UNIVERSIDAD MICHOACANA DE SAN NICOLÁS DE HIDALGO

FACULTAD DE INGENIERIA ELECTRICA

DISEÑO Y CONSTRUCCIÓN DEL SISTEMA DE CONTROL Y MONITOREO DE UNA ESCALADORA ELECTROMECÁNICA

TESIS

QUE PARA OBTENER EL TITULO DE:
INGENIERO ELECTRICISTA

PRESENTA
CARLOS AURELIO CAMPOS ALVARADO

ASESOR DE TESIS
ING. FÉLIX JIMÉNEZ PÉREZ

ABRIL DEL 2008.

Agradecimientos

A mi padre **Humberto Campos Oregón** que con su apoyo logre concluir este y otros trabajos.

A mi madre **Gloria Alvarado González** por su dedicación a la familia, ya que sin su esfuerzo no habría podido llegar hasta aquí.

A mis hermanos **Humberto**, **Aarón** y **Lidice** que aunque estén lejos siempre hay comunicación con ustedes.

A mis amigos **Fernando** y **Andrés** que a lo largo de este camino siempre encontramos motivos para festejar.

A mi asesor el **Ing. Félix Jiménez Pérez** por el tiempo, la paciencia y comprensión durante la elaboración de este trabajo.

A los laboratoristas **Ing. Israel Luna Reyes** e **Ing. Carlos Alberto Salas Mier** por aportar buena vibra en el laboratorio.

A mi sensei de Judo **Gustavo Alonso Guerrero Rascón** y a mis compañeros del dojo por hacer de los entrenamientos algo distinto y memorable.

Y a todas las personas que estuvieron involucradas directa o indirectamente durante la formación de mi persona como ingeniero ya que sin las charlas, consejos, regaños e interminable número de anécdotas he aprendido más allá de lo académico.

Dedicatoria

A Dios que me ha encaminado a seguir adelante en los momentos difíciles.

A mis padres que con su ejemplo, me han enseñado que solo con mucho esfuerzo se pueden cosechar logros y me han animado a terminar cada uno de los proyectos.

A los amigos que siempre han estado ahí, y a los nuevos que encontramos necesario un momento de reunión.

Resumen del Trabajo

Se presenta el diseño y construcción del sistema de control y monitoreo de una escaladora electromecánica se presento debido a la falta de cuidado en la instalación de la misma en el gimnasio de CU, el equipo no funcionó.

Al analizar los componentes de la escaladora y resolver que solo el sistema de control resulto dañado, fue factible desarrollar un nuevo sistema de control aprovechando el material disponible en el laboratorio.

Se desarrolló software para el microcontrolador 16F877A con el objetivo de además de hacer funcionar la escaladora, se leyeran datos en una memoria externa y también tuviera una interfaz grafica con una pantalla grafica LCD. La combinación de la memoria y la pantalla grafica LCD con el microcontrolador dio como resultando un control gráfico vistoso y sencillo de operar, siempre respetando el diseño original de la escaladora.

Contenido

Agradecimientos ii

Dedicatoria iii

Resumen del trabajo iv

Contenido v

Lista de Figuras viii

Lista de Tablas xii

Lista de Símbolos y Abreviaciones xiii

Capítulo 1. Introducción 1

1.1. Antecedentes ……..... 1

1.2. Objetivo ……..... 2

1.3. Justificación ……. 2

1.4. Metodología ……. 3

1.5. Descripción de los capítulos 3

Capítulo 2. La Escaladora 4

2.1. Acerca de la empresa Stairmaster manufacturadora de la escaladora … 4

2.2. Descripcion general ……... 4

2.2.1. El sistema de pedales …… 5

2.2.2. El mecanismo de poleas y cadenas ……... 7

2.2.3. El alternador como freno dinámico …….. 8

2.2.4. La fuente de la escaladora …… 10

2.2.5. Diagrama de bloques del sistema eléctrico ……. 10

2.3. Método de control de tensión 11

2.3.1. El freno dinámico ……..... 12

2.3.2. Control de tensión ……..... 12

2.3.3. Conexión de la resistencia ……... 13

2.4. Estimación de parámetros …….. 15

2.4.1. El sensor de velocidad …….. 15

Capítulo 3. El Sistema de Control Implementado 16

3.1. Diferencias entre el sistema de control original y el sistema de control implementado …. 16

3.2. El microcontrolador utilizado …… 17

3.2.1. Los puertos paralelos …… 18

3.2.2. El puerto IIC ……. 18

3.2.3 El modulo CCP en modo PWM …… 19

3.2.4. El modulo CCP en modo IC ……. 20

3.3. La memoria EEPROM externa ……. 20

3.3.1. La memoria utilizada …… 20

3.3.2. Características del bus …….. 22

3.3.3. Byte de control ……. 24

3.3.4. Lectura de memoria …….. 26

3.3.5. Organización de la memoria ……. 27

3.4. La pantalla grafica LCD …… 34

3.4.1. Características de la pantalla utilizada ……. 34

3.4.2. Rutinas implementadas …… 38

3.4.3. Conexión de la pantalla grafica LCD al microcontrolador …….. 51

3.5. Teclado de la escaladora …… 55

3.5.1. Descripción del teclado …… 56

3.5.2. Funciones del teclado ……... 56

3.5.3. Conexión del teclado al microcontrolador ……... 57

3.6. Control de tensión de la escaladora ……... 58

3.6.1. El modo PWM del microcontrolador ……... 58

3.6.2. Rango de operación …….. 62

3.6.3. Ecuación de control de PWM vs Tensión …… 62

3.7. Medición de velocidad de intensidad de ejercicio ……. 64

3.7.1. El sensor de velocidad …….. 64

3.7.2. Caracterización del sensor …… 64

3.7.3. Conexión del sensor al microcontrolador ……. 66
3.7.4. Nivel de voltaje ……. 67
3.7.5. Determinación de los parámetros de medición de velocidad vs frecuencia ……... 67
3.8. Las fuentes de voltaje …… 68
3.8.1. La fuente de 5 V ……... 68
3.8.2. La fuente de -13 V …… 69
3.8.3. La fuente de 750 V de CA …… 70
3.8.4. Programación de la memoria EEPROM ……... 71
3.9. Circuitos impresos elaborados ……... 72

Capítulo 4. Pruebas de Operación 76
4.1. Calibración …… 78
4.2. Manual de operación ……. 81

Capítulo 5. Conclusiones y Trabajos Futuros 84
5.1. Conclusiones ……. 84
5.2. Trabajos Futuros …… 85

Apéndice. Listado del Programa Principal 87

Bibliografía 105

Lista de Figuras

2. 1 La escaladora STAIRMASTER 4200PT…… 5

2. 2 Diagrama de bloques del sistema original…….. 6

2. 3 Tablero original de la escaladora STAIRMASTER 4200PT……. 6

2. 4 Los pedales de la escaladora STAIRMASTER 4200PT…… 7

2. 5 Sistema de cadenas y poleas……... 7

2. 6 Diagrama eléctrico de conexión entre el alternador y el rectificador……. 8

2. 7 Foto de un puente rectificador (Nanodiodo)…….. 8

2. 8 Alternador de la escaladora STAIRMASTER 4200PT……. 9

2. 9 Diagrama eléctrico del alternador…….. 10

2. 10 Fuente de la escaladora……. 11

2. 11 Diagrama eléctrico de la escaladora……. 11

2. 12 Control original (vista inferior)…… 12

2. 13 Control original (vista superior)……... 13

2. 14 Bornes del alternador para realizar conexiones…… 13

2. 15 Resistencia cerámica del alternador……. 14

2. 16 Diagrama de conexión de la resistencia con el alternador……... 14

3. 1 Diagrama a bloques del sistema de control implementado…… 16

3. 2 Diagrama de terminales del microcontrolador PIC 16F877A…… 17

3. 3 Variación del ciclo de trabajo……. 19

3. 4 Memoria EEPROM…… 20

3. 5 Diagrama a bloques de la memoria EEPROM……... 21

3. 6 Diagrama de terminales de la memoria EEPROM con encapsulado DIP…. 21

3. 7 Secuencia en la transferencia de datos en el bus…… 23

3. 8 Tiempo de reconocimiento de datos…….. 24

3. 9 Descripción de los bits del byte de control……. 25
3. 10 Representación del byte de control y dirección del primer byte de datos… 25
3. 11 Diagrama de la lectura de la dirección actual……... 26
3. 12 Diagrama de la lectura secuencial…… 27
3. 13 Imágenes Generadas en BMP, compiladas y guardadas en la memoria EEPROM……... 28
3. 14 Diagrama de conexión del microcontrolador con la memoria EEPROM… 29
3. 15 Pantalla de fondo impresa durante la realización de las rutinas……... 30
3. 16 Perfil lineal……... 30
3. 17 Perfil escalón…… 31
3. 18 Perfil rampa…….. 31
3. 19 Perfil diente de sierra…… 31
3. 20 Pantalla de escudos editada en el programa FASTLCD…….. 32
3. 21 Tabla de bytes de la imagen de los escudos de la universidad y de la facultad…….. 33
3. 22 Programa LCDCompile…… 33
3. 23 Perfiles a partir de la dirección 7000_h mostradas en el programa XVI32… 34
3. 24 Pantalla grafica LCD AND 1741……. 35
3. 25 Distribución de píxeles en la pantalla grafica LCD……. 35
3. 26 Posición de los píxeles dentro del grupo…….. 36
3. 27 Distribución de los grupos de píxeles…….. 36
3. 28 Diagrama a bloques de conexiones internas y externas de la pantalla……. 36
3. 29 Luz fluorescente……... 38
3. 30 Distribución de los bits de estado en las líneas de datos de la pantalla grafica LCD…….. 38
3. 31 Diagramas de flujo del funcionamiento de la pantalla grafica LCD para introducir datos…….. 39
3. 32 Representación de los píxeles agrupados en grupos, en la parte izquierda de la pantalla…….. 42
3. 33 Representación de los píxeles agrupados en grupos, en la parte derecha de la pantalla……... 42

3. 34 Nueva distribución de los píxeles en eje x y eje y…… 44
3. 35 Representación del orden de los píxeles en un grupo de ellos……. 44
3. 36 Representación de los píxeles en bytes…… 51
3. 37 Diagrama esquemático de las conexiones de la pantalla grafica LCD……. 52
3. 38 Conexión de las líneas de datos del microcontrolador a la pantalla grafica LCD……... 53
3. 39 Conexión de las líneas de control del microcontrolador a la pantalla grafica LCD……... 53
3. 40 Pruebas con la pantalla grafica LCD…… 54
3. 41 Conexión de la pantalla grafica LCD a las fuentes de alimentación…… 54
3. 42 Conexión del selector de voltaje al circuito backlite…… 55
3. 43 Teclado original de la escaladora……. 55
3. 44 Diagrama de conexión de una tecla…….. 56
3. 45 Diagrama de conexión del teclado con el microcontrolador…… 57
3. 46 Diagrama del circuito de excitación del campo……... 59
3. 47 Periodo y ciclo de trabajo……. 59
3. 48 Pasos de resistencia mecánica vs % PWM……... 63
3. 49 Grafica voltaje amplificado vs % PWM…….. 64
3. 50 Terminales en el costado del alternador……... 66
3. 51 Diagrama de conexión del microcontrolador con el tacogenerador……. 66
3. 52 Grafica frecuencia contra % velocidad…… 68
3. 53 Diagrama de la fuente de 5 V……... 69
3. 54 Conexión de un regulador positivo para lograr un voltaje negativo…… 69
3. 55 Diagrama de conexiones de la fuente de -13 V con la de 5 V……..…… 70
3. 56 Backlite montado en el tablero……. 70
3. 57 Programa IC-PROG……. 71
3. 58 Programador PROPIC2…… 72
3. 59 Diseños del circuito impreso para el control…… 72
3. 60 Circuito impreso con componentes para el control…….. 73
3. 61 Diseños del circuito impreso para el control de potencia……. 73
3. 62 Circuito impreso con componentes para el control de potencia……... 74

3. 63 Colocación de los circuitos en el tablero…….. 74
3. 64 Pantalla original y colocación de la pantalla gráfica en el tablero……... 75
3. 65 Pantalla grafica LCD y control instalados…… 75

4. 1 Pantalla de presentación……. 76
4. 2 Pantalla autores…….. 76
4. 3 Pantalla de selección de rutina……... 77
4. 4 Pantalla de selección de intensidad…… 77
4. 5 Pantalla de selección de tiempo de ejercicio……. 78
4. 6 Uso de la escaladora……..... 79
4. 7 Pantalla de ejercicio con rutina Fat Burner…… 79
4. 8 Pantalla de ejercicio con rutina Manual……. 80
4. 9 Pantalla de ejercicio con rutina Aerobic Training…….. 80
4. 10 Pantalla de ejercicio con rutina Steady Pace…… 81
4. 11 Pantalla UMSNH y FIE…… 81
4. 12 Pantalla autores……. 82
4. 13 Pantalla de selección de perfil de trabajo……. 82
4. 14 Pantalla de selección de perfil de trabajo (Nivel 5)…….. 83
4. 15 Pantalla de selección de tiempo (5 minutos)…… 83

Lista de Tablas

3. 1 Asignación de los puertos utilizados del microcontrolador……... 18
3. 2 Descripción de terminales de la memoria EEPROM……. 22
3. 3 Mapa de la memoria 24FC512……... 29
3. 4 Temperatura contra voltaje de trabajo…… 37
3. 5 Orden resultante del residuo después de restarle -7 y multiplicarlo por -1… 44
3. 6 Configuración sugerida de la memoria RAM de la pantalla grafica LCD[6]……. 50
3. 7 Tabla de conexión entre el microcontrolador y los demás dispositivos……. 52
3. 8 Ejemplo de frecuencias pwm y resoluciones a 20 MHz……. 62
3. 9 Nivel PWM y voltaje de operación…… 63
3. 10 Datos de Frecuencia y velocidad……. 67

Lista de Símbolos y Abreviaturas

W	watts
V	volts
Ω	ohms
K	kilo
M	mega
Hz	hertz
μ	micro
s	segundos
Tpwm	es el periodo de la onda PWM
D	es el tiempo que dura el ciclo de trabajo de la onda PWM
Tosc	Periodo de oscilación del cristal
Fpwm	Es la frecuencia de la onda PWM

Capítulo 1

Introducción

1.1. Antecedentes

Este trabajo se realizó en base a que se cuenta con una escaladora, la cual tiene dañado el sistema de control. Por lo cual se presentó la oportunidad de realizar el diseño y construcción del sistema de control y monitoreo de dicha escaladora electromecánica. El diseño del trabajo se encuentra dividido en 4 sistemas básicos:

1.- El sistema de control
2.- El sistema de control de tensión de la escaladora
3.- El sistema mecánico
4.- La interfaz con el usuario

Este proyecto tiene como objetivo principal no solo diseñar un sistema, si no mejorar el desempeño del sistema original optimizando las funciones y la interfaz con el usuario.

El sistema de control propuesto está basado en un microcontrolador. El sistema mecánico se analizó y fue viable utilizar el original, ya que se encuentra en perfecto estado debido a que no resultó afectado al dañarse el sistema de control, y se ajustaba perfectamente a las necesidades. Adicionalmente de diseño el control de potencia. La interfaz grafica fue pensada considerando que la original quedó inservible al quemarse el sistema de control, además que se mejoró para que fuera aun mas fácil de usar teniendo en cuenta que el laboratorio cuenta con algunas pantallas graficas LCD (pantalla de cristal liquido), así que, fue una buena oportunidad de utilizarlas.

El uso por primera vez de una pantalla grafica en el laboratorio, nos compromete a desarrollar software fácil de entender y utilizar, para que otras personas puedan hacer uso del mismo. Al final del proyecto se dejarán rutinas lo suficientemente comentadas a manera de funciones.

Esta misma situación se tiene al utilizar una memoria serial EEPROM (memoria de solo lectura, eléctricamente programable y borrable).

1.2. Objetivo

Los objetivos que persigue este trabajo son:

1.- Realizar el diseño y construcción de un sistema de control y monitoreo para la escaladora marca Stairmaster modelo 4200PT, usando un microcontrolador para sustituir el control dañado.

2.- Diseñar una interfaz grafica que sea amigable con el usuario. Todo esto basado en un microcontrolador.

3.- Implementar un sistema para configurar el tiempo de ejecución del ejercicio y el nivel de tensión en los pedales con la posibilidad de variar la intensidad durante la ejecución al presionar la tecla de incremento o decremento en el teclado de la escaladora.

1.3. Justificación

El presente trabajo surge de la necesidad de reparar una escaladora ubicada en el gimnasio de CU que fue adquirida en el extranjero, debido a que el costo de reparación del sistema de control supera el del sistema completo, considerando la inversión que se requiere para sustituir la parte dañada, el envio y los gastos de importación. Esto permitió pensar como alternativa de solución el desarrollo y construcción de un sistema basado en microcontrolador y agregar una interfaz gráfica con el usuario.

El diagnóstico que se realizó del control original a partir de los comentarios vertidos por la persona que instaló la escaladora fue que sufrió daños a consecuencia de la alimentación con sobrevoltaje, debido a que la escaladora posee una fuente de corriente directa pero debido a una omisión conectaron la escaladora a una fuente trifásica de 220 volts(V) de corriente alterna(CA). El daño abarca el sistema de control de la escaladora a tal grado que no fue posible realizar su reparación, por este motivo se procedio a diseñar y desarrollar un sistema de control “nuevo” debido a que en caso contrario la escaladora quedaría inservible**.** Esto nos obliga a considerar el costo del servicio en la adquisición de

cualquier equipo, esta consideración se debe de realizar aun más en equipos de importación.

Las características generales del sistema de control y monitoreo se tomaron de sus especificaciones generales debido a que no se tiene una escaladora de referencia. De estas características se incluye la medición de velocidad de escalamiento, la cual sirve de referencia al usuario al momento de estar ejercitándose para conocer el ritmo de ejercicio.

1.4. Metodología

La metodología que se siguió para resolver el proyecto fue en primer lugar recopilación de información de los dispositivos a utilizar. En segundo lugar se interpretó esta información y se usó para hacer la intercomunicación entre los dispositivos en un montaje en protoboard. En tercer lugar se realizaron pruebas para que cumpliera con un óptimo desempeño y se hicieron las correcciones pertinentes.

1.5. Contenido de la tesis

En el capitulo 1 se describe brevemente este trabajo, se presentan los antecedentes y los objetivos del trabajo.

En el capitulo 2 se describen las especificaciones del sistema de control original de la escaladora, y cada uno de los elementos que la conforman. En que condiciones se recibió la escaladora, las ventajas del uso de la escaladora.

En el capitulo 3 se muestra el nuevo diseño del sistema de control, como se configuraron los diferentes dispositivos y software que se empleo.

En el capitulo 4 se muestra la escaladora en uso con el nuevo sistema de control.

En el capitulo 5 se describe la experiencia de realizar este trabajo.

Capítulo 2

La Escaladora

En este capitulo se presenta un panorama general tanto del funcionamiento como de la operación de la escaladora, esto permite diseñar el sistema de monitoreo y control de la misma. Se analizan las ventajas de esta escaladora, los componentes principales y el daño que sufrió.

2.1. Antecedentes de la empresa Stairmaster manufacturadora de la escaladora

La empresa Stairmaster es una leyenda en los mejores gimnasios del mundo, desde la presentación en 1983 de su primera escaladora llamada StepMill. Stairmaster ha establecido el estándar de los productos de ejercicios de alta calidad y rendimiento gracias a su combinación de visión, innovación, investigación y calidad sin igual con una dedicación hacia los resultados. Adquirida en 2002 por Nautilus permanece a la vanguardia en equipos de ejercicio. [1]

2.2. Descripción general

El uso de escaladoras en la actualidad es muy difundido debido a que en 1996 un estudio de Surgeon-General sobre actividad física y salud reveló que 20 minutos en una escaladora quema tantas calorías como 30 minutos de otros tipos de ejercicios cardiovasculares.

La escaladora Stairmaster 4200PT ofrece un sistema independiente al ejercitarse, la forma patentada de los pedales y su sistema electrónico de frenado Stairmaster, la hacen muy adecuada para el desarrollo del ejercicio [1]. La escaladora se muestra en la Figura 2.1 [2].

De fábrica esta escaladora incluye:

1.- Cuatro programas de ejercicios, cada uno con 20 niveles de intensidad.

2.- Indicadores numéricos que permiten al usuario leer cuantos escalones ha subido, a cuantos pisos equivalen estos escalones, el tiempo que lleva ejercitándose y las calorías que ha perdido.

3.- Alarma audible para comunicar al usuario que la rutina ha terminado.

4.- Botones para seleccionar el tipo de ejercicio, para reiniciar la escaladora y para aumentar y disminuir la intensidad del ejercicio durante la realización de la rutina.

5.- Rutina rápida de 15 minutos que inicia al apretar el botón Enter.

Figura 2. 1 La escaladora STAIRMASTER 4200PT.

En la Figura 2.2 se muestra en diagrama de bloques el sistema de control original de la escaladora. En la Figura 2.3 se muestra el aspecto original del tablero de monitoreo y control original de la escaladora, es importante apreciar la interfaz con el usuario.

2.2.1. El sistema de pedales

Su sistema de paso patentado, permite un correcto movimiento biometricalmente, esto quiere decir que el impacto en las articulaciones es bajo y por lo tanto no las afecta. Sus pedales tienen movimiento independiente y su geometría con barras antiderrape

permiten mantener un alto nivel de seguridad, previniendo cualquier tipo de accidente. En la Figura 2.4 se muestran estos pedales[2].

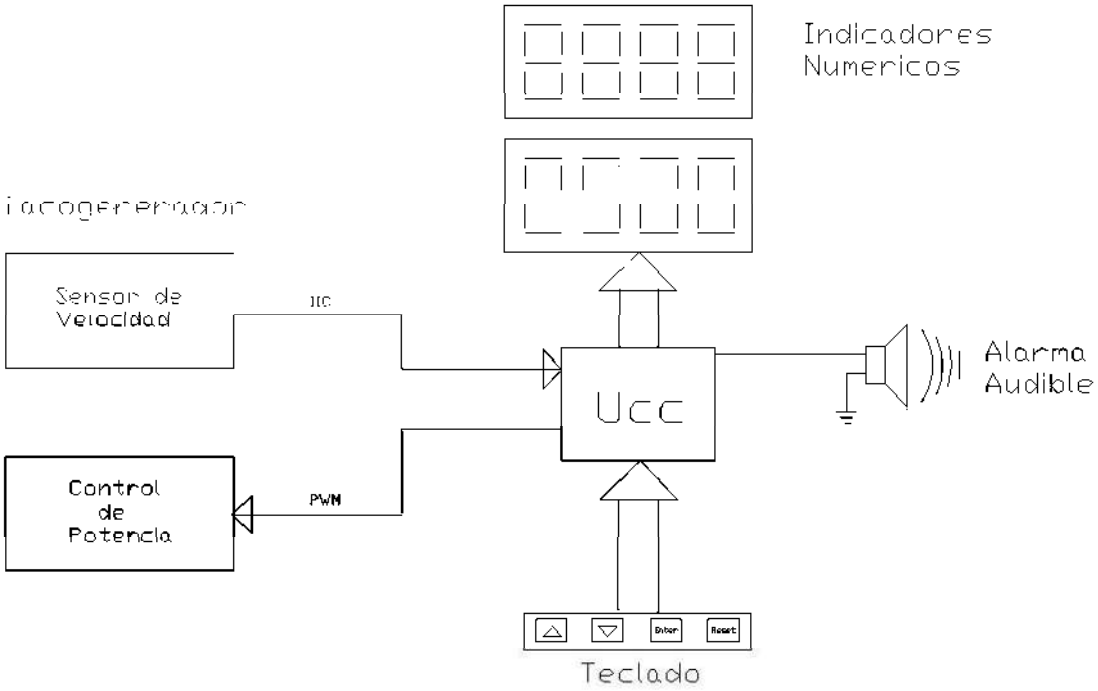

Figura 2. 2 Diagrama de bloques del sistema original.

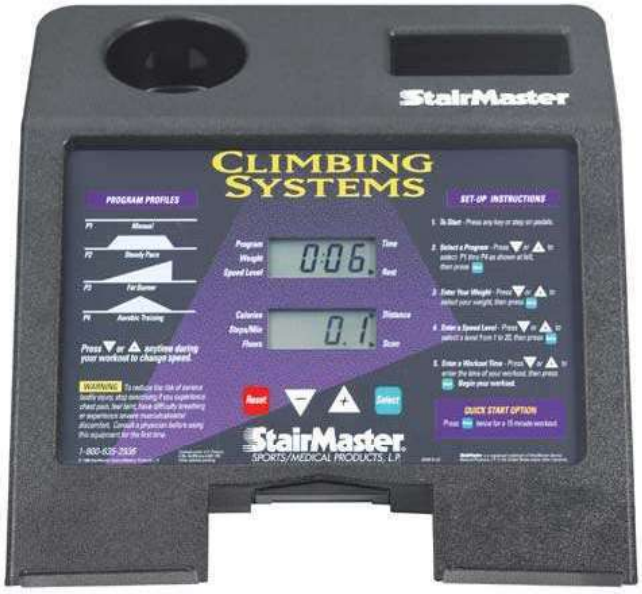

Figura 2. 3 Tablero original de la escaladora STAIRMASTER 4200PT.

Figura 2. 4 Los pedales de la escaladora STAIRMASTER 4200PT.

2.2.2. El mecanismo de cadenas y poleas

Las características de funcionamiento del sistema electrónico del alternador con el mecanismo de cadenas y poleas permiten un control preciso del descenso del pedal permitiendo un amplio rango de usuarios para ejercitarse de una manera confortable. Este mecanismo se muestra en la Figura 2.5.

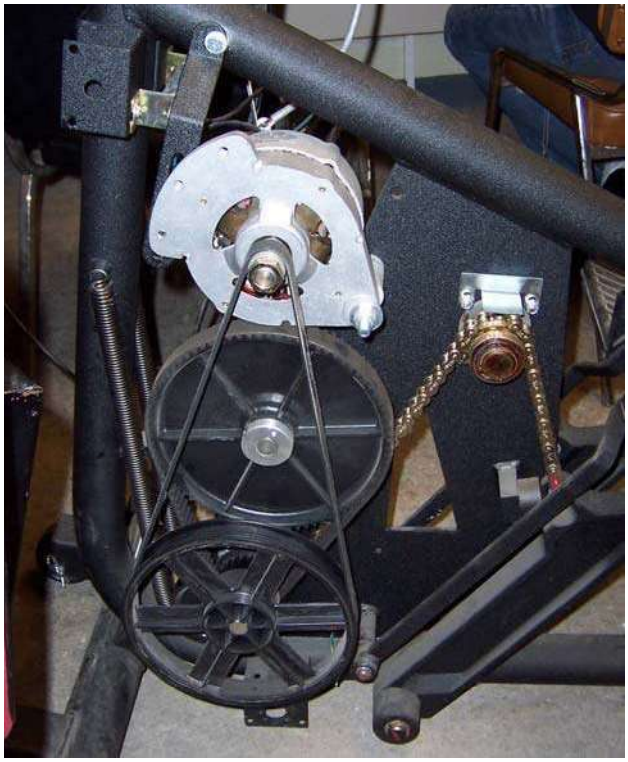

Figura 2. 5 Sistema de cadenas y poleas.

2.2.3. El alternador como freno dinámico

El control de la tensión en los pedales se debe a la acción de oposición del alternador al movimiento en los pedales, mas adelante se presentarán los detalles al respecto, a continuación se muestran los conceptos básicos.

Diferencia entre un alternador y un generador.

El alternador produce Corriente Alterna (CA). A veces puede presentar confusión al estar hablando de los alternadores porque se utilizan comúnmente en los automóviles, y todas las partes eléctricas del automóvil funcionan con CD, pero el alternador es un generador sincrónico que suministra CA de una determinada frecuencia, la que es proporcional a la velocidad de rotación y se hace la conversión a CD por medio de un puente rectificador. En la figura 2.6 se muestra el diagrama de conexión eléctrica de los devanados del alternador con el puente rectificador y en la figura 2.7 se muestra la forma física del puente rectificador.

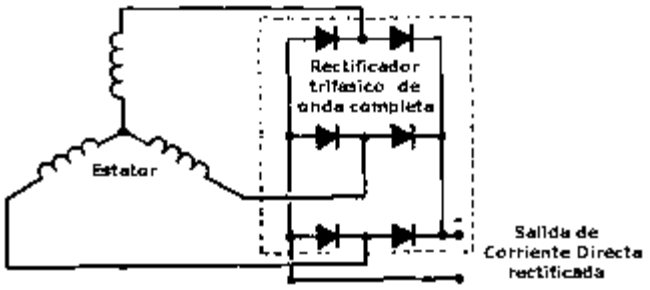

Figura 2. 6 Diagrama eléctrico de conexión entre el alternador y el rectificador.

Figura 2. 7 Foto de un puente rectificador (Nanodiodo).

Básicamente está constituido por un estator en el que se aloja el arrollamiento de armadura o inducido y por un rotor tripolar o multipolar en el que se encuentra el arrollamiento de campo o inductor. Éste se excita con corriente continua que normalmente proviene de un pequeño dínamo denominado excitatriz (en el caso de este proyecto proviene de una fuente de CD). El principio de funcionamiento es simple, el rotor produce un campo magnético, la densidad de flujo en el entrehierro tiene una distribución prácticamente sinusoidal. Al ponerse en movimiento el rotor, las líneas de fuerza cortan a los conductores ubicados en el estator generándose en ellos una fuerza electromotriz, la cual varía de la misma forma como varía la densidad de flujo en el entrehierro, es decir sinusoidalmente. El alternador es mostrado en la figura 2.8 [2].

Figura 2. 8 Alternador de la escaladora STAIRMASTER 4200PT.

La frecuencia de la onda generada dependerá de la velocidad de la máquina, cuando circula corriente por la armadura se produce otro campo magnético de reacción de armadura que en la máquina trifásica tiene las características de girar en la misma dirección y a la misma velocidad del rotor. Este campo interacciona con el campo del rotor y se produce un torque electromagnético que tiende a alinearlos.

El freno dinámico

El freno dinámico para máquinas de CA, es aquel donde se aplica entre sus bornes una tensión continua, dependiendo del valor de la tensión CD aplicada se obtendrá un freno comprendido desde un 0% (sin freno) hasta un 120% del torque nominal, es muy

importante aclarar que el uso prolongado de este freno provoca recalentamientos a las máquinas ya que la energía se concentrará en ella. En la figura 2.9 se muestra el diagrama eléctrico del alternador.

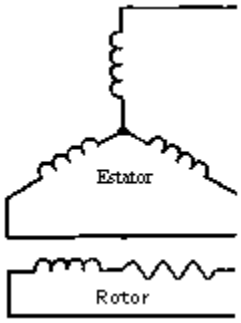

Figura 2. 9 Diagrama eléctrico del alternador.

Consumo de la energía

Como anteriormente se describió, los recalentamientos que pudieran ser producidos debido a la energía que se genera en el uso de la máquina de CA como freno dinámico pueden dañarla. Esta energía es generada por el usuario de la escaladora al tiempo de estar pedaleando lo que hace que debido al sistema de cadenas y poleas se transmita este movimiento en forma giratoria al rotor del alternador que a su vez. En las terminales del alternador se coloca una resistencia, esta resistencia es la encargada de disipar la energía sin ocasionar un desgaste de la maquina de CA.

2.2.4. La fuente de la escaladora

Esta fuente se alimenta con 127 volts(V) de CA para energizar con 12 V a la escaladora en todos sus sistemas. La fuente de la escaladora se muestra en la figura 2.10 [2].

2.2.5. Diagrama de bloques del sistema eléctrico

La figura 2.11 muestra un diagrama de bloques del sistema eléctrico de la escaladora.

Figura 2. 10 Fuente de la escaladora.

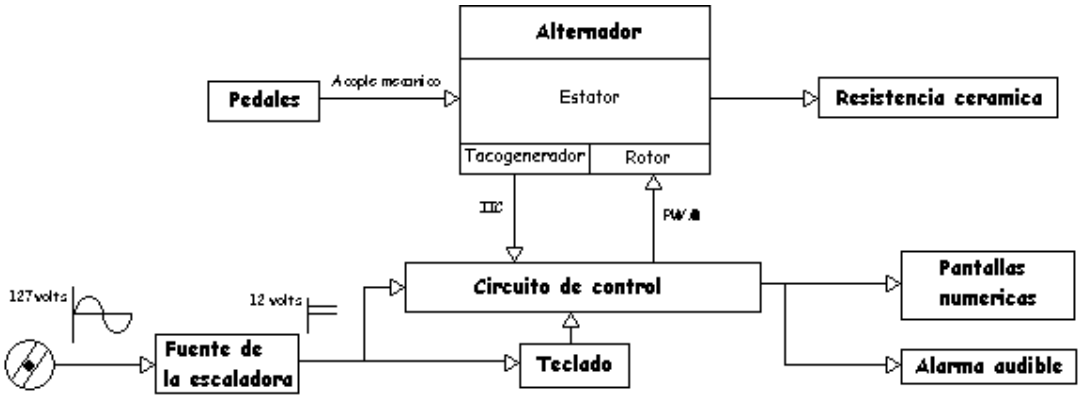

Figura 2. 11 Diagrama de bloques del sistema eléctrico de la escaladora.

2.3. Método de control de tensión

Para utilizar un alternador como freno dinámico es necesario aplicar un voltaje de corriente directa en el rotor. Entonces el control de tensión en los pedales es realizado por el microcontrolador del sistema variando la corriente en el rotor o campo.

2.3.1. El freno dinámico

Al utilizar un alternador como freno dinámico, el voltaje de salida del generador se puede controlar con la variación de la corriente del rotor, ya que en un alternador al

incrementar la corriente de campo se incrementa el voltaje en las terminales del generador para una misma velocidad del rotor.

2.3.2. Control de tensión

Para controlar la tensión en los pedales de la escaladora, el microcontrolador realiza de manera electrónica el control de la corriente del rotor. En las figuras 2.12 y 2.13 se puede observar la tarjeta de control original dañada.

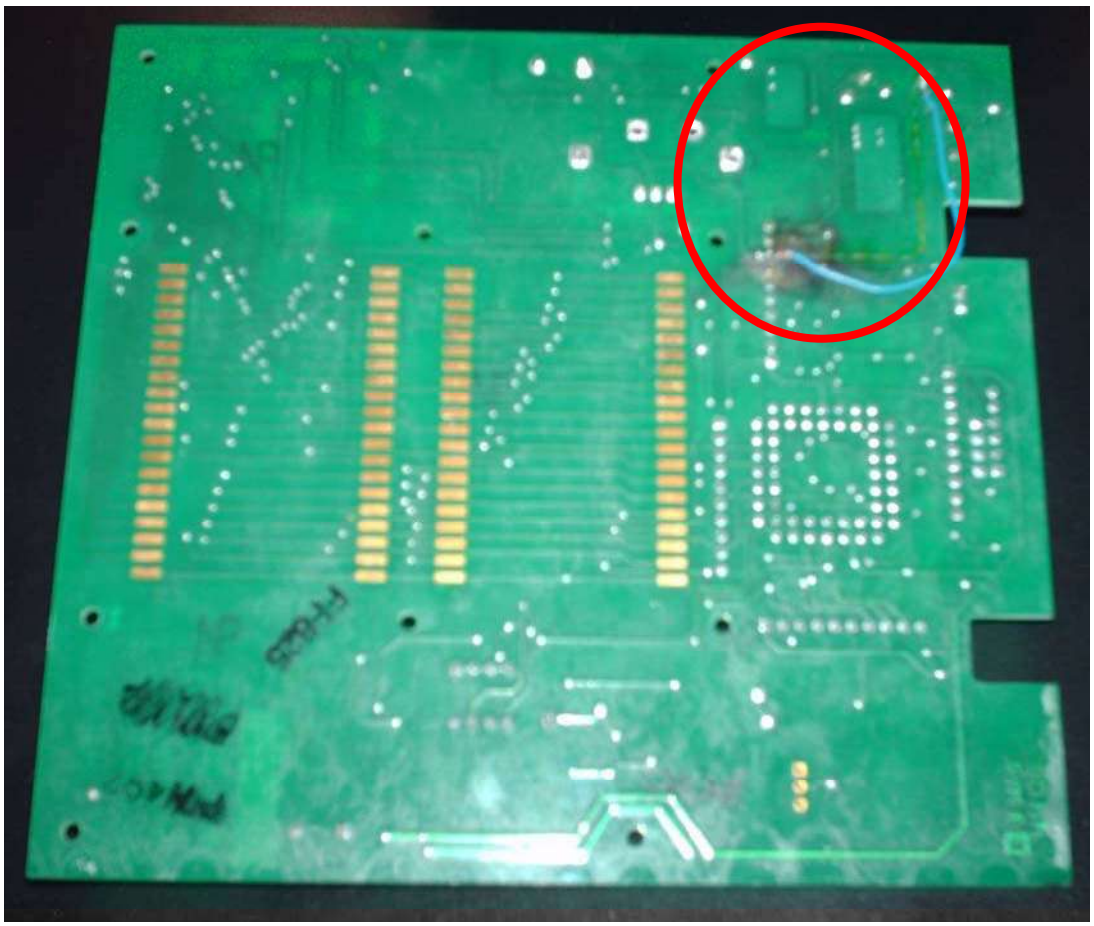

Figura 2. 12 Control original (vista inferior).

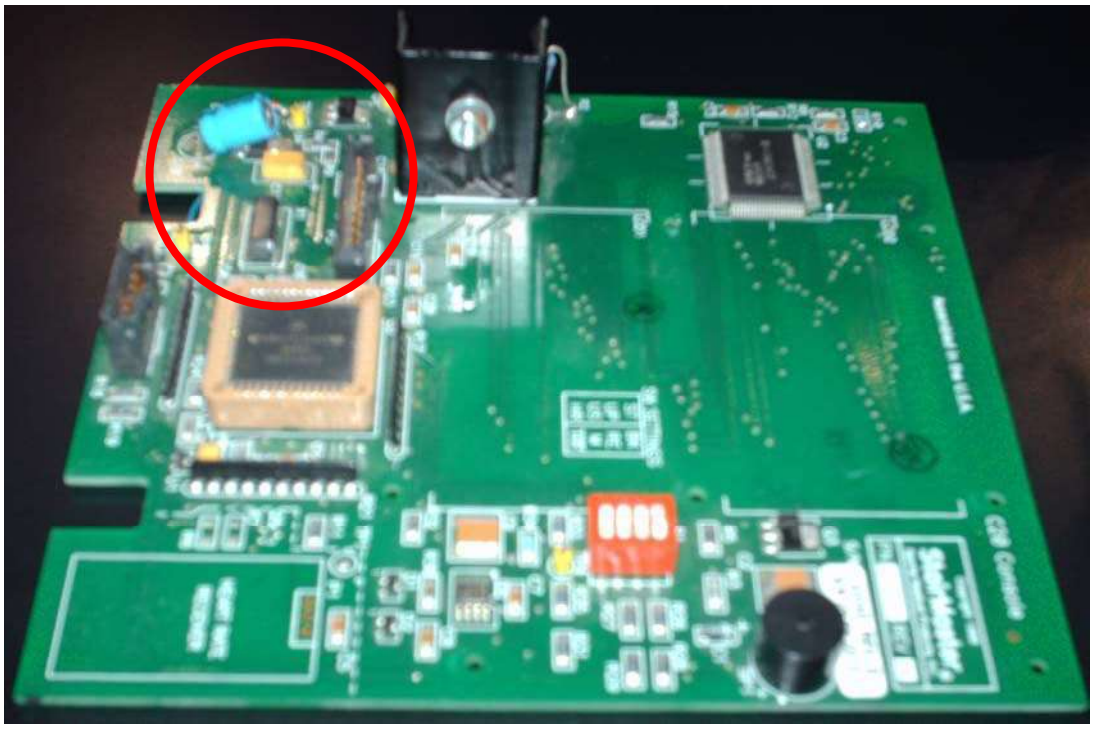

Figura 2. 13 Control original (vista superior).

2.3.3. Conexión de la resistencia

El alternador tiene bornes para permitir la conexión de la resistencia con el campo, de esta manera la energía generada por el usuario se disipa en forma de calor (hasta 100 W). La figura 2.14 muestra los bornes dentro del recuadro rojo. [2]

Figura 2. 14 Bornes del alternador para realizar conexiones.

El valor de la resistencia es constante (2.5Ω) y lo que varía es el voltaje y por lo tanto la corriente generada por el alternador debido a la acción del usuario en los pedales. La figura 2.15 muestra la resistencia cerámica que va conectada al alternador [2]. La figura 2.16 muestra el diagrama de conexión eléctrica del alternador con la resistencia.

Figura 2. 15 Resistencia cerámica del alternador.

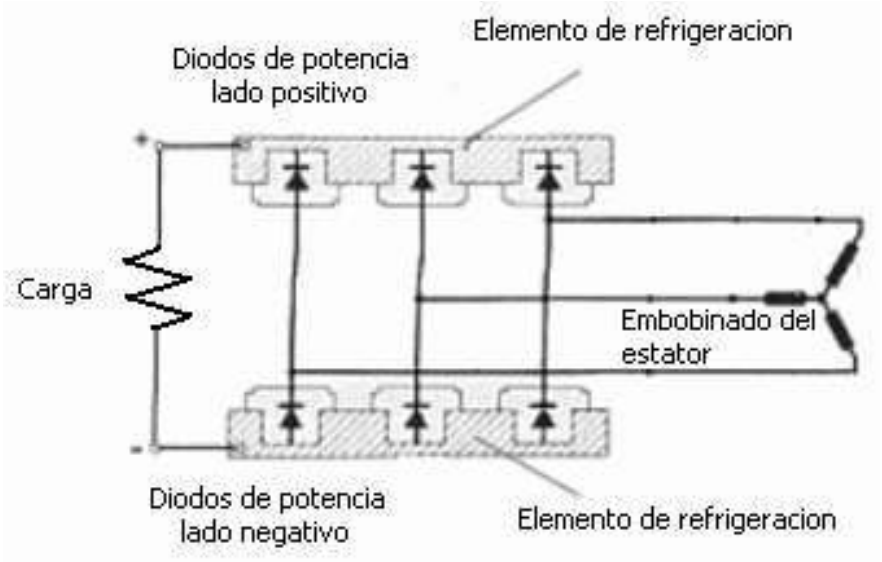

Figura 2. 16 Diagrama de conexión de la resistencia con el alternador.

2.4. Estimación de parámetros

Para la realización correcta del ejercicio es necesario que la escaladora sea adecuada a los usuarios y lo que esperan estos de ella, cuando una persona realiza ejercicio en una maquina puede conocer el tiempo, pero no puede conocer la distancia que habría recorrido de haberlo hecho en un lugar abierto, además gracias a la tecnología de los microcontroladores se puede conocer de manera estimada las calorías quemadas en la mayor parte de los aparatos electrónicos de ejercicio.

2.4.1. El sensor de velocidad

Para estimar la distancia, la cantidad de calorías quemadas así como la continuidad del ejercicio se precisa de un sensor que proporcione estos parámetros. El alternador cuenta con un encoder que permite conocer la velocidad del rotor, este sensor envía un tren de pulsos con una frecuencia y amplitud de voltaje que varia con la velocidad del rotor. El voltaje de salida se encuentra en un rango de 0 a 12 V, el cual depende de la propia velocidad del rotor y del campo. Utilizando esta señal se pueden estimar todas las variables anteriormente mencionadas.

Capítulo 3

El Sistema de Control Implementado

En este capitulo se presenta el análisis y construcción del sistema de control implementado. Se describen las características y la configuración de los dispositivos; y la construcción de los circuitos. El diseño del nuevo sistema de control, se realizó de acuerdo al diagrama mostrado en la figura 3.1.

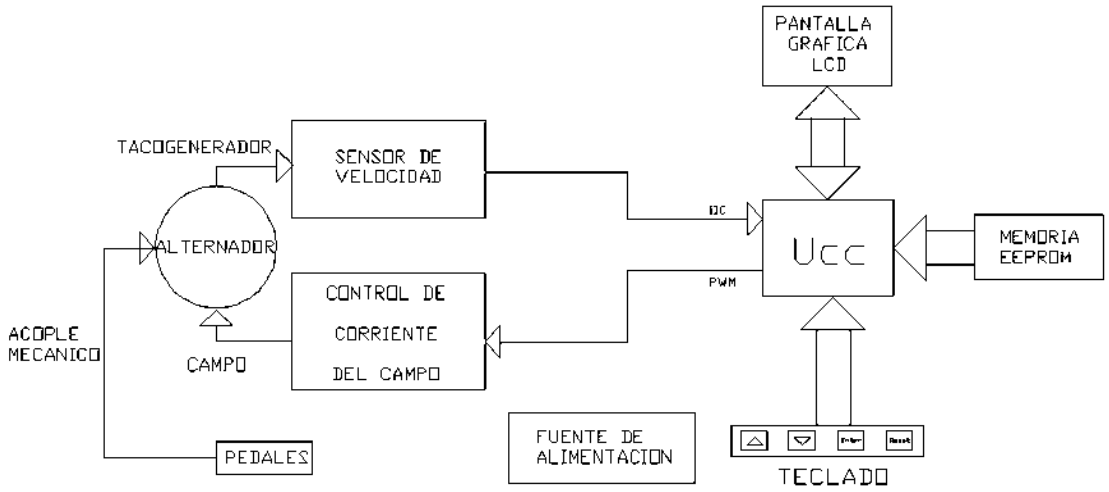

Figura 3. 1 Diagrama a bloques del sistema de control implementado

3.1. Diferencias entre el sistema de control original y el sistema de control implementado

Haciendo un análisis comparativo para notar los cambios con el sistema original, el cambio más importante radica en el hecho de que el sistema original contaba con dos pantallas numéricas digitales y el nuevo posee una pantalla grafica LCD. El proyecto no incluye la alarma audible que tenia control original, la cual daba aviso de que terminó la rutina, debido a que en nuestro diseño el control se deshabilita y se muestra una pantalla de fin de rutina, el proceso de las rutinas así como los diferentes perfiles permanecen iguales.

3.2. El microcontrolador utilizado

El diseño del sistema de control y monitoreo se basa en el microcontrolador PIC16F877A de microchip del cual es mostrado el diagrama de terminales en la figura 3.2 [3]. Este cuenta con las siguientes características principales:

- Línea de datos de 8 bits
- Arquitectura Harvard
- Tecnología RISC
- Tecnología de construcción CMOS
- Maneja 35 instrucciones de 14 bits
- El tipo de memoria de almacenamiento de programa es tipo Flash.
- El empaquetado es tipo DIP (paquete dual en-línea del ingles Dual In-line Package) de 40 terminales.
- Tiene 5 puertos paralelos de entrada/salida (denominados A, B, C, D y E).

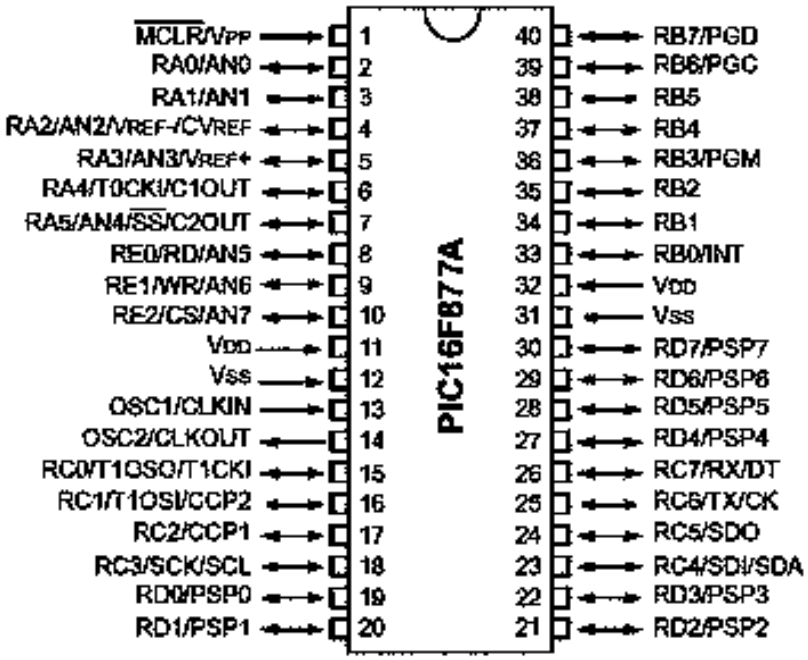

Figura 3. 2 Diagrama de terminales del microcontrolador PIC 16F877A.

3.2.1. Los puertos paralelos

En este microcontrolador cada una de las líneas de los puertos paralelos se puede configurar como de entrada o de salida de manera individual, así como configurar cada una de las líneas de los puertos dentro de la ejecución del programa.

A continuación se muestra la tabla 3.1 donde se presentan los puertos utilizados y la función que realiza dentro del sistema.

Tabla 3. 1 Asignación de los puertos utilizados del microcontrolador.

PUERTO USADO	PARTE DEL SISTEMA INTERCONECTADA
PUERTO A	
RA0, RA1 y RA2	Teclado de la escaladora
RESET	Reset del sistema
PUERTO B	
RB0, RB1, RB2 y RB3	Líneas de control de la pantalla grafica LCD
PUERTO C	
RC1	Alternador (IIC - captura de pulsos para medir velocidad)
RC2	Alternador (PWM - intensidad de corriente para el freno dinámico)
RC3 y RC4	Memoria EEPROM (data, clock)
PUERTO D	
RD0-RD7	Líneas de datos de la pantalla grafica LCD

3.2.2. El puerto IIC

El puerto IIC o I2C (pronunciación "I cuadrado C") es una interfaz de comunicación bidireccional de dos cables. Fue inventada por Phillips y el nombre se encuentra registrado. Esta es la razón por la cual otros fabricantes utilizan otro nombre para el mismo protocolo. Atmel lo llama "two wire interface" (TWI) o "interfaz de dos cables". A pesar de esto se han otorgado licencias a más de 50 compañías, encontrándonos con mas 1000 dispositivos electrónicos compatibles con I2C. Originalmente fue especificado para 100 kbits/s, e intencionalmente para el control simple de señales, esto sumado a su bajo costo, versatilidad técnica y simplicidad aseguraron su popularidad.

Este modulo se utilizó para comunicarse con una memoria serie del tipo 24FC512 de 64 Kbytes. En esta memoria se almacena la mayor parte de la información que se presenta en la pantalla grafica LCD.

3.2.3. El modulo CCP en modo PWM (modulación por ancho de pulso)

Estos módulos se denominan módulos CCP (de las siglas en ingles Capture, Compare, PWM). Este microcontrolador posee 2 módulos CCP programables que pueden operar en 3 modos ya sea como PWM de hasta 10 bits, entrada de captura de 16 bits, o como salida de comparación de 16 bits. Estos módulos comparten algunos de los recursos del microcontrolador como son el TIMER 1 y el TIMER 2. El modo PWM permite entre otras funciones realizar el control de una variable analógica utilizando una salida digital, otra de las ventajas de este módulo es que para realizar el control de una señal analógica, disipa una pequeña cantidad de potencia ya que opera en modo conmutación al variar el ciclo de trabajo de una señal y manteniendo su frecuencia constante.[4]

En la figura 3.3 se muestra como se controla el nivel de voltaje, al modificar el ciclo de trabajo de la señal. Una de las características principales de este tipo de señales es que la frecuencia permanece constante.

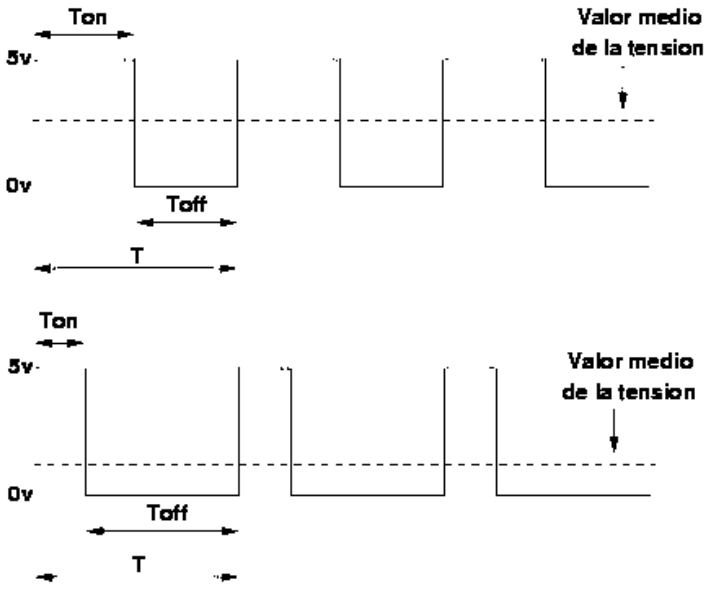

Figura 3. 3 Variación del ciclo de trabajo.

3.2.4. El modulo CCP en modo IC (Captura)

Cuando se activa el módulo CCP en modo IC, se configura el flanco que se desea que lo dispare, ya sea en subida o bajada, y cada vez que se presente dicho flanco en la terminal correspondiente se copia el valor de TMR1, de 16 bits, en la pareja de registros CCPRL y CCPRH. Cada vez que llega un flanco se tendrá en CCPR el valor en ese momento de TMR1. Si se habilita la solicitud de interrupción del CCP, se producirá además una petición de servicio para esta interrupción cada vez que el flanco configurado se presente en la terminal correspondiente, y al conocer este valor, es posible se puede determinar la frecuencia de la señal de entrada.

3.3. La memoria EEPROM externa

Se utilizó una memoria externa debido a que la cantidad de memoria de almacenamiento de datos del microcontrolador no es suficiente. La figura 3.4 muestra la memoria instalada en uno de los circuitos desarrollados.

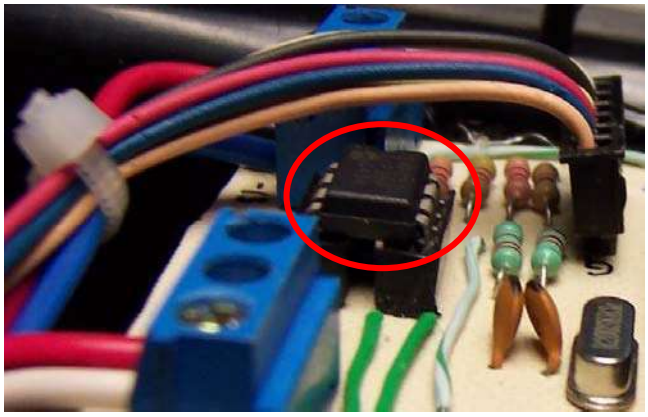

Figura 3. 4 Memoria EEPROM.

3.3.1. La memoria utilizada

Se utilizó la memoria 24FC512 la cual es fabricada por Microchip. El diagrama de bloques se muestra en la figura 3.5 [5]. El diagrama de terminales es mostrado en la figura 3.6 [5]. Sus características principales son:

- Es pequeña (8 terminales) en encapsulado tipo DIP.

- Bajo consumo de energía.
- Es confiable, utiliza la tecnología llamada PEEC con la que se consigue retención de datos de 200 años, y mas de 1’000,000 de ciclos de borrado y escritura.
- Es rápida puede operar a una velocidad en la línea de datos de hasta 1 MHz.
- Utiliza un puerto serie de tipo I^2C.
- Tiene una capacidad de almacenamiento de 512 kbits (64 Kbytes).

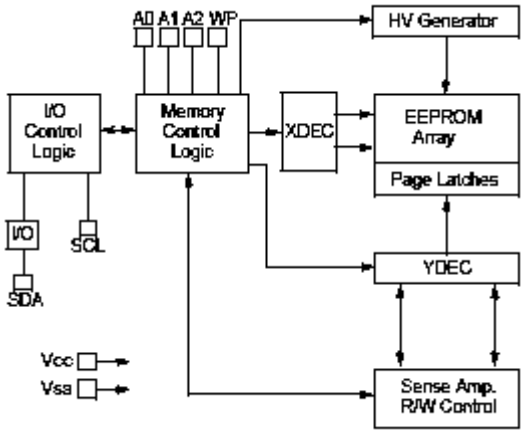

Figura 3. 5 Diagrama a bloques de la memoria EEPROM.

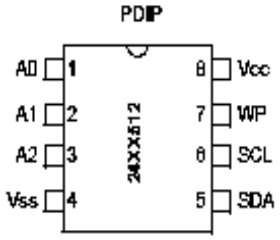

Figura 3. 6 Diagrama de terminales de la memoria EEPROM con encapsulado DIP.

Las entradas A0, A1 y A2 son utilizadas por el 24FC512 para operaciones de varios dispositivos. Los niveles lógicos en estas entradas son comparadas con los correspondientes bits en la dirección del esclavo. El chip es seleccionado si coinciden. Hasta 8 dispositivos pueden ser conectados a las mismas líneas utilizando diferentes combinaciones de bits

En muchas aplicaciones, las entradas A0, A1y A2 del chip de direccionamiento son difíciles de poner a ‘0’ o a ’1’. Para aplicaciones en las que estas terminales son controladas por un microcontrolador u otro dispositivo lógico programable, las terminales del chip de direccionamiento deben ser puestos a ‘0’ ó ‘1’ lógicos antes de que la operación del

dispositivo normal proceda. . La tabla 3.2 muestra la descripción de terminales de la memoria.

Tabla 3. 2 Descripción de terminales de la memoria EEPROM.

TERMINAL	NOMBRE	FUNCION
1	A0	Direccionamiento
2	A1	Direccionamiento
3	A2	Direccionamiento
4	VSS	Tierra
5	SDA	Datos
6	SCL	Reloj
7	WP	Bloqueo de escritura
8	VCC	+ 2.5 a 5.5 V

***Serial Data* (SDA)**

Esta es una terminal bidireccional usada para transferencia de direcciones y entrada de datos y salida de datos del dispositivo. La línea SDA requiere una resistencia pull-up a Vcc (típicamente 10kΩ para 100kHz, 2kΩ para 400kHz y 1Mhz).

Para una transferencia de datos normal, la línea SDA permite cambiar solo durante el estado en bajo de la línea SCL. Los cambios durante la línea SCL en alto están reservados para indicar las condiciones de inicio y fin.

***Serial Clock* (SCL)**

Esta terminal se utiliza para sincronizar la transferencia de datos hacia y desde el dispositivo.

La capacidad de esta memoria permite almacenar hasta un total de 14 pantallas de información en un chip de 8 terminales, esto hace posible colocar una gran cantidad de información y permite desarrollar una interfaz de usuario muy amigable.

3.3.2. Características del bus

El protocolo del bus tiene las siguientes características:

- La transferencia de datos puede ser iniciada solo cuando el bus no esté ocupado.

- Durante la transferencia de datos, la línea de datos debe permanecer estable siempre que la línea del reloj esté a nivel alto.
- Los cambios en la línea de datos, mientras la línea de reloj, deberá ser interpretada como condición de comienzo o de parada.

Las diferentes condiciones del bus se definen en los términos A, B, C y D mostrados en la figura 3.7 [5].

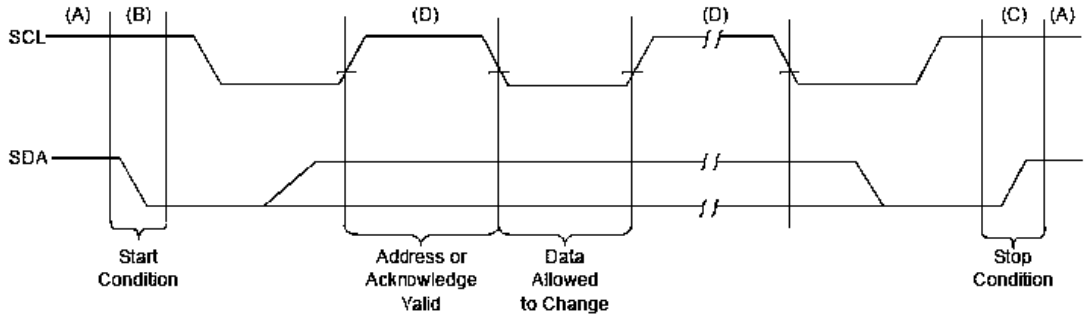

Figura 3. 7 Secuencia en la transferencia de datos en el bus.

A. Bus no ocupado o libre. Ambas líneas (SCL y SDA) permanecen en alto.

B. Condición de inicio de transferencia de datos (*Start condition*). Una transición de bajada de la línea SDA mientras SCL está a nivel alto determina una condición de inicio. Todos los comandos deben ser precedidos por una condición de inicio.

C. Condición de paro de transferencia de datos (*Stop condition*). Una transición de subida de la línea SDA, mientras SCL está a nivel alto determina una condición de paro. Todas las operaciones deben acabar con una condición de paro.

D. Datos validos (Acknowledge valid). El estado de la línea de datos representa datos válidos cuando, después de la condición de inicio, SDA es estable para la duración alta del periodo de SCL. Los datos deben ser cambiados durante el periodo bajo de SCL. Hay un bit de datos por pulso de reloj.

Cada transferencia de datos es iniciada con una condición de inicio y finalizada con una condición de paro. El número de bytes de datos transferidos entre las condiciones de inicio y paro son determinadas por el dispositivo maestro.

Bit de reconocimiento (Acknowledge bit). Cada dispositivo receptor debe generar una señal de reconocimiento después de la recepción de cada byte. El dispositivo maestro debe generar un pulso de reloj extra el cual es asociado con este bit de reconocimiento. Lo anterior se muestra en la figura 3.8.

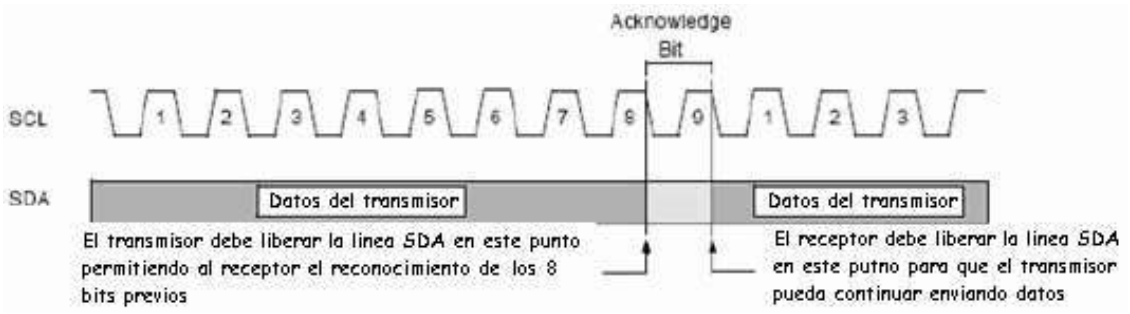

Figura 3. 8 Tiempo de reconocimiento de datos.

NOTA: La memoria 24FC512 no genera ningún bit de reconocimiento si un ciclo de programa interno está en proceso. Un dispositivo de reconocimiento pone a nivel bajo la línea de SDA durante el reconocimiento del pulso de reloj. La línea de SDA está estable durante el periodo alto de reconocimiento relacionado con el pulso de reloj.
Por supuesto, el sistema y el tiempo de mantenimiento deben tomarse en cuenta.
Durante las lecturas, al finalizar la transmisión de datos, se genera un bit de reconocimiento en el último byte que ha sido contabilizado. En este caso, la memoria (24FC512) dejará a nivel alto la línea de datos para que se pueda generar la condición de Paro.

3.3.3. Byte de control

Después de la condición de inicio, el primer byte recibido por la memoria EEPROM se le conoce como byte de control. El formato de este byte de control se muestra en la siguiente figura 3.9 [5].

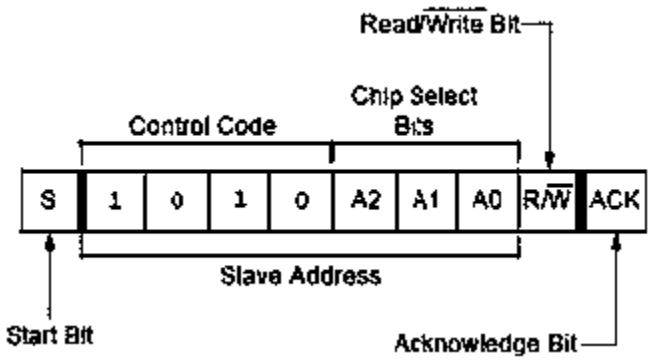

Figura 3. 9 Descripción de los bits del byte de control.

El byte de control consiste en un código de 4 bits de control; para el 24FC512 este es puesto como '1010' binario para las operaciones de lectura y escritura. Los siguientes 3 bits del byte de control son bits de selección del Chip (A2, A1 y A0).

Los bits de selección de chip permiten el uso de hasta 8 dispositivos 24FC512 en el mismo bus y son usados para seleccionar a qué dispositivo se está accediendo. Los bits de selección de chip en el byte de control deben corresponder con los niveles lógicos en los correspondientes pines A0, A1 y A2 para la respuesta del dispositivo.

El último bit del byte de control define la operación a ser realizada. Cuando se pone a 1 se selecciona la operación de lectura y cuando se pone 0 se selecciona la operación de escritura. Los siguientes 2 bytes recibidos definen la dirección del primer byte de datos (Figura 3.10 [5]).

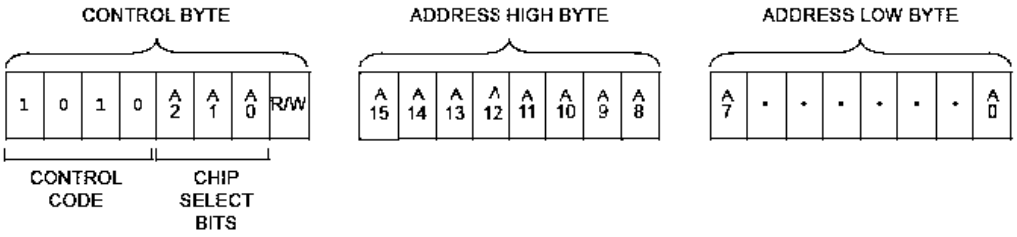

Figura 3. 10 Representación del byte de control y dirección del primer byte de datos.

Es importante tener en cuenta que todos los bits de los 2 bytes de dirección (A15…A0) son utilizados, por lo que no hay ningún bit que obedezca a la condición "*don't*

care". Los bits de dirección superior son los primeros en transferirse, seguido por los bits menos significativos.

En resumen, después de recibir la condición de inicio, la memoria EEPROM 24FC512 revisa la línea SDA en espera del byte de control (1010), y después, de los bits de selección de chip. Enseguida genera el bit de reconocimiento, y finalmente dependiendo del estado del bit R/W, la memoria EEPROM seleccionará la operación de lectura o escritura.

3.3.4. Lectura de memoria

Si se opta por la lectura de la dirección actual, solo es necesario llamar a la función de lectura, y verificar constantemente si hay generación de ACK, ya que tal como ocurre en el proceso de escritura, este bit indicará si se debe continuar o terminar el proceso generándose la condición de paro (Stop). La figura 3.11 muestra lo escrito anteriormente[5].

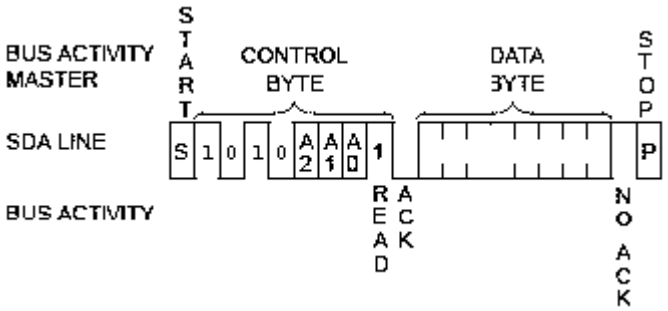

Figura 3. 11 Diagrama de la lectura de la dirección actual.

Ahora bien, si se quiere realizar una lectura secuencial o una aleatoria, se debe antes de comenzar a leer, mover el puntero interno de nuestra memoria. Para realizar esto se debe generar un proceso de escritura, donde se enviará la dirección a la cual queremos acceder, de manera de mover el apuntador hacia esa dirección. Una vez realizado este envío, al primer ACK igual a cero se sale de la rutina de escritura, sin generar una condición de Stop sino que entrando inmediatamente a la rutina de lectura. De esta forma se realiza el proceso de lectura desde la dirección en la cual hemos dejado el puntero interno. Esto es mostrado en el diagrama de la figura 3.12.

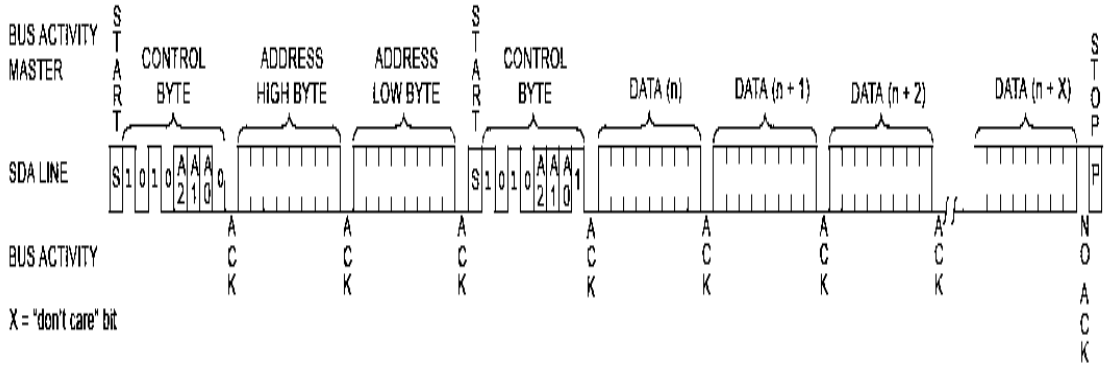

Figura 3. 12 Diagrama de la lectura secuencial.

Cabe destacar que la escritura en la memoria EEPROM se realizó desde el programador PROPIC2 con el software IC-PROG como se describirá mas adelante, por lo que no fue necesaria la implementación de una rutina de escritura a la memoria EEPROM.

3.3.5. Organización de la memoria

La memoria EEPROM se utilizó para almacenar los datos correspondientes a las imágenes presentadas en la pantalla grafica LCD, así como los valores de tensión correspondientes a los perfiles que se muestran en la pantalla de trabajo.

Pantallas realizadas

A partir de imágenes hechas en el programa Paint, se realizó la conversión de la imagen a valores hexadecimales.

Se utilizaron 6 graficas diferentes (figura 3.13), cada una de ellas contiene valores para cada uno de los 30720 píxeles de la pantalla, para hacer el archivo de los datos de cada grafica primero se convirtió el formato BMP de la imagen original a un formato para pantallas graficas LCD en formato BAS, esto sirve para quitar datos inservibles como encabezado del archivo y formato. Después se desarrolló un software que permite utilizar estos datos y compilarlos de manera que todas las imágenes se encuentren contenidas en el mismo archivo binario el software desarrollado es el LCDcompile. Ya que tenemos cada imagen en forma de lista hexadecimal se editó todo en un archivo con ayuda del software xvi32, y distribuimos las graficas cada 1000_h posiciones hexadecimal, iniciando por la

dirección 0000_h, (al hacer esto estamos asegurando un orden y que entre cada grafica tengamos un pequeño espacio libre) a partir de la 7000_h se colocaron los perfiles de cada tipo de ejercicio, el área grafica de la pantalla consta de 240 píxeles de largo se utilizaron 200 para dibujar el perfil del ejercicio, ese valor se utilizó para mostrar en la pantalla la posición vertical del píxel y se envía la salida del PWM a ese valor. La posición horizontal depende del tiempo.

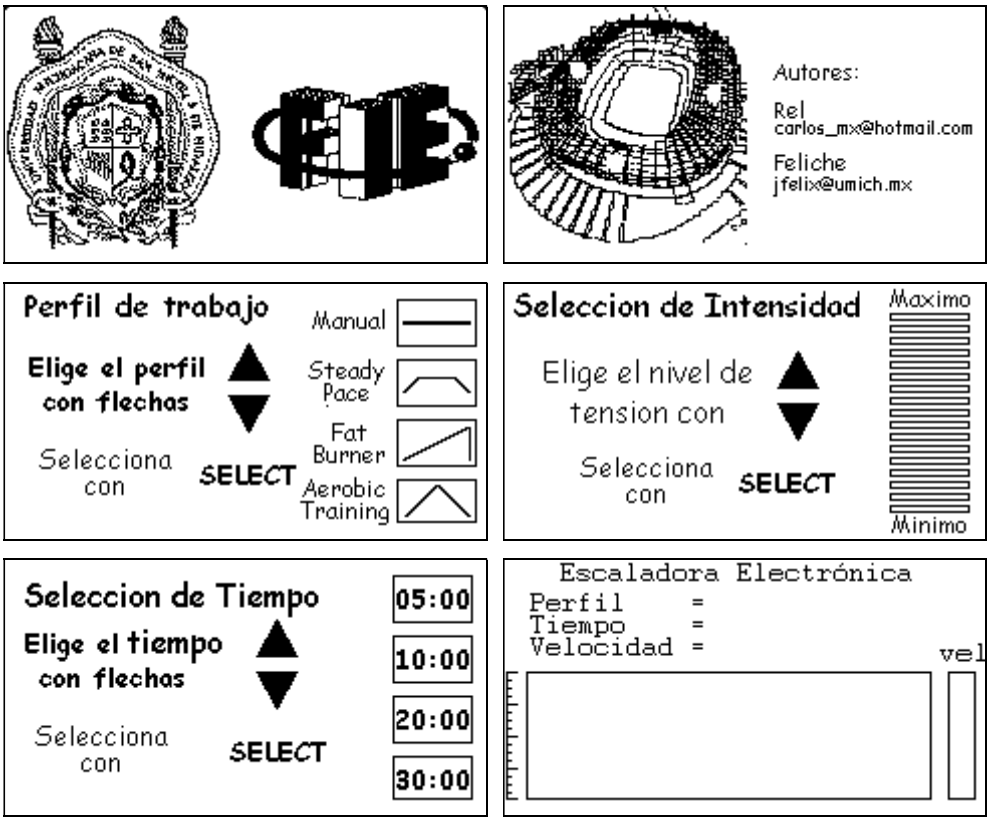

Figura 3. 13 Imágenes Generadas en BMP, compiladas y guardadas en la memoria EEPROM.

Si analizamos la cantidad de datos a introducir en la memoria quedaría de la siguiente manera: cada pantalla es representada por 3840 bytes de datos (F00h bytes) pero para tener una división entre una y otra, usaremos 4096 bytes por cada una y así los datos de la primer pantalla quedará distribuida desde la localidad 0h hasta la localidad FFFh, es decir, utilizara 4kbytes de la capacidad de la memoria para cada pantalla, lo que hace un total de 24 kbytes de la capacidad de la memoria si se suman toda la información usada para las pantallas, y 200 bytes para las coordenadas utilizadas para ir indicando el tipo de

rutina en la pantalla de trabajo. Las cuales son 4 en total, es decir, 800 bytes. La distribución de datos en la memoria es la mostrada en la tabla 3.3. El diagrama de conexión del microcontrolador con la memoria Eeprom es mostrado en la figura 3.14.

Tabla 3. 3 Mapa de la memoria 24FC512.

DIRECCION	DATOS CORRESPONDIENTES EN LA DIRECCION
0 – FFFh	IMAGEN DE ESCUDOS
1000-1FFFh	IMAGEN DE AUTORES
2000-2FFFh	IMAGEN DE TIPOS DE RUTINA
3000-3FFFh	IMAGEN DE NIVEL DE INTENSIDAD
4000-4FFFh	IMAGEN DE SELECCION DE TIEMPO
5000-5FFFh	IMAGEN DE RUTINA DE TRABAJO
6000-6FFFh	VACIO
7000-73FFh	COORDENADAS DE RUTINAS
7400-FFFFh	VACIO

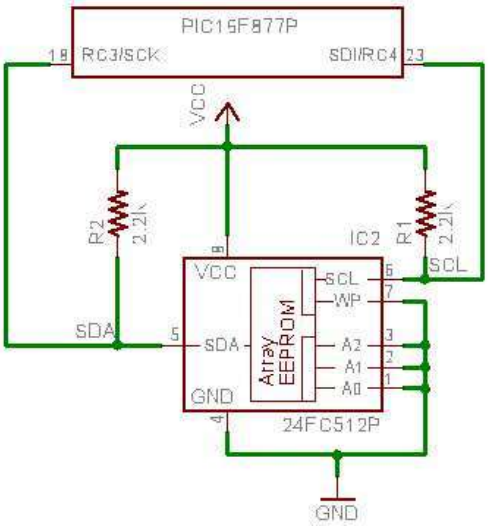

Figura 3. 14 Diagrama de conexión del microcontrolador con la memoria EEPROM.

Perfiles de trabajo

Los 4 perfiles se distribuyeron en 200 datos cada uno distribuidos de la siguiente manera. Se toma en cuenta que el cuadro en el que se graficaran los perfiles el cual es mostrado en la figura 3.15, tiene dimensiones de 200 píxeles de largo y 62 píxeles de alto.

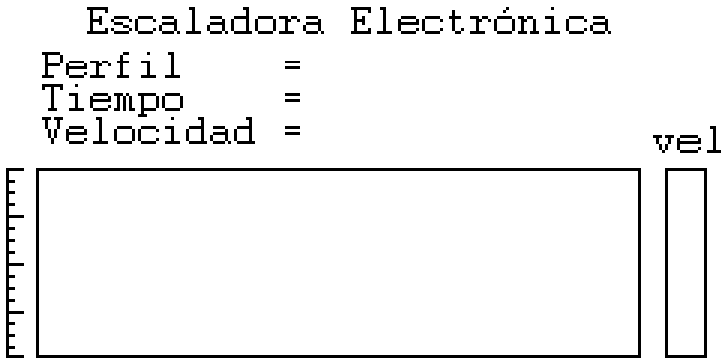

Figura 3. 15 Pantalla de fondo impresa durante la realización de las rutinas.

El perfil lineal

Comienza en el byte 7000h, cada byte representa el valor del píxel en el eje y de la pantalla grafica LCD, como es una línea se repite el mismo valor 200 veces (figura 3.16).

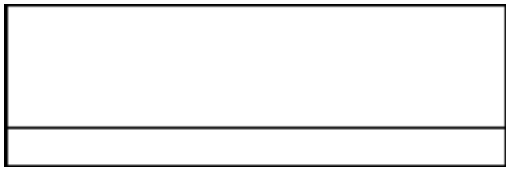

Figura 3. 16 Perfil lineal.

El perfil escalón

La figura 3.17 representa dos trapecios isósceles es decir, al principio la tensión va aumentando, después de aumentar la tensión una cuarta parte de la rutina, se mantiene fija dos cuartas partes para después decrementar e igualar el punto de inicio, esto se repite una vez. Y así se forma la figura de los dos escalones.

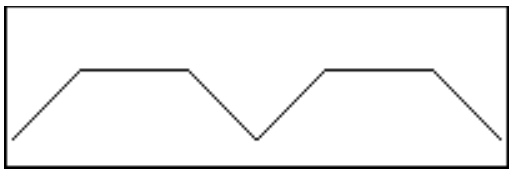

Figura 3. 17 Perfil escalón.

El perfil rampa

El perfil rampa mostrado en la figura 3.18 es una línea recta inclinada que se va incrementando a lo largo del tiempo, después regresa al punto inicial, y se repite dos veces.

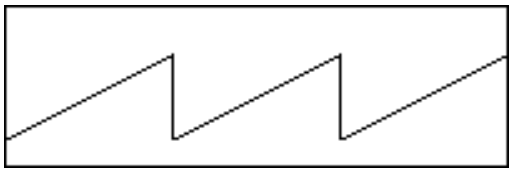

Figura 3. 18 Perfil rampa.

El perfil diente de sierra

El perfil diente de sierra se forma de 3 triángulos isósceles que aumentan gradualmente y al llegar a un valor máximo comienza a disminuir hasta igualar el punto inicial, se repite dos veces. Lo cual se muestra en la figura 3.19.

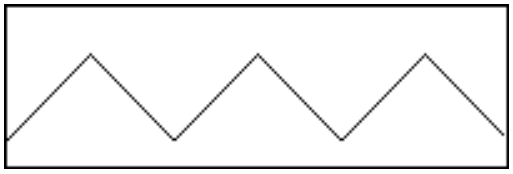

Figura 3. 19 Perfil diente de sierra.

Las variaciones en los niveles de intensidad expresadas gráficamente se realizan agregando un desplazamiento a cada punto de la grafica, se redibuja y se actualiza el valor de la tensión.

Construcción de las imágenes

Los pasos conforme se introdujo cada imagen fueron los siguientes:

1. Se utilizó el programa FASTLCD para dibujar las imágenes a presentar en la pantalla grafica LCD. Todas estas imágenes se guardaron en formato bmp. En la figura 3.20 se muestra el programa con la pantalla de escudos en uso.

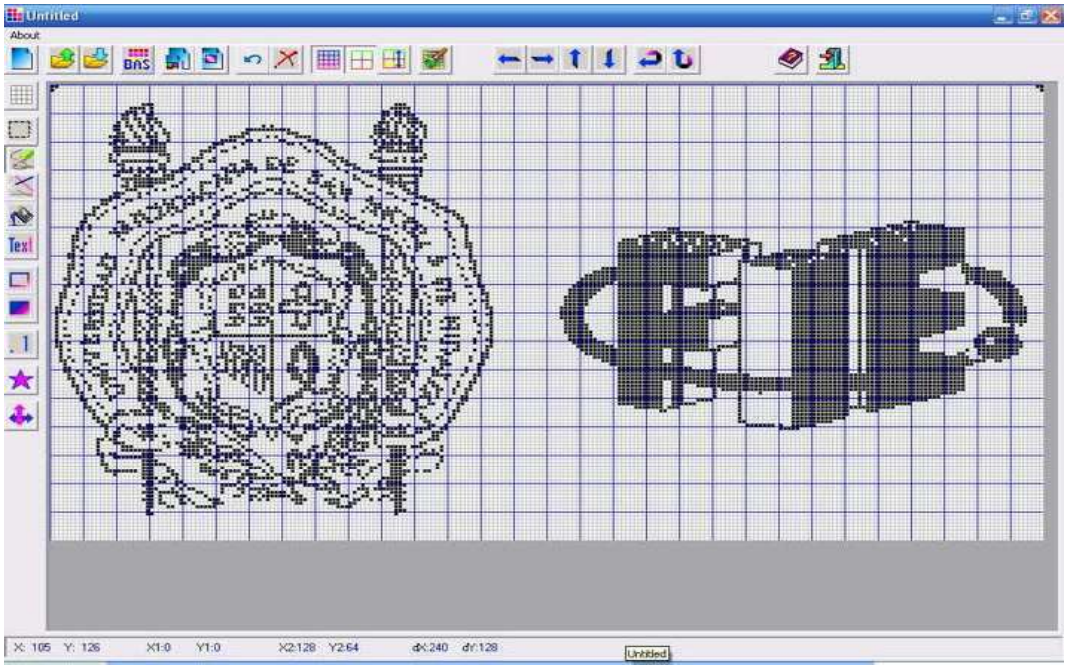

Figura 3. 20 Pantalla de escudos editada en el programa FASTLCD.

2.

La ventaja de utilizar este programa es que convierte los mapas de bits (bmp) en tablas de bytes (en formato de texto de la forma 0xFF), otra ventaja es que el programa corta el encabezado de las imágenes y solo guarda en forma de texto los píxeles que están representados en cada grafica. La tabla de bytes de la imagen de los escudos es mostrada en la figura 3.21.

3. El paso siguiente fue utilizar un programa desarrollado por el Ing. Félix Jiménez Pérez llamado LCDCompile.

Este programa permite pasar los archivos generados por FAST LCD a formato Binario y concatenar hasta 16 diferentes pantallas. En este trabajo se emplean 6

pantallas como se vio anteriormente. La figura 3.22 muestra el programa antes mencionado.

El archivo resultante es un archivo binario (.bin) que contiene la información concatenada de cada una de las imágenes. Pero no es la única información que necesitaremos en la memoria EEPROM, hay que considerar los patrones de ejercicio y desarrollar las tablas para cada patrón y finalmente agregarlas al archivo bin.

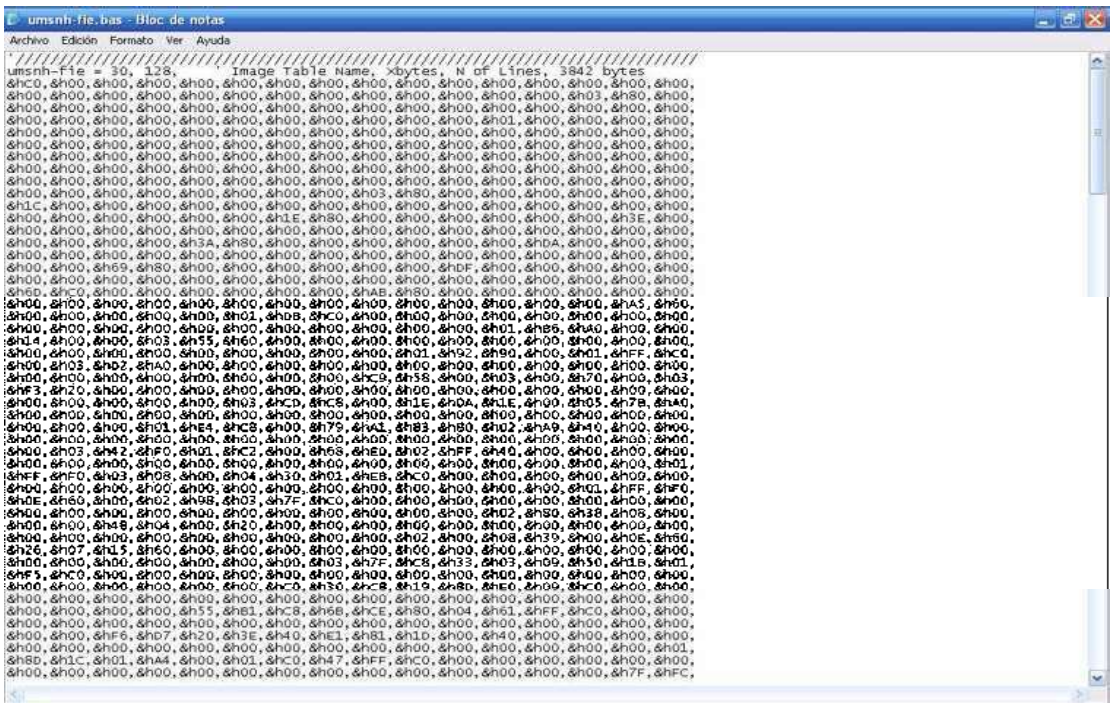

Figura 3. 21 Tabla de bytes de la imagen de los escudos de la universidad y de la facultad.

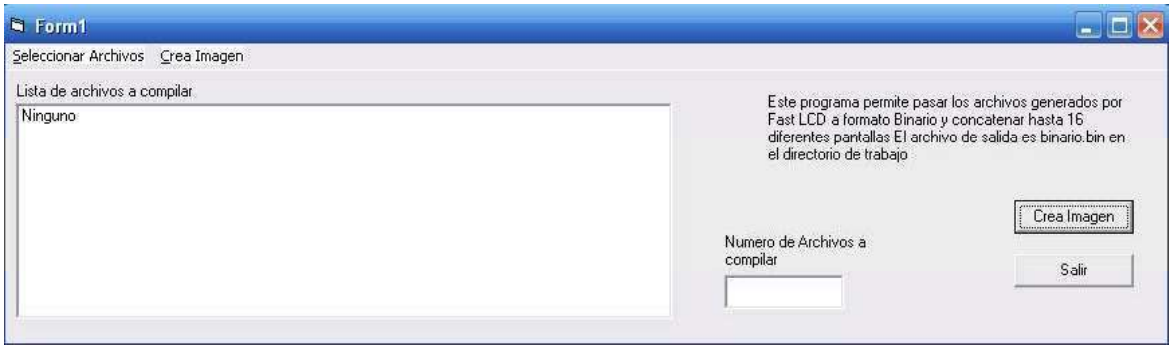

Figura 3. 22 Programa LCDCompile.

3. Para modificar el archivo anterior se uso el editor hexadecimal llamado XVI32(figura 3.23), que tiene ventajas al uso del IC-PROG como el copiar, mover y pegar varias celdas.

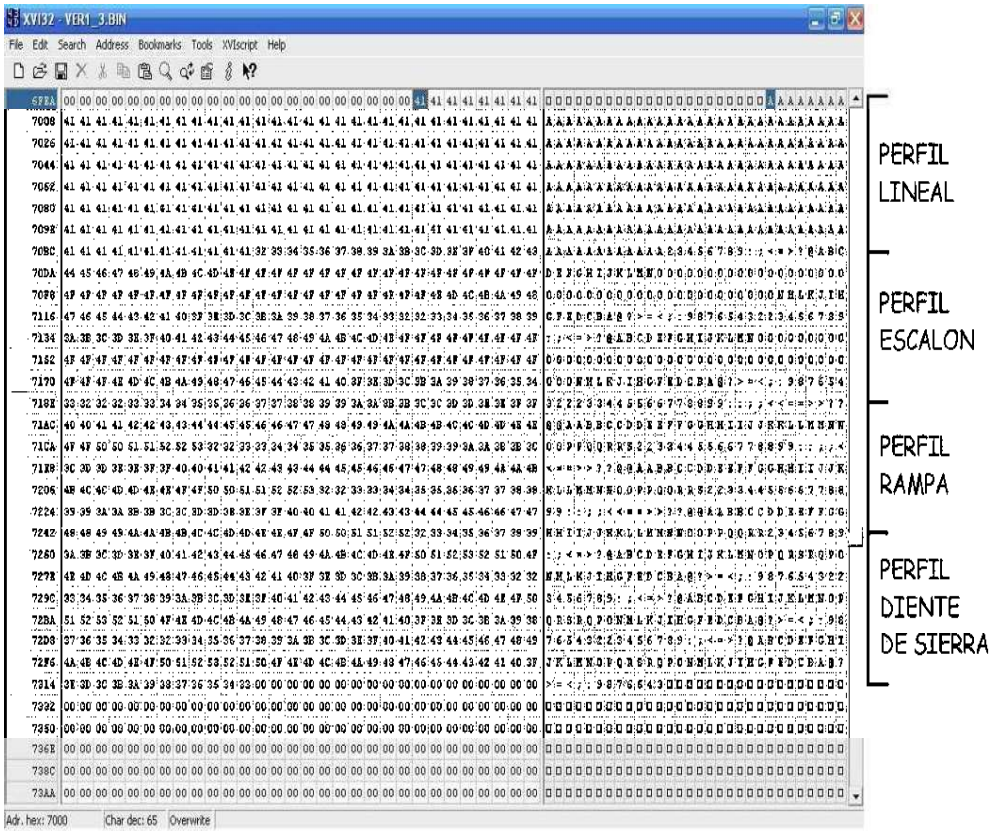

Figura 3. 23 Perfiles a partir de la dirección 7000_h mostradas en el programa XVI32.

3.4. La pantalla grafica LCD

La pantalla grafica LCD es la interfaz de comunicación de la escaladora con el usuario por medio de imágenes.

3.4.1. Características de la pantalla utilizada

La pantalla utilizada es la AND 1741, el cual presenta las siguientes características:

- Es monocromática (blanco-azul).
- Cuenta con backlight, lectura nítida, con alto contraste.

- Amplio rango de temperatura de operación (0 a 50 º C).
- Memoria RAM de 8k bytes.

Durante las pruebas no se presentó ningún problema de visibilidad con respecto a la pantalla grafica LCD, aun en condición de luces apagadas.

Tamaño

El tamaño de la pantalla es de 170 mm x 105 mm (incluyendo el marco), 126mm x 70mm en la parte visible un tamaño adecuado si pensamos que la mayor parte de los que utilizaran la escaladora son mayores de 20 años. La pantalla se muestra en la figura 3.24[6].

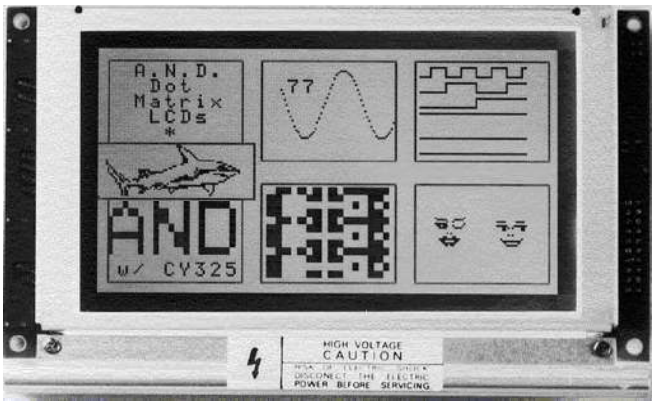

Figura 3. 24 Pantalla grafica LCD AND 1741.

Píxeles

La resolución de la pantalla es de 240 x128 píxeles, es decir 30720 píxeles agrupados en 3840 bytes como es mostrado en la figura 3.25.

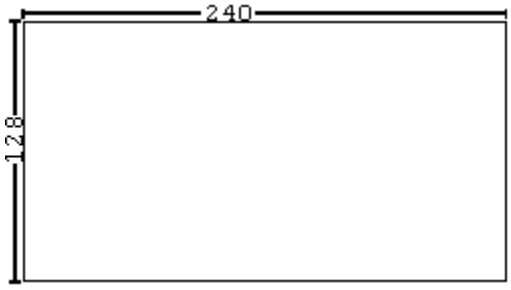

Figura 3. 25 Distribución de píxeles en la pantalla grafica LCD.

Al trabajar en modo grafico, los píxeles están distribuidos en grupos horizontales de 8 píxeles representado uno de ellos en la figura 3.26. Si tomamos en cuenta que son 30720 píxeles, por lo tanto, el número de grupos total se contabilizó en 3840(figura 3.27).

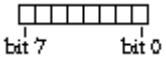

Figura 3. 26 Posición de los píxeles dentro del grupo.

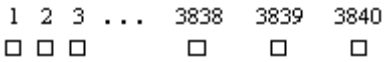

Figura 3. 27 Distribución de los grupos de píxeles.

Aunque en la actualidad hay pantallas usadas en celulares o en PDA´s con mayor resolución, para el uso que se le va a dar en este trabajo es suficiente. En modo texto tiene capacidad 40 caracteres x 18 líneas, y tiene dos fuentes, una de 8x8 píxeles y la otra de 6x8 píxeles. Sin que esto cambie la capacidad de 40 caracteres por línea. En la figura 3.28 se muestra el diagrama a bloques de conexiones de la pantalla [6].

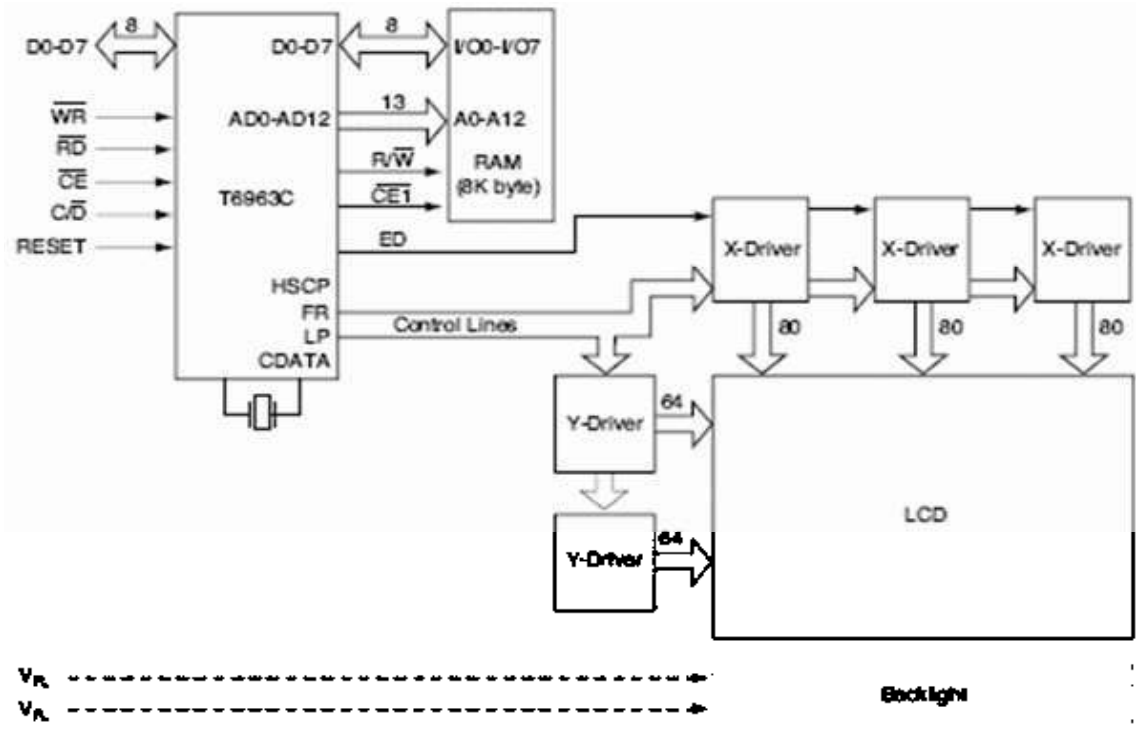

Figura 3. 28 Diagrama a bloques de conexiones internas y externas de la pantalla.

Controlador T6963C

El controlador T6963 utiliza comunicación asíncrona es decir no utiliza señal de reloj y se comunica con el exterior a través de 13 terminales, 8 de ellas para las líneas de datos (D0-D7), WR prepara a la pantalla para la escritura de datos, RD prepara la pantalla para la lectura de datos, CE habilita el chip, C/D se utiliza conjuntamente con WR y RD para leer y escribir datos en memoria respectivamente, ejecutar comandos y ver el estado de la pantalla.

Voltajes de operación

La pantalla grafica LCD necesita de distintos voltajes, los cuales son:

5 volts: este voltaje es para la alimentación general de la pantalla grafica LCD.

-13 volts: Este voltaje es útil para ajustar el contraste de la pantalla grafica LCD (llamada también V_{EE}), cuenta con un rango de voltaje, en función de un rango de temperatura de operación. Este voltaje es el adecuado para utilizarla a la temperatura ambiente de Morelia.

La manera en que se obtiene el valor de 13 volts es la siguiente:

Tomando como temperatura ambiente mayor a 25 grados centígrados, y la pantalla grafica LCD del tipo BST. De la tabla 3.4 el voltaje que le corresponde es 18.5 [6].

Tabla 3. 4 Temperatura contra voltaje de trabajo.

Temperatura	V_{DD}-V_{EE} (BST)
0ºC	20,0
+25ºC	18,5
+50ºC	16,6

La ecuación que corresponde a estos datos es la siguiente:

Tenemos que

$$V_{DD} - V_{EE} = 18.5 \text{ V} \qquad (3.1)$$

Despejando de la ecuación 3.1

$$V_{EE} = V_{DD} - 18.5 = 5 - 18.5 = -13.5 \qquad (3.2)$$

Se redondea para tener un margen de trabajo ya que por lo regular la temperatura sobrepasa los 25 grados.

750 volts: Este voltaje es utilizado para encender la lámpara de fondo de la pantalla grafica LCD a este tipo de lámpara se le conoce como lámparas de cátodo frió (CCFL Cold Cathode Fluorescent Lighting), sirve para que la imagen sea brillante y en condiciones de poca iluminación se pueda observar claramente. La figura 3.29 es un ejemplo de este tipo de lamparas.

Figura 3. 29 Luz fluorescente

Las características de la lámpara son las siguientes:

Vmax=1500 rms

I= 10 mA rms

f= 80 hz

3.4.2. Rutinas implementadas

Antes de enviar a la pantalla grafica LCD cualquier instrucción tenemos que leer el estado en que se encuentra, para realizar la lectura se coloca un 1 lógico en C/D y un 0 en RD, esto prepara a las líneas de datos para acceder al registro de estado de la pantalla grafica LCD. El registro de estado de la pantalla grafica LCD se describe en la Figura 3.30 [6].

STA7	STA6	STA5	STA4	STA3	STA2	STA1	STA0
MSB							LSB

Figura 3. 30 Distribución de los bits de estado en las líneas de datos de la pantalla grafica LCD.

Consta de 8 bits, iniciando de derecha a izquierda, cada bit describe las siguientes situaciones.

STA0 (Ocupado1): Verifica la capacidad para ejecutar una instrucción.

STA1 (Ocupado2): Verifica la capacidad para leer o escribir datos.

STA2: Verifica la capacidad de leer datos en modo automático.

STA3: Verifica la capacidad de escribir datos en modo automático.

STA4: Sin ningún propósito.

STA5: Verifica la posibilidad de que el controlador realice una operación.

STA6: Condición de error.

STA7: Condición de parpadeo.

Para la realización de este trabajo solo se requirió del uso de los estados 0, 1, 3 y 5.

Para el correcto funcionamiento de la pantalla grafica LCD se tienen que seguir los diagramas de flujo de la figura 3.31.

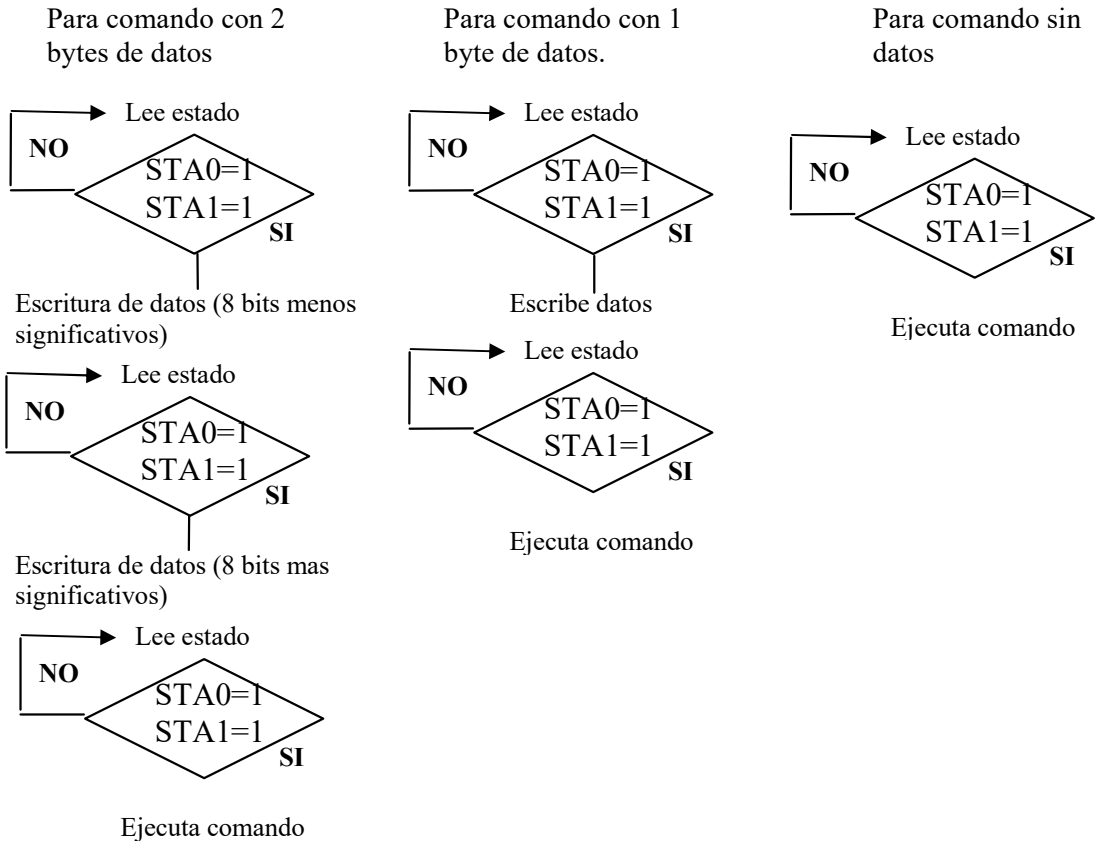

Figura 3. 31 Diagramas de flujo del funcionamiento de la pantalla grafica LCD para introducir datos.

Para el funcionamiento adecuado se requiere de implementar funciones para el control de la pantalla grafica LCD entre las que se encuentran:

Command Esta función prepara a la pantalla grafica LCD para mandar un comando, primero se lee el registro de estado y espera hasta que los bits STA0 y STA1 estén en alto, posteriormente se envía el comando, y deshabilita la escritura y la pantalla grafica LCD.

```
void Command(unsigned char Comando)
{
        Busy();                         // Espera a que no este ocupado
        CD = 1;                         // Modo Comando
        WR = 0;                         // Activa la escritura
        CE = 0;                         // Activa el display

        BUS = Comando;          // Envía el comando
        WR = 1;                         // Deshabilita la escritura
        CE = 1;                         // Deshabilito el display
}
```

ReadStatus Esta función lee el estado de registro, para esto es indispensable colocar los bits de las líneas de control de la pantalla grafica LCD CD=1, RD=0 y CE=0. Enseguida se leen las líneas de datos y se guarda en la variable Dato. Deshabilita la lectura y la pantalla grafica LCD. Deja las líneas de datos en modo salida.

```
unsigned char ReadStatus(void)
{
unsigned char Dato;
        TRISD = 0xFF;           // Bus de entrada
        CD = 1;                         // Modo Comando
        RD = 0;                         // Activa la lectura
        CE = 0;                         // Activa el display
        NOP();
        Dato = BUS;             // Lee el dato y lo regresa
```

```
        RD = 1;                         // Deshabilita la lectura
        CE = 1;                         // Deshabilito el display
        TRISD = 0x00;           // Retorna a modo salida
        return Dato;            // retorno el dato leido
}
```

Data Esta función permite enviar un dato a la pantalla grafica LCD, las líneas del control de la pantalla grafica LCD se colocan CD=0, WR=0 y CE=0. Se escribe el dato en las líneas de datos. Deshabilita la lectura y la pantalla grafica LCD.

```
void Data(unsigned char Dato)
{
        Busy();                         // Espera a que no este ocupado
        CD = 0;                         // Modo Comando
        WR = 0;                         // Activa la escritura
        CE = 0;                         // Activa el display
        BUS = Dato;             // Envía el comando
        WR = 1;                         // Deshabilita la escritura
        CE = 1;                         // Deshabilito el display
        CD = 1;                         // No es necesario !!!!!
}
```

DataAuto Esta función permite recibir datos continuo en la pantalla grafica LCD, espera que no este ocupado, las líneas del control de la pantalla grafica LCD se colocan CD=0, WR=0 y CE=0. Se escribe el dato en las líneas de datos. Deshabilita la lectura y la pantalla grafica LCD.

```
void DataAuto(unsigned char Dato)
{
        bit3();                 // Espera a que no este ocupado
        CD = 0;                         // Modo Comando
        WR = 0;                         // Activa la escritura
        CE = 0;                         // Activa el display
```

```
        BUS = Dato;              // Envía el comando
        WR = 1;                  // Deshabilita la escritura
        CE = 1;                  // Deshabilito el display
        CD = 1;                  // No es necesario !!!!!
}
```

Redistribución de los píxeles

A partir de las graficas 3.25, 3.26 y 3.27 la distribución de píxeles se resume en las figuras 3.32 y 3.33. A partir de estas graficas se planteo una redistribución de píxeles de una manera más natural tipo primer cuadrante del plano cartesiano.

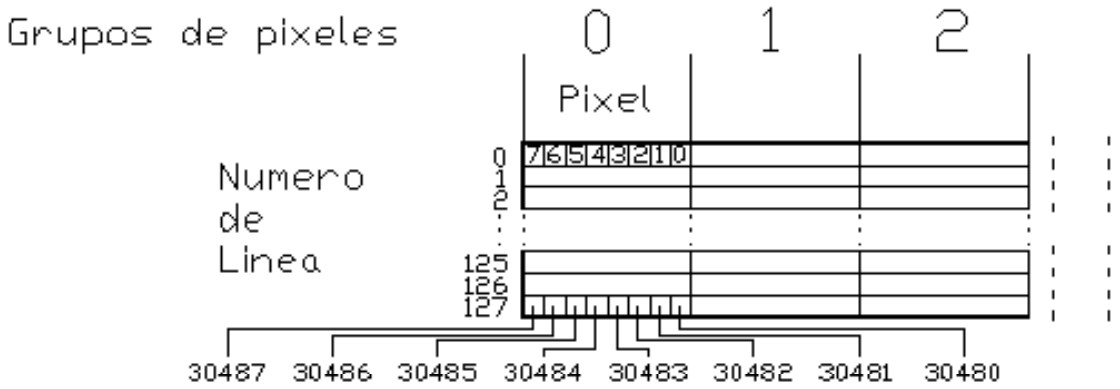

Figura 3. 32 Representación de los píxeles agrupados en grupos, en la parte izquierda de la pantalla.

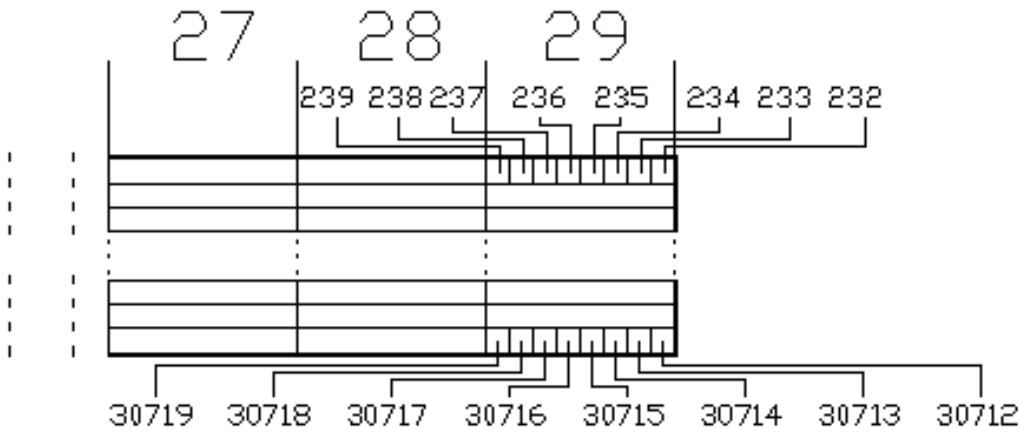

Figura 3. 33 Representación de los píxeles agrupados en grupos, en la parte derecha de la pantalla.

Esta distribución fue hecha mediante la programación de una función teniendo en cuenta que a partir de un dato en X y un dato en Y se obtenga la dirección que la pantalla grafica LCD esta distribución se hace mas "natural" como se muestra en la figura 3.34.

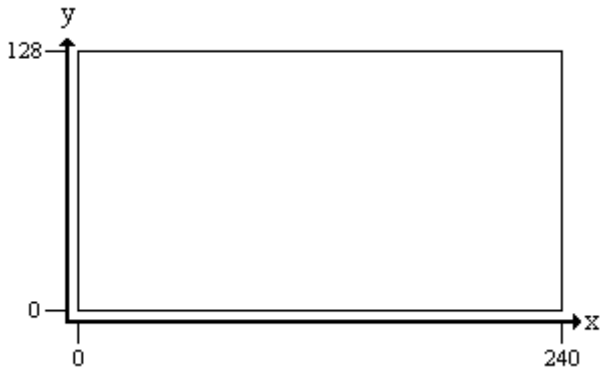

Figura 3. 34 Nueva distribución de los píxeles en eje x y eje y.

Coordenada (X,Y)

Aunque hay dos cosas a considerar, la primera son los bloques de 8 píxeles en forma horizontal y todos continuos comenzando por el píxel colocado en la parte superior a la izquierda.

1.- Primeramente se les resta una unidad tanto a la coordenada en x como a la de y. Esto se muestra en las ecuaciones 3.3 y 3.4:

$$x = X - 1 \quad (3.3)$$
$$y = Y - 1 \quad (3.4)$$

2.- Se obtiene el residuo de x entre 8, para saber cual es la posición del bit en el grupo de 8 píxeles. Esto se muestra en la ecuación 3.5.

$$bit = x\%8 \quad (3.5)$$

Sin embargo, aun no se puede usar esta posición porque en el grupo de píxeles el MSB esta a la izquierda y el LBS a la derecha como lo muestra la figura 3.35.

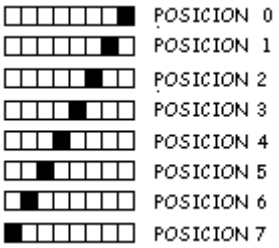

Figura 3. 35 Representación del orden de los píxeles en un grupo de ellos.

3.- Para hacer coincidir el píxel con el residuo, se le restan 7. Con esto se logra invertir la posición del residuo, y se multiplica por -1 para convertirlo al número positivo que se requiere como lo muestran los resultados de la tabla 3.5. Las ecuaciones 3.6 y 3.7 son las correspondientes a este paso.

$$bit = bit - 7 \tag{3.6}$$

$$bit = -bit \tag{3.7}$$

Tabla 3. 5 Orden resultante del residuo después de restarle 7 y multiplicarlo por -1.

Residuo	Orden resultante
7	0
6	1
5	2
4	3
3	4
2	5
1	6
0	7

4.- Ya se tiene la posición dentro del grupo de píxeles, enseguida se procede a conocer el número del grupo de píxeles en el que se encuentra. Para esto simplemente se divide la coordenada de x entre 8. Para esto se recorren los bits a la derecha 3 veces. Como se muestra en la ecuación 3.8.

$$x = x >> 3 \tag{3.8}$$

5.- Después se utiliza una variable llamada Address (dirección) donde se guarda la dirección del grupo de bits después de restarle a 127 la coordenada de y, ya que estamos considerando que la coordenada la tomamos en cuenta teniendo el cuadrante 1 de un sistema coordenado a partir de la parte inferior izquierda. Se multiplica por 30 porque es el número de grupos de píxeles en una línea. Finalmente la suma de lo anterior con x resulta en la dirección del grupo al cual el píxel pertenece. Como se muestra en la ecuación 3.9.

$$Address = x + ((127 - y) * 30) \tag{3.9}$$

6.- El valor máximo o posición final que podría alcanzar nuestra variable Address es 3839(0EFF), sin embargo para enviar esa dirección al controlador de la pantalla grafica LCD es necesario dividirla en dos partes ya que la línea de datos es de 8 bits, entonces se guarda la parte mas significativa que podría ser de 0^{**}_h a E^{**}_h en AddressH y la parte $*00_h$ hasta $*FF_h$ en AddressL, y así ya queda enviar al controlador de la pantalla grafica LCD la dirección. Las ecuaciones 3.10 y 3.11 son las correspondientes a este paso.

$$AddrH = Address / 0x100 \tag{3.10}$$

$$AddrL = Address \% 0x100 \tag{3.11}$$

Ejemplo:

Para probar el funcionamiento de la función anterior aplicaremos los pasos anteriores a un punto el cual se localiza en el punto intermedio de la pantalla, la dirección de este punto es P= (120,64).

Paso 1.- Aplicando las ecuaciones 3.2 y 3.3:

x = X – 1 = 120 – 1 y = Y – 1 = 64 – 1

x = 119 y = 63

Paso 2.- Aplicando la ecuación 3.4:

bit = x % 8 = 119 % 8 = 7

Paso 3.- Aplicando las ecuaciones 3.5 y 3.6:

bit = bit – 7 = 7 – 7 = 0

bit = -bit = -0 = 0

Paso 4.- Aplicando la ecuación 3.7:

x = x >> 3 = 119 >>3 = 14

Paso 5.- Aplicando la ecuación 3.8:

Address = x + ((127 – y) * 30) = 14 + ((127 – 63) * 30)

Address = 14 + (64*30) = 14 + 1920 = 1934 = 0x78E

Paso 6.- Aplicando las ecuaciones 3.9 y 3.10:

AddrH = Address / 0x100 = 0x78E / 0x100 = 7

AddrL = Address % 0x100 = 0x78E % 0x100 = 8E

SetPixel

Coloca un píxel

SetPixel(x,y)

La función SetPixel utiliza la función anterior, y a partir de una coordenada x y otra y, coloca la pantalla grafica LCD en modo grafico, obtiene la dirección del display para esas coordenadas, y envía el comando para que imprima el píxel antes mencionado.

```
void SetPixel(unsigned char x, unsigned char y)
{
unsigned int Address;
unsigned char AddrH;
unsigned char AddrL;
unsigned char Bit;

    x=x-1;
    y=y-1;
    Bit = (x%8);
    Bit = Bit-7;
    Bit = -Bit;
    Bit = Bit|0xF8;
```

```
x=x>>3;
Address= x+((127-y)*30);
AddrH = Address / 0x100;
AddrL = Address % 0x100;

Data(AddrL);            // LSB Posicion de Inicio de Graficos = 0x0000
Data(AddrH);            // MSB
Command(0x24);              // Inicio de la direccion de Graficos

Command(Bit);               // Impresion de Pixeles

}
```

ClearPixel

Limpiar un píxel

ClearPixel(x,y)

Esta función difiere únicamente de la anterior al cambiar el estado del píxel enviado.

```
void ClearPixel(unsigned char x, unsigned char y)
{
unsigned int Address;
unsigned char AddrH;
unsigned char AddrL;
unsigned char Bit;

x=x-1;
y=y-1;
Bit = (x%8);
Bit = Bit-7;
Bit = -Bit;
Bit = Bit|0xF0;
```

```
        x=x>>3;
        Address= x+((127-y)*30);
        AddrH = Address / 0x100;
        AddrL = Address % 0x100;

        Data(AddrL);            // LSB Posicion de Inicio de Graficos = 0x0000
        Data(AddrH);            // MSB
        Command(0x24);          // Inicio de la direccion de Graficos

        Command(Bit);                   // Impresion de Pixeles
}
```

Línea

Dibuja una línea

Linea(x1,y1,x2,y2)

Se realizaron dos funciones, una para las líneas horizontales y otra para líneas verticales. Para la función que realiza una línea horizontal se toma en cuenta el punto de inicio a la izquierda de la pantalla en “x”, el punto de término en “x”, y la altura de la línea, además si es para imprimir el píxel o para limpiar su estado. Se establece un ciclo en el cual imprime el píxel de inicio, y sigue pintando uno por uno hasta llegar al píxel de término.

Para la línea vertical varia en que los datos de entrada son el punto de inicio en el eje “y” es el que esta mas cercano a la parte baja de la pantalla, y el punto de termino en “y”, y la posición en el eje “x” de la línea.

```
void Linea(unsigned char x1, unsigned char y1, unsigned char x2, unsigned char y2)
{
        float m;
        float b;

        m=(y2-y1)/(x2-x1);
```

```
        b=(m*x1)-y1;

        if(y1==y2) LineaHorizontal(x1,x2,y1);

        if(x1==x2) LineaVertical(y1,y2,x1);

        if((x2-x1)>(y2-y1))
        {
        unsigned char i;
                for(i=x1;i<x2;i++)
                {
                unsigned char y;
                y=(m*i)+b;
                SetPixel(i,y);
                }
        }
        if((y2-y1)>(x2-x1))
        {
        unsigned char i;
                for(i=y1;i<y2;i++)
                {
                unsigned char x;
                x=(i-b)/m;
                SetPixel(x,i);
                }
        }
}
```

Imprimir grafico

Para la impresión de un grafico en la pantalla LCD se debe primero comprender la configuración de la distribución de la memoria RAM para su manejo. El controlador

T6963C posee internamente 8 Kbytes de memoria RAM. Una sugerencia del fabricante para la distribución de la memoria RAM para la pantalla AND 1741 con fuente de texto de 8x8 se muestra en la tabla 3.6 [6].

Tabla 3. 6 Configuración sugerida de la memoria RAM de la pantalla grafica LCD.

AND	711, 1391, 1741, 1781, 8 x 8
Text Home	1000H
Text Area	01EH
Graphic Home	0000H
Graphic Area	01EH
Ext. CG Home	1000H
Attribute Home	0D00H
Valid Address	0000H to 1FFFH

La pantalla grafica LCD como se observa en la figura anterior, se puede configurar para que opere en modo texto, en modo grafico, usando un generador de caracteres externos, usando atributos especiales de la pantalla los cuales son invertir los píxeles en cierto texto, o en cierta imagen; y que parpadee alguna parte del texto o alguna imagen. Para hacer uso de estas características se tienen disponibles los 8 Kbytes(0x0000-0x1FFF) de memoria del controlador T6963C.

La configuración indicada para el proyecto es la siguiente:

Dirección inicial de gráficos. Utilizaremos la sugerencia del fabricante, es decir, la dirección inicial de gráficos corresponde a 0x0000.

Área gráfica. Esta área se configura de acuerdo al numero de grupos de píxeles que utilizaremos para nuestro caso serán 30, es decir, 0x001E. Esto representa los 30 bytes de 8 bits, es decir, los 240 píxeles como se muestra en la figura.3.36.

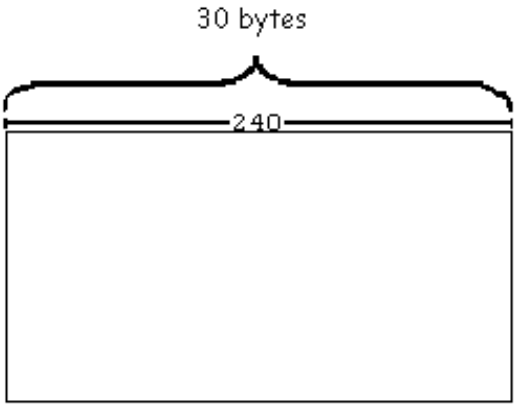

Figura 3. 36 Representación de los píxeles en bytes.

Dirección inicial del texto. Como cada grafica que corresponde a la pantalla utiliza 3840 bytes(0x0F00) y para guardar distancia entre una y otra abarcan 4096 bytes (0x0FFF). La dirección sugerida del fabricante en el byte 0x1000 es adecuada.

Área de texto. Esta también será igual a 30, que corresponde al número máximo de caracteres por línea de texto.

Para imprimir en la pantalla grafica LCD las imágenes previamente guardadas en la memoria EEPROM, se utilizo la siguiente función:

Imprimir gráfico (Dirección en la memoria EEPROM de pantalla a imprimir)

Esta función realiza la impresión en la pantalla grafica LCD de la siguiente manera:

1.- Recibe la dirección de la pantalla a imprimir.

2.- Inicia la pantalla grafica LCD en modo grafico.

3.- Copia desde la memoria EEPROM byte por byte del primer renglón en la memoria RAM de la pantalla grafica LCD.

4.- Realiza la misma operación con los demás renglones.

5.- Muestra los datos en la pantalla.

3.4.3. Conexión de la pantalla grafica LCD al microcontrolador

La pantalla grafica LCD cuenta con 20 terminales con diferentes funciones cada una, las cuales se conecta de acuerdo a la tabla 3.7. En la figura 3.37 se muestra el diagrama esquemático de conexión entre el microcontrolador y la pantalla LCD.

Tabla 3. 7 Tabla de conexión entre el microcontrolador y los demás dispositivos.

No. Terminal	Señal	Significado	Conectado con:
1	FGND	Tierra física	GND
2	GND	Referencia	GND
3	VDD	Alimentación (5 V)	Fuente de + 5 V
4	VEE	Alimentación del controlador (-13.5 V)	Fuente de - 13.5 V
5	WR	Escritura de datos	Puerto B
6	RD	Lectura de datos	
7	CE	Habilitar chip	
8	C/D	Dato/comando	
9	NC	Sin conexión	
10	RESET	Reset del controlador	Terminal 1 (MCLR) del microcontrolador
11	D0	Entrada / salida de datos (LSB)	Puerto D
12	D1	Entrada / salida de datos	
13	D2	Entrada / salida de datos	
14	D3	Entrada / salida de datos	
15	D4	Entrada / salida de datos	
16	D5	Entrada / salida de datos	
17	D6	Entrada / salida de datos	
18	D7	Entrada / salida de datos (MSB)	
19	FS	Selección de fuente para el modo texto (VDD 6x8 pixeles GND 8x8 pixeles)	GND
20	RV	Tipo de imagen (VDD Imagen normal GDN Imagen negativa)	Fuente + 5 V

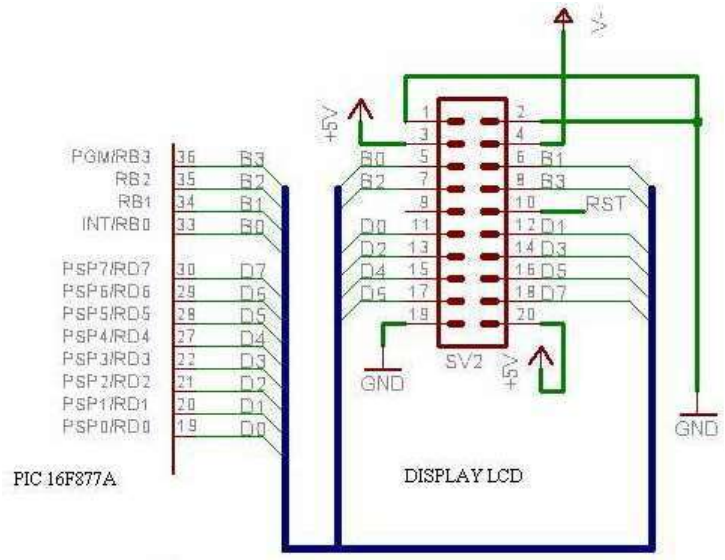

Figura 3. 37 Diagrama esquemático de las conexiones de la pantalla grafica LCD.

Las líneas de datos

Las líneas de datos sirven para comunicar al microcontrolador con la pantalla grafica LCD, tanto para enviar como para recibir información. En la figura 3.38 se muestra la manera en que están interconectados.

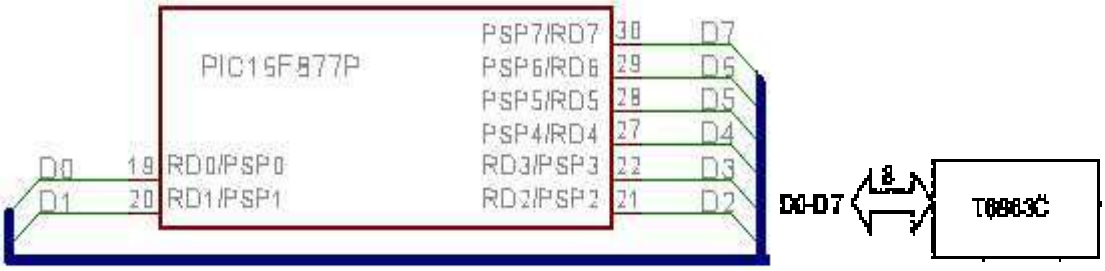

Figura 3. 38 Conexión de las líneas de datos del microcontrolador a la pantalla grafica LCD.

Las líneas de control

Las líneas de control sirven para modificar el modo de operación de la pantalla grafica LCD principalmente para prepararla para enviar o recibir información. En la figura 3.39 se muestra la manera en que están interconectados el microcontrolador y la pantalla grafica LCD.

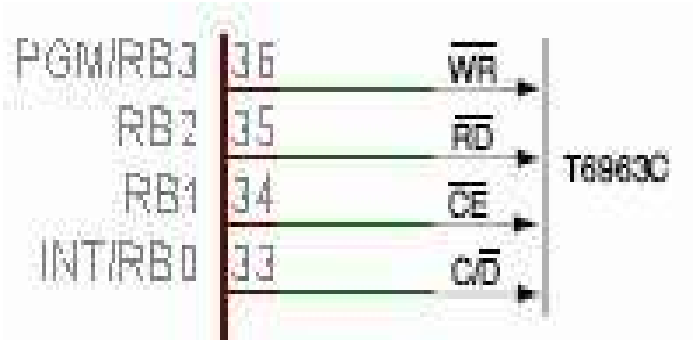

Figura 3. 39 Conexión de las líneas de control del microcontrolador a la pantalla grafica LCD.

Antes de la realización del circuito de control del alternador, se implementó por separado la graficación en la pantalla grafica LCD como se muestra en la figura 3.40.

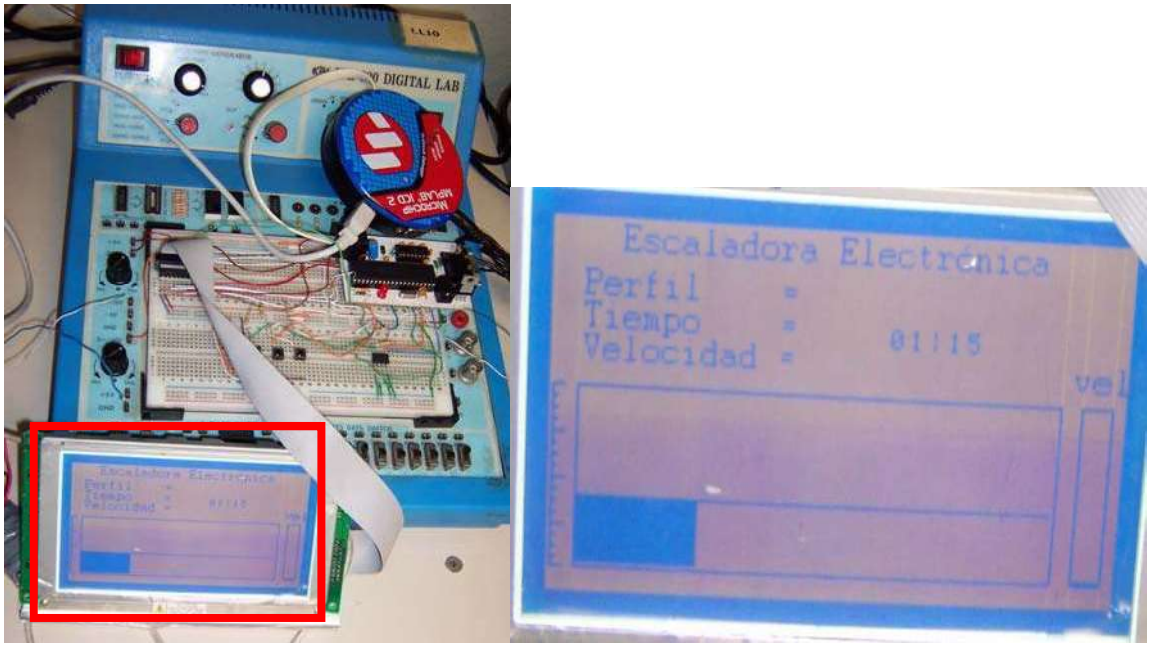

Figura 3. 40 Pruebas con la pantalla grafica LCD.

Voltajes de operación

Para la operación de la pantalla grafica LCD se requieren las siguientes fuentes de voltaje: en VDD (terminal 3) = +5 volts, en VEE (terminal 4) = -13.5 volts y en RV (terminal 20) = 5 volts. La alimentación RV es exclusivamente para configuración de la imagen. Lo anterior es mostrado en la figura 3.41.

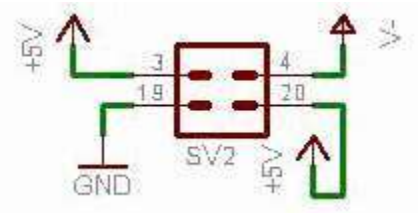

Figura 3. 41 Conexión de la pantalla grafica LCD a las fuentes de alimentación.

Para la operación de la luz de fondo se construyó un selector de voltaje porque como el circuito fue obtenido de un escáner y fue implementado al final, se conectó a 12 volts mediante un selector para proporcionar el voltaje al circuito inversor y proporcionar los 750 volts requeridos por la lámpara. El diagrama de conexión del selector de voltaje se muestra en la figura 3.42.

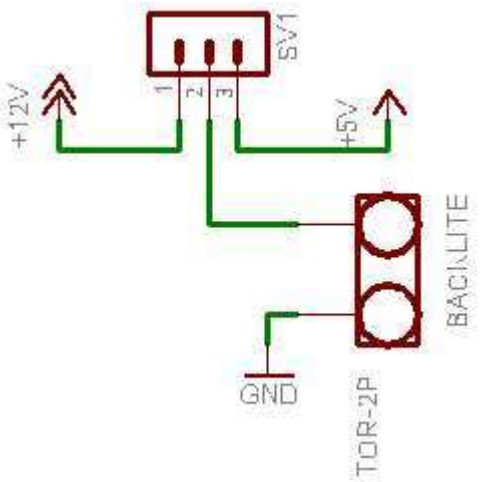

Figura 3. 42 Conexión del inversor de voltaje al circuito backlite.

3.5. Teclado de la escaladora

Debido a que el teclado se encontró en perfecto estado y contribuye a la armonía estética de la escaladora se decidió incluirlo en el nuevo diseño.

Características del teclado:

- Es de tipo membrana.
- Consta de 4 botones (arriba, abajo, enter y reset).
- Teclas del tipo normalmente abierto.
- Tienen una resistencia de 10 Ω al presionar una tecla.

Figura 3. 43 Teclado original de la escaladora.

Este teclado se conectó al puerto A del microcontrolador y recibe las señales del teclado (la tecla de reset va conectada directamente al MCLR que reinicia el microcontrolador).

Estas teclas son activas en bajo y requieren una resistencia pull up. Funciona de la siguiente manera: mientras que ningún botón es accionado el voltaje en la terminal de salida es 5 volts y cuando se acciona algún botón cambia a un valor de cero volts.

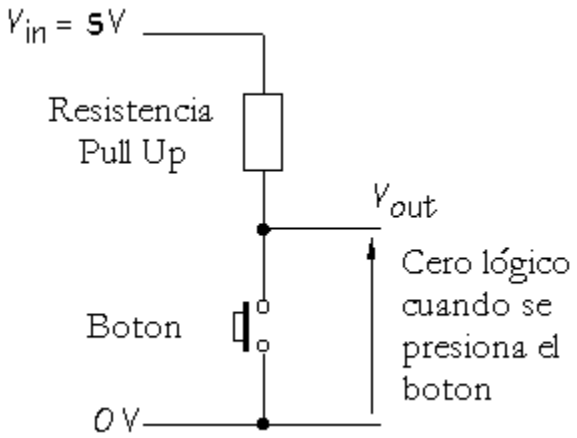

Figura 3. 44 Diagrama de conexión de una tecla.

3.5.1. Descripción del teclado

Se utilizan los mismos 4 botones implementados en la consola de control original los cuales son de izquierda a derecha: Reset, flecha arriba, flecha abajo y enter.

3.5.2. Funciones del teclado

El teclado sirve para seleccionar las condiciones de la rutina que se va a realizar, la flecha de incremento o decremento van a aumentar o disminuir las condiciones de tiempo y tensión respectivamente, y cambiar el tipo de rutina a realizar. Es importante recordar que si se presiona más de un botón el microcontrolador detectara el botón que fue presionado primero debido a que el proceso para detectar una tecla excede la rapidez con la cual se pueden mover los dedos para presionar los botones.

Botón 

este botón reinicia el microcontrolador debido a que está conectado a la terminal MCLR.

Botón  este botón sirve para mover el cuadro hacia abajo en las pantallas de selección de ejercicio, y en la pantalla de trabajo sirve para disminuir el nivel de tensión del ejercicio.

Botón  este botón sirve para mover el cuadro hacia arriba en las pantallas de selección de ejercicio, y en la pantalla de trabajo sirve para aumentar el nivel de tensión del ejercicio.

Botón 

este botón solo funciona en los menús de selección de rutina, ya que al presionarlo pasa a la siguiente pantalla, definiendo a cada paso las características de la rutina que se va a realizar.

3.5.3. Conexión del teclado al microcontrolador

En la figura 3.45 se observa como la terminal MCLR del microcontrolador esta conectada a su vez con la terminal de reset de la pantalla grafica LCD, lo que ocasiona que al apretar el botón reset de la escaladora el microcontrolador y la pantalla grafica LCD se reinicien.

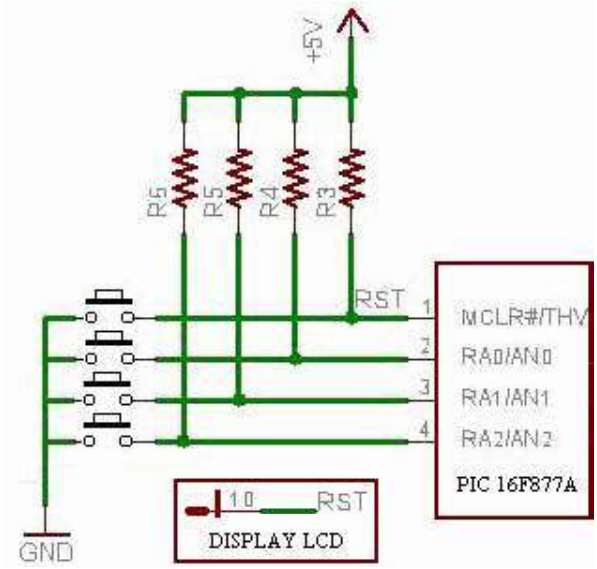

Figura 3. 45 Diagrama de conexión del teclado con el microcontrolador.

3.6. Control de tensión de la escaladora

La tensión mecánica de la escaladora se obtiene controlando la corriente del rotor. El control de tensión de la escaladora lo realiza el microcontrolador PIC 16F877A, el método de control de tensión requiere de una corriente que no es capaz de suministrar el microcontrolador, así el sistema de potencia tiene como función principal proveer la corriente que requiere el rotor del alternador.

El circuito de la figura 3.46 muestra el diagrama esquemático del control de potencia, el control de la corriente esta basado en PWM, esto requiere que los transistores operen en conmutación. Una de las conexiones del rotor se encuentra fija a GND, por esto se utiliza un transistor PNP, pero en este caso el transistor NPN BC338(Q1) funciona como un interruptor abriendo y cerrando el circuito, además de invertir el voltaje en la base de Q2, cuando la señal esta en alto se cierra ya que se alimenta la base. Esto provoca en el transistor PNP TIP42C(Q2) la salida invertida y con esto logre alimentar el alternador el cual esta conectado en su parte positiva a referencia con una corriente mas alta que la que puede proporcionar por si mismo el microcontrolador. El diodo zener (D5) funciona durante la desactivación del alternador lo cual provoca una corriente de descarga en sentido inverso que pone en peligro el elemento electrónico utilizado para su activación. Un diodo polarizado inversamente cortocircuita dicha corriente y elimina el problema. Se le llama comúnmente diodo volante.

3.6.1. El modo PWM del microcontrolador

El Modulo PWM del PIC 16F877A tiene una resolución máxima de 10 bit y su salida se encuentra en la terminal RC2, por lo cual se configura como salida en el registro Tris C.

Periodo PWM

Una salida PWM tiene una base de tiempo (periodo) y un tiempo que salida se mantiene en alto (ciclo útil). Como se muestra en la siguiente figura 3.47.

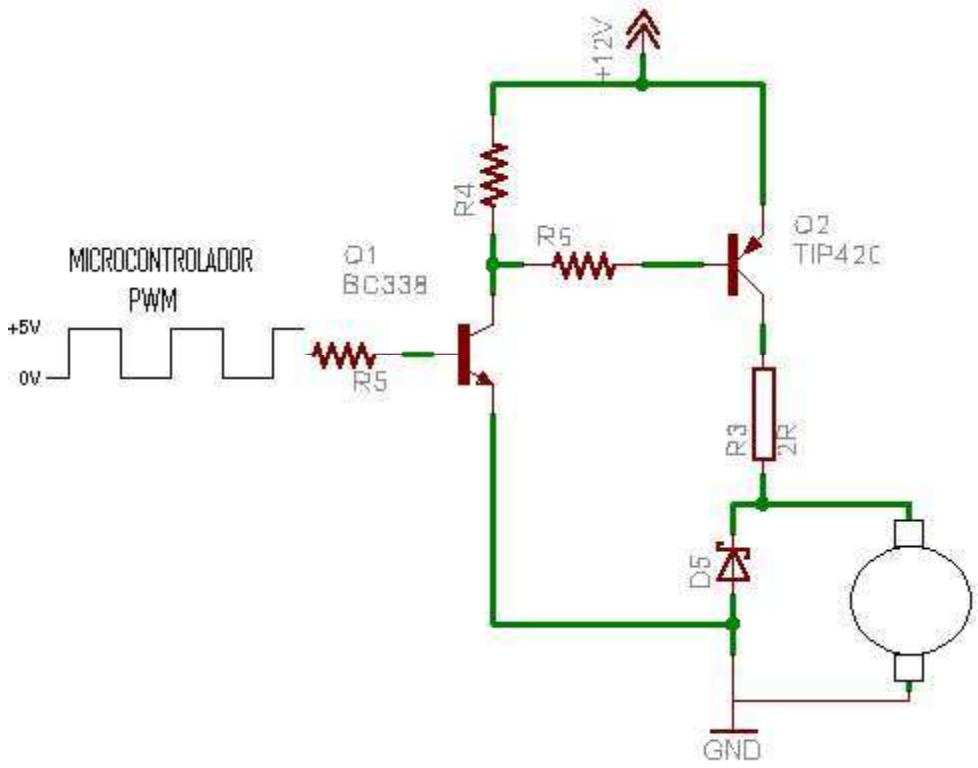

Figura 3. 46 Diagrama del circuito de excitación del campo.

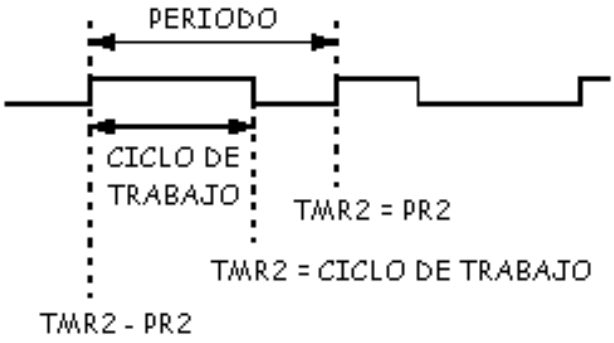

Figura 3. 47 Periodo y ciclo de trabajo.

El periodo del modulo PWM es configurado dando un valor al registro PR2. Este periodo puede ser calculado usando la ecuación 3.12.

$$Tpwm = [(PR2)+1]*4*Tosc*preescalador \qquad (3.12)$$

Donde

$Tpwm$ es el periodo de la onda PWM en RC2..

$PR2$ es el valor del registro PR2.

$Tosc$ es el periodo de oscilación del cristal conectado al microcontrolador.

$preescalador$ el valor del preescalador del timmer 2 (los valores posibles son 1,2, 4 y 16).

Además, frecuencia es:

$$Fpwm = \frac{1}{Tpwm} \tag{3.13}$$

Cuando el valor del Timer 2 alcanza el valor de PR2, los siguientes eventos ocurren en el siguiente ciclo de tiempo:

- TMR2 es limpiado.
- La terminal CCP1 es puesto a 1.
- El periodo de trabajo del PWM es cambiado de CCPR1H a CCPR1L.

Ciclo de trabajo del PWM.

El ciclo útil del PWM se especifica escribiendo en el registro CCPR1L más los bits 5,4 del registro CCP1CON. Se dispone de una resolución de 10 bits. El registro CPR1L contiene los ocho bits más significativos y el registro CCP1CON contiene los dos bits menos significativos. Este valor de 10 bits esta representado por CCPR1L:CCP1CON<5:4>.

El ciclo de trabajo se obtiene usando la ecuación 3.14.

$$\mathrm{D} = (CCPR1L : CCP1CON < 5:4 >) * Tosc * preescalador \tag{3.14}$$

Donde:

D es el tiempo que dura el ciclo de trabajo de la onda PWM.

$(CCPR1L:CCP1CON<5:4>)$ es el valor de 10 bits entre los registros CCPR1L y CCP1CON

$Tosc$ es el período de oscilación del cristal conectado al microcontrolador

$preescalador$ es el valor del preescalador del timmer 2 (los valores posibles son 1,2, 4 y 16)

Para modificar el ciclo de trabajo, en cualquier momento se puede escribir a CCPR1L y CCP1CON<5:4>, pero estos valores no serán tomados en consideración hasta que el Timer2 alcance al PR2 y reinicie su operación tomando en cuenta los nuevos valores.

Para calcular la máxima resolución del PWM a determinada frecuencia usamos la ecuación 3.15.

$$\text{Resolución} = \frac{\log\left(\frac{\text{Fosc}}{\text{Fpwm}}\right)}{\log(2)} bits \qquad (3.15)$$

Configuración del modulo PWM:

Pasos para la configuración del Modulo PWM del PIC 16F877A

1. Configurar el Periodo dando un valor al registro PR2

2. Configurar el Ciclo de Trabajo escribiendo en: CCPR1L:CCP1CON<5:4>

3. Limpiar el Tris C2 para asignar la salida del modulo en el registro CCP1

4. Asignar el valor del preescaldor del Timer 2 al registro T2CON

5. Configurar el modulo CCP1 para operación PWM.

La configuración que se aplicó para el uso del PWM en este proyecto es la siguiente:

$PR2 = 0xFF = 255$

$Tosc = 5\exp{-8}$ para un cristal de 20MHz.

$preescalador = 16$

Sustituyendo $Tpwm$ de la ecuación 3.11 en la ecuación 3.12 se tiene:

$$Fpwm = \frac{1}{[(PR2)+1]*4*Tosc*preescalador} \quad (3.16)$$

Sustituyendo valores se tiene

$$Fpwm = \frac{1}{[(255)+1]*4*(5\exp-8)*16} = 1220.70Hz$$

$$Fpwm = 1.22KHz$$

Esta frecuencia se elige de acuerdo a la tabla 3.8.

Tabla 3. 8 Ejemplo de frecuencias pwm y resoluciones con cristal de 20 MHz.

PWM FREQUENCY	1,22 kHz	4,88 kHz	19,53 kHz	78,12 kHz	156kHz	208,3 kHz
Timer Prescaler (1, 4, 16)	16	4	1	1	1	1
PR2 Value	0xFFh	0xFFh	0xFFh	0x3Fh	0x1Fh	0x17h
Maximum Resolution (bits)	10	10	10	8	7	5,5

El valor T_{pwm} puede tener 256 valores, porque la variable salida que es usada para modificar el valor del PWM solo modifica el registro CCPR1L es decir los 8 bits más significativos, porque los 2 bits restantes no implican un cambio significativo.

3.6.2. Rango de operación

Existen 17 niveles de intensidad en las rutinas de operación de la escaladora son 17. Iniciando desde el número 1 el cual es el menor nivel de intensidad, con incrementos de 1 en 1, hasta llegar al 17. Cada nivel corresponde a una variación de 5.88 % del ciclo de trabajo del PWM.

En la figura 3.48, se aprecia la relación entre la variación del PWM y los diferentes niveles de intensidad de la escaladora.

3.6.3. Ecuación de control de PWM vs Tensión

Para el uso del PWM en el microcontrolador se configuró el timer 2 con las siguientes características.

Salida de 8 bits. Es decir tiene un rango de variación de 0 a 255 valores.

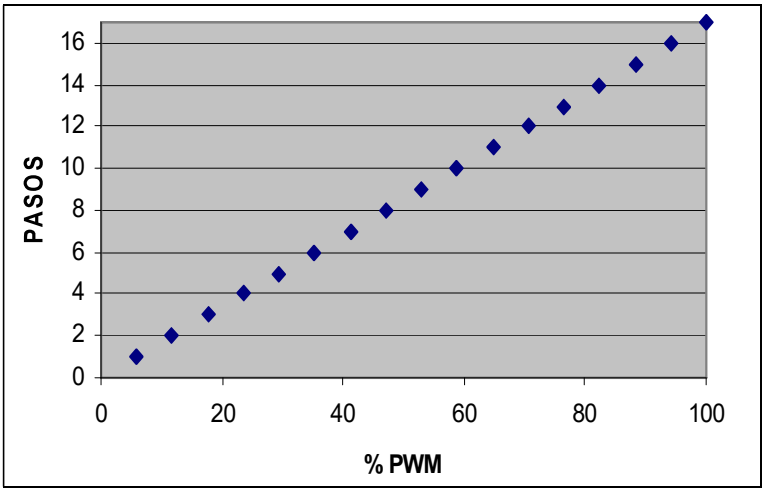

Figura 3. 48 Pasos de resistencia mecánica vs % PWM.

El voltaje de salida del microcontrolador en modo PWM se conmuta entre 0 y 5V, sin embargo con el circuito amplificador se modifica de 0 a 12V.

Para detallar el incremento respecto al PWM y el voltaje medio se muestra la siguiente tabla. El valor inicial de la salida es 15 porque la grafica que corresponde a la selección de nivel de intensidad consta de 17 espacios para variar el nivel. Al dividir 255 entre 17 resulta 15. Y no existe el nivel 0 porque sin resistencia en los pedales no es posible a la escaladora ofrecer resistencia al peso del usuario.

Tabla 3. 9 Nivel PWM y voltaje de operación.

Nivel	Salida Digital PWM	% PWM	Voltaje en el campo
1	15	5.88	0.71
2	30	11.76	1.41
3	45	17.65	2.12
4	60	23.53	2.82
5	75	29.41	3.53
6	90	35.29	4.24
7	105	41.18	4.94
8	120	47.06	5.65
9	135	52.94	6.35
10	150	58.82	7.06
11	165	64.71	7.76
12	180	70.59	8.47
13	195	76.47	9.18
14	210	82.35	9.88
15	225	88.24	10.59
16	240	94.12	11.29
17	255	100.00	12.00

La grafica 3.49 corresponde a la ecuación de una recta, al obtener la ecuación se encuentra que el voltaje VCD es VCD= 0.12 PWM.

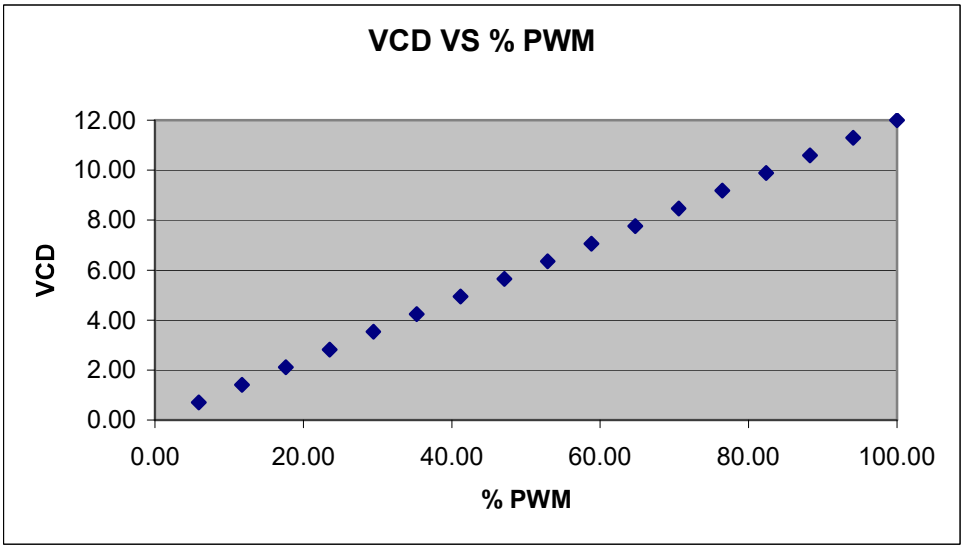

Figura 3. 49 Grafica voltaje amplificado vs % PWM.

3.7. Medición de velocidad de intensidad de ejercicio

Esta medición sirve al usuario para observar la intensidad de su ejercicio, este parámetro es útil para estimar los demás parámetros.

3.7.1. El sensor de velocidad

El alternador cuenta con un encoder interconectado que permite tener una señal de frecuencia y voltaje proporcional a la velocidad de giro del alternador y por tanto la velocidad del pedaleo.

3.7.2. Caracterización del sensor

La captura de la señal se realizó de manera experimental después de medir la frecuencia con un pedaleo suave (que hace girar lentamente al alternador) y con un pedaleo intenso (que hace girar rápidamente el alternador), el rango dentro del cual estuvieron las medidas fue el siguiente.

Frecuencia de salida del encoder:

Frecuencia mínima= 10 Hz.

Frecuencia máxima= 1 kHz

La medición se realizó utilizando el modulo CCP1 del microcontrolador, así los cálculos para realizar las mediciones de manera adecuada son los siguientes:

Fosc= 20 Mhz (Valor nominal del cristal utilizado)

La captura se realiza utilizando el timer1, el cual es de 16 bits.

El preescaler utilizado es 8 debido a que el rango que se requiere medir se encuentra entre los valores que se pueden capturar, así :

0.2 useg(2^{16})*8=0.104 Fmin= 9.53 Hz
0.2 useg(2^{0})*8= 1.6useg Fmax=625 k Hz
Donde 2^{16} es la cuenta máxima del timer1 y (2^{0})=1 es la cuenta mínima.

El número de pulsos que el microcontrolador utiliza para capturar en el rango de operación se calcula de la siguiente manera:

A 10 Hz
$0.1/1.6x10^{-6}$=62 500 pulsos.

A 1000 Hz
$0.001/1.6x10^{-6}$= 625 pulsos.

Esto quiere decir que mientras más baja sea la frecuencia, se requieren mas cuentas del timer1, como sucede a 10 Hz.

3.7.3. Conexión del sensor al microcontrolador

El encoder esta conectado al microcontrolador mediante una terminal en la parte exterior del alternador como se muestra en la figura 3.50 [2].

Figura 3. 50 Terminales en el costado del alternador.

Como se aprecia en la figura anterior, hay 4 terminales, las cuales hacen referencia de izquierda a derecha a +12, neutro, pwm y encoder respectivamente. En la figura 3.51 se muestra el diagrama de conexión del microcontrolador con el encoder.

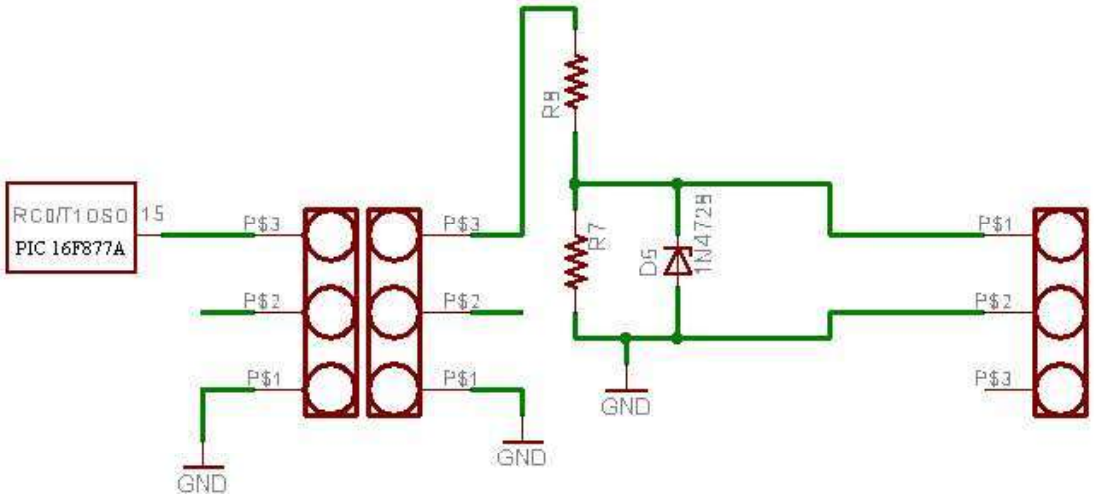

Figura 3. 51 Diagrama de conexión del microcontrolador con el encoder.

3.7.4. Nivel de voltaje

El rango de voltaje que entrega el encoder va desde 100 mV hasta 5 V. Los 100 mV son generados cuando la persona que esta pedaleando la escaladora lo hace con una velocidad muy baja, esto sucede cuando se aplica el mayor nivel de intensidad y por ende en los pedales se ejerce la mayor oposición al pedaleo. Caso contrario durante la generación de 5 volts a la velocidad de pedaleo más rápida. Pero a pesar de contar con un nivel de voltaje que es útil para medir la velocidad, se utilizó la frecuencia como variable para medir la velocidad. Ya que la frecuencia es menos sensible al ruido eléctrico.

3.7.5. Determinación de los parámetros de medición de velocidad vs frecuencia

Los valores intermedios en la tabla 3.10 son aproximados debido a que al pedalear en la escaladora se observó que tendían a ser proporcionales a la velocidad de pedaleo.

Tabla 3. 10 Datos de Frecuencia y velocidad.

Frecuencia (Hz)	% Velocidad
10	1
100	10
200	20
300	30
400	40
500	50
600	60
700	70
800	80
900	90
1000	100

La velocidad se tomó en porcentaje con referencia en la máxima velocidad al pedalear. En la grafica 3.52 se puede observar que la respuesta del sensor es aproximadamente lineal con respecto al pedaleo del usuario.

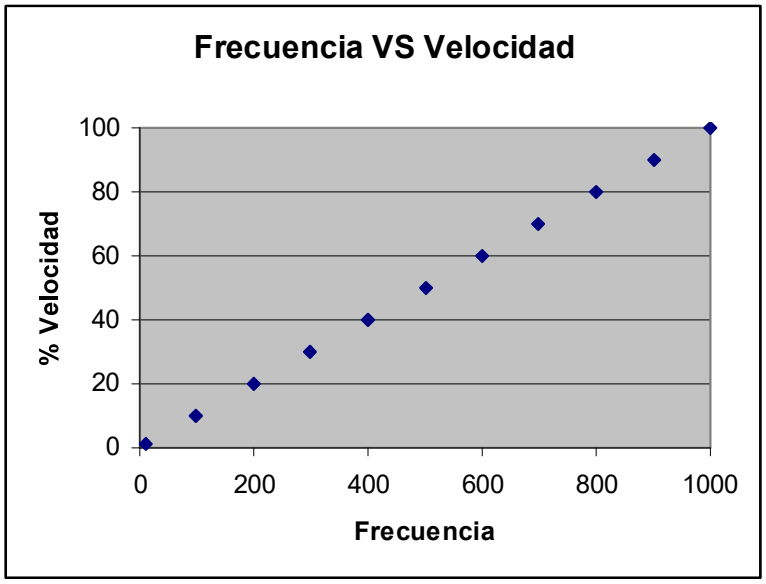

Figura 3. 52 Grafica frecuencia contra % velocidad.

La ecuación % Velocidad = 0.1 Frecuencia corresponde a la respuesta del encoder.

3.8. Las fuentes de voltaje

Al igual que el microcontrolador que funciona con una fuente de 5 V, para la operación de la pantalla grafica LCD se necesitan diferentes voltajes, en esta parte se describen las fuentes que suministran dichos voltajes.

3.8.1. La fuente de 5 V

Para alimentar con 5 volts de la fuente de la escaladora de 12V, se diseñó un circuito con un regulador 7805 y dos capacitores, uno de ellos se encargara de filtrar las bajas frecuencias y el otro las altas. El diagrama de esta fuente se muestra en la figura 3.53.

Esta fuente utilizada por la mayor parte de los circuitos implementados por lo que se requiere de una gran cantidad de corriente, entre los dispositivos que utilizan la fuente están:

- El microcontrolador
- La memoria
- La pantalla grafica LCD
- El teclado

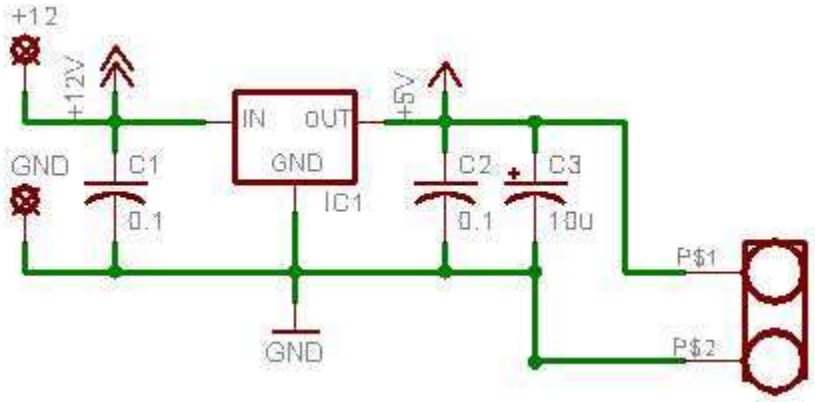

Figura 3. 53 Diagrama de la fuente de 5 V.

3.8.2. La fuente de -13 V

Para alimentar el contraste(-18V) de la pantalla grafica LCD se plantea un arreglo de fuentes.

Primero para lograr un voltaje negativo, se conecta la salida del regulador ajustable LM317 a la referencia(figura 3.54), esto permite controlar el voltaje de manera simple, y se ajusta para que a la salida haya 13volts de manera permanente. Esta conexión permite regular el voltaje de salida.

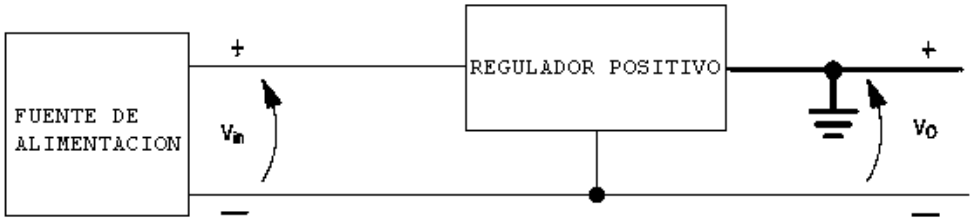

Figura 3. 54 Conexión de un regulador positivo para lograr un voltaje negativo.

Como se observa en la figura 3.55, se conecta la referencia con la salida positiva.

Así para obtener los 18 volts que se necesitan justificados en los cálculos anteriores, se toma como positivo la salida de la fuente de 5 volts y la parte negativa de la fuente de 13(es decir -13), para que esto funcione, es necesario contar con una fuente de CA, en este caso un pequeño transformador.

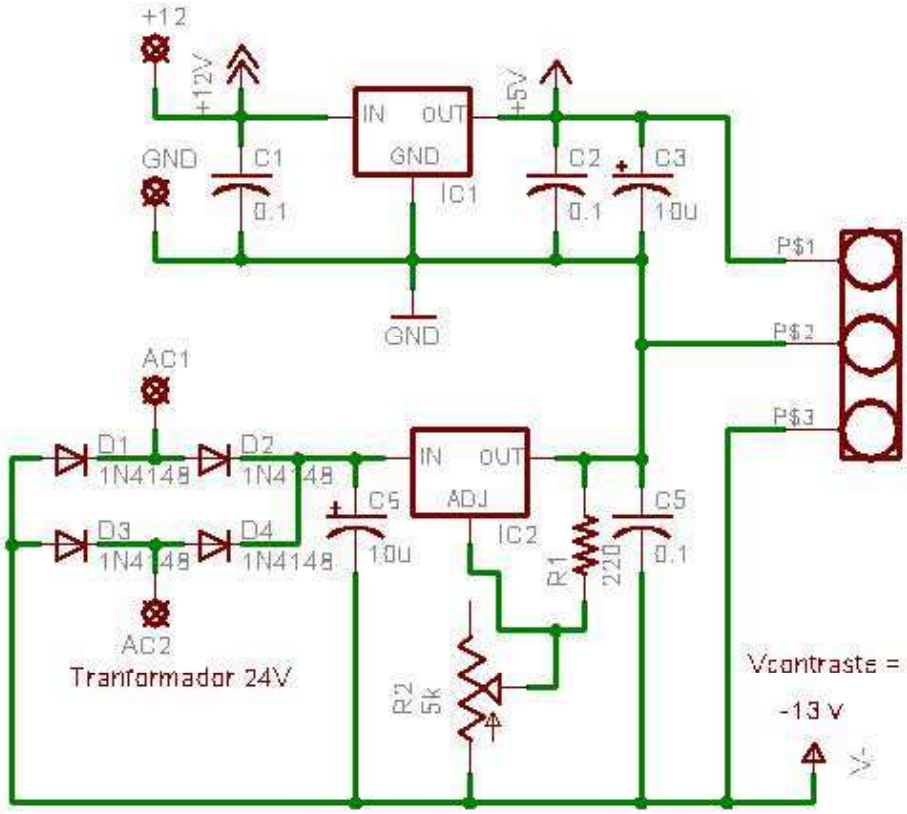

Figura 3. 55 Diagrama de conexiones de la fuente de -13 volts con la de 5 volts.

3.8.3. La fuente de 750 V de CA

La fuente de alimentación del backlite fue obtenida de un scanner dañado, simplemente se alimentó de una fuente de 12 volts y la salida se conectó directamente al modulo de la pantalla grafica LCD. La figura 3.56 muestra el backlite.

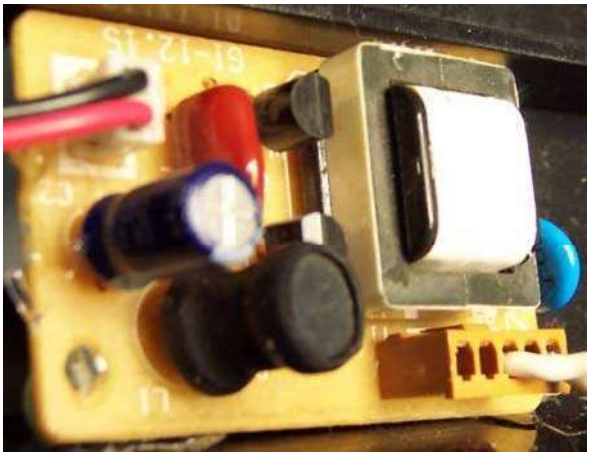

Figura 3. 56 Backlite montado en el tablero.

Este modulo proporciona el voltaje adecuado para el funcionamiento de la lámpara proporcionando 750 V_{ac} a una frecuencia de 35 KHz.

3.8.4. Programación de la memoria EEPROM

El Software IC-PROG

El software IC-PROG permite programar gran cantidad de dispositivos electrónicos, entre ellos microcontroladores y memorias, además de ser un software de fácil uso soporta el programador Propic2 disponible en el laboratorio. A pesar que el software no soporta la memoria que se utiliza, el uso de un protocolo genérico como el I^2C en memorias hace que sea posible programarla para almacenar la información de las graficas. La figura 3.57 muestra una captura de pantalla del software antes mencionado.

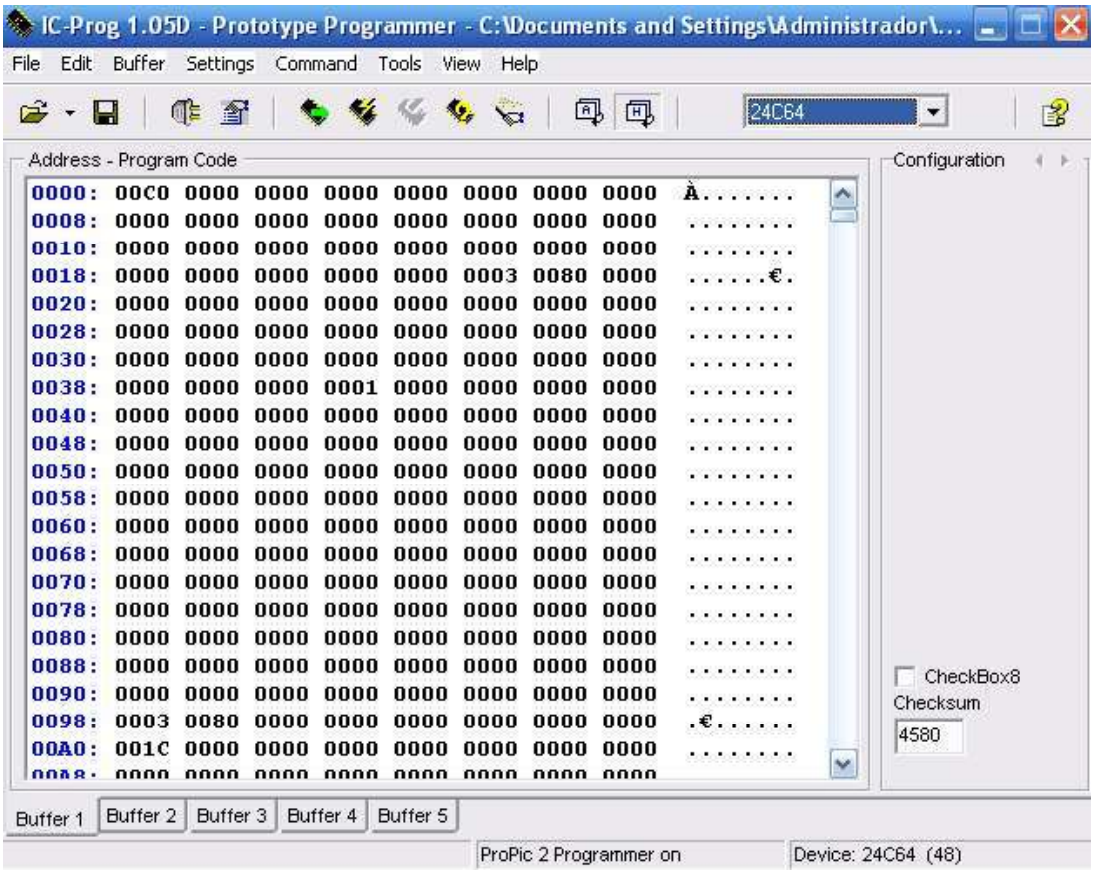

Figura 3. 57 Programa IC-PROG.

El Programador ProPic2

El programador ProPic2 usa el puerto paralelo para hacer la interconexión con el software IC-PROG y es utilizado para almacenar las graficas y datos en la memoria utilizada. La figura 3.58 muestra el programador utilizado.

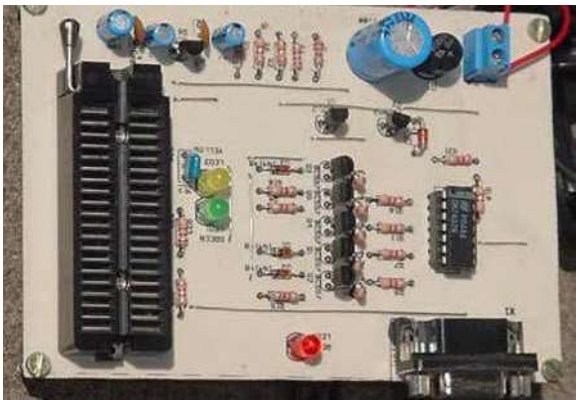

Figura 3. 58 Programador PROPIC2.

3.9 Circuitos impresos elaborados

Las siguientes imágenes fueron tomadas durante el desarrollo del proyecto se presentan: los circuitos desarrollados y la pantalla instalada.

Circuito de control se observa que contiene el microcontrolador, el conector de terminales de la pantalla grafica LCD, dos conectores de tres terminales para distribución de las señales de hacia el circuito de control de tensión y las terminales del teclado. La figura 3.59 ilustra los diseños del circuito impreso de control y la figura 3.60 muestra el circuito impreso instalado en la escaladora.

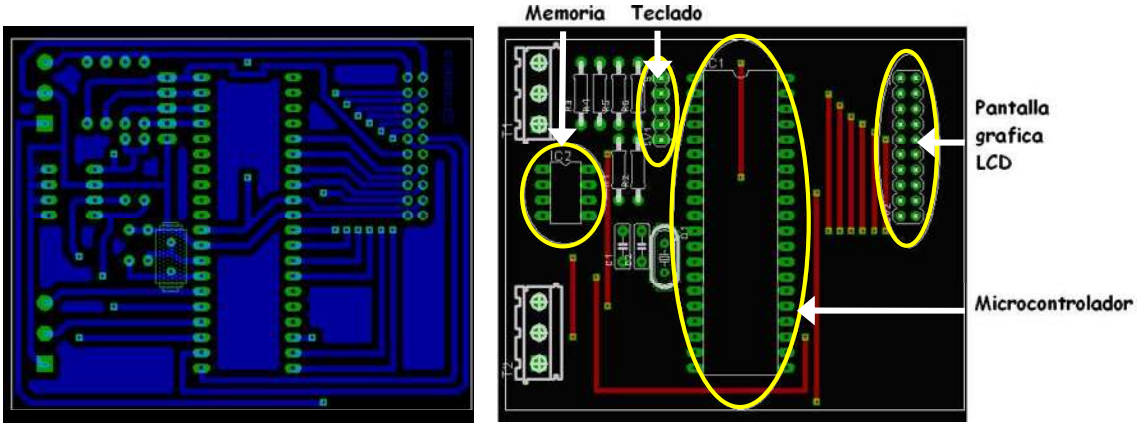

Figura 3. 59 Diseños del circuito impreso para el control.

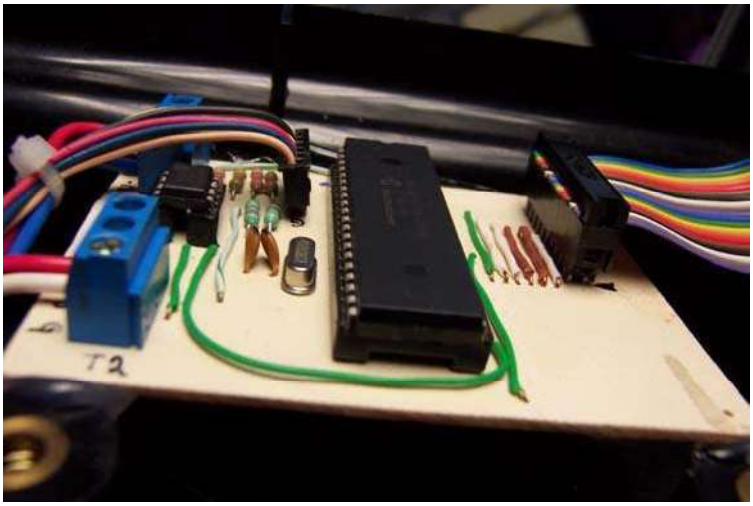

Figura 3. 60 Circuito impreso con componentes para el control.

Control de tensión de la escaladora

En este circuito se encuentran las fuentes de 5 volts y -13.5 volts, la alimentación del backlite y la alimentación del alternador. Además recibe la señal del encoder y la dirige al microcontrolador para medir la velocidad. La figura 3.61 muestra los diseños del circuito de control de potencia.

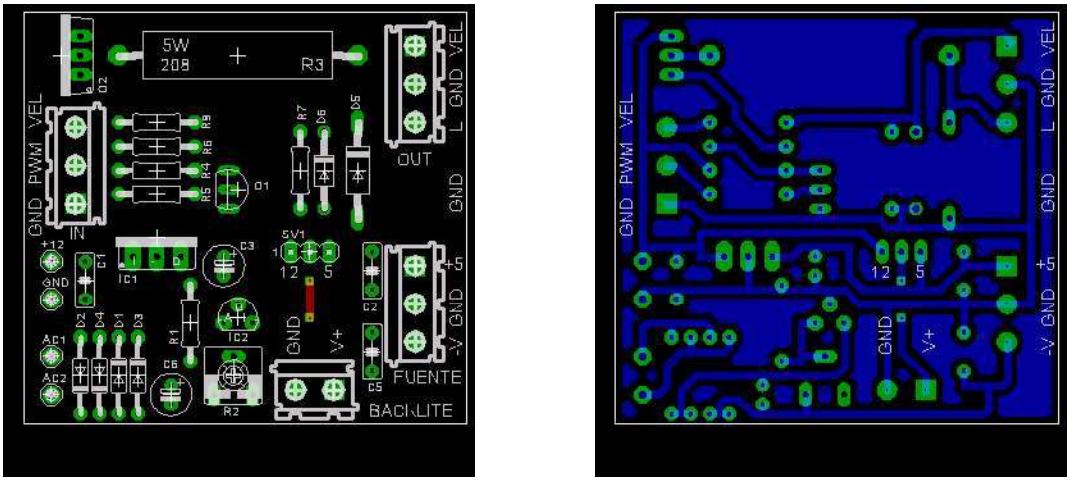

Figura 3. 61 Diseños del circuito impreso para el control de potencia.

En la figura 3.62 se observa el selector de voltaje que alimenta el backlite con 12V.

Ambos circuitos impresos se colocaron en el tablero de control de manera que permiten realizar el control de la escaladora.

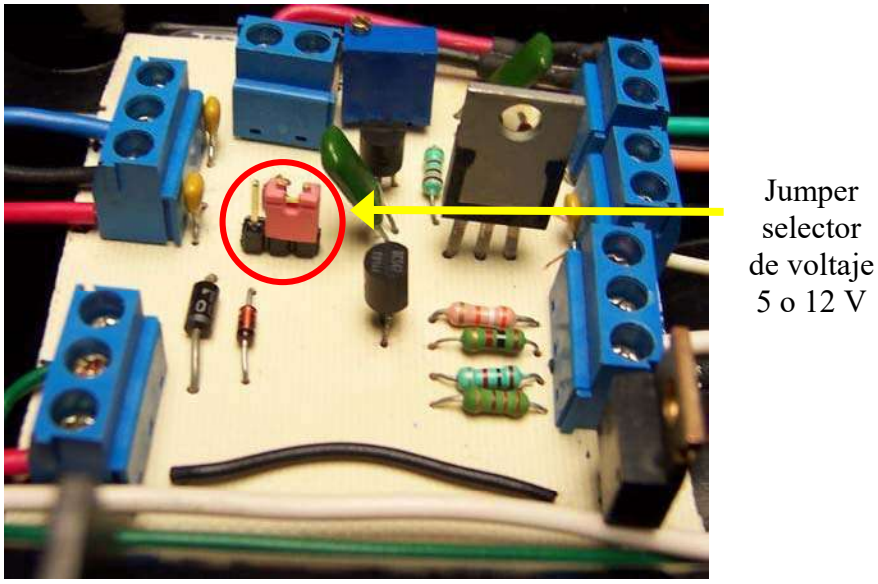

Figura 3. 62 Circuito impreso con componentes para el control de potencia.

El total de la tarea de monitoreo y control lo realiza el microcontrolador, esto permite tener pocos componentes en las placas de los circuitos impresos. La reutilización del tablero de control permite tener una vista agradable al usuario pero requiere de adaptar los componentes diseñados al tamaño del tablero.

Instalación de los circuitos en el tablero.

En la figura 3.63 se muestra la distribución de los circuitos dentro del tablero, se observa en la parte derecha el circuito de control de potencia, en la parte media el circuito del backlite y en la parte izquierda el circuito de control.

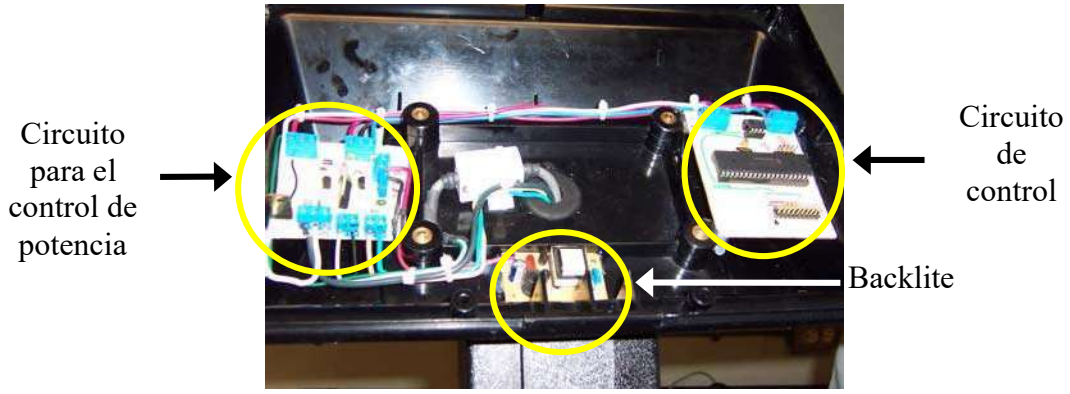

Figura 3. 63 Colocación de los circuitos en el tablero.

El circuito del backlite se colocó lo más alejado de los otros dos debido a que su alto voltaje puede provocar un arco eléctrico, lo cual ocasiona daños en los circuitos.

Colocación de la pantalla en el tablero

Para la colocación de la pantalla grafica LCD fue necesario quitar las pantallas numéricas LCD, además de abrir la cavidad necesaria para colocar el display, se coloco un acrílico transparente debido a que los píxeles de la pantalla pueden resultar dañados al ser la pantalla grafica LCD tocada o golpeada. En la figura 3.64 se muestra el antes y el después del tablero de la escaladora.

Figura 3. 64 Pantalla original y colocación de la pantalla gráfica en el tablero.

Para finalizar se observa como quedo montada la pantalla grafica LCD en el tablero de la escaladora en la figura 3.65.

Figura 3. 65 Pantalla grafica LCD y control instalados.

Capítulo 4

Pruebas de Operación

Las pruebas de operación consistieron en poner en funcionamiento de manera continua, además de colocar a varias personas a probar el funcionamiento del mismo. Para determinar su confiabilidad, esta se dejó encendida más de 72 horas.

A continuación se presentan en las figuras 4.1 y 4.2 las pantallas diseñadas para la presentación de la escaladora en la pantalla grafica LCD de la escaladora.

Figura 4. 1 Pantalla de presentación.

Figura 4. 2 Pantalla autores.

Las figuras 4.3, 4.4 y 4.5 muestran las pantallas diseñadas para la configuración de la rutina a realizar. En la figura 4.3 el menú de perfil de trabajo cuenta con un recuadro alrededor de la opción a seleccionar.

Figura 4. 1 Pantalla de selección de rutina.

En la figura 4.4 el menú de selección de intensidad cuenta con rectángulos que se van rellenando de color conforme aumenta el nivel de intensidad, en cambio si disminuye el nivel quedan los rectángulos en blanco.

Figura 4. 2 Pantalla de selección de intensidad.

En la selección de tiempo (figura 4.5), la selección de tiempo esta auxiliado tambien por un recuadro que rodea la opción por seleccionar.

Figura 4. 3 Pantalla de selección de tiempo de ejercicio.

4.1. Calibración

Para adecuar la tensión en los pedales se estableció de acuerdo a un rango de peso del usuario entre 60 y 90 kilogramos, y se probó por diferentes personas en este rango y todos tuvieron buena opinión del desempeño de la escaladora. De cualquier manera si el usuario llega a seleccionar un nivel demasiado alto o bajo, tiene la oportunidad de modificar la tensión en cualquier momento de la rutina, esto se realiza utilizando las flechas  y  o bien se puede terminar el programa con la tecla  .

En la figura 4.6 se muestra como se posiciona una persona sobre la escaladora para hacer uso de esta [1].

Las figuras 4.7, 4.8, 4.9 y 4.10 muestran a la escaladora durante el desarrollo de las rutinas, se observa que al paso del tiempo se va graficando líneas por debajo de la figura de la rutina.

Figura 4. 4 Uso de la escaladora.

La figura 4.7 refleja el uso de la escaladora con una rutina de Fat Burner, con un nivel inicial de intensidad de 8 y un tiempo de rutina de 20 minutos. El nivel de intensidad fue disminuido 3 niveles al inicio de la rutina.

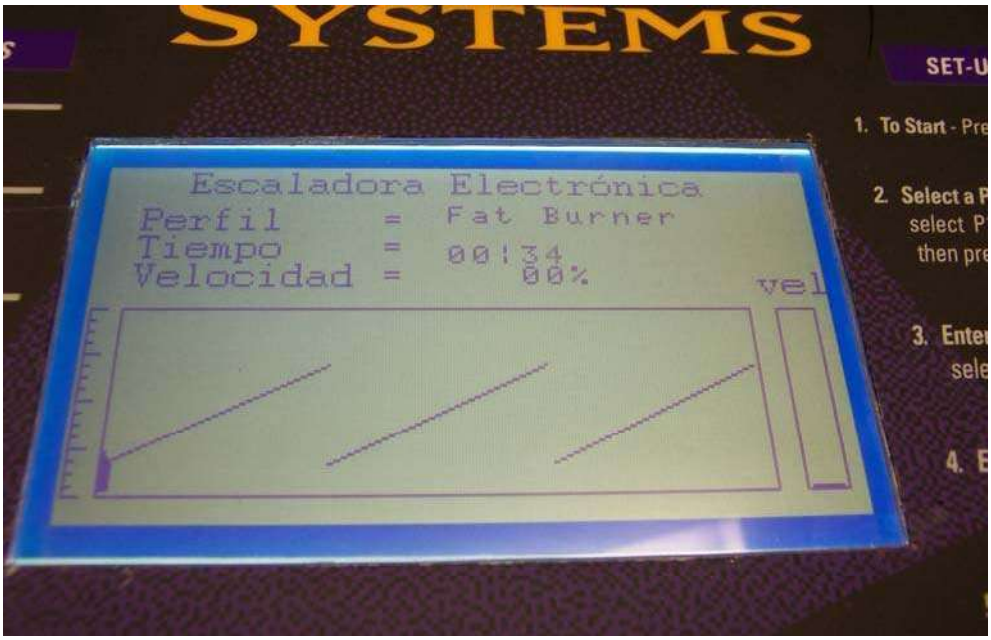

Figura 4. 5 Pantalla de ejercicio con rutina Fat Burner.

La figura 4.8 muestra la rutina manual con 5 minutos de tiempo de rutina y un nivel de intensidad 5.

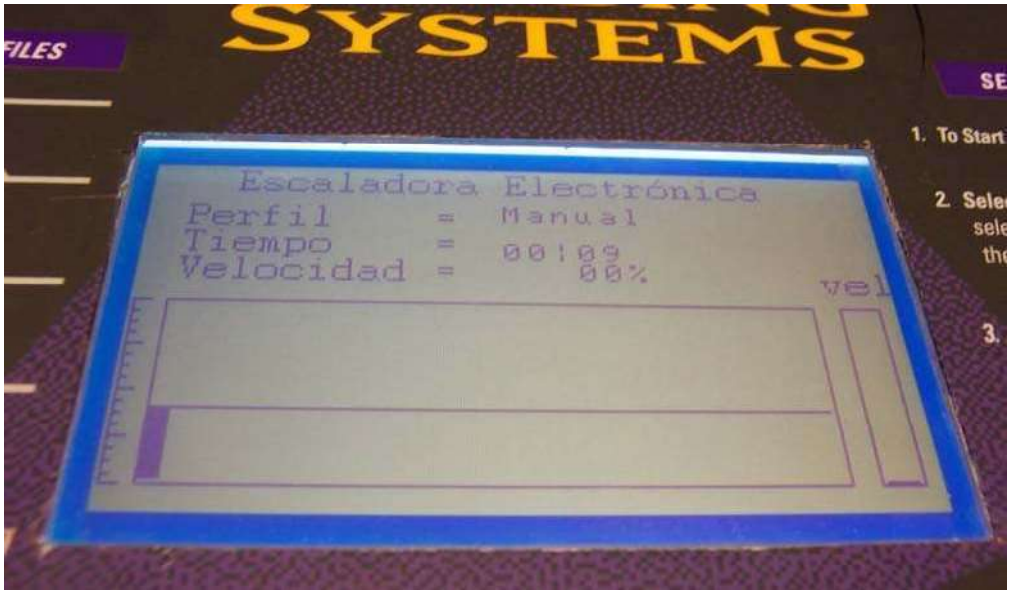

Figura 4. 6 Pantalla de ejercicio con rutina Manual.

La figura 4.9 muestra la rutina Aerobic Training configurada en un nivel de intensidad inicial 5, el cual es cambiado a 17 en los primeros segundos. El tiempo de rutina seleccionado es 5 minutos.

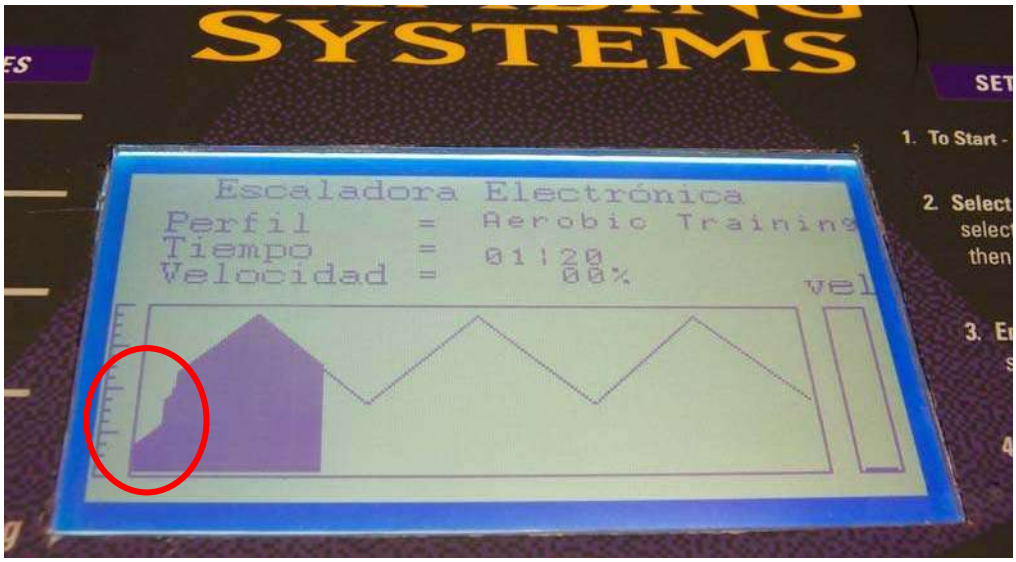

Figura 4. 7 Pantalla de ejercicio con rutina Aerobic Training.

La figura 4.10 muestra el uso de la escaladora con la rutina Steady Pace. La duración de la rutina de 5 minutos, el nivel de intensidad inicial 5 el cual es incrementado a 11.

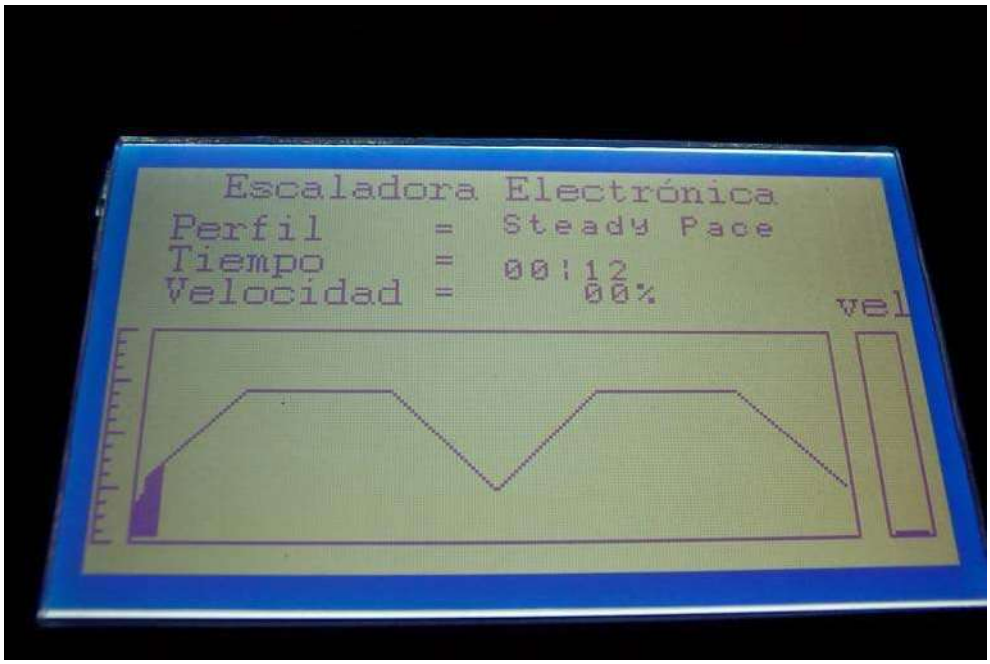

Figura 4. 8 Pantalla de ejercicio con rutina Steady Pace.

4.2. Manual de operación

Al comenzar a operar se despliega una pantalla inicial haciendo referencia la logo de UMSNH y FIE, como las pantallas graficas LCD al mantener una imagen fija son susceptibles a quemárseles los píxeles donde se encuentra la imagen. Por lo tanto, la pantalla inicial se intercambia con otra llamada autores, que se muestran en las figuras 4.11 y 4.12.

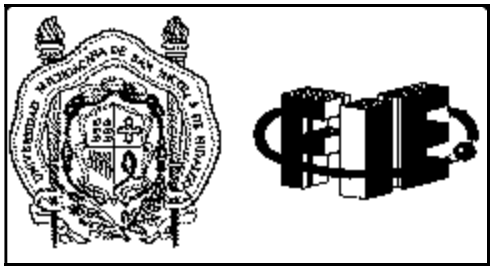

Figura 4. 9 Pantalla UMSNH y FIE.

Figura 4. 10 Pantalla autores.

Al apretar cualquier tecla, cambia a la primer pantalla que configura la rutina que se va a ejercitar, comenzando por configurar el tipo de rutina de las 4 disponibles (figura 4.13), podemos cambiar la rutina enmarcada por un rectángulo presionando la tecla de arriba o abajo según corresponda, el rectángulo se forma con las coordenadas previamente analizadas de las imágenes que se diseñaron.

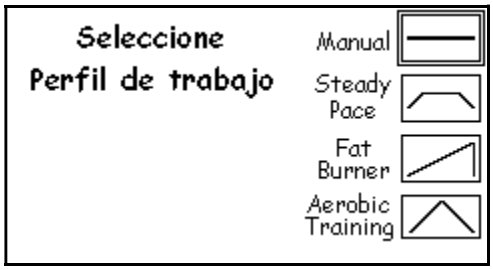

Figura 4. 11 Pantalla de selección de perfil de trabajo.

Al presionar el botón Select se guarda y pasa a la pantalla de selección de la intensidad del ejercicio.

En el software original de la escaladora se tienen 20 niveles diferentes de intensidad del ejercicio, debido a que la pantalla estaba limitada por el número de píxeles a lo alto y como nuestra interfaz tiene que ser amigable y tener buena presentación, por tales motivos, se decidió establecer la variable de NP (Número de pasos) en 17 niveles, la pantalla de selección de nivel de intensidad se muestra en la figura 4.14.

Figura 4. 12 Pantalla de selección de perfil de trabajo (Nivel 5).

De igual forma con las flechas podemos graduar el nivel de intensidad al presiona arriba tensionaremos mas los pedales, y con la tecla abajo reduciremos el nivel de tensión. Se tomó en cuenta para fijar un nivel de tensión predeterminado, una persona de peso promedio de 75 kg, ya que se puede perder la noción de la intensidad haciendo el trabajo para uno mismo. Después de elegir el nivel deseado se continúa con la pantalla de tiempo mostrada en la figura 4.15.

Figura 4. 13 Pantalla de selección de tiempo (5 minutos).

Al igual que la pantalla de selección de tipo de rutina, esta pantalla muestra de nuevo un recuadro con el que el usuario identifica que selección corresponde a la duración de la rutina.

Capitulo 5

Conclusiones y Trabajos Futuros

5.1. Conclusiones

Tras finalizar el trabajo se cumplieron los tres objetivos propuestos en un principio. En primer lugar se cumplió la realización del diseño y la construcción de un sistema de control y monitoreo para la escaladora. La interfaz fue agradable y fácil de entender y usar por los usuarios. Las instrucciones en pantalla son suficientes para la configuración de la rutina de trabajo y se puede variar la tensión en los pedales en cualquier momento.

Aunque en el desarrollo de aplicaciones basadas en microcontroladores los retos son diversos, en este caso el uso de una pantalla gráfica LCD demandó una gran cantidad de esfuerzo para comprender el funcionamiento y operación de la misma. Sin embargo los resultados fueron satisfactorios. Por otra parte el uso de un alternador como freno dinámico presenta una aplicación a estos dispositivos que fueron creados para aplicaciones específicas de generación. En este resumen es prácticamente imposible colocar cada uno de los detalles en el desarrollo de este sistema, sin embargo, se presenta una idea de la construcción y planteamiento del problema.

Cumplir con un trabajo disponible, y ver el resultado de los alcances que se pueden lograr con las bases inculcadas en la facultad motiva a seguir trabajando en cualquier otro proyecto que llegara a estar disponible para desarrollarlo.

Recordando la facultad, al ver microcontroladores primero se hicieron ejercicios en lenguaje ensamblador y usando el mplab 6. Para la realización del proyecto se utilizó el mplab 7.5 y compilador en C.

Algo muy significativo, es realizar una mejor interpretación de las hojas de datos de los fabricantes, el vocabulario técnico de las especificaciones, el uso de un editor hexadecimal, el manejo del programa IC-PROG y del programador PRO-PIC.

Lo más importante fue aprender a valorar el trabajo que se realizó, y estar dispuesto a enfrentar los retos con las mismas ganas.

El planteamiento inicial fue teniendo en cuenta el microcontrolador y la pantalla grafica LCD. Las especificaciones expuestas por el fabricante se pueden considerar un poco confusas porque no hubo trabajos previos sobre los cuales uno podría basarse.

Debido a que lo realizado tenía que tener armonía con el diseño original de la escaladora, se cuidó mucho la estética en la colocación de la pantalla grafica LCD y los circuitos.

En un principio para dar un ejemplo, se pensó en usar la memoria rom del microcontrolador para guardar los datos de los perfiles del ejercicio, sin embargo, eso nos ocupaba la mayor parte de la memoria disponible, y sin tener pantallas de presentación para la elección de los elementos. Por lo tanto se decidió hacer la modificación y colocar una memoria EEPROM externa, obteniendo seis veces más almacenamiento que la disponible en el microcontrolador.

Faltó probar más el equipo se utilizaron compañeros que llegaban al laboratorio, pero nunca se le dio un uso excesivo como podría ser usado en un gimnasio.

Aunque lo más importante es que a los compañeros les pareció un buen trabajo, sin embargo, como no fue posible darle seguimiento a la opinión de usuarios expertos al desempeño que ofrece el nuevo diseño no es fácil visualizar como se podría mejorar más.

5.2. Trabajos futuros

Lo desarrollado en este trabajo se puede utilizar tanto para mejorar el diseño del mismo, como para utilizar partes del mismo para crear cosas diferentes.

Para el mejoramiento del control de la escaladora una mejora es la inclusión en la pantalla de trabajo de la información de escalones y pisos subidos; y calorías para que se asemeje con exactitud al diseño original. Otra mejora que podría aplicarse es que al momento de subirse el usuario, la escaladora mantenga los escalones firmes para mantener al usuario cómodamente en lo que configura la rutina de ejercicio.

Para el uso de las partes desarrolladas, se puede pensar en el uso de la memoria para acceso de datos, incluso desarrollar las funciones para escritura para tomar mediciones de cualquier tipo. Para la pantalla se podrían desarrollar osciloscopios o aplicaciones donde se necesite una interfaz visual agradable.

El limite es la imaginación, solo se necesita tiempo y esfuerzo y seguramente este trabajo formará parte de otros en el futuro.

Apéndice. Listado del programa principal

```
/*********************************************************/
/* Programa para controlar el display gráfico AND 1741     */
/* El bus de datos esta en el puerto D del 16F877A  */
/*      Pin                              */
/*      11-18   D0-D7 a D0-D7            */
/*      5               WR a RB0                */
/*      6               RD a RB1                */
/*      7               CE a RB2                */
/*      8               CD a RB3                */
/*                                       */
/*                      Conexiones adicionales*/
/*                                       */
/*      3               VDD = +5                */
/*      2               GND = 0 Volts  */
/*      4               VEE = VDD-VEE = 18.5 = -13.5 a tierra              */
/*      19              FS = +5V(font 6*8) o Gnd(font 8*8)                 */
/*      20              RV = +5V(positivo) o gnd(Inverso)                  */
/*      10              Rst = +5V(reset a cero)                            */
/*                                       */
/*********************************************************/

#include <pic.h>
#include "delay.c"
#include "LCD_Graf.h"
#include "i2c.h"
#include "pwm.h"
#include "Teclado.h"
#include "Delay1.c"

unsigned char ConvASCII(unsigned char);
void EnCadena(void);
void LetreroFin(void);
void Letrero(unsigned char);
void Print_Page(unsigned int);
void Forma(unsigned char,unsigned char,unsigned char, unsigned char);
void ConfiguraTMR0(void);
void interrupt Tiempo (void);
void bin2ASCII(unsigned char);
void ByteporByte(unsigned char, unsigned char);
void selectminutos(unsigned char);
void selectmenu(unsigned char);
void selectintensidad(unsigned char);
void Vel_Porcentaje(unsigned int);

unsigned char sobref_TMR0;
unsigned char MedioSegundo;
unsigned int TiempoImpresion;
unsigned char baseTiempo;
unsigned char Segundos;
unsigned char Minutos;
unsigned char UM;
unsigned char DM;
unsigned int AddressM;
unsigned char aux;
unsigned char U;
unsigned char minutos;
unsigned int Velocidad;

main()
{
unsigned char i,Bandera,adrh,adrl;
unsigned int addr;
unsigned char tecla;
unsigned char Felegida;
unsigned char tmp;
unsigned char menu;
unsigned int Address;
```

```
unsigned char j;
unsigned char Velactual;
unsigned char NP;
unsigned int tiempo;
unsigned char salir;
        TRISC1=0; //ENERGIZA EEPROM
        RC1=0;
        DelayMs(50);
        RC1=1;

        SetUp();
        CleanText();
        CleanGraphics();

        init_i2c();
        ConfiguraPWM();
        Conf_PORTA();
        SalidaPWM(0);
        RC0=1;
        T1CON=0x07;
        ConfiguraTMR0();

        i2c_read(NOACK);
        init_i2c();
        i2c_stop();

        Command(0x9C);


        i=0;
        addr = 0;
        Bandera = 0;
        // Imprime Escudos
        Print_Page(0x0000);           // Pantalla
        Data(0x00);                           // Menos significativo
        Data(0x0F);                           // Más significativo
        Command(0x24);                // Inicio de la pagina grafica de LCD
        Command(0xB0);                // Modo Auto

        // Carga a memoria de display la pagina 2
        Address=0x1000;
        for(i=0;i<128;i++)
        {
                Set_Address(Address);
                i2c_repStart();
                i2c_write(READ_EE);
                for(j=0;j<29;j++)                     // Debe de ser menos uno para
                {                                     // leer el ultimo con NOACK
                        DataAuto(i2c_read(ACK));
                }
                DataAuto(i2c_read(NOACK));    // Lee el ultimo dato sin ACK;
                        // Imprime el ultimo dato
                i2c_stop();                                   // Fin de lectura
                Address = Address + 30;               // Lee la siguiente renglón
        }
        Command(0xB2);

        // scroll de la pantalla

        i=0;
        Bandera=0;
        while(1)
        {
                tecla=Teclado();
                if(Bandera == 0)
                {
                        adrh = addr >> 8;
                        adrl = addr & 0x00FF;
                        Data(adrl);           // coloca en el inicio de los gráficos
                        Data(adrh);
```

```
                Command(0x42);          // ejecuta el comando
                DelayMs(30);
                        if(tecla!=0)
                        {
                                salir=1;
                                break;
                        }
                i++;
                addr = addr + 30;
                if(i==128)
                {
                        Bandera = 1;
                        i=0;
                        for(tiempo=0;tiempo<2000;tiempo++)
                        {
                                DelayMs(1);
                                if(tecla!=0)
                                {
                                        salir=1;
                                        break;
                                }
                        }
                }
        }
        else
        {
                adrh = addr >> 8;
                adrl = addr & 0x00FF;
                Data(adrl);             // coloca en el inicio de los gráficos
                Data(adrh);
                Command(0x42);          // ejecuta el comando
                DelayMs(30);
                        if(tecla!=0)
                        {
                                salir=1;
                                break;
                        }
                i--;
                addr = addr - 30;
                if(i==128)
                {
                        Bandera = 0;
                        i=0;
                        for(tiempo=0;tiempo<2000;tiempo++)
                        {
                                DelayMs(1);
                                if(tecla!=0)
                                {
                                        salir=1;
                                        break;
                                }
                        }
                }
        }
        if(salir==1) break;
}


CleanGraphics();

// Imprime Menu Forma

Print_Page(0x2000);
menu=1;
selectmenu(menu);
        while(1)
        {
                tecla = Teclado();

                if(tecla == INCREMENTA)
```

```
                {
                menu--;
                if(menu<1) menu=4;
                selectmenu(menu);
                }
                if(tecla == DECREMENTA)
                {
                menu++;
                if(menu>4) menu=1;
                selectmenu(menu);
                }
                if(tecla == SELECT)
                {
                Felegida=menu;
                break;
                }
        }


// Imprime menú intensidad

Print_Page(0x3000);
menu=1;
selectintensidad(menu);
        while(1)
        {
                tecla = Teclado();

                if(tecla == INCREMENTA)
                {
                menu++;
                if(menu>17) menu=17;
                selectintensidad(menu);
                }
                if(tecla == DECREMENTA)
                {
                menu--;
                if(menu<1) menu=1;
                selectintensidad(menu);
                }
                if(tecla == SELECT)
                {
                NP=menu;
                break;
                }
        }

// Imprime menu tiempo

Print_Page(0x4000);
menu=1;
selectmenu(menu);
        while(1)
        {
                tecla = Teclado();

                if(tecla == INCREMENTA)
                {
                menu--;
                if(menu<1) menu=4;
                selectmenu(menu);
                }
                if(tecla == DECREMENTA)
                {
                menu++;
                if(menu>4) menu=1;
                selectmenu(menu);
                }
                if(tecla == SELECT)
                {
```

```
                        minutos=menu;
                        break;
                        }
                }
                Print_Page(0x5000);
                Forma(NP,1,Felegida, 0);
                selectminutos(minutos);
                Letrero(Felegida);
        // Parte para introducir el tiempo

                Segundos=0;
                Minutos=0;
                tmp=0;
//              tmp2=0;
                U=0;
                TiempoImpresion=0;
                baseTiempo=0;

                Set_Address(AddressM);
                i2c_repStart();
                i2c_write(READ_EE);

                Velactual=0;
                Velocidad=0;

                while(1)
                {
                                EnCadena();
                                tecla = Teclado();
                                if(tecla == INCREMENTA)
                                {
                                        GIE=0;
                                        T0CS=1;
                                        i2c_read(NOACK);
                                        i2c_stop();

                                        Forma(NP,0, Felegida,tmp);
                                        NP++;
                                        if(NP >17) NP = 17;
                                        Forma(NP,1,Felegida,tmp);
                                        Set_Address(AddressM);
                                        i2c_repStart();
                                        i2c_write(READ_EE);
                                        T0CS=0;
                                        GIE=1;
                                }
                                if(tecla == DECREMENTA)
                                {
                                        GIE=0;
                                        T0CS=1;
                                        i2c_read(NOACK);
                                        i2c_stop();

                                        Forma(NP,0, Felegida,tmp);
                                        NP--;
                                        if(NP < 1) NP = 1;
                                        Forma(NP,1, Felegida,tmp);
                                        Set_Address(AddressM);
                                        i2c_repStart();
                                        i2c_write(READ_EE);
                                        T0CS=0;
                                        GIE=1;
                                }
                                if( TiempoImpresion > 0)
                                {
                                        GIE=0;
                                        T0CS=1;
                                        ByteporByte(tmp,NP);
                                        TiempoImpresion=0;
                                        tmp++;
```

```
                                        if (Felegida==1)
                                        {
                                                SalidaPWM(U+(NP*3));
                                        }
                                        else
                                        {
                                                SalidaPWM(U+NP);
                                        }
                                        Cuadro(224,10,233, (Velactual+10),0);
                                        Cuadro(224,10,233, (Velocidad+10),1);
                                        Cuadro(226,10,231, (Velactual+10),0);
                                        Cuadro(226,10,231, (Velocidad+10),1);
                                        Cuadro(228,10,229, (Velactual+10),0);
                                        Cuadro(228,10,229, (Velocidad+10),1);
                                        Velactual=Velocidad;
                                        Vel_Porcentaje(Velactual);
                                        T0CS=0;
                                        GIE=1;
                                }

                                if( Minutos > aux )
                                {
                                EnCadena();
                                ByteporByte(tmp, NP);
                                break;
                                }

                }
                i2c_read(NOACK);
                i2c_stop();
                CleanText();
                CleanGraphics();
                LetreroFin();
                CleanText();
                CleanGraphics();
                Print_Page(0x8000);
                Delay1s(3);
                SalidaPWM(0);
}

void Forma(unsigned char NP, unsigned char Imprime, unsigned char Felegida, unsigned char
inicio)
{
unsigned char puntos,l;

        if(Felegida==1)
        {
                AddressM = 0x7000+(int)inicio;
        }
        else if(Felegida==2)
        {
                AddressM = 0x70C8+(int)inicio;
        }
        else if(Felegida==3)
        {
                AddressM = 0x7190+(int)inicio;
        }
        else if(Felegida==4)
        {
                AddressM = 0x7258+(int)inicio;
        }

        Set_Address(AddressM);
        i2c_repStart();
        i2c_write(READ_EE);
        l=i2c_read(ACK);

        for(puntos=inicio;puntos<199;puntos++)
        {
                l=i2c_read(ACK);
```

```
                if(Imprime==1)
                {
                        SetPixel((13+puntos),(l-32+NP));
                }

                if(Imprime==0)
                {
                        ClearPixel((13+puntos),(l-32+NP));
                }
        // Lee el ultimo dato sin ACK;                                          // Imprime el
ultimo dato
                                        // Fin de lectura
        }
        DataAuto(i2c_read(NOACK));
        i2c_stop();
        Command(0xB2);
}

void EnCadena(void)
{
        Data(0x86);
        Data(0x1E);

    Command(0x24);
        Command(0xB0);
        bin2ASCII(Minutos);
        DataAuto(UM);
        DataAuto(DM);
        DataAuto(0x5C);
        bin2ASCII(Segundos);
        DataAuto(UM);
        DataAuto(DM);
        Command(0xB2);

        Data(0x00);
        Data(0x00);
        Command(0x9C);
}

void LetreroFin(void)
{

        unsigned int tiempo;
        Data(0xF5);
        Data(0x1E);

        Command(0x24);
        Command(0xB0);

        ConvASCII('R');
        ConvASCII('u');
        ConvASCII('t');
        ConvASCII('i');
        ConvASCII('n');
        ConvASCII('a');
        ConvASCII(' ');
        ConvASCII('f');
        ConvASCII('i');
        ConvASCII('n');
        ConvASCII('a');
        ConvASCII('l');
        ConvASCII('i');
        ConvASCII('z');
        ConvASCII('a');
        ConvASCII('d');
        ConvASCII('a');


        Command(0xB2);
```

```
        Data(0x00);
        Data(0x00);

        for(tiempo=0;tiempo<5000;tiempo++)
        {
                DelayMs(1);
        }

        Command(0x9C);

}

unsigned char ConvASCII(unsigned char L)
{
        L=L-0x20;
        DataAuto(L);
        return 0;
}


void bin2ASCII(unsigned char L)
{
        UM=(L/10)+0x10;
        DM=(L%10)+0x10;
}

void Print_Page(unsigned int Address)
{
unsigned char i,j;

        // lee 1 página de 32 * 128

        Data(0x00);                         // Menos significativo
        Data(0x00);                         // Más significativo
        Command(0x24);              // Inicio de la pagina grafica de LCD

        Command(0xB0);              // Modo Auto


        for(i=0;i<128;i++)
        {
                Set_Address(Address);
                i2c_repStart();
                i2c_write(READ_EE);
                for(j=0;j<29;j++)                   // Debe de ser menos uno para
                {                                                       // leer el ultimo con
NOACK
                        DataAuto(i2c_read(ACK));
                }
                DataAuto(i2c_read(NOACK));    // Lee el ultimo dato sin ACK;
                        // Imprime el ultimo dato
                i2c_stop();                                     // Fin de lectura
                Address = Address + 30;             // Lee la siguiente renglon
        }
        Command(0xB2);
}



void ConfiguraTMR0(void)
        {
        OPTION=0b11010111;
        T0IE=1;
        GIE=1;
        }

void interrupt Tiempo (void)
{
        if(T0IF==1)
        {
```

```
        T0IF=0;
        sobref_TMR0++;
                if(sobref_TMR0==38)
                {
                        sobref_TMR0 = 0;
                        baseTiempo++;          // cada 1/2 seg
                        MedioSegundo++;
                        if(MedioSegundo==2)
                        {
                                Velocidad=TMR1;
                                TMR1=0;
                                if (Velocidad > 976)  Velocidad = 976;

                                Velocidad = Velocidad >> 4;

                                Segundos++;
                                MedioSegundo=0;

                                if(Segundos>59)
                                {
                                        Segundos=0;
                                        Minutos++;
                                }
                        }

                        if(minutos==1)
                        {
                                if(baseTiempo==3)
                                {
                                        TiempoImpresion++; //Cada 1.5 Seg.
                                        baseTiempo=0;
                                }
                        }

                        if(minutos==2)
                        {
                                if(baseTiempo==6)
                                {
                                        TiempoImpresion++; //Cada 3 Seg.
                                        baseTiempo=0;
                                }
                        }

                        if(minutos==3)
                        {
                                if(baseTiempo==12)
                                {
                                        TiempoImpresion++; //Cada 1.5 Seg.
                                        baseTiempo=0;
                                }
                        }

                        if(minutos==4)
                        {
                                if(baseTiempo==18)
                                {
                                        TiempoImpresion++; //Cada 3 Seg.
                                        baseTiempo=0;
                                }
                        }
                }
        }
}

void ByteporByte(unsigned char tmp,unsigned char NP)
{
                                        U=i2c_read(ACK);
                                        LineaVertical(11,(U+(NP)-32),(13+tmp),1);
}
```

```
void selectminutos(unsigned char tiempo)
{

        Data(0x6B);
        Data(0x10);

        Command(0x24);

        Command(0xB0);

        if(minutos==1)
        {
                DataAuto(0x10);
                DataAuto(0x15);
                aux=4;
        }
        if(minutos==2)
        {
                DataAuto(0x11);
                DataAuto(0x10);
                aux=9;
        }
        if(minutos==3)
        {
                DataAuto(0x12);
                DataAuto(0x10);
                aux=19;
        }
        if(minutos==4)
        {
                DataAuto(0x13);
                DataAuto(0x10);
                aux=29;
        }

        DataAuto(0x5C);
        DataAuto(0x10);
        DataAuto(0x10);

        Command(0xB2);

        Data(0x00);
        Data(0x00);

        Command(0x9C);

}

void selectmenu(unsigned char menu)
{

        if(menu==1)
        {
        Cuadro(191,6,236,33,0);
        Cuadro(192,7,235,32,0);
        Cuadro(191,64,236,93,0);
        Cuadro(192,65,235,92,0);
        Cuadro(191,94,236,123,1);
        Cuadro(192,95,235,122,1);
        }
        if(menu==2)
        {
        Cuadro(191,94,236,123,0);
        Cuadro(192,95,235,122,0);
        Cuadro(191,34,236,63,0);
        Cuadro(192,35,235,62,0);
        Cuadro(191,64,236,93,1);
        Cuadro(192,65,235,92,1);
        }
        if(menu==3)
```

```
        {
        Cuadro(191,64,236,93,0);
        Cuadro(192,65,235,92,0);
        Cuadro(191,6,236,33,0);
        Cuadro(192,7,235,32,0);
        Cuadro(191,34,236,63,1);
        Cuadro(192,35,235,62,1);
        }
        if(menu==4)
        {
        Cuadro(191,34,236,63,0);
        Cuadro(192,35,235,62,0);
        Cuadro(191,94,236,123,0);
        Cuadro(192,95,235,122,0);
        Cuadro(191,6,236,33,1);
        Cuadro(192,7,235,32,1);
        }

}

void selectintensidad(unsigned char menu)
{

        if(menu==1)
        {
        LineaHorizontal(194,231,21,0);
        LineaHorizontal(194,231,22,0);
        LineaHorizontal(194,231,15,1);
        LineaHorizontal(194,231,16,1);
        }
        if(menu==2)
        {
        LineaHorizontal(194,231,27,0);
        LineaHorizontal(194,231,28,0);
        LineaHorizontal(194,231,21,1);
        LineaHorizontal(194,231,22,1);
        }
        if(menu==3)
        {
        LineaHorizontal(194,231,33,0);
        LineaHorizontal(194,231,34,0);
        LineaHorizontal(194,231,27,1);
        LineaHorizontal(194,231,28,1);
        }
        if(menu==4)
        {
        LineaHorizontal(194,231,39,0);
        LineaHorizontal(194,231,40,0);
        LineaHorizontal(194,231,33,1);
        LineaHorizontal(194,231,34,1);
        }
        if(menu==5)
        {
        LineaHorizontal(194,231,45,0);
        LineaHorizontal(194,231,46,0);
        LineaHorizontal(194,231,39,1);
        LineaHorizontal(194,231,40,1);
        }
        if(menu==6)
        {
        LineaHorizontal(194,231,51,0);
        LineaHorizontal(194,231,52,0);
        LineaHorizontal(194,231,45,1);
        LineaHorizontal(194,231,46,1);
        }
        if(menu==7)
        {
        LineaHorizontal(194,231,57,0);
        LineaHorizontal(194,231,58,0);
        LineaHorizontal(194,231,51,1);
```

```
LineaHorizontal(194,231,52,1);
}
if(menu==8)
{
LineaHorizontal(194,231,63,0);
LineaHorizontal(194,231,64,0);
LineaHorizontal(194,231,57,1);
LineaHorizontal(194,231,58,1);
}
if(menu==9)
{
LineaHorizontal(194,231,69,0);
LineaHorizontal(194,231,70,0);
LineaHorizontal(194,231,63,1);
LineaHorizontal(194,231,64,1);
}
if(menu==10)
{
LineaHorizontal(194,231,75,0);
LineaHorizontal(194,231,76,0);
LineaHorizontal(194,231,69,1);
LineaHorizontal(194,231,70,1);
}
if(menu==11)
{
LineaHorizontal(194,231,81,0);
LineaHorizontal(194,231,82,0);
LineaHorizontal(194,231,75,1);
LineaHorizontal(194,231,76,1);
}
if(menu==12)
{
LineaHorizontal(194,231,87,0);
LineaHorizontal(194,231,88,0);
LineaHorizontal(194,231,81,1);
LineaHorizontal(194,231,82,1);
}
if(menu==13)
{
LineaHorizontal(194,231,93,0);
LineaHorizontal(194,231,94,0);
LineaHorizontal(194,231,87,1);
LineaHorizontal(194,231,88,1);
}
if(menu==14)
{
LineaHorizontal(194,231,99,0);
LineaHorizontal(194,231,100,0);
LineaHorizontal(194,231,93,1);
LineaHorizontal(194,231,94,1);
}
if(menu==15)
{
LineaHorizontal(194,231,105,0);
LineaHorizontal(194,231,106,0);
LineaHorizontal(194,231,99,1);
LineaHorizontal(194,231,100,1);
}
if(menu==16)
{
LineaHorizontal(194,231,111,0);
LineaHorizontal(194,231,112,0);
LineaHorizontal(194,231,105,1);
LineaHorizontal(194,231,106,1);
}
if(menu==17)
{
LineaHorizontal(194,231,111,1);
LineaHorizontal(194,231,112,1);
}
```

```
}

void Letrero(unsigned char Forma)
{
         if(Forma==1)
        {
                Data(0x4A);
                Data(0x1E);
                Command(0x24);
                Command(0xB0);
                ConvASCII('M');
                ConvASCII('a');
                ConvASCII('n');
                ConvASCII('u');
                ConvASCII('a');
                ConvASCII('l');
                Command(0xB2);
                Data(0x00);
                Data(0x00);

        }
         if(Forma==2)
        {
                Data(0x4A);
                Data(0x1E);
                Command(0x24);
                Command(0xB0);
                ConvASCII('S');
                ConvASCII('t');
                ConvASCII('e');
                ConvASCII('a');
                ConvASCII('d');
                ConvASCII('y');
                ConvASCII(' ');
                ConvASCII('P');
                ConvASCII('a');
                ConvASCII('c');
                ConvASCII('e');
                Command(0xB2);
                Data(0x00);
                Data(0x00);

        }
         if(Forma==3)
        {
                Data(0x4A);
                Data(0x1E);
                Command(0x24);
                Command(0xB0);
                ConvASCII('F');
                ConvASCII('a');
                ConvASCII('t');
                ConvASCII(' ');
                ConvASCII('B');
                ConvASCII('u');
                ConvASCII('r');
                ConvASCII('n');
                ConvASCII('e');
                ConvASCII('r');
                Command(0xB2);
                Data(0x00);
                Data(0x00);

        }
         if(Forma==4)
        {
                Data(0x4A);
                Data(0x1E);
                Command(0x24);
```

```
                Command(0xB0);
                ConvASCII('A');
                ConvASCII('e');
                ConvASCII('r');
                ConvASCII('o');
                ConvASCII('b');
                ConvASCII('i');
                ConvASCII('c');
                ConvASCII(' ');
                ConvASCII('T');
                ConvASCII('r');
                ConvASCII('a');
                ConvASCII('i');
                ConvASCII('n');
                ConvASCII('i');
                ConvASCII('n');
                ConvASCII('g');


                Command(0xB2);
                Data(0x00);
                Data(0x00);

        }
}

void Vel_Porcentaje(unsigned int Vel_Por)
{
        if(Vel_Por > 49) Vel_Por = 49;
        Vel_Por = Vel_Por << 1;
        Data(0xA7);
        Data(0x1E);

    Command(0x24);
        Command(0xB0);
        bin2ASCII(Vel_Por);
        DataAuto(UM);
        DataAuto(DM);
        DataAuto(0x05);

        Command(0xB2);
        Data(0x00);
        Data(0x00);
        Command(0x9C);
}
```

Rutina de manejo del display LCD

```
/*****************************/
/* Driver De pantalla Grafica */
/*****************************/
void SetUp(void)
{
        PORTD = 0x00;           // Inicalizo el bus a cero
        TRISD = 0;                      // Al inicio como salida
        PORTB = 0x0F;           // Desactiva todas las lineas de control
        TRISB = 0xF0;           // Lineas de control como salidas

        Ready();

        Command(0x80);          // Modo de programacion

        Data(0x00);                     // Posicion de Texto = 0x1E00
        Data(0x1E);
        Command(0x40);          // Salto al inicio de la RAM

        Data(0x1E);                     // coloca el área de texto
        Data(0x00);
        Command(0x41);          // cuarenta caracteres
```

```
        Data(0x00);                     // coloca en el inicio de los graficos
        Data(0x00);
        Command(0x42);          // ejecuta el comando

        Data(0x1E);                     // coloca el área gráfica
        Data(0x00);
        Command(0x43);          // 40 * 6 puntos
}

void SetPixel(unsigned char x, unsigned char y)
{
unsigned int Address;
unsigned char AddrH;
unsigned char AddrL;
unsigned char Bit;

        x=x-1;
        y=y-1;
        Bit = (x%8);
        Bit = Bit-7;
        Bit = -Bit;
        Bit = Bit|0xF8;
        x=x>>3;
        Address= x+((127-y)*30);
        AddrH = Address / 0x100;
        AddrL = Address % 0x100;

        Data(AddrL);                    // LSB Posicion de Inicio de Graficos = 0x0000
        Data(AddrH);                    // MSB
        Command(0x24);                  // Inicio de la direccion de Graficos

        Command(Bit);                   // Impresion de Pixeles

}

void ClearPixel(unsigned char x, unsigned char y)
{
unsigned int Address;
unsigned char AddrH;
unsigned char AddrL;
unsigned char Bit;

        x=x-1;
        y=y-1;
        Bit = (x%8);
        Bit = Bit-7;
        Bit = -Bit;
        Bit = Bit|0xF0;
        x=x>>3;
        Address= x+((127-y)*30);
        AddrH = Address / 0x100;
        AddrL = Address % 0x100;

        Data(AddrL);                    // LSB Posicion de Inicio de Graficos = 0x0000
        Data(AddrH);                    // MSB
        Command(0x24);          // Inicio de la direccion de Graficos

        Command(Bit);                   // Impresion de Pixeles
}

void Command(unsigned char Comando)
{
        Busy();                         // Espera a que no este ocupado
        CD = 1;                         // Modo Comando
        WR = 0;                         // Activa la escritura
        CE = 0;                         // Activa el display

        BUS = Comando;          // Envía el comando
        WR = 1;                         // Deshabilita la escritura
        CE = 1;                         // Deshabilito el display
```

```
}

void Data(unsigned char Dato)
{
	Busy();				// Espera a que no este ocupado
	CD = 0;				// Modo Comando
	WR = 0;				// Activa la escritura
	CE = 0;				// Activa el display
	BUS = Dato;			// Envía el comando
	WR = 1;				// Deshabilita la escritura
	CE = 1;				// Deshabilito el display
	CD = 1;				// No es necesario !!!!!
}
void DataAuto(unsigned char Dato)
{
	bit3();				// Espera a que no este ocupado
	CD = 0;				// Modo Comando
	WR = 0;				// Activa la escritura
	CE = 0;				// Activa el display
	BUS = Dato;			// Envía el comando
	WR = 1;				// Deshabilita la escritura
	CE = 1;				// Deshabilito el display
	CD = 1;				// No es necesario !!!!!
}
unsigned char ReadStatus(void)
{
unsigned char Dato;
	TRISD = 0xFF;		// Bus de entrada
	CD = 1;				// Modo Comando
	RD = 0;				// Activa la lectura
	CE = 0;				// Activa el display
	NOP();
	Dato = BUS;			// Lee el dato y lo regresa
	RD = 1;				// Deshabilita la lectura
	CE = 1;				// Deshabilito el display
	TRISD = 0x00;		// Retorna a modo salida
	return Dato;		// retorno el dato leido
}

void bit3()
{
unsigned char x;		// Bit de estado de modo continuo
	while(1)
	{
		x=ReadStatus();
		x=x&0b00001000;
		if(x!=0) break;
	}
}

void Busy(void)
{
unsigned char Status;		// Espera a que no este ocupado
	do{
		Status = ReadStatus();// Lee el estado 8 bits
		Status = Status & 0x03;	// Solo los 2 bits LSB
	}while(Status != 0x03);		// Espera mientras no es 0x03
}

void Ready(void)
{
unsigned char Status;		// Espera a que no este ocupado
	do{
		Status = ReadStatus();// Lee el estado 8 bits
		Status = Status & 0x20;	// Solo los 2 bits LSB
	}while(Status != 0x20);		// Espera mientras no es 0x03
}

void CleanText()
{
```

```
unsigned int i;
        Data(0x00);
        Data(0x1E);
        Command(0x24);

        Command(0xB0);
        for (i=0;i<720;i++)
        {
                DataAuto(0x00);
        }
        Command(0xB2);
//      Command(0x94);          // Display ON Text
}

void CleanGraphics()
{
unsigned int i;
        Data(0x00);
        Data(0x00);
        Command(0x24);          //Trabajar Modo grafico

        Command(0xB0);          //Iniciar AutoModo
        for (i=0;i<7540;i++)            //Normal 3840
        {
                DataAuto(0x00);
        }
        Command(0x42);          //Ejecutar comando
        Command(0xB2);          // Automode OFF
        Command(0x9C);          // Display ON Text ON
}
void LineaHorizontal(unsigned char x1, unsigned char x2, unsigned char y1, unsigned char
imprime)
{
        unsigned char i;

        for(i=x1;i<=x2;i++)
        {
                if(imprime==1)
                {
                        SetPixel(i,y1);
                }
                else
                {
                        ClearPixel(i,y1);
                }
        }
}

void LineaVertical(unsigned char y1, unsigned char y2, unsigned char x1, unsigned char
imprime)
{
unsigned char i;
        for(i=y1;i<=y2;i++)
        {
                if(imprime==1)
                {
                        SetPixel(x1,i);
                }
                else
                {
                        ClearPixel(x1,i);
                }
        }
}

void Cuadro(unsigned char x1, unsigned char y1,unsigned char x2, unsigned char y2, unsigned
char imprime)
{
        if(imprime == 1)
        {
```

```
                LineaVertical(y1,y2,x1,1);
                LineaVertical(y1,y2,x2,1);
                LineaHorizontal(x1,x2,y1,1);
                LineaHorizontal(x1,x2,y2,1);
        }

        else
        {
                LineaVertical(y1,y2,x1,0);
                LineaVertical(y1,y2,x2,0);
                LineaHorizontal(x1,x2,y1,0);
                LineaHorizontal(x1,x2,y2,0);
        }
}
```

Bibliografía

[1] www.stairmaster.com

[2] www.sportsmith.net

[3] Hoja de datos del microcontrolador PIC16F877A.

[4] Notas de microcontroladores Félix Jiménez Pérez.

[5] Hoja de datos de la memoria EEPROM.

[6] Hoja de datos de la pantalla grafica LCD AND 1741.