UNIVERSIDAD DE SEVILLA

ESCUELA SUPERIOR DE INGENIEROS

DEPARTAMENTO DE INGENIERÍA ELECTRÓNICA

Proyecto fin de carrera

Clasificador de colores utilizando un microcontrolador PIC

Tutor:
Dr. D. Johnathan Noel Tombs
Alumno:
Prudencio Luna Belizón

Sevilla, Noviembre de 2003

ÍNDICE

ÍNDICE

INTRODUCCIÓN

1. INTRODUCCIÓN

El título oficial de este proyecto fin de carrera es **Clasificador de colores utilizando un microcontrolador PIC**. Consiste en el diseño, usando un microcontrolador, un sensor y un dispositivo grabador/reproductor de sonido, de un sistema que distinga el color y que, posteriormente, indique el nombre del color con voz.

En la actualidad existen dispositivos que diferencian el color haciendo uso de elementos ópticos: lentes y prismas, que son capaces de separar la luz en sus distintas componentes espectrales. Estos sensores son bastante complicados y caros, aunque permiten distinguir entre un gran número de colores. La novedad que presenta este proyecto fin de carrera es que, utilizando dispositivos optoelectrónicos comunes tales como el diodo emisor de luz y una LDR, pueden llegar a distinguir colores, aunque en menor número que los sensores que hacen uso de componentes ópticos.

Otra novedad que introducimos en el proyecto es la posibilidad de oír el nombre del color que el sensor y el microcontrolador detecten. De esta tarea se encarga un dispositivo grabador/reproductor de sonidos. Para ello se ha construido un circuito que grabe los nombres de los colores en el dispositivo.

Tras esta breve introducción, se explicará más detalladamente cada uno de los tres bloques funcionales: sensor, microcontrolador y dispositivo reproductor de sonido; en este último, podemos separar los componentes de la tarjeta del indicador de color con voz.

OBJETIVOS

2. OBJETIVOS

Las principales aplicaciones de este proyecto son el diseño y el desarrollo de prototipos de aplicación terapéutica para personas que padezcan discapacidades físicas y/o mentales de diversa índole, adaptando sistemas electroópticos a dichas terapias. El sensor óptico tiene interés para personas invidentes y con problemas visuales de color. Está dirigido a personas con deficiencias visuales parciales (principalmente daltónicos) o totales.

Para abaratar el coste del sensor se han empleado componentes electrónicos simples, como resistencias, leds, etc. Se podría haber realizado con lentes y cristales especiales, pero habría encarecido significativamente el producto.

La presentación de la información del color al usuario se realiza mediante un sistema de voz compuesto por un chip diseñado especialmente para almacenar y reproducir sonidos y un amplificador de audio para mejorar la calidad del sonido en el exterior.

El manejo del dispositivo está simplificado gracias a un único pulsador, que inicia la secuencia de todo el proceso: identificar el color de un objeto y reproducir el nombre del color en cuestión.

La alimentación del indicador se realiza mediante una batería tipo “PP3” de 9 V. Su pequeño tamaño contribuye a disminuir el volumen y el peso totales del indicador.

BASE TEORICA

3. BASE TEÓRICA

3.1. La luz

Según la óptica ondulatoria, la luz es una onda cuyo movimiento puede describirse mediante una función que depende de la posición y del tiempo: $\psi(\mathrm{r},t)$, que se llama función de onda.

3.2. Ondas monocromáticas

Las ondas monocromáticas son aquellas que tienen una única frecuencia y responden a la siguiente ecuación de onda:

$$\psi(\mathrm{r},t) = A\cos(kx \pm \omega t)$$

donde *A* es la amplitud, que se corresponde con la máxima perturbación, *k* es el número de onda y ω es la frecuencia temporal angular. También podemos escribir esta ecuación de la siguiente forma:

$$\psi(\mathrm{r},t) = A\cos\left[2\pi\upsilon\left(\frac{x}{v} \pm t\right)\right]$$

siendo υ la frecuencia temporal (número de ondas por unidad de tiempo) y *v* la velocidad de desplazamiento en la dirección de la onda. La longitud de onda, λ,

es el número de unidades de longitud por onda:

$$\lambda = \frac{2\pi}{k}$$

El periodo temporal, T, es la cantidad de tiempo que una onda completa tarda en superar a un observador estacionario:

$$T = \frac{\lambda}{v}$$

La frecuencia temporal angular, ω, es:

$$w = \frac{2\pi}{T} = 2\pi\upsilon$$

Aunque, una forma más sencilla de escribir la ecuación de onda es:

$$\psi(\mathrm{r},t) = A\cos(\omega t \pm \varphi(\mathrm{r}))$$

donde φ es la fase.

Normalmente se trabaja no en forma de cosenos ni senos sino en forma exponencial:

$$\Psi(\mathrm{r},t) = Ae^{j\cdot\varphi(\mathrm{r})}\cdot e^{j\cdot wt} = \Psi(\mathrm{r})\cdot e^{j\cdot wt}$$

3.3. Ondas policromáticas

La luz es una onda policromática cuyo comportamiento en los medios materiales se estudia descomponiéndola en ondas monocromáticas mediante el análisis de Fourier. Además, se debe comprobar cada componente en el medio y, posteriormente, aplicar el principio de superposición.

Podemos definir matemáticamente las ondas policromáticas como suma de ondas monocromáticas de frecuencia ν:

$$\psi(\mathrm{r},t) = \int_{-\infty}^{\infty} \Psi_{\upsilon}(\mathrm{r}) \cdot e^{j \cdot 2 \cdot \pi \cdot \upsilon \cdot t} d\upsilon$$

En la siguiente figura se puede observar que, al hacer incidir un haz de luz blanca (onda policromática, que contiene todas las longitudes de onda) en un prisma, ésta se descompone en sus distintas componentes.

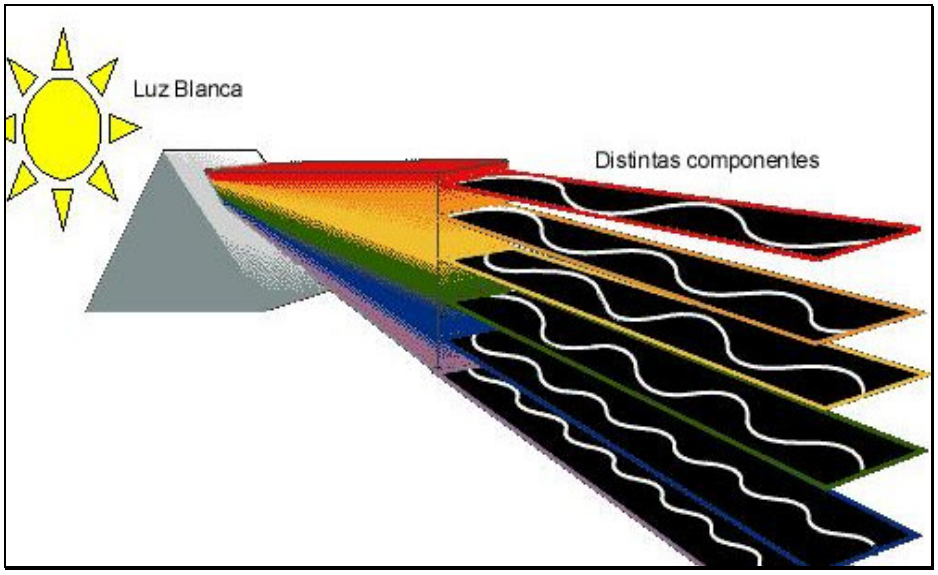

Figura 1

3.4. Sistema RGB

Frecuentemente, en todos los aparatos electrónicos y en el software informático se utiliza el sistema RGB para representar o capturar los colores de los distintos objetos. Este sistema se basa en caracterizar cada color como combinación lineal de los colores rojo, verde y azul (en inglés *red, green and blue: RGB*).

$$color = \lambda_1 \cdot rojo + \lambda_2 \cdot verde + \lambda_3 \cdot azul$$

donde: $\lambda_1, \lambda_2, \lambda_3 \in [0,1]$

De este modo, el color negro se obtendría para (λ_1=0,λ_2=0,λ_3=0), el azul "puro" para (0,0,1), el rojo "puro" (1,0,0) y el verde "puro" para (0,1,0). Variando los coefientes λ_i se pueden obtener todos los colores (ver figura 2).

Figura 2

En el caso de los monitores con la tecnología TRC se utilizan tres

cañones de electrones, cada uno de los cuales se corresponden con los colores rojo, verde y azul. Un cañón hace incidir un haz de electrones en cada uno de los píxeles con mayor o menor intensidad. El aumento de la intensidad se corresponde con un aumento de la cantidad de color en el píxel. La suma de los tres colores en cada píxel dará como resultado un color en concreto.

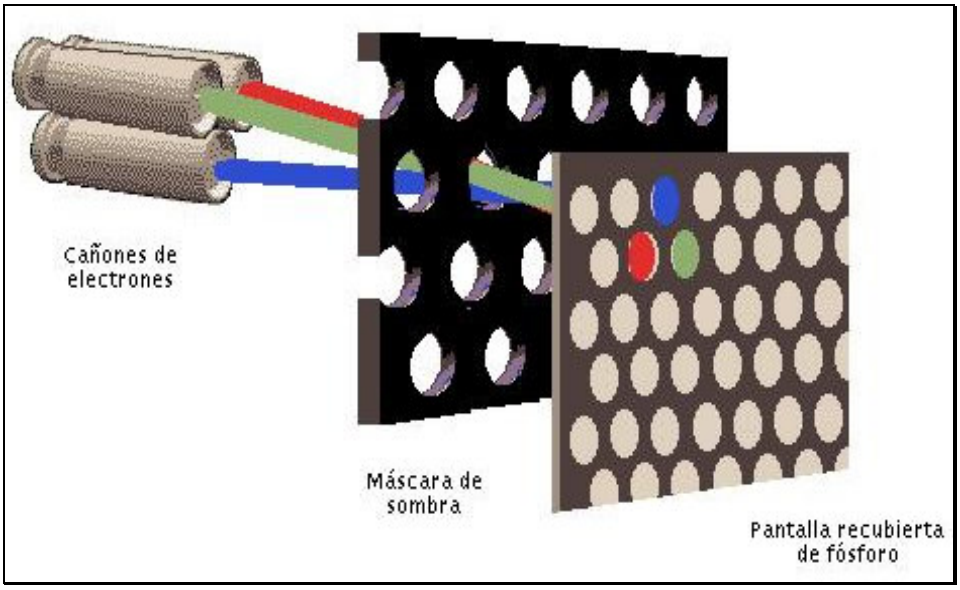

Figura 3

Acerca de las nuevas pantallas TFT, aunque tienen una tecnología muy diferente, también se basan en esta teoría. En este caso, se utiliza en cada píxel una combinación de tres emisores ópticos de los colores rojo, verde y azul. El controlador gráfico de la pantalla TFT, cuando recibe la señal de imagen, varía la intensidad luminosa de estos emisores ópticos hasta alcanzar en cada píxel el color deseado (como combinación lineal de los tres colores).

Las cámaras digitales, ya sean fotográficas o videocámaras, siguen el proceso inverso. En el objetivo del aparato hay sensores optoelectrónicos que varían alguna magnitud eléctrica en función de la cantidad de radiación

luminosa que capten. Para poder distinguir el color se emplean filtros ópticos de manera que, a cada sensor optoelectrónico le corresponde un filtro óptico, el cual sólo deja pasar la luz de unas determinadas frecuencias υ. En este caso, las frecuencias son aquellas correspondientes a los colores rojo, verde, y azul. A cada sensor sólo llegará una intensidad luminosa en función de su filtro.

3.5. El color en los objetos

Todos los objetos tienen la propiedad de absorber y reflejar ciertas radiaciones electromagnéticas. La mayoría de los colores que experimentamos normalmente son mezclas de longitudes de onda que provienen de la absorción parcial de la luz blanca. Casi todos los objetos deben su color a los filtros, pigmentos o pinturas, que absorben determinadas longitudes de onda de la luz blanca y reflejan las demás; estas longitudes de onda reflejadas o transmitidas son las que producen la sensación de color, que se conoce como color pigmento.

Nuestra percepción del color de las partes de una escena no sólo depende de la cantidad de luz de las diferentes longitudes de onda que nos llega de ellas. Cuando sacamos un objeto iluminado con luz artificial —que contiene mucha luz rojiza de altas longitudes de onda— a la luz del día —que contiene más luz azulada de longitudes de onda cortas— la composición de la luz reflejada por el objeto cambia mucho. Sin embargo, no solemos percibir ningún cambio en el color del objeto. Esta constancia del color se debe a la capacidad del sistema formado por el ojo y el cerebro para comparar la información sobre longitudes de onda procedente de todas las partes de una escena.

En las siguientes figuras podemos apreciar lo explicado.

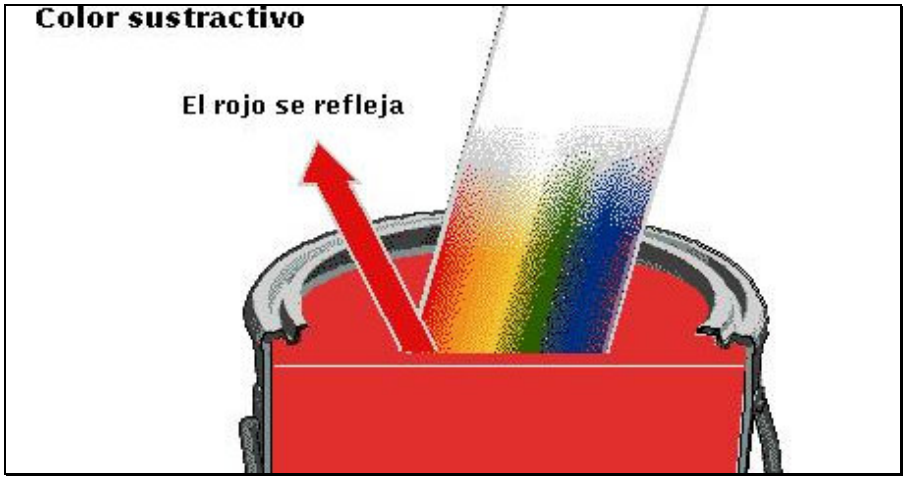

Figura 4

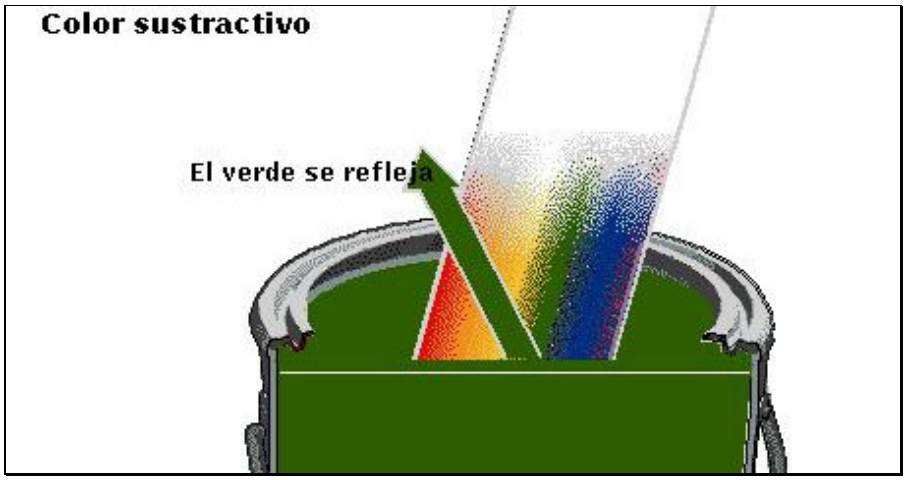

Figura 5

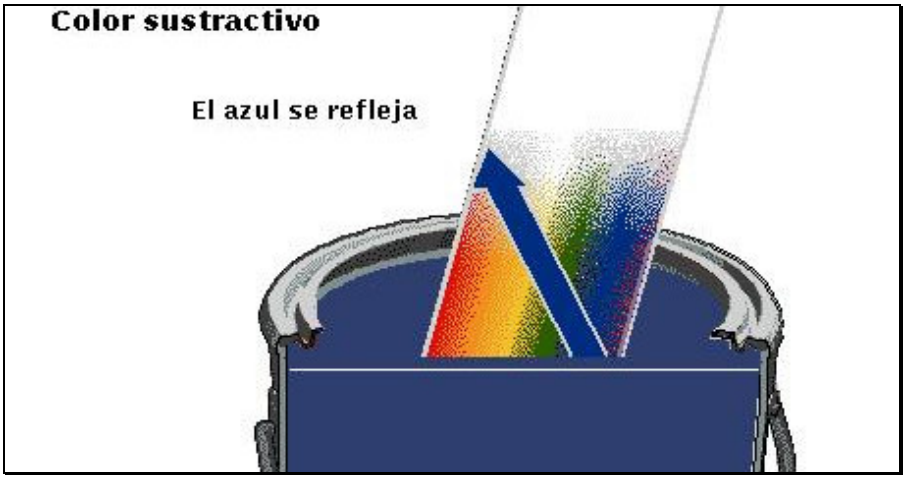

Figura 6

Figura 7


Color sustractivo
No se refleja ningún color;
todos son absorbidos

Figura 8

HARDWARE

LA PLACA

4. LA PLACA

4.1. Introducción

En las figuras siguientes, figuras 9 y 10 podemos ver una fotografía de la placa del circuito. Podemos dividir el circuito de la placa en cuatro bloques claramente diferenciados: la circuitería de alimentación, la circuitería del sensor, y el sensor, el microcontrolador y el dispositivo de reproducción de voz. Cada uno de estos bloques va a desempeñar una función concreta dentro del circuito. En próximos apartados se hará un estudio detallado de cada uno de estos bloques.

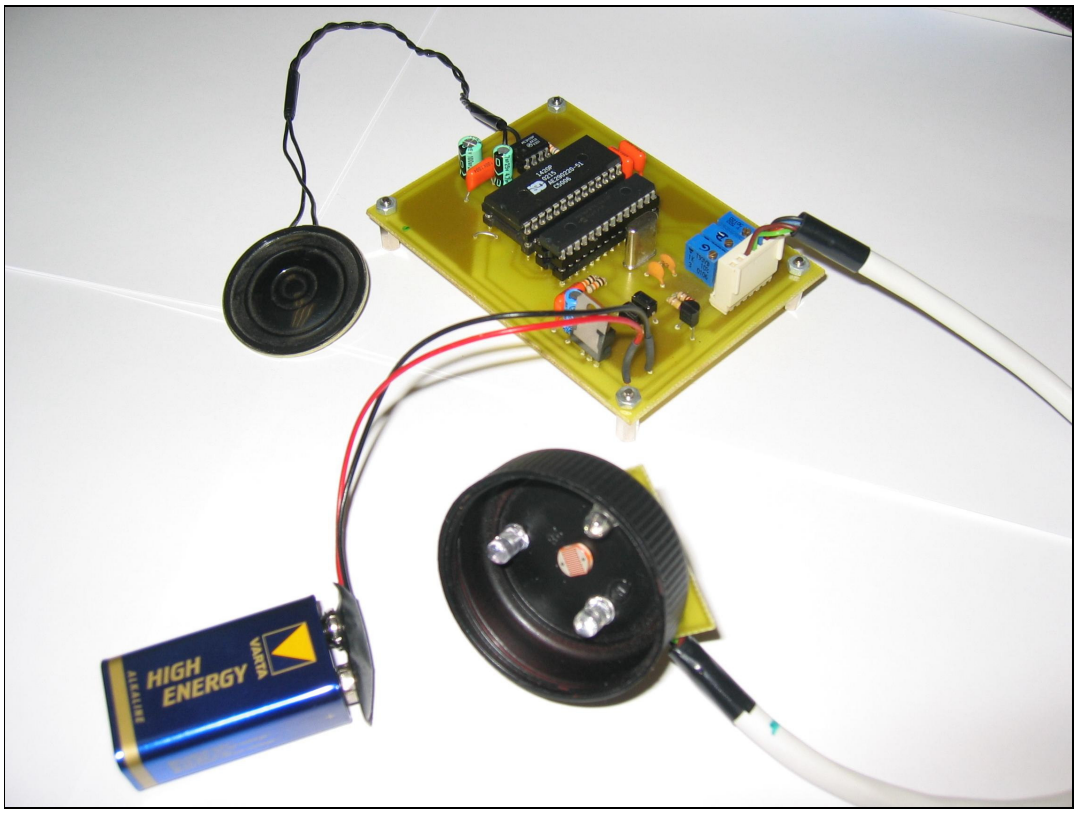

Figura 9

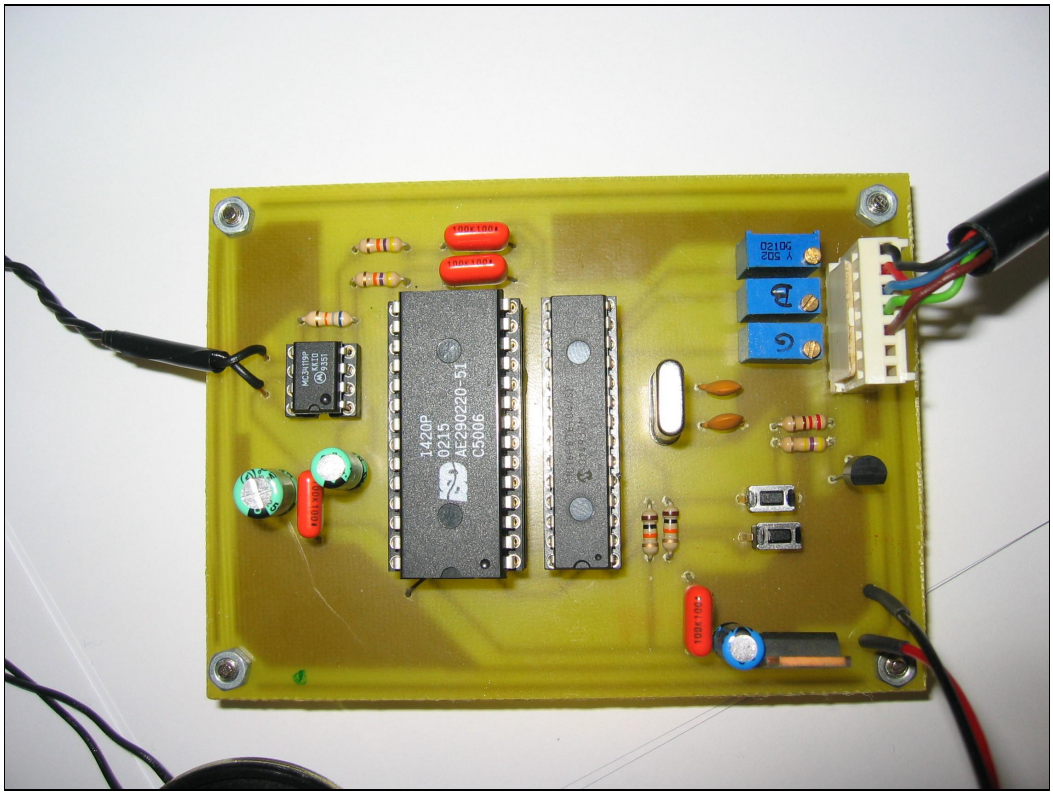

Figura 10

De forma general podemos representar en un gráfico los cuatro bloques en los que se ha dividido el circuito de la placa. Las flechas nos indican el tipo de relación, unidireccional o bidireccional, que existe entre los tres bloques.

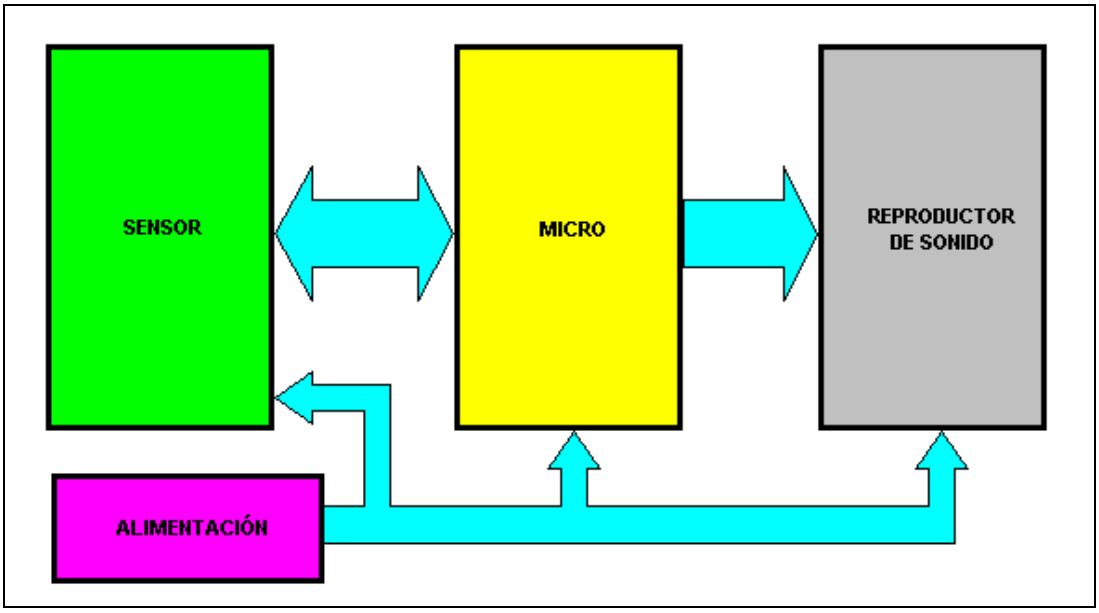

Figura 11

4.1.1. El circuito Alimentación

Parte del circuito se encarga de proporcionar la tensión adecuada a los demás elementos del circuito para su correcto funcionamiento. Transforma la tensión de la pila, 9 voltios, a los 5 voltios que necesitan tanto el microcontrolador como el dispositivo reproductor de voz.

4.1.2. El microcontrolador

El microcontrolador es un elemento importante dentro de nuestro sistema. Por un lado, se va a encargar de controlar el sensor y captar la señal que recibe de éste, y por otro lado, decidirá el color que se está sensando en función de las distintas señales que ha recibido del sensor. Una vez que ya tiene decidido el color actuará sobre el dispositivo reproductor de sonido para que éste reproduzca el sonido (voz) que indica el color que se está sensando.

4.1.3. La circuitería del sensor y el sensor

Se ha hecho una diferenciación entre la circuitería del sensor y el sensor. La razón se debe a localización de los componentes: ya sea en la placa o en el exterior de ésta. Esta implantación de dichos componentes del sensor en la placa tiene como objetivos disminuir el peso y el tamaño del sensor. La circuitería del sensor en la placa se encarga de actuar como interfaz entre el sensor y el microcontrolador; sus funciones son: control de la intensidad luminosa de los diodos led que forman parte del sensor y mantener constante la tensión que sirve de referencia para el circuito divisor de tensión, del cual toma la tensión el convertidor A/D del microcontrolador.

4.1.4. El dispositivo reproductor de sonido

El dispositivo reproductor de sonido tiene almacenados los nombres de los 8 colores que el sistema es capaz de distinguir. Una vez que llega la señal de control del microcontrolador, éste reproduce uno de estos nombres. Debido a que el dispositivo no tiene suficiente potencia de salida es necesario un amplificador de audio que transmita la señal al altavoz.

EL CIRCUITO DE ALIMENTACIÓN

5. EL CIRCUITO DE ALIMENTACIÓN

El circuito de alimentación es el encargado de suministrar la tensión necesaria para el correcto funcionamiento de los dispositivos, en nuestro caso 5 voltios (Vdd).

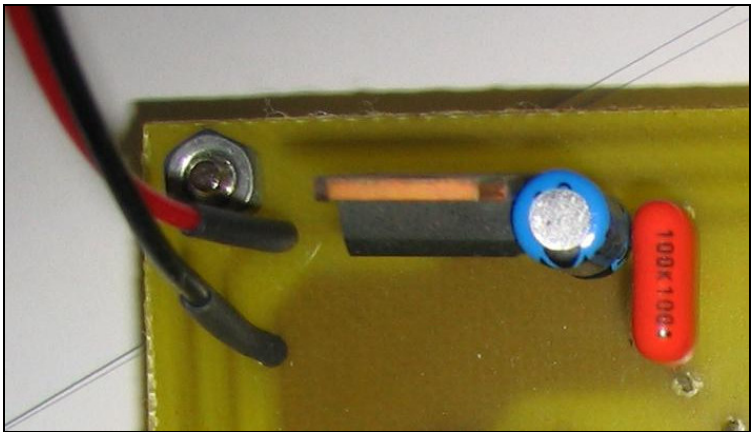

Figura 12

Toda la tensión del circuito la proporciona una batería del tipo PP3 de 9V. Ésta se conecta al circuito mediante un par de cables, que, por un lado, están soldados a la placa, y, por el otro, tienen un conector para baterías de tipo PP3. Para la adaptación de los niveles de 9 voltios a los 5 voltios necesarios, se ha utilizado un regulador de tensión UA7805, que puede suministrar una corriente de hasta un amperio. El UA7805 es un dispositivo de tres terminales; el primero, V_i, es por donde se le introduce la tensión de entrada, que puede variar entre los 7 a 24 voltios. El segundo, GND, donde se conecta la masa del circuito. El tercero, V_o, que proporciona la tensión en voltios que indique los dos últimos dígitos del identificador del dispositivo, en este caso, 5 voltios. Para asegurar que la tensión de salida en V_o es la deseada, es necesario mantener en V_i una tensión dentro del rango indicado

por el fabricante. Si la tensión en el terminal V_i es inferior a este rango, la respuesta del regulador no será la adecuada. Para ayudar a mantener estabilizada la tensión a la salida del regulador se emplean un par de condensadores de 22uF y 100nF.

En la siguiente figura podemos ver el esquema eléctrico del circuito de alimentación.

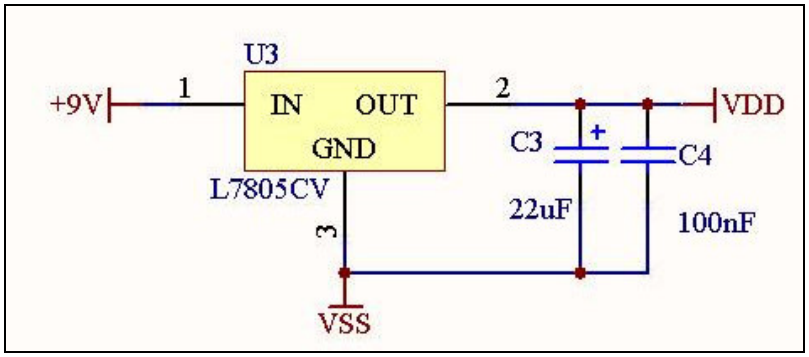

Figura 13

EL SENSOR

6. EL SENSOR

6.1. Introducción

Si recordamos lo visto en el apartado anterior (base teórica), los cuerpos se comportan de forma diferente, absorbiendo más o menos cantidad de luz, en función del tipo de luz que incide sobre ellos. Por ello, se ha fabricado un tipo de sensor que aproveche este fenómeno físico. Se pensó en la construcción de dos tipos distintos de sensores que denominamos: sensor usando luz blanca y sensor con diodos led. Ambos se basan en el mismo principio: distinguir la cantidad de luz roja, verde y azul que absorbe el objeto bajo test.

6.2. Sensor con luz blanca

Lo primero que necesitamos será crear un sensor que sea opaco y cubra al objeto para que no interfiera la luz exterior, ya que podría dar problemas. El sensor se basaría en una luz blanca que iluminaría al objeto en cuestión; éste absorbería las componentes de la luz blanca dependiendo de su naturaleza. Posteriormente, la luz no absorbida por el objeto debe pasar a través de tres filtros ópticos, correspondientes a los colores rojo, verde, y azul. Para ello, podríamos usar tres papeles transparentes de colores rojo, verde y azul que se pueden encontrar en cualquier papelería. Tras cada filtro habría un sensor que mediría la cantidad de luz que de cada uno de estos tres colores que no ha sido absorbida por el objeto. Para una mejor comprensión veamos un posible construcción de este sensor.

Figura 14

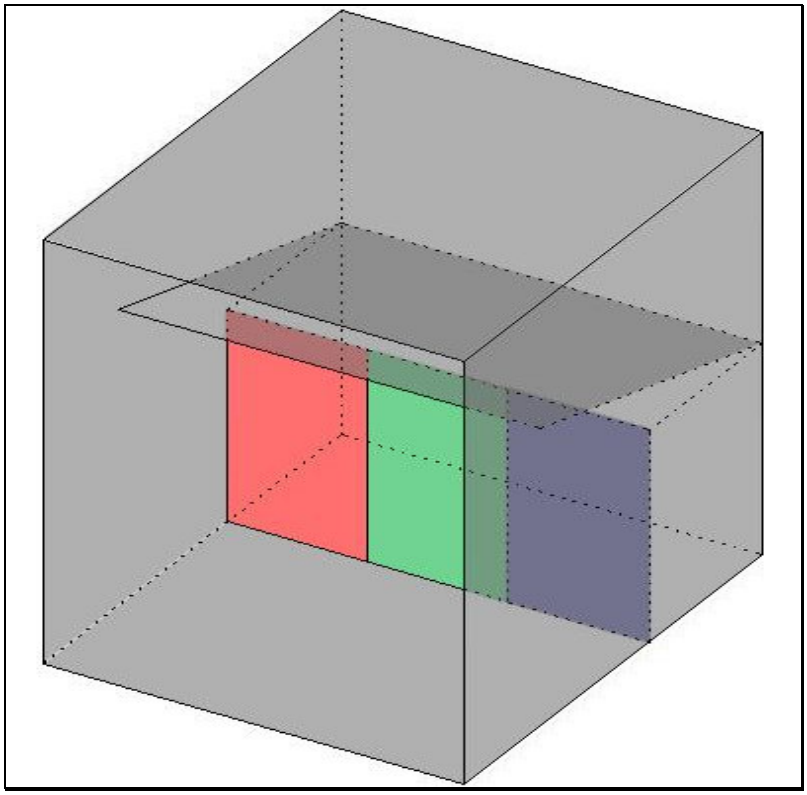

Figura 15

El sensor está basado en el funcionamiento de las pantallas CCD o los escáneres. Pero tenía el problema de ser más complicada su construcción y necesitaría más sensores que el modelo siguiente, lo que podría ocasionar más errores y problemas.

6.3. El sensor con diodos led

Esta es la segunda alternativa en la que se pensó, y la que finalmente se realizó por la facilidad que presentaba su construcción. Consta de tres led de encapsulado transparente (alto brillo) de los colores rojo, verde y azul. Éstos se colocan dentro de una cápsula opaca formando un triángulo equilátero cuyo baricentro sería el centro de la circunferencia que forma la cápsula y donde colocaríamos nuestro sensor de luz, en este caso una LDR.

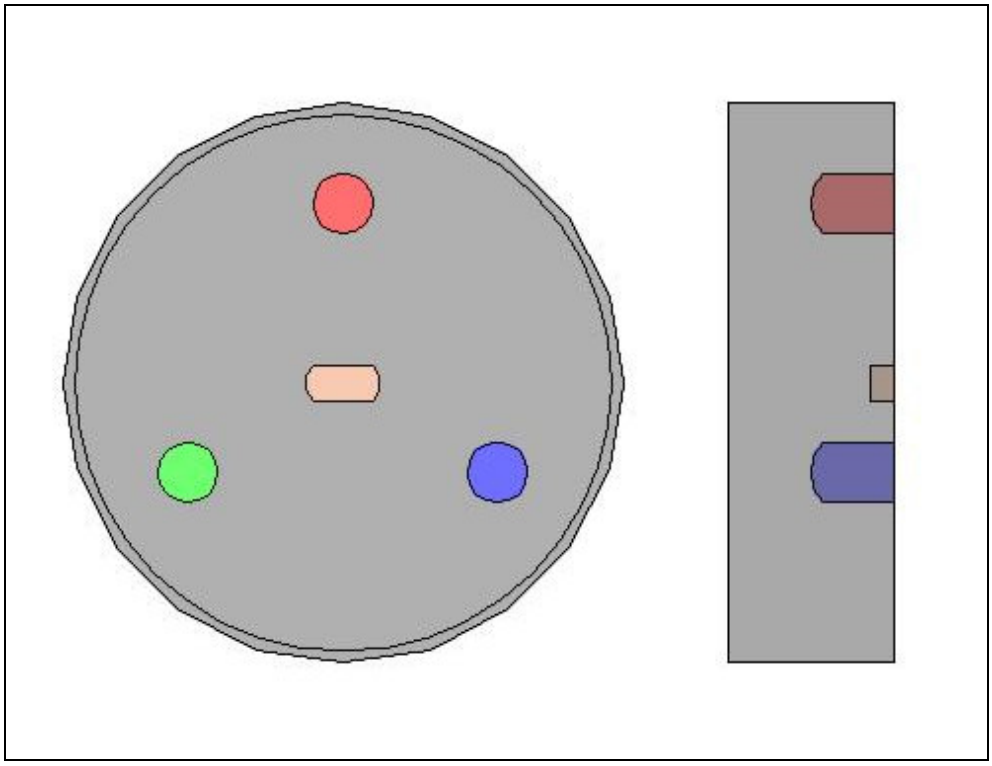

Figura 16

En la siguiente imagen podemos ver una fotografía del sensor ya construido.

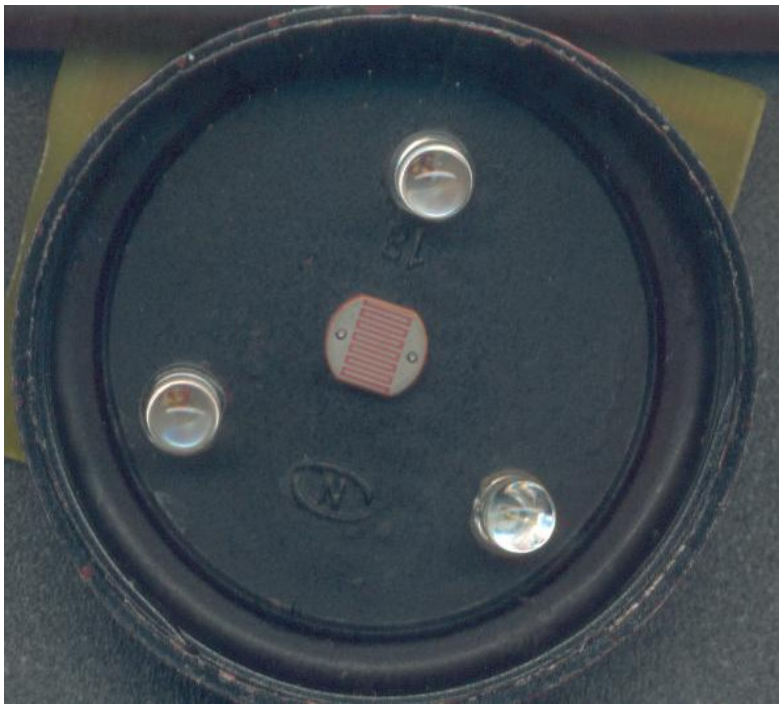

Figura 17

El funcionamiento de este sensor es muy sencillo. Inicialmente los leds están apagados, y se van a encender y apagar de forma sucesiva, de manera que nunca haya dos leds encendidos a la vez. La secuencia de encendido, es decir, qué led se enciende primero, y cuál el último, es indiferente. Imaginemos que queremos ver el color de un objeto que, en este ejemplo es de color azul "puro". Al encender el led rojo, la luz ambiente será menor en el sensor, ya que el objeto sólo refleja la luz azul (y absorbe la roja). Si encendemos el led verde, ocurrirá lo mismo: la luz ambiente será menor. En cambio, con el led azul, la luz en el sensor será mayor.

En la siguiente tabla se puede observar para distintos colores la cantidad de luz relativa que habrá en el interior del encapsulado cuando están encendido los distintos leds.

Color objeto	Led rojo	Led verde	Led azul
Blanco	+ luz	+ luz	+ luz
Negro	- luz	- luz	- luz
Rojo	+ luz	- luz	- luz
Verde	- luz	+ luz	+ luz
Azul	- luz	- luz	+ luz

Por tanto, podemos decir que, para determinar el color de los objetos, nuestro sensor (encapsulado junto con los tres leds y la LDR) medirá la cantidad de luz ambiente que queda en el interior de la cápsula cada vez que esté uno de los tres leds encendido. Para medir la luz ambiente de la capsula utilizamos la LDR que tiene la propiedad de variar su resistencia con la variación de la luz que incide sobre ella.

6.3.1. La circuitería del sensor

La figura 18, de la página siguiente, muestra la parte de la placa que se encarga de actuar como interfaz entre el microcontrolador y el sensor.

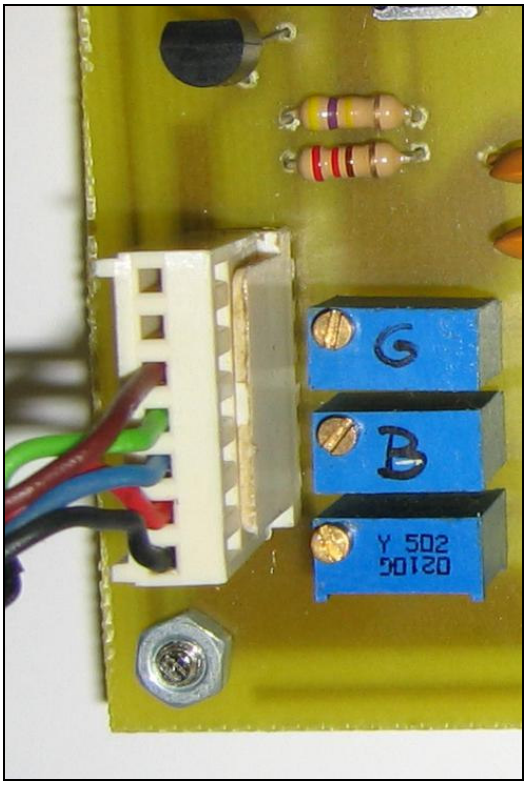

Figura 18

El circuito que va a actuar como transductor de la magnitud luminosa a la magnitud eléctrica es un simple divisor de tensión. Para que la tensión a dividir, tensión de referencia, sea siempre la misma, se usa un regulador de tensión comercial, 78L05. La tensión que nos proporciona es de unos 5 voltios, que serán divididos entre las resistencias del divisor de tensión, las cuales harán llegar menos de 5 voltios a la entrada del convertidor analógico digital del microcontrolador. Para alimentar el 78L05 se usará una pila del PP3 de 9 voltios. Las resistencias para el divisor de tensión que se

han utilizado son de 220Ω y 470kΩ en serie con la LDR. La tensión que se va a tomar como señal, que entrará directamente en la patilla 5 del microcontrolador, será la suma de las tensiones que caigan en la LDR y en la resistencia de 220Ω.

Los leds están directamente gobernados por el microcontrolador a través de tres resistencias iguales, limitadoras de la corriente, cada una en serie con su correspondiente led. El problema consistía en que cada led proporcionaba una intensidad luminosa diferente. Esto podía ser debido a varias causas: las diferencias constructivas que pueden presentar los diodos led, las tolerancias de las resistencias en serie con los diodos y, por último, que la tensión de salida de los diferentes pines de un mismo puerto del microcontrolador no sea la misma. Para evitar estos problemas se recurrió a la sustitución de cada una de las resistencias que estaban en serie con los diodos led por tres potenciómetros multivueltas, que permiten calibrar la intensidad lumínica de cada led, variando su resistencia poco a poco mediante un pequeño tornillo.

En la siguiente página podemos ver el esquema completo del circuito del sensor.

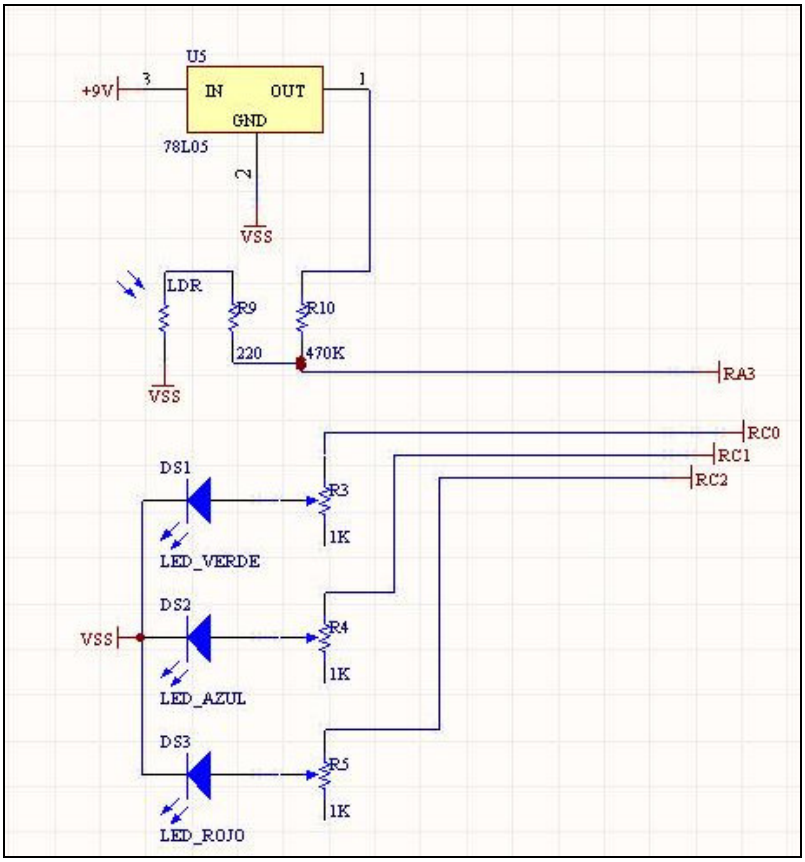
U5
+9V
3
IN
OUT
1
GND
78L05
C1
VSS
LDR
R9
220
R10
470K
VSS
RA3
RC0
RC1
RC2
DS1
R3
1K
LED_VERDE
DS2
VSS
R4
1K
LED_AZUL
DS3
R5
1K
LED_ROJO

Figura 19

EL MICROCONTROLADOR

7. EL MICROCONTROLADOR

7.1. Introducción

Para controlar tanto el sensor como el reproductor de voz se ha optado por un microcontrolador de la marca Microchip; en particular, por el modelo 16F876. La elección de este modelo se basa en que este microcontrolador tiene un convertidor A/D, por lo que no tendremos que introducir un elemento más en la placa, y, también, porque el número de puertos que presenta es suficiente para controlar todos los dispositivos. Además, la memoria Flash de programa que presenta también es suficiente.

En la siguiente figura podemos ver una fotografía del microcontrolador en la placa:

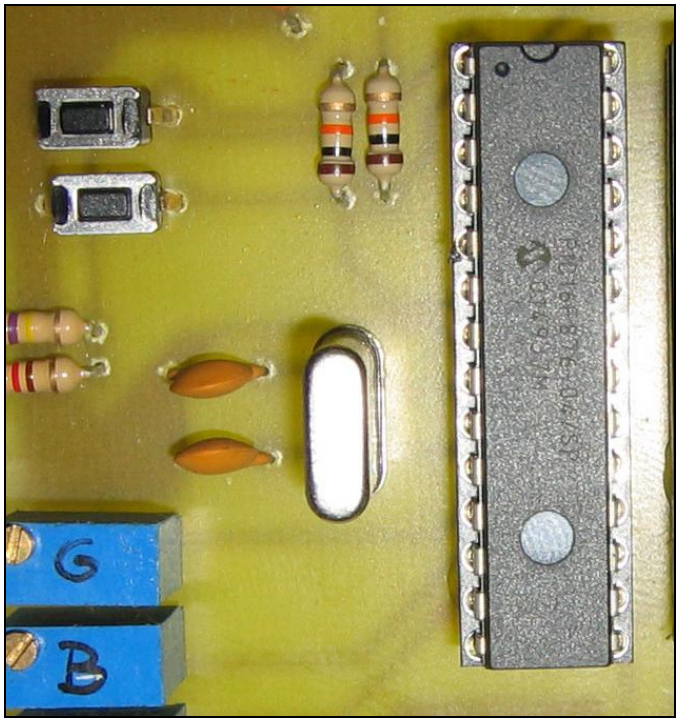

Figura 20

7.2. Características del PIC16F876

Microcontrolador PIC de gama media, se caracteriza por tener una memoria de programa de tipo FLASH, memoria de datos RAM y EEPROM. Viene en un encapsulado de 28 pines con tres puertos, A, B y C de entrada y de salida. Entre los principales recursos de este microcontrolador son:

- Procesador de arquitectura RISC.
- Juego de 35 instrucciones con 14 bits de longitud. Todas ellas se ejecutan en un ciclo de instrucción (depende de la frecuencia de reloj externa), menos las instrucciones de salto, que tardan 2 ciclos.
- Frecuencia máxima de funcionamiento de 20 Mhz.
- Hasta 8 K de 14 bits para la memoria de programa tipo FLASH.
- Hasta 368 bytes de memoria de datos RAM.
- Hasta 256 bytes de memoria de datos EEPROM.
- Hasta 14 fuentes de interrupción interna y externa.
- Pila con 8 niveles.
- Modos de direccionamiento directo, indirecto y relativo.
- Perro guardián (WDT).
- Código de protección programable.
- Modo SLEEP de bajo consumo.
- Programación en serie en circuito con dos patitas.
- Voltaje de alimentación comprendidor entre 2 y 5,5 V.
- Alta entrada o salida de corriente de hasta 25 mA.
- Bajo consumo (menos de 0.6 mA a 3V y 4Mhz).

- Periféricos como:

  - TMR0 ◊ Contador/temporizador de 8 bits con predivisor.
  - TMR1 ◊ Contador/temporizador de 16 bits con predivisor:
  - TMR2 ◊ Contador/temporizador de 8 bits con predivisor y postdivisor.
  - Dos modulos de captura/comparación PWM.
  - Convertidor multicanal A/D de 10 bits.
  - Módulo de comunicacoón serie síncrona (MSSP), con modo SPI e I2C.
  - Interfaz de comunicacón serie SCI, también conocido como USART.
  - Puerta paralela esclava (PSP).

7.3. El microcontrolador en la placa

En la siguiente figura podemos observar el diagrama de pines del PIC:

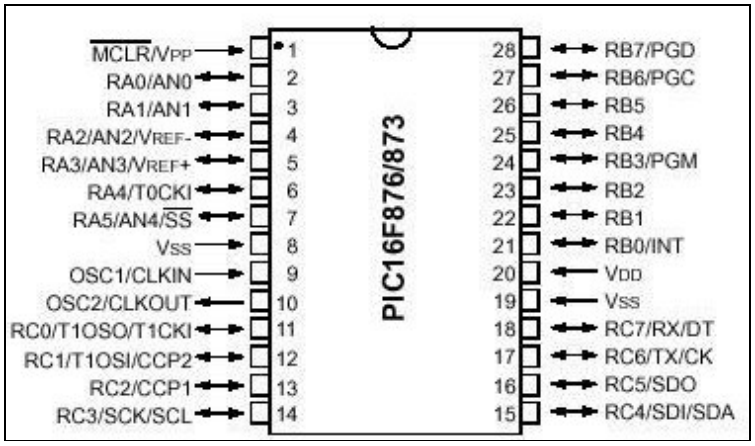

Figura 21

$\overline{MCLR}$/Vpp (pin 1). Entrada de RESET (activo a nivel bajo) o voltaje de entrada de programación. En la placa se utiliza como RESET del sistema. Normalmente esta pata está a nivel alto mediante una resistencia que la conecta a VDD. Al presionar un pulsador normalmente abierto se introduce un nivel bajo y por tanto provoca la inicialización del sistema. Gracias a la resistencia evitamos un cortocircuito entre VDD y VSS cuando se cierra el interruptor normalmente abierto.

VDD (pin 20). Entrada de alimentación. En la placa se utiliza un regulador que mantiene un nivel de aproximadamente unos 5 voltios.

VSS (pines 8 y 19). Masa.

OSC1/CLKIN (pin 9). Entrada del reloj externo. Se ha elegido un cristal de 4 Mhz.

OSC2/CLKOUT (pin 10). Salida del reloj externo.

7.3.1. Descripción de los puertos

Puerto A

De este puerto de 6 entradas/salidas sólo se van a emplear dos pines: RA3/AN3/Vref+ y RA4/TOCKI . El primero de ellos será la entrada al convertidor analógico/digital del microcontrolador. El segundo pin es la entrada, que denominaremos “captura”, que estará controlada por un pulsador. Cuando esta entrada esté a nivel bajo, será cuando indiquemos al microcontrolador que inicie el proceso de captura de la señal eléctrica (voltaje) por RA3.

Puerto B

Puerto de 8 pines de entradas/salidas. Todos se emplearán para controlar los pines A[0-7] del circuito integrado ISD1420. Con este puerto se pretende seleccionar las distintas opciones de reproducción que presenta el integrado ISD1420. La relación de conexión de este puerto con el ISD1420 es la siguiente:

PUERTO B	ISD1420
RB0-Pin21	A0-Pin1
RB1-Pin22	A1-Pin2
RB2-Pin23	A2-Pin3
RB3-Pin24	A3-Pin4
RB4-Pin25	A4-Pin5

PUERTO B	ISD1420
RB5-Pin26	A5-Pin6
RB6-Pin27	A6-Pin9
RB7-Pin28	A7-Pin10

Puerto C

Puerto de 8 pines de entradas/salidas que se va a emplear tanto para el control de los leds del sensor como para el control de la reproducción del dispositivo ISD1420. La relación de conexión es la siguiente:

PUERTO C	LEDs
RC0-Pin11	Led verde
RC1-Pin12	Led azul
RC2-Pin13	Led rojo

PUERTO C	ISD1420
RC6-Pin17	$\overline{\text{PLAYL}}$ - Pin23
RC7-Pin18	$\overline{\text{PLAYE}}$ - Pin24

7.4. El circuito oscilador

Todo microcontrolador necesita un circuito oscilador, cuyo diseño viene dado por el fabricante del dispositivo en función de la frecuencia de funcionamiento elegida. El microcontrolador elegido puede trabajar a una frecuencia de hasta 20Mhz; sin embargo, debido a que las necesidades de velocidad no son críticas, se ha elegido una frecuencia de 4Mhz ya que así consume menos el circuito.

Entre las distintas opciones de circuitos osciladores que muestra el fabricante, se ha elegido el circuito que emplea un cristal de cuarzo, en nuestro caso, un cristal de 4Mhz. Este circuito incluye además del cristal un par de condensadores cuyos valores están tabulados para las distintas frecuencias del cristal de cuarzo. Para una frecuencia de 4Mhz del cristal de cuarzo el fabricante da unos valores de los condensadores de van en el rango de 15pF-68pF. Para el circuito se han empleados dos condensadores de 33pF. En la siguiente figura se puede ver el esquema del circuito:

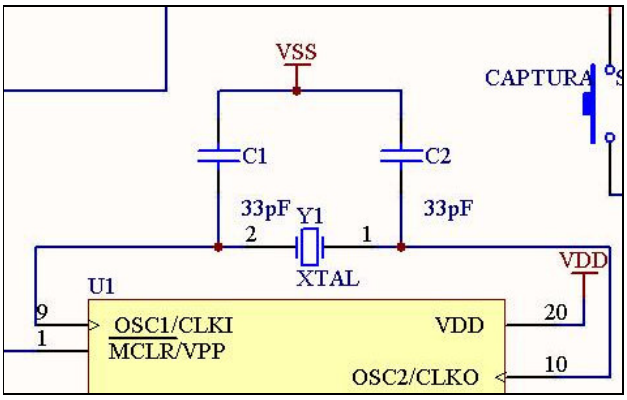

Figura 22

7.5. Los controles

Tanto para el reset como para el inicio del funcionamiento se emplea el mismo circuito. Ambas entradas del microcontrolador son activas a nivel de tensión bajo, por lo que normalmente estará conectada a V_{DD}. Para llevar el nivel bajo a estas entradas se utilizan dos pulsadores, normalmente abiertos. Para evitar el cortocircuito cada vez que se accione uno de estos dos pulsadores, se utilizan dos resistencias de 10KΩ, cada una asociada a una de las entradas. Los circuitos de estos dos pulsadores los podemos ver en la siguiente figura:

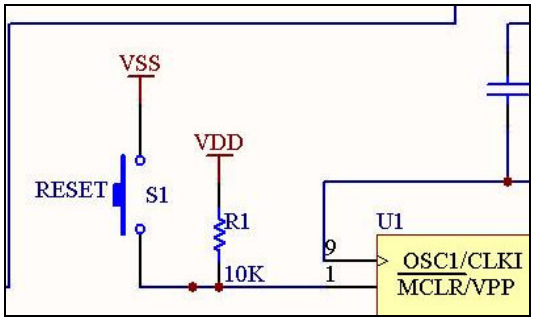

Figura 23

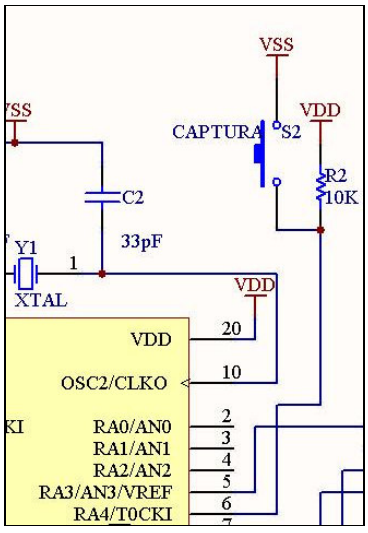

Figura 24

EL CIRCUITO REPRODUCTOR DE VOZ

8. EL CIRCUITO REPRODUCTOR DE VOZ

8.1. Introducción

El circuito reproductor de voz se encarga de reproducir los distintos mensajes, los nombres de los colores, que contiene en su memoria. Recibe una señal de control procedente del microcontrolador que le indica cuál de los distintos mensajes de voz que tiene almacenados es el que se debe reproducir. Debido a que no tiene demasiada potencia para atacar el altavoz, es necesario utilizar un amplificador. En la siguiente imagen podemos ver una fotografía del circuito reproductor de voz.

Figura 25

8.2. Integrados ISD

Estos dispositivos permiten almacenar mensajes de hasta varios minutos y tiene la gran ventaja de que se almacena en memoria no volátil. Se utiliza en muy diferentes aplicaciones: juguetes, grabadoras de voz portátiles, contestadores automáticos, sistemas de seguridad, etc.

Entre sus características destacan:

- Chip grabador/ reproductor de voz de fácil uso.
- Alta calidad de reproducción.
- Reproducción activa por nivel o por pulso.
- Modo automático de apagado.
- Entra automáticamente en modo standby seguidamente de un ciclo de reproducción o de grabación.
- Bajo consumo en standby.
- Direccionamiento de memoria para almacenar varios mensajes.
- Almacena los mensajes hasta 100 años.
- Oscilador interno.
- Permite 100.000 ciclos de grabación.

En el diseño de este proyecto se ha utilizado la serie ISD1400, que permite almacenar mensajes de corta duración con una alta calidad. En especial, se ha optado por el dispositivo ISD1420 que, como indican sus dos últimas cifras, permite almacenar mensajes con una duración de hasta 20 segundos.

8.3. ISD1420

En la siguiente figura tenemos una imagen de los pines del dispositivo ISD1420.

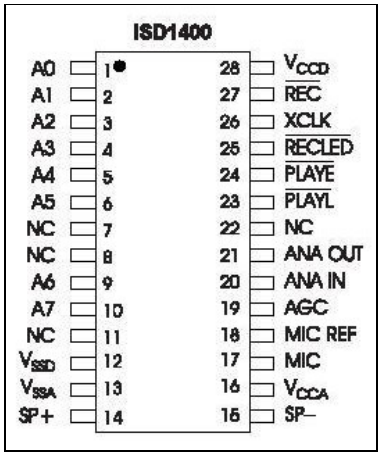

Figura 26

La descripción de los pines es la siguiente:

V_{CCA}/V_{CCD} (pines 16 y 28). Son las respectivas entradas de alimentación para los circuitos internos analógico y digital del ISD1400.

V_{SSA}/V_{SSD} (pines 13 y 12). Son las masas de los circuitos internos analógico y digital, respectivamente.

$\overline{REC}$ (pin 27) . Es la entrada que activa un ciclo de grabación cuando está a nivel bajo, que debe mantenerse durante todo el proceso de grabación. Esta señal tiene preferencia frente a la reproducción, es decir, si

se está reproduciendo un mensaje, y activamos $\overline{\text{REC}}$, inmediatamente se pasa a un ciclo de grabación.

$\overline{\text{PLAYE}}$ (pin 24). Empieza un ciclo de reproducción cuando se detecta una transición a nivel bajo. Esta reproducción se mantendrá hasta que internamente se encuentra un EOM (fin del mensaje).

$\overline{\text{PLAYL}}$ (pin 23). Empieza un ciclo de reproducción cuando se detecta una transición a nivel bajo. Esta reproducción se mantendrá hasta que $\overline{\text{PLAYL}}$ vuelve a pasar a nivel alto.

$\overline{\text{RECLED}}$ (pin 25). Salida que permanecerá a nivel bajo durante un ciclo de grabación. Esta salida puede usarse para controlar un led que nos indique cuándo estamos en un ciclo de grabación (ayudará a la persona que está grabando el mensaje para saber cuando debe hablar).

MIC (pin 17). Entrada del micrófono. Transfiere su señal al pre-amplificador interno. El control de ganancia automática (AGC) controla la ganancia de este pre-amplificador.

MIC REF (pin 18). Entrada invertida para el pre-amplificador del micrófono; elimina el ruido.

AGC (pin 19). Ajusta dinámicamente la ganancia del pre-amplificador interno dependiendo de su nivel de voltaje.

ANA OUT (pin 21). Salida del pre-amplificador para el usuario. La ganancia de voltaje del amplificador está determinado por el nivel de voltaje en el pin AGC.

ANA IN (pin 20). Transfiere la señal de entrada al chip para grabar.

XCLK (pin 26). Entrada para una señal externa de reloj. Internamente el dispositivo tiene un reloj que garantiza una frecuencia adecuada para la grabación y reproducción de los mensajes. Si no se va a utilizar esta entrada se debe conectar a masa.

SP+, SP- (pines 14 y 15). Salida directa hacia el altavoz si tiene una impedancia mínima de 16Ω.

A0-A7 (pines 1, 2, 3, 4, 5, 6, 9 y 10). Entradas de dirección que tienen dos funciones dependiendo del nivel de tensión en los dos bits más significativos (MSB) de la dirección. Si cada uno de los dos bits más significativos están a nivel bajo, el resto de las entradas son interpretadas como bits de direcciones y son usadas como direcciones de comienzo para el ciclo de grabación o de reproducción. Cuando los bits más significativos, A6 y A7, están a nivel alto, entonces entramos en los modos operacionales que nos ofrece este dispositivo.

8.4. Descripción de los modos operacionales del ISD1420

Este dispositivo está diseñado para trabajar en varios modos de operación y permite una mayor funcionalidad con el mínimo uso de componentes externos. Cuando los bits más significativos (A6 y A7) están a nivel alto, el resto de bits de direcciones son interpretados como bits de modo y no como bits de direcciones. Por lo tanto, el modo operacional y el modo de direccionamiento son incompatibles.

Existen cuatro modos operacionales que son los siguientes:

- A0 –Mensaje en cola-. Permite al usuario saltar a través de los mensajes sin el conocimiento de las direcciones físicas de cada uno de los mensajes. Cada pulso bajo de entrada de control causa que el puntero interno de dirección salte al siguiente mensaje.

- A1 –Borrar los marcadores de final de mensaje-. Permite grabar secuencialmente mensajes para ser combinados en un mensaje simple con un sólo marcador de fin de mensaje. Es decir, en este modo, los mensajes son grabados secuencialmente y después serán reproducidos como un único mensaje.

- A3 –Mensaje continuo-. Permite repetir de forma continua el mensaje localizado al principio del espacio de direcciones.

- A4 –Direccionamiento consecutivo-. Durante una operación normal, el puntero de dirección se inicializará cuando un mensaje es reproducido hasta un marcador de fin de mensaje. Este modo operacional inhibe la inicialización del puntero de dirección, permitiendo que los mensajes puedan ser grabados o reproducidos consecutivamente.

8.5. La circuitería del ISD1420 en la placa

Debido a que en la placa el ISD1420 sólo actúa como reproductor, no no necesita ninguna circuitería para las entradas analógicas de este dispositivo.

Para controlar los modos operacionales (pines [A0-A7]) se ha utilizado el puerto B, [RB0-RB7], del microcontrolador. Para actuar sobre los controles de reproducción del ISD se han utilizado dos pines del puerto C del micro, RC6 y RC7.

Debido a que durante las pruebas se observó que el sonido del altavoz no era suficiente cuando era atacado directamente por el ISD, se utilizó un circuito amplificador de audio. El fabricante del ISD propone varios circuitos para este fin; entre ellos, uno que usa un circuito integrado, MC34119, que se corresponde con un amplificador de audio.

En la siguiente figura vemos el esquema del circuito:

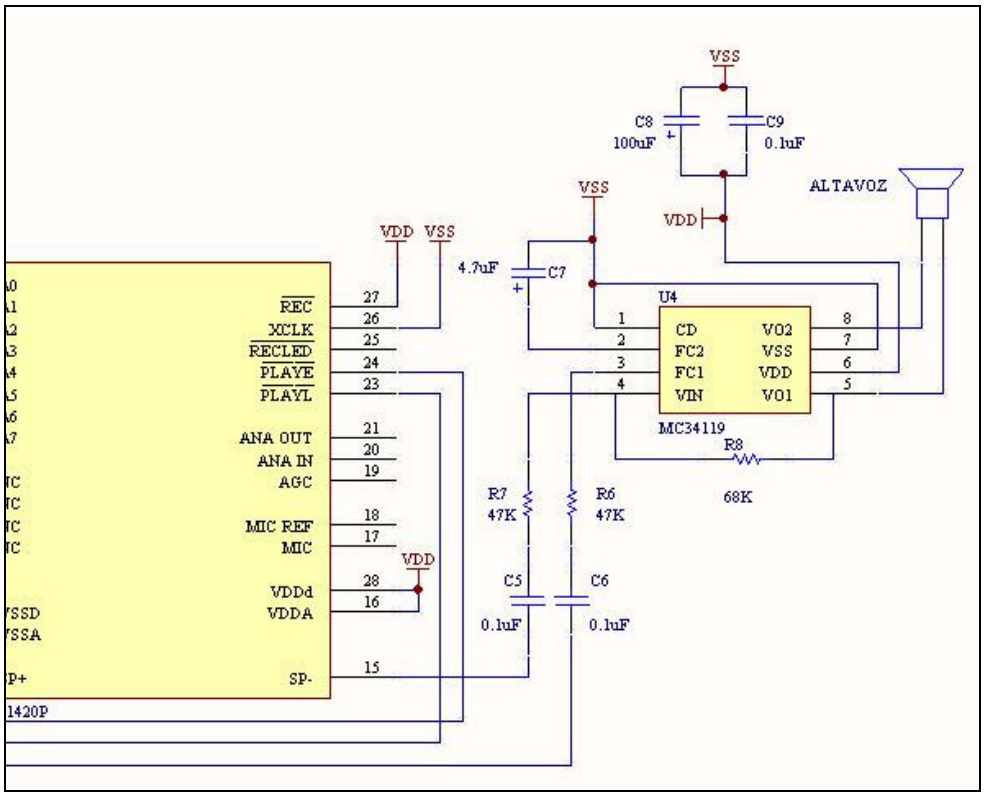

Figura 27

EL PROGRAMADOR DEL ISD

9. EL PROGRAMADOR DEL ISD

9.1. Introducción

Para introducir los nombre de los colores en la memoria del dispositivo ISD se ha construido la placa de la figura.

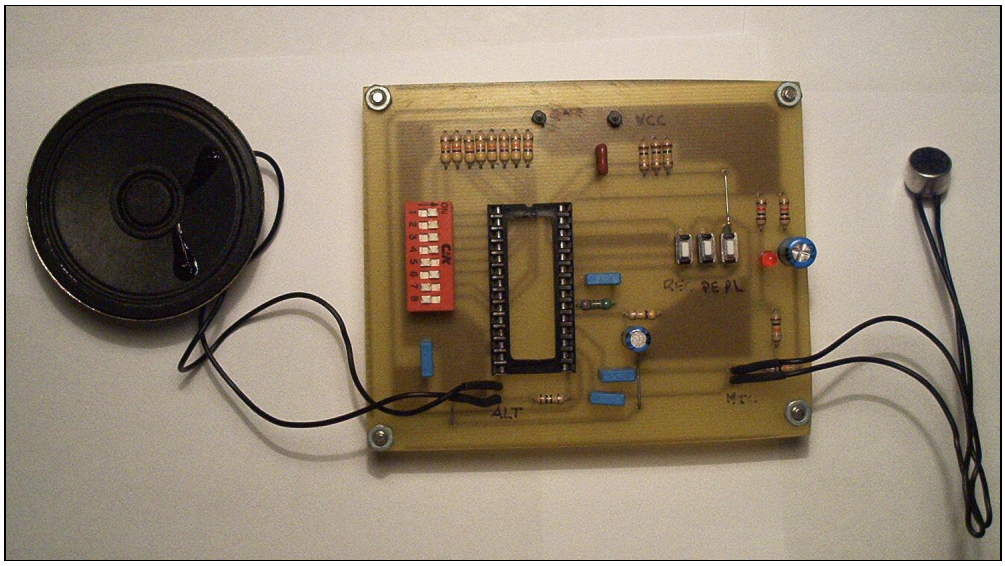

Figura 28

Como la memoria del ISD1420 no es volátil, una vez programado y verificado cada uno de los distintos mensajes, sólo hay que sacarlo del zócalo de la placa anterior e introducirlo en el zócalo de la placa para la que ha sido programado.

9.2. El circuito de la placa

En la hoja de datos de los dispositivos de la serie ISD1400 aparecen distintos circuitos propuestos por el fabricante para la grabación de los mensajes. Para la construcción de la placa anterior se ha utilizado una combinación de los circuitos de las siguientes figuras para obtener el diseño del circuito de la placa del grabador, que se encuentra en el anexo.

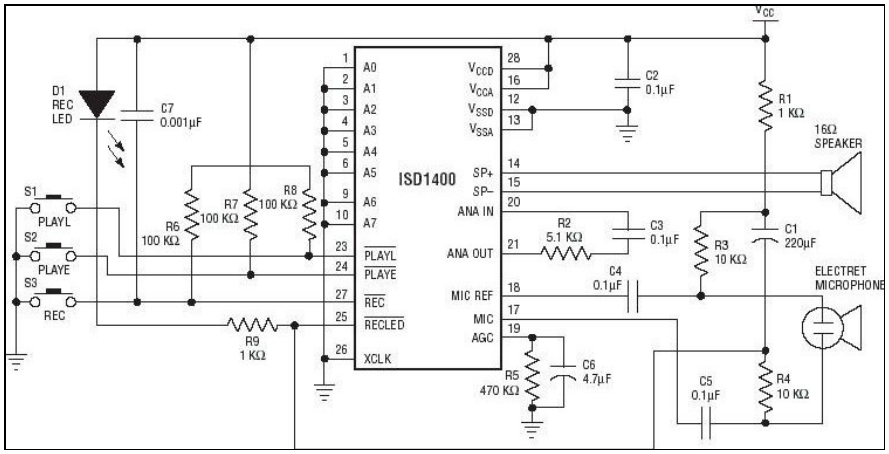

Figura 29

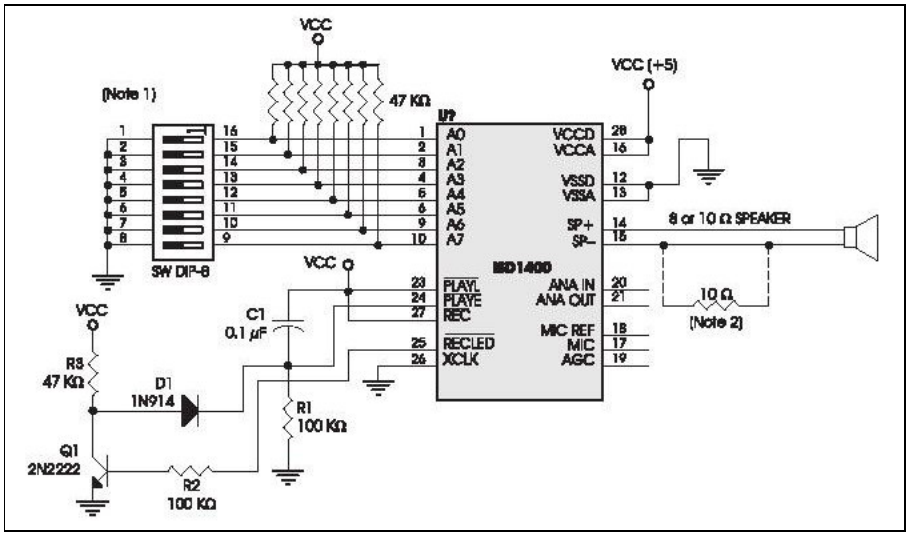

Figura 30

Como se puede apreciar en el esquema del anexo, el circuito es el mismo que el de la figura 22, pero añadiendo los micro-interruptores y las resistencias a las entradas [A0-A7], como aparece en el esquema de la figura 23. Como el altavoz empleado en la construcción tiene una impedancia de 8Ω, hay que añadir una resistencia de 10Ω en serie para aumentar la resistencia a la salida de SP- y SP+.

9.3. Modo de uso de la placa

Con la placa sin tensión, introducimos el dispositivo ISD en su zócalo de 28 pines.

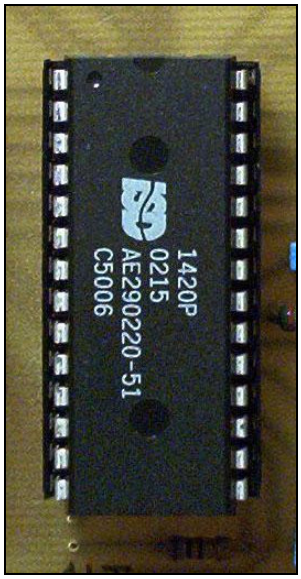

Figura 31

Una vez que tenemos el ISD en su zócalo alimentamos la placa con una fuente de alimentación de 5 voltios entre los terminales Vcc y GND que se

muestran en la siguiente figura.

Figura 32

Ahora podemos grabar un mensaje pulsando el botón REC, que inicia un ciclo de grabación. Este ciclo permanecerá activo mientras mantengamos pulsado el botón REC. Durante el ciclo de grabación el led rojo estará encendido y sólo se debe dictar el mensaje usando el micrófono.

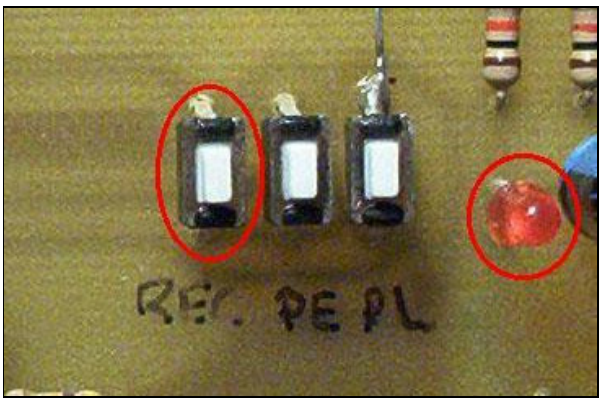

Figura 33

Una vez que el mensaje ha sido almacenado en el dispositivo, debemos verificar que éste está correctamente grabado, y que no existe ningún tipo de problemas. Para ello, únicamente hay que reproducirlo y comprobar que está bien. Como vimos anteriormente, existen dos tipos de reproducciones

distintas: una activada por pulso y la otra activa por nivel. Mediante la primera, cuando apliquemos un pulso de nivel bajo a su correspondiente entrada de control, PLAYE, el mensaje se reproducirá hasta el final. En la segunda, el mensaje se reproducirá mientras exista un nivel bajo en su entrada de control PLAYL.

Para utilizar la reproducción del mensaje de forma continuada, tendremos que pulsar una vez PE, y el mensaje se reproducirá hasta el final.

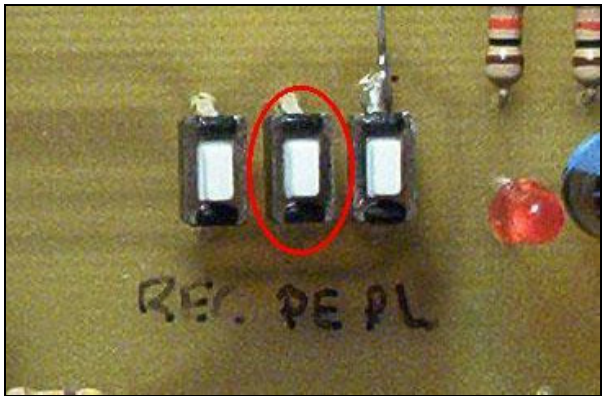

Figura 34

Si, por el contrario, no queremos una reproducción continua, sólo tenemos que pulsar PL.

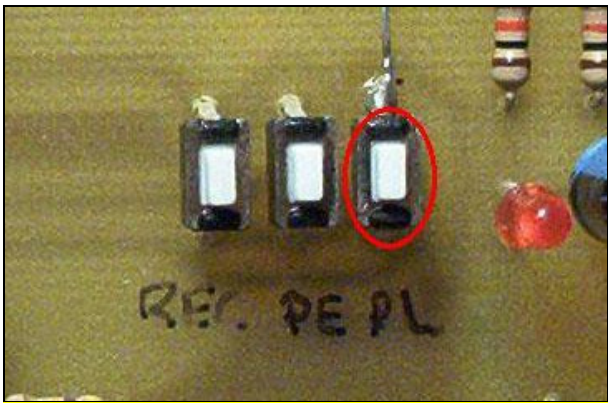

Figura 35

Todo lo visto anteriormente nos valdría para la grabación y reproducción de un sólo mensaje. Sin embargo, si lo que queremos es utilizar los modos operacionales para la grabación de varios mensajes, la reproducción de un determinado mensaje de entre varios, etc, tendremos que usar los micro-interruptores que se incluyen en la placa.

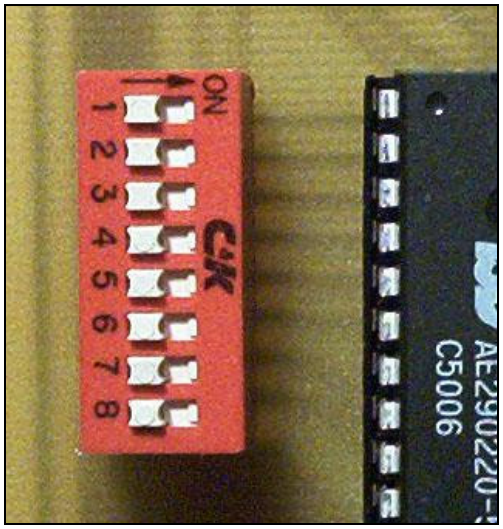

Figura 36

Para activar los modos operacionales, se deben colocar los micro-interruptores 7 y 8 en posición de ON. En la siguiente tabla se presenta una relación entre los micro-interruptores y los distintos modos operacionales en los que puede trabajar el dispositivo, y que se explican en el 6.4.

MICRO - INTERRUPTOR	MODO OP.
1	Mensaje en cola
2	Borrar EOM
4	Mensaje continuo
5	Dereccionamiento consecutivo

SOFTWARE

10. SOFTWARE

10.1. Introducción

Para la realización de este proyecto se han empleado varios tipos de software: unos para el desarrollo, simulación y programación del microcontrolador, y otros para la creación de los esquemas eléctricos y de la PCB.

Para el desarrollo, la simulación y programación del microcontrolador, se han utilizado tres programas distintos: MPLAB, CC5X y el ICPROG. MPLAB es el software que el fabricante del microcontrolador, Microchip, pone gratuitamente a disposición de los usuarios para la creación y simulación de programas escritos en ensamblador. CC5X es un compilador de C que permite la creación de programas de hasta 1K para varios de los microcontroladores de Microchip. Y, por último, tenemos el ICPROG, que permite la grabación en la memoria flash del microcontrolador del programa desarrollado por el usuario.

10.2. MPLAB y CC5X

MPLAB es un entorno de desarrollo integrado (IDE) bajo Windows para microcontroladores de Microchips. MPLAB permite escribir, depurar y optimizar las aplicaciones desarrolladas para los microcontroladores. Incluye un editor de texto, un simulador y un gestor de proyecto.

Otra posibilidad de MPLAB es la utilización de compiladores de lenguajes de más alto nivel, como C y BASIC. Entre ellos se ha elegido el compilador de C, CC5X, de distribución gratuita, que puede generar módulos con código de 1Kbyte.

La siguiente imagen se corresponde con el entorno de desarrollo del MPLAB.

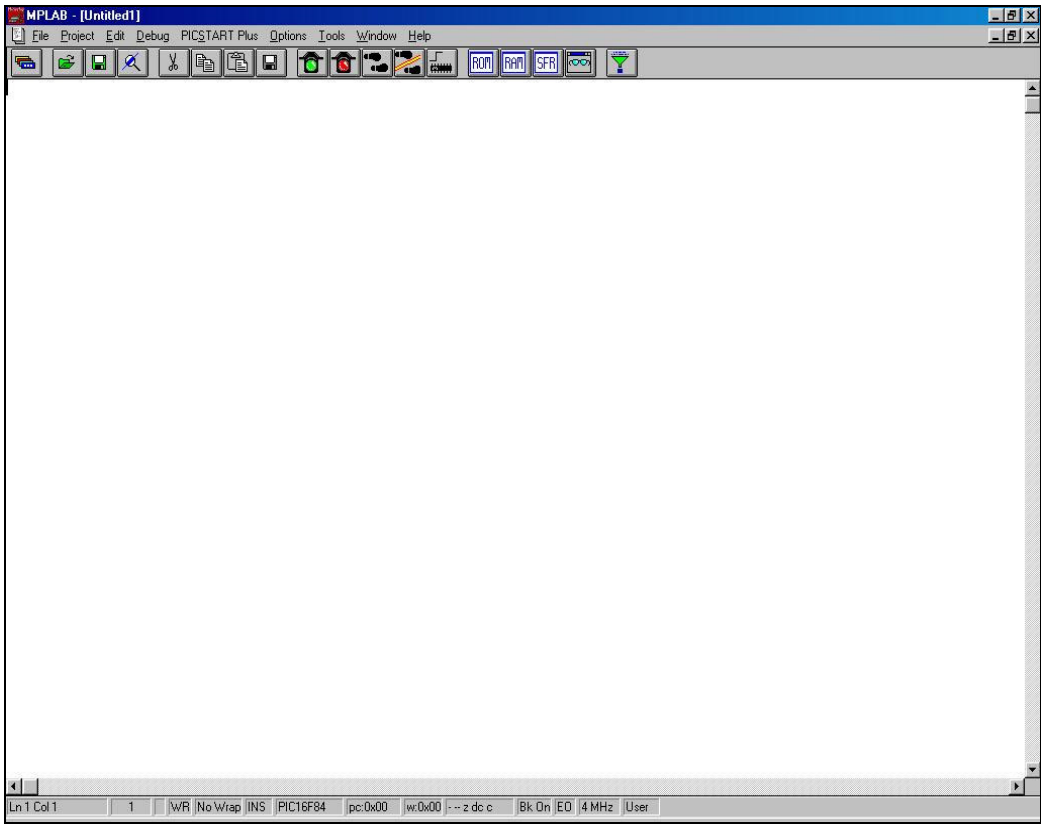

Figura 37

10.2.1. Creación de un proyecto utilizando MPLAB y CC5X

Para poder programar un microcontrolador en lenguaje C necesitamos un compilador de C para el microcontrolador. El compilador permite escribir el programa en C que, al ser un lenguaje de programación de más alto nivel que el ensamblador, nos va a facilitar la programación. La función del compilador es

traducir el programa escrito en C a lenguaje de más bajo nivel, lenguaje ensamblador y, una vez que ya lo tenemos, el ensamblador lo traduce al lenguaje que entiende internamente microcontrolador, en nuestro caso, el PIC16F876. El IDE MPLAB permite utilizar un compilador de C, que en nuestro caso será el CC5X. Hay que destacar que el MPLAB y el CC5X son software de distribución gratuita y demostración respectivamente.

Antes de poder utilizar el compilador CC5X, que está instalado en una carpeta de nuestro disco duro, tendremos que copiar los archivos CC5X.M y TLCC5X.INI de la carpeta del compilador a la carpeta donde se encuentra instalado el MPLAB. Una vez hecho esto, ya podemos iniciar MPLAB. Vamos al menú "Project" y seleccionamos "Install Language" (Project->Install Language), apareciendo la siguiente ventana:

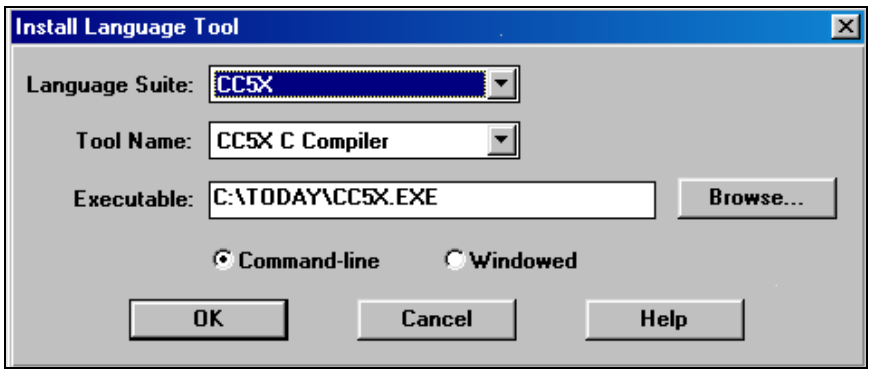

Figura 38

En "Language Suite" seleccionamos CC5X. Luego, en "Tool Name" seleccionamos CC5X C Compiler, y, en "Executable" buscamos la carpeta donde tenemos instalado el CC5X el archivo CC5X.exe. Tenemos que elegir

Command -line, ya que el creador del compilador nos dice que sólo trabaja de este modo. Terminado esto, pulsamos “OK” y ya tenemos instalado el compilador CC5X como compilador de C.

Para crear un nuevo proyecto, vamos al menú “Project” y seleccionamos “New Project” (Project ->New Project). Aparece una ventana en la que podemos elegir la localización del archivo y el nombre del proyecto (*.pjt). La extensión de nombre de archivos pjt se corresponde con los archivos de proyecto en MPLAB.

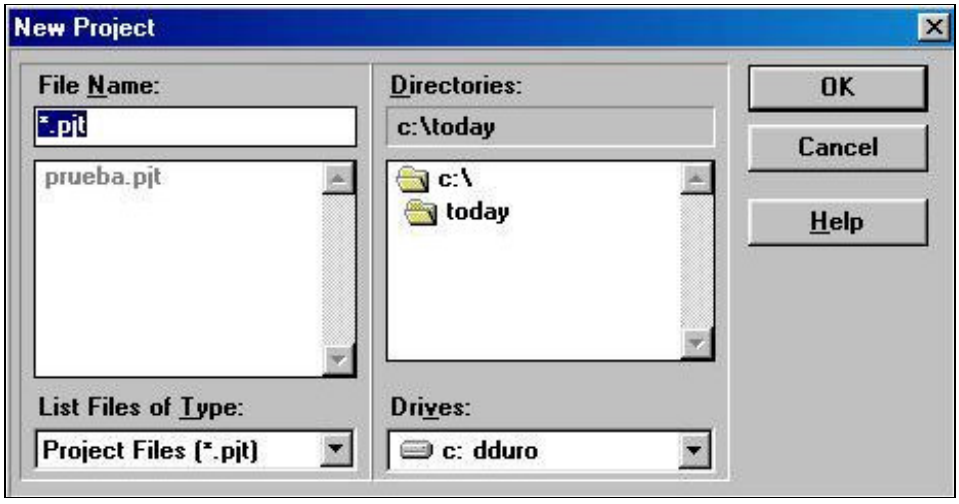

Figura 39

Una vez grabado el proyecto; por ejemplo, con el nombre “prueba”, hay que seleccionar las opciones del proyecto. En el menú “Project” elegimos “Edit Project” (Project->Edit Project). Aparece la ventana de la página siguiente.

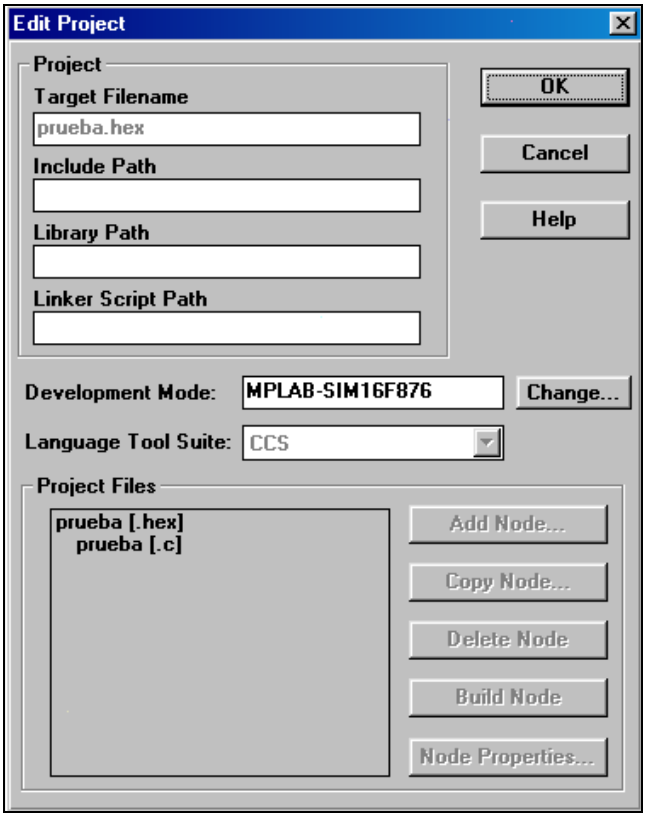

Figura 40

MPLAB nos sugiere un nombre para el "Target Filename" (nombre del archivo resultado), que será el archivo hexadecimal. En la pestaña "Language Tool Suite" elegimos el nombre de nuestro compilador, CC5X. Para cambiar la opción "Development Mode" pulsamos "Change", apareciendo una nueva ventana:

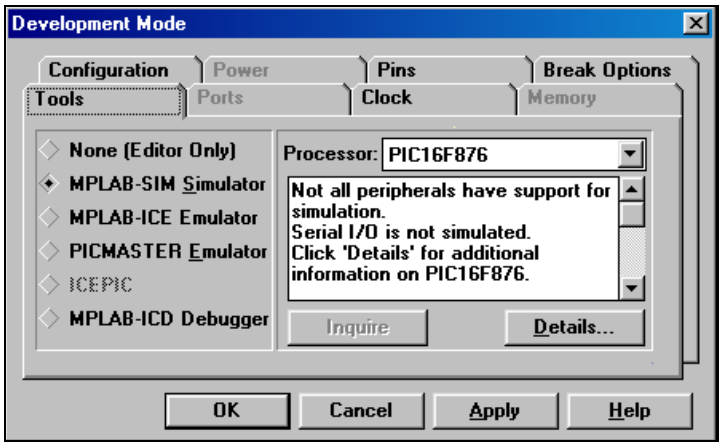

Figura 41

Esta ventana nos permite elegir el microcontrolador, en nuestro caso, el 16F876, varias opciones de éste, y la posibilidad de la simulación. En "Processor" seleccionamos el dispositivo 16F876 y señalamos la opción "MPLAB-SIM Simulator" para poder simular el programa posteriormente. Una vez que tengamos seleccionado lo anterior, pulsamos "OK" y volvemos a la ventana de "Edit Project" inicial.

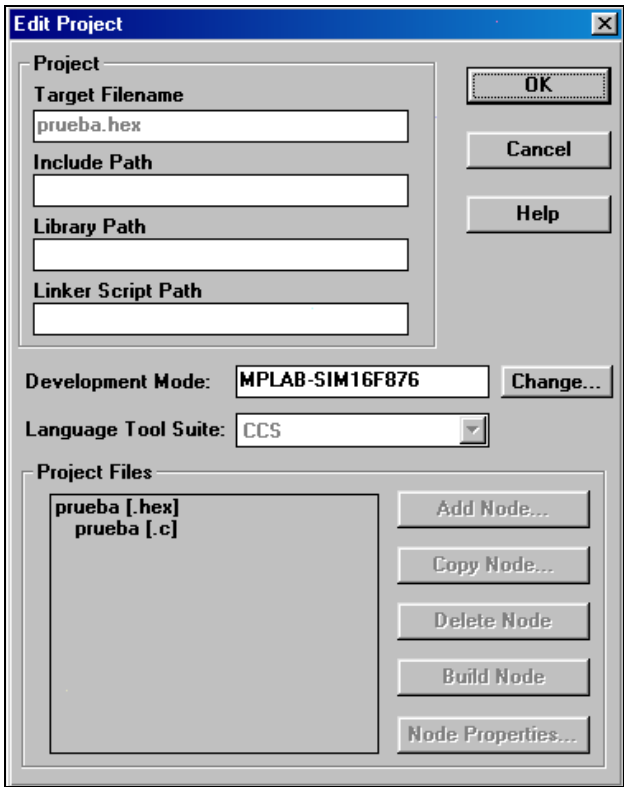

Figura 42

En “Project Files” seleccionamos el archivo con extensión “*.hex”, al hacer esto se nos activará el botón “Node Properties”. Lo pulsamos y accederemos a una nueva ventana.

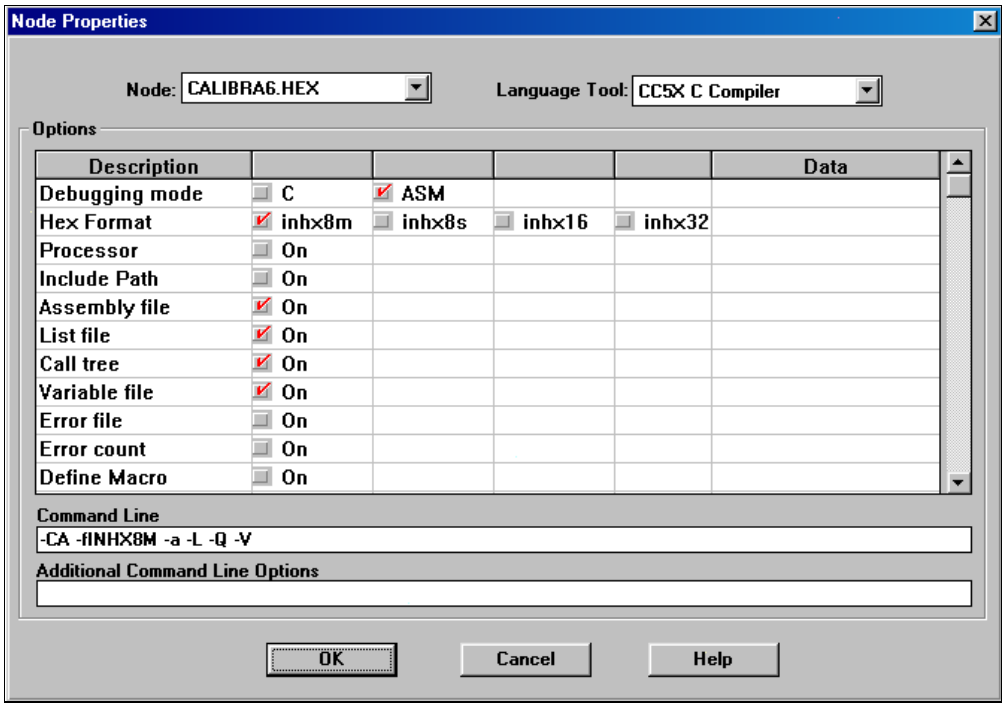

Figura 43

El contenido de esta ventana variará en función del lenguaje que tengamos seleccionados, que, en nuestro, caso es CC5X. Cada vez que elijamos algo, se producirá un cambio en la línea de comandos “Command Line” que aparece en la parte inferior de la pantalla. Una de las opciones que podemos destacar es “Debugging mode”, es decir, el modo de depuración. Nos da dos opciones, en C o en ASM (ensamblador); si elegimos C, a la hora de simular el programa paso a paso veremos como va evolucionando el programa en nuestro código fuente en C. Sin embargo, si elegimos ASM, entonces apareceráe l código fuente en C y, de manera automática, una pantalla con el código ensamblador correspondiente a nuestro programa en C.

Una vez que hayamos realizado todas nuestra elecciones, salimos de esta ventana pulsando “OK”, volviendo a paracer la ventana de “Project Edit”. De nuevo en esta ventana, seleccionamos en “Project Files”, el archivo con la extensión , (*.hex). Una vez que ya hemos salido de la ventana de “Node properties” se activará el botón “Add Node”, lo pulsamos y aparecerá la siguiente ventana:

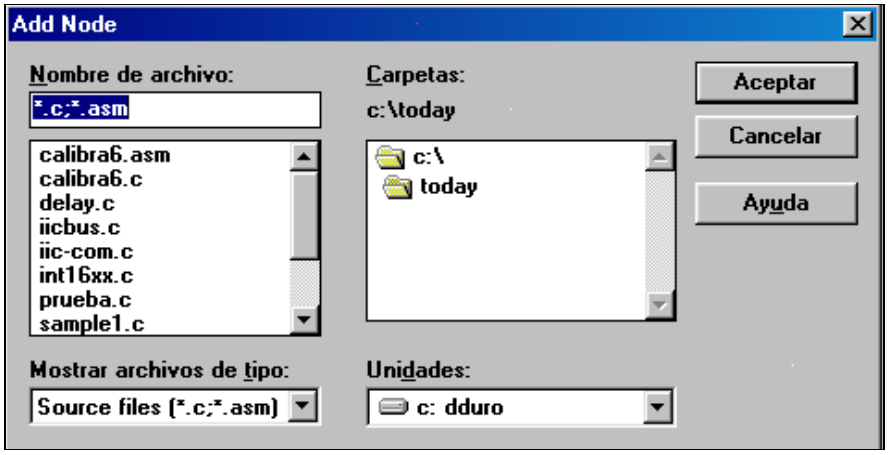

Figura 44

Como en el ejemplo estamos suponiendo que tenemos nuestro código fuente en C en el archivo “prueba.c”, sólo tenemos que indicar dónde se encuentra nuestro archivo con el código fuente, ya sea en ensamblador o en C. Como estamos haciendo uso de un compilador de C, sólo podemos elegir los archivos con extensión (*.c), y en nuestro caso particular elegiríamos el archivo “prueba.c”. Pulsamos Aceptar y de nuevo volveremos a la ventada de “Project Edit”. Como ya hemos terminado de seleccionar lo s cambios para el proyecto, pulsamos “OK” para salir de la ventana y volvemos al entorno de desarrollo del MPLAB.

El archivo “prueba.c” que se ha utilizado en el ejemplo anterior se supone ya creado. Podemos crear este archivo desde cualquier editor de t exto que nos ofrece Windows, o podemos hacer uso del propio editor del MPLAB.

Para crear un código fuente nuevo nos vamos al menú “File” y elegimos “New” (Files ->New). Al hacer esto volveremos directamente a la ventana principal del programa. Basta con situar el ratón encima de la ventana blanca y teclear nuestro código.

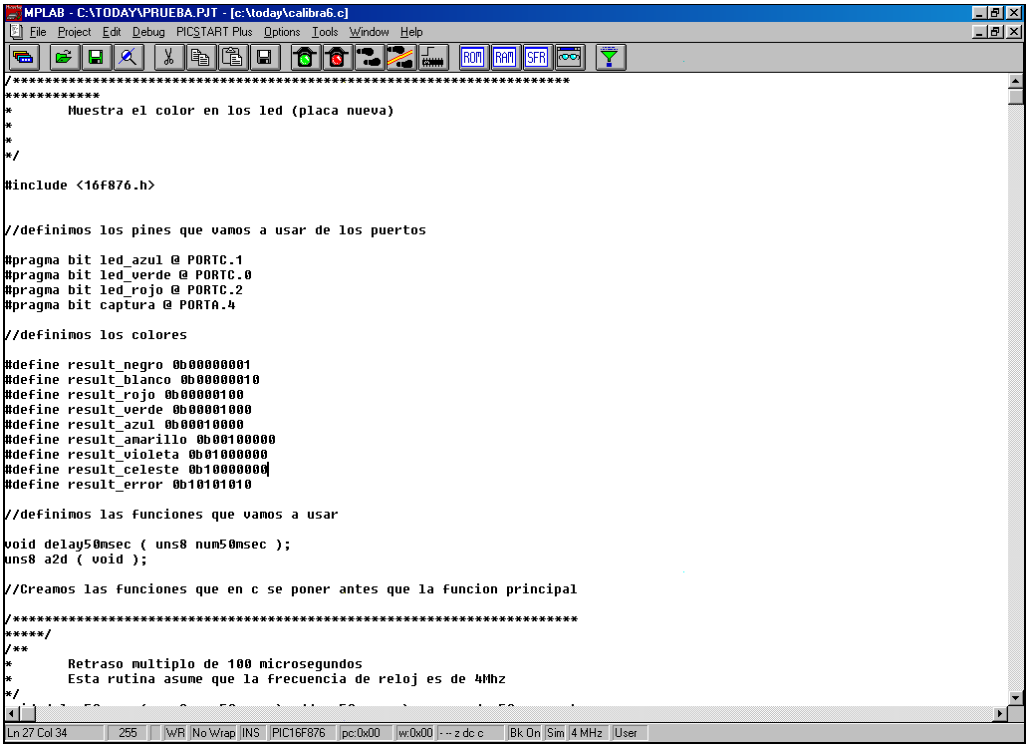

Figura 45

Sólo nos falta guardar el código creado. Para ello vamos a “File” y seleccionamos “Save as...” (File->Save as...), apareciendo la ventana siguiente:

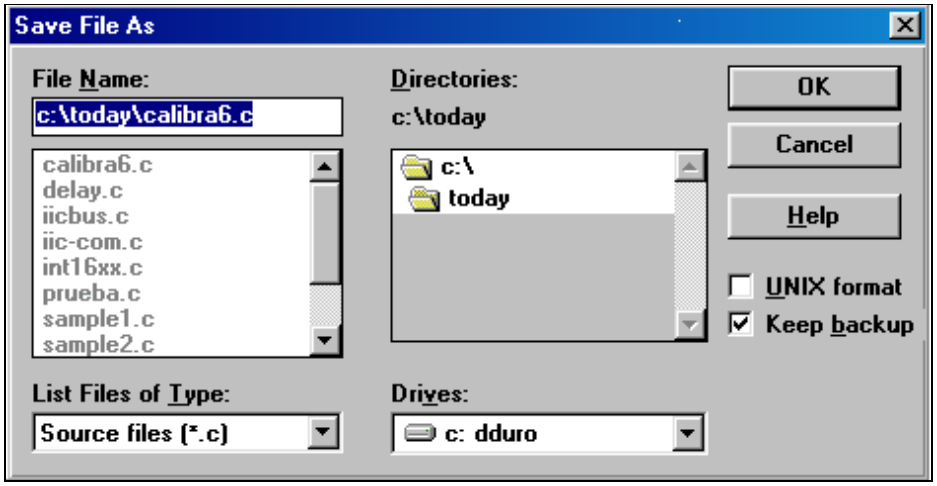

Figura 46

Normalmente, en esta ventana, por defecto nos guarda los archivo con la extensión (*.asm), pero, como anteriormente le hemos indicado que el código fuente está en lenguaje C, por defecto ya nos guarda el archivo con extensión (*.c).

10.2.2. Ensamble del archivo fuente

El ensamble del archivo se puede realizar de varias maneras: mediante la escritura de comandos en línea en Msdos o usando las aplicaciones de Windows. Usaremos la segunda forma por ser más cómoda. Vamos al menú “Project” y elegimos “Build all” (Project->Build all”). De ese modo se ejecutará tanto el compilador CC5X para la obtención del código asm a partir del código en C, y posteriormente, el ensamblador del MPLAB convierte este, en el código entendible por el microcontrolador. Una vez realizado el proceso de ensamble, aparecerá la siguiente vetana “Build Results”:

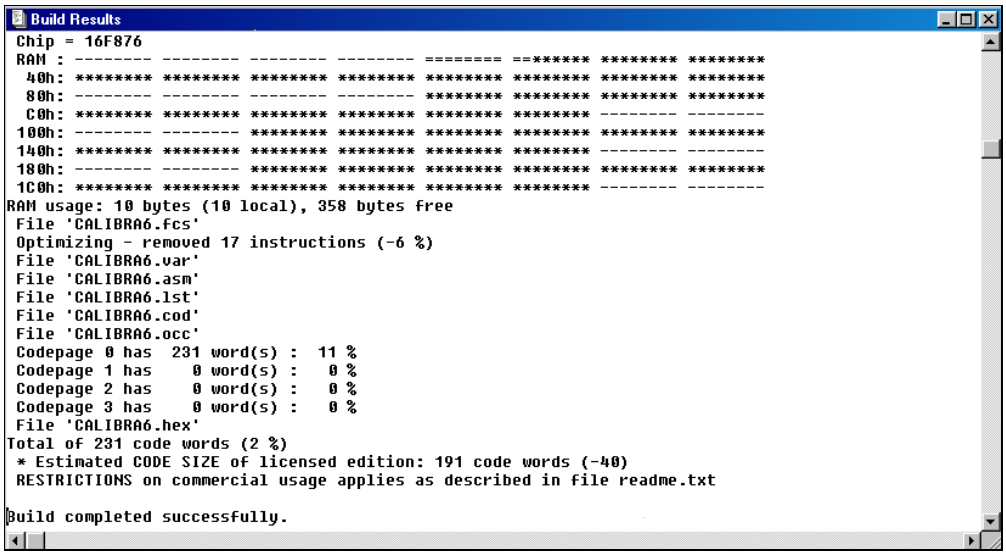

Figura 47

Si existen errores, los mensajes de error nos informaran sobre ellos y, haciendo doble clic en ellos, nos situamos en la línea de código fuente que contiene dicho error. Una vez que hayamos realizado los cambios oportunos para intentar solucionarlo, debemos volver a ensamblar el código mediante (Project ->buildall).

Cuando no haya errores en el código fuente, aparecerá en la ventana “Build Results” el mensaje “Build co mpleted successfully”. Ya estamos en disposición de simular el proyecto.

10.2.3. Simulación y depuración de un programa

Ya que está el programa ensamblado podemos simularlo. La simulación permite verificar que el programa hace realmente lo deseado. Todas las herramientas de simulación se encuentran dentro del menú “Debug”, en la barra de herramientas de MPLAB.

En primer lugar debemos inicializar el sistema para asegurarnos de que nos encontramos en las condiciones iniciales. Para ello nos vamos al menú "Debug " y dentro de este a "Run" seleccionando "Reset" (Debut->Run ->Reset). A continuación haremos una simulación paso a paso; para ello, volvemos al menú "Debug" y dentro de "Run" seleccionamos "Step" (Debug ->Run ->Step). Este tipo de simulación nos permite ejec utar línea por línea del código. La primera vez que ejecutemos una simulación paso por paso, aparecerá el código del programa escrito en ensamblador, y como comentario, nuestro código en C, ya que fue una de las opciones que elegidas en el menú de "Project ->Edit Properties". La línea de código de ensamblador que se ejecutará la siguiente vez que elijamos una simulación por pasos estará resaltada en colornegro.

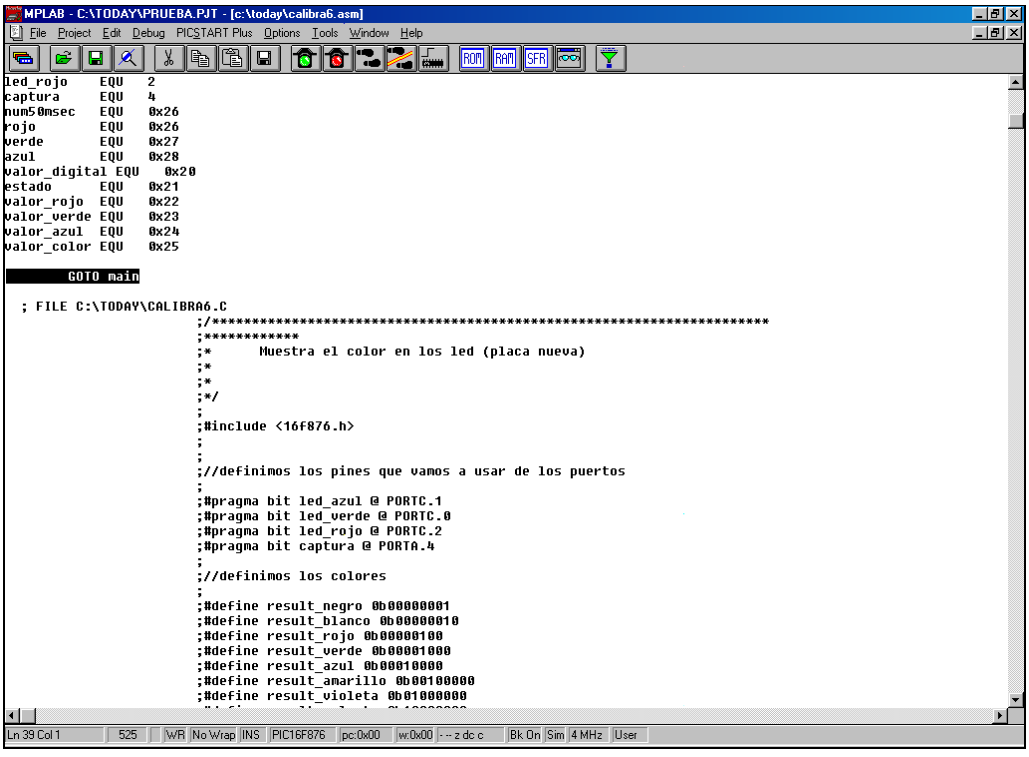

Figura 48

10.2.4. Ventanas de seguimiento

Para seguir la evolución durante la ejecución del programa de la memoria ram, de los registros especiales y de la memoria eeprom, necesitamos las ventanas de seguimiento que podemos encontrar en el menú "Windows" de la barra de herramientas del MPLAB.

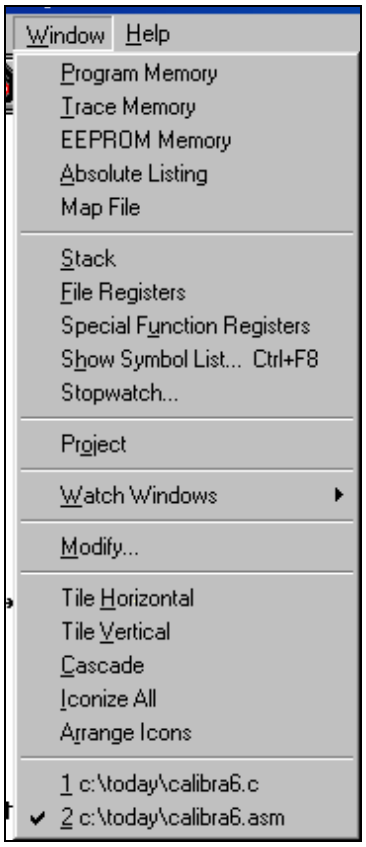

Figura 49

Cada vez que se seleccione una de estas ventanas, se abrirá ésta en el entorno del MPLAB. Debido a que a veces necesitaremos más de tres ventanas es recomendable utilizar las opciones que aparecen en el menú "Windows" al final, como son "Tile Horizontal" y "Tile Vertical", que ordenarán

las ventanas de forma que podamos visualizarlas de forma más cómoda, como podemos apreciar en la siguiente figura:

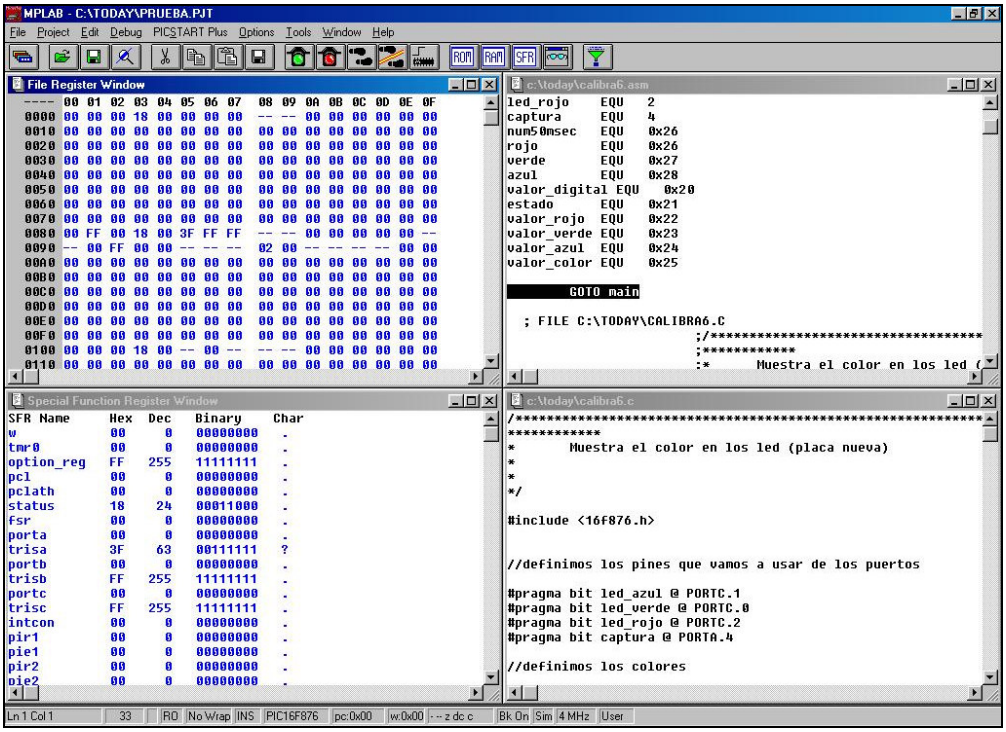

Figura 50

Otras de las posibles soluciones para poder controlar mejor la evolución de determinados registros durante la ejecución del programa es mediante la creación de una ventana en la que introduciremos aquellos valores QUE nos interesan especialmente. Para ello, nos vamos dentro del menú “Windows” y, dentro de éste, a “Watch Windows”, seleccionando “New Watch Windows...” (Windows->Watch Windows->New Watch Windows), con lo que aparece la imagen de la siguiente página.

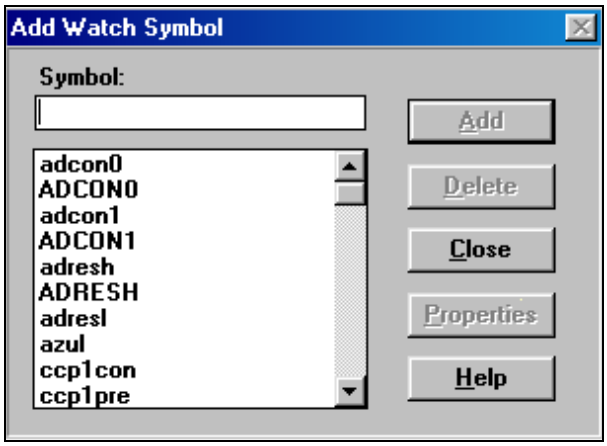

Figura 51

Seleccionamos las variables que queramos introducir en la ventana de seguimiento. Una vez seleccionada, pulsamos “Add” y, cuando las tengamos todas introducidas, entonces pulsamos “Close”. Entonces, ya tendremos en nuestra ventana de seguimiento las variables seleccionadas y podemos ver su evolución durante la ejecución del programa durante la simulación.

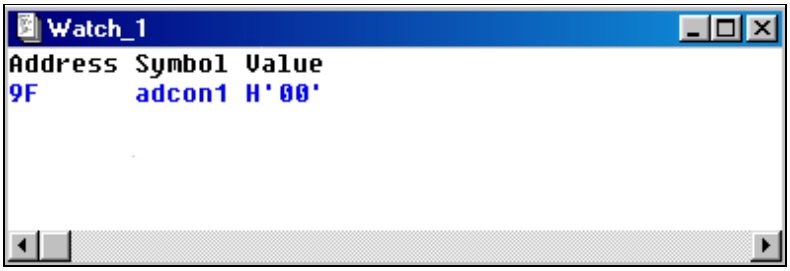

Figura 52

En nuestro caso sólo se ha añadido el registro especial “adcon1” como se observa en la figura superior, donde se ve que actualmente tiene un valor hexadecimal de 0.

10.2.5. Punto de interrupción

Si hacemos una ejecución entera, es decir, no paso a paso, el programa se ejecutará desde el principio hasta el final de forma inmediata. Esto impedirá que podamos seguir el cambio de las variables importantes dentro del programa. Para utilizar esta opción y tener simultaneamente controladas a las variables podemos introducir el denominado punto de interrupción. Éste marcará hasta dónde se ejecutará el programa; así, cuando llegue a ese punto la ejecución continua del programa parará y podremos observar cómo han evolucionado las variables. Para activar esta opción, tenemos que hacer clic sobre la ventana del código en asm, una vez activa la ventana, pulsamos el botón derecho del ratón y aparecerá un menú emergente. Dentro de éste seleccionamos “Break point(s)” justo cuando el cursor se encuentre en la línea del código fuente que queremos que sea punto de interrupción del programa. Al hacer esto, automáticamente cambiará de color la línea.

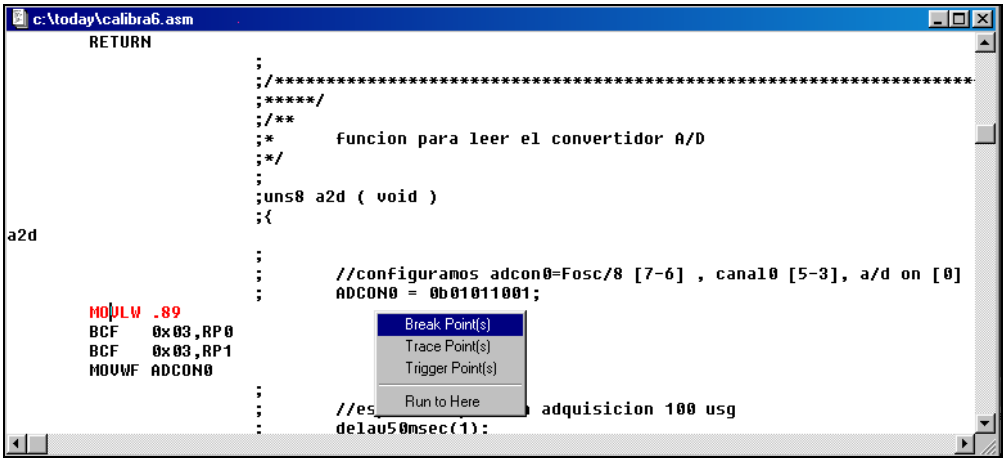

Figura 53

Una vez que tengamos marcado nuestra punto de interrupción, inicializamos la ejecución del programa mediante un “Reset” (Debug->Run->Reset), y seleccionamos una ejecución continua: “Run” (Debug->Run->Run).

10.2.6. Obtención del archivo *.hex

Una vez que hemos simulado y depurado el programa, sólo nos queda obtener el archivo (*.hex) para programar el microcontrolador. Este archivo se crea automáticamente cada vez que pulsamos “Build All” (Project->Build All) en el directorio donde se encuentra nuestro código fuente. Una vez que tenemos este archivo basta con llevarlo hasta el microcontrolador. Para ello, utilizaremos otro software denominado ICPROG, que detallaremos más adelante.

CÓDIGO FUENTE

11. CÓDIGO FUENTE

```
/*********************************************************************
******
*       Programa final
*
*
*/

#include <16f876.h>


//definimos los pines que vamos a usar de los puertos

#pragma bit led_azul @ PORTC.1  //definimos los pines del sensor
#pragma bit led_verde @ PORTC.0
#pragma bit led_rojo @ PORTC.2
#pragma bit captura @ PORTA.4
#pragma bit A0 @ PORTB.7       //definimos los bits del puerto B con
lo de los controles del isd
#pragma bit A1 @ PORTB.6
#pragma bit A2 @ PORTB.5
#pragma bit A3 @ PORTB.4
#pragma bit A4 @ PORTB.3
#pragma bit A5 @ PORTB.2
#pragma bit A6 @ PORTB.1
#pragma bit A7 @ PORTB.0
#pragma bit playe @ PORTC.7
#pragma bit playl @ PORTC.6
```

```
//definimos los colores

#define result_negro 1
#define result_blanco 2
#define result_rojo 3
#define result_verde 4
#define result_azul 5
#define result_amarillo 6
#define result_violeta 7
#define result_celeste 8
#define result_error 9

//definimos las funciones que vamos a usar

void delay50msec ( uns8 num50msec );
uns8 a2d ( void );
void mensaje ( uns8 num_mensaje );


//Creamos las funciones

/************************************************************************
*****/
/**
*       Retraso múltiplo de 100 microsegundos
*       Esta rutina asume que la frecuencia de reloj es de 4Mhz
*/
void delay50msec ( uns8 num50msec )  //num50mseg--> numero de
50mseg retraso
{
uns8  inner;
```

```
//insertamos código ensamblador

#asm

delay_var   bcf   INTCON,2   ;desconectamos flag de rebosamiento
            movlw 0x3c       ;complemento hex de 195
            movwf TMR0       ;cargamos el TMR0

intervalo   btfss INTCON,2   ;rebosamiento del TMRO?
            goto  intervalo  ;todavía no
            decfsz num50msec,F ;decrementa contador intervalos
            goto delay_var   ;repite el intervalo de 50 ms

#endasm

} //delay50msec

/*************************************************************************
*****/
/**
*       Función para leer el convertidor A/D
*/

uns8 a2d ( void )
{

        //configuramos adcon0=Fosc/8 [7-6] , canal0 [5-3], a/d on [0]
        ADCON0 = 0b01011001;
```

```
        //esperamos para la adquisicion 100 usg
        delay50msec(1);

        //empezamos la conversión
        ADCON0 |= 0b00000100;

        //esperamos hasta que se realice la conversión
        while (ADCON0 & 0b00000100);

        return ADRESH;

} //a2d

/*************************************************************************
*****/
/**
*       Función que distingue colores
*/

uns8 color( uns8 rojo, uns8 verde, uns8 azul )
{


        if ( ( rojo <= 12 ) && ( verde <= 12 ) && ( azul <= 12 ) )

                return (result_blanco);

        else if ( ( rojo >= 27 ) && ( verde >= 27 ) && ( azul >= 27 ) )

                return (result_negro);
```

```
        else if ( ( rojo <= 15 ) && ( verde >= 12 ) && ( azul >= 12 ) )
                return (result_rojo);
        else if ( ( rojo >= 12 ) && ( verde <= 17 ) && ( azul >= 12 ) )
                return (result_verde);
        else if ( ( rojo >= 12 ) && ( verde >= 12 ) && ( azul <= 15 ) )
                return (result_azul);
        else if ( ( rojo <= 15 ) && ( verde <= 15 ) && ( azul >= 12 ) )
                return (result_amarillo);
        else if ( ( rojo <= 15 ) && ( verde >= 12 ) && ( azul <= 15 ) )
                return (result_violeta);
        else if ( ( rojo >= 12 ) && ( verde <= 15 ) && ( azul <= 15 ) )
                return (result_celeste);
        else
                return (result_error);
} // color
```

```
/**************************************************************************
*****/
/**
*       Función para elegir el mensaje
*/


void mensaje ( uns8 num_mensaje )
{
        uns8 indice = 0;  //lo inicializamos a 0
        while ( indice <  (num_mensaje -1) )
        {
                A0 = 1;                 //reproducimos la direccion sin
sonido
                delay50msec ( 1 );
                playl = 0;
                delay50msec ( 1 );
                playl = 1;
                delay50msec ( 1 );
                A0 = 0;
                indice = indice + 1;
        }

} //mensaje
```

```
/*************************************************************************
*****/
/**
*       Función principal
*/

void main ( void )
{

uns8 valor_digital;
uns8 estado; //valor que  servirá para bucle switch
uns8 valor_rojo;
uns8 valor_verde;
uns8 valor_azul;
uns8 valor_color;

        //inicializamos los puertos

        PORTB = 0b00000000;
        TRISB = 0b00000000;

        //RB7 O led
        //RB6 O led
        //RB5 O led
        //RB4 O led
        //RB3 O led
        //RB2 O led
        //RB1 O led
        //RB0 O led

        PORTC = 0b00000000;
```

```
TRISC = 0b00000000;

//RC7 O playe
//RC6 O playl
//RC5 O nada
//RC4 O nada
//RC3 O nada
//RC2 O led rojo
//RC1 O led azul
//RC0 O led verde


PORTA = 0b00000000;
TRISA = 0b00111111;

//RA5 O nada
//RA4 I captura
//RA3 O analógico
//RA2 O nada
//RA1 O nada
//RA0 O nada

PORTB = 0b00000000;
PORTC = 0b00000000;

//configuramos el registro option con un preescaler para tmro
de 256

OPTION = 0b00000111;
```

```
//todas las señales de entrada a/d

ADCON1 = 0b00000000;

valor_digital = 0;
estado = 0;
A7 = 1;        //indicamos al lisd que queremos poner el modo de operaciones
A6 = 1;
A5 = 0;
A4 = 0;
A3 = 0;
A2 = 0;
A1 = 0;
A0 = 0;
PORTC = 0b11111000;
while (1)

{
        if ( captura == 0 )
        {
                led_rojo = 0;
                led_verde = 0;
                led_azul = 1;
                delay50msec(10);
                valor_digital = a2d();
                led_rojo = 0;
                led_verde = 0;
                led_azul = 0;
                valor_azul = valor_digital;
```

```
                led_rojo = 0;
                led_verde = 1;
                led_azul = 0;
                delay50msec(10);
                valor_digital = a2d();
                led_rojo = 0;
                led_verde = 0;
                led_azul = 0;
                valor_verde = valor_digital;

                led_rojo = 1;
                led_verde = 0;
                led_azul = 0;
                delay50msec(10);
                valor_digital = a2d();
                led_rojo = 0;
                led_verde = 0;
                led_azul = 0;
                valor_rojo = valor_digital;
                delay50msec(10);

                valor_color = color(valor_rojo, valor_verde,
                valor_azul);
                A4 = 1; //inicializamos las direcciones de
memoria del isd para empezar en la 0
                mensaje ( valor_color );
                playe = 0;
                delay50msec ( 1 );
                playe = 1;
                delay50msec ( 20 );

        }
```

```
                else
                {
                        A4 = 0;    //para resetear las direcciones de
memoria
                }

        } //while


} // main
```

CONCLUSIONES

12. CONCLUSIONES

El indicador trata de reducir los diferentes colores que existen en la realidad a los ocho que discrimina este dispositivo. Esto conlleva una serie de problemas; por ejemplo, un objeto de color muy claro, el indicador reconocerá el color blanco. Si, por el contrario, probamos con un objeto de color oscuro, lo verá como negro. Además, el brillo de los objetos confunde la identificación del color; es el caso del negro metalizado, el cual refleja más luz que el propio negro, por lo que el indicador reconocerá cualquier otro color.

EL microcontrolador utilizado (16F876) lleva incorporado el conversor A/D, lo que reduce las dimensiones de la placa al no necesitar componentes externos. Otra característica es la gran cantidad de pines de entrada y salida que posee en un encapsulado de pequeño tamaño, lo que le confiere una gran versatilidad.

El dispositivo reproductor de sonido, ISD1420, junto con el microcontrolador, se puede destinar a una gran cantidad de aplicaciones; por ejemplo, en juguetes, alarmas, contestadores automáticos... También permite reproducir uno o varios mensajes en distintos modos (modo continuo, modo selectivo, etc.) y grabarlos rápidamente. Sin embargo, al ir de un mensaje a otro, el ISD1420 emite un chasquido inevitablemente, como indica el fabricante en las hojas de características.

Este proyecto ha mostrado los fundamentos físicos del color. Además, se ha probado que, conociendo y estudiando las propiedades físicas de distintos materiales (electromagnetismo, conductividad eléctrica, temperatura, etc.), junto con el uso de componentes electrónicos simples, podemos construir aparatos de medida de dichas propiedades. Estos aparatos sirven en campos como la medicina, la tecnología, la didáctica y, en nuestro caso, la ayuda a discapacitados.

BIBLIOGRAFÍA

13. BIBLIOGRAFÍA

LIBROS

Título **Electrónica digital y microprocesadores / Eduardo Santamaría**
Publicación Madrid Universidad Pontificia de Comillas, 1993

Título **Electrónica digital y microprogramable / Antonio J. Gil Padilla, Fernando Remiro Domínguez, Luis Cuesta García**
Publicación MCGraw-Hill, 1997

Título **Microcontroladores PIC: La solución en un chip / José Mª Angulo Usategui, Eugenio Martín Cuenca, Ignacio Angulo Martínez**
Publicación Paraninfo, 2001

Título **Microcontroladores PIC: Diseño práctico de aplicaciones / José Mª Angulo Usategui, Ignacio Angulo Martínez**
Publicación McGraw-Hill, 2001

PÁGINAS WEB

http://www.microchip.com

http://ic-prog.net

http://isd.com

ANEXO

ESQUEMAS

HOJA DE CARACTERÍSTICAS DE COMPONENTES