Universidad Tecnológica Nacional

Facultad Regional Villa María

Departamento de Electrónica

Juego de laberinto láser

Autor:

Palacios, Pablo Martín.

2021

Acreditación:

Fecha: 23 de junio de 2021

Comité Evaluador

Presidente:

1º Vocal:

2º Vocal:

Dedicatorias

A mis padres, que hicieron todo lo que estuvo a su alcance, para posibilitar mis estudios.

A mi familia, quienes son un pilar fundamental en mi vida.

A mi madrina, que sin estar presente físicamente, me guio con sus enseñanzas.

A mis amigos, los cuales me dieron un empujón en los momentos que lo necesite.

Agradecimientos

Gracias a mi familia, la cual me brindó su apoyo durante estos años y son una parte indispensable en mi vida.

Gracias a mis amigos, que al igual que mi familia, me dieron su apoyo y fueron un sostén a lo largo de estos años, con los que pude compartir momentos inolvidables por dentro y fuera de la universidad.

Gracias a todos los docentes y no docentes de la facultad, por su labor diaria y su gran aporte, hacia la educación.

Por último, gracias a todos mis compañeros que aportaron su tiempo y dedicación en ayudarme.

Memoria Descriptiva

Este proyecto consiste en realizar un juego de laberinto láser, refiriéndose al mismo, como una sala rectangular llenada con una red de rayos láser, donde el propósito y objetivo de los jugadores, es intentar atravesar el laberinto a tiempo sin "tocar" los rayos, porque ante una interrupción del haz, se decrementará un cronometro y se activará una alarma a modo de penalización.

La red de rayos visibles se logra utilizando módulos laser de color verde en conjunto con espejos, aprovechando el principio de reflexión de la luz. Las interrupciones del haz son censadas por fotorresistencias LDR.

El tiempo de partida cronometrado se muestra en un tablero de leds rgb de 3 dígitos, se utiliza un módulo de audio en conjunto con un parlante para reproducir un sonido de alarma ante la interrupción del haz, la sala es iluminada a través de un cartel, con una tira de led rgb, controlada por un módulo, se utiliza un display LCD y pulsadores para establecer parámetros y controlar la interfaz del juego.

Todo esto se logra a través de un microcontrolador PIC, cuyo firmware se desarrolla y se programa en lenguaje C.

ÍNDICE

Título **Pág.**

Dedicatorias 3
Agradecimientos 4
Memoria Descriptiva 5
Introducción 8
Objetivos 9
1. Marco Teórico 10
1.1 Láser 10
1.2 Modulación PWM 11
1.3 Multiplexado de display 7 segmentos 12
1.4 Reflexión de la luz 12
2. Desarrollo de hardware 13
2.1 Módulos láser 14
2.2 Módulo de audio 15
2.3 Módulo para tiras RGB 16
2.4 Tablero RGB 18
2.5 Display LCD 20
2.6 Módulo de sensores 22
2.7 Pulsadores 23
2.8 Fuente de alimentación 23
2.9 Microcontrolador 24
2.9.1 Acerca del controlador PIC16F887 24
2.10 Configuración del oscilador 25
2.11 Configuración del TIMER1 26
2.12 Modo PWM con TIMER2 26
2.13 Configuración interrupción RB0 27
2.14 Conexión del microcontrolador 28
3. Desarrollo de firmware 29
3.1 Subrutinas 29
3.1.1 Configuración 29

3.1.2 Función RGB 30
3.1.3 Print 32
3.1.4 Cambio 32
3.1.5 Menu 32
3.1.6 Reset 32
3.1.7 Ready 33
3.1.8 Game 33
3.1.9 Cronometro 33
3.1.10 Multiplexado 33
3.1.11 Botones 34
3.1.12 TMR1 34
3.1.13 RB0 35
3.1.14 WTV020 35
3.2 Rutina principal 37
4. Evaluación Final del Sistema 39
5. Conclusiones 41
Bibliografía 43
Anexo I 44
Anexo II 45

Introducción

Un juego de laberinto láser es una sala rectangular llena de rayos láser, cuyo propósito y objetivo de los jugadores, es intentar atravesarlo a tiempo sin "tocarlos", ya que ante una interrupción del haz, se decrementa un cronometro y se activa una alarma a modo de penalización.

La idea detrás del juego viene de películas como Entrapment, Mission impossible y ocean's twelve.

Este proyecto nace para ser destinado a una empresa de eventos infantiles, llamada *La Casa de Zarpy*, por lo tanto, las características del mismo fueron adaptadas acorde a las peticiones de la empresa.

El sistema se divide en dos partes, por un lado el hardware, que referencia a los módulos físicos y electrónicos, y por otra parte, el firmware, encargado de la lógica para controlar dicho hardware.

Se muestra el diagrama en bloques del sistema en la **Fig. Nº 1**.

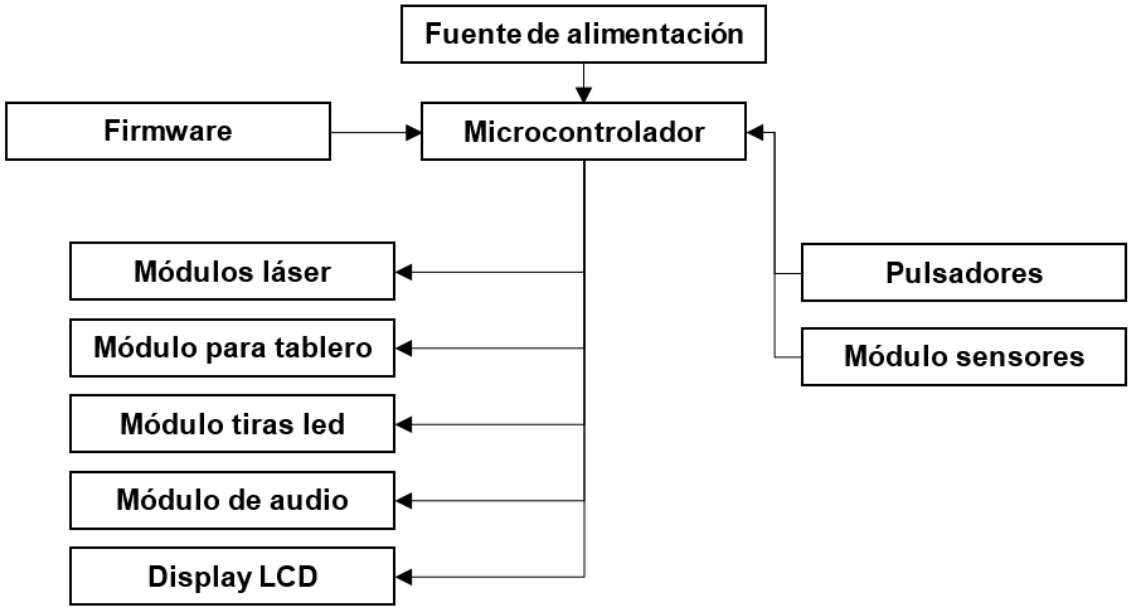

Fig. Nº 1 *Diagrama en bloques de las partes del laberinto láser.*

Objetivos

- *Objetivos generales*

Brindar en el mercado local un producto innovador en el ámbito de entretenimiento totalmente personalizado, que sea rentable, atractivo y apto para todo público.

- *Objetivos particulares*

Desarrollar un sistema seguro y robusto que cumpla los requisitos del cliente, y las expectativas de los consumidores para crear una buena imagen profesional.

1. Marco Teórico

1.1 Láser

Un láser (del acrónimo inglés LASER, ***light amplification by stimulated emission of radiation***, en español, luz amplificada por emisión de radiación estimulada) es un dispositivo que utiliza un efecto de la mecánica cuántica, la emisión inducida o estimulada, para generar un haz de luz coherente tanto espacial como temporalmente.

La coherencia espacial se corresponde con la capacidad de un haz para permanecer con un pequeño tamaño al transmitirse por el vacío en largas distancias y la coherencia temporal se relaciona con la capacidad para concentrar la emisión en un rango espectral muy estrecho.

Clasificación de láseres según UNE EN 60825-1/A2-2002

Según la peligrosidad de los láseres y en función del Límite de Emisión Accesible (LEA) se pueden clasificar los láseres en las siguientes categorías de riesgo:

- Clase 1: Seguros en condiciones razonables de utilización.
- Clase 1M: Como la Clase 1, pero no seguros cuando se miran a través de instrumentos ópticos como lupas o binoculares.
- Clase 2: Láseres visibles (400 a 700 [nm]). Los reflejos de aversión protegen el ojo aunque se utilicen con instrumentos ópticos.
- Clase 2M: Como la Clase 2, pero no seguros cuando se utilizan instrumentos ópticos.
- Clase 3R: Láseres cuya visión directa es potencialmente peligrosa pero el riesgo es menor y necesitan menos requisitos de fabricación y medidas de control que la Clase 3B.
- Clase 3B: La visión directa del haz es siempre peligrosa, mientras que la reflexión difusa es normalmente segura.
- Clase 4: La exposición directa de ojos y piel siempre es peligrosa y la reflexión difusa normalmente también. Pueden originar incendios y explosiones.

La FDA[1] de los EE. UU. determinó que la clase láser 3 podría provocar lesiones a los ojos si la ve directamente por aproximadamente 0,25 segundos, aunque se ha citado la evidencia de que la exposición a los rayos láser visible es "generalmente" limitada por el reflejo de parpadeo del ojo, que se han programado en estudios recientes, justo por debajo de 0,25 segundos.

[1] *FDA del acrónimo en inglés (Food and drug administration) es la agencia del gobierno de los Estados Unidos responsable de la regulación de alimentos, medicamentos, cosméticos, aparatos médicos, productos biológicos y derivados sanguíneos.*

Luminosidad del rayo láser

El brillo y luminosidad de un rayo láser, no depende solamente de la potencia óptica del láser, también depende de la respuesta cromática del ojo humano, esto hace que por la misma potencia óptica, el láser de color verde parezca más brillante que otros colores porque el ojo humano es más sensible a bajos niveles de luz en la región verde del espectro (longitud de onda de 520 a 571 nm), se observa el espectro de luz visible en la **Fig. Nº 2**.

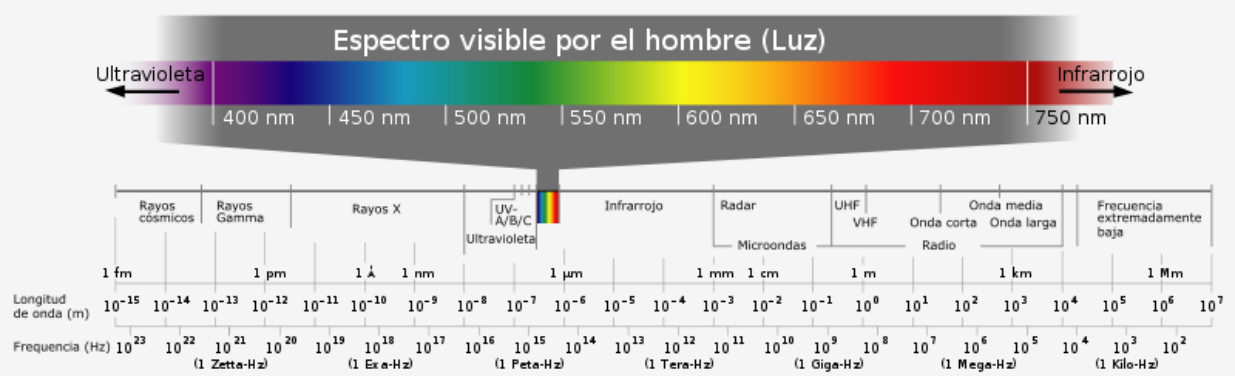

Fig. Nº 2 *Espectro de luz visible.*

1.2 Modulación PWM

La modulación por ancho de pulsos (también conocida como PWM, siglas en inglés de **pulse-width modulation**) de una señal o fuente de energía es una técnica en la que se modifica el ciclo de trabajo de una señal periódica (una cuadrada, por ejemplo), ya sea para transmitir información a través de un canal de comunicaciones o para controlar la cantidad de energía que se envía a una carga.

El ciclo de trabajo de una señal periódica es el ancho relativo de su parte positiva en relación con el período. Expresado matemáticamente en la **Ecu. 1**, esto gráficamente se observa en la **Fig. Nº 3**.

$$D = \tau / T \qquad \text{(Ecu. 1)}$$

- D es el ciclo de trabajo.
- τ es el tiempo en que la función es positiva (ancho del pulso).
- T es el período de la función.

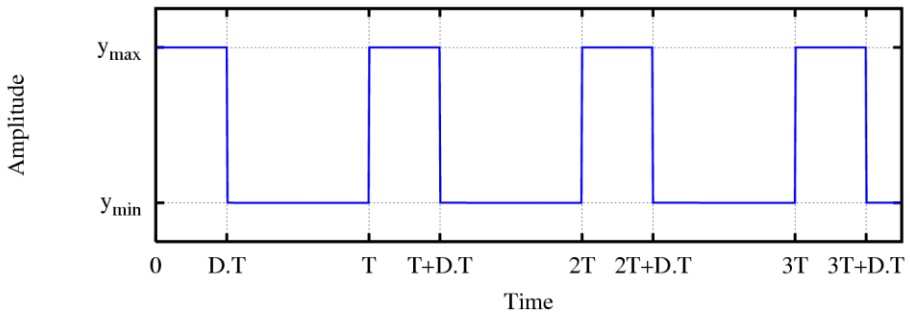

Fig. Nº 3 *Una señal de onda cuadrada, mostrando el ciclo de trabajo* ***D***.

1.3 Multiplexado de display 7 segmentos

La multiplexación es la técnica de combinar dos o más señales, y transmitirlas por un solo medio de transmisión. En electrónica, la multiplexación se refiere al mismo concepto si se trata de buses de datos que haya que compartir entre varios dispositivos, en este caso, display 7 segmentos.

La multiplexación en displays, simplemente consiste en encender un único display, mostrar el número y luego apagarlo, para encender el display siguiente. El truco de encender y apagar el display a una alta velocidad, permite engañar al ojo humano, y tener la sensación de que todos los displays se encuentran energizados al mismo tiempo, pero en realidad, si lo vemos en cámara lenta, los displays de 7 segmentos de ánodo o cátodo común están haciendo la secuencia de multiplexación.

1.4 Reflexión de la luz

La reflexión ocurre cuando los rayos de luz que inciden en una superficie chocan en ella, se desvían y regresan al medio que salieron formando un ángulo igual al de la luz incidente.

Es el cambio de dirección, en el mismo medio, que experimenta un rayo luminoso al incidir oblicuamente sobre una superficie, **Fig. Nº 4**. Para este caso las leyes de la reflexión son las siguientes:

- 1a. ley: El rayo incidente, el rayo reflejado y la normal, se encuentran en un mismo plano.
- 2a. ley: El ángulo de incidencia es igual al ángulo de reflexión.

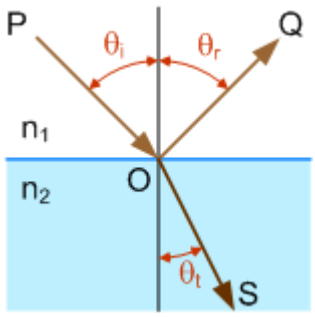

Fig. Nº 4 *Reflexión de la luz.*

2. Desarrollo de hardware

En este apartado, se presenta el desarrollo del hardware, con las 5 placas PCB[2] que se construyeron, las cuales forman parte del sistema mostrado en la **Fig. Nº 1**.

Se observa el diagrama general de conexión de estas placas en la **Fig. Nº 5**.

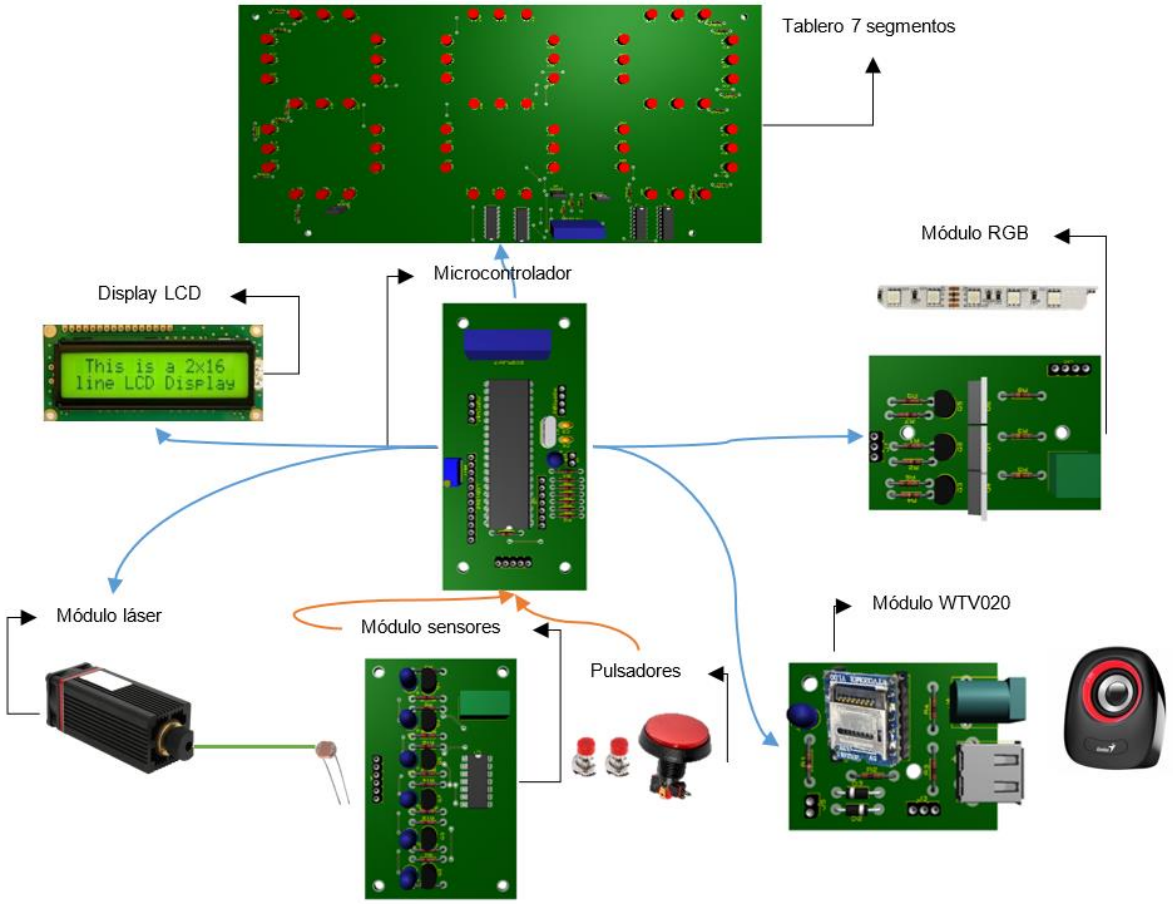

Fig. Nº 5 *Diagrama general con módulos del sistema.*

Unidad central de procesamiento

Es la placa que está en el centro, esta contiene el microcontrolador PIC, componente principal, encargado de controlar los distintos módulos de entrada y salida.

Módulos de entradas digitales

Estos módulos se observan unidos al microcontrolador a través de flechas color naranja.

[2] *PCB: acrónimo de Printed Circuit Board (placa de circuito impreso).*

1. Módulo de sensores: Esta plaqueta maneja los sensores LDR, estos sirven para detectar las interrupciones de luz proveniente de los láseres.
2. Pulsadores: Estos no cuentan con ningún PCB, están conectados a través de cables, y sirven para operar y configurar los distintos parámetros del juego, así como activar y desactivar el cronometro del juego.

Módulos de salidas digitales

Estos módulos se observan unidos al microcontrolador a través de flechas color celeste.

1. Módulos láser: Para crear la red de rayos, se usan seis emisores de luz en conjunto con seis espejos para obtener doce rayos de luz que cruzan la sala de juego.
2. Módulo de audio: Encargado de reproducir un sonido mp3, al recibir la señal lógica del PIC, en un parlante cuando se detecta una interrupción de algún haz.
3. Módulo controladora RGB: Controladora de luz para el cartel **LASER ZARPY**.
4. Tablero 7 segmentos de tres dígitos: Donde se muestra el tiempo de cada equipo.
5. Display LCD que sirve para mostrar mensajes e información a través de caracteres, números o símbolos.

De acuerdo con el diagrama en bloques mostrado en la **Fig. Nº 1** se procederá a explicar detalladamente la función que cumple cada módulo en el sistema.

2.1 Módulos láser

Se utilizaron 6 láseres de color verde **Fig. Nº 6**, las características principales se muestran en la **tabla I**. Estos componentes son los encargados de crear la red de rayos visibles en conjunto con espejos.

Fig. Nº 6 *Módulo láser verde.*

Módulo láser	
Longitud de onda	532 [nm]
Potencia	50 [mW]
Alimentación	12 [V]
Habilitación	TTL

Tabla I: *Características básicas de los módulos láser.*

Para controlar el encendido del láser, se necesitan $5\,[V]$, se utilizará el pin C3 del microcontrolador PIC16F887, este pin es el número 18.

2.2 Módulo de audio

Módulo WTV020M01 y parlante utilizados, se muestran en la **Fig. Nº 7.** Las características básicas del mismo se detallan en la **tabla II**.

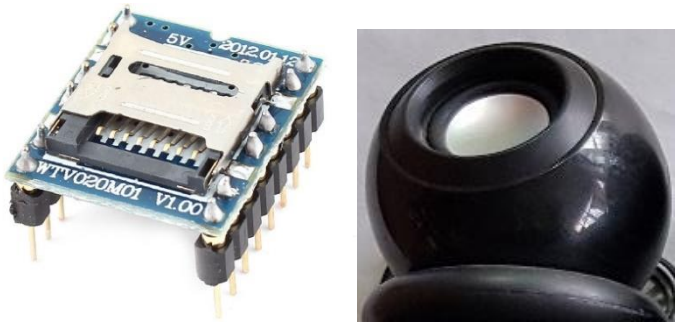

Fig. Nº 7 *Módulo WTV020 y parlante utilizado.*

Este módulo de voz necesita de una tarjeta de memoria microSD para almacenar los archivos de audios a reproducir, una consideración a tener en cuenta, es que el dispositivo lee ficheros de audio en formato .ad4, por lo cual, es necesario transformar el fichero .mp3 a dicho formato.

Módulo WTV020	
Alimentación	2.5 – 3.6 [V]
Tarjetas microSD	Hasta 1 GB
Formato de archivo	AD4 y WAV

Tabla II: *Características básicas del módulo WTV020.*

Las conexiones se muestran en la **Fig. Nº 8**, las entradas **PIN_C5, PIN_C6 y PIN_C7**, vienen del microcontrolador, los pines hacen referencia a datos, clock y reset respectivamente.

El circuito se alimenta con $5\ [V]$, pero como el módulo trabaja con menos tensión, se ponen en serie dos diodos **1N4007** para reducir el voltaje a $3.6\ [V]$. El LED conectado es indicativo, este se enciende cuando el modulo está reproduciendo audio.

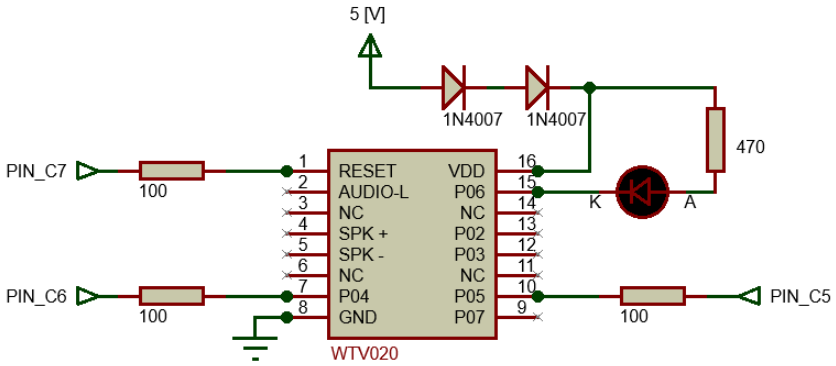

Fig. Nº 8 *Conexiones básicas módulo WTV020.*

2.3 Módulo para tiras RGB

Para iluminar el cartel del juego, se usarán tiras led RGB 5050, y se utilizarán mosfet de potencia, estos componentes, nos permiten manejar corrientes más elevadas.

Los mosfet cuentan con tres terminales de salida, que se llaman: Drain, Source y Gate. La corriente principal pasa entre Source y Drain (Isd) mientras que el control de esta corriente se obtiene aplicando una tensión sobre el terminal Gate (respecto al terminal Source) conocida como V_{GS}.

Existen "*power mosfets*" de dos tipos: los de canal N y los de canal P, **Fig. Nº 9**.

La diferencia entre estos está en la polaridad de conexión Source - Drain y en el hecho que la tensión de Gate de los mosfet de canal P es negativa (las mismas diferencias que existen entre los transistores NPN y PNP).

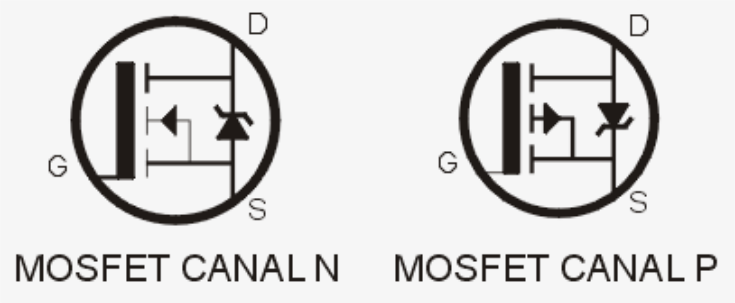

Fig. Nº 9 *Símbolos de los mosfet canal n y p.*

En base a la aplicación, un mosfet de potencia puede trabajar en la "región lineal" o en

"saturación". En los sistemas analógicos, como por ejemplo, en las etapas de salida de los amplificadores de audio, los mosfet trabajan en la región lineal mientras que en los sistemas digitales, en los cuales se usan como interruptores digitales de potencia, estos trabajan en corte (OFF) o saturación (ON).

Cuando un mosfet se encuentra en saturación, el valor de resistenca interno entre Source y Drain (Rsd) es muy bajo y por lo tanto, la disipación de potencia en él será poco significativa no obstante la corriente que lo atraviesa pueda ser muy elevada.

Para llevar un mosfet a la saturación, es necesario que la tensión de control en el terminal Gate sea suficientemente alta **Fig. Nº 10** y esto podría ser un problema si usáramos directamente la baja tensión de salida de un microcontrolador. Por ejemplo, para saturar un transistor bipolar (tipo BC548) se necesita superar la tensión de umbral de la base que es aproximadamente $0.6\ [V]$.

Este valor se puede obtener tranquilamente con un microcontrolador, por el contrario, la tensión necesaria para poner en conducción un mosfet (llamada "tensión de umbral" V_{TH}) es mucho más elevada (algunos volts).

Aunque alcanzando este valor, no sería suficiente, ya que deberíamos salir de la región lineal de trabajo para llevarlo a la saturación. Si no fuera así, la conducción no sería plena y por lo tanto parte de la potencia se disiparía en el mosfet en forma de calor, según la Ecu. 1.

$$P_{MOSFET} = V_{SD} * I_{SD} \qquad \textit{(Ecu. 1)}$$

Donde V_{SD} es la caída de tensión entre la fuente y drenador, e I_{SD} la corriente que pasa a través de él.

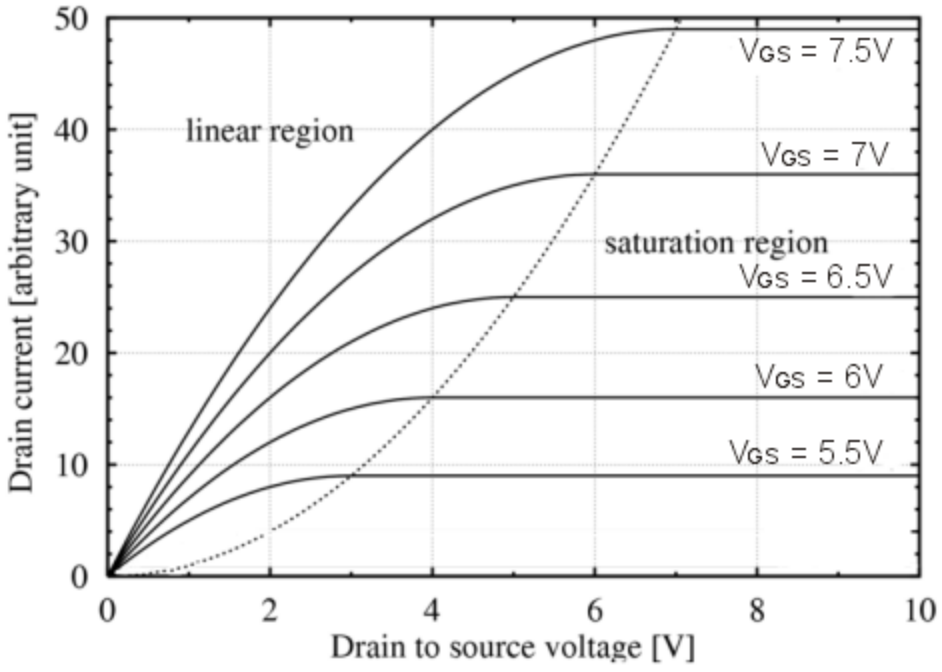

Fig. Nº 10 *Curvas de conducción de un MOSFET con las dos regiones de trabajo (lineal y saturación) para distintas tensiones de Gate.*

Se muestra la conexión del circuito en la **Fig. Nº 11** para un solo color, para los otros dos se sigue el mismo principio.

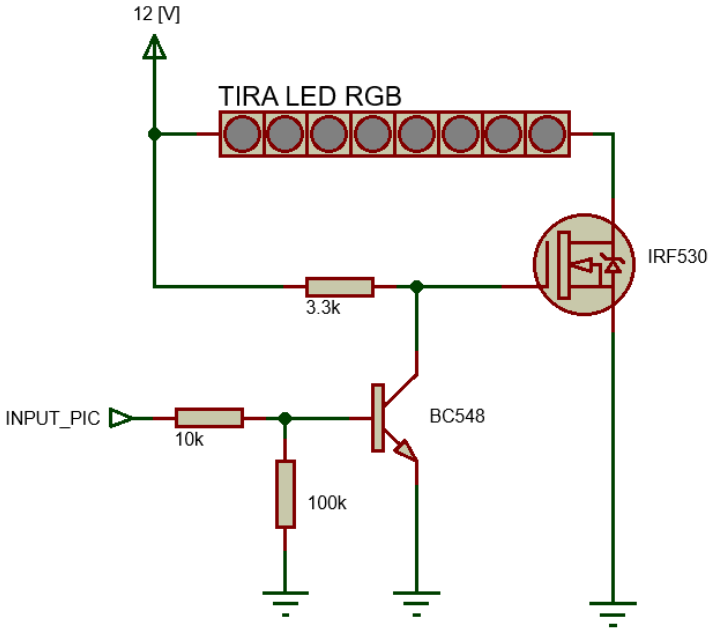

Fig. Nº 11 *Conexiones con módulo RGB.*

Cuando la salida del microcontrolador tiene un nivel lógico bajo $0\ [V]$, el transistor **BC548** no conduce y por lo tanto, su colector, que se encuentra conectado al Gate del mosfet tendrá un potencial positivo de $12\ [V]$ a través de la resistencia a positivo.

Cuando la salida del microcontrolador pasa a nivel alto $5\ [V]$, el transistor conduce y lleva el Gate del mosfet a $0\ [V]$, por lo tanto el mosfet deja de conducir.

Este circuito tiene el defecto que trabaja al contrario, es decir, se activa cuando el nivel de salida del microcontrolador es bajo.

No obstante esto, tiene la ventaja que la tensión de Gate alcanza la tensión máxima de alimentación garantizando la completa saturación de cualquier tipo de mosfet que conectemos.

Cuando cambia el nivel lógico de control, por un instante el mosfet absorbe una cierta corriente que carga el capacitor interno del terminal Gate. La resistencia de $3.3\ [k\Omega]$ sirve para limitar esta corriente inicial. Un valor bajo permite la carga rápida de este capacitor y por lo tanto una conmutación más veloz del mosfet, útil si quisiéramos usar una regulación de potencia por impulsos (PWM).

2.4 Tablero RGB

Para mostrar el tiempo de juego de cada equipo, se implementó un tablero 7 segmentos de 3 dígitos con leds RGB, esto para facilitar la visualización de lectura a los jugadores, si el tablero está prendido de color azul, indica el tiempo para el equipo azul, y viceversa, si el mismo está en color rojo, este representa el tiempo para el equipo rojo.

El tablero cuenta con las dimensiones de $300\ x\ 150\ [mm]$, mientras que cada segmento tiene un tamaño de $70\ x\ 120\ [mm]$, por cada segmento se utilizarán 3 led RGB de 5mm, ánodo

común y conexionados entre sí en paralelo.

Los valores de resistencia utilizados, se calcularon para cada color, teniendo en cuenta la caída de tensión de la red.

Para el color azul por ejemplo, tenemos como dato que cada led consume $I_F = 20\ [mA]$ y en caso de la tensión $V_F = 3.5\ [V]$.

Teniendo en cuenta la tensión de alimentación $V_{CC} = 5\ [V]$ y la caída de tensión en el transistor BC548 de aproximadamente $0.7\ [V]$, aplicando la ley de Kirchhoff y la ley de Ohm, obtenemos el valor de resistencia correspondiente.

$$R_{Azul} = \frac{\Delta V}{I_F} = \frac{5\ [V] - 3.5\ [V] - 0.7\ [V]}{20\ [mA]} = 40\ [\Omega] \qquad \textit{(Ecu. 2)}$$

En caso del color rojo, teniendo como dato $V_F = 2.9\ [V]$ e igual $I_F = 20\ [mA]$, obtenemos un valor de resistencia necesaria $R_{Rojo} = 70\ [\Omega]$.

En la práctica, estos valores pueden diferenciar, debido a la caída de tensión de los cables y conectores.

Para controlar el tablero RGB, necesitamos utilizar del PIC 11 pines, de los cuales 4, son para comandar el color azul, otros 4 para el color rojo y 3 pines para el multiplexado, se muestra en la **Fig. Nº 12**, la conexión entre integrados CD4511, ULN2003A y resistencias, la explicación se hará para un solo color, el funcionamiento es idéntico para el otro color.

El CD4511 es un decodificador BCD a 7 segmentos, este tiene un circuito lógico combinacional que acepta un digito decimal en BCD en la entrada (4 bits de entrada del PIC) y genera salidas en binario, apropiadas para controlar 7 segmentos, esto nos permite ahorrar 3 pines del microcontrolador por color.

Estos pines de salida, se conectan a un ULN2003, el mismo está compuesto internamente por 7 drivers independientes entre sí, de esta forma podemos darle la referencia a GND a cada segmento que queramos encender, estos se conectan a través de una resistencia.

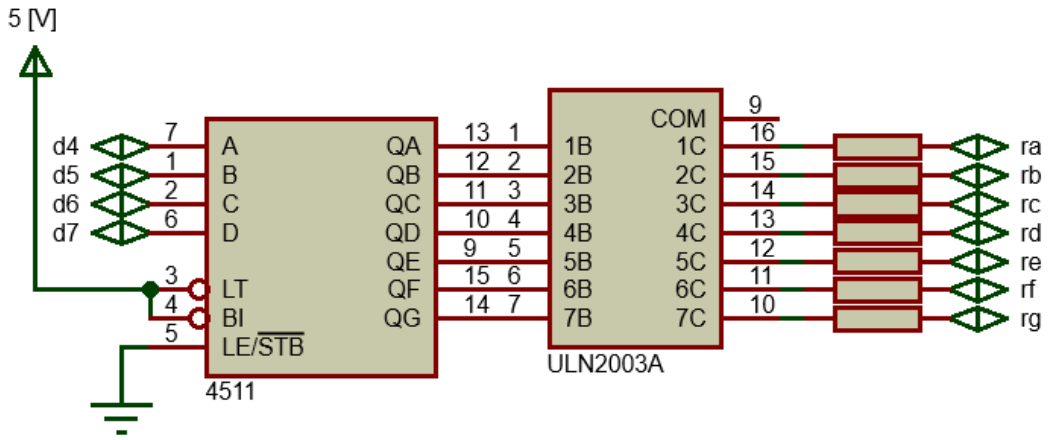

Fig. Nº 12 *Conexiones básicas tablero RGB.*

Un transistor BC548 se usa como llave electrónica, se muestra en la **Fig. Nº 13**, la base del mismo está conectada a través de una resistencia a un pin del μC. El colector está conectado al positivo de $5\ [V]$, y el emisor conectado al ánodo de los leds de los 7 segmentos de cada display, y cómo se mencionó más arriba, el cátodo de los leds están conectados a través de una resistencia a los pines de salida del ULN2003A.

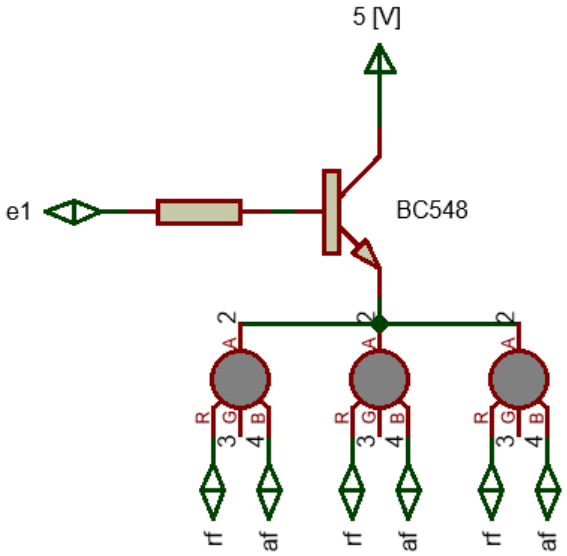

Fig. Nº 13 *Conexiones básicas tablero RGB.*

2.5 Display LCD

Se utilizará un display LCD de 16x2 caracteres de fondo verde y caracteres negros con retroiluminación led de color verde.

El mismo servirá de interfaz gráfica entre el microcontrolador y el operador, se irán mostrando los parámetros a configurar.

Alguna de las características básicas del mismo, se detallan a continuación:

- Tipo Green Backlight (Fondo verde con letras negras).
- Formato 16 caracteres por línea, 2 líneas en total.
- Alimentación $5\ [V]$.

En la **tabla III** se detalla el significado de cada pin del LCD.

Display LCD 16x2	
V_{SS}	Tierra
V_{DD}	Tensión de alimentación
V_{EE}	Tensión para regular contraste
R_S	Bit de instrucción
R/W	Bit de lectura/escritura
E	Bit de habilitación
D_4	Bit de datos
D_5	Bit de datos
D_6	Bit de datos
D_7	Bit de datos
A	Ánodo de led luz de fondo de pantalla
K	Cátodo de led luz de fondo de pantalla

Tabla III: *Pines LCD 16x2.*

Se integró en el PCB del microcontrolador una parte dedicada al control del display LCD, consta de una tira de pines junto a una resistencia variable para modificar el brillo de la pantalla, 6 pines del puerto B, son utilizados para mandar datos y controlar el LCD, las conexiones se muestran en la **Fig. Nº 14**.

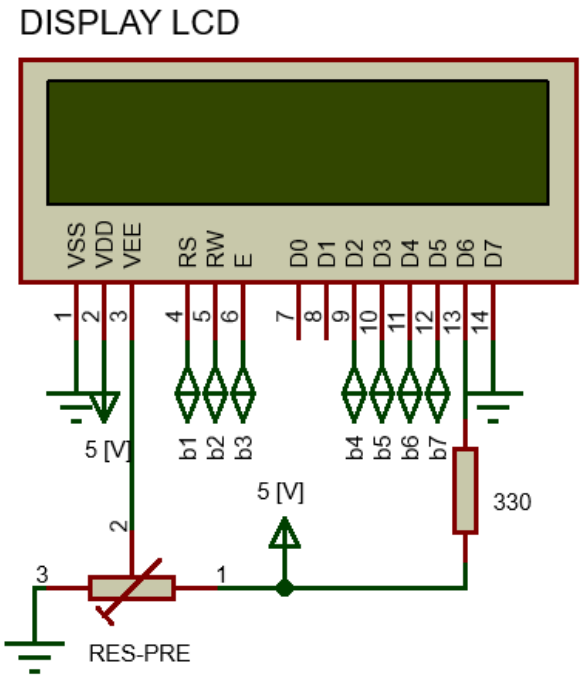

Fig. Nº 14 *Conexiones básicas para display LCD.*

2.6 Módulo de sensores

Para detectar las interrupciones de los haces de luz, se utilizaran sensores LDR (Light-dependent resistor).

Un fotorresistor o fotorresistencia es un componente electrónico cuya resistencia se modifica, (normalmente disminuye) con el aumento de intensidad de luz incidente. Su cuerpo está formado por una célula fotorreceptora y dos patas, como los que se muestran en la **Fig. Nº 15**.

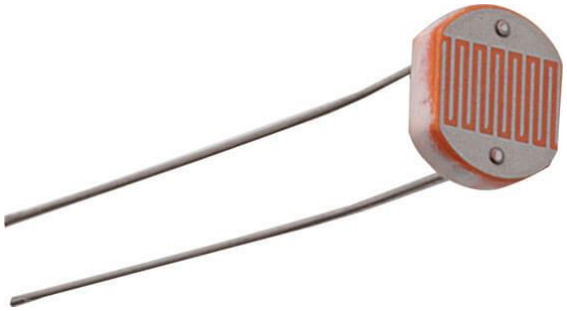

Fig. Nº 15 *Fotorresistor.*

Este módulo es el encargado de avisarle al microcontrolador cuándo un haz de luz se interrumpe. Las interrupciones de luz láser se detectan por medio de 6 sensores LDR[3], el funcionamiento de la lógica se describe brevemente a continuación.

En la **Fig. Nº 16** vemos representado parte del módulo, se explicará para un sensor, el funcionamiento es idéntico para los demás.

Cuando el sensor LDR recibe luz, su impedancia interna es tan baja que permite el flujo de corriente a través del mismo, esto provoca que la tensión en la base del transistor sea insuficiente para polarizarlo, haciendo que la salida s_x esté en $0\ [V]$.

Si el sensor LDR no recibe luz, la base del transistor se pone a $5\ [V]$, esto provoca que el mismo se polarice y que haya una conducción de corriente, encendiendo el led y poniendo la salida s_x a $5\ [V]$.

Este pin de salida s_3 está conexionado a una entrada de la compuerta OR CD4072, esta compuerta al tener una entrada en valor alto, lleva la salida a valor alto.

La salida de la compuerta está conectada al pin RB_O del microcontrolador.

[3] *LDR: acrónimo de Light-dependent resistor (Resistencia dependiente de luz).*

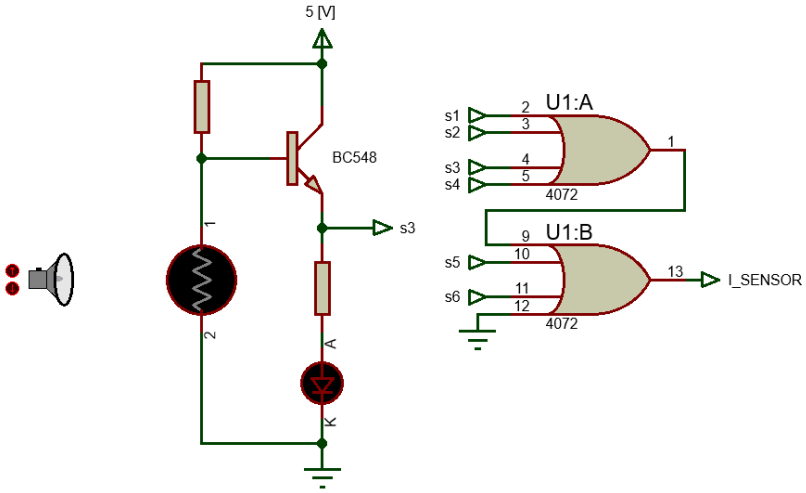

Fig. Nº 16 *Módulo para sensores.*

2.7 Pulsadores

Para interactuar con el juego, se utilizaran 3 pulsadores, de los cuales 2 los usará el operador, y servirán para configurar los parámetros y tener control del juego en sí, el restante lo utilizaran los jugadores, este servirá para iniciar y detener el cronometro de juego, se muestran los pulsadores utilizados en la **Fig. Nº 17**.

Fig. Nº 17 *Pulsadores utilizados.*

2.8 Fuente de alimentación

Se utilizará una fuente ATX de 300W **Fig. Nº 18** (“Advanced Technology eXtended”), esta se encargará básicamente de transformar la corriente alterna de la línea eléctrica $220\ [V]$ en corriente directa; así como reducir su voltaje a los valores necesarios para alimentar el hardware del juego, por ejemplo, la salida de $12\ [V]$ para los módulos laser, $5\ [V]$ para el resto del hardware detallado anteriormente y GND $0\ [V]$ como referencia.

Se eligió una fuente de este tipo, ya que son confiables, duraderas, robustas, y tienen sus protecciones internas en caso de variaciones en el voltaje de la red.

Se harán modificaciones para que la fuente se encienda automáticamente al recibir tensión de la red eléctrica, también se quitarán los cables de la placa que no se utilicen, dejando solo aquellos con las tensiones a utilizar. La fuente se colocará en un lugar seguro dentro de la habitación, quedando fija y se encenderá con un interruptor/llave de luz puesto en un lugar de fácil acceso para el operador.

Fig. Nº 18 *Fuente ATX.*

2.9 Microcontrolador

Como unidad central de control y procesamiento se utilizará un PIC16F887 de Microchip Technology, capaz de realizar diferentes actividades que requieran del procesamiento de datos digitales y del control y comunicación digital de diferentes dispositivos. Haciendo uso de estas prestaciones el PIC procesa información cargada por el usuario, controla el ON/OFF de los módulos láser, lee la medición de los sensores y controla un tablero digital, unas tiras de luces led RGB y un módulo de sonido. A continuación se describirá el PIC y los periféricos utilizados, luego se explicará el código de programación ejecutado por el mismo.

2.9.1 Acerca del controlador PIC16F887

Algunas de las características del controlador se listan a continuación, la elección del mismo se basa en algunas de estas características.

- El voltaje de alimentación operable es de 2.0V a 5.5V.
- La frecuencia de operación es de 0 a 20 MHz.
- Cuenta con un oscilador interno de alta precisión calibrado de fábrica.
- Tiene 35 pines de entrada/salida de alta corriente de fuente y de drenado.
- Tiene tres temporizadores/contadores independientes.
- Módulo PWM incorporado.

Se eligió la presentación en encapsulado PDIP de 40 pines, en la **Fig. Nº 19** se muestra el esquema del controlador con sus pines correspondientes.

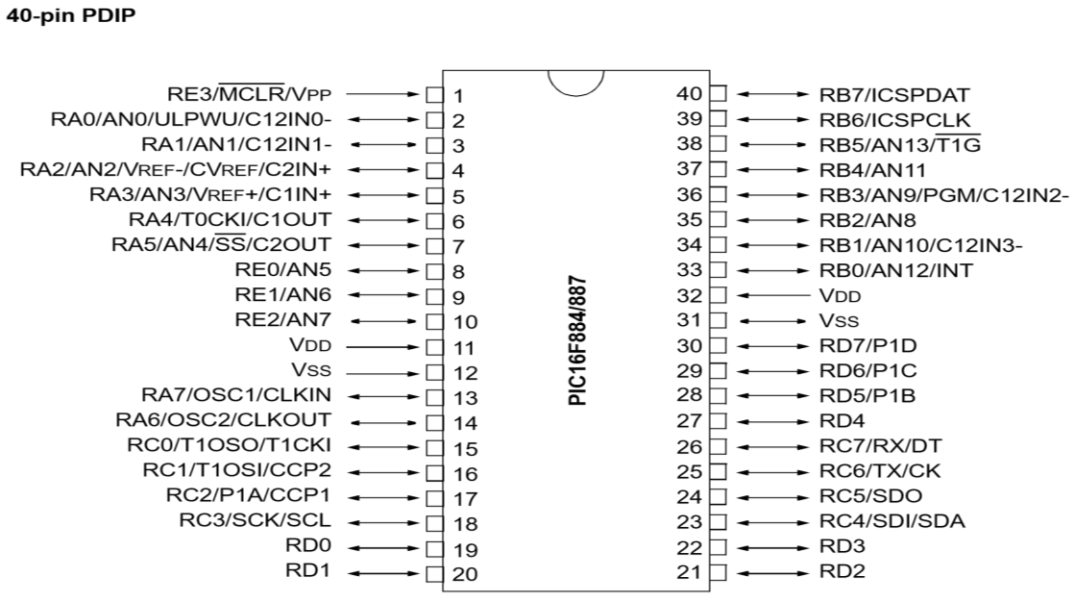

Fig. Nº 19. *Microcontrolador PIC16F887.*

2.10 Configuración del oscilador

El módulo de oscilador de este microcontrolador tiene una amplia variedad de fuentes de reloj y características de selección que permiten que el mismo sea usado en un amplio rango de aplicaciones mientras maximizan el rendimiento y disminuyen el consumo. En este caso se utilizará como fuente de reloj externa un cristal de cuarzo de 20 MHz, el circuito utilizado se puede observar en la **Fig. Nº 20**. El módulo de reloj será configurado en HS, modo adecuado para resonadores de alta frecuencia.

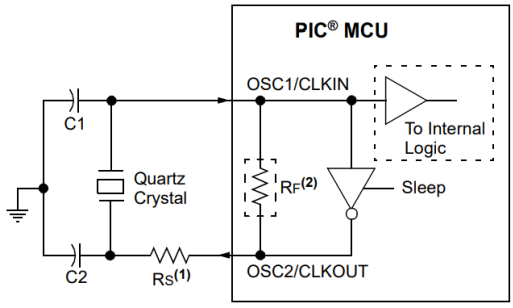

Fig. Nº 20. *Operación con cristal de cuarzo.*

Donde:

- C_1 y C_2 son capacitores de cerámica de $15\ pF$.
- R_s es utilizada para cristales de cuarzo de baja frecuencia, en este caso no se utiliza.
- OSC1/CLKIN y OSC2/CLKOUT equivalen a los pines 13 y 14 del PIC.
- Cristal de cuarzo de $20\ MHz$.

Cabe destacar que un ciclo de instrucción consta de cuatro periodos de oscilador, por lo tanto, para una frecuencia de $20\,MHz$, esto dará un valor normal de tiempo de ejecución de instrucción de $\frac{1}{\frac{F_{OSC}}{4}} = \frac{1}{\frac{20\,MHz}{4}} = 0.2\,us$.

2.11 Configuración del TIMER1

Se necesita un periodo de tiempo fijo, de 1 segundo, para lograr la aplicación de un cronometro, el mismo servirá para mostrar el tiempo de juego en el tablero RGB.

El TIMER1 del microcontrolador PIC16F887 **Fig. Nº 21** es un temporizador/contador de 16 bits al que se accede a través del par de registros TMR1H:TMR1L. Cuando se usa con una fuente de reloj interna, el módulo funciona como temporizador, incrementando dicho par de registros en múltiplos de F_{OSC} según lo determine el preescaler, que puede establecerse como divisor en 1, 2, 4 u 8 de la entrada del reloj. Cuando dicho registro supera el valor de $FFFF_h$ produce una interrupción por desbordamiento.

Para calcular el tiempo de desbordamiento del TIMER1, se utiliza la siguiente ecuación:

$$T = T_{CM} * Prescaler * (65536 - Carga\,TMR1) \qquad \textit{(Ecu. 3)}$$

Donde $T_{CM} = \frac{4}{F_{OSC}}$ y $F_{OSC} = 20\,MHz$.

Teniendo en cuenta la ecuación Ecu 17, y utilizando un prescaler de 8, podemos deducir que el tiempo máximo de desbordamiento es de $\cong 104.8\,ms$.

Para tener un periodo de 1 segundo, una opción es definir un $T = 100\,ms$ y repetirlo 10 veces, entonces para lograr esto usamos un prescaler de 8 y cargamos al TIMER1 con un valor de 3036.

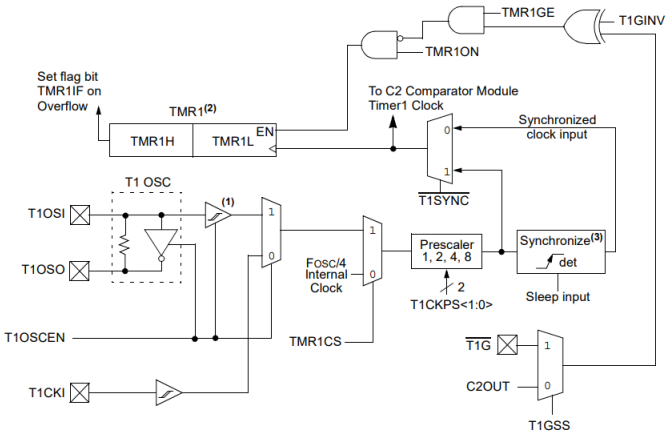

Fig. Nº 21. *Operación con cristal de cuarzo.*

2.12 Modo PWM con TIMER2

Se necesita controlar unas luces led RGB de un cartel y lograr un efecto llamado “FADE”, esto significa aumentar y disminuir el brillo de las luces de manera gradual y progresiva.

El modo PWM (Pulse Width Modulation) o MODULACIÓN DE ANCHO DE PULSO, permite

obtener en los pines CCPx una señal periódica en la que se puede modificar su ciclo de trabajo (Duty Cicle). Es decir, puede variarse el tiempo en el cual la señal está a nivel alto (T_{ON}) frente al tiempo que está a nivel bajo (T_{OFF}), ver **Fig. Nº 22**.De esta forma, la tensión media aplicada a la carga es proporcional al T_{ON}, haciendo uso de este principio, podremos lograr el efecto FADE.

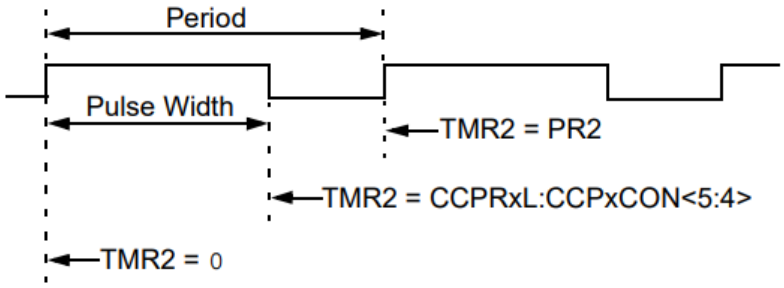

Fig. Nº 22 *Señal PWM.*

El periodo de la señal PWM se obtiene de configurar el TIMER2 y el contenido del registro PR2. Para calcular el periodo de la señal PWM se utiliza la siguiente ecuación (Ecu. 4).

$$PWMT = (PR2 + 1) * 4 * T_{OSC} * (Valor\ del\ preescaler\ TMR2) \qquad \textit{(Ecu. 4)}$$

Si queremos trabajar con un periodo de $PWMT = \frac{1}{10\ k\ Hz}$, con un preescaler de 4, debemos averiguar el valor de PR2, haciendo uso de la ecuación, obtenemos que el mismo tiene que ser $PR2 = 124$. Para la aplicación FADE en el cartel led RGB, tenemos que variar el ciclo de trabajo entre $(0 - 100)\%$ del total del periodo. Esto significa, que si el ciclo de trabajo de la onda PWM es mayor que el periodo (100%), la señal que sale por el pin CCP_x estará siempre en 1.

2.13 Configuración interrupción RB0

En el momento de juego, se necesita que ante un corte de luz láser, suene la alarma y se decremente el cronometro de manera instantánea.

Con las interrupciones en un microcontrolador, lograremos que el dispositivo pare de repente la tarea que está realizando para que haga o realice una función con urgencia y después continúe haciendo su rutina habitual.

Haciendo uso de la interrupción RB0, común a la mayoría de los PIC, esta permite generar una interrupción tras el cambio de nivel de alto a bajo o de bajo a alto en la entrada RB0 leyendo los flancos de bajada, o flanco de subida respectivamente **Fig. Nº 23**.

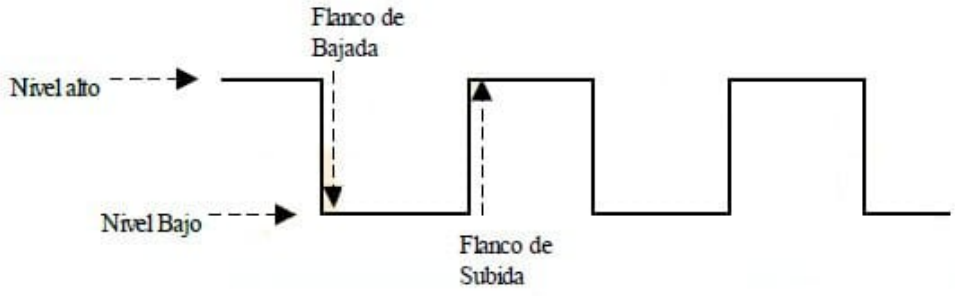

Fig. Nº 23 *Flanco de bajada/Flanco de subida.*

2.14 Conexión del microcontrolador

El microcontrolador se alimentará con 5 [V], en la **Fig. Nº 24** se observan las conexiones del microcontrolador con los periféricos y módulos, se destaca:

- Conector llamado “PICKIT2” utilizado para facilitar la conexión para programar el PIC.
- Led D1 indica encendido del PIC.
- Se observan etiquetas en los pines del PIC al lugar donde están conexionados.

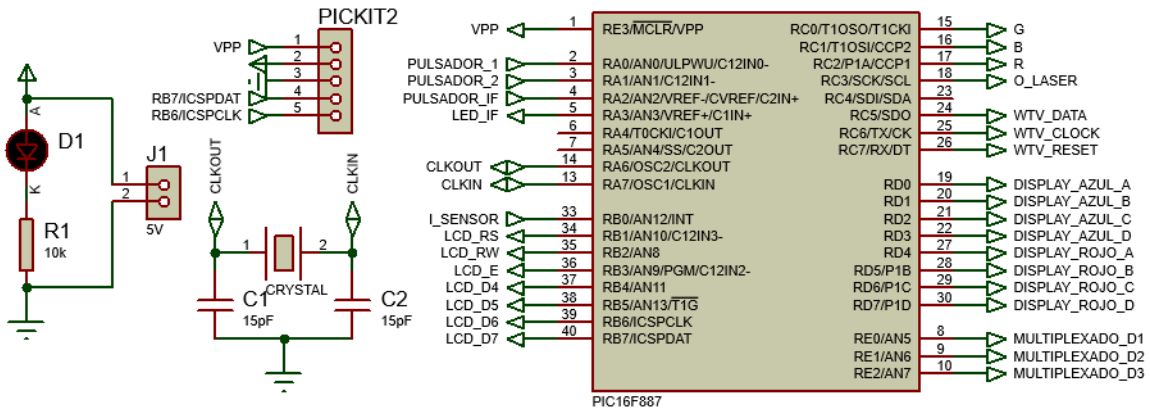

Fig. Nº 24 *Módulo microcontrolador.*

3. Desarrollo de firmware

El firmware es la otra parte importante del sistema mostrado en la **Fig. Nº 1**, este es el soporte lógico e intangible que hace posible la realización de tareas específicas del hardware.

El código que ejecuta el PIC fue escrito en lenguaje C.

A grandes rasgos puede decirse que el código cuenta con una rutina principal y funciones o subrutinas que son llamadas a lo largo del programa.

3.1 Subrutinas

Para cada subrutina, se hizo una función VOID, las mismas se describen a continuación.

3.1.1 Configuración

Esta subrutina de configuración está destinada a setear los puertos como entrada o salida, establecer salidas en alto o bajo, iniciar LCD, configurar TIMER1, TIMER2 y habilitar interrupciones, como se observa en la **Fig. Nº 25**.

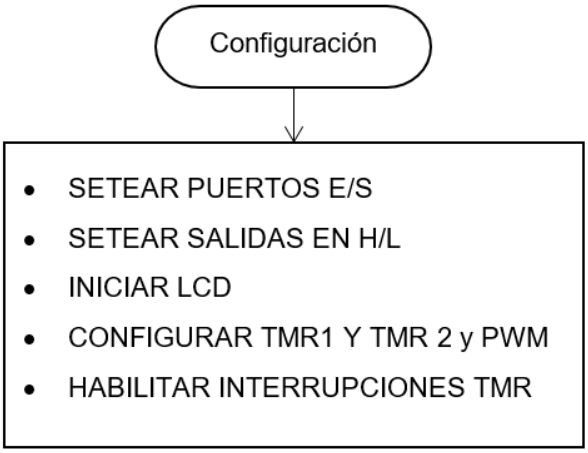

Fig. Nº 25 *Subrutina configuración.*

Como primer punto seteamos los puertos como entradas o como salidas, como se observa en la imagen **Fig. Nº 13** los puertos se definieron:

- $set\_tris\_a(0b00000111)$
- $set\_tris\_b(0b00000001)$
- $set\_tris\_c(0b00000000)$
- $set\_tris\_d(0b00000000)$
- $set\_tris\_e(0b0000)$

Los bits más significativos están a la izquierda $(X_7X_6X_5X_4X_3X_2X_1X_0)$ y un 1 equivale a

entrada, y un 0 como salida.

En cuanto a los bits configurados como salidas se llevan a cero con la función $output\_x(0)$ excepto los bits $C_2, C_1\ y\ C_0$ correspondientes al control de tiras led RGB, que se setean en estado alto.

El LCD se inicia con la función $lcd\_init()$ y las interrupciones se habilitan con $enable\_interrupts(GLOBAL)$.

En cuanto al $TMR1$ este se configura de la siguiente manera:

- $setup\_timer\_1(T1\_INTERNAL|T1\_DIV\_BY\_8)$ define el preescaler en 8.
- $set\_timer1(3036)$ recarga la variable del timer1
- $enable\_interrupts(INT\_TIMER1)$

El $TMR2$ y el módulo PWM se configura de la siguiente manera:

- $setup\_timer\_2(T2\_DIV\_BY\_4,124,1)$ se define el preescaler en 4, el registro PR2 en 124 y el postcaler en 1.
- $enable\_interrupts(INT\_TIMER2)$
- $setup\_ccp\_1(CCP\_PWM)$
- $setup\_ccp\_2(CCP\_PWM)$

3.1.2 Función RGB

Esta subrutina está destinada a controlar el color y efecto de luz en el cartel led RGB a través de las salidas PWM, se observa el diagrama en bloques en la **Fig. Nº 26**.

Para controlar el color verde se usa la función $output\_high()$ y $output\_low()$ apuntado al pin correspondiente. En cuanto al color rojo y azul, se utiliza la función $set\_pwm1\_duty(x)$ y $set\_pwm2\_duty(x)$ variando x entre 0 y 124 para mantener encendido o apagado al respectivo color. Cuenta con nueve modos de operación, se detallan a continuación:

- RGB 1 es el efecto FADE, en el cuál se logra una combinación variada de colores. En el primer paso se apaga el color verde, se mantiene el color rojo prendido y se enciende de forma gradual el color azul. Esto se hace con un ciclo FOR que varié el valor del PWM entre 124 y 0. Se utiliza el mismo principio para las demás combinaciones.
- RGB 2 es el efecto respiración de la luz roja, en donde está se enciende y apaga progresivamente.
- RGB 3 es igual al efecto anterior pero con el color azul.
- RGB 4 prende y apaga la luz rojo en tiempos de 300 milisegundos.
- RGB 5 igual al anterior pero en color azul.

- RGB 6 produce luz blanca.
- RGB 7 mantiene las luces apagadas.
- RGB 8 mantiene la luz en color azul.
- RGB 9 mantiene la luz en color rojo.

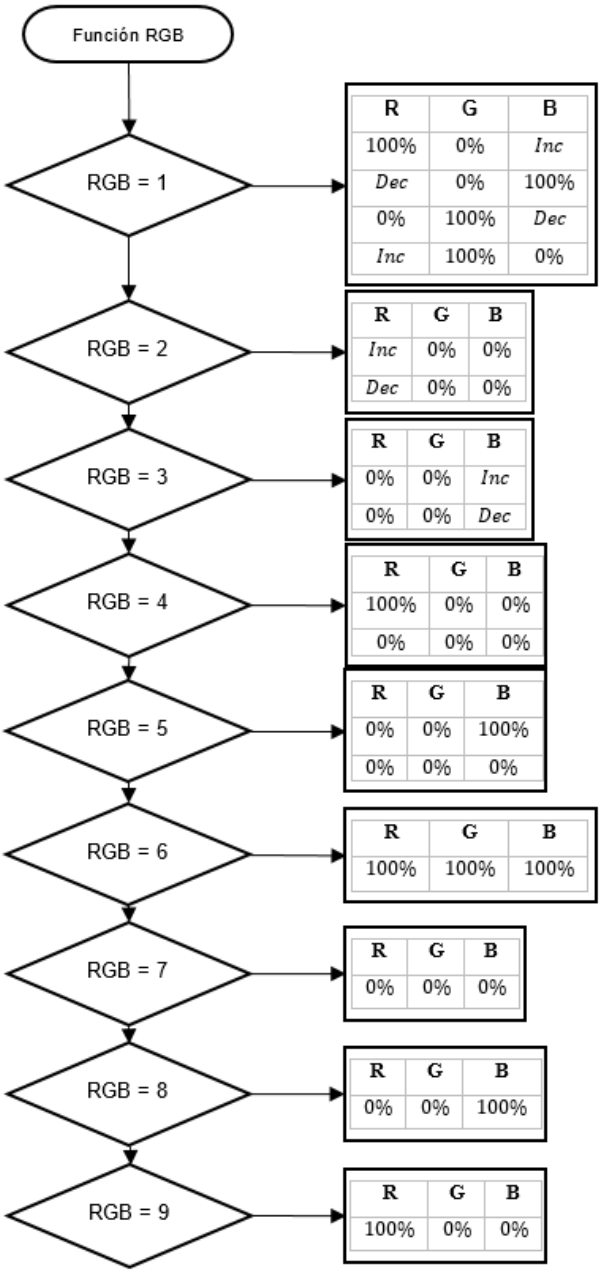

Fig. Nº 26 *Subrutina FRGB.*

3.1.3 Print

Esta subrutina está destinada a controlar lo que se muestra en el LCD 16x2, utilizando las siguientes funciones podemos mostrar en el display los caracteres deseados.

- $lcd\_init()$ se utilizó para iniciar la librería del lcd 16x2.
- $lcd\_gotoxy(x, y)$ para posicionar la escritura del primer carácter deseado.
- $lcd\_putc()$ para escribir los caracteres que se mostrarán.

3.1.4 Cambio

Esta subrutina se activa cuando los jugadores presionan el pulsador por segunda vez luego de haber superado el laberinto, esto es para que cambie el turno de juego para el otro equipo, se activa además una señalización de luz indicando el color del equipo que tiene que jugar, se observa el diagrama de bloques en la **Fig. Nº 27**.

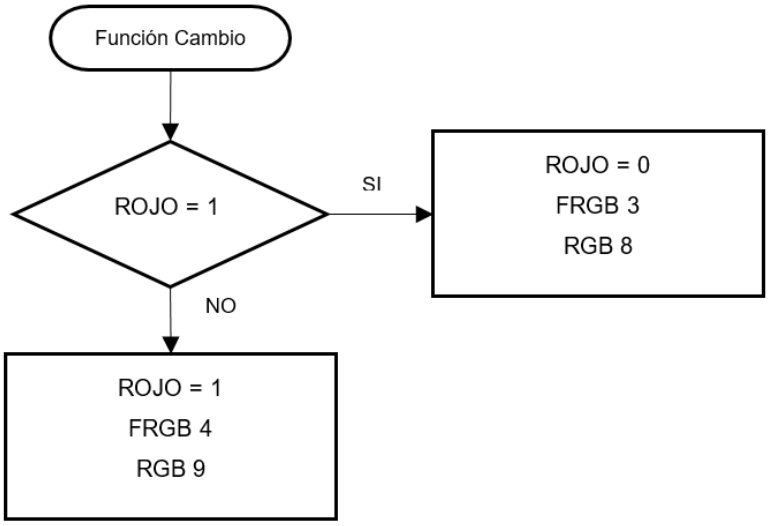

Fig. Nº 27 *Subrutina cambio.*

3.1.5 Menu

En esta subrutina se configuran las variables temporales del juego, en primer lugar el tiempo de juego para los equipos, este puede ser de 600 segundos o 900 segundos, y en segundo lugar se configura la penalización al interrumpir el haz, que puede ser de 5, 10, 15 o 20 segundos.

3.1.6 Reset

Esta subrutina sirve para resetear las variables y estado de pines utilizados en el programa, se llevan a su estado inicial, sirve para cuando termina el juego o para reiniciarlo en cualquier

momento, sin necesidad de estar apagando y encendiendo todo el sistema.

3.1.7 Ready

Esta subrutina está hecha para mostrar las variables temporales elegidas y confirmar si desea continuar e iniciar el juego, o cancelar y volver a elegirlas.

3.1.8 Game

En esta subrutina se encienden los láseres, el led del pulsador, setea las variables temporales y habilita la interrupción $RB0$.

3.1.9 Cronometro

Esta subrutina ejecuta la función del cronometo para ambos equipos, la misma se repite cada segundo, siempre que esté habilitado el juego, al presionar el pulsador de inicio.

El tiempo de cronometro se separa en 3 variables distintas, (centésima, décima y unidad). Cada segundo se decrementa la variable unitaria hasta hacerla cero, cuando sucede esto, se establece nuevamente en 9, y se decrementa la variable decimal, cuando ambas variables llegan a cero, hacen decrementar la centésima en una unidad, el mismo proceso se repite hasta que lleguen las tres variables a cero.

Cabe aclarar que la variable de cambio de equipo es la que define qué tiempo se decrementará.

En caso de que se interrumpa el haz de luz, se decrementa además la penalización correspondiente.

3.1.10 Multiplexado

Como se vio más arriba, el tablero tiene tres dígitos 7 segmentos formados por leds RGB, de los cuales se aprovechan los colores azul y rojo, en esta subrutina se realiza la multiplexación de estos 3 digitos utilizando solamente 11 pines del microcontrolador.

Para comenzar, se definen dos vectores de 10 posiciones, cada uno con la sentencia siguiente:

- $segr[10] = \{0x0F, 0x1F, 0x2F, 0x3F, 0x4F, 0x5F, 0x6F, 0x7F, 0x8F, 0x9F\}$
- $sega[10] = \{0xF0,0xF1,0xF2,0xF3,0xF4,0xF5,0xF6,0xF7,0xF8,0xF9\}$

Se explicará el proceso utilizando como ejemplo la multiplexación del color rojo, para el color azul es el mismo principio.

Analizando el vector $segr[10]$ vemos que cada posición, está entre comas, y cada una de estas, hace referencia a un número en hexadecimal, por ejemplo, la posición número 2, tiene el valor $0xF1$, que pasado a binario es $0b11110001$.

Observando los valores de las posiciones, vemos que se repite un patrón, donde todos los números en hexadecimal tienen la F al principio, esto equivale en binario a 1111, y que el otro

valor en hexadecimal varía entre 0 y 9, esto en binario equivale a $\{0000, 0001, \dots, 1001\}$, por lo tanto si usamos la sentencia $output\_d(segr[2])$, esto hace que los bits de salida del puerto d se carguen con el valor del vector de la posición 2, en hexadecimal es $0xF1$ y en binario, $1_{D7}1_{D6}1_{D5}1_{D4}0_{D3}0_{D2}0_{D1}1_{D0}$.

Como se vio en el hardware, los bits menos significativos apuntan al conversor BCD a 7 segmentos del color rojo, en este caso, se pintaría el número 2 en los display.

Pero para mostrar los tres números del cronometro (céntima, décima y unidad), se utiliza la multiplexación, que consiste en encender y apagar cada display lo suficientemente rápido para que el ojo humano no lo note, para esto se utilizan las siguientes sentencias.

$output\_d(segr[centr])$;

$output\_high(pin\_e2)$;

$delay\_us(500)$;

$output\_low(pin\_e2)$;

$output\_d(segr[decr])$;

$output\_high(pin\_e1)$;

$delay\_us(500)$;

$output\_low(pin\_e1)$;

$output\_d(segr[unir])$;

$output\_high(pin\_e0)$;

$delay\_us(500)$;

$output\_low(pin\_e0)$;

Al principio se carga el PORTD con el valor de la centésima, se habilita el primer display con el pin E2 por 500 microsegundos y luego se apaga, seguido a esto se carga el PORTD con el valor decimal, se habilita el segundo display con el pin E1 por 500 microsegundos y luego se apaga, de igual manera se hace para mostrar el valor unitario.

Esto se repite constantemente, de esta manera se utilizan solo 11 pines, 4 para el color rojo, 4 para el color azul y 3 pines para encender y apagar cada display.

3.1.11 Botones

Esta subrutina es llamada por el MAIN principal durante el momento del juego una vez configurado, acá se establece la función de cada pulsador.

El pulsador de pared, es el encargado de activar o desactivar el cronometro de cada equipo.

Los otros dos pulsadores, ubicados en la caja, uno sirve para salir del juego y reiniciarlo, mientras que el otro sirve para pausar el juego.

3.1.12 TMR1

Esta subrutina es la que se activa cuando hay una interrupción del TIMER1, según los

cálculos teóricos realizados, está función se activa cada $100\ [ms]$, al activarse incrementamos una bandera y seteamos nuevamente el valor del TMR1 en 3036.

Esta bandera cuando llega a cierto valor, activa la función de cronometro y demás condicionales, en la práctica este valor se da aproximadamente cada un segundo.

3.1.13 RB0

Esta subrutina es la que se activa cuando hay una interrupción en el pin RB0, se activa por flanco de subida. Si el pulsador fue presionado y comenzó el cronometro, se cumple la condición de bandera y se setean 3 banderas más, una que indica que se interrumpió un haz de luz en el momento de juego, otra que sirve para habilitar el módulo de sonido $wtv = 1$, y la última para descontar la penalización extra del cronometro.

3.1.14 WTV020

Este punto hace referencia a las funciones involucradas en el manejo del módulo de sonido WTV020, en este caso hay involucradas 4 funciones, que se nombran a continuación.

- $void\ resetmodulo();$
- $void\ play(int\ track);$
- $void\ stop();$
- $void\ sentcmd(int\ cmd);$

Teniendo en cuenta la información proveniente del manual del módulo, observamos la forma de onda **Fig. Nº 28** de las señales correspondientes. Inicialmente vemos que para utilizar el módulo tenemos que resetearlo, para esto dedicaremos un pin exclusivo, el cual se activará al principio con un tiempo de $5\ ms$. Luego de $300\ ms$, utilizando otro pin para el clock, mandaremos un pulso de $2\ ms$ en el mismo, y luego pulsos de $200\ us$ junto a los datos.

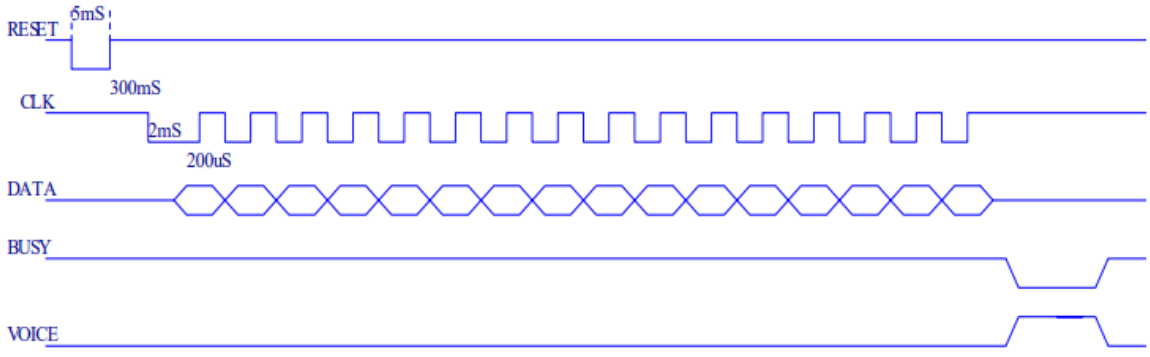

Fig. Nº 28 *Señales para utilizar el módulo WTV020.*

Después de enviar 16 bits de datos, indicando el track a reproducir, el módulo se pone en modo ocupado e instantáneamente reproduce el track mp3.

La función de ***resetmodulo()*** contiene las siguientes instrucciones.

$output\_high(clockPin)$;

$output\_high(resetPin)$;

$delay\_ms(1)$;

$output\_low(resetPin)$;

$delay\_ms(5)$;

$output\_high(resetPin)$;

$delay\_ms(300)$;

Teniendo en cuenta lo explicado anteriormente, vemos que se ponen ambas salidas (clock y reset) en alto, luego de $1\ ms$, se manda a estado bajo el pin reset por $5\ ms$ para resetearlo, luego se produce el delay de $300\ ms$ como indica la **Fig. Nº 28**.

Para la función ***play(int track)*** definimos las siguientes instrucciones.

$sendcmd(track)$;

$wtv = 0$;

Al llamar la función play en el main principal, se le carga el valor "1", entonces la función $sendcmd$ se carga con el valor de "1". Luego vemos que la bandera wtv se iguala a cero. Para la función ***stop()*** se carga el valor de $sendcmd$ con $0xFFFF$, este valor tomado según el manual de usuario, sirve para detener la reproducción del track mp3.

Para la función ***sendcmd(int cmd)*** se definen las siguientes instrucciones.

$long\ int\ i$;

$output\_low(clockPin)$;

$delay\_us(1950)$;

$for(i = 0x8000;\ i > 0;\ i >>= 1)$

{

$output\_low(clockPin)$;

$delay\_us(50)$;

$if(cmd\ \&\ i)\ output\_high(dataPin)$;

$else\ output\_low(dataPin)$;

$delay\_us(50)$;

$output\_high(clockPin)$;

$delay\_us(100)$;

}

$delay\_us(1900)$;

Definimos i como un entero de 16 bits, que es el formato admitido por el módulo de sonido. Luego de eso, se produce el pulso de clock por casi $2\ ms \cong 1950\ us$.

Luego se observa un ciclo for que comienza desde el valor $0x8000$ lo que en binario equivale a $1000\ 0000\ 0000\ 0000 = 16\ bits$, la sentencia $i \gg= 1$ hace la rotación del bit hacia la derecha, por cada pasada de $200\ us$ compara el valor cmd con el valor de 16 bits de i que va rotando, cuando estos dos valores son iguales, el pin de data se pone en valor alto.

Cuando termina el ciclo for, inicia la reproducción del track mp3, se agrega al último un delay de $2\ ms$ indicando que termino de enviarse el paquete de datos.

3.2 Rutina principal

Se presenta en la **Fig. Nº 29** el diagrama en bloques de la rutina principal.

El programa comienza llamando a la subrutina de **configuracion**, en la misma, se configuran los puertos como entrada o salida, se setean las salidas en su valor correspondiente, se habilitan las interrupciones, se inicia el lcd, etc.

Seguido a esto, se presenta el sistema con un mensaje personalizado en el display LCD, utilizando la función **print** y un efecto de luces en el cartel “Laser Zarpy” con la función **frgb**.

Al terminar la presentación del sistema, entra en un ciclo $do - while$ que se repite constantemente al tener una condición que se cumple siempre, las primeras dos funciones que se llaman, tienen como función, resetear las variables utilizadas a lo largo del programa para que las mismas queden seteadas en su valor inicial, estas son **reset** y **resetmodulo**.

Seguido se utiliza la función **frgb** para dejar al cartel con luz blanca que sirve de iluminación para la sala, mientras el operador configura los parámetros del juego.

Estos parámetros se configuran, cuando se llama a la función **menú**, acá el sistema primero da dos opciones de tiempo de juego, 10 o 15 minutos, luego de elegir cualquiera de ellas, el sistema solicita que se elija la penalización por interrupción del haz, las opciones son 5, 10, 15 o 20 segundos.

Al terminar de configurar, entra a la subrutina ready, en esta se indican los parámetros seleccionados recientemente, y es acá donde se puede confirmar o cancelar y volver a configurar los parámetros, gracias a la bandera salir.

En caso de cancelar, la bandera *salir* se setea en 1, esto hace que se salteen las funciones por debajo y vuelva a la función **reset** dando la posibilidad de configurar nuevamente los parámetros.

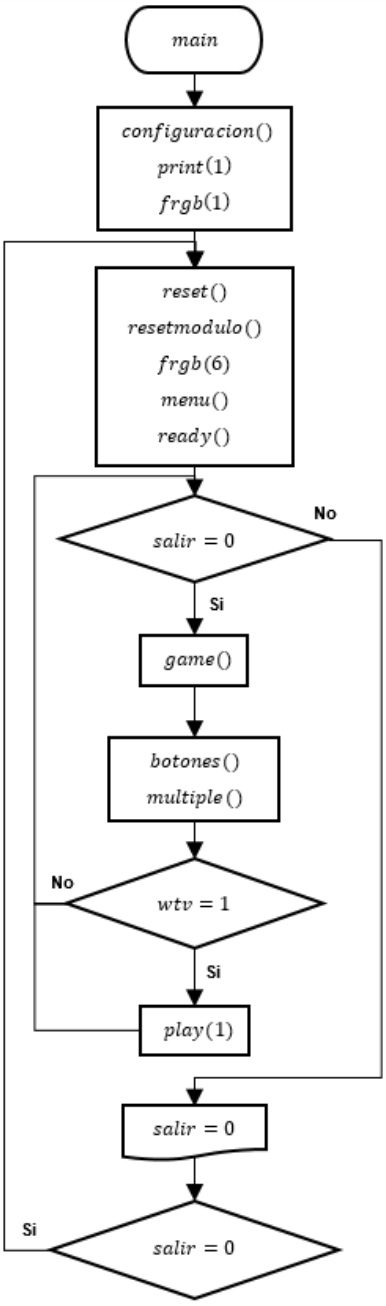

Fig. Nº 29 *Diagrama en bloques rutina principal.*

En caso de confirmar los parámetros, la bandera *salir* se iguala a cero, condición necesaria para entrar en el ciclo *while* donde se ejecutan de forma continua la función **botones** y **multiple**, y en caso de cumplir la condición de bandera *wtv* igual a 1, se ejecuta la función **play** encargada de reproducir el sonido de alarma.

Desde la función **botones** podemos hacer uso de los pulsadores, y con un pulsador podemos salir del juego para volver al menú, igualando la bandera *salir* a 1.

4. Evaluación Final del Sistema

Luego de las simulaciones correspondientes se procedió a diseñar los circuitos electrónicos para cada placa, la fabricación de estas, fueron realizadas sobre placas epoxy FR4, brindando un mayor grado de calidad y durabilidad.

Estos módulos de hardware deberán ser instalados en una sala, para esto se desarrolló, un gabinete o caja específico para cada uno.

Los diseños de gabinetes, cajas y demás se realizaron utilizando software de modelado 3D. Luego se generaron los archivos correspondientes y se imprimieron en 3D con material plástico denominado PLA.

Box principal

Este incluye el microcontrolador, módulo de audio, módulo de control de RGB, display LCD y pulsadores, **Fig. Nº 30**, dentro del mismo las placas están interconectadas entre sí, y para la comunicación con el exterior, se usó un conector DB9 donde viene la alimentación y conexiones con demás periféricos.

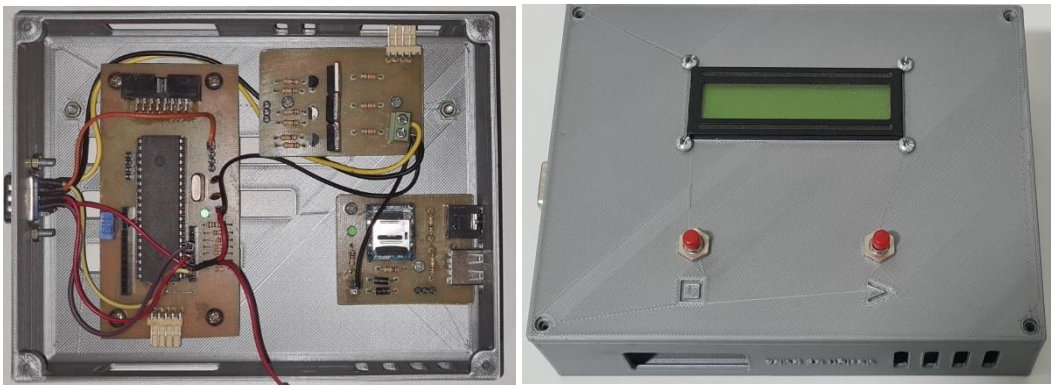

Fig. Nº 30 *Box principal.*

Box de sensores

Este incluye el pcb que contiene los sensores LDR, se muestra en la **Fig. Nº 31**.

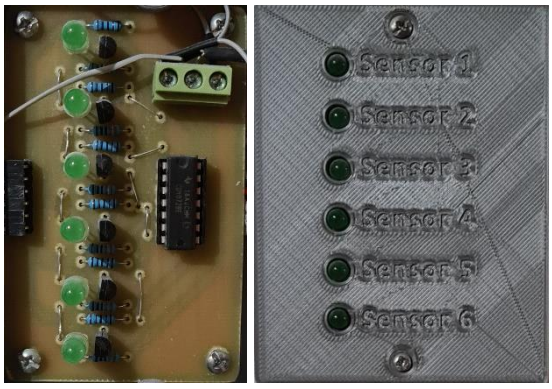

Fig. Nº 31 *Box de sensores.*

Tablero 7 segmentos

Este incluye el pcb del display 7 segmentos, como se muestra en la **Fig. Nº 32**.

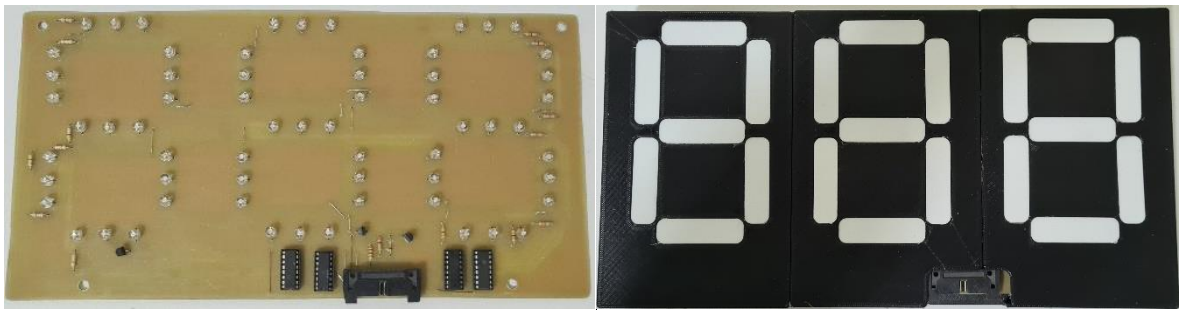

Fig. Nº 32 *Box display 7 segmentos.*

En anexo se agregan algunas imágenes del juego laberinto láser instalado.

5. Conclusiones

Al finalizar el proyecto, se obtuvo el resultado deseado, un laberinto láser instalado en una sala rectangular, que contiene un tablero para mostrar el tiempo, una alarma, un cartel luminoso y una red de rayos formada por 6 láseres en conjunto con 6 espejos, que terminan en 6 sensores.

El desarrollo del mismo nos proporcionó el reto de cumplir y resolver ciertos parámetros de diseño, se puede decir que, se lograron los objetivos planteados al comienzo de este proyecto, obteniendo excelentes resultados.

Queda decir que, si se quisiera avanzar en mejorar el sistema, se debería pensar en un rediseño.

Por parte del hardware, el microcontrolador elegido tiene sus limitaciones, como por ejemplo, tener solamente dos salidas PWM, esto limitó al momento de controlar los efectos de luces led y por otro lado, no tener disponibilidad de conexión con PC, esto hace que el entorno gráfico del juego sea muy pobre en aspectos visuales.

En cuanto al firmware, este podría ser actualizado para incluir nuevas variantes o modalidades de juegos.

Bibliografía

[1] Compilador C CCS y simulador PROTEUS para microcontroladores PIC, Eduardo García Breijo, 1º edición, Alfaomega Grupo Editor, México, 2008.

[2] PIC16F882/883/884/886/887, datasheet, 28/40/44-Pin, Enhanced Flash-Based 8-Bit CMOS Microcontrollers with nanoWatt Technology, Microchip Technology Inc, 2007.

[3] WTV020-SD MODULE, datasheet, WTV020-SD-20S and WTV020-SD-16P, Guangzho Waytronic Electronic, 2017.

[4] Light Dependent Resistor - LDR, datasheet, SUNROM, 2008.

[5] WH1602C Character 16x2, datasheet, WINSTAR, 2008.

[6] CMOS BCD-to-7-Segment Latch Decoder Drivers, CD4511B Types, datasheet, Texas Instruments, 1998.

[7] IRF530, Power MOSFET, datasheet, Vishay Siliconix, 2015.

[8] ULN2003A, Seven darlington arrays, datasheet, STMicroelectronics, 2002.

[9] U.S Food and Drug; Laser products and instruments; accedido 20/04/2021; https://www.fda.gov/radiation-emitting-products/home-business-and-entertainment-products/laser-products-and-instruments.

Anexo I

Se adjuntan imágenes del juego laberinto láser instalado.

Fig. Nº 33. *Sala laberinto láser.*

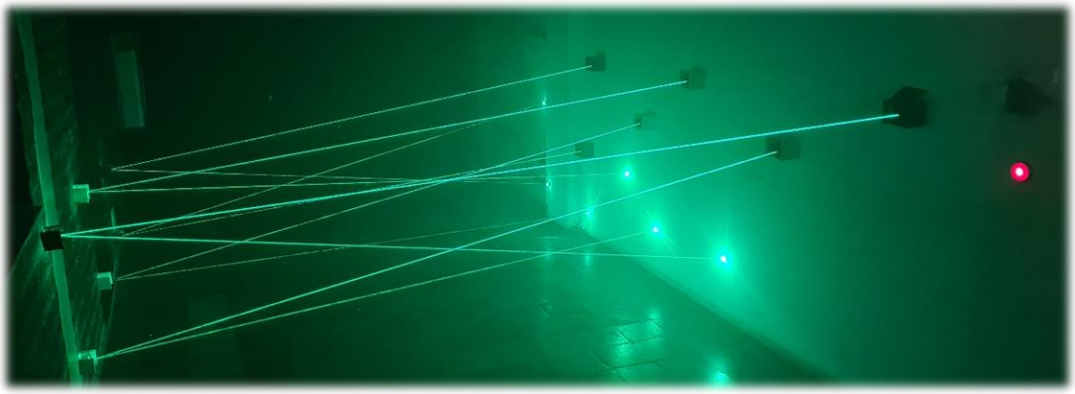

Fig. Nº 34. *Sala laberinto láser.*

Anexo II

Se adjuntan hojas de datos de los componentes más importantes utilizados en este proyecto.