadena de 8 bits, que será decodificada para saber a qué “botón” se le hizo clic.

Se decidió enviar un solo carácter para identificar a los botones; porque, agiliza la comunicación entre el servidor web y el hardware, ya que si se intentaba enviar el código IR en formato hexadecimal, por un lado demoraba la comunicación con el circuito, por la razón que la clase *php_serial.class.php* se trata de una versión free, y permite un número limitado de letras del alfabeto. Como Arduino™ es una herramienta programable, tiene la capacidad de generar las señales IR a ráfagas como se aprendió en el anterior capítulo.

La Tabla 3.2 contiene la asignación de los caracteres para la representación de los botones tanto en HTML5 como en PHP y la tarjeta Arduino ™.

Tabla 3.2.

Asignación de números y caracteres para la comunicación serial.

Botón	HTML5 *value*	PHP/Arduino ™
ROJO	1	K
VERDE	2	N
AMARILLO	3	3
AZUL	4	U
UP	5	5
DOWN	6	6
LEFT	7	P
RIGHT	8	M
OK	9	9
EXIT	10	A
MENU	11	B
VOL+	12	V
VOL-	13	D
CH+	14	Y
CH-	15	Z
INFO	16	G
TEXT	17	H

3.4. ACCESO VIRTUAL MEDIANTE HTML5

Es conocido que para el acceso virtual se utilizará Internet, y para ello HTML5 ayuda con el diseño de una página web, además la solución al problema sugiere que la página web este ligada y se comunique correctamente con la tarjeta Arduino ™, de manera que se torna necesario que en la pc esté puesto en marcha un servidor, para montar la página web y para que de éste lado de la comunicación (lado del servidor) pueda correr PHP que da el soporte a dicha comunicación entre acceso virtual y tarjeta Arduino ™. Adicionalmente, la página debe tener un diseño amigable con el usuario final, también debe ser intuitiva, fácil de usar, y tener correspondencia, es decir, que los botones virtuales cumplan su función correctamente.

3.4.1. Herramientas y Componentes para Montaje del Servidor

Lo que a continuación se describe son herramientas de software necesarias para el soporte al servidor y la comunicación con la tarjeta Arduino™.

XAMPP: Es un servidor distribuido como software libre, es multiplataforma y está compuesto de: un servidor web (Apache) con los intérpretes para lenguajes PHP y Perl, y de una base de datos MySQL™. Su siglas significan X de multiplataforma, A de Apache, M de MySQL™, P de PHP, y P de Perl.

Netbeans 8.0.1: Es el software para programación en java; pero soporta HTML5, servidor Apache, e intérpretes para PHP, así que viene bien utilizarlo.

HTML5: Es el lenguaje de marcado con el que escribiremos la página web principal.

CSS3: Hojas estilo cascada es otro lenguaje para dar estilo a los componentes estructurados dentro de HTML5. También viene bien usarlo

junto con HTML5 ya que permite mucha más versatilidad al momento de apuntar dichos componentes.

Pixlr: es una aplicación online hecha por [1] con la que se manipularon las imágenes para hacerlas de fondo transparente.

PHP: lenguaje de programación de propósito general, capaz de ser embebido dentro de una página web, y correr sus líneas de código del lado del servidor, para de esta manera aprovechar muchas de sus funciones y características.

php_serial.class.php: es una clase creada por [12] para comunicaciones con el puerto COM o puerto serial de una pc.

3.5. ESTRUCTURA DE LA PÁGINA CON HTML5

La página web tiene una sección principal denominada *wrapper* (envoltura), y 3 secciones dentro de esta, llamadas *header*, *controlframe*, y *content*, y están básicamente distribuidas como se muestra en la Figura 3.12

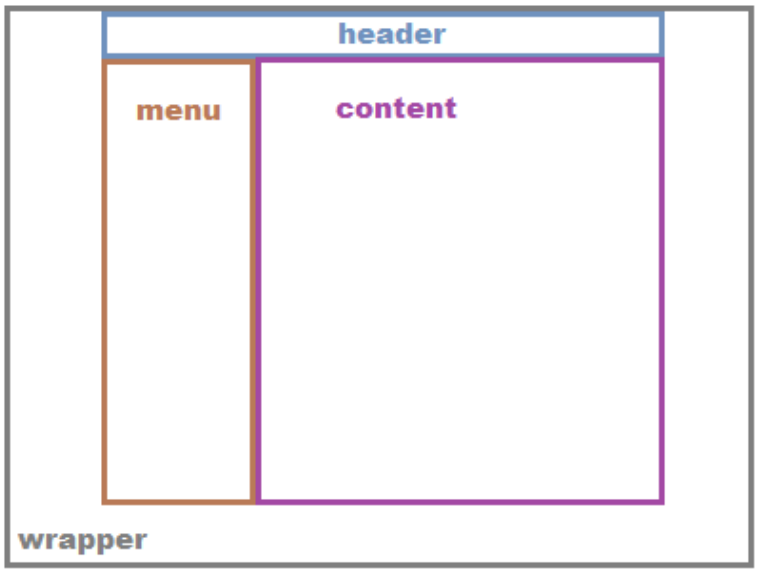


Figura 3.12: Distribución de *divs* de la página web.

El control remoto virtual ira ubicado en la sección controlframe, dónde colocaremos primero un iframe que contendrá a su vez la página web del control remoto virtual *index2.php*. En la sección header es para colocar un menú de opciones: descargas, contacto, redes sociales. La sección content es para el reproductor de video. Esta primera página web, pretende quedar como guía para la plataforma de pruebas final.

Una de las ventajas o facilidades que brinda la escritura de páginas web apoyadas con la ayuda de CSS es la separación de la estructura (distribución de componentes en un navegador) de la página, de su presentación (apariencia de los componentes).

Con el propósito de dejar en claro la potencialidad del lenguaje de marcado HTML5, se ubicó un reproductor de video en la sección *content*, ya que HtML5 incluye en sus líneas de instrucción etiquetas multimedia evitando de esta manera la necesidad de instalar componentes extras que antes solo con estos se podía reproducir multimedia como Flash Player.

3.5.1. Distribución y Diseño de los Botones

Para distribuir los botones del control virtual se ocupó el espacio delimitado por un inline frame, el diseño tuvo que ser dentro de este marco para evitar que al hacer clic en cualquier botón, éste recargue la página entera, reseteando el video a cero segundos de reproducción, la solución fue incluir dentro del iframe todos los botones del control, para que al hacer clic en cualquier botón recargue únicamente la página insertada en el iframe, y así mantener al video en reproducción continua.

Para conseguir una mejor estilización de los botones se copiaron imágenes prediseñadas para cada botón, y se modificaron para tener fondo transparente con la aplicación pixlr online de la empresa [1], de manera que

el botón consiga formas diferentes a rectangulares, la siguiente figura deja ver la aplicación pixlr con una de las imágenes de los botones.

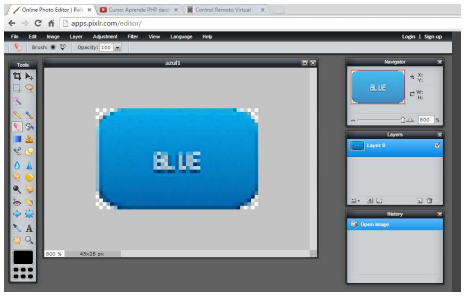


Figura 3.13: Herramienta online “pixlr” [1].

Claramente se observa en la Figura 3.13 que las esquinas de la imagen del botón azul, tienen cuadriculas, ya que de esta manera se representa al fondo transparente, una imagen que permita ver el fondo de la página web es más presentable ya que su fondo cambiaría según el fondo de la página.

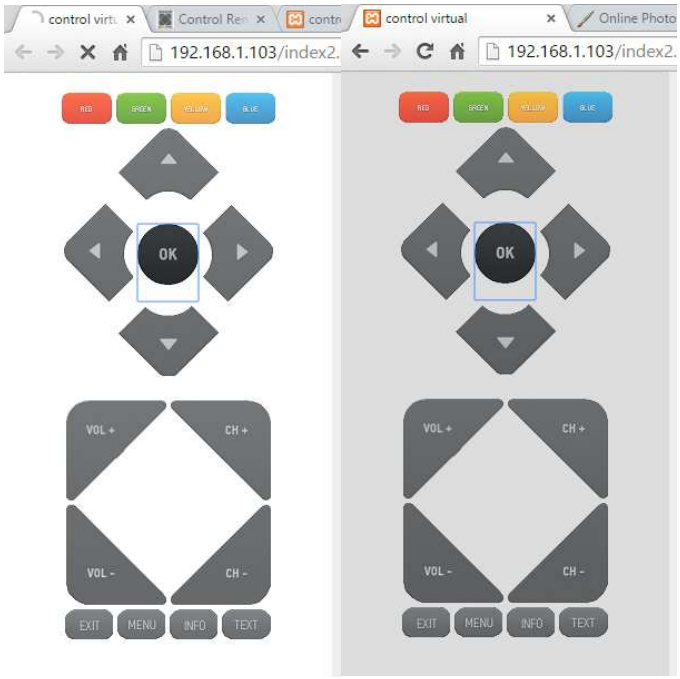


Figura 3.14: Imágenes con fondo transparente y fondo de la página color #ddd.

La Figura 3.14 permite visualizar la ubicación de cada imagen con fondo transparente, de cada botón si observamos la imagen del botón *ok* se puede ver el tamaño verdadero de su botón, que permite visualizar el fondo de la página ya que su propio fondo es transparente.

3.5.2. Utilidades y Aplicaciones de PHP

El archivo *index2.php* está compuesto de la página HTML del control virtual y de la configuración para el puerto COM hecha en PHP por [12].

La clase *php_serial.class.php* permite crear un puerto serial COM y especificar sus atributos a los valores que se necesiten, la comunicación del servidor con el hardware utiliza la configuración para el puerto COM de 9600 baudios 8 bits de información y 1 bit de parada. También este archivo de la clase permite abrir el puerto, enviar información por el puerto y cerrar el puerto, entonces se asignó un numero o una letra mayúscula a cada botón de nuestro control remoto virtual, para representarlo tanto en HTML (dentro de *index2.php*) como en PHP (del mismo *index2.php*) y también en Arduino ™. Las siguientes figuras muestran un ejemplo de la asignación de números y letras en las 3 partes del código, necesarias para la comunicación RS-232.

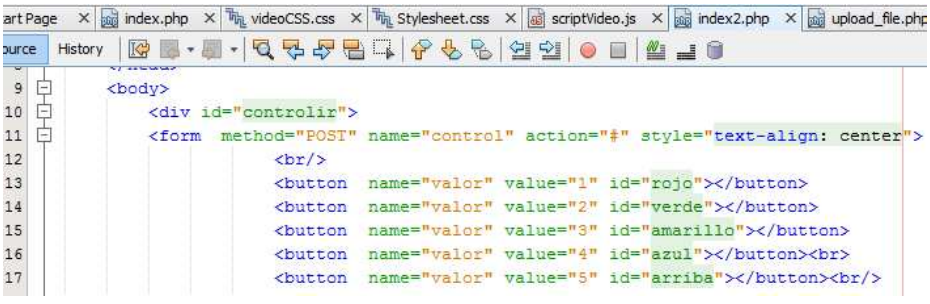


Figura 3.15: Asignación de números al atributo *value* de los botones en html5.

La Figura 3.15 muestra una sección de la enumeración de los botones virtuales que van del 1 al 17, es fácil darse cuenta también que todos los botones tienen el mismo *name* y diferente *value*, este mismo *name* se asigna

en la variable *$valor* en la sección de código correspondiente a php mostrado en la Figura 3.16, con el bucle switch *($valor*) que recibe los números del 1 al 17 y envía por el puerto serial el nuevo carácter asignado a los botones, carácter con el que serán identificados el Arduino ™.

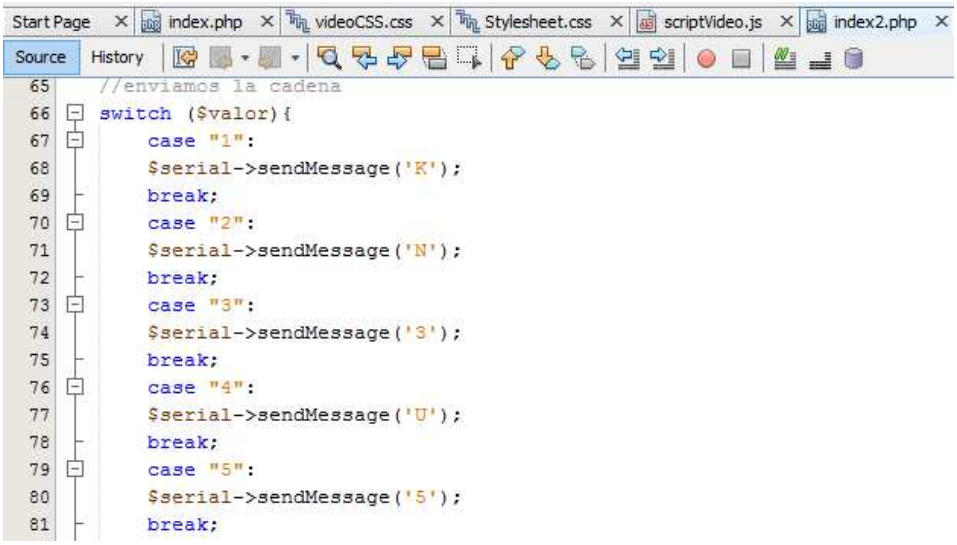


Figura 3.16: Asignación de números y letras que se enviaran por el puerto COM3.

Ya del otro lado de la comunicación serial entre servidor-tarjeta Arduino ™ se reciben los nuevos caracteres asignados a los botones, en la tarjeta Arduino™ se emplea nuevamente el bucle switch para identificar dicho carácter y de esta manera dar la orden para reproducir el código IR correspondiente a cada botón. La Figura 3.17 contiene parte del bucle switch en Arduino ™.

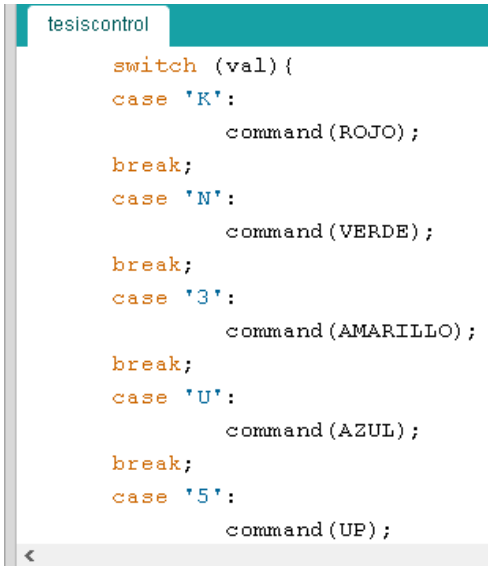


Figura 3.17: Recepción de los valores por puerto serial en Arduino ™.

Otra de las utilidades de PHP que se puede usar, es la carga de archivos que también se implementó en esta plataforma, previa a la plataforma de pruebas de usabilidad de aplicaciones interactivas final; el archivo *upload_file.php* contiene la programación para subir archivos imagen con las extensiones jpg, jpeg, gif, png únicamente, para la carga de archivos imagen se puso en práctica la técnica usada por [13], en esta fuente hay una serie de 10 videos para aprender PHP desde cero, lo que básicamente hacemos en la carga de archivos es; mediante el método *POST* enviar los archivos desde una variable input tipo archivo y un botón tipo *submit*, hacia PHP y mediante la sentencia *$_FILE* acceder a los atributos del archivo, así con variables auxiliares y comandos para operar sobre cadenas de caracteres, por ejemplo, restringir las extensión del archivo buscando dentro del nombre del archivo (cadena de caracteres) la extensión del archivo usado, o también para restringir el tamaño del archivo.

3.5.3. Apariencia de los componentes estilizados mediante CSS3

CSS3 tiene una gran variedad de selectores, para acceder a los elementos dentro de HTML, ya sea por su clase, pseudo clase, o por su ID, y la hoja de estilos dentro de su bloque de declaración de estilos, dar valores a las propiedades de los elementos HMTL5, los selectores permiten ubicar uno o varios elementos haciendo que las hojas CSS ahorren código al cambiar las propiedades de muchos elementos de una misma clase a la vez. Los archivos que contienen las hojas de estilo en cascada son *Stylesheet.css* y *videoCSS.css*.

En el diseño del acceso virtual se utilizó los selectores *::selection*, *[atributo*=valor]*, *input:in-range*, los navegadores actualizados interpretan CSS3, donde están incluidos los selectores de versiones anteriores de CSS. A continuación el código de la hoja de estilos *Stylesheet.css* en la Figura 3.18 las 11 primeras líneas de código modifican todos los elementos de la página web **::selection{…}* cambia únicamente el color cuando seleccionemos cualquier elemento, **{….}* elimina el margen y relleno de todos los elementos con el propósito de resetear todos estos elementos y que se ejecuten los cambios posteriores de manera eficaz.

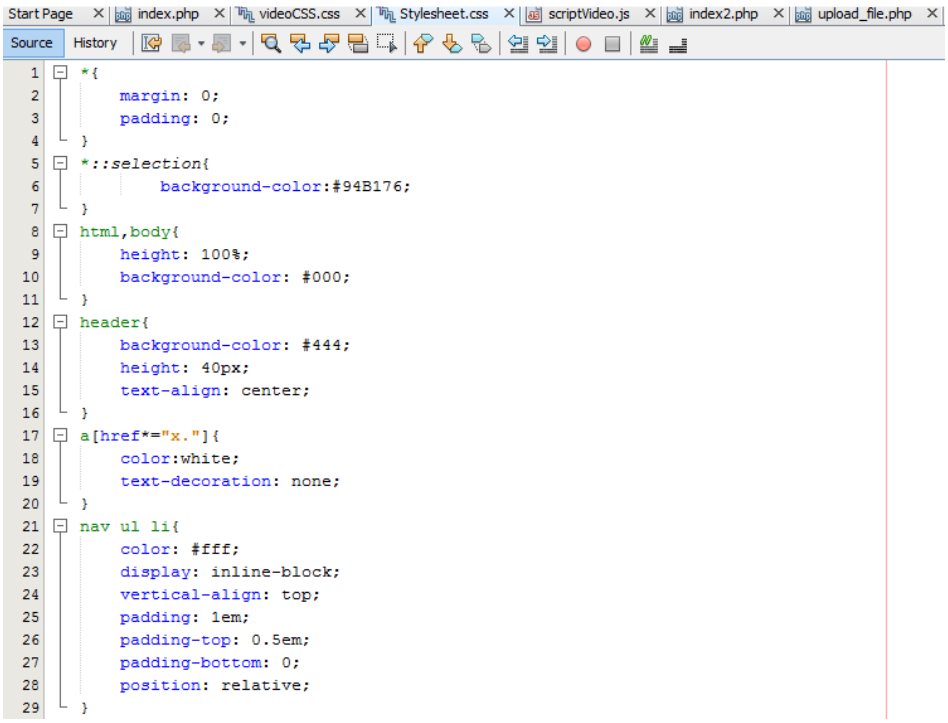
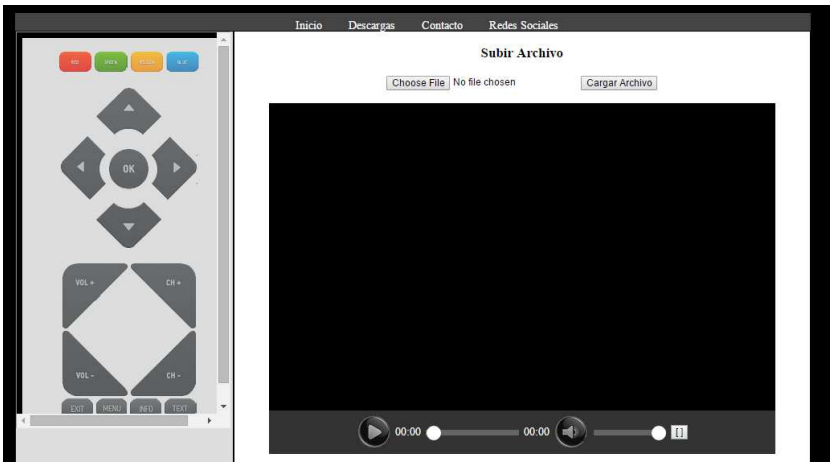


Figura 3.18: Cambio de la apariencia de elementos HTML5.

La Figura 3.19 ejemplifica el uso de **{margin: 0; pading: 0;}*, en la figura anterior también se ve el selector *a[href*=”x.”]*, con este selector se ubica el enlace *a* de *Inicio* que re-direcciona hacia la página principal. También se ve la secuencia de la etiqueta *nav ul li* que apunta al elemento *li* siempre y cuando esté después de un *ul* que pertenece a un *nav*, es decir solo si la secuencia se cumple, desde ahí todos los selectores que comiencen con *nav*, sirven para dar formato a las listas y los campos de *header*.

(a)



(b)

Figura 3.19: a) con **{margin: 0; pading: 0;}*, b) sin **{margin: 0; pading: 0;}*

3.5.4. JavaScript para el Reproductor de Video

Otra de las ventajas de los navegadores es que reconocen muchos lenguajes soporte para HTML5, hay lenguajes para ejecutarse del lado del cliente y otros lenguajes para ejecutarse del lado del servidor, la siguiente herramienta JavaScript se ejecuta del lado del cliente, ya que, a diferencia de su opuesto PHP (del lado del servidor) éste no necesita instalaciones extras para que el navegador entienda que se trata de un archivo de extensión .js. En el caso de que el navegador no ejecute JavaScript es porque seguramente

se encuentra desactivado, ya que esta característica puede ser supervisada por el cliente. En la cabecera de *index.php* se enlaza la página principal con el archivo *scriptVideo.js* mediante la siguiente sentencia:

<script src="scriptVideo.js" type="text/javascript"></script>

Luego el archivo JavaScript *scriptVideo.js*, tiene siete funciones necesarias para manipular los controladores del reproductor de video, el documento comienza declarando todas las variables que se utilizan en él, y seguidamente la primera función donde se inicializan estas variables, mediante el comando *document.getElementById('elemento')*; captura el elemento del documento HTML por medio de su ID que ira entre comillas simples dentro de los paréntesis del comando, a continuación de esta función se ejecuta la siguiente línea:

window.onload=inicializarplayer;

Para que en el momento exacto en que se carga la página ejecute la inicialización de todas las variables del archivo JavaScript, la función de inicialización también declara eventos escucha (Listener) para los controladores (botones y barras) del reproductor de video, eventos que se activan al dar un clic, al cambiar el rango de una barra, o para actualizar el tiempo del video.

La siguiente función *playPause()* verifica si el video esta pausado o en reproducción, en esta función se cambia la imagen del *botonplaypause* según el estado del video.

La función *playMute()* verifica si el video esta silenciado o con sonido, en esta función se cambia la imagen del *volumenbtn* según el estado del video/

La función *vidSeek()* actualiza el tiempo actual del video según la posición a la que se deslice la barra *seekslider*.

La función *seektimeupdate()* actualiza la posición de *seekslider* según el tiempo actual del video, también calcula el tiempo de duración del video y cambia los campos de texto *curtimetext* y *durtimetext*, campos que fueron ubicados en los controladores del video para dar a conocer el tiempo actual y la duración del video; y además verifica si estos campos son iguales y resetean las variables para permitir reproducir el video nuevamente.

La función *vidSound()* actualiza el volumen del video según la posición de la barra *soundslider*.

Para el efecto de fullscreen se creó otro archivo, el *index3.php*, el mismo que reproduce el video como *background* de la página web utilizando la información de la fuente [14], hacerlo de esta manera es muy útil, ya que se puede observar la transmisión en pantalla completa como se muestra en la Figura 3.20, y se mantiene visible el control remoto virtual. Cabe recalcar que si se usaba la propiedad fullscreen propia del reproductor de video, no se podía de manera alguna tener visible el control virtual, incluso los controladores de video personalizados cambian por los controladores default.

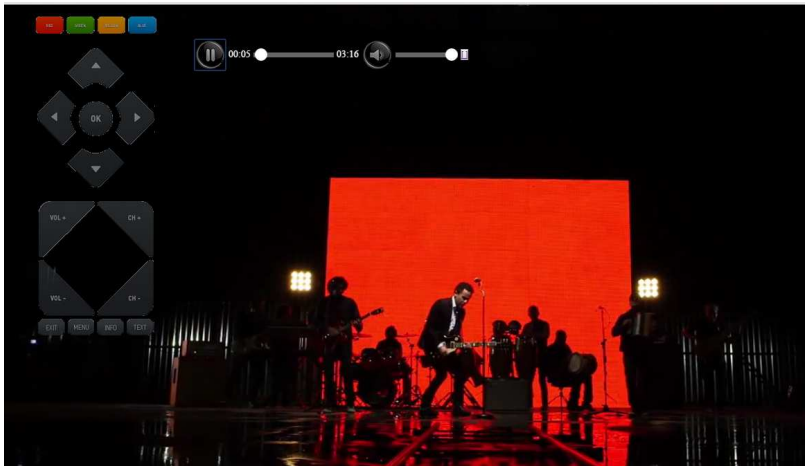


Figura 3.20: Pantalla Completa, video más control remoto virtual.

El archivo *index3.php* trabaja en conjunto con *index2.php* y las hojas de estilos en cascada *videoCSS.css* y *fullscreen.css*, con la primera hoja de estilos, para reutilizar el código de los controladores de video y con la segunda hoja de estilos, para ubicar el video en toda la pantalla del navegador.

CAPÍTULO 4

SIMULACION Y PRUEBAS DE COMUNICACIÓN ENTRE EL ACCESO VIRTUAL Y LA TARJETA ARDUINO™ GENERADORA DE SEÑALES IR

Para probar el funcionamiento del prototipo de un control remoto con acceso virtual, nos ayudamos de las siguientes herramientas: un panel de control conocido como XAMPP para “montar” el servidor que contenga la página web y los interpretes de PHP, un emulador de puertos COM, VSPE para simular la comunicación serial entre Arduino™ (hardware transmisor de señales IR), PHP, y el software Proteus v8.0 para simular el hardware del control IR.

4.1. SIMULACION DE COMUNICACIÓN SERIAL

Como ya se mencionó se ocupara tanto un emulador de puertos serial y un simulador de circuitos con soporte para Arduino ™, en las siguientes figuras se dará indicaciones de cómo realizar la comunicación serial. Se puede ocupar el emulador que más se familiarice, para una nueva configuración de puertos se hace clic en *Create new device…* como se ve en la Figura 4.1

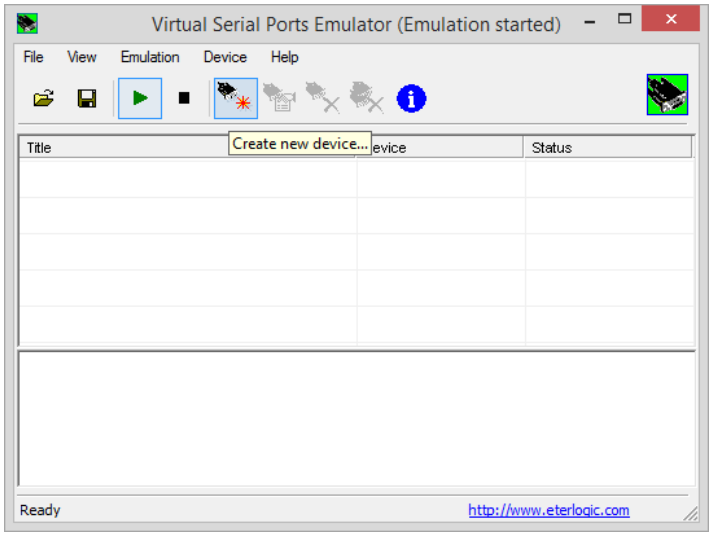


Figura 4.1: VSPE software emulador de puertos COM virtuales.

Luego se elige en la pestaña *Device type* la opción *Pair*, se puede ver en la Figura 4.2 que se crearan dos puertos COM virtuales, después de da clic en *Next* > y en *Finish*.

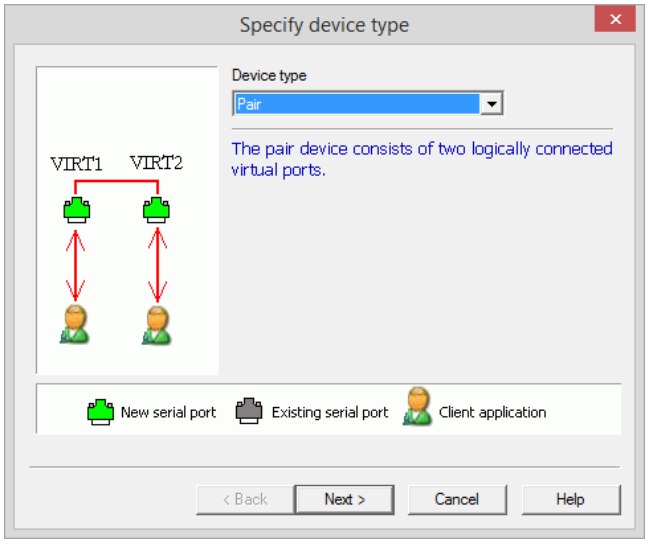


Figura 4.2: Selección de la comunicación a realizar.

La Figura 4.3 representa los puertos COM creados virtualmente para la simulación son COM1 y COM2, al crear los puertos, el inicio es inmediato.

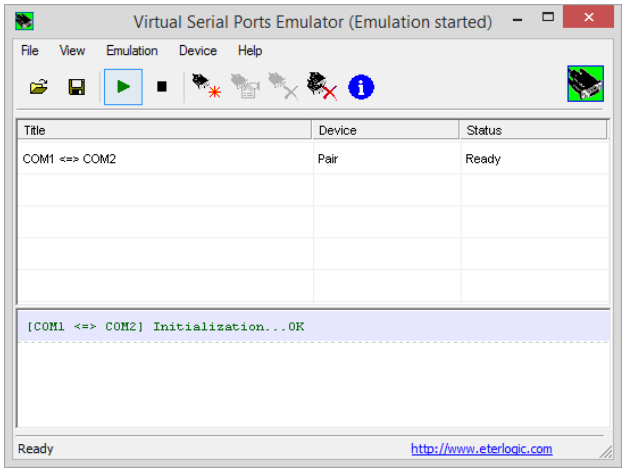


Figura 4.3: Emulación de puertos COM puesta en marcha.

4.1.1. Asignación de puertos COM

Es necesario ser cuidadoso con la asignación de los puertos, y revisar con que puerto se compila el programa en Arduino ™, también revisar el número de puerto que ya esté utilizado en la pc que trabajará como servidor, porque muchos dispositivos conectados a la pc mediante USB, al instalar sus drivers instalan también un puerto COM X, al que se conectaran para interactuar con la pc.

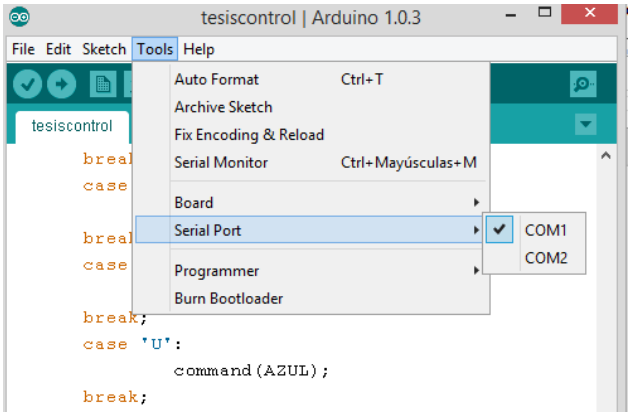
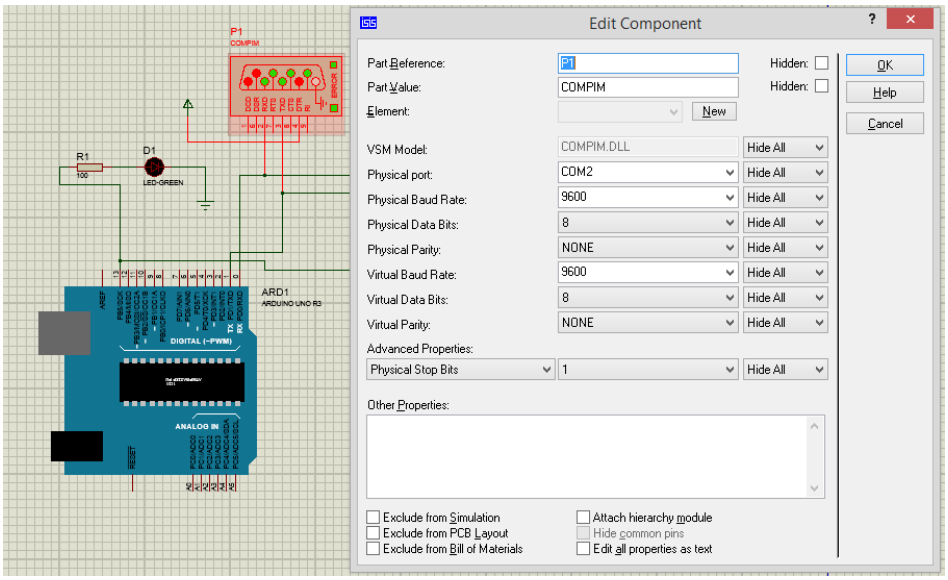


Figura 4.4: Selección de puerto COM en Arduino™ para la compilación.

Como vimos los puertos virtuales que se crearon fueron COM 1 y COM 2 y por eso aparecen al momento de seleccionar uno en Arduino ™, mírese la Figura 4.4, para la compilación se puede elegir cualquier puerto de los que aparecen en Arduino ™, pero hay que cerciorarse que dicho puerto este en desuso.



(a)

```
45  include 'php_serial.class.php';
46
47  //Inicializa la clase
48  $serial = new phpSerial();
49
50  //Especifica el puerto que va a utilizar, en é
51  $serial->deviceSet("COM2");
52
53  //Establece los parámetros del puerto serie
54  $serial->confBaudRate(9600);
55  $serial->confParity("none");
56  $serial->confCharacterLength(8);
57  $serial->confStopBits(1);
58  $serial->confFlowControl("none");
```

(b)

Figura 4.5: Asignación del mismo número de puerto COM, en Arduino™ y PHP.

Si asignamos el mismo valor en ambos lados de la comunicación sucederá lo que la Figura 4.6 dentro del frame debajo de los botones se despliegan los mensajes de que la comunicación no se llevó a cabo.

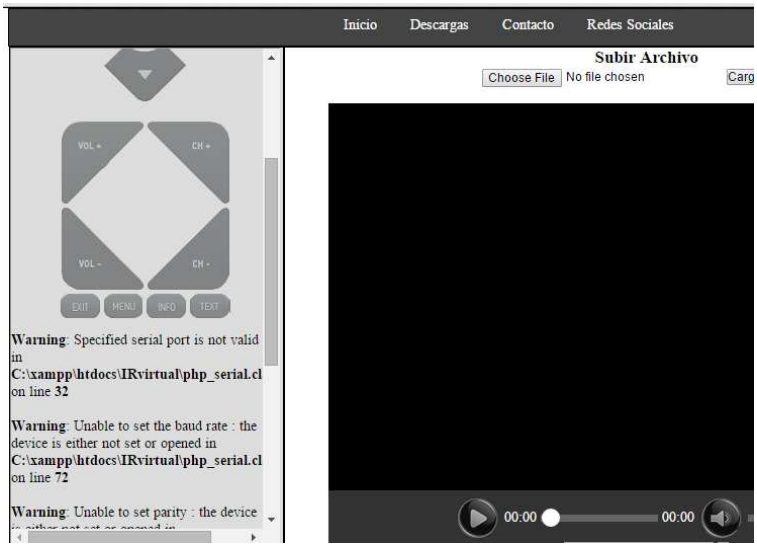


Figura 4.6: Error causado por la asignación del mismo número de puerto COM.

4.1.2. Proteus v8.0 y Arduino ™

EL circuito usado para la simulación en Proteus v8.0 es el mostrado en la Figura 4.7, el mismo que consta de un terminal virtual para escribir lo que recibe el circuito por el puerto COM 1, y de un osciloscopio para ver las señales IR que emite el circuito después de identificar los bits de entrada a que botón representa.

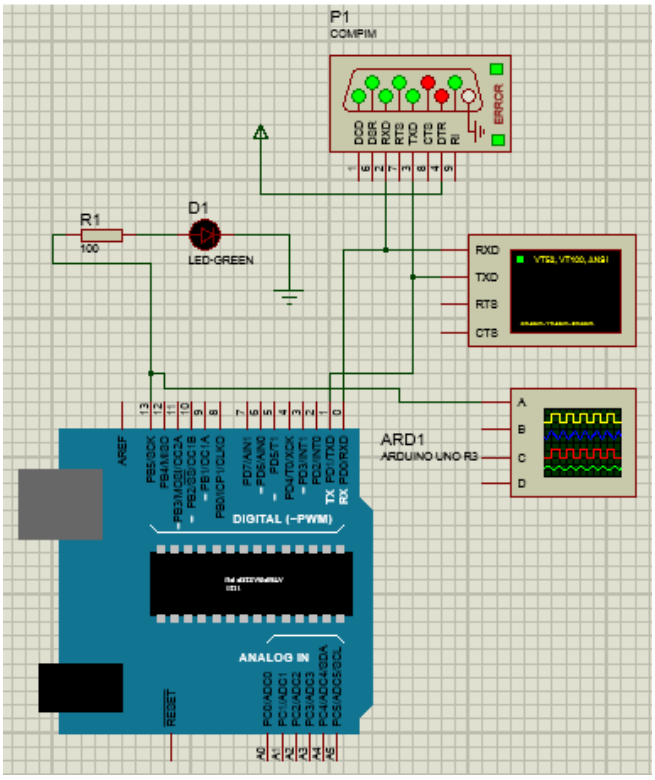


Figura 4.7: Circuito en Proteus v8.0.

Desde el compilador de Arduino™ se puede conseguir el directorio donde se guarda sus archivos temporales de los cuales el de nuestro interés es el .hex, al realizar la verificación del código escrito en el programa Arduino™ se crean estos 2 archivos y en la parte inferior de la ventana del programa se puede copiar la dirección de su ubicación. En esta Figura 4.8 se muestra el directorio donde fueron guardados los archivos.

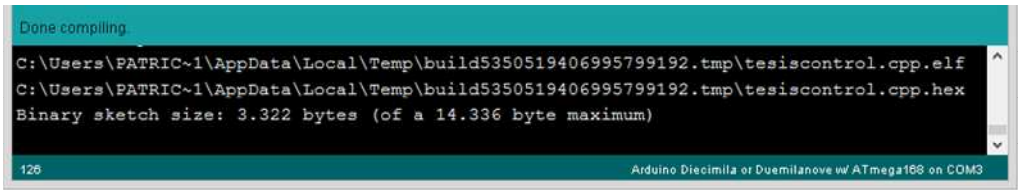


Figura 4.8: directorio donde se guardan los archivos de la compilación Arduino™.

Copiamos esta dirección y la pegamos en Proteus v8.0, abriendo el editor del componente (de la tarjeta Arduino ^{TM}) y pegando en *Program File*, le diremos a la tarjeta con que archivo debe trabajar.

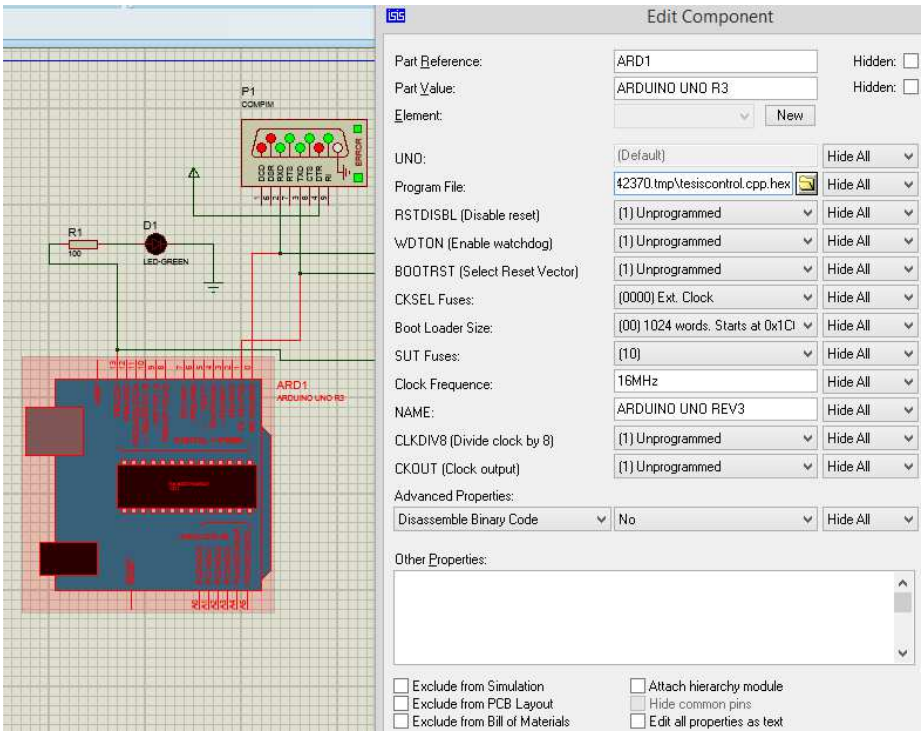
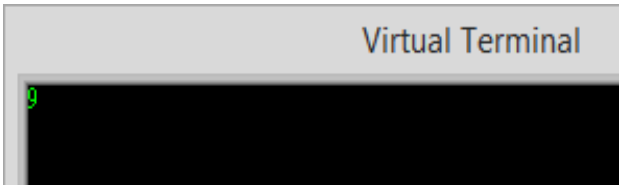
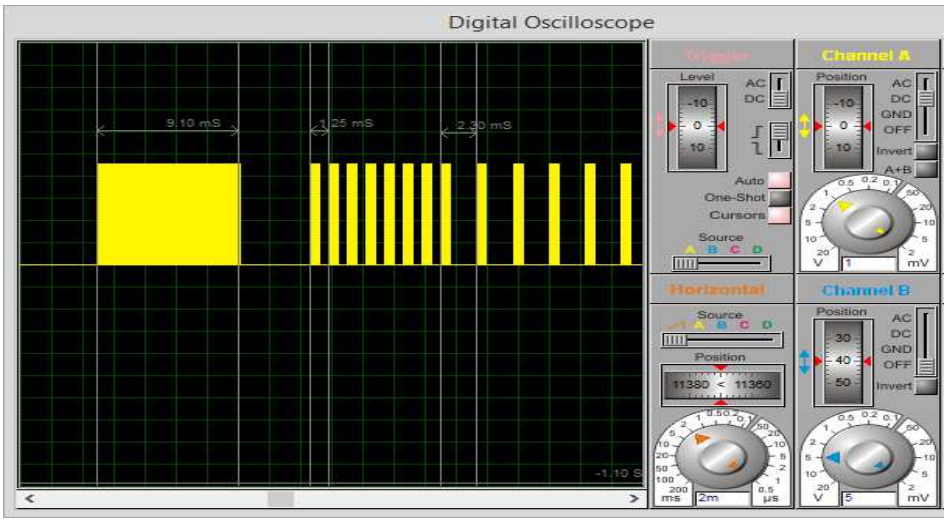


Figura 4.9: *Edit Component* de la tarjeta Arduino™.

En la Figura 4.10 se ve la simulación del prototipo en Proteus v8.0 y se puede apreciar los tiempos tanto del pulso guía como de los pulsos de información y con la ayuda de los cursores medir la duración de cada uno, esta medida varia debido a los ciclos de instrucción que se demoren cada línea de código en ser ejecutada, lo siguiente es ajustar mediante el método de prueba y error, para que la señal se acerque a la deseada, vale la pena recalcar que los decodificadores tiene una pequeña tolerancia a la duración de los pulsos y a la frecuencia de las ráfagas.



(a)



(b)

Figura 4.10: a) carácter recibido por el puerto serial, b) señal emitida por la tarjeta.

4.2. PRUEBAS USABILIDAD

Como objetivo específico tenemos analizar la usabilidad del prototipo de control remoto IR, a continuación se presentan las pruebas de correspondencia, intuitivo y manejable.

4.2.1. Pruebas de correspondencia entre HTML5, iframe y PHP

Para realizar esta prueba se hace clic en cada botón y en el orden indicado por la Figura 4.11 y en la siguiente imagen esta la recepción por medio del puerto COM 1 en la simulación con Proteus v8.0, aún se utilizó esta simulación ya que permite corroborar los caracteres mediante la terminal virtual.

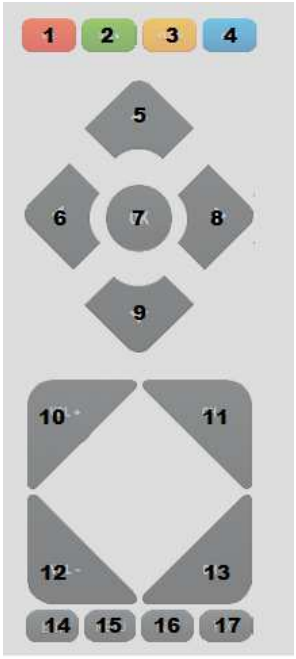


Figura 4.11: Orden en el que se cliqueó cada botón

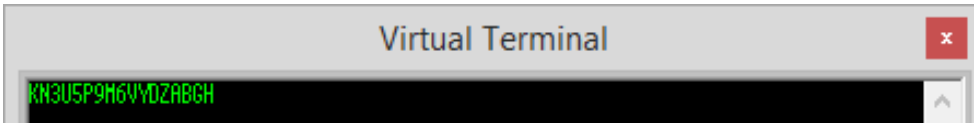


Figura 4.12: Caracteres recibidos por el puerto COM 1 en la simulación.

Para una mejor observación se tabula en la Tabla 4.1 , y se compara con los valores asignados en la Tabla 3.2, para verificar si es o no correcto el carácter recibido.

Tabla 4.1.

Tabla comparativa para pruebas de correspondencia.

Orden Clic	Carácter recibido	Corresponde a	Valor en HTML5
1	K	ROJO	1
2	N	VERDE	2
3	3	AMARILLO	3
4	U	AZUL	4
5	5	UP	5
6	P	LEFT	7
7	9	OK	9
8	M	RIGHT	8

Orden Clic	Carácter recibido	Corresponde a	Valor en HTML5
9	6	DOWN	6
10	V	VOL+	12
11	Y	CH+	14
12	D	VOL-	13
13	Z	CH-	15
14	A	EXIT	10
15	B	MENU	11
16	G	INFO	16
17	H	TEXT	17

La cuarta fila de la Tabla 4.1 se llena según el orden expresado en la Figura 4.13 y si verificamos estos valores son correctos.

```
<button  name="valor" value="1" ></button>
<button  name="valor" value="2" ></button>
<button  name="valor" value="3" ></button>
<button  name="valor" value="4" ></button><br><br>

<button  name="valor" value="5" ></button><br/>
<button  name="valor" value="7" ></button>
<button  name="valor" value="9" ></button>
<button  name="valor" value="8" ></button><br/>
<button  name="valor" value="6" ></button><br/><br/>

<button  name="valor" value="12" ></button>
<button  name="valor" value="14" ></button><br/>
<button  name="valor" value="13" ></button>
<button  name="valor" value="15" ></button><br/>

<button  name="valor" value="10" ></button>
<button  name="valor" value="11" ></button>
<button  name="valor" value="16" ></button>
<button  name="valor" value="17" ></button>
```

Figura 4.13: Orden de los botones en HTML5, distribución en el navegador web.

4.2.2. Pruebas de usabilidad como página intuitiva y manejable

En vez de usar el recuadro de los botones en HTML5, se utilizó como fondo del botón una imagen, esto se da gracias a CSS. Las imágenes usadas tiene el color la forma y el texto que hace intuitivo saber qué hace cada botón, además, se puede ver a primera vista y ubicar de manera rápida cada botón,

es acertado mostrar figuras que se familiaricen con el usuario y debido a este uso el control IR virtual es manejable.

Se modifica la opacidad de los botones, que serán opacos mientras no se apunte con el cursor del mouse; una vez que sean apuntados recuperan su visibilidad hasta que el cursor deje de apuntarlo. Esta variación de la opacidad pone a la mano un escenario dinámico con el usuario y amigable también.

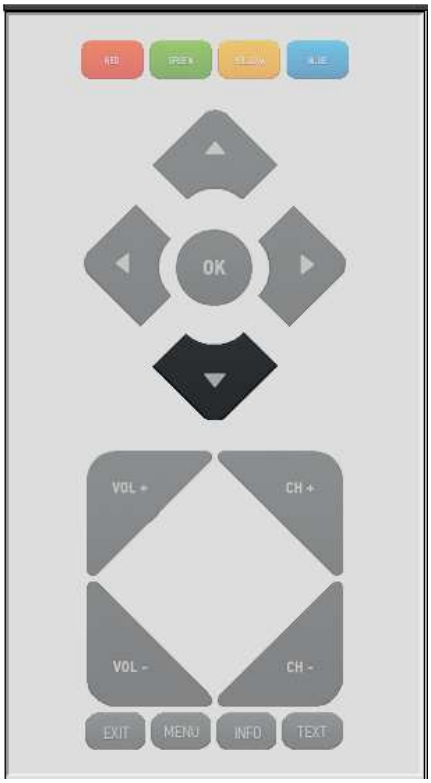


Figura 4.14: Ejemplo del cambio de opacidad de un botón.

En esta Figura 4.14 está el ejemplo del cambio de opacidad y también permite ver que la imagen de los botones tiene la figura que corresponde a la acción que realiza cada botón.

4.3. PRUEBAS DE LEVANTAMIENTO DEL SERVIDOR

Después de abrir el panel de control XAMPP el programa verifica si hay inconvenientes con algún puerto que este siendo ocupado por otra aplicación y que dará problemas con alguna de sus aplicaciones. Para corregir esto se pueden cambiar los puertos que ocupa el servidor apache en la pestaña Config ubicado a la derecha del panel de control, si se inicia correctamente y después de hacer clic en *Admin* del servidor Apache, ver Figura 2.9, el navegador debe mostrar esta dirección *http://localhost/xampp/*. Ahora nada más queda, en una pestaña nueva del navegador digitar la dirección IP del servidor. Si miramos en la parte inferior de la Figura 4.15 en la barra de tareas no se muestra Netbeans ejecutándose.

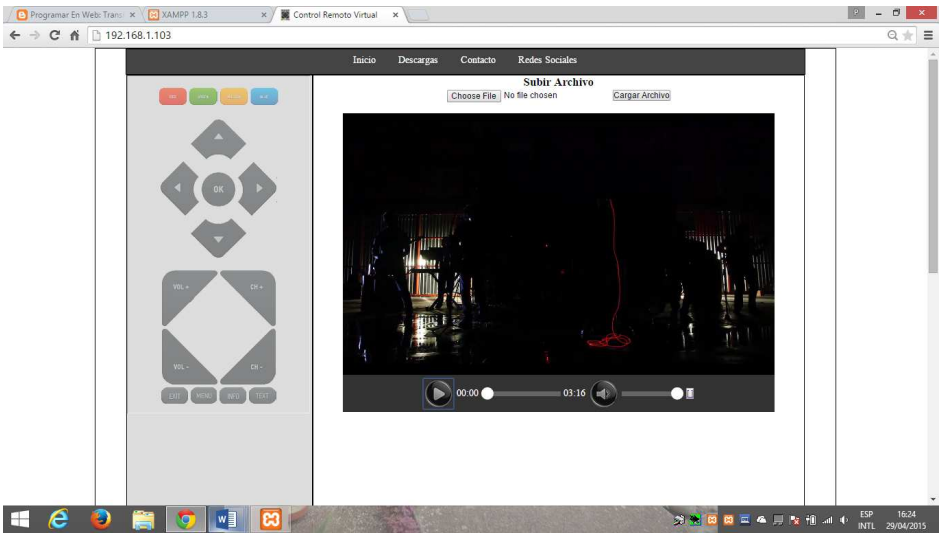


Figura 4.15: Acceso virtual levantado con XAMPP

Si apagamos desde el panel de control el servidor Apache y recargamos la página del navegador se verá lo que indica la Figura 4.16. Demostrando que el panel de control XAMPP es capaz de levantar un servidor para levantar las páginas diseñadas.

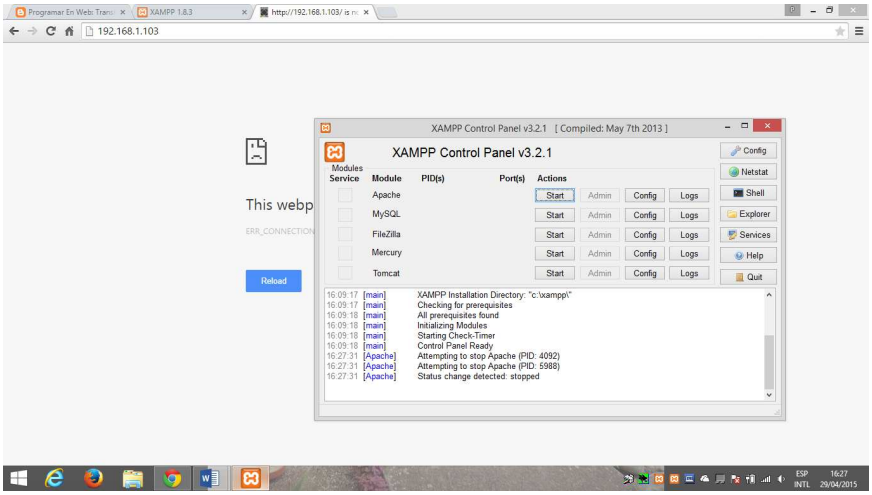


Figura 4.16: Servidor Apache apagado

Todo esto para montar un servidor de red local, ya que el propósito es llegar a los diseñadores de toda la región, la Universidad de las Fuerzas Armadas ESPE debe dar cabida en su servidor todas las aplicaciones que se diseñen en los laboratorios de ESPE TV.

4.4. PRUEBAS DEL PROTOTIPO DE CONTROL REMOTO Y EL ACCESO VIRTUAL

El escenario propuesto se muestra en la Figura 4.17, aquí podemos ver que el prototipo se encuentra instalado y configurado en un servidor de los laboratorios ESPE TV, de manera que al escribir la dirección *url* de la página en un navegador web cliente, se pueda utilizar el control remoto virtual y este se sirva de la tarjeta Arduino ™ para generar la señal IR correspondiente a cada botón y así manipular el STB del laboratorio.

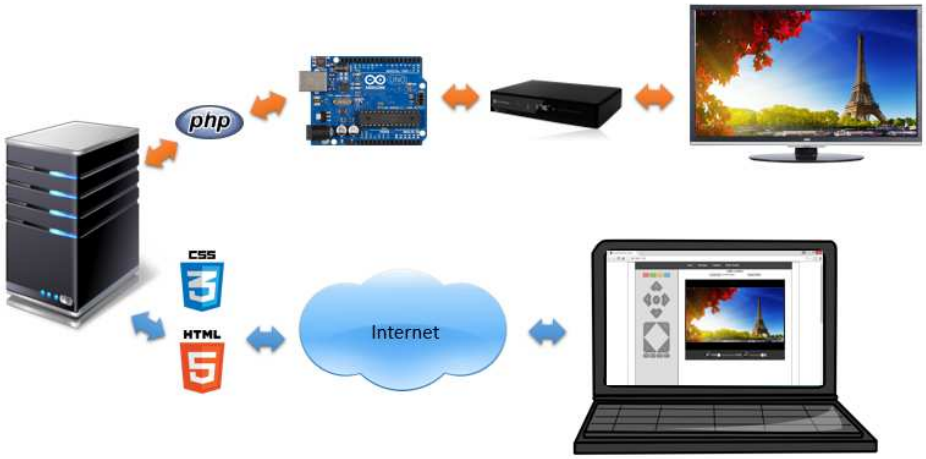


Figura 4.17: Escenario de la plataforma virtual.

4.4.1. Duración de ráfagas y espacios generados en Arduino ™

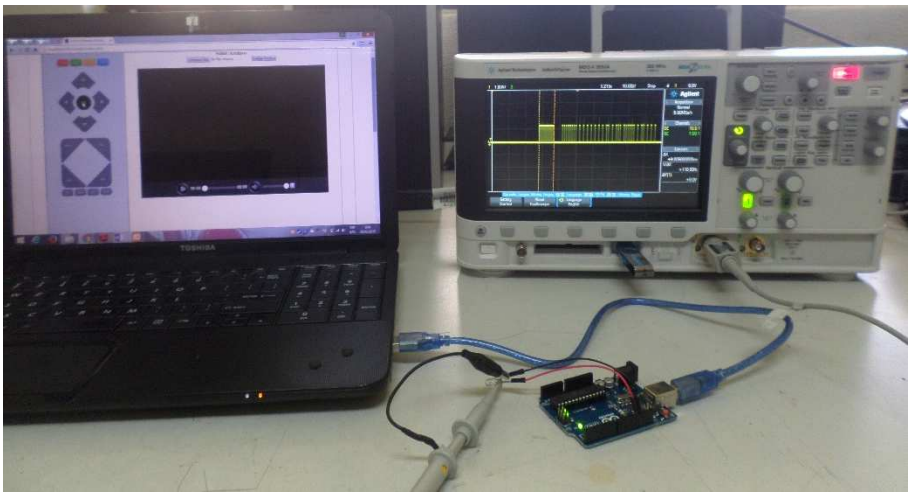


Figura 4.18: medición de la duración de las ráfagas y espacios.

Mediante un osciloscopio se procedió a medir la duración de las ráfagas tanto del primer pulso guía de 9ms como de los “0”s y “1”s, luego fue necesario cambiar el código en Arduino ™ para ajustar la duración de las ráfagas y espacios, los valores que se cambiaron en el código son *period=29* y *wait_time=11*, en las funciones *on0 on1* y *firstcommand.*

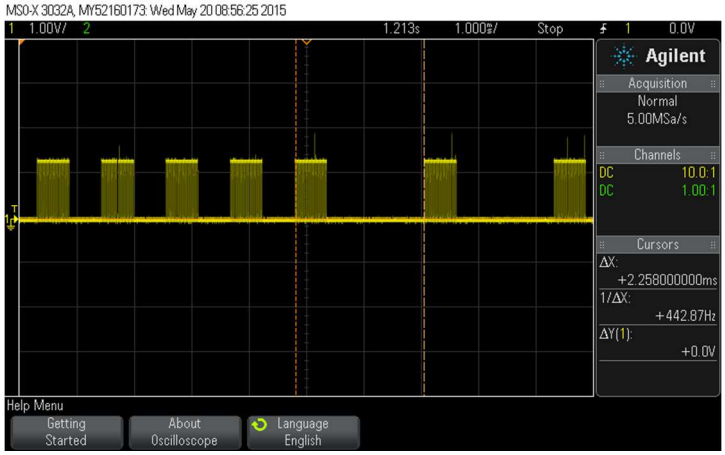


Figura 4.19: ráfaga más distancia larga (representación de "1").

Mirando la Figura 4.19, notamos que se trata de la representación de un "1" donde su duración es 2,258ms, valor que se acerca lo suficiente a los 2.25ms que es la duración que especifica la codificación NEC. A continuación en la Tabla 4.2 se tabula los valores medidos con el osciloscopio de la señal IR que reproduce la tarjeta Arduino ™, además se calcula la desviación, para tener una estimación de la tolerancia del estándar utilizado en la generación de señales infrarrojas. Si se desea verificar estos valores se puede revisar tanto la sección 2.3 y ANEXO "C".

Tabla 4.2.

Duración de las ráfagas y espacios.

	Valor especificado en el estándar	Valor medido con el osciloscopio	Error
Pulso guía	9ms	9,0098ms	**1,43111111**
Espacio después del pulso guía	4,5ms	4,5644ms	**2,22222222**
Duración de la ráfaga	562,5µs	550µs	**0,24888889**
Duración de un bit cero	1,125ms	1,1278ms	**0,35555556**
Duración de un bit uno	2,25ms	2,258ms	**0,07179487**
Duración total de la codificación NEC	68,25ms	68,201ms	**1,43111111**

4.4.2. Pruebas de Retardo

Para revisar los retardos además de la decisión de enviar un solo carácter que represente a cada botón dentro de Arduino ™, se tomó en cuenta la apreciación del usuario manejando el acceso virtual, y con esta apreciación se decidió retomar el código y en una parte del código grabado en la tarjeta, se encontró el retardo de 40ms que se sumaba a los 68,2ms de la información de un botón pulsado, después de retirar este retardo la experiencia del usuario mejoro de manera satisfactoria.

El escenario es el mostrado en la Figura 4.20, el CPU que se ve en la figura funcionará temporalmente como servidor para que levante la página web, para el propósito se cargó una aplicación embebida en el STB (equipo debajo del televisor) y se sometió el prototipo a pruebas de usabilidad.

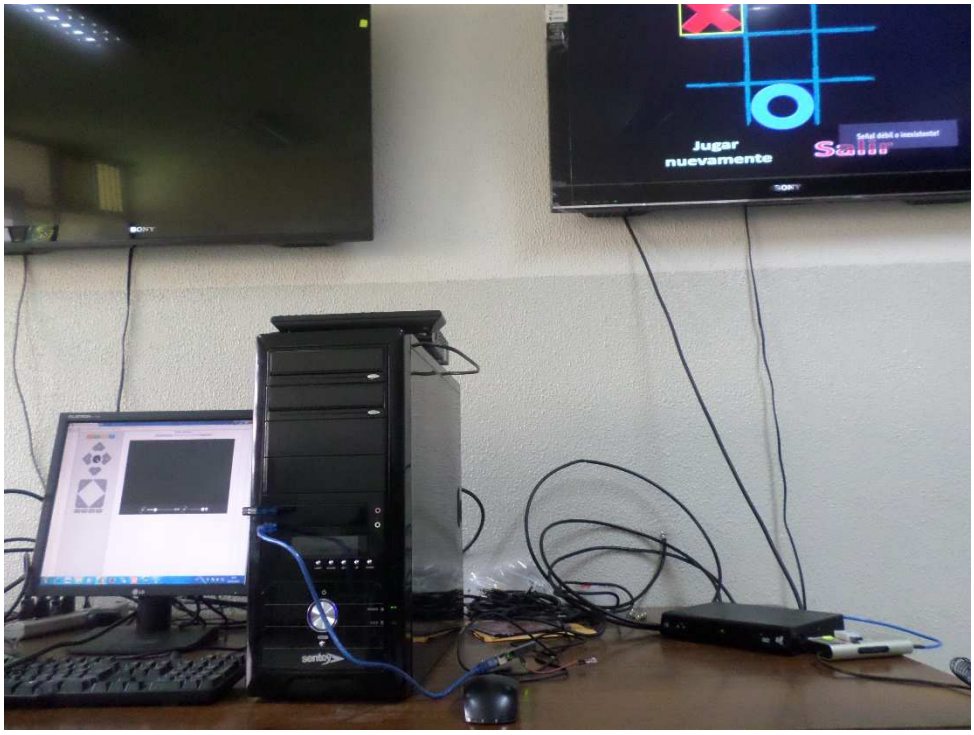


Figura 4.20: Servidor del laboratorio de ESPE TV

Recordando que el control virtual se encuentra dentro de un iframe, y al hacer clic en uno de los botones este iframe se recarga dando un tiempo de espera para poder presionar otro botón o el mismo botón, dicho lapso de tiempo es obviado por el usuario de la web ya que es costumbre que las páginas recarguen, esto es según la velocidad del servidor que está dando soporte a la página.

Figura 4.21: Tarjeta Arduino ™ UNO R3 apuntando con IRLED hacia el STB.

CAPÍTULO 5

CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

- Se realizó el prototipo de un control remoto IR con codificación NEC en Arduino ™ y el acceso virtual mediante HTML5, CSS3 y PHP, el hardware se comunica con el software mediante el estándar RS-232 y el escenario levanta la página web gracias a XAMPP.

- Se hizo el proceso de comunicación mediante RS-232, PHP crea una instancia de puerto COM x y envía un carácter diferente asignado a cada botón del control virtual, que luego es interpretado por la tarjeta Arduino ™, quien genera el código infrarrojo y lo transmite en una de sus salidas digitales.

- La recepción y decodificación de las señales IR se basa en el análisis de su forma de onda con la Herramienta MatLab.® a una frecuencia de muestreo de 8000 muestras por segundo.

- Se implementó mediante el hardware Arduino ™ UNO R3 el control remoto IR usando una salida digital que reproduce las ráfagas para la representación de bit lógicos “0”s y “1”s a una frecuencia de 38kHz, y la programación contiene las funciones respectivas para generar las ráfagas del pulso guía, de los “0”s y “1” además de los espacios respectivos según la codificación NEC.

- Se programó en HTML5 el acceso virtual y gracias a que este lenguaje de marcado puede vincularse con el lenguaje de programación PHP se

logró enlazar la página web con el hardware Arduino ™, el estilo gráfico es hecho con CSS3 evitando la imagen default de los botones y asignando una imagen en el atributo url de cada botón.

- Se analizaron los resultados luego de las pruebas de comunicación, retardos y correspondencia; en el servidor se instaló el driver de la tarjeta Arduino ™ manualmente, se instaló también el panel de control XAMPP que contiene el servidor Apache y los intérpretes para PHP.

5.2. RECOMENDACIONES

- En este reto efectivamente se trata de una señal codificada con características NEC, en caso de que el control remoto del set-top-box venga con diferentes especificaciones se puede revisar [3] para realizar los debidos cambios y decodificar el nuevo estándar.

- Arduino™ no permite compilar y grabar el código a menos que el número de puerto COM este cerrado o libre para su uso. Es indispensable asignar un numero diferente para cada lado de la comunicación así no habrá inconveniente alguno al enviar o recibir datos.

- En su *pin out* los receptores de señales infrarrojas de tres pines, envían V+ al no recibir la ráfaga de señal infrarroja y su salida cambia a cero voltios cuando si recibe señal infrarroja de manera que se debe tener cuidado al momento de interpretar la señal recibida. Y por eso es que se usó el transistor pnp 2N3906 para invertir la señal.

- Las ráfagas no son generadas por el led infrarrojo se debe generar con el hardware si se usa un receptor de tres pines conectado a un

osciloscopio no se lograra ver las ráfagas tan solo los pulsos que representan la información, es indispensable entender este concepto.

- HTML5 tiene etiquetas que permiten encontrar el tamaño de la pantalla del dispositivo que ejecuta la página web. La sugerencia es usar porcentajes para modificar el tamaño de los elementos que forman parte de la página web así su tamaño estará relacionado con el tamaño de la pantalla del dispositivo.

- PHP con sus métodos $_POST y $_GET envía datos de una página a otra usando también el nombre de los elementos HTML, y de estos dos métodos el más seguro es $_POST porque envía de manera oculta los datos mientras que $_GET deja visible en la url de la página que aparece en el navegador por ejemplo: http://www.ejemplo.com/?nombre1=valor1&nombre2=valor2.

- Si se desea modificar el tiempo en el que inicia la reproducción del video por ejemplo para el modo fullscreen, es mejor que no se modifiquen los controles de reproducción y sonido del video, porque al crear el script para estos controles necesariamente se usan los eventos listener del video donde hay que basarse en el deslizamiento de los input tipo rango para modificar el tiempo de actual de reproducción.

- Si se desea tener más seguridad cuando se usa PHP, se puede enviar por el puerto serial los strings con más caracteres y mezclados entre letras y números, ya que este trabajo es publicado, rápidamente se podría saber los caracteres utilizados para la comunicación serial pero como queda claro las funcionalidades de este tipo de lenguajes permite desarrollar proyectos de mayor alcance explotando esa seguridad que brindan los lenguajes de programación del lado del servidor.

REFERENCIAS BIBLIOGRÁFICAS

[1] Autodesk, “http://www.autodesk.com/,” 2014. [En línea]. Available: http://apps.pixlr.com/editor/.

[2] Altium, “http://www.altium.com/,” 06 06 2013. [En línea]. Available: http://techdocs.altium.com/display/FPGA/NEC+Infrared+Transmission+Protocol.

[3] sbprojects, “http://www.sbprojects.com/index.php,” 8 11 2011. [En línea]. Available: http://www.sbprojects.com/knowledge/ir/index.php.

[4] Altium, “http://www.altium.com/,” 06 11 2013. [En línea]. Available: http://techdocs.altium.com/display/FPGA/Infrared+Communication+Concepts.

[5] hypertextual, “http://www.hipertextual.com/,” 28 05 2013. [En línea]. Available: http://hipertextual.com/archivo/2013/05/entendiendo-html5-guia-para-principiantes/.

[6] D. Barcia, “http://www.maestrosdelweb.com/,” 08 11 2003. [En línea]. Available: http://www.maestrosdelweb.com/introcss/.

[7] w3schools, “http://www.w3schools.com/,” 1999-2015. [En línea]. Available: http://www.w3schools.com/cssref/css_selectors.asp.

[8] The PHP Group, “php.net,” 15 05 2015. [En línea]. Available: http://php.net/manual/es/intro-whatis.php.

[9] oscarah, “www.crysol.org,” 17 05 2007. [En línea]. Available: http://crysol.org/node/689.

[10] D. Xiloj, “https://libertadelectronica.wordpress.com/,” 1 12 2009. [En línea].Available:

https://libertadelectronica.wordpress.com/2009/12/01/decodificando-el-infrarrojo-capturando-la-accion/.

[11] M. Ivey, “http://www.zovirl.com/,” 13 11 2008. [En línea]. Available: http://www.zovirl.com/2008/11/12/building-a-universal-remote-with-an-arduino/.

[12] R. Sanchez, “http://www.phpclasses.org/,” 1 6 2007. [En línea]. Available: http://www.phpclasses.org/browse/file/17926.html.

[13] codejobs, “https://www.youtube.com/user/codejobs,” 27 08 2012. [En línea]. Available: https://www.youtube.com/watch?v=JPUbXL7QCAU.

[14] D. Storey, “https://disqus.com/by/dudleystorey/,” 6 10 2014. [En línea]. Available: http://codepen.io/dudleystorey/pen/knqyK.