UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

DISEÑO DE UN DISPOSITIVO DIGITAL AUXILIAR AL ENTRENAMIENTO Y A LA PRÁCTICA DEL CICLISMO EMPLEANDO EL MICROCONTROLADOR PIC16F877.

T E S I S

QUE PARA OBTENER EL TÍTULO DE:

INGENIERO ELÉCTRICO ELECTRÓNICO

P R E S E N T A:

ÍGOR CLAVEL HERRERA

DIRECTOR DE TESIS:

ING. MOISÉS EUGENIO RUEDA GUTIÉRREZ

MÉXICO, D. F. 2004

Dedicatoria

Dedicado a ti ***Angélica****, por todo tu amor, cariño, y comprensión. Porque sin ti no hubiera podido concluir mi carrera. Por estar siempre conmigo y por convidarme de una familia y un hogar. Por el amor que te tengo y por ser mi amiga. Por todo esto y por muchas cosas más, es para ti este trabajo. Gracias Amorcita. TA=82=83260* $[(\infty^{\infty}) *]$!

No siempre puedes obtener lo que deseas, pero si lo intentas, puede que a veces encuentres lo que necesitas.

Parafraseando a
MICK JAGGER Y KEITH RICHARDS

Agradecimientos

A Nene, mi abuela, por brindarme un hogar, y al igual que a mis tías Vivian y Beatriz, por su apoyo para comenzar mi carrera.

A mi familia adoptiva, Angélica, sus papás Yolanda y Esteban y sus hermanos Yolanda y Esteban. Gracias por hacerme parte de su familia, por su apoyo y por mi carrera. ¿Cómo podría yo agradecerles lo suficiente? Imposible.

Al VMO, por la convivencia, la camaradería y por qué no, por algo de buen futbol. Porque en esos compañeros encontré amistades que permanecerán a pesar de la distancia, el tiempo y el olvido.

A Gonzalo López de Haro, Abigail Serralde, Miguel Figueroa, Gloria Loranca, Joel Gómez, Osvaldo Pereida, Jesús Patiño y a Aurelio Sánchez. Por la excelente bienvenida a mi regreso y por todo el apoyo que me han brindado.

A Moisés Rueda, de manera muy especial, por compartirme un poco de su experiencia y conocimientos, como profesor y más tarde como director de tesis. Gracias por aceptar dirigirme, por tenerme paciencia y por encontrar siempre el tiempo necesario para atenderme.

A Yukihiro Minami, David Vázquez, Juan Manuel Castillo, Larry Escobar y mis otros profesores.

A mi Universidad y a mi Facultad.

Índice

Introducción

El objetivo del presente trabajo es diseñar un dispositivo digital de bajo costo que permita a los ciclistas contar con la información básica de su desempeño físico para evitar accidentes y llevar a cabo correctamente el entrenamiento y la práctica de su deporte. En el mercado existen algunos productos, que aunque relativamente económicos, solo presentan medidas de velocidad, distancia y tiempo. Al incluir otras funciones que permiten el monitoreo el estado físico del ciclista y el desarrollo de planes de entrenamiento, su costo se eleva considerablemente haciendo difícil que se invierta en un accesorio que en ocasiones puede costar más que la propia bicicleta.

El ciclismo, como la mayoría de los deportes, es una actividad que puede proporcionar beneficios a la salud practicándolo de manera adecuada. Es una actividad que se puede desarrollar desde la niñez y hasta una edad avanzada y que no es tan exigente con el cuerpo pues no existe un contacto constante y violento con el suelo. Sin embargo, su práctica requiere de ciertos cuidados, que por considerarse a ésta como una actividad recreativa y de paseo, pueden no ser tomados en cuenta.

Un esfuerzo para el que el organismo no esté preparado, las condiciones ambientales y los malos hábitos de entrenamiento contribuyen a que se puedan presentar problemas en la salud si no se tienen los cuidados necesarios. Las insolaciones, los golpes de calor, la insuficiencia respiratoria, la fatiga y un mal funcionamiento cardiovascular son algunos de ellos. Los casos más graves pueden resultar en una muerte súbita, mientras que en los de menor gravedad pueden ocurrir mareos o malestares que afectarán el control sobre la bicicleta pudiendo provocar accidentes.

Las caídas son los accidentes más comunes. Aunque se presentan desde una corta altura, la velocidad y las características del terreno pueden agravar las lesiones sufridas, llegando incluso a ser mortales. Las causas de una caída van desde los malestares físicos ya mencionados hasta los objetos extraños en la superficie donde se rueda, pero también por la falta de pericia de los ciclistas y por la imprudencia de peatones y conductores de vehículos.

En nuestro país existen pocas instalaciones adecuadas para la práctica del ciclismo y el traslado hasta ellas resulta difícil para muchas personas, por lo que practican en las calles en horarios en que, aunque circulen pocos automóviles, las condiciones de visibilidad no son las más adecuadas. Al no existir una cultura de respeto al ciclista, son comunes los casos de atropellamiento como los que han sufrido personas al pedalear por el anillo periférico de la Ciudad de México durante los fines de semana.

Con el dispositivo a diseñar se busca proporcionar a los ciclistas los elementos básicos para el monitoreo de su condición física y de las condiciones ambientales que les permita cuidar su salud. Además podrán llevar un control adecuado de las distancias recorridas y tiempos empleados para el calentamiento y el entrenamiento. Algunos parámetros ambientales, como

humedad y presión, no son considerados debido a que repercuten de manera importante en el costo total del dispositivo, pero se incluirán medidas que permitan monitorear su influencia en le estado físico del ciclista. Se buscará que se adapte a cualquier bicicleta y que ayude a aumentar las condiciones de seguridad del ciclista al circular cerca de automóviles.

Para el diseño de un sistema digital se pueden emplear circuitos lógicos y secuenciales, dispositivos lógicos programables, microprocesadores o microcontroladores. En los primeros casos la implementación requiere bastantes componentes y los cambios en las funciones implican una modificación del hardware, aunque éste sea programado. Los microprocesadores tienen las mejores características de capacidad, generalidad y desempeño pero su costo es demasiado elevado. Para el desarrollo del dispositivo se seleccionó un microcontrolador de ocho bits debido a que ofrece el mejor equilibrio entre costo y facilidad de implementación.

Sin embargo, el mundo real no es compatible directamente con los sistemas digitales, por lo que se debieron seleccionar o diseñar las interfaces analógicas que mejor se adaptaran al entorno del diseño y que no representaran un incremento sustancial en el costo total del dispositivo.

El capítulo I describe al ciclismo y a su entorno en cuanto a las características de la bicicleta, del entrenamiento y del desempeño del organismo. Es en éste donde se distinguen las necesidades que el dispositivo debe cubrir, las cuales son analizadas en el segundo capítulo, en el que se definen las características que debe tener el dispositivo para cubrir las necesidades descritas en el capítulo antecedente.

El capítulo III del trabajo introduce a la teoría de los microcontroladores y a algunas interfaces analógicas. Es ahí donde se determina el modelo mas adecuado para el dispositivo a desarrollar, el cual se describe a detalle en el cuarto capítulo.

A partir de lo definido en el capítulo II, el dispositivo se desarrolla en cuanto a hardware y software en los capítulos quinto y sexto respectivamente. En el séptimo y último capítulo se describen las herramientas empleadas, el proceso de la implementación y las pruebas realizadas tanto al hardware como al software para verificar que el funcionamiento del dispositivo fuera el adecuado.

Capítulo I

El Ciclismo y su Entorno

1.1. Breve historia de la bicicleta

Las primeras nociones documentadas de la bicicleta son algunos bocetos de Leonardo da Vinci hacia el año 1490, en los que ya se distinguía la idea de una transmisión mediante una cadena y un par de pedales.

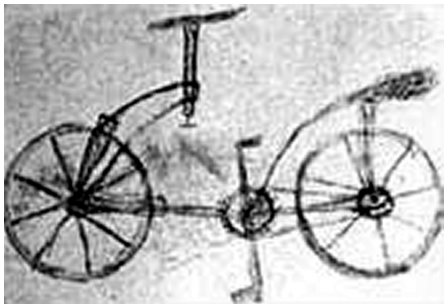


Figura 1.1: Boceto de Leonardo da Vinci existente en la biblioteca Ambrosiana de Milán.

Los ancestros de la bicicleta actual son el celerífero y velocífero, inventados por el conde Mede de Sivrac por el año 1790. Se componían de un bastidor de madera y un par de ruedas. Debido a que estos artefactos no contaban con una dirección móvil y el movimiento se generaba con la acción directa de los pies sobre el piso no se reconoce al conde de Sivrac como el inventor de la bicicleta [1].

En 1817 el barón Drais agregó una dirección móvil al velocípedo, con lo que el conductor podía hacer girar la rueda delantera y así cambiar de dirección. Esta máquina, llamada *draisiana* en su honor, es considerada como la primera bicicleta.

Figura 1.2: La primera bicicleta, obra del alemán Drais.

El siguiente cambio que significó una verdadera evolución fue el que hizo que el velocípedo dejara de impulsarse por la acción directa de los pies en el piso. Esta nueva característica se le

debe al francés Ernest Michaux que en 1861 inventó los pedales al incorporarle un par de manivelas a la rueda delantera para que se pudiera hacer girar con los pies.

Figura 1.3: Incorporación de los pedales al velocípedo.

Algunos años más tarde aparece la denominada *Grand Bi*, versión del velocípedo famosa por su rueda delantera de enormes proporciones. Esto se pudo lograr gracias al cambio de la madera por el metal en el cuadro y en parte de las ruedas. A pesar de contar con un mejor rendimiento, gracias a la enorme diferencia del radio de los pedales y el de la rueda, esta máquina será recordada por su también enorme inseguridad.

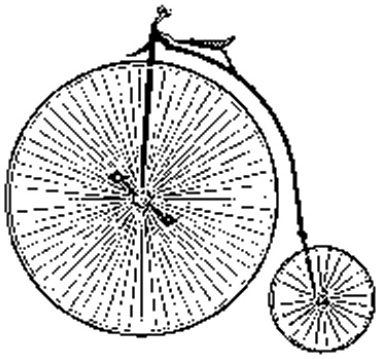


Figura 1.4: La *Grand Bi.*

Posteriormente se regresó a ruedas de similar diámetro y poco a poco se fueron incorporando los elementos que caracterizan a las bicicletas de la actualidad, como es la tracción por medio de cadena y las ruedas con recubrimiento de caucho que posteriormente fueron cambiadas por los neumáticos.

A principios del siglo veinte comenzaron las competiciones y campeonatos internacionales, motivándose la mejora de la bicicleta con la finalidad de ayudar a los ciclistas a tener un mejor rendimiento. Con la aparición de diversas modalidades de ciclismo se han ido agregando características más específicas que permiten mejorar el desempeño de la bicicleta en cada una de estas modalidades. Ejemplo esto es la aparición del cambio de velocidades.

En la bicicleta de ruta se ha buscado reducir el peso sin alterar la resistencia de la estructura, lo que ha permitido incorporar nuevos materiales como el aluminio y la fibra de carbono. También se han perfeccionado los mecanismos del cambio de velocidades y de los frenos.

Las bicicletas de pista se han hecho cada vez más aerodinámicas con la finalidad de reducir los tiempos necesarios para cubrir las diferentes distancias en que se compite. Uno de los principales incentivos a través de la historia de estas competencias es el record de la hora.[1]

1.2. El deporte ciclista

En un inicio la bicicleta fue creada para proporcionar un transporte personal que sustituyera al caballo, principal medio de transporte en esa época. Al transcurrir de los años esta máquina se fue incorporando a diversos ámbitos de la sociedad, dejando de ser únicamente un elemento auxiliar para trasladarse de un punto a otro con poco esfuerzo.

Uno de los juguetes más apreciados por los niños es un triciclo y posteriormente, una bicicleta. A pesar de que para los pequeños no es mas que un juego que les otorga un cierto grado de libertad, el *andar en bicicleta* les representa un ejercicio y el desarrollo de su equilibrio.

La bicicleta es también una herramienta de trabajo para diversos oficios, desde el cerrajero o el afilador, hasta el lechero y más recientemente los *bicitaxis*.[2] En países como China representa el principal medio de transporte beneficiándose con esto el medio ambiente pues es una opción no contaminante.

El subirse a la bicicleta representa también una actividad recreativa y saludable. Son comunes los paseos familiares montados en bicicleta, ya sea en un parque o en el campo respirando aire fresco. El surgimiento del ciclismo como deporte se debe por un lado a esto y por otro al continuo deseo del hombre por competir y demostrar su superioridad física o mental. Muchas disciplinas como la lucha, el boxeo o el futbol, han surgido por esta misma razón.

El ejercitarse pedaleando de manera moderada no significa una tortura por lo que muchas personas practican el ciclismo para mantenerse en forma y evitar el sedentarismo. Sin embargo, cuando una persona decide practicar más en serio este deporte debe tomar en cuenta algunos aspectos para evitar que se vuelva contraproducente. Como muchas cosas en la vida, practicar el ciclismo en forma excesiva y sin cuidado puede crear problemas de salud.

Una práctica más seria de este deporte no significa únicamente entrenar y competir en pruebas ciclistas, independientemente del nivel de competencia, sino también realizarlo periódicamente con el objetivo de desarrollar las capacidades físicas o como apoyo a la práctica de otro deporte. En cualquier caso el deportista debe cuidar aspectos técnicos, físicos y médicos para que el ejercicio le resulte beneficioso.

1.2.1. Características y beneficios del ciclismo

El ciclismo tiene la peculiaridad de requerir muy poco gasto energético para desplazarse de un lugar a otro. No exige un esfuerzo continuo, se puede dejar de pedalear por cortos periodos de tiempo para descansar las piernas y la bicicleta continuará moviéndose.

[1] En esta prueba se trata de recorrer la mayor distancia posible en una hora.

[2] Esta modalidad de transporte público cubre las necesidades de transporte en zonas donde el acceso a vehículos automotores está restringido. En la Ciudad de México hay bicitaxis en el Centro Histórico y Xochimilco entre otros lugares.

El ciclista va sentado en la bicicleta con lo que sus piernas realizan movimientos naturales y solo ejercen fuerza sobre los pedales. Éstos descienden oponiendo poca resistencia, lo que reduce la aparición de lesiones en los tendones y en las articulaciones. Cuando hay un contacto violento de los pies con el suelo, como el que se presenta al correr o al saltar, estas lesiones se presentan con mucho mayor frecuencia.

Para practicarlo no se requiere tener músculos muy desarrollados, pues es más importante hacerlos trabajar durante largos períodos de tiempo. Los músculos deben entonces alimentarse con oxígeno enviándoles grandes cantidades de sangre. El corazón es el encargado de hacerlo, por lo que es el órgano que más trabaja y el que más se desarrolla con la práctica constante.

Además de mejorar las capacidades cardiovasculares (circulación y presión arterial, disminución de la frecuencia cardiaca en el esfuerzo y en el reposo) también mejora la ventilación de los pulmones y la flexibilidad de las articulaciones. Llevando una buena disciplina al practicar este deporte se promueve una buena alimentación, se evitan el sobrepeso, el tabaquismo y el alcoholismo.

1.2.2. Organización del deporte

El ciclismo es un deporte federado regido por la Unión Ciclista Internacional. La UCI es una asociación internacional que reglamenta y controla el ciclismo, organiza las competencias internacionales y colabora con el Comité Olímpico Internacional para la participación de los ciclistas en los Juegos Olímpicos [2]. Las lenguas oficiales de la UCI son el francés y el inglés, por lo que los estatutos y reglamentos son redactados en estos idiomas.[1] Sus miembros son las federaciones nacionales de ciclismo, que a su vez organizan y controlan la práctica del ciclismo en sus respectivos países.

Las federaciones nacionales agrupan a ciclistas amateurs y profesionales, organizan competencias nacionales y colaboran con la UCI para la realización de eventos internacionales. En México se han realizado pruebas internacionales tanto de pista como de ruta a las que han asistido ciclistas de primer nivel.

Las características de las pruebas definen las diferentes disciplinas del ciclismo.

Carretera

Es la más conocida pues el ciclismo profesional se desarrolla en esta disciplina. Como su nombre lo indica se corre en asfalto, ya sea dentro de las ciudades y poblaciones o viajando entre ellas. Las pruebas pueden ser de un día o de varios días e incluso semanas de duración (pruebas por etapas). Las competencias internacionales más conocidas son los campeonatos mundiales y las tres grandes pruebas que se llevan a cabo en España (*La Vuelta*), Italia (*El Giro*) y en Francia (*El Tour*).

Las pruebas y etapas consisten en cubrir una distancia determinada donde el ganador es el primero en llegar o el que haga menos tiempo (*contra reloj*). Se pueden desarrollar en llanos o en montaña, lo que exige dosificar el esfuerzo. La distancia recorrida en las etapas no pasa de 250 kilómetros siendo más cortas las pruebas contra el reloj.

[1] La Real Federación Española de Ciclismo publica en su página de Internet (www.rfec.com) una traducción al español de estos documentos, que es la versión consultada para el presente trabajo.

Pista

Se lleva a cabo en pistas especiales para bicicletas con dimensiones específicas. La longitud de la pista puede ser de 333.33 metros o 250 metros y presentan cierta inclinación o peralte. En la mayoría de las pruebas lo importante no es llegar primero sino hacer el menor tiempo posible, por lo que los corredores deben lograr altas velocidades.

Ciclo-cross

Surge en Europa debido a que en el invierno es difícil practicar ciclismo de carretera por la presencia de lluvia y nieve. Se desarrolla en circuitos trazados en el campo y sobre una superficie de tierra y hierba que una vez comenzada la prueba se convierte en lodo. La intensidad del esfuerzo invertido es altísima por lo que se requiere una excelente condición física. Hay partes de las pruebas que por la dificultad del terreno deben ser recorridas a pié cargando la bicicleta.

Mountain Bike

Surge en Estados Unidos en los años setenta como una nueva opción de realizar deportes ecológicos. Se desarrolla en bosques, ríos, playas y en general por toda la gama de terrenos disponibles en la naturaleza, incluido el hielo. Las competencias de esta disciplina se caracterizan por requerir un alto grado de esfuerzo para salvar cualquier tipo de obstáculo.

Los jóvenes han adoptado esta modalidad como entretenimiento y aventura y en general es una modalidad que las familias pueden practicar juntos pues permite interactuar con la naturaleza mientras se realiza un paseo, pudiéndose entonces considerar como una versión del cicloturismo.

Cicloturismo

Se refiere a la práctica general del ciclismo fuera de cualquier competencia. Se trata de paseos o excursiones, en solitario o en grupo, por recorridos previamente establecidos. Generalmente se desarrolla en carreteras aunque hay casos en que se mezcla con el Mountain Bike para realizarse a campo traviesa pero por terrenos llanos que no presenten obstáculos significativos.

Esta disciplina es elegida por aquellas personas que buscan mantener una buena forma física mientras conviven con otras personas y con la naturaleza. Las distancias dependen de la capacidad de cada ciclista y pueden ir desde unas decenas de kilómetros hasta cientos de kilómetros pero recorridos en tiempos razonables (varias horas). El ritmo al cual se cubren los recorridos es bajo (alrededor de los 20 km/h) y se pedalea con bajo esfuerzo.

Triatlón

Esta disciplina es el resultado de la mezcla de tres pruebas: natación, carrera ciclista y carrera a pié. Exige una alta capacidad física para poder cubrir las tres distancias que individualmente son de por sí rigurosas. La parte realizada sobre la bicicleta puede ir desde veinte kilómetros y hasta cerca de los doscientos para los profesionales.

Otro elemento que dificulta aún más esta disciplina es que se realiza en lugares donde hay agua, por lo que generalmente el nivel de humedad es alto y repercute negativamente en el desempeño físico del deportista.

Otras disciplinas

Existen otras disciplinas en que se necesita tener buen equilibrio sobre la bicicleta y no tanto una buena capacidad física. El Ciclo-polo es una versión del Polo pero los participantes se montan sobre bicicletas. El BMX se desarrolla en pistas cortas que incluyen obstáculos como rampas mientras que en el ciclismo artístico y el estilo libre se deben realizar giros, saltos y piruetas de distintos grados de dificultad.

1.2.3. El ciclismo en México

El ciclismo de competencia es totalmente amateur, aunque se han dado casos de ciclistas de ruta que ingresan a las filas de equipos profesionales en Estados Unidos o Europa. No existe mucho apoyo como sucede en el ciclismo amateur europeo y los corredores dependen de instituciones que pueden contar con recursos suficientes para mantener programas de ciclismo. Algunos ejemplos son la Universidad Nacional Autónoma de México, el Instituto Mexicano del Seguro Social, universidades públicas y privadas e institutos estatales del deporte que apoyan en diferente medida al ciclismo amateur.

La Federación Mexicana de Ciclismo es la encargada de promover y controlar la práctica de este deporte. En México se han realizado competencias internacionales como la Ruta México (en distintas épocas) o campeonatos de ciclismo de pista (en su mayoría regionales). El evento más relevante fue la olimpiada de 1968. El velódromo olímpico fue considerada la pista más rápida en esa época y era el lugar predilecto por los ciclistas que competían por el record de la hora [3].

La práctica del ciclismo recreativo es la más frecuente y la bicicleta de montaña es la principal elección para llevarla a cabo. El ciclismo todo terreno ha tenido un auge y familias completas compran bicicletas de montaña para realizar paseos mas o menos regulares en lugares donde puedan convivir con la naturaleza. La frecuencia de estas salidas depende principalmente de las posibilidades que tengan las familias para salir de las ciudades a lugares como el Ajusco en la Ciudad de México o como La Marquesa o El Ocotal en el Estado de México. En el Distrito Federal se han implementado algunos programas en que ciertas calles (en su mayoría del centro histórico) son cerradas al transito vehicular los fines de semana para dar paso a los ciclistas. La falta de participación de éstos ha provocado que estos programas sean cancelados en lugar de expandirse a otras zonas.

El ciclismo representa una inversión pues se debe adquirir tanto la bicicleta como diversos accesorios. Por esto es más común que los mexicanos prefieran deportes que requieren de poco o ningún equipamiento (como el futbol, el atletismo o el básquetbol) a pesar de que la fisiología común del mexicano presenta aptitudes para pruebas de fondo (maratón y marcha principalmente) que les ayudarían a destacar en el ciclismo. Los ciclistas mexicanos que han competido en el ámbito profesional se han caracterizado por ser especialistas en etapas de montaña en las que se requiere bastante resistencia.

Generalmente se opta por adquirir una bicicleta de montaña porque su precio es menor que el de las de carretera. Se pueden encontrar bicicletas desde mil pesos y su precio va aumentando conforme presentan mas y mejores características (como la suspensión o cuadros más ligeros y rígidos). Las bicicletas de ruta comienzan alrededor de los mil quinientos pesos en sus versiones más austeras y los modelos especialmente diseñados para la competición comienzan en un precio cercano a los diez mil pesos. Muchas veces la compra de la bicicleta consume todos los recursos y la adquisición de accesorios complementarios como los equipos de

seguridad o de apoyo como el casco, los cronómetros con odómetro y velocímetro o los monitores de pulso, se dejan para después.

Otra característica del ciclismo en México es que no dispone de muchos lugares para su práctica. En la Ciudad de México hay instalaciones deportivas donde se puede pedalear sin riesgo pero no toda la gente puede trasladarse hasta ellas. En la UNAM el equipo de ciclismo practica en los circuitos de Ciudad Universitaria y en el Velódromo Olímpico. Es común ver los fines de semana a ciclistas pedaleando por el Periférico muy temprano para aprovechar al máximo el poco tiempo en que la cantidad de automóviles les permite circular con cierta seguridad.

En las carreteras del interior de la República los ciclistas pueden rodar largas distancias pero siempre deben ir acompañados de un automóvil que los vaya protegiendo. Es muy diferente a países europeos en que el respeto al ciclista está muy arraigado en la sociedad. En España por ejemplo, algunas carreteras incluyen un carril específico para los ciclistas y en algunos casos existe una zona de seguridad entre éste y los carriles para automóviles.

1.3. La bicicleta

1.3.1. Principales tipos de bicicleta

La bicicleta se compone de diversos elementos básicos y su forma depende de la disciplina para la que sea diseñada. Por ejemplo, las ruedas para carretera son delgadas mientras que las de bicicletas de montaña son más anchas y resistentes para soportar el uso rudo.

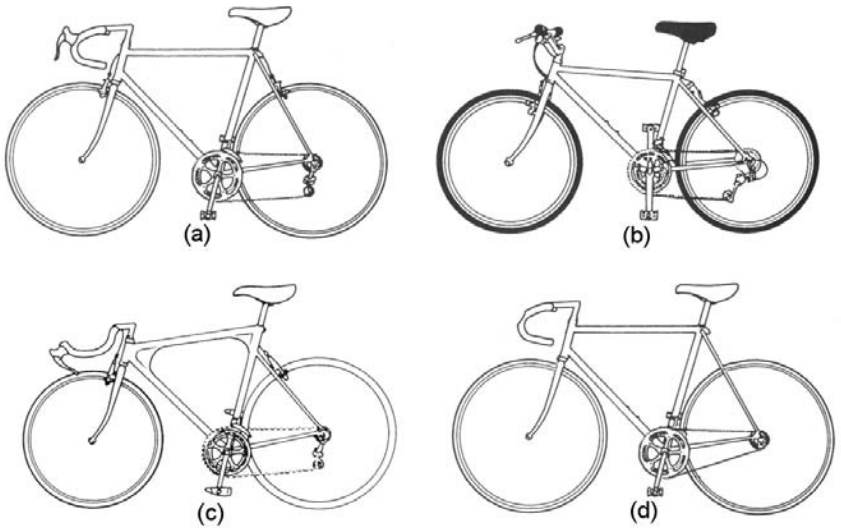


Figura 1.5: Los diferentes tipos de bicicleta: (a) Carretera, (b) Montaña, (c) Contrarreloj, (d) Pista.

Bicicleta de carretera

Se emplea para rodar sobre superficies regulares como el asfalto y el concreto. Debe ser ligera sin perder rigidez y el material con que se fabrique depende de la aplicación. Puede ser empleada para competencias, cicloturismo o excursión. Hay versiones muy sencillas empleadas

en las ciudades como transporte. Para competencias de ciclo cross se le deben colocar ruedas un poco más gruesas para que pueda rodar en terrenos difíciles.

Las dimensiones y la rigidez dependerán del uso que se le dé, siendo las bicicletas de competición las más ligeras y las de ciclo cross las más rígidas. Para el uso general, como el paseo o el cicloturismo, los materiales deben aportar resistencia para aumentar la durabilidad pero no deben alterar demasiado el costo total para mantener accesible el costo de las bicicletas.

Bicicleta de montaña

Es una bicicleta *todo terreno* creada en Estados Unidos que originó el Mountain Bike. Debe ser muy resistente por lo que el grosor de los tubulares es mayor que el de otros tipos de bicicleta. Las ruedas son anchas y los neumáticos tienen una superficie que facilita circular en superficies irregulares y resbalosas. Hay neumáticos que incluyen picos metálicos que permiten rodar sobre nieve y hielo.

En los últimos años este tipo ha sido el predilecto de las personas que únicamente quieren una bicicleta para pasear. Mezcla la forma básica de la bicicleta de carretera con las características de manejo y resistencia de las bicicletas tipo BMX que son de menor tamaño y que no cuentan con cambios de velocidad (anteriormente eran consideradas como un juguete).

Se emplea en cicloturismo, como bicicleta de ciudad, como transporte y en competencias a campo traviesa. Para competición se le han agregado sistemas de suspensión que permiten absorber las vibraciones debidas al constante choque de las ruedas con el suelo.

Bicicleta de pista

Esta bicicleta no tiene cambios de velocidad ni frenos. Su composición facilita la velocidad, elemento básico de las pruebas que se levan a cabo en la pista. El manillar y el cuadro perfilado permiten al corredor inclinarse de manera que se minimiza la resistencia al aire del ciclista. Los materiales empleados hacen que estas bicicletas sean las más ligeras (entre seis y ocho kilogramos) sin que pierdan su resistencia. Al circular sobre superficies lisas no requieren absorber demasiadas vibraciones por lo que la ruedas y la dirección pueden pesar solo algunos gramos.

Bicicleta contrarreloj

Es básicamente una bicicleta de carretera a la que se agregan elementos de las bicicletas de velocidad como el cuadro perfilado y el manillar. A diferencia de la bicicleta de pista, ésta sí cuenta con frenos y con cambios de velocidad.

Se emplea en las etapas contra el reloj en competencias de ruta y en la etapa ciclista del triatlón. De hecho el manillar con soporte para los codos empleado tanto en este tipo como en el de pista fue una aportación de esta disciplina.

1.3.2. Componentes de la bicicleta

Las dimensiones de los diferentes elementos que componen la bicicleta deben ser determinadas por la talla del ciclista que la ocupará. Una bicicleta *a la medida* resulta costosa por lo que prácticamente solo algunos ciclistas pueden adquirirla. La mayoría de los ciclistas profesionales son dotados con bicicletas con dimensiones acordes a sus medidas.

Se deben emplear los componentes que mejor se adapten a la morfología de cada ciclista para evitar lesiones en los músculos y en las articulaciones, además de que se intenta asegurar un rendimiento cercano al óptimo que permita un mejor desarrollar de las capacidades físicas.

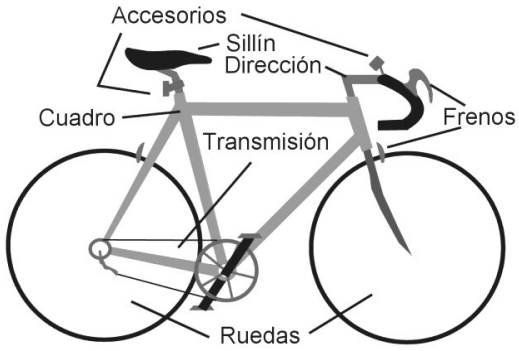


Figura 1.6: Componentes básicos de la bicicleta.

A continuación se describen los elementos básicos que componen a la bicicleta y las consideraciones que se deben tener en cuanto a sus dimensiones.

El cuadro

Es el elemento principal de la estructura que conforma una bicicleta y es en él donde se fijan los demás componentes. La forma de la bicicleta depende de la forma del cuadro, así como gran parte del peso y la rigidez. Los materiales con que se fabrican son el acero (para bicicletas económicas y de uso general), el aluminio reforzado, el titanio y en la actualidad los materiales compuestos de carbono que proporcionan alta rigidez con un peso mínimo.

A mediados de la década de los noventa se habían logrado excelentes relaciones peso-rigidez empleando el aluminio reforzado con aleaciones de cobre o zinc. Las estructuras tubulares estaban *conificadas* de manera que en las uniones el grosor era mayor que en el centro. Cuando parecía que esta tecnología iba a permanecer vigente por muchos años, se comenzaron a emplear fibra y compuestos de carbono para la fabricación de los elementos de la dirección y la horquilla. Posteriormente estos materiales fueron incorporados en la fabricación del cuadro con lo que en la actualidad representan la máxima tecnología y que como tal no es accesible para la mayoría de los ciclistas.

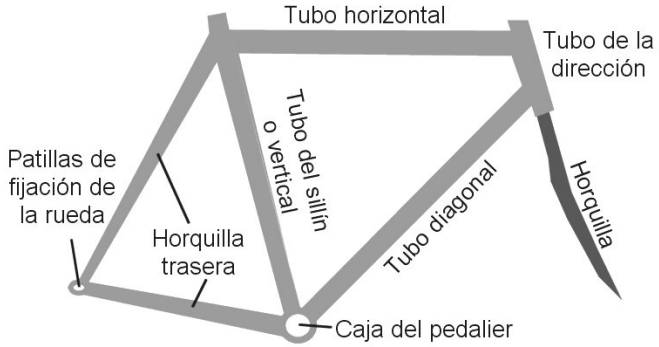


Figura 1.7: Piezas que conforman el cuadro.

El cuadro se compone de tres tubos centrales (Horizontal, diagonal y tubo del sillín) y la horquilla trasera que sujeta la rueda. El tubo horizontal define la longitud del cuadro y sus valores varían desde 49 y hasta 61 centímetros. El ángulo agudo que forman este último y el tubo vertical varía entre 72º y 75º. El tubo del sillín tiene una longitud de entre 49 y 62 centímetros [4].

La altura del cuadro es la distancia entre el piso y el tubo horizontal y debe ser igual a la longitud de la entrepierna del ciclista. A partir de esto los cuadros son clasificados por tallas según la longitud del tubo vertical y dependiendo de la altura del ciclista. La tabla 1.1 muestra algunos valores a manera de ejemplo [5].

Altura del ciclista [metros]	Altura del cuadro [centímetros]
1.60 – 1.65	51 – 53
1.65 – 1.70	53 – 55
1.70 – 1.75	55 – 57
1.75 – 180	57 – 58
1.80 – 1.85	58 – 59
1.85 – 1.90	59 – 60

Tabla 1.1: Altura del cuadro dependiendo de la altura del ciclista.

La horquilla es la encargada de conectar la dirección con la rueda delantera para hacer girar la bicicleta. Se introduce a la dirección, que a su vez se encuentra en el interior del tubo de la dirección. Debe ser capaz de absorber las vibraciones que transmite la rueda. La parte inferior tiene una desviación que hace que el eje de la rueda delantera no coincida con el eje sobre el que está el tubo de la dirección. La distancia entre estos dos se denomina *flecha de la horquilla* y para asegurar la estabilidad de la dirección debe estar entre 4 y 4.5 centímetros.

La dirección

Mediante la dirección el ciclista puede guiar la bicicleta. Incluye el conjunto de elementos dentro del tubo de la dirección, el manillar (también conocido como manubrio) y la potencia. Esta última es un tubo en forma de escuadra que conecta el manillar con el tubo de la dirección y la horquilla.

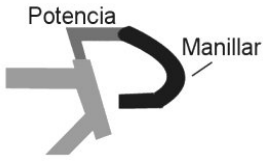


Figura 1.8: Componentes de la dirección.

El ancho del manillar debe coincidir con los hombros del ciclista. Si es más angosto afectará la respiración y si es más ancho los músculos del pecho tendrán que sostener el tórax en lugar de que únicamente lo hagan los brazos [6].

La potencia tiene como finalidad alejar el manillar del cuerpo del ciclista y su longitud dependerá de la talla del ciclista pues entre más alto sea el corredor más largos tendrá los brazos y deberá estar más alejado del manillar para conservar una postura cómoda. Además se debe considerar

que dos personas con la misma altura pueden tener medidas de tórax y brazos diferentes. La tabla 1.3 indica algunos valores para la longitud de la potencia dependiendo de la altura del tubo vertical [7].

Talla bicicleta	Longitud potencia
51 – 53 cm	7 – 8 cm
53 – 55 cm	8 - 9 cm
55 – 57 cm	9 - 10 cm
57 – 59 cm	10 - 11 cm
59 – 60 cm	11 - 12 cm

Tabla 1.2: Valores comunes de la longitud de la potencia a partir de la longitud del tubo del sillín.

La transmisión

Es el mecanismo que permite transmitir la fuerza de las piernas a la bicicleta para que ésta avance. El primer elemento son los pedales, que permiten colocar los pies sobre las bielas. Hay pedales simples donde el pie actúa sobre ellos sin ningún tipo de fijación o de sujeción automática que permiten tener el pie asegurado al pedal, pero que en caso de una caída permiten liberarlo fácilmente.

Los pedales están conectados a las bielas que a su vez van montadas sobre el eje pedalier, que está dentro de la caja del pedalier. Las bielas tienen una longitud típica de 17 centímetros y montados sobre la biela derecha van los platos (dos o tres para las bicicletas con cambios de velocidad o uno para las bicicletas de pista).

La cadena transmite la fuerza desde el conjunto del pedalier (pedal, bielas y eje) a la rueda trasera. Si la bicicleta incluye cambios de velocidad la cadena cambia de un plato a otro por medio del desviador. La cadena se une a la rueda trasera mediante el piñón, que es el conjunto de varias coronas (discos dentados más pequeños que los platos) o una sola si no hay cambios de velocidad.

El cambio trasero es el encargado de cambiar la cadena de una corona a otra. Es accionado por medio de un cable de acero conectado a una palanca. De igual manera el desviador está conectada a su respectiva palanca. Estas palancas pueden estar colocadas en el tubo diagonal, cerca del tubo de la dirección, o en el manillar. En las bicicletas de carrera se emplea un conjunto de freno-palanca de cambio que permite cambiar de velocidad sin soltar el manillar.

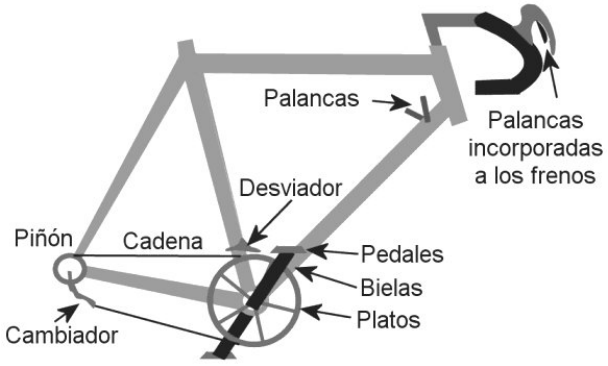


Figura 1.9: Transmisión de la bicicleta.

Las ruedas y los frenos

El elemento que permite a la rueda girar es el buje. Éste se conecta a la llanta mediante radios y sobre la llanta se colocan las cubiertas que pueden ser neumáticos o tubulares. Los frenos se colocan en las horquillas y presionan la llanta para detenerla.

La rueda delantera se le conoce como *directriz* y la trasera como *motriz* y el diámetro de ellas está regulado por la UCI. El valor mínimo es de 55 centímetros y el máximo es de 70 centímetros, en ambos casos incluidas las cubiertas [8]. Generalmente se clasifican por *rodada* que es la expresión del diámetro en pulgadas. Una rueda de rodada 27 tendrá, por ejemplo, un diámetro de 68.58 centímetros.

El sillín

El ciclista se apoya sobre la bicicleta en tres puntos (manillar, pedales y sillín) pero es el sillín el que mayor comodidad debe brindarle pues es en éste en donde se apoya la mayor parte del peso del corredor. La forma depende del tipo de bicicleta, para las de paseo se busca un mayor acojinamiento con el fin de evitar molestias posteriores al ejercicio. En competencias de ruta se desea que sea ligero y confortable.

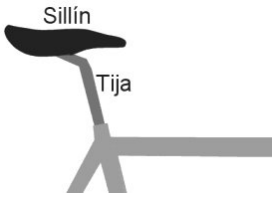


Figura 1.10: El sillín es el punto en que se apoya la mayor parte del peso del ciclista.

Se apoya en la tija del sillín, que a su vez se introduce en el tubo vertical. Se debe ajustar su altura de manera que el ciclista flexione correctamente sus piernas al pedalear.

Accesorios

Existen diversos accesorios que auxilian al ciclista de diferentes formas. Las bases para colocar botellas son importantes para que el ciclista se mantenga hidratado en largos recorridos. También se emplean elementos que permitan a otros ciclistas y a vehículos automotores percatarse de la presencia de la bicicleta.

Los accesorios electrónicos permiten al ciclista conocer los datos de tiempo y distancia de los recorridos que realiza, así como monitoreo de la frecuencia cardiaca para cuidar que ésta no llegue a ciertos valores en que la salud corra algún riesgo. Los aparatos disponibles en la actualidad cuentan con cronómetro, odómetro (medición de distancia), velocímetro y en algunos casos un indicador de altitud.

Su costo aumenta junto con el número de características que contiene. Actualmente se puede conseguir un crono-odómetro[1] con velocímetro por tan solo doscientos pesos. Cuando se trata de un monitor de ritmo cardiaco o de un crono-odómetro que lo incluya los precios se disparan por arriba de los mil pesos.

[1] Dispositivos que mide tanto tiempo (cronómetro) como distancia recorrida (odómetro).

1.3.3. Mecánica de la transmisión

La transmisión tiene el objetivo de hacer girar las ruedas lo más rápido posible con la menor cadencia de pedaleo. El diámetro de los platos es mayor que el de la corona, por lo que un giro de los pedales representará más de uno para el piñón y la rueda trasera. Esta relación se calcula dividiendo el número de dientes del plato entre el número de dientes del piñón y nos dará como resultado las vueltas que da la rueda trasera por cada giro del plato (un pedaleo completo).

$$relación = \frac{n^o\ dientes\ plato}{n^o\ dientes\ piñón} \qquad (1.1)$$

La tabla 1.3 presenta distintas relaciones para valores comunes de dientes de coronas y platos:

Dientes piñón	Dientes plato						
	49	**50**	**51**	**52**	**53**	**54**	**55**
12	4.08	4.17	4.25	4.33	4.42	4.50	4.58
13	3.77	3.85	3.92	4.00	4.08	4.15	4.23
14	3.50	3.57	3.64	3.71	3.79	3.86	3.93
15	3.27	3.33	3.40	3.47	3.53	3.60	3.67
16	3.06	3.13	3.19	3.25	3.31	3.38	3.44
17	2.88	2.94	3.00	3.06	3.12	3.18	3.24
18	2.72	2.78	2.83	2.89	2.94	3.00	3.06
19	2.58	2.63	2.68	2.74	2.79	2.84	2.89

Tabla 1.3: Relaciones entre diferentes platos y coronas.

Para trasladar estas relaciones a distancia se debe conocer el perímetro de las ruedas, que es la distancia que recorren por cada vuelta que dan. A partir del diámetro se puede obtener multiplicando por π (3.1415927 aproximadamente) empleando la ecuación para obtener el perímetro de una circunferencia a partir de su diámetro:

$$Perímetro = \pi \times Diámetro \qquad (1.2)$$

A partir de ésta se puede obtener esta distancia para las diferentes ruedas. Como la rodada se expresa en pulgadas se debe multiplicar por 2.54 para obtener el diámetro en centímetros.

Rodada	Diámetro [cm]	Perímetro [m]
22	55.9	1.76
23	58.4	1.84
24	61.0	1.92
25	63.5	1.99
26	66.0	2.07
27	68.6	2.15

Tabla 1.4: Perímetro de las diferentes ruedas.

Ahora se puede conocer la distancia recorrida por la bicicleta en cada giro completo de los pedales, multiplicando el número de vueltas que da la rueda trasera en cada pedaleo por la distancia recorrida por ésta en cada giro (perímetro de la rueda). A esto se le conoce como *desarrollo* y se obtiene mediante:

$$Desarrollo = Perímetro\ rueda \times \frac{n^o\ dientes\ plato}{n^o\ dientes\ piñón} \qquad (1.3)$$

Dientes piñón	Dientes plato						
	49	**50**	**51**	**52**	**53**	**54**	**55**
12	8.80	8.98	9.16	9.34	9.52	9.70	9.87
13	8.12	8.29	8.45	8.62	8.78	8.95	9.12
14	7.54	7.69	7.85	8.00	8.16	8.31	8.46
15	7.04	7.18	7.33	7.47	7.61	7.76	7.90
16	6.60	6.73	6.87	7.00	7.14	7.27	7.41
17	6.21	6.34	6.46	6.59	6.72	6.84	6.97
18	5.87	5.98	6.10	6.22	6.34	6.46	6.58
19	5.56	5.67	5.78	5.90	6.01	6.12	6.24

Tabla 1.5: Desarrollo (en metros) para ruedas de 27 pulgadas de diámetro (68.58 centímetros) y un perímetro de 2.15 metros.

1.4. Salud y ciclismo

Para poder practicar el ciclismo o cualquier otro deporte no es necesario conocer con exactitud la forma en que trabaja el organismo, pero se deben cuidar ciertos aspectos médicos y fisiológicos para evitar lesiones y otros padecimientos graves. El seguimiento médico es importante en cualquier disciplina deportiva y el ciclismo no es una excepción.

El ciclismo es un deporte con pocas contraindicaciones y puede ser practicado por la mayoría de las personas aunque pueden surgir problemas si se sobrepasan las capacidades físicas del individuo. Se debe cuidar la intensidad con que se practica (hay que evitar el exceso) además de que es importante tener una noción de dónde se encuentran los límites del organismo.

1.4.1. El corazón

El cuerpo humano funciona gracias a muchos tipos de células que realizan diferentes funciones y su principal fuente de energía es el alimento. Este es procesado por el aparato digestivo para obtener los azúcares, las grasas, las proteínas, las vitaminas, el agua y otros compuestos básicos que son transportados por el torrente sanguíneo hasta las células. El corazón es el encargado de bombear la sangre a través de las venas y las arterias de forma continua por lo que nunca descansa. Funciona durante muchos años y es fundamental para que el organismo viva por lo que se debe tener cuidado de no exigirle demasiado.

En la práctica del ciclismo el corazón debe proporcionarle a los músculos oxígeno en grandes cantidades y por largos períodos de tiempo. Por esto se dice que el ciclismo es principalmente practicado con el corazón. El ciclista debe mejorar el desempeño de su corazón durante el esfuerzo para mejorar su desempeño en la bicicleta.

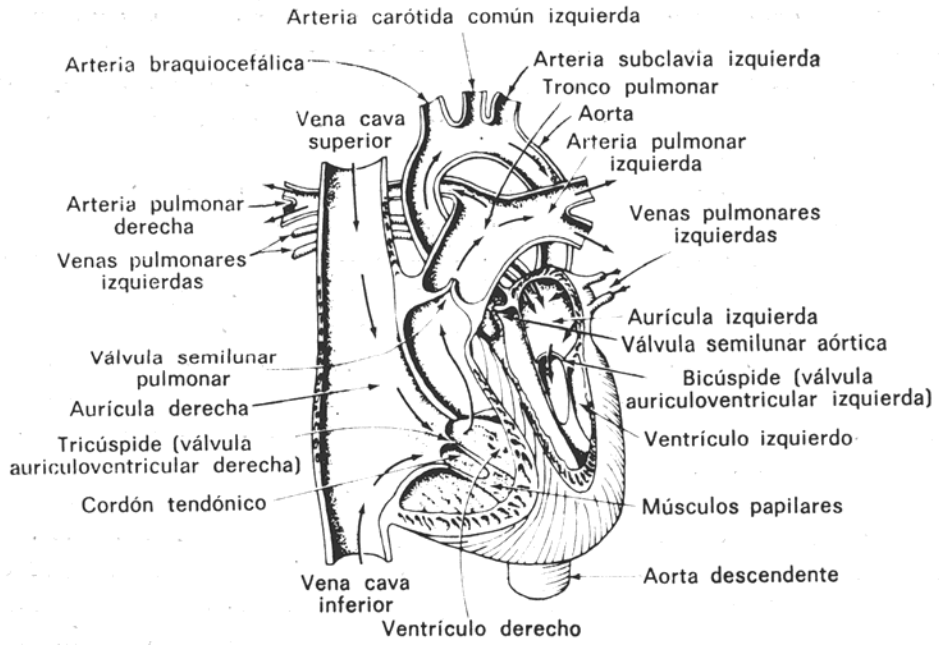


Figura 1.11: Estructura interna del corazón.

Funcionamiento general

“El corazón se puede considerar como una bomba de dos etapas, dispuestas físicamente en paralelo pero con el torrente sanguíneo atravesándolas en serie. La mitad derecha del corazón, conocida como *corazón derecho*, es la bomba que suministra sangre a los pulmones para que se oxigene, mientras que el *corazón izquierdo* suministra sangre al resto del sistema” [9]. Cada uno de ellos está rodeado de músculos que al contraerse producen el bombeo de la sangre.

Cada latido del corazón consta de dos etapas. Durante la *sístole* se contraen los músculos para bombear la sangre al organismo por medio de las arterias *aorta* y *pulmonar*. En la *diástole* los músculos se relajan y se expanden las cavidades del corazón para permitir que se vuelvan a llenar con sangre.

El control de la frecuencia cardiaca lo lleva el *nódulo sino auricular* que genera un pequeño impulso eléctrico para contraer los músculos del corazón y que se conoce como *onda P*. Al momento de la contracción se produce un segundo pulso eléctrico de mayor amplitud que se conforma de tres ondas, *Q, R* y *S*. Antes de comenzar la *diástole* los músculos se relajan produciendo el tercer impulso eléctrico (*onda T*) que caracteriza el funcionamiento del corazón y se conoce como *electrocardiograma*.

La figura 1.12 muestra el comportamiento de la presión de la sangre en el interior del corazón asociando cada instante con las componentes del electrocardiograma [10]. El pulso cardiaco se puede determinar mediante estas dos ondas, cuando se produce un complejo *QRS*, o cuando se presenta un aumento de presión.

La presión arterial disminuye a medida que el fluido sanguíneo se distribuye por el organismo. Las paredes de los vasos sanguíneos son elásticas y al presentarse el pulso de la presión se expanden produciendo un amortiguamiento en las pulsaciones.

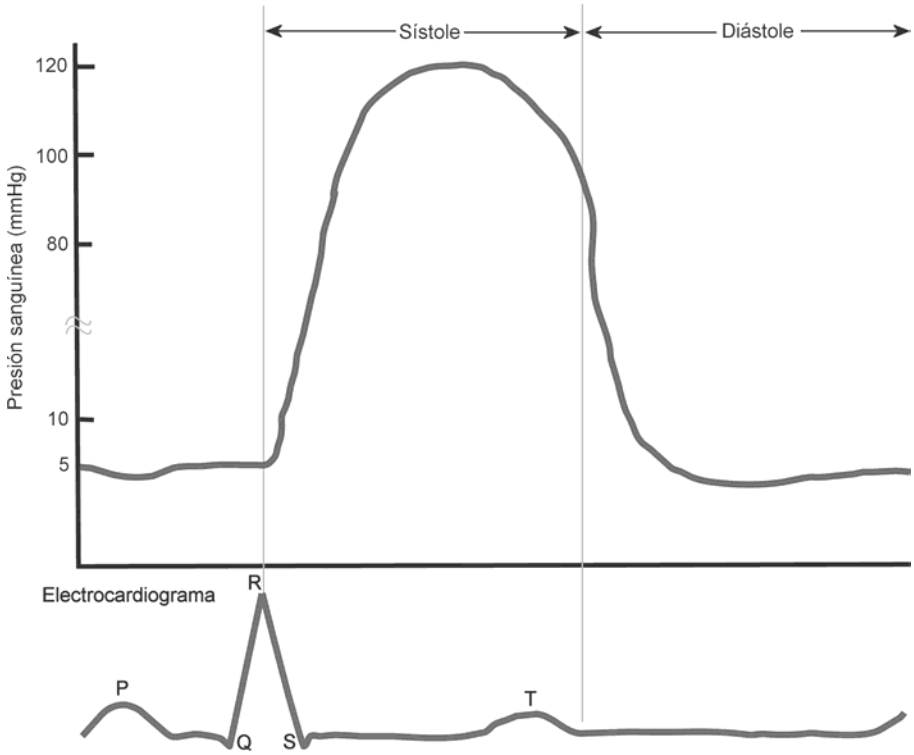


Figura 1.12: Ondas de presión y eléctricas relacionadas con el funcionamiento del corazón.

Oxigenación de la sangre

Las células requieren de oxígeno para realizar la combustión. El oxígeno del aire es captado por los pulmones y de ahí pasa al torrente sanguíneo donde es transportado hasta las células. El bióxido de carbono que resulta de la combustión es devuelto por este mismo medio a los pulmones, donde es expulsado del organismo.

Dentro de los glóbulos de la sangre el oxígeno es almacenado por una sustancia llamada *hemoglobina* que es la que determina el color de la sangre. El nivel de absorción de luz de ésta dependerá de la cantidad de oxígeno que contenga. La máxima diferencia de absorción entre la sangre oxigenada y la no oxigenada se presenta con una longitud de onda de 6500 Å. La mínima diferencia se presentará con una longitud de onda de 8050 Å en que la absorción es igual en la sangre con o sin oxígeno [11].

Mediante la medición y comparación del nivel de absorción de luz en estas dos longitudes de onda se puede cuantificar el nivel de oxigenación de la sangre. Uno de los instrumentos que se utilizan para este propósito es el *oxímetro de oreja*, que consiste en una pinza con emisor de luz de un lado y dos sensores en el otro que captan las longitudes de onda antes mencionadas. La pinza se coloca en el lóbulo de la oreja y la luz es absorbida parcialmente por las arterias capilares de esta zona.

Funcionamiento durante al esfuerzo

Cuando los músculos trabajan en esfuerzo sus necesidades aumentan considerablemente con respecto al estado de reposo. Para cubrir este incremento en la demanda de oxígeno por parte de los músculos la respiración se acelera para captar una mayor cantidad de aire.

Al ser mayor la cantidad de oxígeno que la sangre debe transportar, el corazón responde aumentando tanto el volumen de sangre bombeada en cada latido como la frecuencia con que esto ocurre (frecuencia cardiaca). El incremento del volumen de sangre bombeado produce un aumento en la presión arterial. Además del aumento de la frecuencia cardiaca y de la tensión arterial el organismo disminuye el suministro de sangre a otros órganos.

Frecuencia cardiaca

Se refiere a la frecuencia con que late el corazón y se define como la cantidad de latidos o pulsos que realiza por minuto. La frecuencia cardiaca no solo aumenta con el trabajo muscular, los demás órganos también pueden incrementar sus necesidades de oxígeno. Las emociones también pueden alteran su valor. Es normal percibir un aumento del pulso ante situaciones tensas o como consecuencia de un sobresalto.

La *frecuencia cardiaca en reposo* cubre las necesidades del organismo cuando no realiza esfuerzo y es el nivel de referencia para valorar el incremento de la frecuencia cardiaca. La mayoría de las personas tienen una frecuencia en reposo de entre 60 y 80 latidos por minuto. La práctica regular de un deporte ayuda a tener a una frecuencia en reposo cada vez menor.

La *Frecuencia Cardiaca Máxima* (FCMax) es la máxima frecuencia con la que puede bombear sangre el corazón. Aunque su valor es diferente para cada individuo y debe ser determinado mediante la realización de pruebas físicas de esfuerzo, existe un método empírico para calcular su valor teórico [12]:

$$FCMax = 220 - edad \qquad (1.4)$$

Edad	FCMax
20	200
25	195
30	190
35	185
40	180
45	175
50	170

Tabla 1.6: Frecuencia Cardiaca Máxima según la edad.

La *frecuencia cardiaca de reserva* es la diferencia entre la frecuencia en reposo y la máxima [13]. Se le llama *de reserva* porque es el intervalo del que dispone el corazón para trabajar en esfuerzo. Mientras más alta sea la frecuencia en reposo (como en el caso de las personas sedentarias) mas pronto se alcanzará el tope. El aumentar esta diferencia es uno de los principales objetivos del entrenamiento del ciclista. Como la frecuencia cardiaca máxima es un valor fijo, la única forma de aumentar la frecuencia de reserva es disminuyendo la de reposo.

1.4.2. Esfuerzo aeróbico y anaeróbico

El esfuerzo aeróbico es aquel en que los músculos trabajan con suficiente oxígeno. En el esfuerzo anaeróbico los músculos trabajan con poco o nada de oxígeno.

El ácido láctico es un residuo producido durante el trabajo muscular. En ausencia de oxígeno su producción es mucho mayor que en presencia de éste. Cuando la cantidad de ácido láctico supera la capacidad del organismo para reciclarlo se produce una saturación que a su vez provoca la fatiga muscular. Cuando el esfuerzo anaeróbico se realiza durante un periodo muy corto y con mucha intensidad no se produce ácido láctico. Esto permite distinguir dos tipos de esfuerzos anaeróbicos: el *láctico* (producción de ácido láctico) y el *aláctico* (no se produce ácido láctico).

Los esfuerzos aeróbico y anaeróbico se relacionan con la forma en que se realiza la actividad física. Al hacerlo de forma prolongada y suavemente el esfuerzo será aeróbico y define el concepto de *resistencia física*. Una buena resistencia significará que el organismo soporta actividades físicas de larga duración.

La otra forma de realizar una actividad física es hacerlo intensamente pero durante un corto periodo de tiempo (esfuerzo anaeróbico). En este caso se habla de la *potencia física* y su nivel indica la capacidad del organismo para realizar actividades intensas.

1.4.3. Problemas para la salud

En el ciclismo, como en todos los deportes, siempre se corre el riesgo de sufrir deterioros en nuestra salud. Las lesiones musculares, en las articulaciones, fracturas y en general la traumatología representan la mayor parte de estos problemas. Sin embargo se pueden presentar también problemas causados por malos hábitos (como errores en el entrenamiento o una mala dieta) o por afecciones respiratorias, cardiovasculares etc. También las condiciones ambientales pueden provocar daños a la salud.

Los problemas de salud más graves son aquellos que ponen en riesgo la vida del ciclista. Existen ciertos síntomas que pueden servir como señales de alerta ante un problema de éstos. Pueden ser dolores de pecho, sensación de ahogo, aumento de la frecuencia cardiaca, latidos del corazón demasiado fuertes, irregularidad del ritmo cardiaco, vómitos, mareos o sensación de frío cuando la temperatura ambiente sea alta.

El ciclista necesita aprender a reconocerlos y cuando se presenten debe parar inmediatamente. Cuando estos síntomas se presentan frecuentemente es necesario acudir con un médico. A continuación se explican algunos problemas que pueden provocar una muerte súbita [14].

Aneurisma cerebral y de la aorta

Es una hernia sobre una arteria del cerebro. Cuando la presión sanguínea aumenta debido al esfuerzo puede romperse, lo que provocaría una hemorragia cerebral. De manera similar, el aneurisma de la aorta es una hernia sobre esta arteria que puede reventarse durante el esfuerzo provocando una hemorragia interna.

Infarto de miocardio

Puede presentarse como consecuencia del esfuerzo físico, presentándose junto con éste o poco después. Consiste en la obstrucción total de las arterias coronarias, que son las encargadas de llevar la sangre al corazón.

Angina de pecho

Es una especie de calambre que sufre el corazón debido a que las arterias coronarias son demasiado angostas y no pueden proveer al corazón de suficiente sangre.

Paro cardiaco

Se origina por trastornos en el ritmo cardiaco que provocan que se detenga el corazón.

Golpe de calor

Se puede presentar cuando la humedad y la temperatura ambiental son altas. En estas condiciones el sudor no se evapora y el cuerpo no se enfría lo suficiente. La temperatura corporal aumenta considerablemente junto con la frecuencia cardiaca. Se pueden presentar mareos y pérdida del conocimiento y en los casos más graves un paro cardiaco.

Lesiones provocadas por caídas

Las caídas generalmente se deben a errores por parte del ciclista o de terceros (otros ciclistas, peatones, motociclistas o conductores de automóviles). También pueden ser provocadas por los efectos de algún síntoma de un problema poco grave. Por ejemplo, como resultado de perder el conocimiento debido a una insolación.

En la mayoría de los casos solo resultan en contusiones y tal vez en fracturas. Pero hay ocasiones en que el resultado de estas lesiones puede ser de mayor gravedad como por el choque de la cabeza con un objeto rígido a alta velocidad, por la caída desde una altura considerable, o por incrustación en el cuerpo de objetos puntiagudos. Para evitar este tipo de accidentes se debe rodar en lugares seguros, por lo que es recomendable buscar sitios especiales para la práctica del ciclismo y evitar practicar donde circulen automóviles o motocicletas. También se deben utilizar aditamentos de seguridad como el casco o reflectores, así como accesorios que previenen de ciertos síntomas que puedan provocar pérdida de control sobre la bicicleta.

1.4.4. Exámenes médicos

Antes de iniciar la práctica del ciclismo es recomendable que se acuda con un médico para realizar exámenes que permitan determinar si el organismo es apto para el esfuerzo. Mediante un interrogatorio se deben detectar antecedentes de afecciones cardiacas en la familia y factores de riesgo como el colesterol, la obesidad y el tabaco. Se deben buscar señales que pudieran indicar la presencia de una afección cardiaca oculta o no tratada [15].

El examen clínico debe incluir estudios cardiovasculares, pulmonares y locomotores. También una prueba de esfuerzo y un electrocardiograma de reposo. Si se detecta alguna anomalía deberán realizarse estudios más detallados para identificar las causas.

La realización de exámenes periódicos permitirán al ciclista conocer el avance de sus capacidades, detectar señales que indiquen sobreentrenamiento y sobre todo señales que indiquen la existencia de algún padecimiento como consecuencia de la práctica.

Es recomendable también que el ciclista lleve un registro de su frecuencia cardiaca en reposo, midiéndose el pulso cada mañana al levantarse. Esto le permitirá verificar la mejoría o el mantenimiento de su capacidad.

Medición de la frecuencia cardiaca en reposo

Aunque existen dispositivos que permiten medir el pulso, el método tradicional de contar los latidos del corazón funciona perfectamente para medir la frecuencia en reposo, es decir, cuando no se está haciendo nada. Si se tratara de conocer la frecuencia cardiaca rodando en la bicicleta entonces sería necesario un dispositivo que permita hacerlo automáticamente.

Para obtener este dato se debe localizar un punto en que los latidos sean fácilmente distinguidos, como el pecho o alguna vena. Se cuentan los latidos durante quince o veinte segundos y se multiplican por cuatro o tres respectivamente para proyectar el valor parcial a todo un minuto.

Test de esfuerzo de Ruffier-Dickson

Este test de esfuerzo cardiorrespiratorio puede dar una idea aproximada del nivel de adaptación y recuperación al esfuerzo que tiene el corazón. El procedimiento que se debe llevar a cabo para su realización es el siguiente [16]:

a) Medir el pulso en reposo (P_0).

b) Realizar 30 flexiones en 45 segundos.

c) Medir el pulso inmediatamente después de concluir las flexiones (P_1).

d) Reposar durante un minuto

e) Transcurrido el minuto volver a medir el pulso (P_2).

f) Calcular:

$$\frac{(P_1 - 70) + 2(P_2 - P_0)}{10} \qquad (1.5)$$

g) Interpretar el resultado mediante la tabla 1.7.

Resultado	Nivel de adaptación y recuperación del corazón
Mayor a 8	Malo
6 a 8	Medio
3 a 6	Bueno
Menor a 3	Muy bueno

Tabla 1.7: Interpretación de los resultados del test de esfuerzo.

1.5. Entrenamiento del ciclista

El entrenamiento tiene como objetivo mejorar las capacidades físicas y el rendimiento del ciclista. El ciclismo es un deporte individual, por lo que su entrenamiento debe estar personalizado considerando aspectos individuales del corredor como peso, talla, edad, nivel, objetivos y marcas.

Se debe elaborar un plan de entrenamiento para trabajar progresivamente. El rendimiento y las capacidades físicas no pueden mejorar de la noche a la mañana, es necesario cubrir ciertas etapas para evitar efectos contraproducentes como la fatiga y las lesiones musculares. Aún en los casos en que el objetivo de los ciclistas sea el de únicamente mantener su forma física, deberán apegarse a un plan de entrenamiento que asegure la conservación de las capacidades actuales. Generalmente bastará con realizar una práctica regular que permita evitar un retroceso debido a un periodo largo de inactividad.

1.5.1. Modificación de las capacidades físicas

La realización de un trabajo físico parte de un nivel base de reservas energéticas y después de consumirse totalmente éstas se presenta la fatiga. Después de ella se debe iniciar el descanso para permitir la recuperación de las reservas de energía. Esta compensación de las reservas continua aún después de alcanzarse el nivel base inicial, con lo que se produce el fenómeno de *sobrecompensación*. Transcurrido cierto tiempo se reestablece el nivel base original [17] (figura 1.13). La duración de la sobrecompensación puede variar desde un par de días hasta algunas semanas.

Mantenimiento

Cuando se deja pasar suficiente tiempo desde el último entrenamiento el siguiente comenzará con el mismo nivel de base debido a que el proceso de sobrecompensación ha concluido. El practicar una vez por semana no representará un avance pero permitirá mantener el mismo nivel de las capacidades físicas.

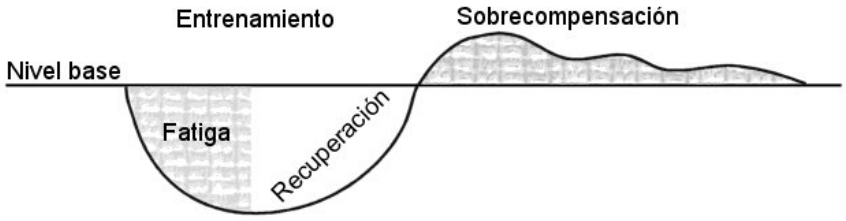


Figura 1.13: Fenómeno de sobrecompensación.

Avance

Los recursos del deportista durante la sobrecompensación son mayores que el nivel de base. Al comenzar un nuevo trabajo físico bajo estas condiciones provocará que la siguiente sobrecompensación alcance un mayor nivel de reservas.

Para que se observe un avance en las capacidades físicas es necesario establecer sesiones de entrenamiento que coincidan con el periodo de sobrecompensación de la práctica anterior [18] (figura 1.14). Con esto el nivel de base de las reservas energéticas aumentará progresivamente.

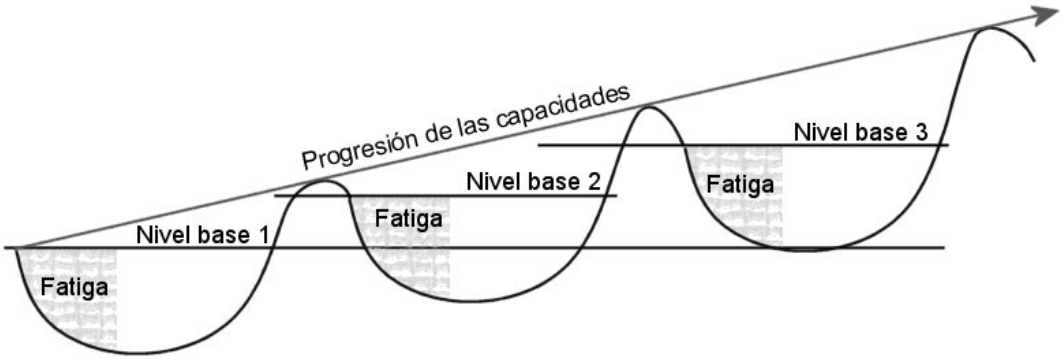


Figura 1.14: Progresión de las capacidades físicas por el aprovechamiento del periodo de sobrecompensación.

Retroceso

El fenómeno del retroceso en las capacidades físicas puede presentarse por dos causas. La primera se debe a largos periodos de inactividad en que el nivel de las reservas de energía disminuye. El organismo olvida fácilmente lo aprendido por lo que no se debe dejar pasar demasiado tiempo sin practicar. Los deportistas que sufren lesiones y paran por un tiempo ven disminuidas sus capacidades por lo que para retomar su nivel deben invertir mayor tiempo en entrenamientos y preparación.

La otra causa es el sobreentrenamiento. Si se vuelve a realizar trabajo físico cuando el organismo aún no se recupera del entrenamiento anterior (durante el periodo de recuperación) el nivel base al que se comienza es menor que el anterior. Si esto se repite varias veces se reducirán las capacidades físicas debido a un *sobreentrenamiento* [19] (figura 1.15). Cuando se perciban los efectos del sobreentrenamiento se debe parar por un tiempo para permitir que el organismo descanse y el nivel de reservas recupere su valor original.

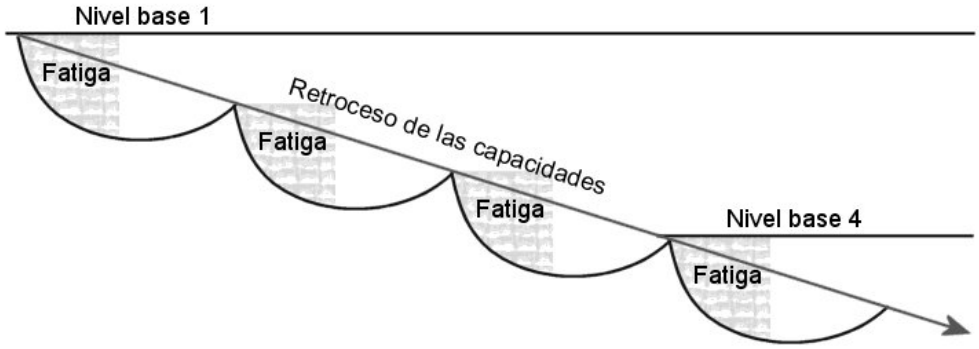


Figura 1.15: Sobreentrenamiento.

1.5.2. Capacidades ciclistas

Las capacidades ciclistas son los tipos de esfuerzo que se pueden efectuar en el ciclismo. Un ciclista puede tener mejor desempeño en alguno de ellos pero todos deben ser entrenados para asegurar un rendimiento general aceptable.

Fondo largo o ligero

Es un esfuerzo que se realiza en presencia de oxígeno (anaerobio). El organismo cuenta con grandes reservas energéticas y se pueden recuperar rápidamente. Durante el fondo se pedalea a bajas velocidades (entre 25 km/h y 35 km/h) y por largos periodos de tiempo. La frecuencia cardiaca se encuentra entre las 130 y las 160 pulsaciones por minuto y la respiración no presenta dificultades.

Fondo rápido o intenso

Este esfuerzo presenta una mayor intensidad pero se conserva dentro de la zona aerobia. Se realiza durante unos minutos (entre 10 y 20) y generalmente se combina con el fondo largo. Las pruebas contrarreloj se realizan bajo este tipo de esfuerzo pues se busca cubrir ciertas distancias en el menor tiempo posible. La velocidad a la que se rueda está entre los 35 y 40 kilómetros por hora y la respiración aún no presenta dificultades. La frecuencia cardiaca va de 160 pulsaciones por minuto y hasta un valor cercano a *200 menos la edad del ciclista*.

La persecución

Se llama así porque es el esfuerzo empleado cuando se trata de alcanzar a otro ciclista. La mayoría de las pruebas de distancia corta, como algunas de pista, se realizan bajo este tipo de esfuerzo, ubicado dentro de la zona anaeróbica láctica. Se alcanzan velocidades entre los 40 y 45 kilómetros por hora, aunque se mantienen por poco tiempo. Las reservas se consumen rápidamente y se produce ácido láctico con lo que se fatigan los músculos. Bajo estas condiciones la respiración aumenta considerablemente produciéndose una sensación de ahogo. La frecuencia cardiaca toma valores comprendidos entre *200 menos la edad del ciclista* y la *FCMax menos diez*.

El sprint

Es el esfuerzo realizado en los últimos metros de una carrera o en pruebas de pista muy cortas. Se alcanza la velocidad máxima (mayor a los 50 km/h) durante algunos instantes. El organismo no ocupa oxígeno y no se produce ácido láctico (anaerobio aláctico). Por el corto tiempo en que se puede mantener, pasa fácilmente al esfuerzo anaeróbico láctico donde sí se produce este ácido. Se pierde el control de la respiración y la frecuencia cardiaca está cercana a la FCMax.

El paseo

No es un esfuerzo sino una actividad de mantenimiento en la que las capacidades físicas no mejoran. No se entrena y por su baja intensidad es recomendable para todas las personas. La frecuencia cardiaca apenas aumenta desde el reposo aunque puede llegar a las 120 pulsaciones por minuto en ciertos periodos. La respiración no presenta dificultad y se puede platicar. La velocidad está por debajo de los veinte kilómetros por hora e incluso se llega al paro total en algunos momentos.

1.5.3. Periodos de transición

El organismo requiere de periodos de adaptación entre las condiciones de reposo total y de intensidad. No puede pasarse del reposo al esfuerzo y viceversa de manera brusca sino que debe hacerse de forma gradual.

Calentamiento

Al inicio del entrenamiento o de la práctica se debe rodar a baja velocidad (menos de 20 kilómetros por hora) por un espacio de 30 minutos o más para que tanto la respiración como la frecuencia cardiaca vayan aumentando poco a poco [20].

Vuelta a la calma

Al concluir un esfuerzo intenso se deben reducir el ritmo cardiaco y la respiración antes de parar totalmente. Una parada brusca puede provocar mareos, pérdida del equilibrio y en algunos casos problemas mas graves. Se debe continuar rodando de manera que la velocidad disminuya poco a poco hasta los 10 kilómetros por hora. Después de bajarse de la bicicleta es recomendable que el ciclista realice algunas flexiones y espere unos 30 segundos antes de sentarse [21].

1.6. Identificación de las necesidades básicas del ciclista

La primera condición que debe cumplir cualquier deporte es que su práctica no represente un riesgo para la salud. Ésta es la razón por la que los deportes extremos, el boxeo, las luchas y otros deportes de riesgo o de contacto son constantemente cuestionados. En el caso particular del ciclismo las necesidades se refieren a dos aspectos:

a) Obtener un beneficio por la práctica o el entrenamiento.
b) Evitar problemas de salud.

Para esto se deben tomar en cuneta las condiciones en que se practica el deporte, además de que ayuda el contar con accesorios electrónicos que permitan observar tanto el desempeño como el estado físico del corredor.

1.6.1. Control del desempeño físico

Para llevar a cabo un correcto plan de entrenamiento es necesario disponer de los datos de distancia y tiempo de manera que se puedan estudiar los efectos de cada sesión en la progresión de las capacidades. Las distintas cualidades ciclistas (sprint, persecución, fondo largo y fondo intenso) presentan características de distancia, duración, velocidad y frecuencia cardiaca que permiten al ciclista saber el tipo de esfuerzo que se está realizando.

Es deseable que el ciclista pueda programar sus sesiones limitando su duración y/o la distancia recorrida, además de controlar tanto su velocidad como su frecuencia cardiaca. Un ejemplo de esto es el calentamiento, en el que se debe rodar a bajas velocidades durante por lo menos 30 minutos para asegurar el acondicionamiento del organismo y su preparación para el esfuerzo. En este caso se debe conocer tanto el tiempo como la velocidad y es importante que esta última se mantenga por debajo de un valor límite.

Para asegurar el avance en las capacidades físicas se debe cuidar que el tiempo empleado para un mismo recorrido (pedaleando a la misma velocidad cada vez) disminuya o en el peor de los casos se mantenga. Un incremento de dicho tiempo es un síntoma de sobreentrenamiento. El organismo no estaría asimilando la rutina de entrenamiento actual, por lo que se debería cambiar.

Aparte de una mejora en la resistencia, se puede monitorear un mejor desempeño en potencia. Un avance en esta cualidad permite al ciclista alcanzar mayores velocidades tanto en fondo intenso (potencia aeróbica) como en sprint y persecución (potencia anaeróbica).

Como la frecuencia cardiaca va ligada a los diferentes tipos de esfuerzo, es un parámetro comúnmente empleado para ubicar el tipo de trabajo realizado. Algunos planes de entrenamiento establecen sesiones de cierto tiempo de duración o cierta distancia de recorrido a diferentes porcentajes de la frecuencia cardiaca máxima del ciclista. Los porcentajes indicados aseguran que el trabajo realizado esté dentro del esfuerzo que se desea entrenar.

1.6.2. Monitoreo del estado físico

Para prevenir problemas graves de salud, el ciclista debe estar pendiente de ciertos síntomas. La principal preocupación debe ser el cuidado del funcionamiento cardiovascular mediante un monitor de pulso cardiaco. Los dispositivos que incluyen esta función son demasiado costosos debido a que emplean elementos de instrumentación que suelen ser caros, por lo que no pueden ser adquiridos por la mayoría ciclistas que generalmente cuentan con recursos económicos limitados.

Es importante resaltar que un accesorio de éstos solo debe considerarse como una medida de monitoreo preventivo y nunca como un mecanismo de diagnóstico, pues no permite conocer con exactitud el estado del corazón. Si el monitor de pulso indica algún problema siempre se debe realizar un electrocardiograma completo para determinar su gravedad.

Al no ser un dispositivo de diagnóstico se pueden emplear técnicas diferentes que permitan reducir su costo. Los electrocardiogramas y la mayoría de los monitores de pulso (de muñeca o de pecho) observan mediante electrodos las diferencias de potencial eléctrico, entre diferentes puntos del cuerpo, y que son provocadas por la acción de bombeo del corazón. Estas señales eléctricas son de baja amplitud y van acompañadas de cierto nivel de ruido, por lo que requieren medirse empleando amplificadores de instrumentación.

Los monitores ópticos detectan la mayor o menor presencia de sangre, debida al bombeo del corazón, en partes del cuerpo que tienen cierta transparencia como el lóbulo de la oreja o la punta del dedo meñique. Se emplean en hospitales y tienen un menor costo que los monitores que emplean electrodos.

Es importante también que se pueda establecer un límite de pulsaciones por minuto ya sea para programas de entrenamiento, que establecen la máxima frecuencia a la que se debe trabajar, o para prevenir que se realicen esfuerzos que lleven la frecuencia cardiaca cerca de su máximo valor (FCMax). El caso más empleado es el último debido a que con esto se pueden prevenir problemas en el funcionamiento cardiovascular (infartos, aneurisma etc.) que suelen presentarse al exigirle demasiado al corazón.

Además del monitoreo del corazón se debe medir la temperatura corporal del ciclista ya que un aumento de ésta puede indicar problemas como la insolación o los golpes de calor. Es importante también conocer la temperatura ambiental a la que el ciclista está trabajando para establecer una referencia para la temperatura corporal. Si un ciclista tiene la sensación de frío puede deberse a que la temperatura externa es baja o a que su temperatura corporal ha aumentado considerablemente. La temperatura ambiental medida debe ser la que percibe directamente el ciclista al rodar, estableciendo con esto la misma referencia para ambos parámetros, pues en ambos casos la acción del viento interviene.

En los problemas derivados de la temperatura también interviene la humedad. El problema que se tiene en este caso es la interpretación de su nivel, pues no siempre que sea elevado se va a presentar una complicación. Además el costo de los sensores de humedad es considerablemente alto y es difícil justificar su inclusión. La humedad afecta principalmente el trabajo de la transpiración, por lo que se puede controlar su efecto sobre el ciclista cuidando que no se presente algún incremento anormal en la temperatura corporal.

1.6.3. Seguridad para el ciclista

Ya se ha mencionado que es muy común que los ciclistas tengan que convivir con automóviles. Los corredores pueden evitar accidentes al no hacerlo, pero pocas veces pueden acceder a alguna instalación adecuada para el ciclismo. Cuando no hay otra opción que correr junto a los automóviles se deben incrementar las precauciones, siendo indispensable el uso de casco y de sistemas de iluminación que permitan a los conductores percatarse de la presencia de la bicicleta.

El corredor debe cuidar sus trayectos para evitar circular demasiado cerca de los coches. Lamentablemente el ciclista depende de las medidas que a su vez tomen los conductores en presencia de bicicletas o peatones (reducción de la velocidad, ceder el paso etc.) y que comúnmente son pasadas por alto.

La elección de ciertos horarios para el entrenamiento (generalmente en las primeras horas del día) permite circular en compañía de pocos vehículos, pero generalmente la iluminación no es muy buena debido a que las sombras son largas y hay poca luz en los pasos a desnivel y en los túneles. La capacidad visual de ciclistas y conductores se vea limitada debido al tiempo que requiere el ojo para ajustarse al cambio en la intensidad luminosa y se presentan pequeños espacios de ceguera por lo que es importante contar con una iluminación que resalte la presencia de la bicicleta durante éstos.

El uso de indicadores luminosos de baterías es la medida más adecuada. Si estos indicadores prendieran de manera automática, dependiendo de la intensidad de la luz existente, se evitaría que el ciclista perdiera de vista su entorno al tener que encenderlos manualmente.

1.6.4. Resumen de requerimientos

Se enlistan a continuación los elementos con los que un ciclista debe contar para cubrir la mayoría de sus necesidades básicas de entrenamiento y salud física.

- Medición de la distancia recorrida.
- Medición del tiempo.
- Medición de la velocidad.
- Posibilidad de establecer límites para la distancia, el tiempo y la velocidad.
- Monitoreo de la frecuencia cardiaca.
- Posibilidad de establecer el límite de la frecuencia cardiaca.
- Monitoreo de la temperatura ambiental.
- Monitoreo de la temperatura corporal.
- Activación automática de un indicador luminoso en condiciones de poca luz.

Un accesorio que cuente con todos estos requerimientos permitirá al ciclista aprovechar mejor el ejercicio y disfrutar más de los paseos, por lo cual el propósito de este trabajo es diseñar un dispositivo que responda a estas necesidades manteniendo un bajo costo.

1.7. Referencias

1 Gérard Porte, *Guía general del ciclismo*, “El ciclismo: un deporte para todos, un deporte múltiple”, Madrid: Ediciones Tutor, 1996, pp. 21-22.

2 *Estatutos de la Unión Ciclista Internacional*, “Capítulo I .- Identidad - Fines”, trad., Real Federación Española de Ciclismo, Homepage: *Real Federación Española de Ciclismo*, bajado el 13 de mayo de 2003, http://www.rfec.com.

3 André Noret y Lucien Bailly, *El Ciclismo. Aspectos técnicos y médicos*, “Una forma particular de entrenamiento: la preparación de Eddy Merckx para el record del mundo bajo oxígeno rarificado”,Barcelona: Ed. Hispano Europea, 1991, pp. 78-84.

4 André Noret y Lucien Bailly, *Op. Cit.*, “Posición en la bicicleta”, pp. 385-391.

5 Gérard Porte, *Op. Cit.*, “Material, equipo, técnica”, p. 108.

6 *Ibíd.*, p. 112.

7 *Ibíd.* p. 113.

8 *Reglamento Generales del Deporte Ciclista*, “Capítulo III - Equipamiento”, trad., Real Federación Española de Ciclismo, Homepage: *Real Federación Española de Ciclismo*, bajado el 13 de mayo de 2003, http://www.rfec.com.

9 Leslie Cromwell (...) y otros, *Instrumentación y medidas biomédicas*, “El sistema cardiovascular”, trad., Ramón Pallás, Barcelona: Ed. Marcombo, 1980, p. 61.

10 *Ibíd*., p. 71.

11 *Ibíd*., “Instrumentación y medidas biomédicas”, p. 188.

12 Gérard Porte, *Op. Cit.*, “Entrenamiento para el ciclismo deportivo”, pp. 172-173.

13 Michel Delore, *Preparación y entrenamiento del ciclista. Guía práctica para todas las especialidades*, “Preparación biológica y mental”, Barcelona: Ed. Hispano Europea, 1998, p. 59.

14 Gérard Porte, *Op. Cit.*, “Los cuidados”, pp. 301-302.

15 Michel Delore, *Op. Cit.*, “Todos y todas en forma”, pp. 13-14.

16 Michel Delore, *Op. Cit.*, “Preparación biológica y mental”, p. 62.

17 Gérard Porte, *Op. Cit.*, “Entrenamiento para el ciclismo deportivo”, p. 161.

18 *Ídem.*

19 *Ibid.* p. 163.

20 Michel Delore, *Op. Cit.*, “Fije sus límites”, p. 33.

21 *Ídem.*

Capítulo II

Diseño del Dispositivo

2.1. Requerimientos del sistema

El dispositivo a diseñar debe responder a las necesidades básicas del ciclista (control y monitoreo del estado físico así como un incremento en su seguridad) descritas en la sección 1.6. Para lograrlo tiene que presentar ciertas características que pueden determinarse respondiendo los siguientes tres cuestionamientos.

2.1.1. ¿Qué va a hacer?

El dispositivo va a interactuar con el ciclista, la bicicleta y el ambiente entorno a ellos. Las funciones básicas que debe desempeñar fueron listadas en el apartado 1.6.4:

- Medición de la distancia recorrida.
- Medición del tiempo.
- Medición de la velocidad.
- Posibilidad de establecer límites para la distancia, el tiempo y la velocidad.
- Monitoreo de la frecuencia cardiaca.
- Posibilidad de establecer el límite de la frecuencia cardiaca.
- Monitoreo de la temperatura ambiental.
- Monitoreo de la temperatura corporal.
- Activación automática de un indicador luminoso en condiciones de poca visibilidad.

Proporcionará al usuario, el ciclista, la información básica sobre su desempeño y estado físico mediante la cuál se auxiliará para desarrollar correctamente la práctica y entrenamiento de su deporte y sin riesgos para su salud.

Básicamente funcionará como un crono-odómetro y velocímetro con funciones de monitoreo para la temperatura y la frecuencia cardiaca. El cálculo de los parámetros cinemáticos únicamente lo realizará cuando lo indique el usuario con los medios de entrada destinados a este propósito.

Las mediciones de temperatura, intensidad luminosa y frecuencia cardiaca se realizarán de forma continua. Mediante la comparación de éstos con sus valores máximos y mínimos permitidos determinará la activación de avisos auditivos y visuales que advertirán al usuario sobre condiciones de riesgo.

El valor de la velocidad será determinado independientemente de que el crono-odómetro esté detenido o funcionando. Sin embargo, el aviso que indicará que su valor es superior a un máximo determinado por el usuario se activará únicamente cuando esté trabajando.

2.1.2. ¿Cómo debe ser?

El aparato será un desarrollo de electrónica analógica y digital. Todas las actividades de control y de procesamiento se deberán desarrollar en el interior de un dispositivo capaz de controlar un proceso de manera autónoma, en este caso un sistema digital de computo, basado en un microcontrolador, empleando sus recursos internos disponibles.

Los medios de entrada acondicionarán las señales para que puedan ser compatibles con las entradas digitales y analógicas con que cuente el dispositivo digital. Además buscarán facilitar las operaciones realizadas por éste.

Deberá ser ligero y portátil para que pueda montarse en una bicicleta. Operará mediante baterías, por lo que su consumo de energía será pequeño. Tendrá la capacidad de funcionar con bicicletas de diferente talla adecuando su operación mediante el ajuste de un parámetro que indicará el tamaño de las ruedas de la bicicleta sobre la que esté montado.

Tendrá la capacidad de almacenar la información que resulte importante para su funcionamiento además de otros datos que resulten de interés para el usuario dentro de su memoria no volátil.

2.1.3. ¿Cómo va a funcionar?

Despliegue de información

El sistema presentará las siguientes lecturas:

- Tiempo transcurrido.
- Distancia recorrida.
- Velocidad actual.
- Frecuencia cardiaca.
- Temperatura ambiente.
- Temperatura corporal.
- Nivel de intensidad de la luz.

Excepto la última lectura, todas serán medidas cuantitativas por lo que se presentará su valor numérico. El nivel de intensidad luminosa será una medida cualitativa que indicará que tan buena o mala es la iluminación por medio de niveles.

Las lecturas de tiempo y distancia se modificarán únicamente al activar el crono-odómetro. Mediante un caracter se indicará si está activo el límite de tiempo y/o de distancia. Además se mostrará el valor límite para la velocidad fijado por el usuario y mediante un símbolo se indicará si está habilitado.

Las lecturas restantes se actualizarán de forma periódica independientemente del estado de funcionamiento del crono-odómetro. El dispositivo activará indicadores visuales y avisos sonoros cuando se presenten ciertas condiciones y los mantendrá mientras continúen.

Condiciones para la activación de indicadores visuales y auditivos

- Cuando la velocidad sea mayor al límite establecido siempre y cuando se encuentre activado y el crono-odómetro esté funcionando.
- Cuando la frecuencia cardiaca sea mayor al valor de frecuencia máxima establecido por el usuario. Esto se verificará siempre que el dispositivo esté en operación normal.
- Cuando la batería esté baja y deba ser reemplazada. Esto se verificará siempre que el dispositivo esté encendido.

En cada caso el indicador visual y la característica sonora de la alarma auditiva serán diferentes. Cuando el nivel de iluminación sea muy bajo se activará el indicador luminoso y no contará con alarma audible para este caso. Cuando el crono-odómetro se detenga de forma automática al llegar a alguno de sus límites (distancia o tiempo) se producirá un sonido de corta duración para indicarlo.

Acciones a realizar con el crono-odómetro detenido

Cuando el crono-odómetro no esté funcionando el usuario podrá:

- Activar el crono-odómetro.
- Poner en ceros las lecturas de distancia y tiempo.
- Almacenar en memoria EEPROM la lectura actual de distancia y tiempo.
- Recuperar de la memoria EEPROM los valores almacenados de tiempo y distancia.
- Pasar al modo de configuración para modificar los límites y parámetros de funcionamiento.

Es importante mencionar que el crono-odómetro no deberá comenzar a funcionar inmediatamente después de ser activado, sino hasta que la bicicleta comience a avanzar.

Acciones a realizar con el crono-odómetro funcionando

Una vez que el crono-odómetro comience a funcionar, el usuario podrá realizar las siguientes operaciones:

- Detener el corno-odómetro.
- Activar o desactivar el límite de velocidad.
- Modificar el valor del límite de velocidad.

El sistema detendrá de forma automática el crono-odómetro cuando la lectura de tiempo y/o la de distancia sea mayor que su valor límite, siempre y cuando esté habilitado.

Modo de configuración

En este modo de operación se presentarán los siguientes valores modificables:

- Límite de tiempo.
- Indicador de activación del límite de tiempo.
- Límite de distancia.
- Indicador de activación del límite de distancia.

- Frecuencia cardiaca máxima.
- Valor de la rodada de la bicicleta en que está montado el dispositivo.

Al entrar en él se detendrán todas las mediciones, reanudándose cuando se regrese a la operación normal. Si el usuario no realiza ninguna modificación durante diez segundos, se retornará automáticamente a este modo..

Solo se modificará un valor a la vez y mediante una cadena de texto se indicará cuál es el parámetro en turno. Antes de salir del modo de configuración se almacenarán los nuevos valores en la memoria EEPROM.

Datos almacenados en memoria no volátil

Los valores que se guardarán en la memoria EEPROM para que estén disponibles independientemente de que se retire la alimentación serán los siguientes:

- Frecuencia cardiaca máxima.
- Rodada de la bicicleta utilizada.
- Límite para el tiempo.
- Límite para la distancia.

Además se podrá almacenar una lectura de tiempo y de distancia para que se pueda llevar un valor acumulado para recorridos efectuados en varias sesiones.

2.2. Intervalos de operación

Con el propósito de optimizar los recursos de hardware y software se limitará la operación del dispositivo a los intervalos de variación más comunes para los diferentes parámetros que intervienen en su funcionamiento.

2.2.1. Rangos de operación para valores de tiempo

Las sesiones de práctica, entrenamiento o competencia en el ciclismo suelen ser largas, excepto en algunas pruebas de pista. La duración de éstas puede variar desde algunas decenas de minutos hasta un par de horas en paseos largos o competencias por etapas.

El valor máximo de operación deberá ser de varias horas pues con esto se permitirá acumular varias sesiones cortas o realizar una sola sesión larga. Un máximo de diez horas empleando un solo dígito funcionará debido a que un solo recorrido difícilmente será tan largo y se pueden acumular varias sesiones de un par de horas de duración.

Para acumular valores mayores se podrán emplear otros medios para llevar el registro de los tiempos parciales y no rebasar el tiempo máximo que pueda almacenar el dispositivo.

En cuanto a la resolución, bastaría con manejar incrementos de un minuto pues los recorridos son mucho más largos que este valor. Sin embargo esta práctica no es muy común en los cronómetros deportivos. Éstos suelen contar con una resolución de por lo menos un segundo para eventos de larga duración y de centésimas de segundo para eventos de corta duración.

Las características temporales descritas también se aplican al límite de tiempo para los recorridos. Únicamente se harán las siguientes consideraciones:

- Al ser tiempos empleados en una sola sesión su valor máximo puede reducirse.
- Se limita el incremento durante su modificación por parte del usuario a cinco minutos para facilitar el cambio de un valor pequeño a uno cercano al máximo.

Parámetro	Rango (horas)	Incremento mínimo
Duración del recorrido	0 - 9:59:59	1 segundo
Límite de tiempo	0 - 3:55:00	5 minutos

Tabla 2.1: Rangos de operación para tiempo.

2.2.2. Rangos de operación para valores de distancia

Exceptuando algunas pruebas de pista, las distancias recorridas en bicicleta se caracterizan por ser largas (generalmente mayores a un kilómetro). Las distancias más grandes se presentan en paseos cicloturistas, competencias por etapas y de triatlón, llegando a ser mayores a los doscientos kilómetros.

En este caso también deberá ser posible acumular las distancias recorridas en varias sesiones. Si se considera un valor límite de mil kilómetros, se podrían acumular hasta tres sesiones largas de trescientos kilómetros y para sumar mayores distancias se podrá recurrir a otros medios de registro.

En cuanto a la resolución, si se incrementara el odómetro cada kilómetro no se podría observar el desempeño en distancias intermedias (1.5 Km por ejemplo). En el otro extremo, si los incrementos fueran del orden de los metros o las decenas de metros, las lecturas variarían demasiado rápido haciendo difícil su lectura.

A final de cuentas, la resolución de esta lectura quedará establecida por el método empleado para realizar el cálculo. En cualquier caso se presentarán al usuario únicamente incrementos mínimos de cien metros por las razones expuestas en el párrafo anterior.

En cuanto al límite de distancia se considerará como valor máximo el de 295 kilómetros y un incremento mínimo de 5 kilómetros al modificar su valor, por razones similares a las expuestas para el límite de tiempo.

Parámetro	Rango (kilómetros)	Incremento mínimo
Distancia recorrida	0 - 999.9	100 metros
Límite de distancia	0 - 295	5 kilómetros

Tabla 2.2: Rangos de operación para tiempo.

2.2.3. Rangos de operación para valores de velocidad

Las características de velocidad de los diferentes tipos de trabajo en ciclismo se indican en el apartado 1.5.2. Las mayores velocidades se presentan durante el *sprint* y pueden ser mayores

a los cincuenta kilómetros por hora (tabla 2.3). Sin embargo, podría llegarse a una velocidad superior a los noventa kilómetros por hora al rodarse cuesta abajo.

Tipo de trabajo	Velocidad [km/h]	
El paseo	Menores a 20	
Fondo largo o ligero	25	35
Fondo rápido o intenso	35	40
La persecución	40	45
El sprint	Mayores a 50	

Tabla 2.3: Características de velocidad de los distintos tipos de trabajo.

La velocidad es una medición indirecta que dependerá del registro de la distancia recorrida en cierto intervalo de tiempo. Para obtener una buena aproximación a la velocidad instantánea, éste debe ser lo más pequeño posible. De cualquier forma la resolución quedará determinada por el valor mínimo de distancia que se pueda medir y por el mínimo intervalo de tiempo que se maneje.

Para la aplicación que se está desarrollando se considerará un incremento mínimo de un kilómetro por hora para mostrar al usuario, aunque en los cálculos realizados se ocupe un valor menor. Este último quedará determinado por el método a emplear para calcular la distancia pues la resolución del tiempo ya se ha establecido en un segundo.

Si se emplea un byte para almacenar el valor de la velocidad, ésta podría ser de hasta 255 km/h. De esta forma queda establecido el valor máximo que podrá manejar el dispositivo y está bastante sobrado pues es prácticamente imposible alcanzar dicho valor sobre una bicicleta.

Para el límite de la velocidad se considerará un valor máximo de 95 km/h y un mínimo de 10 km/h, mayor y menor respectivamente a cualquier valor de velocidad mencionado en el apartado 1.5.2. Los incrementos al modificar su valor serán de 5 km/h.

Parámetro	Rango [km/h]	Incremento mínimo
Velocidad actual mostrada al usuario	0 - 255	1 km/h
Límite de velocidad	10 - 95	5 km/h

Tabla 2.4: Rangos de operación para velocidad.

2.2.4. Rangos de operación para valores de frecuencia cardiaca

En el apartado 1.4.1 se establece que la frecuencia menor se presenta durante el reposo, siendo de entre sesenta y ochenta pulsaciones por minuto. También se menciona que puede disminuirse mediante la práctica constante de un deporte y el valor mínimo que se puede alcanzar dependerá de las cualidades físicas de cada individuo.

La ecuación 1.4 determina el valor máximo que puede alcanzar la frecuencia cardiaca en cada individuo y se observa que nunca podrá ser mayor a las 220 pulsaciones por minuto. La resolución de la frecuencia será de un pulso por segundo debido a que el pulso es una cantidad discreta.

Con respecto a la frecuencia cardiaca máxima se considerará un valor superior de 210 que se asocia con una persona de diez años y un valor mínimo de cien pulsaciones por minuto para permitir colocar al límite de la frecuencia cardiaca por debajo de la FCMax. Los incrementos al momento de modificar su valor serán de 5 pulsos para facilitar el cambio entre un valor pequeño y uno grande.

Parámetro	Rango (plusos por minuto)	Incremento mínimo
Frecuencia cardiaca	0 - 220	1 ppm
Frecuencia cardiaca máxima (FCMax)	100 - 210	5 ppm

Tabla 2.5: Rangos de operación para frecuencia cardiaca.

2.2.5. Rangos de operación para valores de temperatura

La temperatura ambiente puede tener valores un poco menores a 0 ºC y hasta mayores a 60 ºC dependiendo de la ubicación geográfica y de la temporada. En este caso el rango de operación y la resolución dependerán del método empleado para realizar la medición.

Para el presente diseño bastará con que se puedan medir temperaturas dentro de un intervalo de valores adecuados para la práctica de cualquier deporte al aire libre (entre 15 ºC y 40 ºC). El máximo incremento que se presentará al usuario será de 1 ºC pues es la resolución de los termómetros de mercurio que son los más conocidos.

En cuanto a la temperatura corporal, se considerará que puede variar entre los 28 y 38 grados Celsius en condiciones normales y entre los 20 y 44 ºC durante situaciones extremas de hipotermia y fiebre [1]. En aplicaciones médicas se requieren resoluciones menores a medio grado Celsius, pero considerando que el dispositivo a diseñar no se empleará para diagnóstico (apartado 1.6.2) bastará con una resolución interna de este valor. Al igual que con la temperatura ambiente, el mínimo incremento que se le presentará al usuario será de un grado Celsius.

Parámetro	Rango [ºC]	Incremento mínimo
Temperatura ambiente	0 - 60	1 ºC
Temperatura corporal	20 - 44	1 ºC

Tabla 2.6: Rangos de operación para temperaturas.

2.2.6. Rango de operación para la intensidad de la luz

Esta lectura es cualitativa pues se establecerán intervalos de comparación para determinar su nivel con respecto a los valores máximos y mínimos que se puedan medir. Estos límites quedarán establecidos por el método de medición y a partir de ellos se determinarán los intervalos mencionados.

2.3. Definición de entradas y salidas

Las entradas proporcionarán la información necesaria para realizar las acciones para las que será programado el dispositivo a desarrollar. Estas acciones serán tanto mediciones como

instrucciones de control que en conjunto permitirán determinar el procedimiento a seguir para generar las salidas requeridas.

2.3.1. Entradas

Intensidad de la luz

Permitirá determinar el momento en que se activará o desactivará el indicador luminoso de seguridad.

Giro de la rueda

Permitirá conocer la distancia recorrida pues con cada vuelta se avanza una distancia igual al perímetro de la rueda (ver tabla 1.4). También permitirá calcular de forma indirecta la velocidad.

Base de tiempo

Mediante una señal periódica de duración fija y conocida , el dispositivo será capaz de llevar una cuenta del tiempo transcurrido entre dos instantes diferentes. También definirá el intervalo de medición para el cálculo de la velocidad.

Pulso cardiaco

Mediante una cuenta de los pulsos que se presenten en un intervalo de tiempo determinado se podrá calcular la frecuencia cardiaca del ciclista.

Temperatura ambiente y corporal

Proporcionarán al sistema información sobre el estado físico del individuo.

Instrucciones de control

Servirán para controlar el funcionamiento del dispositivo transfiriendo a éste las instrucciones del usuario.

2.3.2. Salidas

Información para el usuario

Se conforma de los resultados de los cálculos y mediciones realizadas por el sistema digital de cómputo y de la información relacionada con su estado actual de operación.

Alarma auditiva

Advertirá al usuario sobre ciertas condiciones de funcionamiento sin que tenga que estar observando el dispositivo en todo momento.

Indicador luminoso de seguridad

Permitirá advertir a los conductores de vehículos automotores y de otras bicicletas sobre la presencia de la bicicleta en los momentos en que las condiciones de luz pudieran disminuir la capacidad visual de ellos.

2.4. Definición de los módulos del sistema

2.4.1. Medios de entrada

Los medios de entrada serán los elementos encargados de realizar las modificaciones necesarias a las señales de entrada para que éstas sean compatibles con las terminales de entrada del microcontrolador que se utilice.

Las señales pueden ser pulsos digitales, niveles lógicos de voltaje o señales analógicas con amplitudes compatibles con algún convertidor analógico-digital. La tabla 2.7 indica el medio de entrada asociado a cada señal que ingresa al sistema. Además se indica el tipo de señal entregado al microcontrolador.

Figura 2.1: Propósito de los medios de entrada.

Medio de entrada	Señal de entrada	Señal entregada al microcontrolador
Sensor de giro	Giro de la rueda	Pulso digital cada vez que se presente
Sensor de intensidad luminosa	Intensidad de la luz	Valor de voltaje
Sensores de temperatura	Temperatura ambiente y corporal	Valor de voltaje
Sensor de pulso	Pulso cardiaco	Pulso digital cada vez que se presente
Oscilador externo	Base de tiempo	Señal de frecuencia conocida
Interruptores	Instrucciones de control	Niveles lógicos de voltaje

Tabla 2.7: Medios de entrada.

2.4.2. Medios de salida

El propósito de los medios de salida es el de hacer que las señales de salida y/o de control generadas por el microcontrolador puedan ser entendidas por loe elementos a gobernar o por los usuarios del sistema.

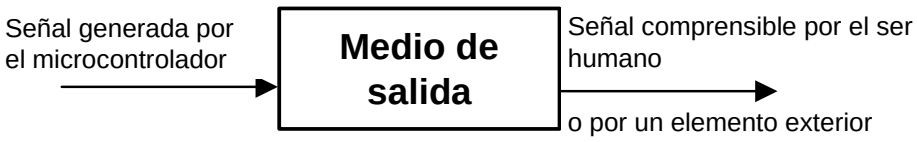


Figura 2.2: Propósito de los medios de salida.

Por lo general los microcontroladores únicamente pueden generar señales digitales en sus salidas. Mediante sus puertos de salida puede representar cantidades discretas de información

o valores lógicos empleando líneas individuales. La tabla 2.8 relaciona cada salida con su medio de salida y con la forma de las señales con que trabajan.

Medio de salida	Señal de salida	Forma inicial	Forma modificada
Dispositivo para despliegue de información	Información para el usuario	Valores lógicos y discretos en binario	Caracteres alfanuméricos y símbolos
Generador de sonido	Alarma auditiva	Señal digital con cierta frecuencia	Sonido de ciertas características
Indicador luminoso	Activación del indicador luminoso	Valor lógico	Encendido o apagado

Tabla 2.8: Medios de salida.

2.4.3. Diagrama a bloques del sistema

Además de los medios de entrada y salida, el sistema también incluirá al microcontrolador, su señal de reloj y la fuente de alimentación. Las señales analógicas de la tabla 2.7 entrarán al microcontrolador por medio de un convertidor analógico-digital.

Aunque se plantea la utilización de un oscilador externo para la base de tiempo, ésta podría generarse en el interior del microcontrolador a partir de su señal de reloj. Durante el desarrollo del hardware se definirán más detalladamente los elementos que conformarán al sistema.

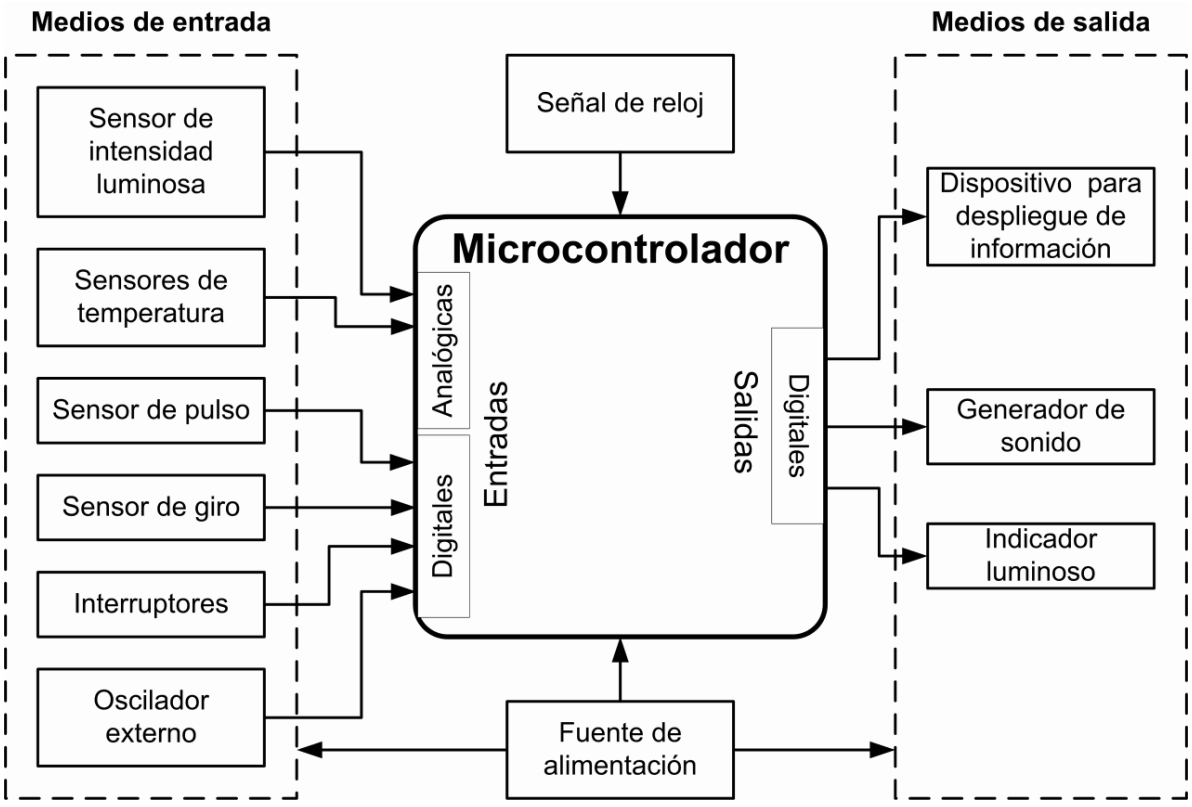


Figura 2.3: Diagrama a bloques del sistema.

2.5. Operación del dispositivo

2.5.1. Diagrama de flujo

El diagrama de flujo de la figura 2.7 describe en forma general el funcionamiento del sistema e incluye los procedimientos básicos que se describen en los siguientes puntos. Durante el desarrollo del software se implementarán las rutinas que conformarán estos procedimientos generales.

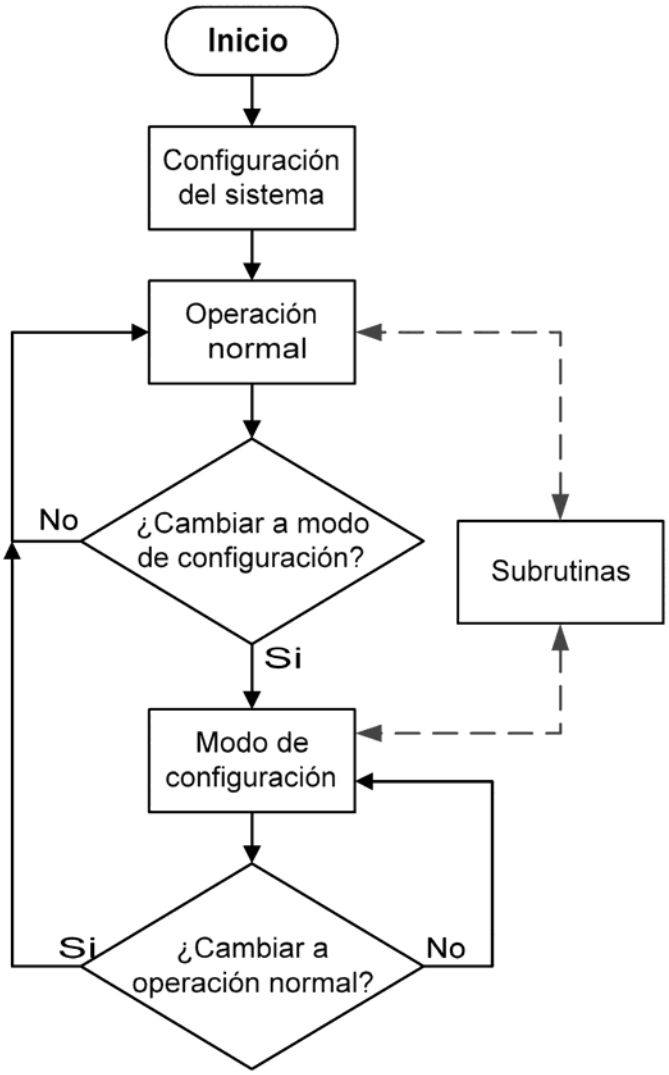


Figura 2.7: Diagrama de flujo del algoritmo general de operación.

2.5.2. Preparación del sistema

Se conformará por todos los procedimientos necesarios para preparar al sistema digital de cómputo para que pueda trabajar con las señales de entrada y que sea capaz de generar las señales de salida. Básicamente constará de tres procedimientos principales que deberán ejecutarse inmediatamente después de comenzar la operación del dispositivo:

1. Asignación de memoria y de terminales de entrada y salida.
2. Configuración de los recursos internos que se utilizarán.
3. Iniciación de variables y recuperación de parámetros.

2.5.3. Operación normal del sistema

Se desarrollarán los procedimientos adecuados para cumplir con los requisitos de funcionamiento del apartado 1.1.3. Se agruparán de la siguiente forma:

a) Procedimientos de cálculo y medición.
b) Procedimientos de generación de salidas.
c) Procedimientos de control.

2.5.4. Modificación de parámetros

Mediante estos procedimientos se realizarán las modificaciones a los parámetros empelados por el programa y que se almacenan en memoria no volátil. Conformarán el modo de configuración establecido en el apartado 1.1.3.

2.5.5. Subrutinas

Serán aquellos procedimientos de propósito general que podrán ejecutarse en cualquier lugar del programa. Tienen el objetivo de optimizar la memoria de programa al evitar la duplicación de procedimientos.

2.6. Referencias

1 Jorge Rodríguez, *Sistema de biomonitoreo médico personal*, “Métodos de medición utilizados”, Universidad Nacional Autónoma de México, Tesis de Maestría, México: El autor, 1994, p. 12.

Capítulo III

El Microcontrolador

3.1. Definición

Los controladores son dispositivos diseñados para determinar el funcionamiento de un sistema, estableciendo señales de salida que dependen tanto de las señales de entrada como de determinados procesos. Su implementación puede ser analógica o digital, siendo esta última la que involucra a los microcontroladores.

La evolución tecnológica ha llevado la implementación física de los controladores discretos desde circuitos de compuertas lógicas hasta los sistemas de mediana y alta escala de integración, que derivaron en los microprocesadores y finalmente a los microcontroladores. Los microprocesadores surgieron como respuesta a la necesidad de una plataforma de hardware dinámica que pudiera ser modificada y acoplada a diferentes sistemas.

Cuando el nivel de integración de los dispositivos semiconductores permitió incorporar algunos los elementos que acompañaban a los microprocesadores (principalmente la memoria) dentro de un solo circuito integrado surgieron los microcontroladores.

La primera familia de éstos fue la TMS1000 de *Texas Instruments* con microprocesador de cuatro bits. Posteriormente *Intel* introdujo el 8048 que contenía un microprocesador de ocho bits junto con memoria RAM, ROM y algunos puertos de entrada y salida dentro de un solo encapsulado [1].

Los microcontroladores son circuitos integrados programables de alta escala de integración que contienen los elementos básicos de una microcomputadora dentro del mismo encapsulado y están enfocados a realizar una sola tarea. Cuentan con cantidades limitadas de memoria y disponen de recursos periféricos adicionales para control de elementos externos. Mediante distintas combinaciones de microprocesador, memoria y periféricos, se forman las *familias de microcontroladores* [2].

Los microcontroladores son sistemas digitales que procesan información discreta[1] representada por medio de dos niveles de voltaje que se ubican dentro de alguno de los dos intervalos de valores que definen al voltaje *alto* y *bajo*. Los valores discretos que representan estos intervalos son *1* y *0* que conforman la numeración binaria. Mediante varios de estos elementos llamados *bits* se representan cantidades también discretas de mayor valor [3].

[1] Aquella información que es representada por un número finito de elementos.

3.1.1. Componentes básicos

La figura 3.1 presenta los componentes de un sistema digital de cómputo implementado mediante un microcontrolador. También se puede implementar con un microprocesador, pero estos no incluyen los otros elementos dentro del mismo encapsulado. La arquitectura interna de los microcontroladores es fija por lo que los recursos están limitados a los disponibles en su interior. A pesar de esto, en algunos casos los fabricantes permiten expandir su capacidad utilizando algunas terminales para acoplar circuitos externos de memoria y recursos periféricos.

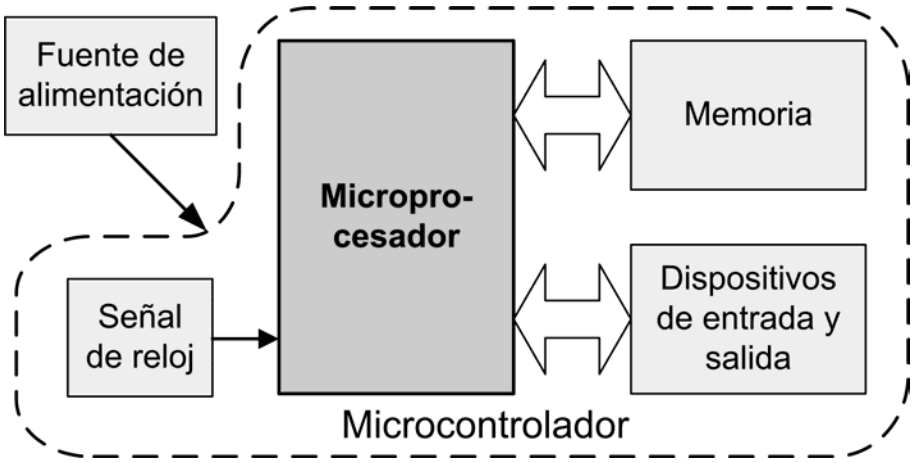


Figura 3.1: Componentes de un Microcontrolador.

Los elementos que comúnmente se encuentran dentro de los microcontroladores son:

- Procesador
- Memoria de datos y de programa
- Dispositivos de entrada y salida
- Circuito de reloj

Se fabrican diferentes modelos de microcontroladores con distintas capacidades de memoria y de recursos de entrada y salida por lo que se puede escoger el más adecuado para cada aplicación. Algunos incluyen otros recursos, como los temporizadores o los convertidores analógico-digital, que permiten que todas las necesidades del controlador se encuentren dentro de un solo encapsulado.

La fuente de alimentación y algunos componentes requeridos para generar la señal de reloj son externos al microcontrolador. El voltaje debe ser de corriente directa, comprendido en cierto rango que depende del microcontrolador, pero debe mantenerse constante para no alterar el funcionamiento del sistema. Cada microcontrolador presenta también diferentes requerimientos en cuanto a los componentes externos que necesita el circuito de reloj.

3.1.2. Clasificación

La principal clasificación de los microcontroladores es el número de bits de la palabra que manejan. Esta palabra define la máxima magnitud de las cantidades discretas que puede emplear su Unidad Lógica Aritmética, presentada más adelante. Es una característica que no

se puede modificar pues está definida en el hardware y por ello existen microcontroladores de 4, 8, 16 y 32 bits.
Una palabra es una entidad de bits que constituyen una unidad para representar información codificada en binario como un número, una instrucción o un carácter alfanumérico. Se emplean cantidades de 2^n siendo la palabra mas común la de ocho bits llamada *byte* [4].

3.1.3. Arquitectura

La arquitectura de los microcontroladores determina la forma en que se conecta el procesador y la unidad de control con la memoria de datos y la de programa [5]. La conexión se hace mediante buses[1] de control, dirección y datos. El bus de dirección determina la localidad de memoria a la que se quiere acceder, y el bus de datos está dedicado a la transmisión de los bits de información. El bus de control determina el elemento en que se encuentra la dirección deseada y controla la transferencia y recepción de los datos.

Arquitectura von Neumann

Existe una sola memoria que contiene datos e instrucciones y se emplea un solo conjunto de buses. Las instrucciones y los datos se transfieren por la misma ruta y pueden ocupar las mismas localidades de memoria.

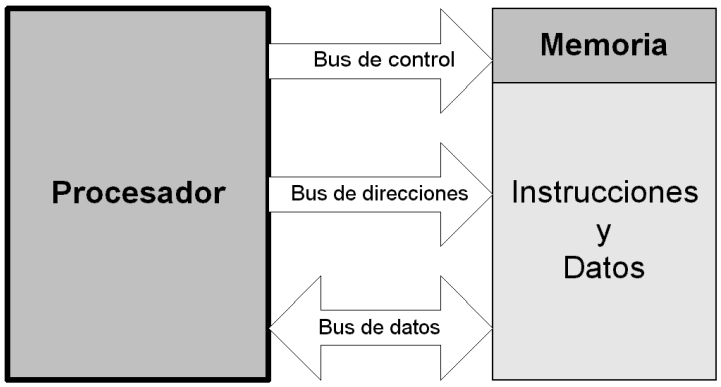
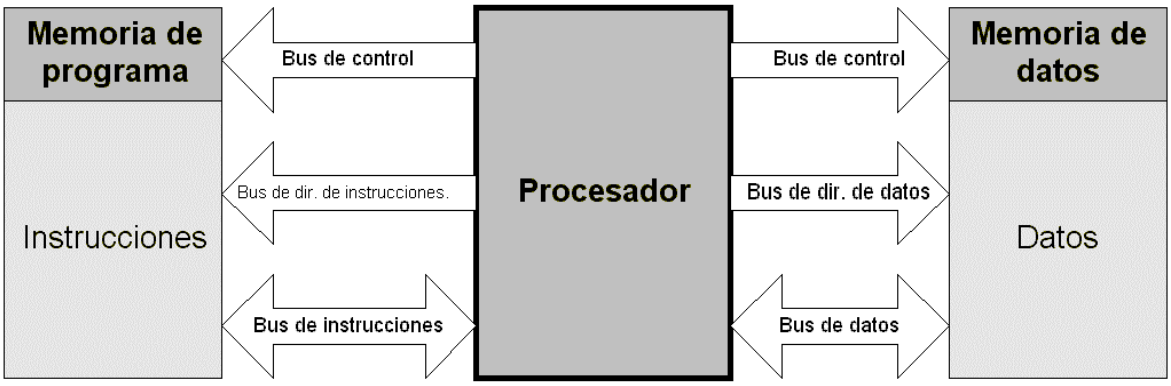


Figura 3.2: Arquitectura von Neumann.



[1] Un bus es un conjunto de líneas con una ruta de transferencia común.

Figura 3.3: Arquitectura Harvard.

Arquitectura Harvard

Las memorias de datos y de programa son independientes y cada una cuenta con su propio conjunto de buses. Los datos y las instrucciones pueden ser de diferente número de bits. Al no compartir la misma ruta de transferencia el procesador puede disponer al mismo tiempo de una instrucción y de un dato en memoria.

Otras arquitecturas

Aunque las arquitecturas anteriores son las mas comunes en microcontroladores, existen otras opciones que han surgido para tratar de hacer el acceso a memoria mas rápido y confiable [6]. Entre ellas están las *Memorias asociativas* que permiten encontrar un dato de memoria verificando todas las direcciones al mismo tiempo. Las *memorias jerarquizadas* emplean tres niveles de memoria con diferente velocidad de acceso, lo que permite que los datos que más se ocupan se puedan acceder rápidamente desde una memoria *cache*. Las *memorias intercaladas* dividen la memoria en bancos y las *memorias segmentadas* la dividen en segmentos destinados a almacenar las instrucciones, los datos y la pila. Otro ejemplo son las memorias etiquetadas que marcan cada localidad de memoria indicando el tipo de datos que almacena, evitando por ejemplo que un dato sea tomado como instrucción.

Aunque algunas de estas nuevas técnicas de organización de la memoria están disponibles en los microcontroladores, como la memoria de datos intercalada en bancos de registros de los microcontroladores PIC, están mas enfocadas a optimizar el manejo de memoria en computadoras de altas prestaciones como las PC o los servidores.

3.2. El procesador

Es el elemento principal del microcontrolador, se encarga de ejecutar las instrucciones del programa almacenado en memoria y del manejo de los demás componentes. Se le llama también *Unidad Central de Procesamiento* o CPU por sus siglas en inglés.

3.2.1. Bloques internos

Modelo de programación

Es un modelo abstracto del los registros mas importantes del microcontrolador y proporciona la información básica sin adentrarse demasiado en el hardware. Es de bastante utilidad para familiarizarse con el funcionamiento interno de un microcontrolador y para facilitar la escritura de programas [7]. La figura 3.4 presenta un ejemplo de modelo de programación.

Unidad Lógica Aritmética (ALU)

Es un circuito de lógica combinacional[1] que ejecuta un número determinado de operaciones básicas (aritméticas y lógicas) sobre uno o dos datos de entrada [8]. El número de líneas de selección de la operación a realizar (k) determinará el número de operaciones disponible en la ALU (2^k).

[1] Circuitos en que las variables se combinan mediante operaciones lógicas.

La figura 3.5 representa el diagrama general de una ALU de *n* bits de entrada, *k* líneas de selección de operación y 2^k operaciones diferentes [9]. Los datos de entrada son procesados mediante la lógica interna destinada a cada operación que se seleccione y el resultado se presenta en las líneas de salida. Los acarreos de entrada y salida se emplean para operaciones aritméticas.

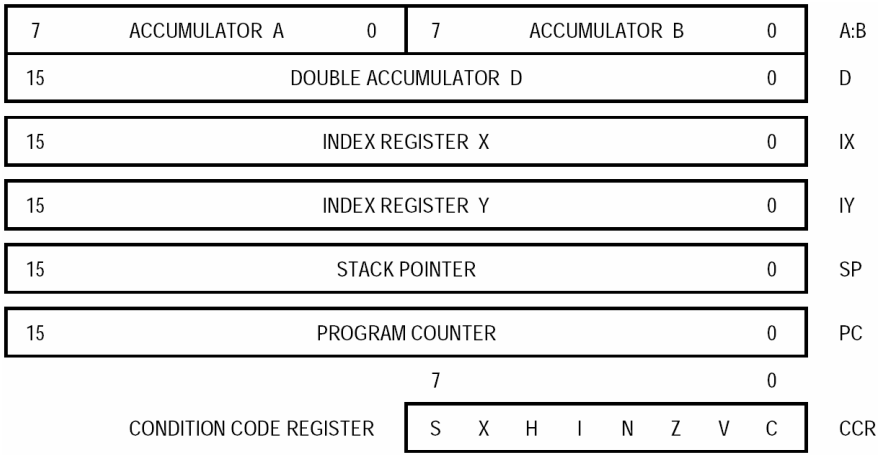


Figura 3.4: Modelo de programación del microcontrolador MC68HC11 de Motorola tomado de *M68HC11 Reference Manual*, Motorola 1991.

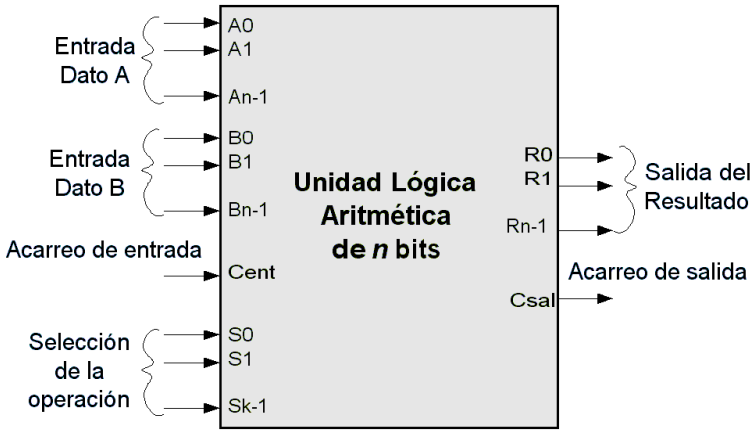


Figura 3.5: Diagrama general de una ALU.

Registros

Los registros almacenan datos temporales que el procesador modifica constantemente y algunos de ellos funcionan como entradas de datos para la ALU.

Otros registros relevantes son el registro de dirección de memoria (MAR) y el registro buffer de memoria (MBR). El primero almacena la dirección de memoria que se va a leer o escribir y este valor es decodificado para activar la celda de memoria. Durante una operación de lectura, el valor de dicha celda es colocado en el MBR, y durante una operación de escritura es el valor almacenado en el MBR el que se copia en la celda [10].

Decodificador de instrucciones

Determina a partir de la instrucción almacenada en el registro de instrucción (IR) , la operación a realizar [11]. Para ello establece, mediante señales de control, las rutas para los datos que se emplearán.

Pila

Es un conjunto de registros usados para almacenar direcciones de la memoria de programa y otros valores de utilidad. En algunos casos la pila se implementa en memoria de datos y un registro almacena la dirección del siguiente espacio vacío de la pila.

Ya sea que se implemente mediante registros o en memoria de datos, la pila trabaja bajo el principio de *el último en entrar es el primero en salir*. El procesador emplea la pila para realizar saltos dentro del programa y recordar el punto de origen. De igual forma sirve para almacenar el estado de los demás registros al momento de ejecutar el salto de manera que al regresar al origen se reestablezcan todos los valores.

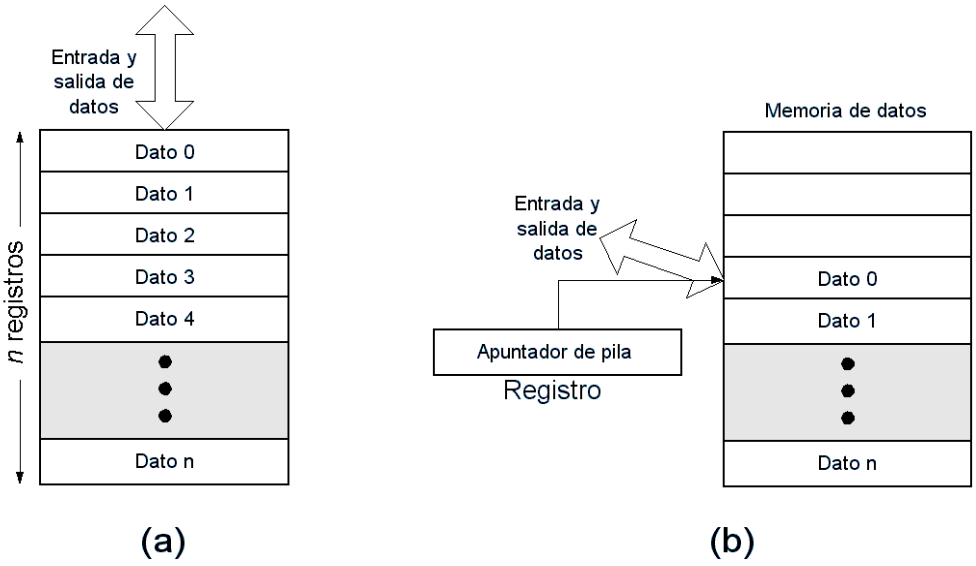


Figura 3.6: Implementación de la pila mediante (a) varios registros que almacenan la información directamente; (b) Un solo registro que almacena la dirección de la memoria de datos en que está el nivel superior de la pila.

Contador de programa

Es un registro que almacena la dirección en memoria de la siguiente instrucción a ejecutar. Tras cada ejecución incrementa su valor en uno, excepto en las instrucciones de salto, en que se carga un valor nuevo. El valor del contador de programa se almacena en la pila antes de realizar un salto. Su tamaño depende del número de bits necesarios para representar todas las direcciones disponibles de la memoria de programa.

Señales de control

Son señales que determinan el funcionamiento del sistema, como la señal del reloj y los indicadores de interrupción.

Buses

Son las líneas que comunican los elementos internos del procesador con la memoria. Existen tres tipos de buses, el de control, de direcciones y de datos.

3.2.2. Ciclo de instrucción

El *ciclo de instrucción* abarca todos los pasos que deben ejecutarse para completar la operación de una instrucción. La unidad de control del procesador debe pasar a través de ciertos estados para que esto se lleve a cabo, dividiéndose en dos fases principales [12].

Durante la fase de búsqueda o *ciclo fetch* la unidad de control envía las señales necesarias para obtener de la memoria el código de la instrucción y sus operandos. Posteriormente, y una vez que se ha completado la fase de búsqueda, el procesador entra en la *fase de ejecución* en la que la unidad de control envía las señales necesarias para que la ALU efectúe la operación.

Las señales que envía la unidad de control en ambas fases están destinadas a efectuar transferencias de información entre distintos componentes del microcontrolador, y se efectúan de acuerdo al *ciclo de reloj* que procede de la señal del reloj del sistema.

Si embargo, dichas señales deben sincronizarse para que la información se tenga en el instante adecuado pues las señales no llegan inmediatamente a su destino y los elementos del microcontrolador pueden tardar en responder.

El *ciclo máquina* se compone de un determinado número de ciclos de reloj de forma que cada paso del ciclo de instrucción se lleve de forma adecuada al sincronizarse las señales que envía la unidad de control. La frecuencia real de operación de un microcontrolador está determinada por el ciclo máquina y es por esto que generalmente es una fracción de la señal de reloj [13].

3.2.3. Instrucciones

Las instrucciones se agrupan por su propósito general, determinan la operación que debe realizar el procesador y los operandos que debe ocupar (de ser necesarios). Se almacenan en la memoria de programa y se componen básicamente de tres campos[1], aunque en algunos casos pueden presentarse más [14]:

1. *Código de la operación*.
2. *Modo de direccionamiento*
3. *Dirección de los operandos*

El primer campo determina cual es la operación a realizar. El segundo y tercero determinan el origen de los operandos, ya sea la memoria o un registro interno. Existen operaciones en que el operando forma parte de la instrucción y sustituye a los dos últimos campos.

Instrucciones para transferencia de información

El valor de los datos no se altera, solo se transfieren de lugar. Su propósito es intercambiar datos entre la memoria, los registros internos del procesador y las terminales de entrada y salida. Cualquiera de estos lugares puede ser destino u origen. Las operaciones de carga, almacenamiento, intercambio y de pila pertenecen a esta categoría.

[1] Grupos de bits dentro del código binario de la instrucción.

Instrucciones aritméticas y lógicas

Estas instrucciones realizan operaciones con los datos. Las operaciones que ejecutan son aritméticas, lógicas o de manipulación y desplazamiento de bits.

Instrucciones para control de programa

Su finalidad es la de alterar el flujo del programa mediante saltos inmediatos o que se realicen dependiendo del estado de algún bit en cierto registro. Básicamente lo que hacen es cambiar el valor almacenado en el contador de programa por la dirección del destino del salto. También sirven para regresar a cierto punto del programa determinado por el salto anterior y cuya dirección se encuentra en la pila.

3.2.4. Modos de direccionamiento

Para reducir el número de bits que ocupa el campo de dirección dentro del código de la instrucción sólo se proporciona una parte de la dirección de memoria de un operando. El resto lo obtiene el decodificador de instrucciones a partir del modo de direccionamiento, que le indica la forma en que debe interpretarse el campo de dirección [15].

La dirección obtenida de esta manera por el decodificador de instrucción se le conoce como *dirección efectiva* y corresponde a la ubicación real del valor del operando que se utilizará en la operación a realizar.

El modo de direccionamiento también indica cuando alguno de los operandos se ubica en un registro interno del procesador. A continuación se describen los modos de direccionamiento más comunes. Cada microcontrolador puede presentar solo algunos de ellos e incluso incorporar otras variantes.

Modo implícito o inherente

En este modo la ubicación del valor de los operandos está definido implícitamente en el código de la operación. En este caso las operaciones están destinadas a trabajar siempre con los mismos registros.

Modo inmediato

El valor del operando está dentro del código de la instrucción en un campo que sustituye al de dirección y no debe obtenerse de la memoria o de algún registro. Al estar incluido dentro de la instrucción se ubicará dentro de la memoria de programa y será un valor constante.

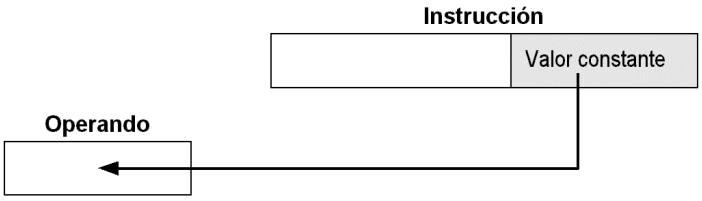


Figura 3.7: Direccionamiento inmediato.

Modo registro

En este modo se indica el registro del procesador que almacena el valor del operando. Se emplea con operaciones que pueden trabajar con diferentes registros.

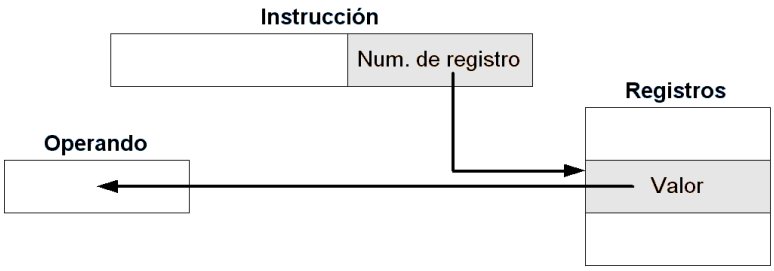


Figura 3.8: Direccionamiento en modo registro.

Modo registro indirecto

Es similar al modo registro aunque en este caso el registro especificado en la instrucción no almacena el valor del operando sino la dirección de memoria donde se ubica.

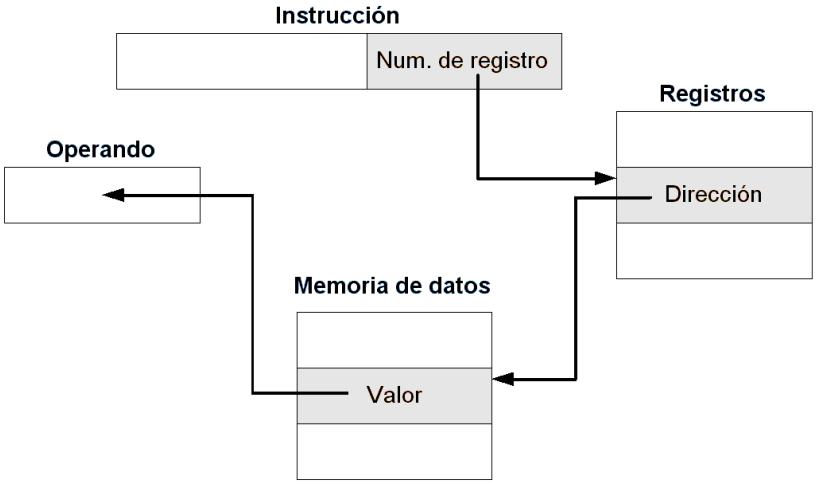


Figura 3.9: Direccionamiento en modo registro indirecto

Modo directo

En este caso la dirección de memoria donde se almacena el valor del operando se incluye completa dentro del campo de dirección. El tamaño de este campo debe ser el adecuado para incluir todos los bits de la dirección, con lo que la longitud del código de la instrucción aumentará.

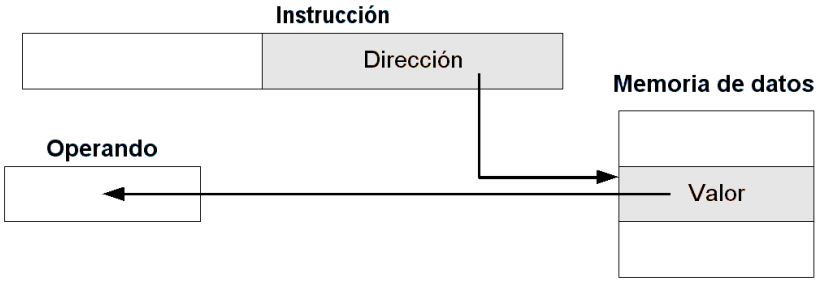


Figura 3.10: Direccionamiento directo

Modo indirecto

Al igual que en el modo directo el campo de dirección contiene una dirección de memoria completa, pero en la localidad correspondiente no se almacena el valor del operando sino la dirección de memoria donde se encuentra.

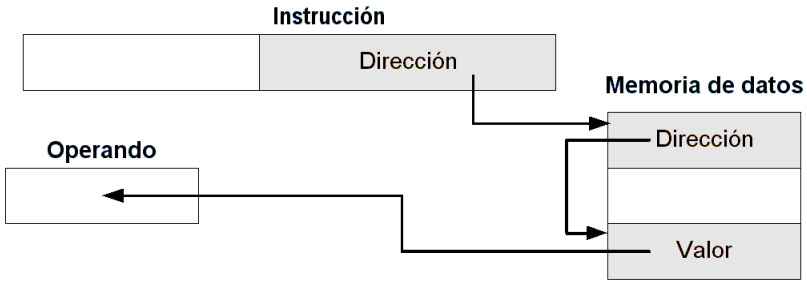


Figura 3.11: Direccionamiento indirecto

Modo relativo

La dirección efectiva donde se ubica el valor del operando se obtiene sumando el contenido del campo de dirección con un registro interno del procesador. Generalmente se emplea el contador de programa de manera que la ubicación del valor será relativa a la posición de la instrucción que lo requiere.

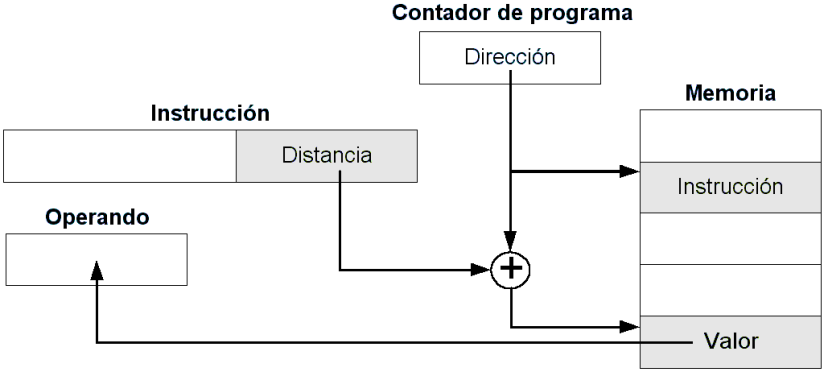


Figura 3.12: Direccionamiento relativo

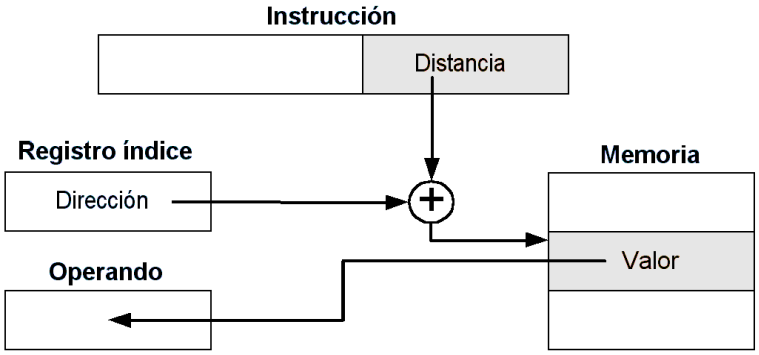


Figura 3.13: Direccionamiento indexado

Modo indexado

La dirección efectiva del valor del operando se obtiene sumando el contenido de un registro de índice interno con el campo de dirección. Estos registros están destinados únicamente a este propósito por lo que se puede modificar su valor sin alterar la ejecución del programa.

3.2.5. Interrupciones

Los microcontroladores tienen dos formas de detectar eventos externos. La primera observa constantemente las terminales de entrada y el estado de los dispositivos periféricos en espera de que algo suceda. A este método de sensar eventos externos se le denomina *pooling, poleo* o *encuesta*. Sin embargo, si se requiere atender inmediatamente un evento y no se puede esperar a que el programa llegue a la parte en que verifica si se ha presentado, este método no es el correcto.

En estos casos es útil el segundo método, en el que interrumpe la secuencia normal del programa es interrumpida en el momento en que se presenta el evento esperado. La ejecución se detiene en un punto indeterminado y se transfiere el control a una rutina destinada a atender el suceso.

Al presentarse el evento se genera una solicitud de interrupción y el procesador determinará si se atiende. Se pueden inhibir algunas interrupciones mediante registros que controlan el funcionamiento del hardware destinado a detectar los eventos que producen las interrupciones.

Cuando una interrupción debe ser atendida el procesador almacena en la pila la dirección almacenada en contador de programa y en ocasiones el valor de los registros que ocupa. En los casos en que el microcontrolador solo almacena el contador de programa, la rutina que atiende la interrupción deberá incluir instrucciones que almacenen y recuperen de memoria el valor de los demás registros.

El siguiente paso que lleva acabo el procesador es determinar la fuente de la interrupción para cargar el contador de programa con la dirección de memoria que corresponda a cada posible evento. Algunos microcontroladores cargan una sola dirección y en la rutina de atención se debe determinar el evento que generó la interrupción.

De cualquier forma la dirección con que se carga el contador de programa se determina mediante hardware y en ella debería iniciar la rutina de atención. Si se coloca en este lugar una instrucción de salto a otra dirección se puede cambiar su ubicación.

Al final de esta rutina se reestablece el estado original del procesador antes de la interrupción recuperando los registros de la pila o de la memoria. Se ejecutará entonces la siguiente instrucción a partir del punto en que se presentó la interrupción y el programa continuará de manera normal hasta que se ocurra otro evento que deba ser atendido.

Las interrupciones se clasifican a partir de la naturaleza del evento que se debe presentar para que sucedan [16].

Interrupciones por hardware externo

Son producidas en el exterior del procesador por cambios de estado en las terminales de entrada o salida, por el desbordamiento de temporizadores, contadores o acumuladores de pulso o por otros dispositivos al terminar de ejecutar cierta función.

Interrupciones por hardware interno

Son producidas en el interior del procesador al presentarse situaciones no manejables como un código de operación desconocido, una división entre cero, cuando se supera la capacidad de la pila o por un intento de escritura en una localidad no permitida o correspondiente a memoria de solo lectura.

Interrupciones de software

Las interrupciones internas y externas están implementadas mediante hardware, pero existen instrucciones que generan una solicitud de interrupción al ejecutarse. En este caso la interrupción es solicitada por software y se emplea para llamar a la rutina de atención desde cualquier lugar del programa y sin tener que esperar a que suceda cierto evento. Es diferente a una llamada a subrutina porque el estado del sistema se reestablece al regresar de ella. En las subrutinas se puede modificar el valor de los registros y al salir no retoman su valor original.

3.2.6. Filosofía del conjunto de instrucciones

La implementación de operaciones complejas permite reducir el software aunque requiere de una mayor cantidad de hardware. Un nivel poco complejo de hardware solo podrá ejecutar operaciones simples por lo que se requerirá de una mayor cantidad de software para realizar algunas funciones.

Esta relación de hardware con software determina dos enfoques principales con respecto a las características del conjunto de instrucciones con que cuenta la unidad central de procesamiento del microcontrolador.

Computadoras de conjunto de instrucciones complejo

La filosofía CISC (*Complex Instruction Set Computers*) busca poner a disposición del diseñador operaciones complejas y potentes que permitan realizar programas compactos y el desarrollo mediante lenguajes de programación de alto nivel [17].

En las arquitecturas CISC la mayoría de las instrucciones pueden acceder directamente a memoria además de que están disponibles la mayoría de los modos de direccionamiento. Los formatos de las instrucciones constan de una longitud en bits variable para cubrir todas estas opciones. El conjunto de instrucciones contiene muchas operaciones tanto elementales que se ejecutan en pocos ciclos, como complejas que requieren de bastantes ciclos para llevarse a cabo.

Computadoras de conjunto de instrucciones reducido

Las arquitecturas RISC (*Reduced Instruction Set Computers*) implementan operaciones elementales que se ejecutan en un solo ciclo. Solo pueden acceder a la memoria con instrucciones de carga o almacenamiento por lo que las operaciones que modifican valores únicamente trabajan con registros. Disponen de sólo algunos modos de direccionamiento y al contar con un repertorio reducido de instrucciones la longitud de los formatos de instrucción es constante.

La filosofía RISC busca un alto rendimiento y la ejecución rápida de las instrucciones [18]. Para reducir el tiempo de ejecución se evitan los accesos a memoria mediante la implementación de un número elevado de registros.

Computadoras de conjunto de instrucciones específico

La filosofía SISC (*Specific Instruction Set Computers*) deriva de las arquitecturas RISC que incluyen únicamente instrucciones que se adaptan a ciertas condiciones de funcionamiento. Las arquitecturas SISC están presentes en los microcontroladores que son diseñados para ejecutar aplicaciones muy específicas.

3.3. Memoria

Los microcontroladores emplean memorias semiconductoras para almacenar datos e instrucciones. Se puede acceder a sus distintas localidades de forma aleatoria, sin ningún orden específico, por lo que se dice que su organización es de acceso aleatorio.

Las memorias que permiten escritura y lectura almacenan la información en circuitos activos, por lo que al no haber alimentación los datos guardados se pierden. Las memorias no volátiles de solo lectura mantienen la información aún cuando no está presente la alimentación pues almacenan la información mediante la presencia o ausencia de diodos o transistores.

Aunque tanto las memorias volátiles como las no volátiles son de acceso aleatorio, el término RAM (*Random Access Memory*) es generalmente empleado para referirse a las memorias volátiles de lectura y escritura [19].

3.3.1. Memorias no volátiles

Memoria de solo lectura (ROM)

La información es escrita como parte del proceso de fabricación y no puede modificarse posteriormente. Se conocen también como *ROM de máscara* pues es en la máscara que se empela para su fabricación donde se define la información que va a almacenar.

Memoria de solo lectura programable (PROM)

Cuando se fabrican almacenan únicamente unos lógicos que pueden posteriormente cambiarse a ceros aplicando pulsos eléctricos de forma selectiva. Este proceso no puede revertirse por lo que la información quedará permanentemente almacenada.

Memoria de solo lectura programable y borrable (EPROM)

Son memorias similares a las PROM en las que se puede revertir varias veces el proceso de programación aplicando luz ultravioleta para reestablecer los unos lógicos. Todo el contenido se borra al mismo tiempo por lo que no se pueden modificar datos selectivamente.

Memoria de solo lectura programable y borrable eléctricamente (EEPROM)

La información se programa y se borra por medio de impulsos eléctricos de manera selectiva, por lo que pueden modificarse sólo algunas localidades mientras que las demás permanecen sin cambios. El número de veces que se pueden borrar y reprogramar es limitado.

Memoria FLASH

Son memorias no volátiles que se escriben y borran eléctricamente. Son más densas que las EEPROM por lo que tienen una mayor capacidad de almacenamiento. También son más rápidas, consumen menos energía y soportan más ciclos de borrado y escritura. Cuando se borra su contenido se hace sobre bloques completos, por lo que no se pueden modificar los datos selectivamente.

3.3.2. Memoria de programa

Está destinada a almacenar las instrucciones que conforman los programas. La capacidad de la memoria empleada determina el tamaño máximo que puede tener el programa que almacenará. En algunos casos esta memoria también se ocupa para guardar datos de utilidad.

3.3.3. Memoria de datos

En ella se almacenan las variables que ocupa el programa. Como estos datos pueden cambiar constantemente se emplean memorias volátiles RAM. Algunos microcontroladores tienen una sección de tipo EEPROM que permite conservar la información aunque no exista alimentación.

3.4. Otros componentes

3.4.1. Reloj del sistema

Los microcontroladores sincronizan su funcionamiento mediante una onda cuadrada que corresponde a los impulsos de reloj del sistema. Esta onda es generada por el circuito de reloj a partir de la señal de un oscilador externo al microcontrolador que emplea cristales de cuarzo, resonadores cerámicos o circuitos RC. Un aumento en la frecuencia del reloj reducirá el tiempo empleado para ejecutar cada operación pero el consumo de energía aumentará [20].

La frecuencia de la señal de reloj es la que determina la velocidad a la que el microcontrolador ejecuta las operaciones, aunque la frecuencia de operación del sistema es siempre menor a ella. Cada uno de los pasos que componen un ciclo de instrucción se ejecuta en un ciclo de la señal de reloj, por lo que el tiempo en que tarda en ejecutarse toda la instrucción es mayor.

3.4.2. Líneas de entrada y salida

Las terminales del microcontrolador están destinadas en su mayoría a las líneas de entrada y salida que le permiten interactuar con el mundo exterior. Pueden funcionar individualmente o agrupadas en *puertos* donde cada línea corresponde a un bit de una palabra.

En algunos casos se pueden configurar para que funcionen como líneas de entrada, de salida o en ambas direcciones. Algunos recursos internos pueden compartir las mismas líneas por lo que se optimiza el espacio dentro del encapsulado.

A los puertos se les asigna un registro o una localidad de memoria para poder leer o cambiar mediante la escritura el estado lógico de sus terminales asociadas. Las líneas individuales generalmente son entradas o salidas de los recursos internos del microcontrolador.

3.4.3. Recursos auxiliares

Las prestaciones de un microcontrolador aumentan con la inclusión de recursos auxiliares aunque con esto se incrementa también el costo. La elección del microcontrolador adecuado para una aplicación específica muchas veces depende de la disponibilidad y flexibilidad de ciertos recursos especiales que faciliten el trabajo del sistema.

Temporizadores

Sirven para llevar una cuenta de impulsos externos o internos. Cuando éstos son generados por el reloj del sistema funcionan como temporizadores y se emplean para establecer periodos de tiempo de cierta duración. Funcionan como contadores de eventos externos cuando los impulsos provienen del exterior por medio de alguna línea de entrada. Si estos impulsos son periódicos, funcionarán como temporizadores trabajando a una frecuencia distinta a la del reloj del sistema.

Convertidores analógico-digital y digital-analógico

Los convertidores analógico-digitales asignan un valor binario a la magnitud del voltaje presente en su entrada. Con ellos se convierte una señal de naturaleza analógica en una de naturaleza digital con el propósito de procesarla. Pueden tener varias líneas de entrada para poder convertir diferentes señales.

Los convertidores digital-analógicos realizan el proceso inverso. Aplican en su línea de salida un voltaje correspondiente a un valor binario.

Moduladores de ancho de pulso

Son circuitos que generan señales con pulsos de duración variable en una terminal de salida. El ancho de estos pulsos representa la información modulada.

Puertos de comunicación

Permiten al microcontrolador comunicarse con otros sistemas mediante algunas terminales de entrada y salida empleando protocolos de comunicación serial (síncrona o asíncrona) y paralela.

Protección contra fallas en la alimentación

Son circuitos que reinician el microcontrolador cuando el voltaje de alimentación tiene un valor menor que el mínimo requerido para su funcionamiento y lo mantienen así mientras no supere dicho nivel.

Protección contra fallas del programa

A este recurso se le llama *Watchdog* (perro guardián) y se implementa mediante un temporizador que reinicia el microcontrolador cada determinado tiempo por lo que debe reiniciarse constantemente su cuenta dentro del programa. Cuando el microcontrolador se bloquee por algún error en el programa no se refrescará el temporizador y se reiniciará el sistema transcurrido cierto tiempo.

Estado de bajo consumo de energía

Permite detener el sistema en espera de un evento externo con el propósito de ahorrar energía. El microcontrolador solo trabajará cuando se requiera y el resto del tiempo prácticamente no consumirá energía.

3.5. Programación

3.5.1. Lenguajes

El lenguaje que entienden los microcontroladores es el código binario de los programas que se almacena en su memoria y es por esto que se le llama *lenguaje máquina* [21]. Para facilitar la realización de los programas, los fabricantes asignan un nombre a cada operación (*mnemónico*) que describe muy brevemente su función y sustituye a su código binario.

Los mnemónicos constituyen el lenguaje ensamblador del microcontrolador y los programas escritos en él deben ser traducidos a código binario antes de almacenarse en la memoria. A este proceso se le conoce como *ensamblado*. Este proceso ha sido automatizado mediante programas de computadora conocidos como *ensambladores* [22] que lo realizan de una forma mas rápida y eficiente.

Los lenguajes de programación de medio y alto nivel, como el *C* o el *BASIC* respectivamente[1], también se emplean para programar microcontroladores. Las instrucciones empleadas por éstos son más similares al lenguaje común. En este caso el software que realiza la traducción se conoce como *compilador* y toma el código fuente del programa para transformarlo a código binario.

Los programas hechos en lenguaje ensamblador ocupan menos espacio en memoria y son más eficientes, pero cuando son demasiado extensos se vuelven confusos y es difícil depurarlos y mantenerlos. La programación en lenguajes de nivel superior es más sencilla pero su código binario ocupa mucho mas espacio en la memoria del microcontrolador debido a que se tienen que implementar estructuras y variables.

3.5.2. Herramientas de desarrollo

La primera herramienta que se requiere para el desarrollo de los programas es un editor de texto. Se emplea para escribir los programas y el archivo de texto generado es procesado por el ensamblador (o por el compilador) para crear el archivo que contiene el código binario.

Para almacenar el programa en la memoria del microcontrolador se requiere de un circuito programador donde se coloca éste y se transfiere el código binario mediante un programa de computadora. Algunos microcontroladores pueden programarse ya montados en el circuito sobre el que trabajarán.

Una herramienta de gran utilidad durante el proceso de prueba y depuración del programa es el simulador. Estos programas de computadora, como su nombre lo indica, simulan el funcionamiento del microcontrolador al ejecutar las instrucciones del programa.

[1] *Cfr.* Herbert Schildt, *C Manual de referencia*, “Introducción al lenguaje C”, España: Ed. McGraw-Hill, 1989, pp. 3-5.

Un entorno de desarrollo es un paquete de software más complejo que incluye todos estos elementos de manera que en él se puede llevar a cabo todo el proceso.

3.6. Interacción con el mundo exterior

La mayoría de las señales en el mundo real son señales analógicas que tienen un comportamiento continuo en el tiempo y representan información no discreta pues sus características pueden tomar un número infinito de valores dentro de cierto rango. Otro aspecto que debe considerarse es que en general estas señales están asociadas a fenómenos físicos de naturaleza *no eléctrica*. Un microcontrolador no pueda procesar estas señales directamente por lo que requiere de un *dispositivo de entrada* y un *acondicionador de señal* que le proporcionen la información en cantidades discretas.

El dispositivo de entrada debe entregar al acondicionador una señal eléctrica proporcional a la señal de entrada independientemente de la naturaleza original de ésta. El acondicionador realizará el proceso necesario para que la señal eléctrica sea compatible con el microcontrolador. Estos elementos conforman un *sistema de medición* donde el microcontrolador es el *dispositivo de salida* [23].

3.6.1. Transductores

Cuando las señales de entrada a un sistema no son eléctricas, el dispositivo de entrada deberá someterla a un proceso de transformación que resulte en una representación eléctrica de la variable física que se está midiendo.

Un transductor se define como "... un dispositivo que convierte una señal de entrada en una señal de salida de otra forma" [24]. Los transductores eléctricos entregan a la salida una señal eléctrica cuya magnitud está en función de la de la variable física que se está midiendo [25].

Los transductores *pasivos* son elementos resistivos, capacitivos o inductivos que sufren una variación en su valor como consecuencia de la energía que reciben de la señal de entrada. Mediante circuitos eléctricos se puede obtener un voltaje o corriente que represente la magnitud de la señal de entrada. Los transductores *activos* entregan directamente un voltaje o una corriente.

La selección del transductor adecuado para una cierta aplicación depende principalmente de la variable física que se desea medir. Para una misma cantidad física pueden existir diversos transductores con diferente principio de funcionamiento. Para tomar una decisión se deben considerar diferentes aspectos como el rango que toma la variable, la exactitud requerida o el entorno en que deberán trabajar, por mencionar algunos [26].

A continuación se mencionan algunas variables físicas de interés para el presente trabajo y los transductores más comunes para su medición [27].

Transductores de temperatura

- **Termistores**
  Son elementos resistivos sensibles a la temperatura (transductores pasivos). Su coeficiente de temperatura es negativo, por lo que al aumentar ésta disminuye el valor

de su resistencia y viceversa. La relación entre resistencia y temperatura no es lineal y se deberán emplear circuitos adicionales que permitan una mejor aproximación lineal dentro del rango de operación.

- **Detectores de resistencia- temperatura**
  Los llamados RTD (por sus siglas en inglés) son alambres que cambian el valor de su resistencia con la variación de la temperatura. El cambio es de sólo algunos ohms y del material seleccionado dependerá que el valor de la resistencia aumente o disminuya cuando la temperatura incremente su valor.

  Algunos materiales comunes son el cobre, el platino, la plata y el níquel. El valor con que se identifica un RTD es el valor de resistencia que presenta a cero grados Celsius.

- **Termopares**
  Estos sensores se fabrican mediante la unión de dos metales diferentes. Al elevarse la temperatura se produce un voltaje muy pequeño cuya magnitud dependerá de los materiales empleados. A este fenómeno se le conoce como *efecto Seebeck*[1].

- **Diodo en polarización directa**
  La ecuación que relaciona la corriente que fluye por un diodo (al polarizarse directamente) con la temperatura se obtiene mediante la aplicación de la física del estado sólido [28]:

$$I = I_S(e^{\frac{kV}{T_K}} - 1)$$

  Donde:
  I_S: Corriente inversa de saturación
  k= 5800 para silicio y k=11600 para germanio
  T_K: Temperatura en grados kelvin

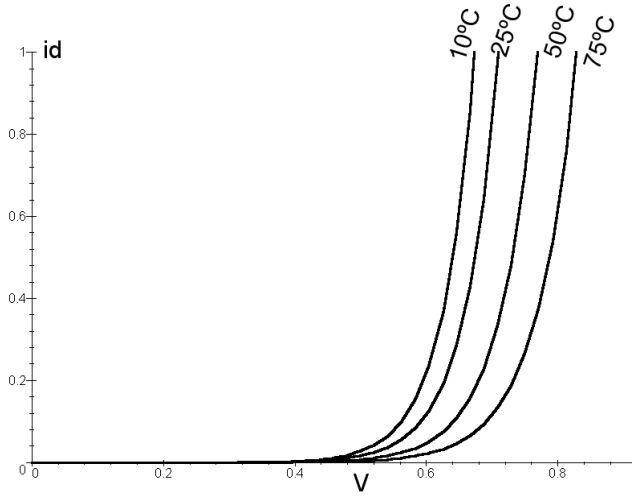


Figura 3.14: Corriente de un diodo de silicio polarizado en directa. Is=1uA, k=5800.

[1] En honor a Thomas Seebeck que lo descubrió en 1821

En la figura 3.14 se puede observar el efecto de la variación de la temperatura sobre el punto de operación del diodo. La variación del voltaje en que comienza a funcionar en directa depende de la temperatura. La principal ventaja de estos transductores es que pueden incluirse dentro del circuito integrado que adecua la señal.

Transductores ópticos

- **Fotorresistencias**
  También son conocidas como celdas fotoconductoras. La resistencia entre sus dos terminales varía linealmente con la intensidad de la luz que incide en su superficie.

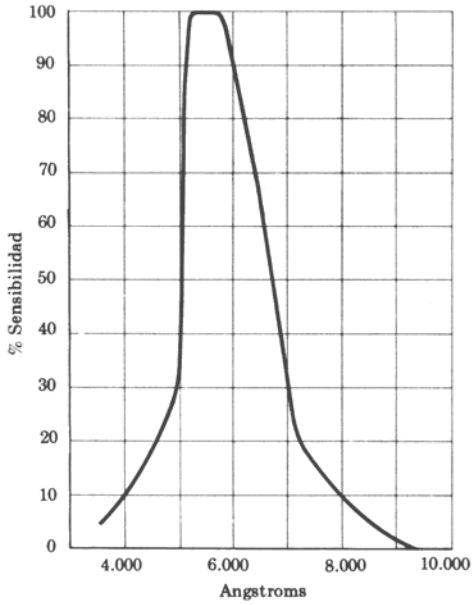


Figura 3.15: Respuesta espectral de una fotorresistencia de cloruro de cadmio [29].

Los materiales más empleados son el sulfato de cadmio y el seleniuro de cadmio. Su respuesta máxima se presenta en una longitud de onda de 5000 y 6000 Å respectivamente[1]. El rango de variación es significativo por lo que permite aplicaciones en que los cambios de iluminación son pequeños [30].

- **Fotodiodos**
  Son diodos que trabajan polarizados inversamente. En ausencia de luz incidente sobre su superficie, la corriente inversa de saturación es muy pequeña, como sucede en general con todos los diodos. Cuando la intensidad de la luz aumenta en la unión de los materiales *n* y *p*, se transmite una mayor energía en forma de fotones aumentando el número de portadores minoritarios. El aumento de electrones libres produce un incremento de la corriente inversa. Este fenómeno aumenta considerablemente para la región de luz infrarroja con respecto al espectro de luz visible como se puede observar en la gráfica de la figura 3.16 [31].

[1] Ángstrom, unidad empleada para expresar la longitud de onda y es equivalente a 10^{-10} metros.

- **Fototransistores**
  El principio de funcionamiento es el mismo que el del fotodiodo. En este caso la unión fotosensible es la de base-colector. La corriente inducida por los fotones será la corriente de base. Los fototransistores tienen una mayor sensibilidad debido a que la corriente de colector es varias veces mayor que la de base (dependiendo del valor de h_{fe} del transistor) [32].

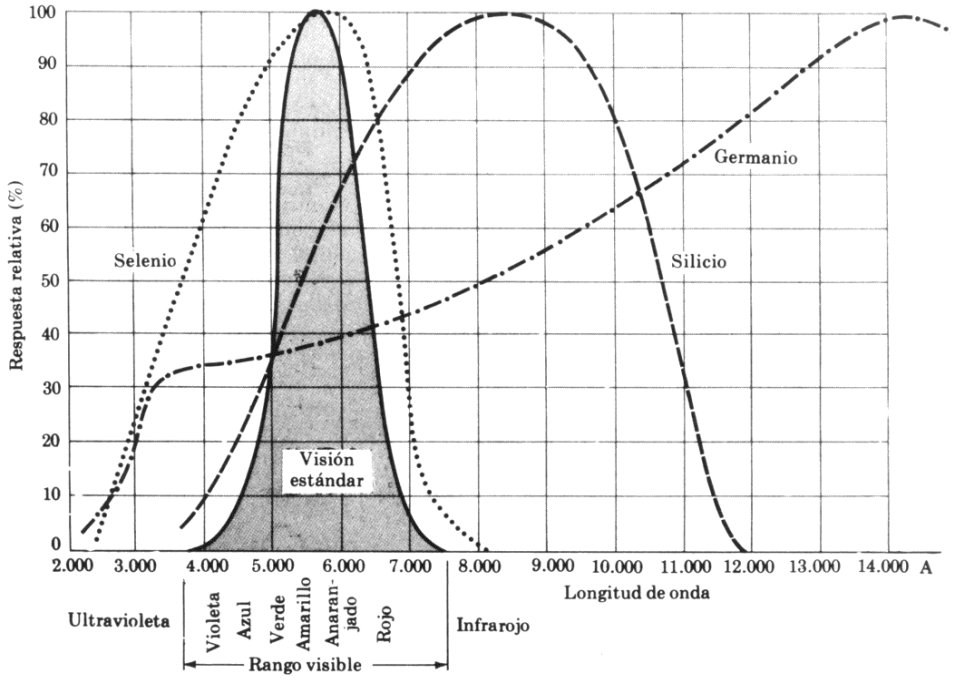


Figura 3.16: Respuesta espectral para materiales semiconductores.

Transductor magnético de efecto Hall

El efecto Hall[1] consiste en la aparición de un pequeño voltaje en una delgada placa de oro, en la cual fluye una corriente eléctrica, al colocar un campo magnético perpendicularmente. Este voltaje es proporcional a la densidad de flujo magnético y a la intensidad de la corriente que circula en la placa. El uso de semiconductores en la construcción del *elemento Hall* permite incluir el transductor dentro del circuito integrado que amplifica el voltaje.

3.6.2. Conversión analógico-digital

El proceso del acondicionador de señal deberá siempre concluir con la conversión de la información analógica en valores digitales que el microcontrolador pueda comprender. El método empleado dependerá de la forma en que se interprete la magnitud de la señal de entrada.

Estado lógico de la amplitud

En algunas aplicaciones se requiere únicamente determinar cierto estado de una señal de entrada para saber si está ausente o presente, si su amplitud es mayor o menor a determinado

[1] Descubierto en 1879 por el Dr. Edwin Hall.

voltaje, o para determinar la frecuencia de la señal. En otros casos se debe restaurar el nivel lógico del voltaje de la señal para que coincida con los rangos que maneja el microcontrolador.

Esta conversión es la más sencilla pues los valores que puede tomar el estado de la señal son únicamente dos. Existen diversos circuitos que pueden realizar esta función, y entre ellos están los comparadores analógicos.

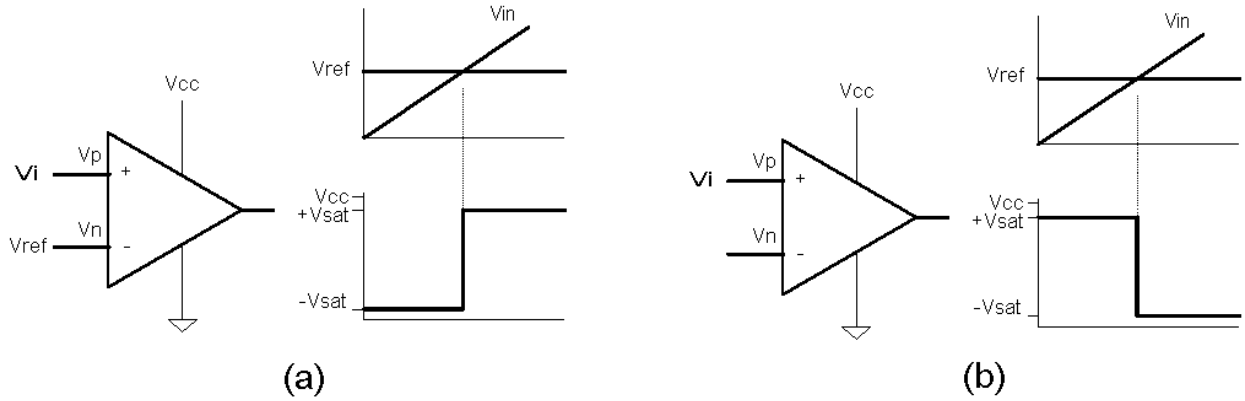


Figura 3.17: Funcionamiento de un comparador de voltaje: (a) No inversor y (b) Inversor.

Un comparador de voltaje es un amplificador operacional que funciona en lazo abierto o con retroalimentación positiva por lo que el voltaje de salida únicamente toma los voltajes de saturación positivo y negativo. Compara el voltaje presente en sus entradas y responde de acuerdo a la siguiente regla:

- Si $V_P > V_N \rightarrow V_O = +V_{SAT}$
- Si $V_N > V_P \rightarrow V_O = -V_{SAT}$

Valor discreto de la amplitud

Cuando se requiere conocer el valor de la amplitud del voltaje de la señal de entrada se debe realizar una conversión analógico-digital para obtener un valor binario equivalente al valor del voltaje de entrada. El valor binario depende del voltaje de referencia utilizado en la conversión.

En las conversiones se compara el voltaje de entrada con varios niveles de voltaje de referencia para determinar dentro de qué intervalo se encuentra. Entre más pequeños sean estos intervalos, mejor resolución tendrá el convertidor.

- **Convertidor A/D paralelo**
  Los convertidores en paralelo emplean varios comparadores de voltaje con un valor de referencia distinto. El voltaje a convertir se aplica a la entrada de todos los comparadores de manera que a la salida se indicará con un estado lógico si el voltaje a convertir es mayor que cada uno de los voltajes de referencia.

  La salida activa con el voltaje de referencia mas alto determinará el valor del voltaje de entrada y mediante un decodificador de prioridad se obtiene el número binario equivalente. Es el método más rápido pero también el más difícil de implementar pues junto con la resolución (n bits) aumenta el número de comparadores empleados (2^n comparadores).

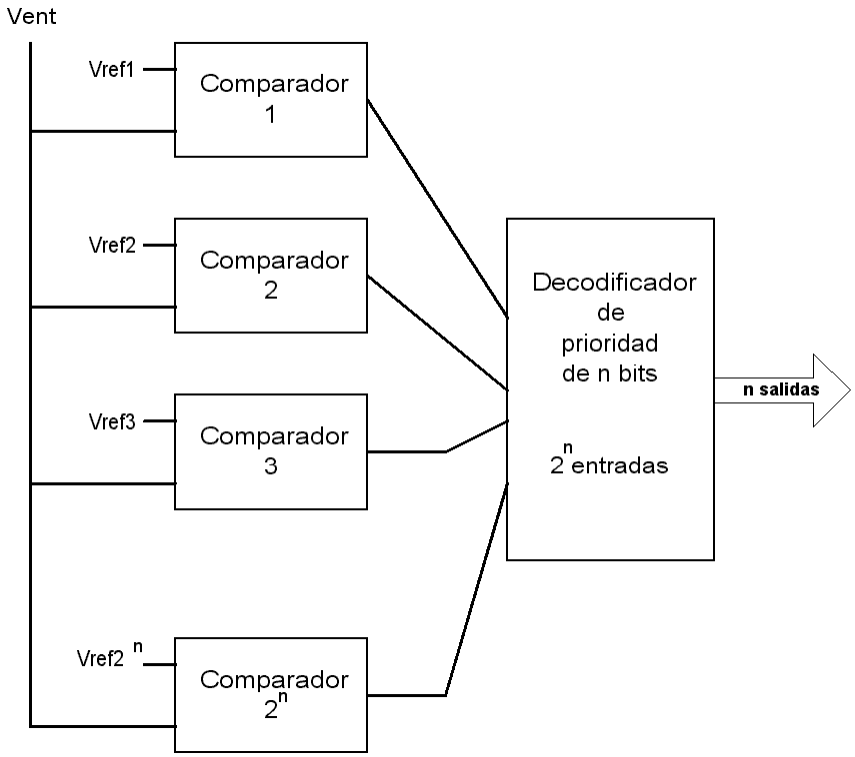


Figura 3.18: Convertidor A/D paralelo.

- **Convertidor A/D de aproximaciones sucesivas**
  Estos convertidores comparan el voltaje de entrada con un voltaje interno que se obtiene mediante un convertidor digital-analógico con las entradas correspondientes al número de bits del convertidor. Un dispositivo programable incrementa bit por bit la palabra que se aplica al convertidor D/A para que aumente el voltaje hasta que sea igual al de la entrada del convertidor. Cuando esto sucede, el valor binario de la conversión será la última palabra introducida al convertidor D/A.

- **Convertidor A/D de contador**
  Su funcionamiento es similar al de aproximaciones sucesivas, salvo que en este caso la palabra a la entrada del convertidor D/A proviene de las salidas de un contador binario que incrementa su valor por medio de una señal de reloj hasta alcanzar el valor del voltaje a convertir.

3.6.3. Interruptores

En muchas aplicaciones los interruptores y pulsadores (*push-button*) se emplean como medios de entrada para señales binarias. Los circuitos básicos para conectar los interruptores se presentan en la figura 3.19. Las resistencias son necesarias para asegurar un estado lógico válido cuando el interruptor esté abierto.

El principal problema que presentan estos dispositivos es de tipo mecánico. Cuando se cierra o se abre el interruptor se generan pequeños rebotes en los contactos que hacen que la salida cambie de estado lógico varias veces antes de tomar su valor final. Este efecto se muestra en la figura 3.20, mientras que en la figura 3.21 se presenta un circuito que puede emplearse para

evitar los efectos transitorios en la salida. Cuando el pulsador está abierto el capacitor se carga con V_{CC} a través de R_1. Al activar el interruptor el capacitor se descarga a través de R_2 hasta alcanzar un valor determinado por el divisor de R_2 con R_1. Los valores de los elementos del circuito determinan el tiempo de carga y descarga del capacitor y el voltaje en R_2 al activarse el pulsador [33].

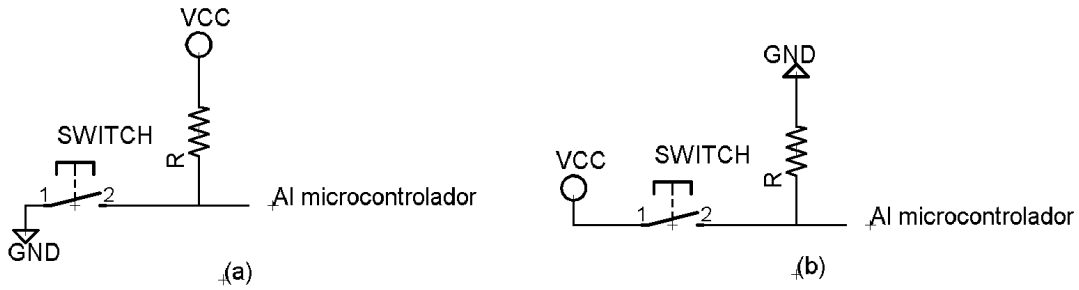


Figura 3.19: Interruptor (a) Activado en 0, (b) Activado en 1.

Figura 3.20: Efecto transitorio o *de rebote* en los interruptores.

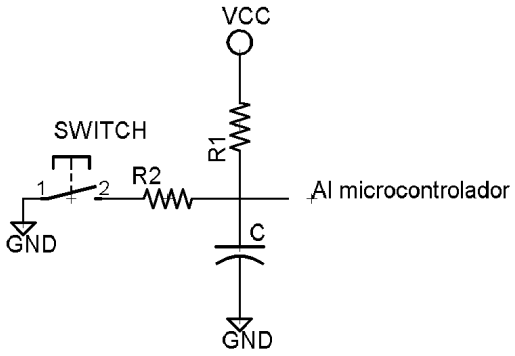


Figura 3.21: Interruptor sin efectos transitorios.

3.7. Selección del microcontrolador

Antes de comenzar a desarrollar los componentes del dispositivo se debía definir cuál modelo de microcontrolador utilizar, debido a que éste determinará las características de las señales entregadas por los dispositivos de entrada y de las señales de control para los dispositivos de salida. El análisis realizado en le capítulo 2 permitió determinar las características deseables del microcontrolador a utilizar:

- Alta capacidad de almacenamiento en memoria de programa para poder implementar las rutinas necesarias para el control del dispositivo.
- Memoria de datos no volátil para almacenar valores de configuración.
- Convertidor analógico – digital para trabajar con señales analógicas.
- Capacidad de contabilizar eventos externos, como las vueltas de la ruda o las pulsaciones del corazón.
- Amplio rango en sus temporizadores para la base de tiempo del cronómetro.
- Flexibilidad de la frecuencia de operación y del voltaje de alimentación.
- Suficientes terminales de entrada y salida para reducir los elementos externos.

Siendo el criterio predominante para seleccionar una opción la relación - costo beneficio [34], se decidió trabajar con microcontroladores de 8 bits. Los de cuatro bits podrían no cumplir con todos los requerimientos, y aunque uno de 16 o 32 facilitaría bastante el desarrollo gracias a su cantidad de memoria, diversos dispositivos de entrada y salida disponibles, así como su alta capacidad de procesamiento, se tendría un incremento considerable en el costo.

En el mercado hay muchos modelos disponibles de microcontroladores de 8 bits, por lo que fue necesario reducir las opciones posibles, pues para evaluar y comparar todas ellas se requeriría invertir demasiado tiempo. Debido a todas las opciones disponibles, solo se consideraron cuatro modelos conocidos de igual número de fabricantes, todos dentro del mismo rango de precio, que pudieran cumplir de alguna forma con los requerimientos. Sus características se resumen en la tabla 3.1.

Modelo	Fabricante / Familia	Memoria interna / externa		Pines	Alimentación
MC68HC11F1	Motorota HC11	1K RAM 512 EEPROM	64K	68	5V
87C51FA	Intel MCS-51	256 RAM 8K EPROM	64K datos 64K programa	44	2.7V - 5.5V
PIC16F877	Microchip PIC16	368 RAM 256 EEPROM 8K FLASH	-	40	2.0V - 5.5V
COP8CBR9	National COP08	1K RAM 32K FLASH	-	44	2.7V - 5.5V

Modelo	Frecuencia máxima de operación	Pines E/S - Puertos	Temporizadores	Convertidor A/D
MC68HC11F1	4 MHz	56 - 7	1 de 16 bits	8 bits 8 canales
87C51FA	16 MHz	32 - 4	3 de 16 bits	-
PIC16F877	20 MHz	33 - 5	2 de 8 bits 1 de 16 bits	10 bits 8 canales
COP8CBR9	20 MHz	39 - 5	4 de 16 bits	10 bits 16 canales

Modelo	Reloj adicional externo para Timer	Instrucciones	Modos de direccionamiento	Vectores de Interrupción
MC68HC11F1	No admite	144	6	26
87C51FA	No admite	38	6	7
PIC16F877	DC - 200 kHz	35	3	14
COP8CBR9	32 kHz	44	6	13

Tabla 3.1: Comparación de las características de cuatro modelos de microcontroladores. Recopilado de *MC68HC11F1 Technical Summary*, Motorola 1997; *MCS 51 MICROCONTROLLER FAMILY USER'S MANUAL*, Intel 1994; *PIC16F87X Data Sheet. 28/40-Pin 8-Bit CMOS FLASH Microcontrollers*, Microchip Technology 2001; y *COP8CBR9 8-Bit CMOS Flash Microcontroller*, National Semiconductor 2003.

De estas cuatro opciones, el modelo de Intel no cuenta con convertidor A/D, por lo que se tiene que utilizar uno externo. Además su memoria de programa interna es del tipo EPROM, lo que retarda el proceso de depuración del programa, además de que no permite que el programa almacene datos permanentes, requiriendo entonces una memoria EEPROM externa.

El HC11F1 de Motorota cubre las carencias del anterior, pero su cantidad de memoria EEPROM es relativamente pequeña, con lo que se incrementa el costo al tener que emplear memoria externa. Además cuenta con un solo temporizador y no tiene mucha flexibilidad en cuanto a voltaje de alimentación y frecuencia de operación.

Los dos modelos restantes presentan ventajas muy similares y beneficios adicionales como un bajo costo, suficiente memoria interna, flexibilidad en frecuencia y alimentación, así como herramientas de desarrollo y documentación disponibles sin costo.
Fue seleccionado el PIC16F877 sobre el COP8 debido a algunas ligeras ventajas indicadas en la tabla 3.2, y aunque este último tiene mas memoria de programa, la cantidad incluida en el PIC es suficientemente grande pues todas sus instrucciones ocupan una sola palabra de memoria, mientras que en el COP solo lo hace el 77% de ellas.

Siendo tan parecidas ambas opciones, además de las ventajas antes mencionadas, fue importante también considerar la mayor experiencia en el desarrollo con modelos más pequeños de la familia PIC, lo que permite reducir el tiempo de aprendizaje del conjunto de instrucciones, características de funcionamiento y recursos del microcontrolador.

PIC16F877	COP8CBR9
Cada instrucción del PIC se ejecuta en un ciclo máquina excepto los saltos que requieren dos.	Se ejecutan en uno o varios ciclos de reloj.
Tiene EEPROM real.	Es virtual, pues se copian bloques de la memoria FLASH a memoria RAM.
Puede operar a 20 MHz.	Requiere un duplicador de frecuencia para alcanzar 20 MHz.
El reloj adicional externo admite frecuencias 0 a 200 kHz.	El reloj adicional externo trabaja únicamente a 32 kHz.
Un solo vector de interrupción.	Un vector distinto para cada fuente de interrupción.
Menor consumo: 3.9 - 5.2 mA @ 10 MHz, 5.5V 0.9 - 1.1 mA @ 3.33 MHz, 4.5 V 53 - 80 µA @ 32 kHz, 5.5V	Mayor consumo: 13.2 mA @ 10 MHz, 5.5V 6 mA @ 3.33 MHz, 4.5V 103 µA @ 32 kHz, 5.5V

Tabla 3.2: Ventajas del PIC16 sobre el COP8. Recopilado de *PIC16F87X Data Sheet. 28/40-Pin 8-Bit CMOS FLASH Microcontrollers*, Microchip Technology 2001; y *COP8CBR9 8-Bit CMOS Flash Microcontroller*, National Semiconductor 2003.

3.8. Referencias

1 Douglas V. Hall, *Microprocessors and interfacing. Programming and hardware*, “Computers, microcomputers and microprocessors”, 3a. ed., USA: Ed. McGraw Hill, 1992, p. 27.

2 William C. Wray, Joseph D. Greenfield, *Using microprocessors and microcomputers. The Motorola Family*, “Introduction to microcomputers”, USA: Ed. Prentice-Hall, 1994, pp. 1-7.

3 M. Morris Mano, Charles R. Kime, *Fundamentos de diseño lógico y computadoras*, “Computadoras digitales e información”, México: Ed. Prentice-Hall Hispanoamericana, 1998, pp. 3-5.

4 *Ibíd.*, “Memoria y dispositivos lógicos programables”, p. 257.

5 Gene H. Miller, *Microcompter engineering*, “Computer fundamentals”, USA: Ed. Prentice-Hall, 1999, p. 39.

6 Neil Willis, *Fundamentos de arquitectura de ordenadores y comunicaciones de datos*, “Memoria”, España: Anaya, 1990, pp. 91-103.

7 Gene H. Miller, *Op. Cit.*, “Instruction subset and machine language”, p. 50.

8 M. Morris Mano, *Op. Cit*.,”Transferencias de registros y rutas de datos”, pp. 331-332.

9 *Ídem*.

10 Neil Willis, *Op. Cit.*, “Memoria”, pp. 88-89.

11 *Ibíd.*, p. 143.

12 Gene H. Miller, *Op. Cit.*, “Computer fundamentals”, pp. 33-37.

13 Gene H. Miller, *Op. Cit.*, “Instruction subset and machine language”, pp. 54-59.

14 M. Morris Mano, *Op. Cit*., ”Arquitectura del conjunto de instrucciones”, p. 428.

15 *Ibíd.*, p. 435.

16 *Ibíd.*, pp. 461-462.

17 *Ibíd*., p. 442.

18 *Ídem*.

19 David A. Hodges, Horace G. Jackson, *Análisis y diseño de circuitos integrados digitales*, “Memorias semiconductoras”, Barcelona: Ed. Gustavo Gili, 1988, p. 378.

20 José Ma. Angulo, Martín Cuenca, *Microcontroladores PIC. La solución en un chip*, “Principios, características y aplicaciones generales”, Barcelona: Ed. Paraninfo, 1998, p. 19.

21 Gene H. Millar, *Op. Cit.*, “Instruction subset and machine language”, p. 62.

22 *Ibíd.*, pp. 113-119.

23 Moisés E. Rueda, Francisco J. Padilla y Tomás Medina, *Diseño e implementación de un sistema de control electrónico para un equipo médico de diálisis peritoneal*, “Medios de entrada-salida”, Universidad Nacional Autónoma de México, Tesis de Licenciatura, México: Los autores, 1990, p. 23.

24 Katsuhiko Ogata, *Ingeniería de control moderna*, trad., Bartolomé Fabian-Frankel, México: Ed. Prentice-Hall Hispanoamericana, 1980, p. 676.

25 Moisés E. Rueda, Francisco J. Padilla y Tomás Medina, *Op. Cit.*, p.24.

26 *Ibíd*., pp. 25-26.

27 Stuart R. Ball, *Analog interfacing to embedded microprocessors. Real world design*, "Sensors", Boston: Ed. Newnes, 2001, pp. 47-71, 82-91.

28 Robert Boylestad y Louis Nashelsky, *Electrónica. Teoría de circuitos*, "Diodos semiconductores", trad., Carlos A. Franco, México: Ed. Prentice-Hall Hispanoamericana,1983, pp. 19-22, 26.

29 *Ibíd*., "Zeners y otros dispositivos de dos terminales", pp.102-103.

30 *Ídem.*

31 *Ibíd*., pp. 98-101.

32 *Ibíd*., "Otros dispositivos y los pnpn", pp. 463-464.

33 José Ma. Angulo, Ignacio Angulo, *Microcontroladores PIC. Diseño práctico de aplicaciones*, "Microcontroladores programables: la solución está en un chip", 2a. ed., Madrid: Ed. McGraw Hill, 1999, pp. 23-25.

34 E. V. Krick, *Introducción a la Ingeniería y al Diseño en la Ingeniería*, "El proceso de diseño: La fase de decisión", México: Ed. Limusa, 1980, p. 162.

Capítulo IV

Microcontrolador PIC16F877

4.1. Los microcontroladores PIC

Microchip, el fabricante de los microcontroladores PIC, ha logrado penetrar en el mercado gracias a la buena relación costo – beneficio de sus microcontroladores, su reducido consumo de energía, facilidad de programación y por la diversidad de herramientas de desarrollo disponibles. Es importante mencionar que el software mas conocido para la programación de estos dispositivos es un desarrollo libre creado por usuarios de los microcontroladores PIC.

Además de contar con una gran diversidad de versiones, lo que facilita la selección del más adecuado para cada aplicación, los microcontroladores PIC presentan un muy buen desempeño en cuanto a la velocidad con que ejecutan su programa y al reducido tamaño que ocupa en memoria el código de éste [1].

A pesar de su reducido número de instrucciones disponibles éstas ocupan sólo una palabra de memoria y se ejecutan en un solo ciclo de instrucción (excepto las de salto) aunque en ocasiones se tengan que ocupar varias para ejecutar una tarea que en otros microcontroladores exista como una única instrucción. Este inconveniente puede superarse gracias a las altas velocidades de procesamiento que alcanzan los PIC, pues son capaces de trabajar con frecuencias superiores a los 20 MHz en algunos modelos.

Otro aspecto a destacar de esta familia de microcontroladores es su alta capacidad de memoria de programa que alcanza en sus modelos mas grandes los 32 kilo bytes para almacenar hasta 16 kilo palabras. Para lograr un tamaño de memoria equivalente en modelos como el Intel 8051 o el Motorota 68HC11 se debe recurrir a memorias externas que elevan el costo final del diseño.

Todas estas prestaciones, además de la disponibilidad de diferentes recursos internos como temporizadores y convertidor A/D, fueron tomadas en cuenta para la selección de un microcontrolador PIC16F877 en el diseño que se desarrolla en el presente trabajo. Al final del capítulo anterior se describió este proceso de selección.

4.1.1. Características comunes

Los microcontroladores PIC (*Peripherial Interface Controller*) surgieron en 1975 con la finalidad de mejorar la capacidad de manejo de las entradas y salidas que tenía microprocesador de 16 bits CP1600 de GI Microelectronics. Posteriormente esta empresa cambió de dueños y de nombre para convertirse en la Arizona Microchip Tecnology. A partir de entonces se impulsó el desarrollo de la línea de microcontroladores PIC [2].

Estos microcontroladores presentan una arquitectura Harvard, por lo que la memoria de datos y programa se conectan al procesador mediante su propio conjuntos de buses. Todas las instrucciones tienen la misma longitud (entre 12 y 16 bits según el modelo) siendo este parámetro el que define la longitud de la palabra que almacena la memoria de programa. La memoria de datos almacena palabras de 8 bits (bytes).

Cuentan con un conjunto de instrucciones reducido (filosofía RISC) de entre 33 y 77 instrucciones, dependiendo de la familia. En su banco de registros están incluidos todos los elementos del sistema, desde los puertos hasta las localidades de memoria de propósito general.

Otra característica común que presentan todos los PIC es la segmentación o *pipe-line*. Mediante esta técnica el procesador obtiene el código de una instrucción en el mismo ciclo en que ejecuta la anterior. Esto permite que cada instrucción se ejecute en un solo ciclo de instrucción. Cuando la instrucción que se ejecuta genera un salto el código almacenado de la siguiente instrucción debe sustituirse por el de la instrucción que se encuentre en el destino del salto, por lo que en estos casos se requieren dos ciclos.

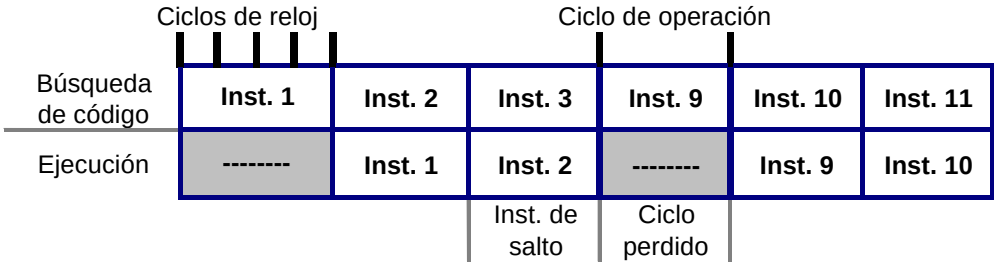


Figura 4.1: Técnica de segmentación.

4.1.2. Familias de microcontroladores PIC

La diversidad de modelos con diferentes capacidades y prestaciones permite seleccionar el mas adecuado para cada aplicación con una excelente relación costo – beneficios. Los diferentes modelos se agrupan por familias que comparten ciertas características y éstas a su vez se ubican dentro de alguna de las cuatro gamas presentadas en la tabla 4.1 [3].

Gama	Familias incluidas	Instrucciones disponibles	Niveles de la pila	Vectores de Interrupción
Básica	PIC12C5xx PIC16C5x	33 de 12 bits	2	No manejan interrupciones
Media	PIC12C6xx PIC16Cxx PIC16F8X PIC16F87x	35 de 14 bits	8	1
Alta	PIC17Cxxx	58 de 16 bits	16	4
Mejorada	PIC18Fxxx	77 de 16 bits	32	4

Tabla 4.1: Microcontroladores PIC agrupados por gamas.

4.2. Características generales y recursos

El PIC16F877 es el modelo con mas prestaciones dentro de su familia (*PIC16XXX*). La memoria de programa es de tipo Flash y permite hasta mil ciclos de borrado y escritura. La memoria de datos que contiene los registros es de tipo RAM y cuenta además con una memoria EEPROM con una vida de cien mil ciclos de borrado y escritura. La siguiente lista indica las principales características y recursos con que cuenta este modelo [4]:

- Encapsulado DIP de 40 pines[1] o PLCC de 44 pines.
- Capacidad para almacenar hasta 8192 palabras en la memoria de programa.
- Memoria de datos con 368 bytes en RAM y 256 bytes en EEPROM.
- Catorce diferentes fuentes de interrupción.
- Direccionamiento directo, indirecto y relativo.
- Temporizador Watchdog para prevenir fallas en el programa.
- Protección para evitar la lectura del código de programa.
- Funcionamiento en modo de bajo consumo de energía.
- Reinicio por baja tensión de alimentación.
- Reinicio automático al conectar la alimentación.
- Retardo para estabilización del reloj y del sistema.
- 33 líneas de entrada y salida organizadas en cinco puertos.
- Dos temporizadores de 8 bits y uno de 16 bits.
- Convertidor analógico-digital de 10 bits con ocho canales de conversión.
- Dos módulos para captura, comparación o modulación por ancho de pulso.
- Comunicación paralela (puerto esclavo) y serie (asíncrona y síncrona).

4.2.1. Terminales

En su mayoría las terminales son compartidas por dos o tres recursos, por lo que debe configurarse su funcionamiento en los registros que controlan cada recurso. La tabla 4.2 contiene las funciones de cada uno de los pines del PIC16F877 [5].

Pin	Nombre	Función
1	/MCLR	Entrada del impulso de reset. Activa en bajo.
	Vpp	Entrada para el voltaje de programación
2	RA0	E/S digital. Bit 0 del puerto A.
	AN0	Entrada analógica. Canal 0 del convertidor A/D.
3	RA1	E/S digital. Bit 1 del puerto A.
	AN1	Entrada analógica. Canal 1 del convertidor A/D.
4	RA2	E/S digital. Bit 2 del puerto A.
	AN2	Entrada analógica. Canal 2 del convertidor A/D.
	Vref (-)	Voltaje negativo de referencia para el convertidor A/D.
5	RA3	E/S digital. Bit 3 del puerto A.
	AN3	Entrada analógica. Canal 3 del convertidor A/D.
	Vref (+)	Voltaje positivo de referencia para el convertidor A/D.
6	RA4	E/S digital. Bit 4 del puerto A.
	T0CKI	Entrada de impulsos para el Timer0
7	RA5	E/S digital. Bit 5 del puerto A.
	AN4	Entrada analógica. Canal 4 del convertidor A/D.

[1] Un *Pin* es una terminal o *patita* de los circuitos integrados.

Pin	Nombre	Función
	/SS	Selección de modo esclavo o maestro para comunicación síncrona. Activa en bajo.
8	RE0	E/S digital. Bit 0 del puerto E.
	/RD	Señal de lectura para puerto paralelo. Activa en bajo.
	AN5	Entrada analógica. Canal 5 del convertidor A/D.
9	RE1	E/S digital. Bit 1 del puerto E.
	/WR	Señal de escritura para puerto paralelo. Activa en bajo.
	AN6	Entrada analógica. Canal 6 del convertidor A/D.
10	RE2	E/S digital. Bit 2 del puerto E.
	/CS	Activación del puerto paralelo. Activa en bajo.
	AN7	Entrada analógica. Canal 7 del convertidor A/D.
11, 32	VDD	Conexión del voltaje de alimentación.
12, 31	VSS	Conexión a tierra.
13	OSCI	Entrada del cristal.
	CLKIN	Entrada de la señal de reloj externa.
14	OSCO	Entrada del cristal.
	CLKOUT	Salida de la señal de reloj.
15	RC0	E/S digital. Bit 0 del puerto C.
	T1OSO	Salida del oscilador del Timer1.
	T1CKI	Entrada de señal de reloj externa para Timer1.
16	RC1	E/S digital. Bit 1 del puerto C.
	T1OSI	Entrada del oscilador del Timer1.
	CCP2	Entrada o salida del módulo CCP2.
17	RC2	E/S digital. Bit 2 del puerto C.
	CCP1	Entrada o salida del módulo CCP1.
18	RC3	E/S digital. Bit 3 del puerto C.
	SCK	Entrada de reloj serie síncrono.
	SCL	Salida síncrona.
19 - 22	RD0 - 3	E/S digitales. Bits 0 al 3 del puerto D.
	PSP0 - 3	Líneas 0 a la 3 de transmisión paralela.
23	RC4	E/S digital. Bit 4 del puerto C.
	SDI, SDA	Entrada de datos en transmisión serie.
24	RC5	E/S digital. Bit 5 del puerto C.
	SDO	Salida de datos en transmisión serie.
25	RC6	E/S digital. Bit 6 del puerto C.
	TX	Transmisión asíncrona.
	CK	Reloj serie síncrono.
26	RC7	E/S digital. Bit 7 del puerto C.
	RX	Recepción asíncrona.
	DT	Recepción de datos en transmisión serie síncrona.
27 - 30	RD4 - 7	E/S digitales. Bits 4 al 7 del puerto D.
	PSP4 - 7	Líneas 4 a la 7 de transmisión paralela.
33	RB0	E/S digital. Bit 0 del puerto B.
	INT	Solicitud de interrupción externa.
34 - 40	RB1 - 7	E/S digitales. Bits 1 al 7 del puerto B.

Tabla 4.2: Funciones asociadas con cada pin.

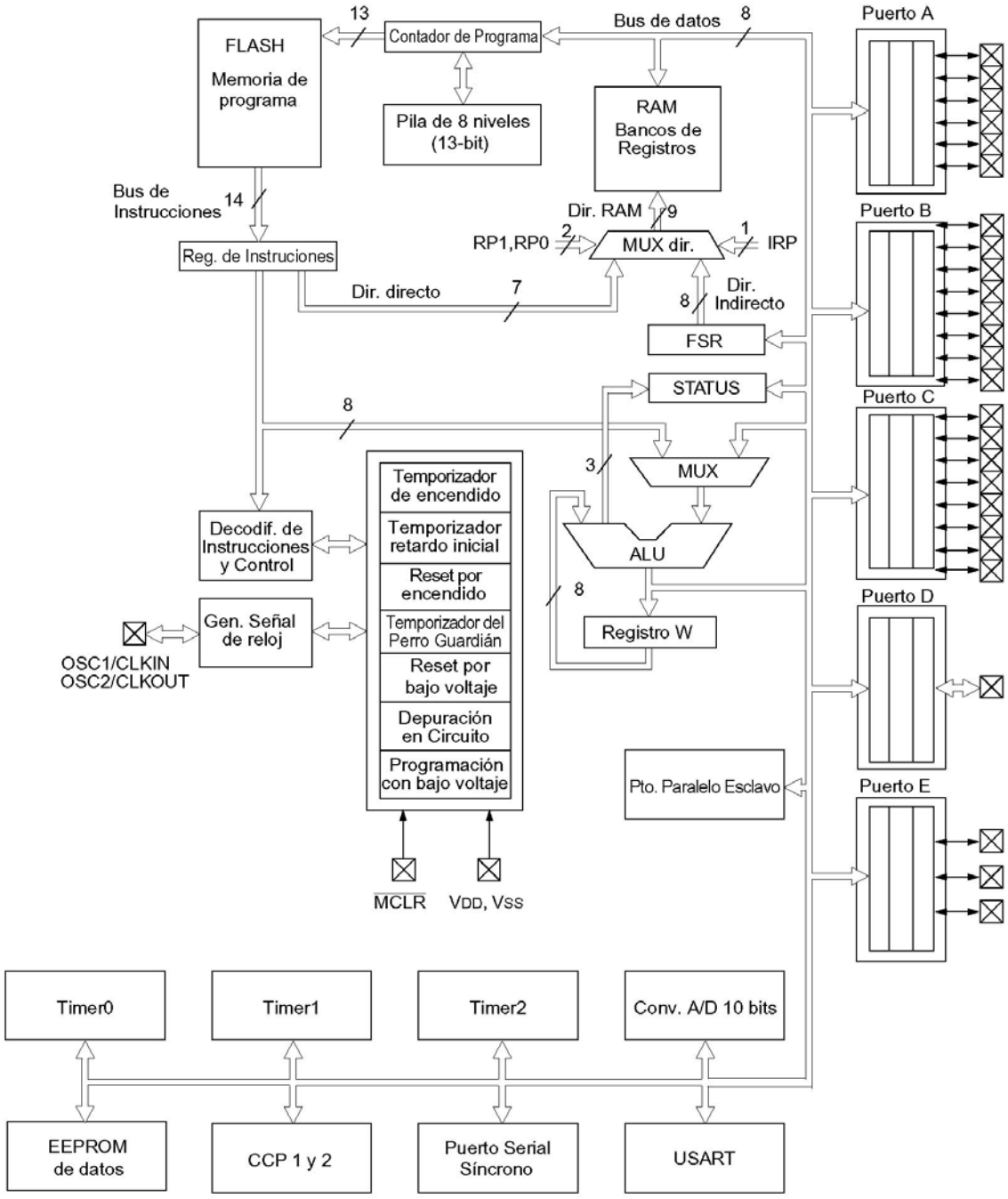


Figura 4.2: Diagrama a bloques de la arquitectura interna del PIC16F877 [6].

4.3. Características eléctricas

Las principales características eléctricas se presentan en la siguiente lista. Los rangos máximos de operación se describen en la tabla 4.3 [7].

- Posibilidad de seleccionar del tipo de oscilador.
- Frecuencia de operación desde DC y hasta 20 MHz.
- Programación serie *en circuito* por medio de dos pines.
- Programación con bajo o alto voltaje (5 y aprox. 12 volts respectivamente).
- Depuración *en circuito* mediante dos pines.
- Amplio rango para el voltaje de alimentación.
- Bajo consumo de corriente (depende del voltaje de alimentación y de la frecuencia de oscilación).

Parámetro	Rango
Voltaje en cualquier pin con respecto a V_{SS} (excepto V_{DD}, MCLR y RA4)	-0.3V a (V_{DD} + 0.3V)
Voltaje en V_{DD} con respecto a V_{SS}	-0.3 a +7.5V
Voltaje en MCLR con respecto a V_{SS}	0 a +14V
Voltaje en RA4 con respecto a V_{SS}	0 a +8.5V
La disipación total de potencia	1.0 W
Corriente máxima saliendo por V_{SS}	300 mA
Corriente máxima entrando por V_{DD}	250 mA
Corriente máxima que recibe o entrega cualquier pin de E/S	25 mA
Corriente máxima entregada o recibida por los puertos A, B y E combinados	200 mA
Corriente máxima entregada o recibida por los puertos C y D combinados	200 mA

Tabla 4.3: Rangos máximos de operación.

4.3.1. Ciclos de reloj y de instrucción

La señal de reloj se descompone internamente en cuatro señales tomando uno de cada cuatro impulsos para llevar a cabo las tareas que componen el ciclo de operación [8]:

1. Incremento del contador de programa
2. Decodificación de la instrucción cargada en el anterior ciclo
3. Ejecución de la instrucción decodificada
4. Obtención del código de la instrucción a ejecutar en el siguiente ciclo

El ciclo de instrucción se compone entonces de cuatro ciclos de reloj por lo que la frecuencia real de operación es una cuarta parte de la frecuencia de reloj.

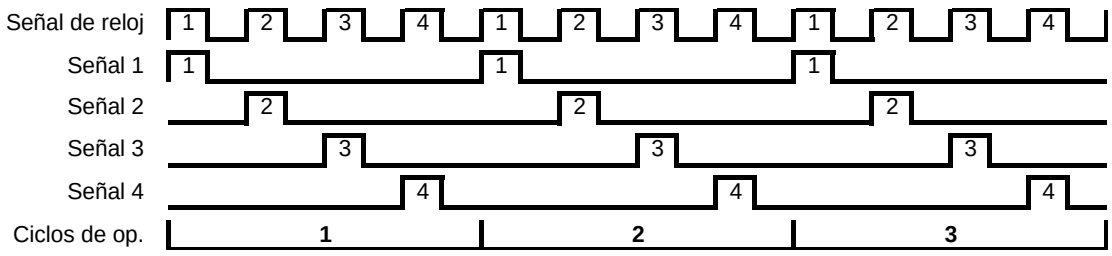


Figura 4.3: División de la señal de reloj.

4.3.2. Fuente de la señal de reloj

El PIC16F877 puede emplear cuatro tipos de osciladores para generar la señal de reloj. El más sencillo se obtiene mediante una resistencia y un capacitor externos que conforman el oscilador para el circuito generador de la señal de reloj (figura 4.4). Es el modo de operación que menos corriente consume pero la frecuencia no mantiene un nivel constante.

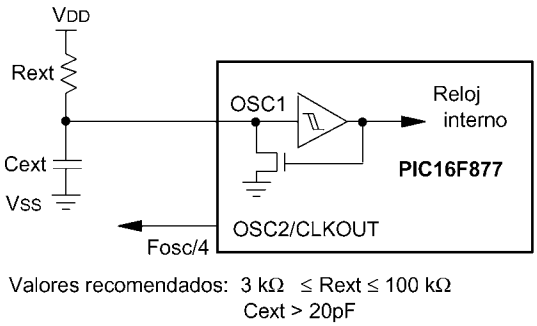


Figura 4.4: Oscilador tipo RC.

Los otros tipos emplean cristales de cuarzo o resonadores conectados a las terminales del circuito oscilador interno y a tierra por medio de dos capacitores (figura 4.5). La tabla 4.3 contiene los rangos de frecuencia del cristal, el rango dentro del cual debe estar el voltaje de alimentación y la corriente de alimentación para cada tipo de oscilador. La tabla 4.4 presenta el valor recomendado para los capacitores [9].

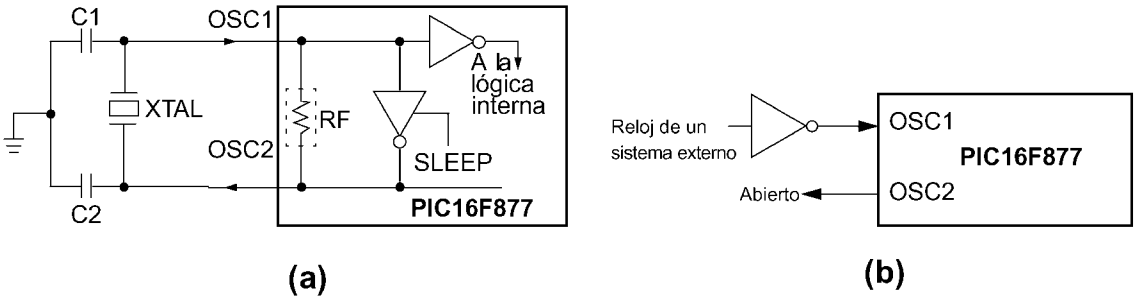


Figura 4.5: (a) Oscilador con cristal o resonador, (b) con señal de reloj externa.

Tipo de oscilador	Rango de Frecuencias	Voltaje de alimentación	Corriente de alimentación (Aprox.)
RC – Resistor / Capacitor	Depende de los valores de R y C	4.0 V – 5.5 V	Depende de la frecuencia
LP – Cristal de bajo consumo	20 kHz – 200 kHz	4.0 V – 5.5 V	30 μA - 110 μA
XT – Cristal / Resonador	200 kHz – 4 MHz	4.0 V – 5.5 V	0.8 mA – 1.8 mA
HS – Cristal / Resonador de alta velocidad	4 MHz – 20 MHz	4.5 V – 5.5 V	1.4 mA – 7.5 mA

Tabla 4.4: Tipos de osciladores

Resonadores cerámicos			Cristales		
Modo	Frecuencia	Capacitores	Modo	Frecuencia	Capacitores
XT	455 kHz	68 - 100 pF	LP	32 kHz	33 pF
	2.0 MHz	15 - 68 pF		200 kHz	15 pF
	4.0 MHz	15 - 68 pF	XT	200 kHz	47-68 pF
HS	8.0 MHz	10 - 68 pF		1 MHz	15 pF
	16.0 MHz	10 - 22 pF		4 MHz	15 pF
			HS	4 MHz	15 pF
				8 MHz	15-33 pF
				20 MHz	15-33 pF

Tabla 4.5: Valor de los capacitores.

4.4. Memoria

4.4.1. Memoria de programa

Se divide en cuatro páginas de 2048 palabras cada una y sus direcciones son de trece bits. Dos de ellos se destinan a la selección de la página y los once restantes determinan la posición dentro de ella. El contador de programa puede pasar de una a otra al incrementarse, pero en las instrucciones de salto o de llamada a subrutina sólo se cargan en él los once bits menos significativos.

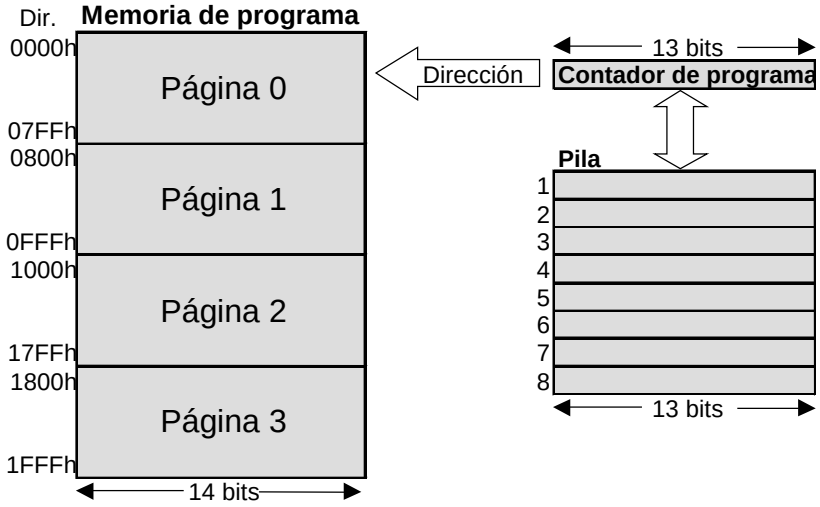


Figura 4.6: Memoria de programa y pila.

Los direccionamientos dentro de la memoria de programa se hacen en modo relativo, sumando la dirección de once bits incluida en el código de la instrucción con los dos bits más significativos el registro PCLATH que contiene la parte alta del contador de programa. Todos los saltos se realizan dentro de una misma página y para lograr que el salto tenga como destino una dirección dentro de otra, se debe modificar el registro PCLATH [10].

En la dirección **0000h** se encuentra el *vector de reset*. Cuando el PIC16F877 reinicia su funcionamiento comienza por ejecutar la instrucción almacenada en esta dirección. El *vector de interrupción* ocupa la dirección **0004h**, que es el destino con que se carga el contador de programa al presentarse una interrupción. Como la distancia entre estos vectores es pequeña se suele colocar en la dirección 0000h una instrucción de salto a otra localidad donde comienza el programa. De igual forma se puede cambiar la ubicación de la rutina de atención para interrupciones colocando en la dirección 0004h otro destino.

La pila

Consta de ocho niveles donde se pueden almacenar palabras de trece bits. Cuando se hace un llamado a subrutina o se presenta una interrupción, el valor del contador de programa se salva automáticamente en el nivel superior de ella y se recupera al ejecutarse las instrucciones de retorno de subrutina o interrupción. Los anidamientos dentro del programa deben limitarse a ocho, o siete cuando estén habilitadas las interrupciones, para que la pila no se desborde y el microcontrolador detenga su funcionamiento.

4.4.2. Memoria de datos

Está compuesta por una RAM estática de 386 bytes y una EEPROM de 256 bytes a la cual se accede mediante seis registros en RAM: dos para la dirección a leer o escribir, dos desde donde se leen o escriben los datos correspondientes y otros dos para controlar su funcionamiento. Los datos y las direcciones de la EEPROM son de solo 8 bits por lo que solo se necesita un byte para la dirección y otro para el dato. Los otros registros se ocupan para escribir y leer la memoria de programa que tiene direcciones de 13 bits y palabras de 14 bits.

La memoria RAM está organizada en cuatro bancos que pueden contener hasta 128 registros cada uno (figura 4.7), aunque algunas localidades no están implementadas o se repiten en varios bancos que contienen tanto registros de funciones específicas como de propósito general [11]. Para seleccionar el banco se emplean dos bits del registro de estado del sistema (RP1 y RP0) que se encuentra en todos ellos. Los bits de los registros pueden modificarse individualmente.

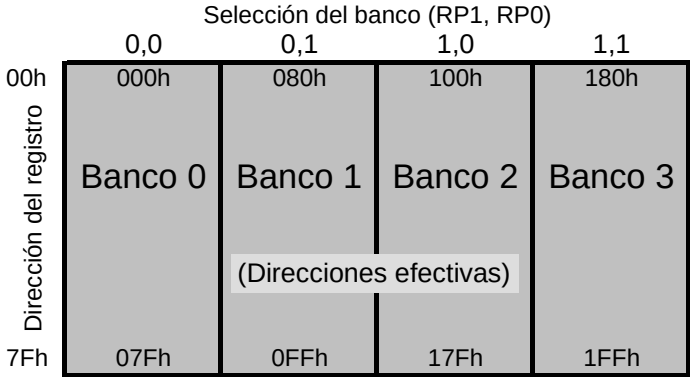


Figura 4.7: Organización de la memoria de datos.

Registros de funciones específicas (SFR)

Son los registros destinados a controlar el funcionamiento de todos los recursos del microcontrolador (tabla 4.6).

Banco 0				Banco 1			
Dir.	**Registro**	**Dir.**	**Registro**	**Dir.**	**Registro**	**Dir.**	**Registro**
00h	INDF	10h	T1CON	81h	INDF	90h	
01h	TMR0	11h	TMR2	81h	OPTION	91h	SSPCON2
02h	PCL	12h	T2CON	82h	PCL	92h	PR2
03h	STATUS	13h	SSPBUF	83h	STATUS	93h	SSPADD
04h	FSR	14h	SSPCON	84h	FSR	94h	SSPSTAT
05h	PORTA	15h	CCPR1L	85h	TRISA	95h	
06h	PORTB	16h	CCPR1H	86h	TRISB	96h	
07h	PORTC	17h	CCP1CON	87h	TRISC	97h	
08h	PORTD	18h	RCSTA	88h	TRISD	98h	TXSTA
09h	PORTE	19h	TXREG	89h	TRISE	99h	SPBRG
0Ah	PCLATH	1Ah	RCREG	8Ah	PCLATH	9Ah	
0Bh	INTCON	1Bh	CCPR2L	8Bh	INTCON	9Bh	
0Ch	PIR1	1Ch	CCPR2H	8Ch	PIE1	9Ch	
0Dh	PIR2	1Dh	CCP2CON	8Dh	PIE2	9Dh	
0Eh	TMR1L	1Eh	ADRESH	8Eh	PCON	9Eh	ADRESL
0Fh	TMR1H	1Fh	ADCON0	8Fh		9Fh	ADCON1
Banco 2				**Banco 3**			
Dir.	**Registro**	**Dir.**	**Registro**	**Dir.**	**Registro**	**Dir.**	**Registro**
101h	INDF	108h		181h	INDF	188h	
101h	TMRO	109h		181h	OPTION	189h	
102h	PCL	10Ah	PCLATH	182h	PCL	18Ah	PCLATH
103h	STATUS	10Bh	INTCON	183h	STATUS	18Bh	INTCON
104h	FSR	10Ch	EEDATA	184h	FSR	18Ch	EECON1
105h		10Dh	EEADR	185h		18Dh	EECON2
106h	PORTB	10Eh	EEDATH	186h	TRISB	18Eh	Reservado
107h		10Fh	EEADRH	187h		18Fh	Reservado

Tabla 4.6: Registros de funciones específicas.

Registros de propósito general (GPR)

Son localidades de memoria disponibles para almacenar las variables ocupadas por el programa (figura 4.8). Los 16 registros en las localidades mas altas del banco 0 son accesibles en todos los bancos.

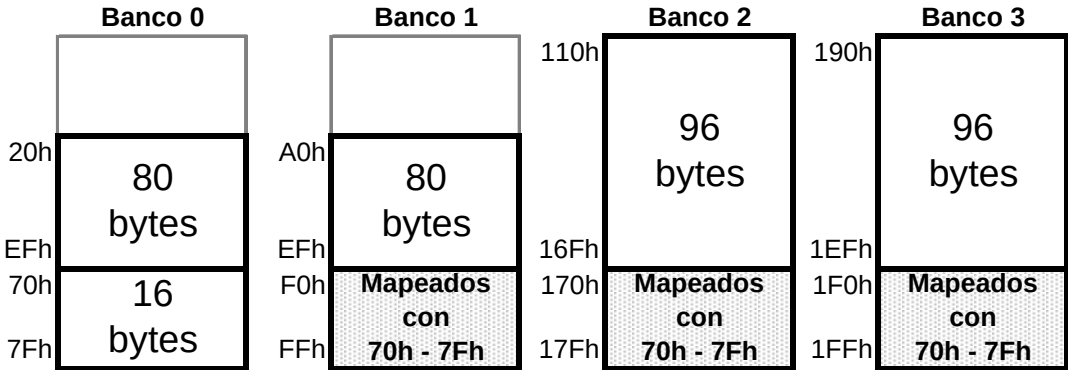


Figura 4.8: Ubicación de los registros de propósito general.

4.4.3. Modos de direccionamiento

Ya se ha mencionado que las direcciones efectivas de la memoria de programa se obtienen mediante el direccionamiento relativo. En el caso de la memoria de datos se emplean solo dos métodos de direccionamiento: directo e indirecto.

En el direccionamiento directo el código de la instrucción proporciona los siete bits de la dirección dentro del banco especificado por los bits RP1 y RP0 del registro de estado.

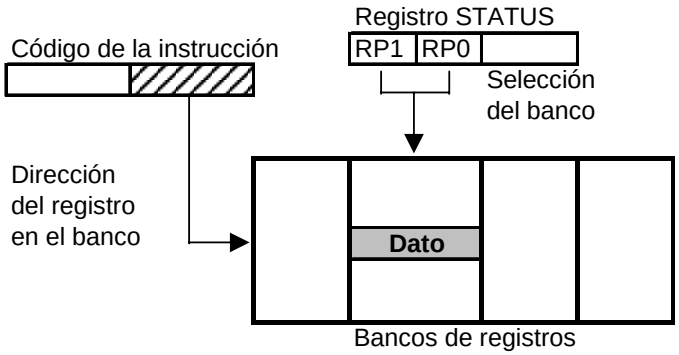


Figura 4.9: Direccionamiento en modo directo.

En el modo indirecto el operando de la instrucción es la dirección del registro INDF. Éste no existe físicamente, sino que es un enrutamiento variable controlado por otro registro llamado FSR. Esto que permite que cualquier otro registro pueda leerse y escribirse mediante la dirección del registro INDF. La instrucción no siempre utilizará la misma dirección como sucede en el direccionamiento directo, lo que permite trabajar con tablas de datos en memoria.

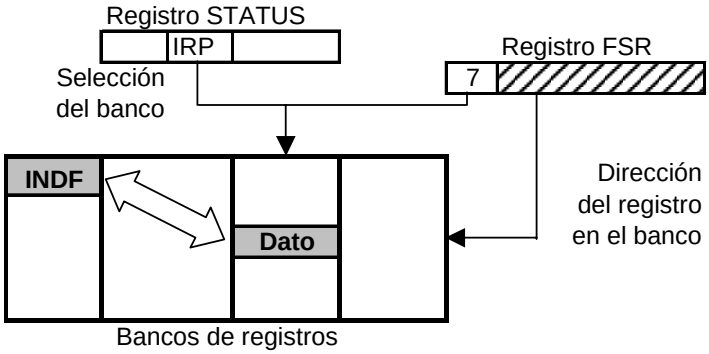


Figura 4.10: Direccionamiento en modo indirecto.

La dirección del registro al que conecta INDF se almacena dentro de los siete primeros bits de un registro llamado FSR. El bit restante se combina con el bit IRP del registro de estado para seleccionar el banco. Lo que se escriba dentro de INDF se verá reflejado en el contenido de la dirección que indica FSR [12].

4.5. Interrupciones

Cuando se presenta una interrupción el procesador del PIC16F877 verifica en primer lugar el bit de permiso global de interrupciones del registro INTCON para ver si debe proceder. Si las interrupciones están permitidas se verifica que fuente solicitó la interrupción para ver si está habilitada. Cuando una solicitud es aceptada se almacena el valor del contador de programa en el nivel superior de la pila y se carga con el valor 0004h, que es el vector de interrupción.

Dentro de la rutina de atención a la interrupción se debe primero que nada guardar el valor de los registros STATUS, PCLATH y W. Se deben guardar sus valores en los registros de las localidades mas altas del banco 0 (70h – 7Fh) debido a que están repetidas en los demás bancos.

Posteriormente, la rutina debe determinar que fuentes de interrupción tienen activa su bandera debido a que se pueden presentar varias solicitudes simultáneas. La primera de ellas genera la interrupción y las siguientes únicamente activarán su bandera. El orden en que se verifiquen éstas determinará la prioridad de las interrupciones.

Después de atender la primera fuente de interrupción se debe bajar su bandera para volver a verificar las demás banderas para atender todas las interrupciones simultáneas antes de salir de la rutina. Antes de regresar a la ejecución normal del programa se deben reestablecer los valores de los registros guardados. El contador de programa tomará su valor inicial de la pila al ejecutarse la instrucción de retorno de interrupción.

La tabla 4.7 presenta las posibles fuentes de interrupción. El registro INTCON tiene un bit para el permiso global de interrupciones (GIE) y otro para habilitar las fuentes de interrupción que no están presentes en él.

Fuente de interrupción	Bit de permiso – registro	Bit de bandera - registro
Interrupción externa (INT)	INTE - INTCON	INTF – INTCON
Desbordamiento del Timer 0	T0IE – INTCON	T0IF – INTCON
Cambio de estado en los pines RB4-RB7	RBIE – INTCON	RBIF – INTCON
Lectura o escritura en el puerto paralelo	PSPIE – PIE1	PSPIF – PIR1
Fin de conversión A/D	ADIE – PIE1	ADIF – PIR1
Recepción USART	RCIE – PIE1	RCIF – PIR1
Transmisión USART	TXIE – PIE1	TXIF – PIR1
Puerto síncrono	SSPIE – PIE1	SSPIF – PIR1
Captura o comparación en el módulo CCP1	CCP1IE – PIE1	CCP1IF – PIR1
Desbordamiento Timer 2	TMR2IE – PIE1	TMR2IF – PIR1
Desbordamiento Timer 1	TMR1IE – PIE1	TMR1IF – PIR1
Fin de escritura en EEPROM	EEIE – PIE2	EEIF – PIR2
Colisión en el bus SSP	BCLIE – PIE2	BCLIF – PIR2
Captura o comparación en el módulo CCP2	CCP2IE – PIE2	CCP2IF – PIR2

Tabla 4.7: Posibles fuentes de interrupción.

4.6. Puertos de entrada y salida

Todos los bits de los puertos pueden configurarse como entradas o salidas digitales. Cuando se lee el valor de un puerto, los bits de salida toman el valor almacenado en el registro y los de entrada el valor lógico de voltaje presente en su terminal. Cuando se escribe en un puerto únicamente se modifica el valor de los bits configurados como salidas.

La salida en las terminales de un puerto se pueden leer o escribir mediante un registro asociado al puerto. Los puertos se identifican mediante una letra, por lo que los registros son PORTA, PORTB, PORTC, PORTD y PORTE. Las terminales que funcionan como salida si su bit respectivo del registro de configuración del puerto (TRISA, TRISB, etc.) tiene el valor de 1. Si el bit está en 0 entonces esa línea del puerto estará configurada como entrada [13].

4.6.1. Puerto A

Cuenta con seis líneas por lo que los bits 6 y 7 de su registro se leen siempre como 0 y no son afectados por la escritura. Cuando se conecta la fuente de alimentación todas las terminales asociadas con este puerto, excepto RA4, quedan asignadas al convertidor A/D. Para asociarlas con el puerto A se debe modificar uno de los registros de control de éste (ADCON1). La terminal RA4 se destina a la entrada de impulsos externos para el Timer0, siempre que éste se configure así.

4.6.2. Puerto B

Cuenta con ocho líneas, tres de las cuales son ocupadas para la programación de la memoria de programa (RB3, RB6 y RB7). Todas las terminales cuentan con una resistencia de *pull-up* que se conecta automáticamente cuando están configuradas como salidas. Para conectarlas cuando las terminales sean entradas se debe poner a 0 el bit /RBPU del registro OPTION. La terminal RB0 puede funcionar como entrada para la solicitud de interrupción externa, mientras que las terminales RB4-RB7 pueden generar una solicitud al cambiar de estado lógico.

4.6.3. Puertos C, D y E

C. Cuenta con ocho líneas que comparten su terminal con varios recursos.
D. Comparte sus ocho líneas únicamente con el puerto paralelo.
E. Únicamente tiene tres líneas que también funcionan como líneas de control para el puerto paralelo.

4.7. Temporizadores

4.7.1. Timer 0

Este temporizador de 8 bits se puede configurar para que los incrementos se generen con cada ciclo de operación o con impulsos externos para trabajar como contador. Cuenta con un predivisor de frecuencia para que se incremente su valor con varios impulsos de reloj dependiendo del valor de los bits PS2-PS0 del registro OPTION. Comparte este predivisor con el *Watchdog* por lo que también se debe especificar en este mismo registro que recurso lo va a emplear.

PS2	PS1	PS0	División de la frecuencia
0	0	0	÷ 2
0	0	1	÷ 4
0	1	0	÷ 8
0	1	1	÷ 16
1	0	0	÷ 32
1	0	1	÷ 64
1	1	0	÷ 128
1	1	1	÷ 256

Tabla 4.8: División de la frecuencia para el Timer 0.

Cuando el Timer 0 se desborda y pasa de FFh a 00h activa una bandera y puede generar una solicitud de interrupción. Su valor puede modificarse en cualquier momento para establecer algún valor inicial.

4.7.2. Timer 1

Este temporizador de 16 bits funciona con los impulsos de la señal de operación del sistema o con impulsos provenientes del exterior. Su valor puede ser modificado en cualquier instante y al ser de 16 bits ocupa dos registros para almacena su cuenta. Cuenta con un predivisor que permite incrementar su valor cada dos, cuatro u ocho impulsos en lugar de uno. Al desbordarse y pasar de FFFFh a 0000h puede generar una solicitud de interrupción.

También puede funcionar como temporizador con una base de tiempo distinta a la del sistema si se conecta un cristal a sus dos terminales que comparte con el puerto C. La configuración del oscilador es similar a la que se emplea en el generador de la señal de reloj del sistema.

4.7.3. Timer 2

Es un temporizador de 8 bits que únicamente funciona con los impulsos del reloj del sistema. Cuenta con un predivisor similar al del Timer 1 que permite dividir su frecuencia de operación hasta por cuatro. Con cada desborde se puede generar una interrupción aunque esto se puede modificar mediante un postdivisor para que esto suceda cada dos, cuatro u ocho desbordes, aumentando el tiempo que transcurre para la interrupción.

4.8. Convertidor analógico-digital

Tiene una resolución de 10 bits y cuenta con ocho diferentes canales de conversión que comparten sus terminales con el puerto A y el puerto E. El voltaje de referencia para la conversión puede provenir del exterior o de la fuente de alimentación. En cualquier caso su valor se obtiene mediante la ecuación [14]:

$$V_{ref} = V_{ref}(+) - V_{ref}(-) \qquad (4.1)$$

Si los voltajes de referencia (+) y (-) provienen de la fuente de alimentación:

$$V_{ref} = V_{DD} - V_{SS} = V_{DD} \qquad (4.2)$$

El voltaje que representa cada bit en binario, o resolución, se obtiene dividiendo el voltaje de referencia entre el número de intervalos de comparación que para 10 bits es:

$$Intervalos = 2^{10} = 1024 \qquad (4.3)$$

Y la resolución se obtiene mediante [15]:

$$\text{Resolución} = \frac{V_{ref}}{1024}\left[\frac{\text{mV}}{\text{bit}}\right] \qquad (4.4)$$

El convertidor se controla mediante los registros ADCON0 y ADCON1. El primero determina el tiempo de conversión y el canal que se empleará para obtener la muestra. Por medio de él se enciende el convertidor y se inician las conversiones.

El registro ADCON1 determina qué terminales funcionan como entradas y salidas digitales para el puerto A y cuáles se ocupan para los canales y voltajes de referencia. Indica también mediante el bit ADFM la forma en que se colocará el resultado de la conversión dentro de los registros ARDES y ADRESL, como se muestra en la figura 4.11.

Antes de realizar cualquier conversión se deben configurar las terminales que funcionarán como canales de conversión, determinar el origen del voltaje de referencia y seleccionar el tiempo de conversión. Para realizar cada conversión se debe seguir el siguiente procedimiento [16]:

- Seleccionar el canal de entrada analógica.
- Activar la interrupción de fin de conversión (opcional).
- Esperar un tiempo para que se complete la adquisición de la muestra de voltaje.
- Iniciar la conversión.
- Esperar a que termine la conversión o esperar a la interrupción.
- Leer el resultado en los registros.

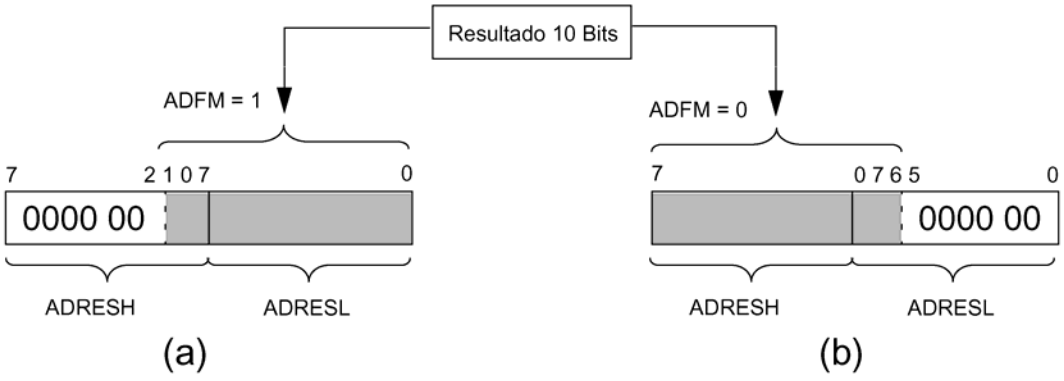


Figura 4.11: Justificación del resultado de la conversión a la derecha (a) y a la izquierda (b).

4.9. Otros recursos

4.9.1. Palabra de configuración

Ocupa la dirección 2007h de la memoria de programa y solo puede modificarse durante la programación. Determina:

- Activación de la depuración *en circuito*.
- Habilitación de escritura en memoria de programa.
- Protección de la memoria EEPROM.
- Habilitación de programación con bajo voltaje.
- Habilitación de reinicio por bajo voltaje.
- Activación del retardo inicial al conectar la alimentación.
- Activación del *Watchdog*.
- Selección del tipo de oscilador.

4.9.2. Palabras de identificación

Son cuatro localidades de la memoria de programa disponibles para almacenar código de identificación. Únicamente se ocupan 4 bits de cada una y se modifican al programar el microcontrolador.

4.9.3. Watchdog

Es un temporizador que se incrementa con impulsos del reloj del sistema y reinicia el sistema al desbordarse. Protege al sistema contra el *congelamiento* por errores en el programa. Comparte el divisor de frecuencia con el Timer 0 y su valor puede ponerse en ceros en cualquier momento.

4.9.4. Modo de bajo consumo

Al entrar en este modo de operación, el PIC16F877 reduce su consumo de energía deteniendo su funcionamiento. El estado de las terminales se mantiene pero el resto de los recursos (menos el Watchdog) dejan de operar. Este modo comienza con la ejecución de la instrucción *seleep* concluye hasta que se presente una interrupción o se reinicie el microcontrolador.

Esto es de especial utilidad en aplicaciones que emplean baterías y en que el microcontrolador no necesariamente debe estar funcionando en todo instante. Por ejemplo, una dispositivo de muestreo sísmico que se ubique en una zona donde no exista red eléctrica y que solo deba tomar lecturas cuando se presente un movimiento telúrico de cierta intensidad (por ejemplo en las faldas de un volcán).

En este caso el PIC entraría en modo de bajo consumo hasta que un componente adicional active la interrupción externa al presentarse las condiciones de intensidad sísmica requeridas para comenzar a tomar lecturas para ser almacenadas en memoria y/o enviadas a una estación receptora. En este caso, aunque algunos elementos necesariamente deberán estar funcionando en todo instante, el microcontrolador sólo trabajará cuando sea necesario.

4.9.5. Módulos CCP

El PIC16F877 cuenta con dos módulos que permiten:

- La **captura** del valor del Timer 1 cuando se presenta un evento en la terminal CCP1 o CCP2.
- La **comparación** del valor en el Timer 1 con respecto al de un par de registros. Cuando estos sean iguales se producirá un evento en la terminal CCP1 o CCP2.
- La **modulación por ancho de pulso**, generando una serie de pulsos de duración variable en la terminal CCP1 o CCP2.

4.9.6. Módulos de comunicación

Dispone de tres módulos que le permiten comunicarse directamente con otros sistemas [17]:

1. Módulo **MSSP** (Puerto serie síncrono) que emplea los protocolos SPI (Serial Peripheral Interface) e I2C (Inter-Integrated Circuit).
2. Módulo **SCI** (Serial Comunications Interface) para comunicación serie síncrona y asíncrona.
3. Puerto paralelo esclavo capaz de conectarse con el bus de datos de un microprocesador.

4.10. Conjunto de instrucciones

Cuenta con 35 instrucciones que se agrupan de acuerdo con el tipo de operandos que ocupan o la función que realizan [18]. La figura 4.12 muestra el formato de instrucción para cada uno.

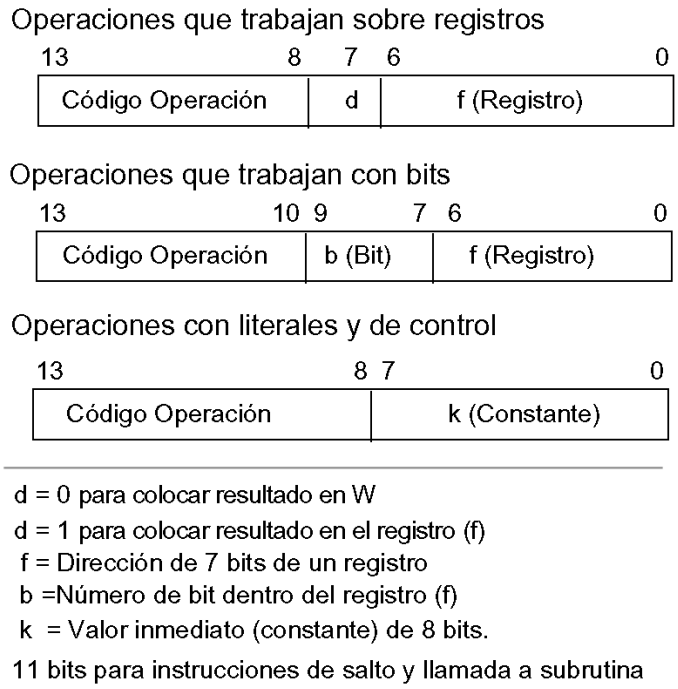


Figura 4.12: Formatos de instrucción.

Las instrucciones que trabajan con registros permiten almacenar el resultado en éstos o en el registro de trabajo W.

Operaciones con constantes y de control		
Nemónico	**Operandos**	**Función**
ADDLW	k	Sumar una constante a W
ANDLW	k	AND de una constante con W
CALL	k	Llamada a subrutina
CLRWDT		El borrar el cronómetro del Watchdog
GOTO	k	Saltar a una dirección
IORLW	k	OR de una constante con W
MOVLW	k	Guardar una constante en W
RETFIE		Volver de interrupción
RETLW	k	Volver con una constante en W
RETURN		Volver de una subrutina
SLEEP		Entrar en el modo de espera
SUBLW	k	Substraer W de una constante
XORLW	k	OR exclusiva de una constante con W
Operaciones que trabajan con registros		
Nemónico	**Operandos**	**Función**
ADDWF	f, d	Sumar W con f
ANDWF	f, d	AND de W con f
CLRF	f	Borrar f
CLRW		Borrar W
COMF	f, d	Complementar f
DECF	f, d	Decrementar f
DECFSZ	f, d	Decrementar f, Salto si 0
INCF	f, d	Incrementar f
INCFSZ	f, d	Incrementar f, Salto si 0
IORWF	f, d	OR de W con f
MOVF	f, d	Mover f
MOVWF	f	Mover W a f
NOP		No operar
RLF	f, d	Rotar f a la izquierda a través del Carry
RRF	f, d	Rotar f a la derecha a través del Carry
SUBWF	f, d	Substraer W de f
SWAPF	f, d	Intercambiar los nibbles[1] en f
XORWF	f, d	OR exclusiva de W con f
Operaciones que trabajan con bits		
Nemónico	**Operandos**	**Función**
BCF	f, b	Poner a 0 un bit de f
BSF	f, b	Poner a 1 un bit de f
BTFSC	f, b	Verifica un bit de f, Salta si está en 0
BTFSS	f, b	Verifica un bit de f, Salta si está en 1

Tabla 4.10: Conjunto de instrucciones del PIC16F877.

[1] Los nibbles son los dos grupos de 4 bits que conforman un byte. Uno constará de los cuatro más significativos y el otro de los cuatro menos significativos.

4.11. Programación y desarrollo

Los microcontroladores PIC16F877 se programan empleando cinco terminales mediante un formato de transmisión y recepción serie.

- V_{DD} conectada a 5 volts.
- V_{SS} conectada a tierra.
- V_{PP} conectada a un voltaje de entre 12 y 14 volts.
- RB6 recibe los impulsos de reloj.
- RB7 recibe los datos a almacenar en serie.

Cuando se programa en con bajo voltaje se deben conectar las terminales RB3 y V_{PP} a 5 volts.

Microchip fabrica programadores para microcontroladores PIC que trabajan en conjunto con las herramientas de software que diseña para el desarrollo con estos dispositivos [19]:

- PRO MATE® II: Programador universal.
- PICSTART® Plus: Programador de prototipos.
- KEELOQ®: Kit de evaluación y programador.

El sencillo mecanismo de programación con que cuentan los PIC permite desarrollar circuitos grabadores económicos que se conectan a la PC mediante un puerto serial o paralelo. Existen también programas de software que controlan el funcionamiento de los circuitos grabadores desde la PC.

En cuanto al software de desarrollo, Microchip distribuye una gran variedad de programas individuales o en grupo que pueden funcionar con sus programadores, simuladores de hardware y emuladores:

- MPASM Assembler: Ensamblador.
- MPLAB™ SIM: Software de simulación.
- MPLAB-C17: Compilador de lenguaje C.
- fuzzyTECH-MP: Sistema de desarrollo para lógica difusa.
- MPLAB® IDE: Entorno de desarrollo para microcontroladores PIC.

Los dos primeros programas se pueden obtener de manera gratuita desde la página de Internet de Microchip en *http://www.microchip.com*, los restantes deben comprarse.

4.12. Referencias

1 José Ma. Angulo, Martín Cuenca, *Microcontroladores PIC. La solución en un chip*, “La familia de los PIC”, Barcelona: Ed. Paraninfo, 1998, p. 42.

2 *Ibíd*., p. 38.

3 José Ma. Angulo, Susana Romero e Ignacio Angulo, *Microcontroladores PIC. Diseño práctico de aplicaciones. Segunda parte: PIC16F87X*, Madrid: Ed. McGraw Hill, 2000, pp 5-6.

4 *PIC16F87X Data Sheet. 28/40-Pin 8-Bit CMOS FLASH Microcontrollers*, Microchip Technology Inc., USA: 2001, p. 1.

5 *Ibíd*., pp. 8-9.

6 *Ibíd*., p. 6.

7 *Ibíd*., pp. 1, 149.

8 José Ma. Angulo, Ignacio Angulo, *Microcontroladores PIC. Diseño práctico de aplicaciones*, “El primer contacto con el PIC16X84”, 2a. ed., Madrid: Ed. McGraw Hill, 1999, pp. 53-54.

9 *PIC16F87X Data Sheet*, *Op. Cit.*, pp. 121-123, 152-153.

10 *Ibíd.*, p. 26.

11 *Ibíd*., p. 12.

12 *Ibíd*., p. 27.

13 *Ibíd*., p. 29.

14 José Ma. Angulo, Susana Romero e Ignacio Angulo, *Op. Cit.*, p. 129.

15 *Ídem*.

16 *PIC16F87X Data Sheet*, *Op. Cit.*, pp. 111-116.

17 *Ibíd*., pp. 38-39, 65-109.

18 *Ibíd*., p. 135.

19 *Ibíd*., pp. 143-146.

Capítulo V

Desarrollo del Hardware

5.1. El microcontrolador

5.1.1. Circuito básico

En el apartado 4.2.1 se describieron las terminales del PIC16F877, de las que 33 son líneas de entrada-salida y las siete restantes están destinadas a las necesidades básicas de funcionamiento del microcontrolador.

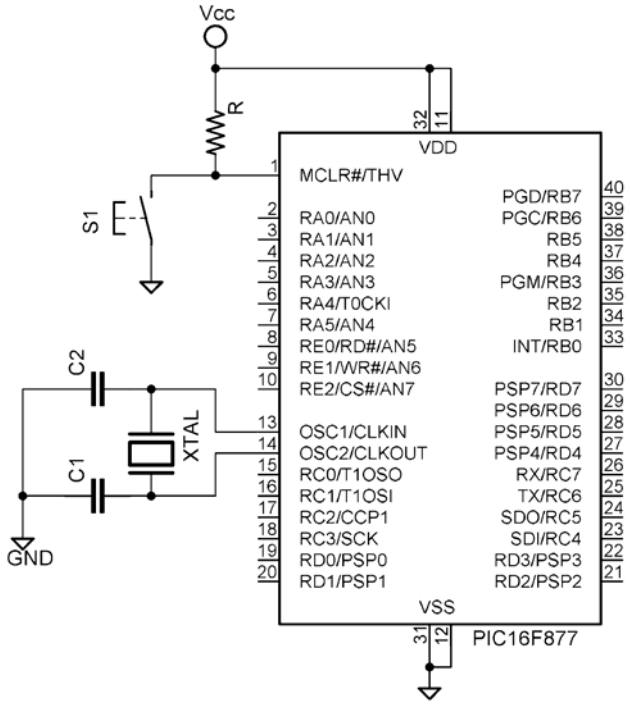


Figura 5.1: Circuito básico para el funcionamiento del PIC16F877.

Las terminales de alimentación están duplicadas por lo que se destinan dos al voltaje positivo y dos a la conexión con la tierra de la fuente. El voltaje debe ser estable y su valor estará entre los 4.5 y los 5.5 volts. En los dispositivos digitales es común el empleo de una alimentación de 5 volts pero este valor se modificará según las necesidades del sistema diseñado.

El PIC16F877 cuenta con un circuito interno para generar el *reset* al conectarse la alimentación, por lo que no requiere de un circuito externo para desempeñar esta función. La terminal /MCLR únicamente se conecta a tierra cuando se desea provocar un reinicio de forma manual. Para el dispositivo diseñado se emplea un interruptor normalmente abierto para que cuando se oprima

se conecte la terminal mencionada a tierra. Cuado el interruptor se encuentre abierto la terminal /MCLR estará conectada a un nivel alto de voltaje [1].

Las dos terminales restantes se emplean para la señal de reloj. Cuando el microcontrolador funciona con su propia señal de reloj se deberá conectar un cristal o resonador cerámico a las terminales OSC1 y OSC2. Si operara mediante un oscilador RC o empleando una señal de reloj de otro sistema únicamente se emplearía la terminal OSC1.

5.1.2. Frecuencia de operación y base de tiempo

Para determinar la frecuencia de operación, y con ello la del oscilador, se consideraron los siguientes aspectos:

- Estabilidad de la señal de reloj.
- Velocidad de procesamiento[1] y capacidad para la medición de intervalos de tiempo.
- Consumo de corriente.
- Costo de los componentes requeridos.

De los cuatro modos de operación para el oscilador del PIC16F877 se descartó el modo RC debido a que la frecuencia es susceptible a las variaciones en el valor de la resistencia, la capacitancia y el voltaje de alimentación. La figura 5.2 muestra su comportamiento con diferentes valores de resistencia y voltaje con un capacitor de 100 pF [2].

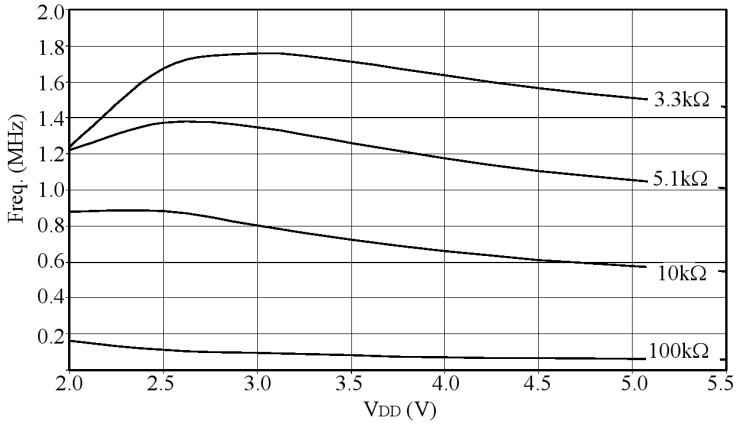


Figura 5.2: Variación de la frecuencia en el modo RC.

El modo LP no fue tomado en cuenta debido a que la frecuencia de operación tiene un valor menor a los 200kHz (tabla 4.4), por lo que la ejecución de las instrucciones toma un mayor tiempo, debido a la mayor duración del periodo de la señal de reloj. Para obtener una alta velocidad de procesamiento se deberá seleccionar uno de los modos de frecuencia alta (XT o HS).

Aunque en el modo HS se obtiene la mayor velocidad de procesamiento, el consumo de corriente puede ser demasiado alto. La gráfica de la figura 5.3 presenta el comportamiento en

[1] Se refiere a las operaciones que se podrían ejecutar en un determinado intervalo de tiempo. Se define por medio de la frecuencia de operación y es una cuarta parte de la señal de reloj.

este modo de la corriente máxima de alimentación (en miliamperes) en función de la frecuencia del cristal (MHz) y para diferentes voltajes de alimentación [3].

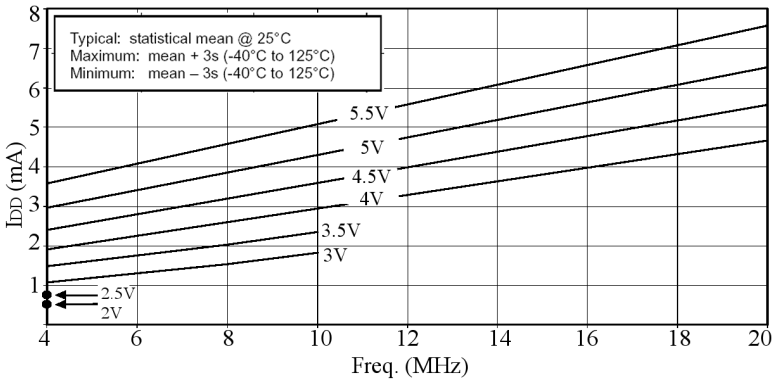


Figura 5.3: Comportamiento de la corriente máxima de alimentación en el modo HS.

Un mayor equilibrio entre consumo y velocidad se presenta en los cristales de frecuencias comprendidas entre uno y cuatro mega hertz. En la figura 5.4 se presentan las variaciones de la corriente máxima y en la figura 5.5 las de la corriente típica para el modo de operación XT [4]. La tabla 5.1 presenta los valores característicos de frecuencia para algunos cristales de valor comercial.

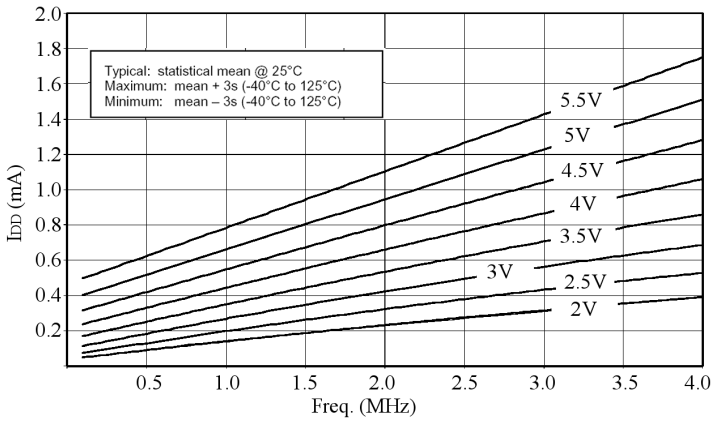


Figura 5.4: Comportamiento de la corriente máxima de alimentación en el modo XT.

Frecuencia del cristal [MHz]	Frecuencia de operación [kHz]	Ciclo de operación [us]
1.000000	250.000	4.000000
2.000000	500.000	2.000000
4.000000	1000.000	1.000000

Tabla 5.1: Frecuencia de operación y capacidad de medición de tiempo de los temporizadores.

En cualquier caso el ciclo de operación es suficientemente corto para ejecutar un buen número de instrucciones por cada segundo (frecuencia de operación), pero el cristal de 4 MHz proporciona un buen equilibrio entre velocidad de operación y consumo de corriente, además de un bajo costo.

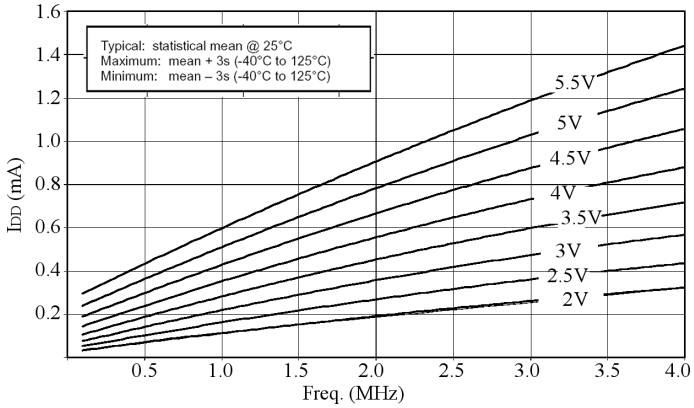


Figura 5.5: Comportamiento de la corriente típica de alimentación en el modo XT.

Base de tiempo

Para la medición de tiempo se empleará el Timer 1 que es el de mayor rango por ser de 16 bits. Cada vez que éste cuente el intervalo correspondiente a su rango, generará una interrupción en la que se llevarán a cabo los procedimientos asociados con el tiempo. En el apartado 2.2.1 se estableció un incremento mínimo de un segundo, que se puede cubrir mediante una o varias solicitudes de interrupción. Cuando el Timer 1 trabaja con el predivisor activado se obtiene su máximo rango mediante:

$$Rango_{Max} = 2^{16} \times \text{Predivisor} = 65536 \times \text{Predivisor} \qquad (5.1)$$

Predivisor	Rango	Empleando 15 bits
Sin	65536	32768
2	131072	65536
4	262144	131072
8	524288	262144

Tabla 5.2: Rango del Timer 1 de 16 bits.

El Timer 1 puede incrementarse con la señal proveniente de otro oscilador conectado a las terminales T1OSO y T1OSI. La señal aplicada a él es independiente a la de reloj, lo que permite seleccionar cualquiera de los cristales de la tabla 5.1 para el funcionamiento del microcontrolador, y otro con una frecuencia adecuada para contar un segundo mediante este temporizador.

Un cristal de 32.768 kHz genera 32768 impulsos cada segundo. Como el rango del Timer 1 sin predivisor es del doble, sería necesario reducirlo a la mitad colocando a 1 su bit más significativo antes de que se presente la primera interrupción y al ocurrir ésta o las siguientes, con lo que se estarían ocupando solo 15 bits del temporizador. La figura 5.6 presenta la conexión de ambos cristales con el PIC16F877.

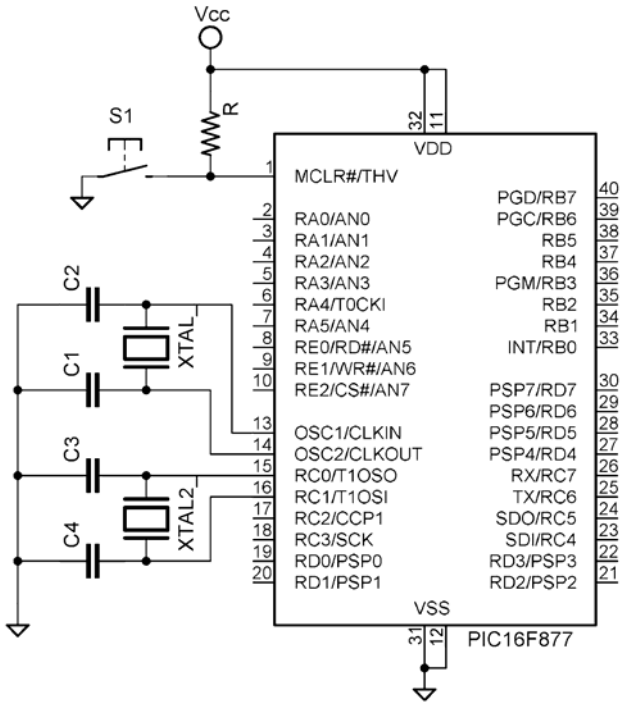


Figura 5.6: Conexión de un cristal para el Timer 1.

5.1.3. Interruptores

La figura 5.7 presenta el circuito empleado para los interruptores. No incluyen circuitos contra rebotes para reducir el número de componentes externos al microcontrolador. Se eligió esta configuración debido a que la corriente circula por las resistencias solo cuando es presionado uno de los interruptores.

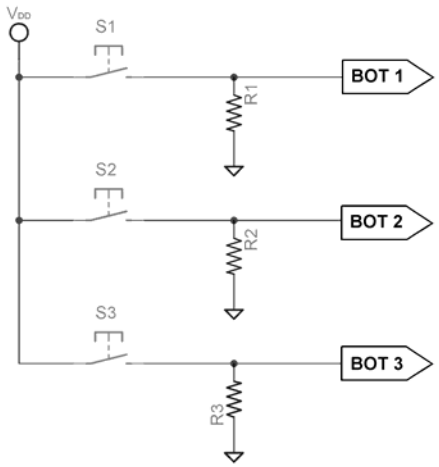


Figura 5.7: Circuito empleado para los interruptores.

El valor mínimo de las resistencias lo determina la máxima corriente que puede proveer o recibir cada terminal del PIC16F877 (25 mA). Para su cálculo se emplea el máximo voltaje de alimentación del microcontrolador:

$$R_{\min} = \frac{V_{DD\_MAX}}{I_{MAX\,/\,PIN}} = \frac{5.5}{25 \times 10^{-3}} = 220\Omega \qquad (5.2)$$

Sin embargo, con la finalidad de consumir menos corriente se emplean resistencias de valor grande, superior a los 100kΩ con lo que la corriente será menor a 55μA.

Para evitar la atención a pulsaciones cortas que sean producto de algún rebote se implementará por software el algoritmo descrito mediante el diagrama de flujo de la figura 5.8.

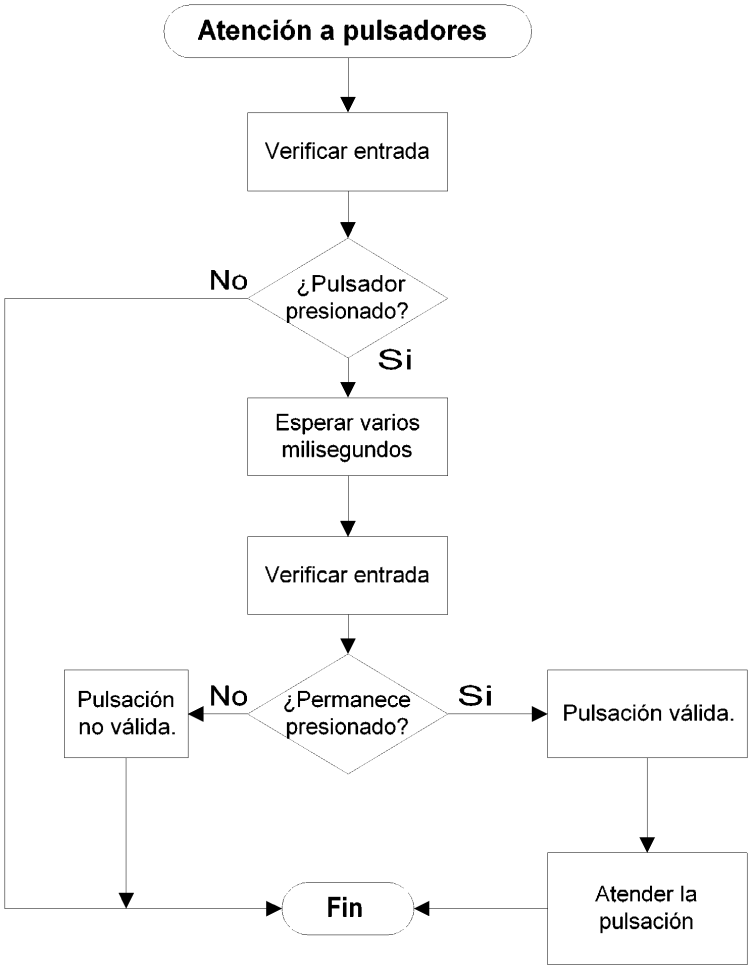


Figura 5.8: Algoritmo para contrarrestar el efecto de los rebotes mediante software.

5.2. Entradas analógicas

5.2.1. Sensores de temperatura

Todos los transductores de temperatura mencionados en el apartado 3.6.1 requieren de otros elementos para obtener una medida eléctrica en voltaje. La ventaja de los transductores de estado sólido (diodo en polarización directa) es que al hacerse de un material semiconductor

pueden incluirse dentro del circuito integrado que adecua la señal. Algunos ejemplos comerciales de sensores con salida analógica se enlistan en la tabla 5.3 [5].

Sensor	Rango de operación	Resolución	Exactitud
LM135	-55 a +150 ºC	10 mV/Kelvin	± 1.5 ºC
LM135A	-55 a +150 ºC	10 mV/Kelvin	± 1 ºC
LM235	-40 a +125 ºC	10 mV/Kelvin	± 1.5 ºC
LM235A	-40 a +125 ºC	10 mV/Kelvin	± 1 ºC
LM334	0 a +70 ºC	Corriente proporcional a un Kelvin	± 6 ºC
LM335	-40 a +100 ºC	10 mV/Kelvin	± 2 ºC
LM335A	-40 a +100 ºC	10 mV/Kelvin	± 1 ºC
LM34	+32 a +212 ºF y -40 a +230 ºF	10 mV/ºF	± 4 ºF, ± 3 ºF
LM35	0 a +100 ºC, -40 a +110 ºC y -55 a +150 ºC	10 mV/ºC	± 2 ºC, ± 1.5 ºC

Tabla 5.3: Modelos comerciales de sensores de temperatura.

El modelo LM35 es un sensor de precisión calibrado en grados Celsius que tiene la ventaja de ser económico. El voltaje en su salida tiene una relación lineal de 10mV por grado Celsius y no requiere ninguna calibración externa para obtener una buena exactitud. Funciona con alimentación única cuyo valor puede variar entre 4 y 30 volts sin modificar su funcionamiento [6].

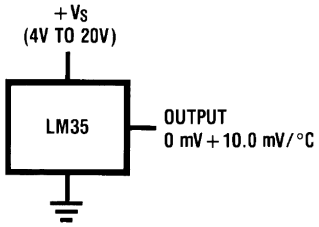


Figura 5.9: Diagrama electrónico del sensor de temperatura LM35.

La versión a emplear será la DZ (LM35DZ) que es la que tiene un menor costo pero se ajusta a las necesidades del diseño. Sus principales características son:

- Rango de temperaturas de 0 ºC a 100 ºC.
- Exactitud típica de ± 0.9 ºC.
- Resolución de 10mV/ºC.
- Alimentación entre +4V y +30V.
- Consumo de corriente de 160 μA.

En una medición la exactitud de la lectura dependerá básicamente de la calibración del sensor y de la resolución del convertidor AD. Aunque los niveles de voltaje que entrega el circuito LM35 son pequeños, no se amplificará la señal pues en este proceso se podría introducir cierto error. Para aprovechar la calibración del sensor se buscará ajustar la resolución del convertidor analógico-digital para aumentar la exactitud de la precisión.

Para el dispositivo se empleará un sensor LM35 para la temperatura ambiente y otro para la corporal. Se conectarán al PIC16F877 por medio de sus canales de conversión.

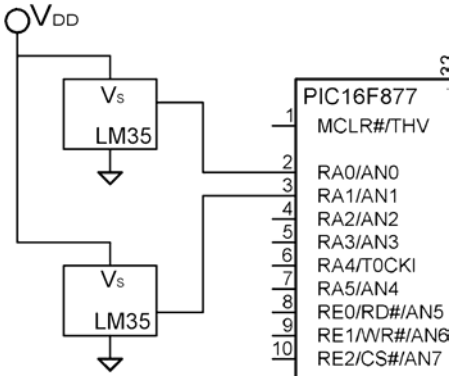


Figura 5.10: Conexión de los sensores de temperatura con el PIC16F877.

5.2.2. Sensor de intensidad luminosa

Se emplea una fotorresistencia para transformar las alteraciones de intensidad de luz en alteraciones del valor de una resistencia. Mediante un divisor de voltaje se obtienen variaciones de voltaje proporcionales a las variaciones de la luz.

Para determinar la ubicación de la fotorresistencia en el divisor (ver figura 5.11) se observó el comportamiento de la ecuación del voltaje de salida del divisor cuando la resistencia que varía es R_2 y cuando lo hace R_1.

$$V_0 = \frac{R_2}{R_1 + R_2} V_{DD} \qquad (5.3)$$

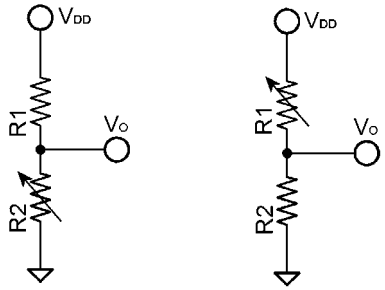


Figura 5.11: Posibles configuraciones de divisor de voltaje. La resistencia que varia representa la fotorresistencia.

La variación de R_2 con R_1 fija tendrá como forma general la de la ecuación 5.4, mientras que para R_2 fija y R_1 variable será la de la ecuación 5.5. En la figura 5.12 se observa este comportamiento.

$$f = \frac{x}{1+x} \qquad (5.4) \qquad\qquad f = \frac{1}{1+x} \qquad (5.5)$$

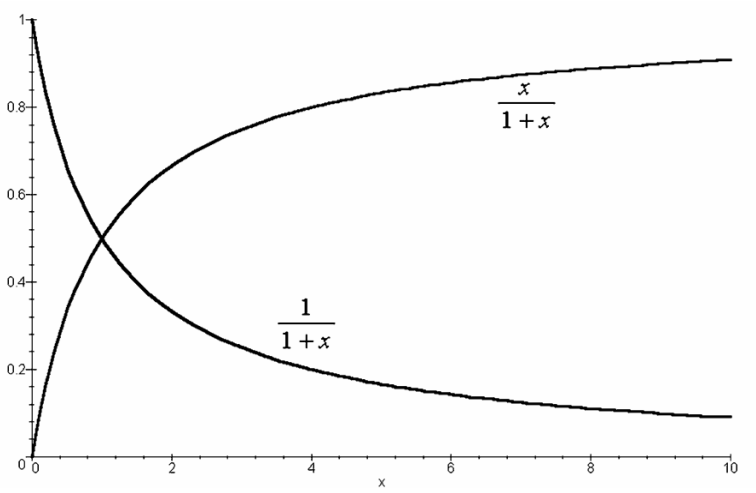


Figura 5.12: Comportamiento de la ecuación del divisor de voltaje cuando varía R_1 y cuando lo hace R_2.

La fotorresistencia aumenta su valor al disminuir la cantidad de luz incidente en su superficie, por lo que el valor de su resistencia tiene un comportamiento inversamente proporcional a la cantidad de luz.

Si en la ecuación 5.3 el valor de R_1 permanece constante y el valor de R_2 se asocia con la fotorresistencia, se invertirá este comportamiento, pues al aumentar la resistencia disminuye el voltaje. Con esto, un incremento en la luz incidente producirá un aumento en el voltaje de salida del divisor.

De la gráfica de la figura 5.12 se puede observar que el comportamiento no es lineal, pero la lectura únicamente se empleará para comparaciones. El valor del voltaje de salida para ciertas condiciones de iluminación determinará los límites de los intervalos mediante los que el microcontrolador comparará las lecturas para determinar el encendido o apagado del indicador luminoso.

Determinación de valores

La fotorresistencia empleada presenta una resistencia de 130 kΩ a plena luz y de 10 MΩ en condiciones de oscuridad. Mediante estos valores se pueden determinar las ecuaciones que determinan el valor de la relación, que se llamará *f*, entre voltaje de salida y el de alimentación en estos dos instantes:

$$\frac{V_{LUZ}}{V_{DD}} = f_{LUZ} = \frac{R}{130\times10^3 + R} \qquad (5.6)$$

$$\frac{V_{LUZ}}{V_{DD}} = f_{OSC} = \frac{R}{10\times10^6 + R} \qquad (5.7)$$

Al expresar las ecuaciones de esta forma se puede calcular el valor de R (resistencia inferior en el divisor) independientemente del valor de V_{DD}. Dicho valor debe permitir obtener el mayor rango de voltaje posible. Esta diferencia queda determinada por:

$$\Delta f = f_{LUZ} - f_{OSC} = \frac{R}{130\times10^3 + R} - \frac{R}{10\times10^6 + R} \qquad (5.8)$$

El valor de R en que se presenta la máxima diferencia se puede obtener derivando esta ecuación con respecto a R e igualando a cero:

$$100\cdot(10\times10^6 - 130\times10^3)\frac{130\times10^3\cdot10\times10^6}{(130\times10^3 + R)^2(10\times10^6 + R)^2} = 0 \qquad (5.9)$$

El valor resultante de resolver la ecuación es de R=1.14 MΩ. Como no es un valor comercial se empleará R=1 MΩ con lo que se obtiene un rango de variaciones del 79.4% de la fuente, desde $0.091V_{DD}$ y hasta $0.885V_{DD}$.

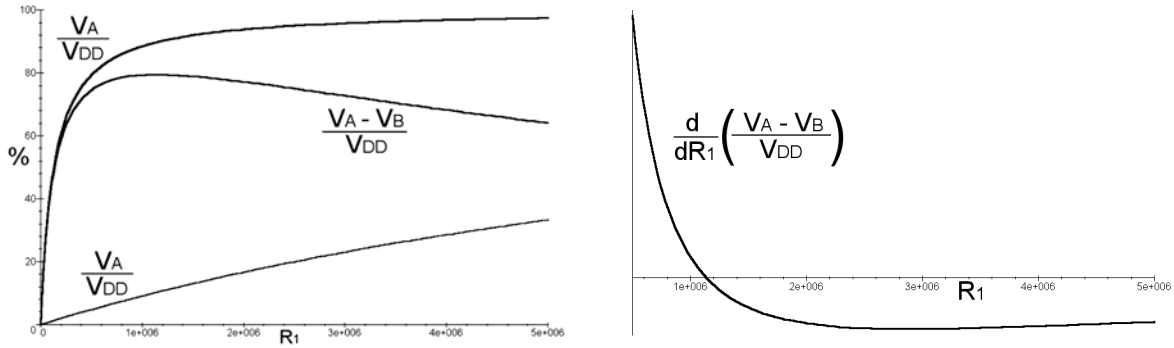


Figura 5.13: Gráfica de las ecuaciones 5.6 a la 5.9.

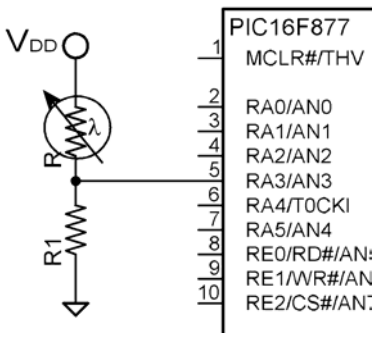


Figura 5.14: Conexión del sensor de intensidad luminosa con el PIC16F877.

La conexión del sensor con el microcontrolador se hará mediante otro canal del convertidor analógico-digital. Al obtenerse un rango amplio de variaciones de voltaje, la resolución del convertidor no es importante. Se empleará la misma que en los sensores de temperatura aunque se tomará la parte alta del resultado.

5.2.3. Voltaje de referencia para el convertidor A/D

La resolución del convertidor depende del voltaje de referencia (ecuación 2.1) y del rango del convertidor (10 bits). El mínimo valor para el voltaje de referencia es de 2 volts, y el máximo de 0.3 volts arriba del voltaje de alimentación [7].

La tabla 5.4 presenta algunas resoluciones en mili volts por unidad binaria para voltajes enteros que cumplen los requisitos mencionados en el párrafo anterior. La proporción de grados Celsius por unidad binaria representa la constante de proporción lineal entre la lectura del convertidor A/D y la temperatura medida por el sensor LM35.

V_{ref} [V]	Resolución (mV/bit)	Proporción (ºC/bit)
2.0	1.953	0.195
3.0	2.930	0.293
4.0	3.906	0.391
5.0	4.883	0.488

Tabla 5.4: Resolución del convertidor A/D.

El set de instrucciones del PIC16F877 no incluye la multiplicación, necesaria para obtener el valor real de una lectura del convertidor. Si se implementara consumiría varios ciclos de operación, además de que se deben representar fracciones decimales mediante números binarios.

Para facilitar las operaciones empleadas para interpretar una lectura del convertidor se puede emplear una resolución entera con lo que el voltaje de referencia toma un valor no entero. Si a pesar de esto se tomara un voltaje de referencia entero se cometería un pequeño error.

Resolución (mV/bit)	Vref [V]	Vref entero [V]	Error %
2	2.05	2	2.34
3	3.07	3	2.34
4	4.10	4	2.34
5	5.12	5	2.34

Tabla 5.5: Resoluciones enteras del convertidor A/D.

Si se emplea una resolución del convertidor que sea una fracción par de la resolución de la señal de entrada, solo se tendrá que dividir el resultado de la conversión entre un múltiplo de 2 mediante corrimientos a la derecha.

Se seleccionó una resolución de 5mV/bit por lo que el voltaje de referencia será de 5.12 volts y se implementó mediante la fuente de alimentación para no utilizar dos voltajes distintos además de reducir los elementos externos. La resolución seleccionada no afecta al sensor de intensidad luminosa, ya que el valor que entrega es proporcional al voltaje de la fuente de alimentación

La resolución del convertidor proporcionará al dispositivo de una mínima lectura de medio grado Celsius. Como se mostrarán únicamente valores enteros de temperatura, se deberá multiplicar por dos el resultado del convertidor mediante un corrimiento a la derecha. El bit menos significativo, que representa medio grado, servirá para redondear el resultado.

5.3. Fuente de alimentación

Una batería se define "...como un dispositivo que convierte la energía química contenida en los materiales activos, en energía eléctrica por medio de reacciones electroquímicas de oxidación y reducción" [8]. A diferencia de las fuentes de alimentación que transforman el voltaje de corriente alterna en corriente directa, las baterías no deben rectificarse ni filtrarse.

Las características básicas de una batería son la tensión entre sus electrodos y la capacidad, que es la cantidad total de electricidad que produce en la reacción química medida en coulombs o en amperes por hora *Ah* [9]. El potencial eléctrico entre sus terminales disminuye a medida que se agota la energía eléctrica producida en su interior. Este fenómeno se observa en la gráfica de la figura 5.15.

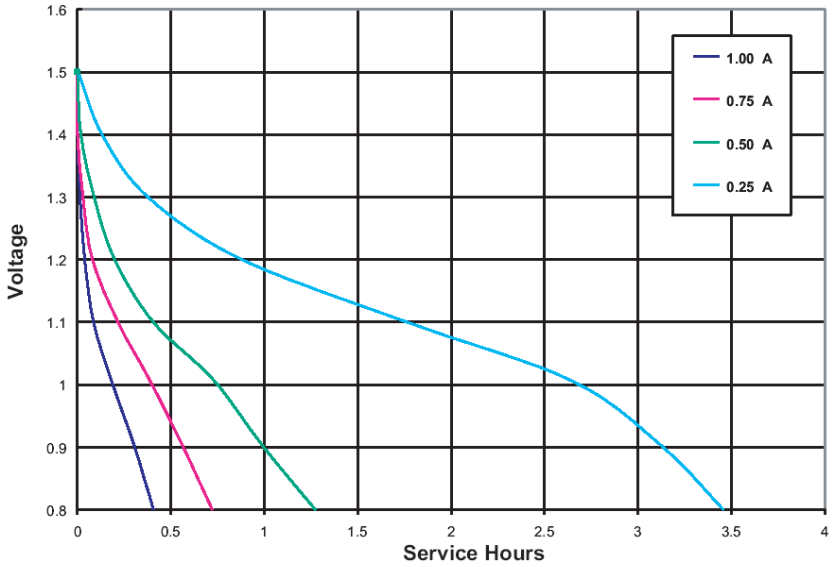


Figura 5.15: Vida útil en horas de servicio de una batería alcalina de 1.5 volts tipo AAA[1].

Mediante tres baterías AAA se pueden obtener 4.5 volts pero su vida útil es corta. El voltaje disminuye significativamente en un lapso corto de tiempo, por lo que el sistema no trabajaría de forma correcta.

El sistema ocupa una batería de 9 volts regulada para obtener el voltaje de operación con que operan sus componentes. Esto permite ampliar el tiempo de servicio debido a que este tipo de baterías almacena mayor energía y a que al ser el voltaje de la batería mayor que el del sistema, tardará más tiempo en disminuir su voltaje a un valor que ya no le permita operar.

[1] Obtenida del sitio de Internet de *Duracell Batteries* en: www.duracell.com.

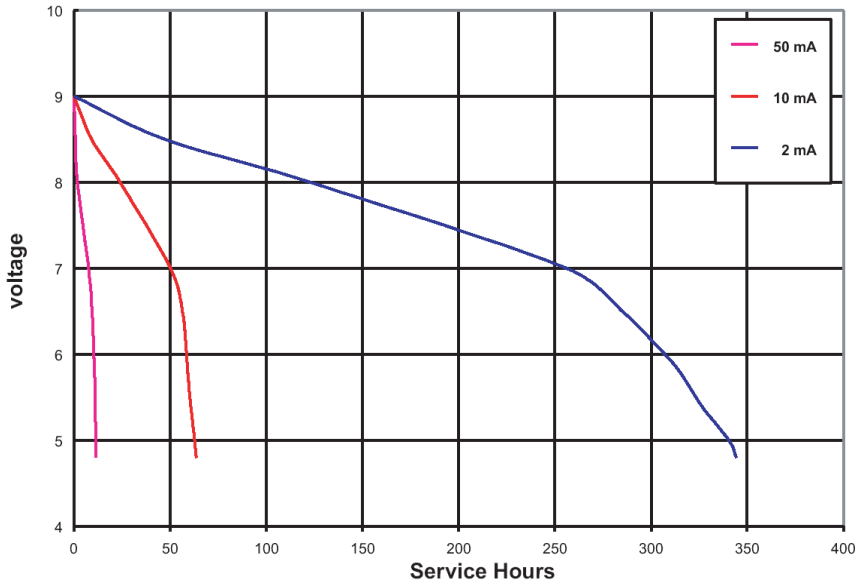


Figura 5.16: Vida útil en horas de servicio de una batería alcalina de 9 volts[1].

En la gráfica anterior se puede observar que si el voltaje del sistema fuera de 5 volts (empleando un regulador[2]) y su consumo fuera de 10 mA, la duración de la vida útil sería de aproximadamente 40 horas.

Otra consideración tomada al definir el voltaje de alimentación del sistema fue el voltaje de referencia para el convertidor A/D. En la tabla 5.5 se observa que al emplear un voltaje de valor entero se produce un error de 2.34% en las mediciones. A éste se agrega el error asociado a la regulación de voltaje de un circuito regulador con valor fijo.

Se seleccionó el regulador de voltaje LM317 al ser un regulador variable en el rango requerido, mismo que se ajusta mediante un par de resistencias. Sus principales características son las siguientes [10]:

- Tolerancia del 1% en el voltaje de salida.
- Regulación de línea del 0.01%/V.
- Regulación de carga del 0.1%.
- Corriente máxima de 1.5A en encapsulado TO-220.
- El rendimiento ajustable abajo a 1.2V
- Rechazo al voltaje de rizo de 80 dB.
- Protección contra cortocircuito en la salida.

[1] Obtenida del sitio de Internet de *Duracell Batteries* en: www.duracell.com.

[2] Un regulador de voltaje común de la familia LM78XX requiere tener en su entrada dos y medio volts más que en su salida para mantener la regulación. Para un regulador LM7805 de 5 volts sería de 7.5 volts.

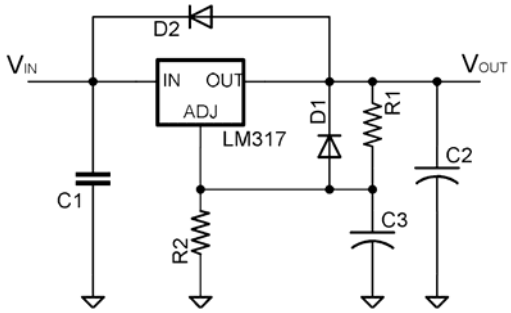


Figura 5.17: Diagrama de conexión propuesto por el fabricante.

El voltaje diferencial entre entrada y salida es de máximo cuarenta volts. Para mantener la regulación debe tener una diferencia mínima de 3 volts. Entre las terminales de salida (OUT) y ajuste (ADJ) existe un valor fijo de voltaje (voltaje de referencia), de manera que la corriente por R_1 será constante y se sumará a la corriente de la terminal de ajuste (I_{ADJ}) para provocar una diferencia de potencial en la resistencia R_2. El voltaje de referencia puede variar entre 1.2 y 1.3 pero su valor típico es de 1.25 volts. La corriente de ajuste tiene un valor típico de 50 μA y puede llegar hasta 100μA.

La ecuación que determina el voltaje a la salida es la siguiente [11]:

$$V_{SAL} = V_{REF}\left(1 + \frac{R_2}{R_1}\right) + I_{ADJ} R_2 \qquad (5.10)$$

El capacitor C_1 únicamente es necesario cuando se usa una fuente rectificada a partir de un voltaje alterno de línea y el regulador se encuentra a más de 15 centímetros del capacitor de filtro. El capacitor C_3 también se ocupa en estos casos y con el mismo propósito. El capacitor C_2 mejora la respuesta transitoria, y aunque el circuito es estable, el fabricante aconseja emplear un capacitor de por lo menos 1μF.

Los diodos se emplean para evitar que los capacitores se descarguen hacia el regulador cuando se desconecta el voltaje en la entrada. A diferencia de otros reguladores de tres terminales, este circuito tiene una protección contra una corriente de hasta 15 A que provenga del capacitor de salida, por lo que pueden omitirse los diodos.

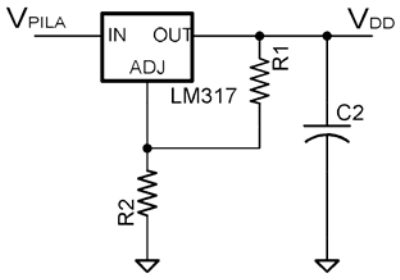


Figura 5.18: Circuito para la fuente del dispositivo.

El voltaje de alimentación debe ser de 5.12 a fin de cumplir con las necesidades de resolución del convertidor y evitar el uso de otros componentes para ajustar su voltaje de referencia externamente. La tabla 5.6 muestra diferentes parejas de resistencia de precisión que se pueden emplear obteniendo errores de menos del 1%.

R_2 [Ω]	R_1 [Ω]	V_O [V]	mV/bit	Error %
3400	1150	5.116	4.9958	0.085
3650	1240	5.112	4.9921	0.158
3480	1180	5.110	4.9907	0.187
3900	1330	5.110	4.9906	0.187
3010	1020	5.089	4.9699	0.601

Tabla 5.6: Valores de resistencias para obtener un voltaje cercano a 5.12 V.

5.4. Dispositivos de salida

5.4.1. Pantalla LCD

El medio de despliegue de información se implementa mediante una pantalla de cristal líquido (LCD) de propósito general TM162 con dos líneas de 16 caracteres. Estas pantallas de propósito general son comúnmente empleadas debido a que el costo de un diseño especial es muy elevado. El medio de conexión y las instrucciones que ejecutan son comunes para una gran mayoría de modelos.

Los datos pueden ser enviados mediante cuatro u ocho líneas, además de las tres líneas para las señales de control. Las instrucciones son las mismas para todos las pantallas de este tipo, por lo que fácilmente se puede cambiar de modelo e incluso de fabricante. En algunos casos tienen iluminación de fondo activada mediante dos líneas.

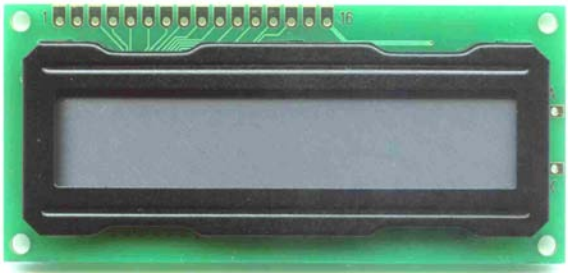


Figura 5.19: Aspecto de la pantalla LCD de 16 caracteres y dos líneas.

Cada espacio disponible para escribir un carácter está asociado con una localidad de su memoria RAM de datos que almacena hasta 40 caracteres por línea aunque solo se muestren algunos. Tiene 64 localidades de RAM destinadas a caracteres definidos por el usuario. Cada uno de éstos ocupa 8 localidades, por lo que solo se pueden usar 8 caracteres de usuario

Las instrucciones que puede ejecutar son las siguientes [12]:

- Borrar pantalla (CLEAR DISPLAY).
- Cursor a posición inicial (HOME).
- Establecer modo de funcionamiento (ENTRY MODE SET).
- Control ON/OFF (DISPLAY ON/OFF).
- Desplazamiento del cursor y de la pantalla (CURSOR DISPLAY SHIFT).
- Modo de transferencia de la información (FUNCTION SET).
- Acceso a posiciones de la RAM de caracteres de usuario (SET CGRAM ADDRESS).
- Acceso a posiciones de la RAM de datos (SET DDRAM ADDRESS).
- Enviar datos a las dos RAM (WRITE DATA TO CG/DD RAM).
- Leer datos de las dos RAM (READ DATA FROM CG/DD RAM).

La presentación de datos en el modo de operación normal es la siguiente:

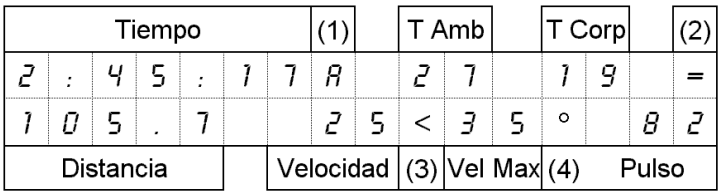


Figura 5.20: Presentación de los datos en operación normal.

Se muestra la lectura de tiempo, distancia, velocidad, temperatura ambiente y corporal además de la frecuencia cardiaca. Mediante indicadores de un solo carácter se indica lo siguiente:

(1) Indica la activación de los límites de: tiempo (‘ **t** ’), distancia (‘ **d** ‘), ambos (‘ **a** ‘), o ninguno (‘ ‘).

(2) Indica el nivel de iluminación: (‘ **=** ‘) para nivel excelente, (‘ **-** ‘) para nivel bueno, (‘ _ ‘) para nivel regular y (‘ ‘) para nivel malo.

(3) Indica la activación (‘ **<** ‘) o desactivación (‘ ‘) del límite de velocidad. También indica cuando la velocidad máxima ha sido superada (‘ **>** ‘).

(4) Indica cuando se presenta un pulso cardiaco. Para indicar que la frecuencia cardiaca máxima ha sido superada, se presenta su valor intermitentemente.

La presentación de datos en el modo de configuración es la siguiente:

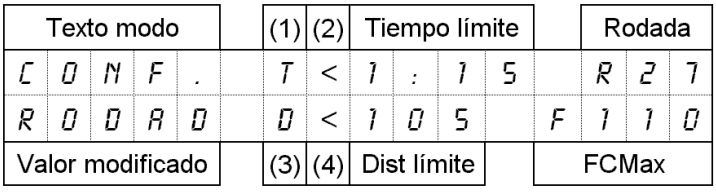


Figura 5.21: Presentación de los datos en modo de configuración.

Se muestra el límite de tiempo y distancia, la frecuencia cardiaca máxima, y la rodada de la bicicleta. Mediante cinco caracteres se informa el parámetro que está siendo modificado. Mediante indicadores de un solo carácter se indica lo siguiente:

(1) Indica que el valor a su derecha corresponde al límite de tiempo.

(2) Indica límite de tiempo activo (‘ **<** ’) o inactivo (‘ ‘).

(3) Indica que el valor a su derecha corresponde al límite de distancia.

(4) Indica límite de distancia activo (‘ **<** ’) o inactivo (‘ ‘).

En el diseño se consideró que el número de terminales permitía conectar el LCD mediante ocho líneas de datos y tres de control, lo que reduce las líneas de código necesarias para comunicarse con la pantalla.

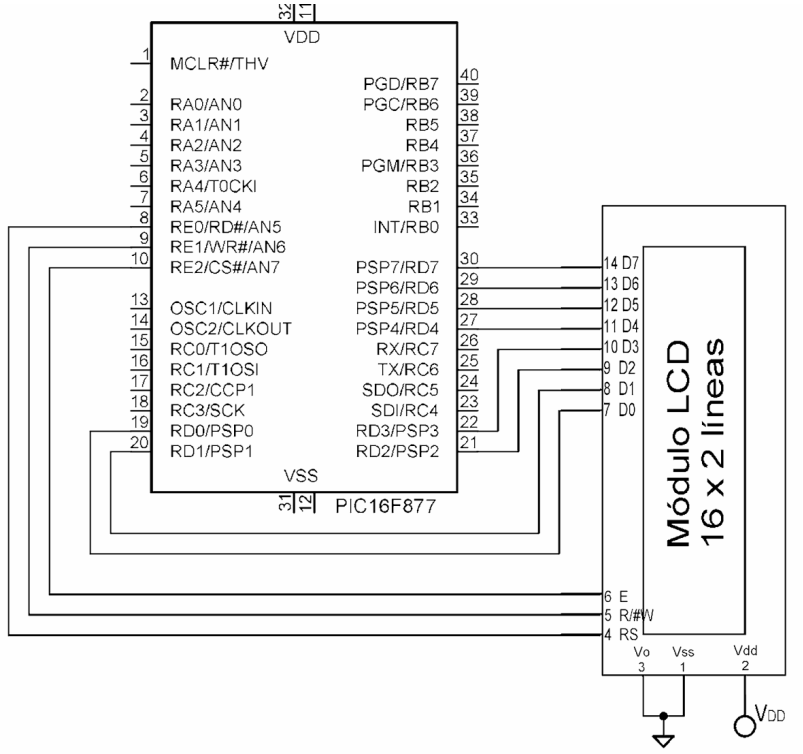


Figura 5.22: Conexión de la pantalla LCD con el PIC16F877 mediante los puertos B y E para datos y control respectivamente.

5.4.2. Alarma auditiva

Para implementar el generador de sonido se emplea un *zumbador* piezoeléctrico o *buzzer*, que genera un sonido de tono constante. Opera con un voltaje de 3 a 28 V de corriente directa y presenta un consumo máximo de corriente de 8mA funcionando a 12V.

Las terminales de salida del PIC16F877 aplican en nivel alto un voltaje de 5.12 volts (de acuerdo con la fuente de alimentación del dispositivo diseñado) y pueden proporcionar hasta 25 mA de corriente. Estos valores son compatibles con los rangos de operación del zumbador, que es habilitado directamente por una terminal del microcontrolador y con una corriente pequeña.

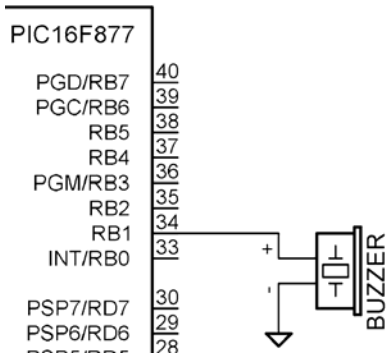


Figura 5.23: Alarma auditiva del dispositivo.

5.4.3. Indicador luminoso

Para este medio de salida se buscó obtener la mayor iluminación con el menor consumo de corriente. Se alimenta mediante un par de baterías tipo AA de 1.5 volts por dos razones:

1. Se ubicará lejos del resto de los componentes.
2. Consumirá más energía que todos los demás componentes.

El uso de focos hubiera permitido un buen nivel de iluminación, pero el consumo de corriente hubiera sido muy alto por lo que las baterías se descargarían rápidamente. Su propósito es el de resaltar la presencia de la bicicleta y no el de iluminar, además de que no debe deslumbrar para evitar accidentes.

Un diodo emisor de luz (LED) se compone de una unión semiconductora *p-n*. Se fabrican con fosfito de galio y generan luz visible al polarizarse directamente mediante un proceso llamado *electroluminiscencia* [13]. Su voltaje de encendido típico es de 2.2 volts y el máximo es de 3 volts. Para el caso de los LED *ultra brillantes* que se emplean, presentan una intensidad luminosa de 0.8 a 1.2 mili candelas, una corriente de 10 mA y un ángulo de iluminación de 30 grados con respecto al plano horizontal.

Para aumentar la capacidad de iluminación se agruparon tres LED en paralelo conectados a las dos pilas que se conectaron en serie para obtener 3 volts. Para activar el indicador se emplea un *optoacoplador*, que es un conjunto de un LED y un fototransistor en un encapsulado cerrado. Al encender el emisor el receptor se pone el transistor en saturación con lo que se permite el paso de corriente entre sus terminales de colector y emisor.

Se emplea el optoacoplador 4N32 con salida *darlington* de dos transistores que soporta una corriente máxima de 150 mA. Para ponerlo a funcionar deben aplicarse 1.2 volts a las terminales del emisor. La corriente consumida por éste es de 10 mA, por lo que el voltaje en la resistencia que conecta la terminal del microcontrolador con el optoacoplador debe de ser de 3.92 volts (la diferencia entre el voltaje de alimentación y el aplicado al diodo). El valor de la resistencia se obtiene mediante la ley de Ohm:

$$R = \frac{V_R}{I_R} = \frac{V_{DD} - V_{LED}}{I_{LED}} = \frac{5.12 - 1.2}{10 \times 10^{-3}} = 392\Omega \qquad (5.11)$$

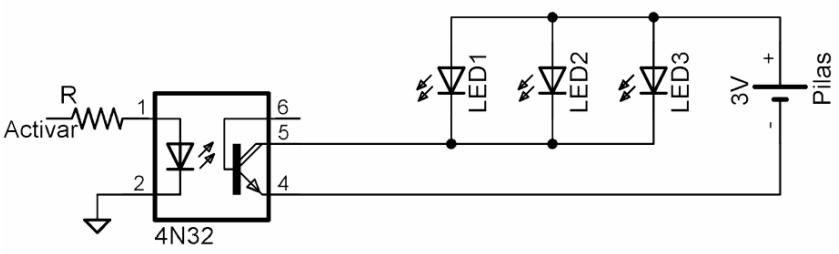


Figura 5.24: Circuito del indicador luminoso. La terminal *Activar* se conecta directamente a una salida del PIC16F877.

5.5. Sensor de giro

5.5.1. Principio de Funcionamiento

Como se menciona en el apartado 2.3.1, la distancia y la velocidad se calcula por medio del perímetro de la rueda y el número de vueltas que dé ésta. Para detectar cada giro se debe colocar un elemento en la rueda, de preferencia la delantera, que provoque algún efecto en otro elemento montado en la horquilla. El medio de interacción puede ser óptico, electro-mecánico, magnético o electro-magnético.

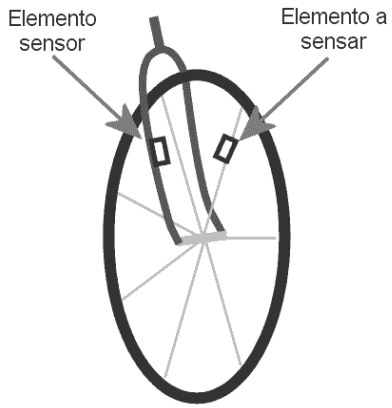


Figura 5.25: Sensor de Giro.

Medios electro-mecánicos

El elemento a detectar debe producir mediante alguna acción mecánica una señal eléctrica de forma similar a los interruptores. Estos medios requieren de un contacto físico entre los dos elementos que generará un desgaste de las piezas. Además, la salida presentará períodos transitorios similares al de los interruptores.

Medios ópticos

Para que el sensor perciba la presencia del elemento a detectar, este último deberá alterar de alguna forma el haz de luz infrarroja que se envía de un dispositivo emisor a un dispositivo receptor. Un sensor de ranura contiene un emisor infrarrojo (fotoled) y un receptor fotosensible (fotodiodo o fototransistor) de manera que el haz de luz pueda interrumpirse al colocar un objeto opaco en la ranura.

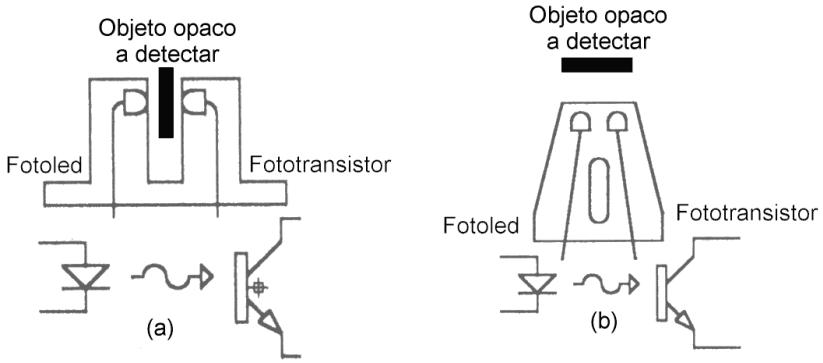


Figura 5.26: Sensor óptico de: (a) ranura y (b) reflectivo.

Estos métodos requieren de una ubicación precisa de sus componentes, pues el haz de luz infrarroja prácticamente describe una línea recta. Además pueden influir otras fuentes luminosas sobre el funcionamiento del receptor y el barro acumulado en las ruedas podría tapar al receptor y/o al transmisor.

Medios magnéticos y electromagnéticos.

En éstos el elemento a detectar es un imán que producirá un efecto en el sensor que depende de la naturaleza de éste. Si se emplea una bobina, el imán produce por inducción electromagnética un pulso de corriente en ella. Si se emplea un sensor de efecto hall, el imán produce un pulso de voltaje.

En el primer caso, la bobina debe tener la mayor cantidad posible de espiras para que la intensidad de la corriente facilite la detección del giro. En el segundo, el elemento sensor tiene un menor tamaño y está incluido dentro de un circuito integrado llamado *switch de efecto hall*. Éste incluye además del sensor de efecto hall, un amplificador y un comparador que permiten tener a la salida un nivel lógico de voltaje que depende de la presencia o la ausencia del imán.

5.5.2. Definición del circuito

La opción de *switch hall* e imán, para el sensor y el elemento a detectar respectivamente, fue seleccionada por no requerir que la posición relativa entre los elementos sea precisa, además de que el tamaño del sensor de giro es más reducido.

El modelo empleado es el LB9051 de Sanyo Electric, que opera con una alimentación de entre 3.6 y 16 volts consumiendo un máximo de 8 mA. Es sensible a campos magnéticos de baja intensidad y su salida es compatible con la lógica de las terminales del PIC16F877. Para que su salida pase de un voltaje alto a uno bajo debe presentarse un flujo en la dirección mostrada en la figura 5.27, y en sentido contrario para que pase de bajo a alto.

Al girar la rueda y acercarse al polo S del imán al sensor, se tiene a la salida un *1* lógico. Al alejarse el imán la dirección del flujo que pasa por el sensor se invierte porque es el polo N el que está cerca de él y se tiene un valor binario 0 (figura 5.28). Como el giro de la rueda debe atenderse inmediatamente, se conectará a la terminal INT para que se produzca una interrupción al presentarse un flanco de subida.

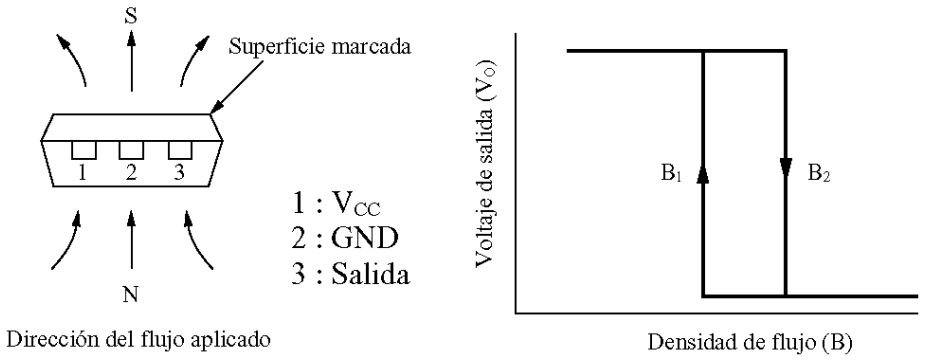


Figura 5.27: Respuesta del sensor LB9051.

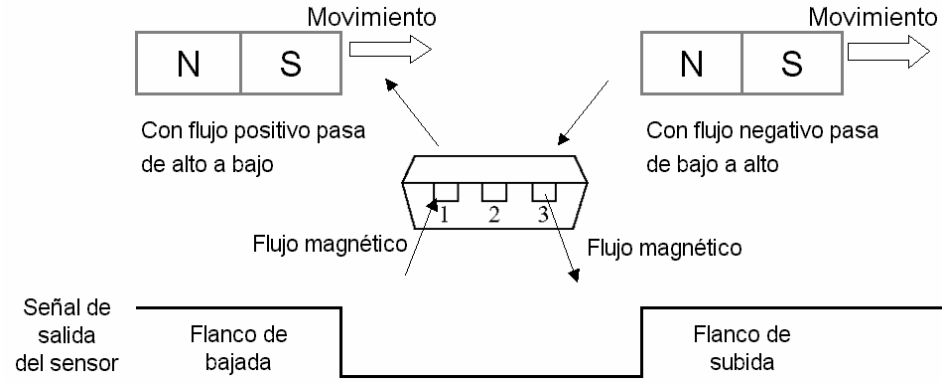


Figura 5.28: Funcionamiento del sensor de giro.

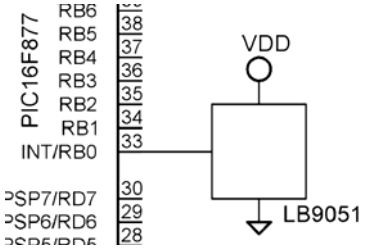


Figura 5.29: Conexión del sensor de giro con el PIC16F877.

5.6. Sensor de pulso

Para este sensor se tenían distintas opciones, tanto para ser desarrolladas como dispositivos comerciales. Los sensores que emplean electrodos implican un contacto eléctrico entre el ciclista y el dispositivo, además de que su costo es elevado. Es por esto que se decidió emplear un sensor óptico que mantiene aislado al ciclista del dispositivo. Diversos dispositivos comerciales son empleados con este propósito pero se desarrolló uno que cumpliera con los requerimientos específicos del dispositivo diseñado.

5.6.1. Principio de funcionamiento

Al producirse un latido del corazón se presentan un pulso de presión y otro eléctrico (apartado 1.4.1). Detectando cualquiera de éstos se puede saber en que instante ocurre un pulso cardiaco. Una medición de la señal eléctrica para detectar el complejo QRS de la figura 1.12 requiere de un contacto directo entre el ciclista y el circuito electrónico. Además de existir un riesgo de choque eléctrico en las terminales, que se encuentran muy cerca del corazón, el costo se eleva por la necesidad de electrodos especiales y amplificadores de instrumentación (apartado 1.6.2).

Cuando se presenta el pulso de presión sanguínea, las arterias aumentan su grosor y la sangre contiene más oxígeno (figura 5.30), Mediante un oxímetro de oreja se puede obtener una medida eléctrica correspondiente a la absorción de luz en la sangre que transportan las arterias capilares del lóbulo de la oreja, y que será máxima al presentarse el pico de presión pues hay mayor obstrucción del haz de luz.

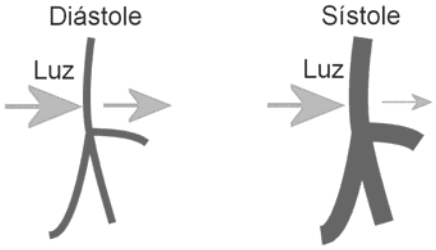


Figura 5.30: Efectos de la luz en las arterias. Durante la sístole el pulso de presión hace que las arterias se ensanchen, por lo que pasa menos luz.

Al detectar un cambio entre el máximo y mínimo nivel de absorción de luz se sabe cuándo ocurre un latido del corazón. Como en esta aplicación no se requiere conocer el porcentaje de oxigenación de la hemoglobina, se emplea un emisor de luz infrarroja para trabajar cerca de la longitud de onda en que la sangre absorbe la misma cantidad de luz sin importar su nivel de oxigenación (apartado 1.4.1). El receptor es un fototransistor que presenta una buena respuesta en la zona infrarroja y tiene protección contra luz visible (figura 3.15).

5.6.2. Definición del circuito

La figura 5.31 presenta el circuito que conforma el oxímetro y que se monta en una pinza para que quede sujeto al lóbulo de la oreja. La resistencia del receptor tiene un valor grande para aumentar la sensibilidad con corrientes pequeñas, debido a que:

$$V_0 = R_2 \cdot I_E \approx R_2 \cdot I_C = R_2 \cdot \beta \cdot I_C \qquad (5.12)$$

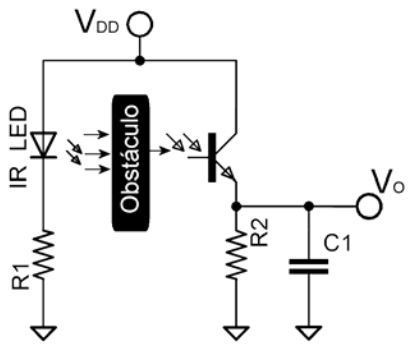


Figura 5.31: Etapa de entrada.

Un valor de 10 MΩ para R_2 permite detectar corrientes de colector de hasta 0.512 μA aproximadamente con lo que se alcanzaría un voltaje de salida prácticamente igual al de la fuente (la diferencia es el voltaje V_{CE} del transistor). La resistencia del fotoled permite que encienda con sus valores de voltaje y corriente nominales:

$$R_1 = \frac{V_R}{I_R} = \frac{V_{DD} - V_{IRLED}}{I_{IRLED}} = \frac{5.12 - 1.3}{10 \times 10^{-3}} = 382\Omega \qquad (5.13)$$

El capacitor C_1 de 0.01μF ayuda a eliminar el ruido de la señal de salida del sensor. Para evitar interferencia de otras señales de frecuencias medias y altas se emplea un filtro paso bajas de primer orden, acoplado a la salida del sensor mediante un seguidor de voltaje [14]. La frecuencia cardiaca siempre será menor a 220 pulsos por minuto, equivalentes a 220/60=3.67 pulsos por segundo o hertz. Empleando un capacitor de 220 nF, el valor de la resistencia se obtiene mediante la ecuación para la frecuencia de corte de un filtro paso bajas de primer orden [15]:

$$f_C = \frac{1}{2\pi \cdot R_3 C_2} \rightarrow R_3 = \frac{1}{2\pi \cdot f_C C_2} \qquad (5.14)$$

Para una frecuencia de corte de 5 Hz el valor de la resistencia es de 144 kΩ. Durante las pruebas del sensor se observó que con esta frecuencia de corte se atenúa demasiado la señal en la banda de paso, por lo que se recorrió la frecuencia de corte hasta 33 Hz empleando una resistencia R_3 de 22 kΩ, con lo que se conserva la amplitud de la señal.

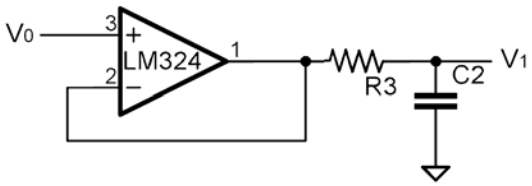


Figura 5.32: Etapa de filtrado.

Mediante el osciloscopio se observó que la amplitud de la componente alterna de la señal del sensor es de aproximadamente 1.8 mV, con un voltaje en DC de 1.6V. El voltaje de salida ya filtrado (V_1) debe amplificarse pero únicamente en su componente de AC. Se probó un

amplificador no inversor para AC que únicamente amplifica el voltaje alterno, y la salida queda sobre un voltaje de directa con un valor de la mitad del de alimentación. Su respuesta era muy inestable, por lo que se optó por emplear un amplificador no inversor de voltaje DC.

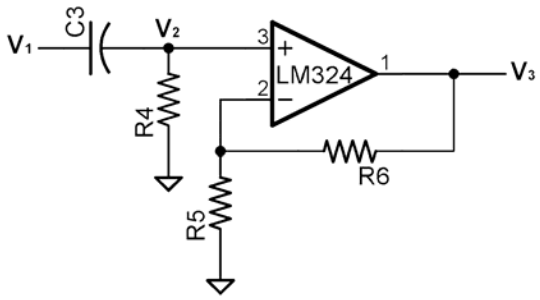


Figura 5.33: Etapa de amplificación.

Como el voltaje fijo de V_1 es aproximadamente mil veces mayor al voltaje variable que representa la señal del pulso, se deja pasar solo este último al amplificador mediante el capacitor C_3 de 1.5 μF. Al conectar a la entrada del operacional el voltaje V_2 resultante se observó que la señal se perdía. Se colocó la resistencia R_4 para colocar a V_2 sobre un voltaje en DC de un valor mucho menor al que tiene V_1. Experimentalmente se observó que un valor de 100 kΩ para esta resistencia genera un voltaje fijo de aproximadamente 2 mV sobre el que se ubica la señal del pulso. Las resistencias R_5=1 kΩ y R_6= 289 kΩ determinan una ganancia de 290 para el voltaje de salida V_3 [16]. Debido a que la amplitud de esta señal puede disminuir en una persona con baja presión sanguínea o en reposo, se colocó una segunda etapa de amplificación igual a la mostrada en la figura 5.33 pero con una ganancia de solo 4.7 mediante una resistencia R_6'= 4.7 kΩ.

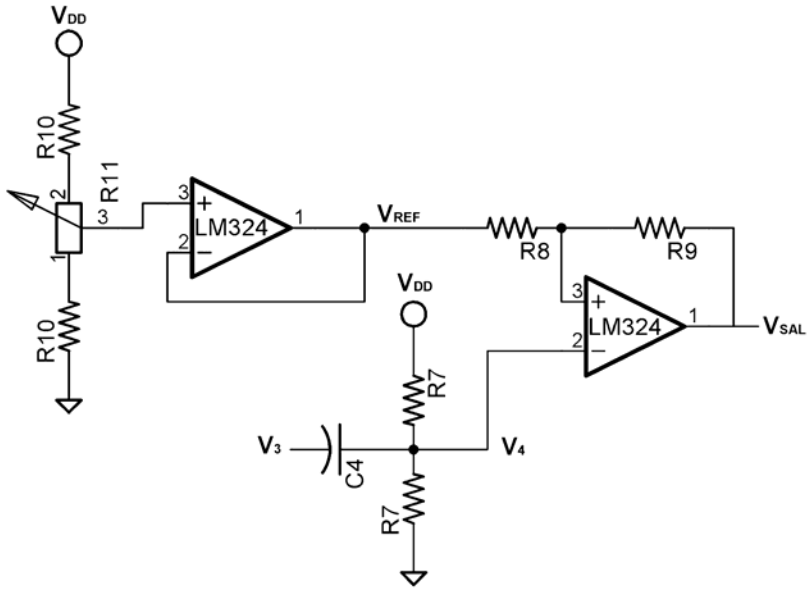


Figura 5.34: Etapa de comparación. La salida V_{SAL} se conecta a la terminal T0CKI del PIC16F877.

Para transformar la señal analógica en una serie de pulsos digitales se emplea un comparador. Con esto se cambia la forma de onda pero se conserva la frecuencia de la señal. Como la amplitud de la señal del pulso disminuye al presentarse éste, la configuración que se utiliza es la de comparador no inversor, como se muestra en la figura 5.34. Se emplea retroalimentación positiva agregando con esto un voltaje de histéresis que evita que el ruido en la señal genere pulsos no válidos [17].

El voltaje V_3 se acopla mediante un capacitor a un voltaje de DC de la mitad de la fuente, con el objeto de que la señal V_4 siempre cruce por este valor. Las resistencias R_7 tienen un valor de 100 kΩ para que la corriente que pase por ellas sea pequeña. Para obtener el valor de las resistencias R_8 y R_9 se debe analizar la retroalimentación positiva.

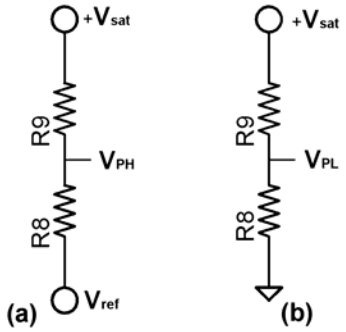


Figura 5.35: Circuitos equivalentes para la retroalimentación positiva.

En la figura 5.35a se presenta el circuito equivalente para cuando el voltaje de salida está en nivel alto ($+V_{sat}$), y en el que el voltaje en la terminal no inversora del comparador (V_P) está determinado por la ecuación 5.15. La figura 5.35b es el circuito equivalente cuando la salida está en nivel bajo ($-V_{sat} \approx 0$ V) y la ecuación 5.16 determina el valor de V_P bajo estas circunstancias.

$$V_{PH} = V_{REF} + \frac{+V_{SAT} - V_{REF}}{R_8 + R_9} R_8 \qquad (5.15)$$

$$V_{PL} = V_{REF} \frac{R_9}{R_8 + R_9} \qquad (5.16)$$

Para facilitar los cálculos se considera que:

$$R_9 = k \cdot R_8 \qquad (5.17)$$

Con esto se pueden modificar las ecuaciones 4.15 y 4.16 para que queden en función la constante de proporcionalidad *k*. Como el voltaje de histéresis es la diferencia entre estos valores de V_P, se restan las ecuaciones 4.18 y 4.19 para dejar a *k* en función del voltaje de histéresis y de saturación positiva.

$$V_{PH} = +V_{SAT} \cdot \left(\frac{1}{k+1}\right) + V_{REF} \cdot \left(\frac{k}{k+1}\right) \qquad (5.18)$$

$$V_{PL} = V_{REF} \cdot \left(\frac{k}{k+1} \right) \qquad (5.19)$$

$$k = \frac{+V_{SAT}}{V_{HIST}} - 1 \qquad (5.20)$$

El valor común del voltaje de saturación positiva es de un volt menor al de alimentación [18], en este caso 4.12V, y se establece un voltaje de histéresis de 180mV. Sustituyendo estos valores en la ecuación anterior se obtiene un valor para *k* de 21.89 ≈ 22, con lo que si R_8= 1 kΩ entonces R_9= 22 kΩ.

El voltaje de referencia del comparador depende de la componente de directa que tenga la señal V_4. Para ajustar su valor se emplea un potenciómetro de 20 kΩ que junto con dos resistencias de 50 kΩ forman un divisor de voltaje que se acopla con el comparador mediante un seguidor de voltaje.

El circuito integrado LM324 que se emplea puede polarizarse con una sola fuente e incluye cuatro amplificadores operacionales en un mismo encapsulado DIP de 14 terminales.

5.7. Referencias

1 *PIC16F87X Data Sheet. 28/40-Pin 8-Bit CMOS FLASH Microcontrollers*, Microchip Technology Inc., USA: 2001, p. 124.

2 *Ibíd*., p. 180.

3 *Ibíd*. p. 177.

4 *Ibíd*. p. 178.

5 *National Analog and Interface Productos Databook* (Libro y CD-ROM), National Semiconductor, USA: 2001, pp. 12-3, 12-4.

6 *Ibíd*. p. 12-10.

7 *PIC16F87X Data Sheet*, *Op. Cit.*, p. 174.

8 José Fuella García, *Acumuladores electroquímicos. Fundamentos, nuevos desarrollos y aplicaciones*, "Nociones básicas. Unidades específicas empleadas", España: Ed. McGraw Hill, 1994, p. 2.

9 *Ibíd.*, pp. 7-8.

10 *National Analog and Interface Productos*, *Op. Cit*., p.16-8.

11 *Ídem*.

12 José Ma. Angulo, Ignacio Angulo, *Microcontroladores PIC. Diseño práctico de aplicaciones*, "Apéndice F: Proyectos con el PIC16F84", 2ª. ed., Madrid: Ed. McGraw Hill, 1999, p. 260-263.

13 Robert Boylestad y Louis Nashelsky, *Electrónica. Teoría de circuitos*, "Zeners y otros dispositivos de dos terminales", trad., Carlos A. Franco, México: Ed. Prentice-Hall Hispanoamericana,1983, p. 106.

14 Luces M. Faulkenberry, *Introducción a los amplificadores operacionales con aplicaciones a CI lineales*, "El Amp-Op básico", México: Ed. Limusa, 1996, pp. 25-27.

15 *Ibíd*., p. 289.

16 *Ibíd*., pp. 27-31.

17 *Ibíd*., pp. 446-455.

18 *Ibíd*., p. 448.

Capítulo VI

Desarrollo del Software

6.1. Metodología de programación

La tabla 6.1 presenta las distintas etapas del proceso de desarrollo de software [1]. En la primera etapa se define el problema y se analiza para determinar las tareas que debe desempeñar el sistema, definir las entradas y salidas, y establecer los procedimientos básicos o algoritmos[1] que deben realizarse para leer las entradas y a partir de ellas establecer las salidas.

El diseño del programa consiste en formular la definición del problema como un programa o algoritmo. Una vez concluida esta etapa se puede codificar éste en un lenguaje de programación, que una vez concluida su depuración debe probarse junto con los componentes del sistema. Cuando ha probado funcionar correctamente se debe documentar su funcionamiento de forma que se facilite su mantenimiento y mejora posterior.

No.	Etapa
1	Definición del problema
2	Diseño del programa
3	Codificación
4	Depuración o verificación
5	Pruebas o validación
6	Documentación
7	Mantenimiento y rediseño

Tabla 6.1: Etapas del proceso de desarrollo de software.

Para facilitar el proceso de desarrollo de software, el programa debe ser fácil de comprender y de modificar. La depuración y prueba del software y del hardware implican un proceso iterativo que requiere de revisar cada elemento tanto de hardware como de software, por lo que se facilita enormemente si cada componente y procedimiento se encuentra correctamente documentado.

Existen diversas técnicas de desarrollo de software que al trabajar en conjunto forman la metodología de programación. Los métodos empleados en este trabajo se describen a continuación [2]. Cada uno aporta características individuales, que al trabajar juntas facilitan el desarrollo del software para el dispositivo bajo diseño.

[1] Un algoritmo es una serie de pasos que deben seguirse para llegar a la solución de una tarea o problema.

6.1.1. Diagramas de flujo

Los diagramas de flujo son representaciones gráficas de un algoritmo, por lo que facilitan su comprensión y son independientes del lenguaje empleado para codificar el programa. Es por esto que no se requieren conocimientos de algún lenguaje de programación para entenderlos. Existen diversas herramientas de cómputo que permiten desarrollar diagramas de flujo, ya sean paquetes dedicados especialmente a ello o procesadores de texto con herramientas de dibujo.

A pesar de sus ventajas, los diagramas de flujo requieren de cierto tiempo para su construcción, por lo que son difíciles de depurar. Solo representan la organización del programa y no contemplan la estructura de los datos y los módulos de entrada y salida. A medida que el programa crece en tamaño, los diagramas pueden hacerse difíciles de construir y entender, por lo que se recomienda emplearlos en conjunto con otras técnicas que dividan el programa en partes pequeñas.

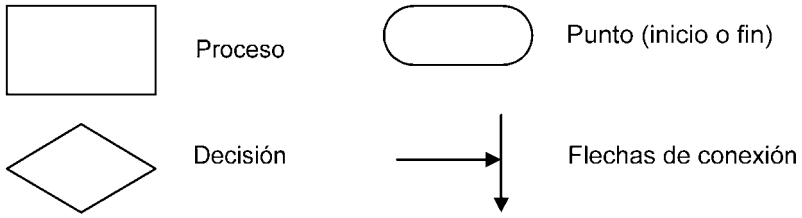


Figura 6.1: Elementos básicos de los diagramas de flujo.

6.1.2. Programación modular

Consiste en dividir el programa en tareas más pequeñas llamadas *módulos*. Éstos pueden ser inclusive otros programas escritos con anterioridad y que funcionan correctamente. Los módulos pequeños son fáciles de escribir y depurar, además de que pueden representarse claramente mediante diagramas de flujo. Los módulos pueden diseñarse para ser empleados en distintas partes del programa, con lo que se optimiza el tamaño de éste.

El principal problema que presenta la programación mediante módulos es determinar su ubicación dentro del programa. Además en algunos casos puede ser muy difícil probarlos y depurarlos por separado, pues en muchos casos requerirán de otros módulos. Para lograr una correcta división del programa en módulos, el problema debe estar definido correctamente.

Programación estructurada

Este método permite mantener separados los módulos empleados en la técnica anterior. Las estructuras empleadas mantienen una secuencia de operaciones clara, con lo que los posibles errores son fáciles de localizar. Aunado a esto, el empleo de estructuras reduce el espacio requerido por el programa al permitir la realización de procesos iterativos que ahorran varias líneas de código repetido.

Las estructuras solo tienen un punto de entrada y uno de salida. Aunque las estructuras son más fáciles de emplear en lenguajes de medio y alto nivel, se pueden construir mediante lenguaje ensamblador aunque requieren de varias líneas. Las estructuras más básicas son las siguientes [3]:

Secuencias

Es una estructura lineal en la que las operaciones se ejecutan una tras otra.

Estructuras condicionales

Ésta permite ejecutar ciertas secuencias dependiendo de ciertas circunstancias. En el caso de la estructura IF-THEN-ELSE se presentan dos distintos casos en que se ejecutan secuencias diferentes cuando la condición es verdadera o falsa. En el caso de la estructura IF-THEN solo existe un posible caso, que es ejecutado o no dependiendo de que la condición sea falsa o verdadera.

Estructuras iterativas

Son secuencias que se ejecutarán varias veces mientras la condición sea verdadera o falsa. En la estructura DO-WHILE la condición se verifica desde un principio, mientras que en la estructura DO-UNTIL se hace al final del ciclo, por lo que las secuencias se ejecutarán por lo menos una vez.

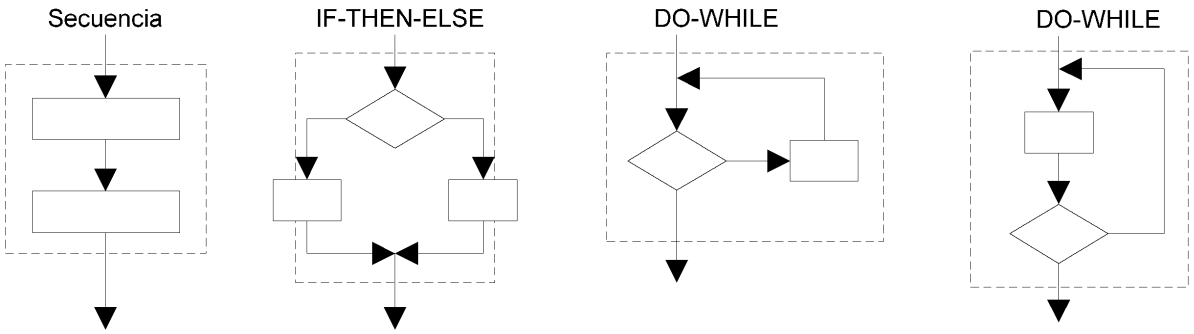


Figura 6.2: Estructuras básicas.

Existen otras estructuras que se conforman de varias estructuras básicas o que son casos especiales de ellas. La estructura de selección CASE se conforma de varias estructuras de selección de forma que se pueden ejecutar distintas secuencias asociadas a diferentes valores de una variable. La estructura FOR es un caso de estructura iterativa que emplea un contador ascendente o descendente para llevar un control del número de veces que se ejecuta una secuencia de operaciones. La condición que se verifica en cada iteración está asociada con el valor del contador.

6.1.3. Diseño descendente (Top – Down)

Esta técnica permite que los módulos o estructuras funcionen juntos y facilita la depuración y prueba del programa completo. En este método se divide el programa en los módulos más generales, los que conformarán el nivel superior del diseño. Posteriormente se expanden éstos al dividirlos en otros módulos hasta representar el programa completo mediante estructuras. Los módulos que aún no han sido escritos se sustituyen con pequeños programas que no realicen ninguna operación, con lo que el programa completo puede ser probado y depurado al mismo tiempo que se avanza en su diseño y codificación.

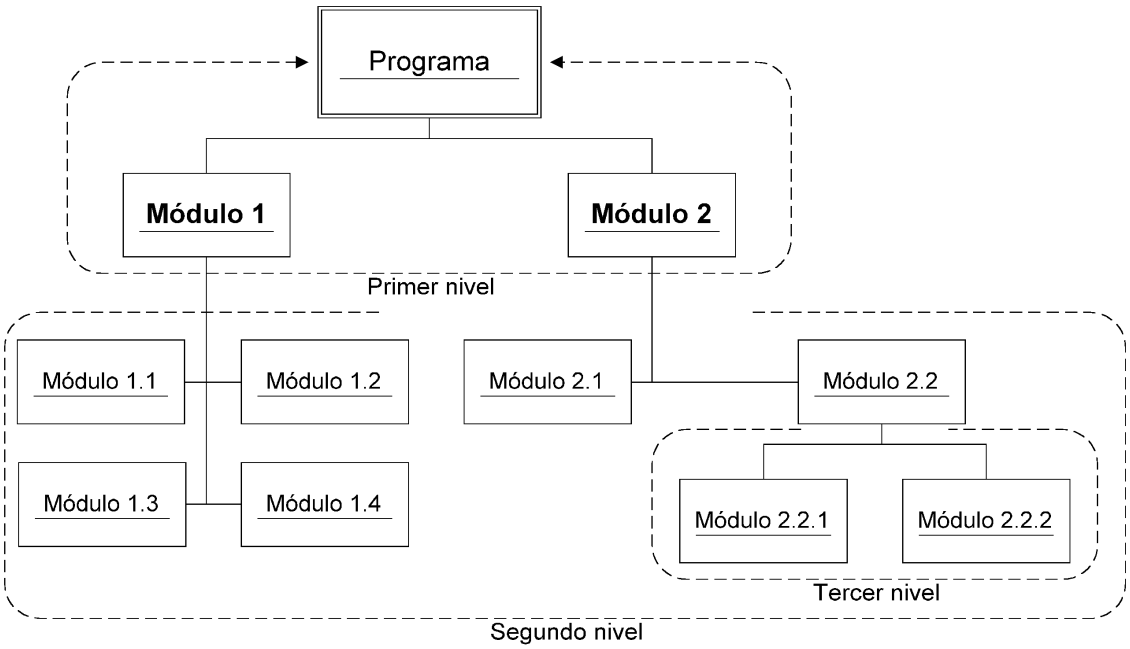


Figura 6.3: Niveles de un diseño descendente.

6.2. Métodos de medición

6.2.1. Medición del tiempo

Se emplean cuatro bytes para las horas, los minutos y los segundos. La interrupción por desborde del Timer 1 sucede cada segundo. Cada vez que esto sucede se ejecutan todos los procedimientos que deben llevarse a cabo en intervalos de uno o varios segundos. Uno de ellos es el incremento del cronómetro. Su funcionamiento se describe mediante el diagrama de flujo de la figura 6.4.

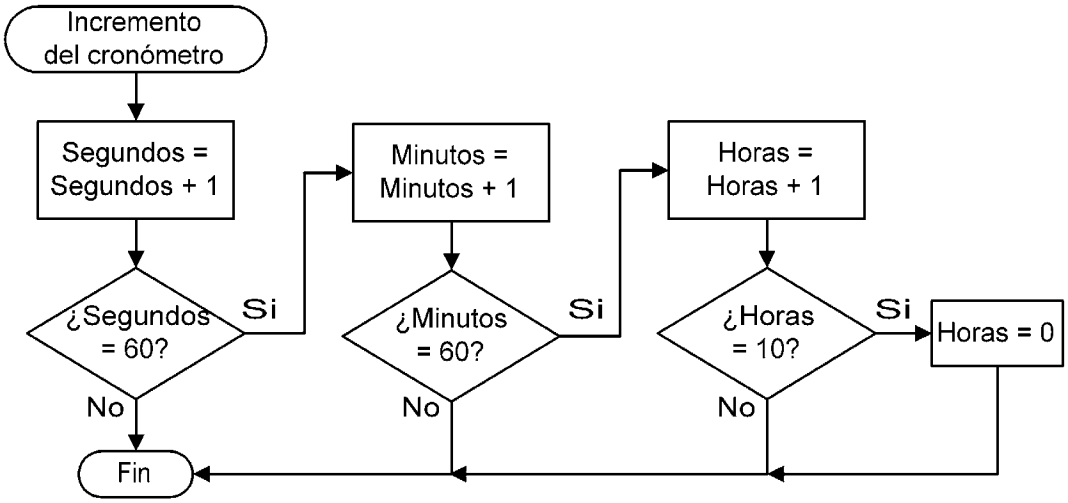


Figura 6.4: Diagrama de flujo del funcionamiento del cronómetro.

Los dos bytes dedicados a almacenar la cuenta de segundos y minutos del cronómetro únicamente almacenan valores comprendidos entre 0 y 59, por lo que al llegar a 60 se produce un acarreo de segundos a minutos o de minutos a horas. El byte para las horas llega hasta 9 y de ahí reinicia la cuenta en 0 sin producir ningún acarreo.

6.2.2. Medición de la distancia

Un incremento en la distancia igual al perímetro de la rueda se da con cada giro de ésta (tabla 1.4). El incremento mínimo es de un metro con 76 centímetros para una bicicleta de rodada 22. Como el valor más pequeño que se mostrará en la pantalla es de cien metros, el incremento de la distancia se realiza cada 50 vueltas para que el incremento mínimo se pueda representar en la pantalla y se ahorre una localidad de memoria que almacenaría centímetros. Se ocupan entonces tres bytes de acuerdo a la figura 6.5.

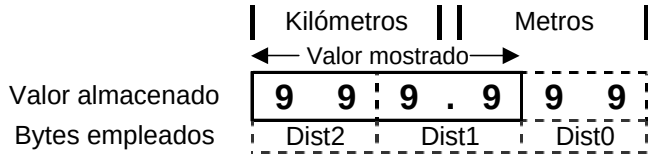


Figura 6.5: Bytes empleados para almacenar la distancia recorrida.

Para mantener la estructura decimal de la numeración, estos bytes cuentan hasta 99 y cuando pasan ese valor se les restan 100 unidades y se genera un acarreo. Los bytes Dist0 y Dist1 se incrementan una distancia igual a 50 veces el perímetro de la rueda al contarse cincuenta giros de ésta. El incremento de cada uno se almacena en dos bytes cuyo valor se cambia al modificar la rodada de la bicicleta en el modo de configuración.

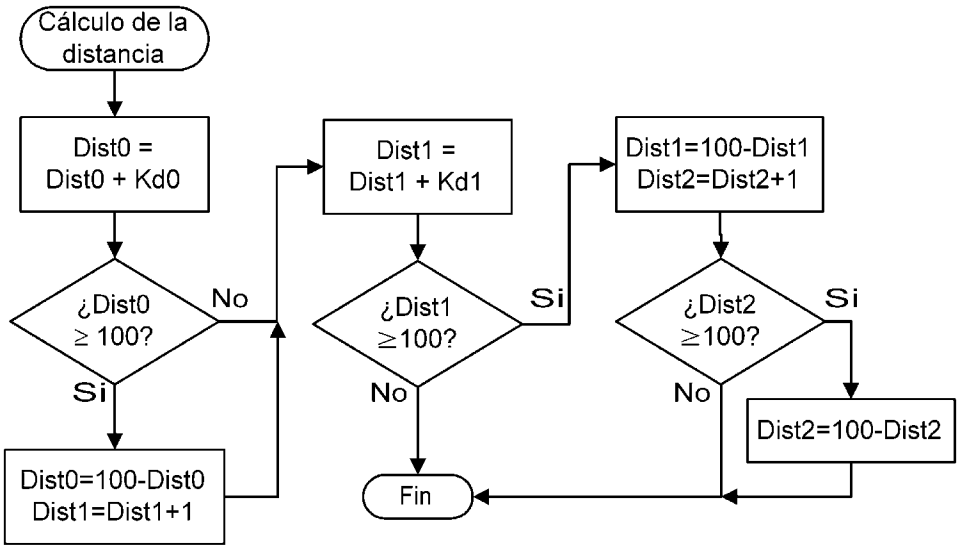


Figura 6.6: Diagrama de flujo del funcionamiento del odómetro.

Rodada	Diámetro [cm]	Perímetro [m]	En 50 Vueltas [m]	Ctte. de dist. K_{D1} x100 m	Ctte. de dist. K_{D0} x 1 m
22	55.9	1.76	88	0	88
23	58.4	1.84	92	0	92
24	61	1.92	96	0	96
25	63.5	1.99	100	1	0
26	66	2.07	104	1	4
27	68.6	2.15	108	1	8

Tabla 6.2: Valores de las constantes de incremento de la distancia con cada 50 vueltas.

6.2.3. Medición de la velocidad

La velocidad instantánea se aproxima calculando la velocidad media en un intervalo corto de tiempo. Cuando ocurre la interrupción externa por un giro de la rueda se incrementa un byte contador de vueltas que permite conocer la distancia recorrida en cierto tiempo al multiplicar su valor por el perímetro de la rueda, con lo que se obtiene la velocidad media:

$$V_{MEDIA} = \frac{d_{RECORRIDA}}{\Delta t} = \frac{P_{RUEDA} \times N_{VUELTAS}}{\Delta t} \qquad (6.1)$$

El perímetro de la rueda está en metros y el incremento de tiempo Δt en segundos, por lo que se ajusta el resultado mediante una constante para obtener la lectura de velocidad en km/h. En una hora hay 3600 segundos, y en un kilómetro 1000 metros, por lo que:

$$V[\tfrac{km}{h}] = \frac{P_{RUEDA} \times N_{VUELTAS}}{\Delta t} \cdot \frac{3600}{1000} = 3.6 \cdot \frac{P_{RUEDA} \times N_{VUELTAS}}{\Delta t} \qquad (6.2)$$

El perímetro de la rueda tiene un valor conocido por el programa, al igual que el intervalo de tiempo. La ecuación 6.2 queda en función únicamente del número de vueltas y de una constante conocida.

$$V = k \times N_{VUELTAS} \qquad \text{donde} \quad k = \frac{3.6 \cdot P_{RUEDA}}{\Delta t} \qquad (6.3)$$

El valor de *k* se define para cada valor de rodada y el incremento de tiempo se determina considerando que debe ser pequeño para aproximar el cálculo a la velocidad instantánea pero debe permitir que la lectura tenga un incremento pequeño. La tabla 6.3 muestra el incremento en la velocidad que representa cada giro de la rueda registrado en el intervalo de tiempo Δt para una bicicleta de rodada 22.

Un valor de uno o dos segundos para Δt es pequeño, pero no permite lecturas de velocidades bajas. Una espera de cuatro o cinco segundos antes de obtener la siguiente lectura es muy prolongada, aunque la velocidad mínima que se puede medir es pequeña. El cálculo de la velocidad se hace cada tres segundos, que está en la mitad y permite incrementos no tan grandes de la lectura de velocidad. La tabla 6.4 contiene el valor de la constante *k* empleada para el cálculo de la velocidad para las diferentes rodadas.

Δt	Velocidad por giro [km/h]
1	6.34
2	3.17
3	2.11
4	1.58
5	1.27

Tabla 6.3: Incremento de la velocidad por giro de la rueda para diferentes valores del intervalo de tiempo en que se realiza la medición. Valores calculados para una bicicleta de rodada 22 y 1.76 metros de perímetro.

Cada tres segundos se actualiza la lectura tomando el número de vueltas que se almacena en un byte y se multiplica por la constante de velocidad k_V, expresada mediante dos bytes (tabla 6.4). Como el conjunto de instrucciones del PIC16F877 no incluye la multiplicación, se ejecutan una serie de sumas como se indica en el diagrama de flujo de la figura 6.7. La velocidad se almacena en dos bytes aunque solo el que guarda la cantidad entera de km/h se muestra en la pantalla.

Rodada	Velocidad mínima (km/h)	Ctte. de velocidad k_{V1} x 1 km/h	Ctte. de velocidad k_{V2} x 0.01 km/h
27	2.58	2	58
26	2.48	2	48
25	2.39	2	39
24	2.30	2	30
23	2.21	2	21
22	2.11	2	11

Tabla 6.4: Velocidad mínima que se puede medir para cada rodada en un intervalo de 3 segundos.

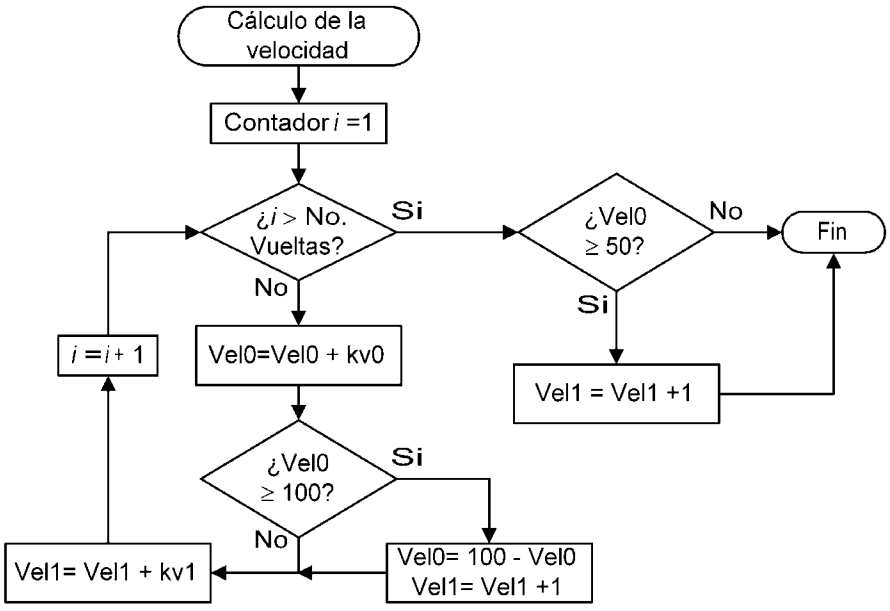


Figura 6.7: Diagrama de flujo para el cálculo de la velocidad.

6.2.4. Medición de temperatura

Como la resolución de la conversión de temperatura es de 5mV/ºC (apartado 5.2.3), cada resultado de 8 bits del convertidor tiene una medida directa que solo requiere que se haga un corrimiento a la derecha. El bit menos significativo, que representa medio ºC, pasa al bit de acarreo del registro de estado del PIC16F877. Por medio de él se determina si se incrementa el valor resultante de la división entre dos (corrimiento a la derecha) con lo que se redondea el resultado a un valor entero de grados Celsius. Las conversiones se realizan cada segundo para permitir tener una lectura estable del valor de la temperatura.

6.2.5. Medición del nivel de intensidad luminosa

La lectura del convertidor se toma de la parte alta del resultado, pues el voltaje del sensor de luz es mayor que el de los sensores de temperatura. Después se compara con los límites que determinan los cuatro intervalos con que se expresa cualitativamente el nivel de iluminación. Estos límites corresponden a las lecturas del convertidor cuando la iluminación es:

- **Mala**. En este intervalo se enciende el indicador luminoso de forma intermitente dos veces por segundo.
- **Regular**. En este intervalo se enciende el indicador luminoso de forma intermitente una vez por segundo.
- **Buena**. La iluminación permite buena visibilidad, por lo que se apaga el indicador luminoso.
- **Excelente**. La iluminación es la mejor y se apaga el indicador luminoso.

6.2.6. Frecuencia cardiaca

La señal proveniente del sensor de pulso entra por la terminal de impulsos del Timer 0. Se emplea este temporizador como acumulador de pulsos cardiacos. Al cabo de 60 segundos se tendría en él la lectura de la frecuencia cardiaca, pero el tiempo de espera para obtener una nueva lectura es demasiado prolongado.

El método empleado es el de tomar 15 lecturas en igual número de segundos para sumarlos y multiplicar el resultado por 4, obteniendo así la proyección del valor en 60 segundos. La multiplicación se realiza mediante dos corrimientos a la izquierda. Para las siguientes actualizaciones se recorren las muestras y se deshecha la más vieja. El valor actual del Timer 0 se suma con las 14 restantes y se repite el proceso. Esto permite que en cada segundo se tenga una nueva lectura de la frecuencia cardiaca utilizando la cuenta de los pulsos que ocurrieron en los 15 segundos recientes.

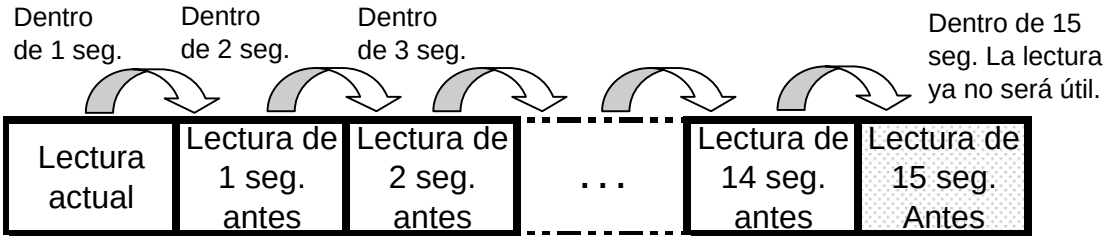


Figura 6.8: Empleo de 15 lecturas de los pulsos cardiacos por cada segundo para obtener la frecuencia cardiaca.

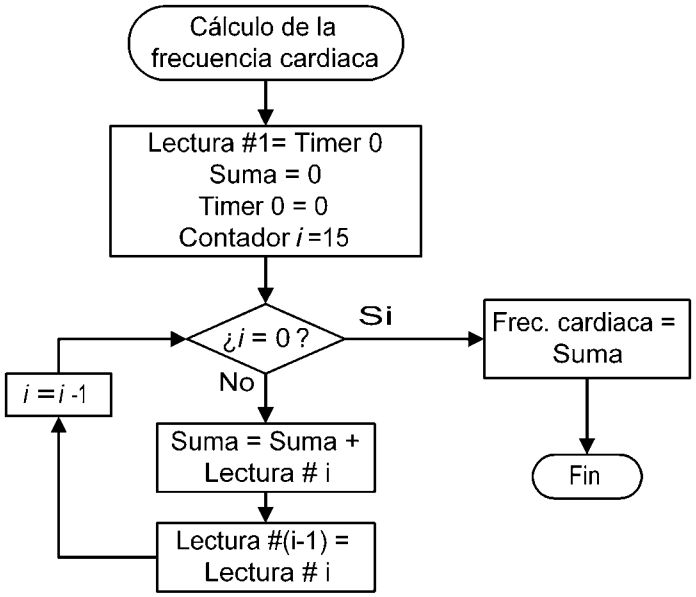


Figura 6.9: Diagrama de flujo para el cálculo de la frecuencia cardiaca.

6.3. Organización del código

Para facilitar el desarrollo del programa y su depuración, se separó en varios archivos de texto para cada uno de los conjuntos generales especificados en el apartado 2.5.2. El ensamblador interpreta consecutivamente las líneas de un archivo incrustado mediante la directiva *INCLUDE* como si formaran parte del archivo en que se hace la referencia.

El resultado del proceso de ensamblado es un solo archivo que contiene el código hexadecimal de cada una de las instrucciones del programa. Éste es el que indica al programador la información que debe almacenar en cada una de las palabras de la memoria de programa del PIC16F877.

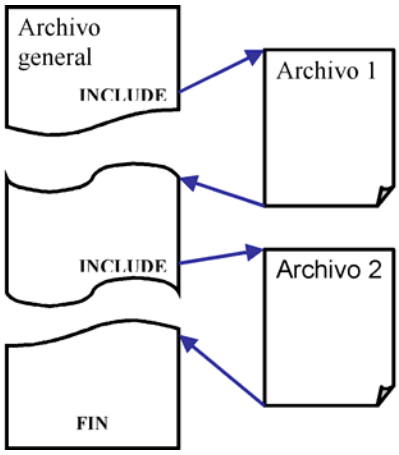


Figura 6.10: Separación de un proyecto en varios archivos.

El archivo general del proyecto se llama ***proyecto.asm*** y contiene los siguientes elementos:

- Inclusión del archivo *PIC16F877.inc* que contiene las definiciones de los registros de propósito específico y que es distribuido junto con el ensamblador MPASM.
- Declaración del modelo de microcontrolador empleado.
- Valor a almacenar en la palabra de configuración al momento de programar el dispositivo.
- Directiva de inclusión para el archivo que contiene la asignación de memoria.
- Directiva de inclusión para el archivo que contiene la iniciación de valores y recuperación de parámetros.
- Instrucción de salto al origen del programa en el vector de reset.
- Directiva de inclusión, a partir del vector de reset, del archivo que contiene las rutinas de atención a interrupciones.
- Directiva de inclusión del archivo que contiene las rutinas de operación normal.
- Directiva de inclusión del archivo que contiene las rutinas de operación en modo de configuración.
- Directiva de inclusión de los archivos que contiene las subrutinas.

Al ser tratados por el ensamblador como uno solo, todos los archivos pueden incluir instrucciones de salto a posiciones dentro de cualquiera de los demás archivos. Las etiquetas asociadas con registros de propósito general, bits o posiciones dentro de la memoria de programa no pueden duplicarse en otros archivos.

6.4. Preparación del microcontrolador

6.4.1. Asignación de memoria

Dentro del código fuente en lenguaje ensamblador se usan etiquetas para asociar un nombre a valores constantes o localidades de la memoria de datos y de programa. El ensamblador sustituye a las etiquetas con el valor o la dirección que tienen asociadas para generar el código hexadecimal del programa.

En el archivo ***asig_mem.asm*** se asigna una etiqueta a cada registro de propósito general o localidad de memoria EEPROM que utiliza el programa para almacenar datos. También se definen los valores de algunas constantes y el nombre de los bits de algunos registros.

Asignación de memoria EEPROM

Esta sección asigna nombres a algunas direcciones de la memoria EEPROM mediante etiquetas que las identifica a lo largo del programa. Al mismo tiempo determina el valor inicial que será almacenado durante la programación del microcontrolador. Aunque su contenido será modificado durante el funcionamiento del dispositivo, se establecen valores iniciales válidos que se emplean durante la primera ejecución que sigue a la programación.

Asignación de registros

Cada una de las variables empleadas en el programa se identifica mediante una etiqueta y se asocia con la dirección del registro que almacena su valor. Las instrucciones del programa hacen referencia a la etiqueta del registro en lugar de incluir su dirección en memoria.

Asignación de nombre a bits

Algunos registros de propósito general definidos en el apartado anterior tienen identificados sus bits mediante una etiqueta asociada con el número de su posición dentro del registro (del 0 al 7). También se asocian algunos bits de los puertos con algún nombre para identificarlos dentro del programa.

Definición de constantes

Algunos valores constantes pueden ser empleados en diferentes lugares del programa. Si se modifica su valor se tendrían que realizar modificaciones en distintos puntos del programa. Al asignarle una etiqueta a un valor constante permite que en el programa se emplee únicamente ésta y que se pueda modificar su valor cambiándolo solo en la definición.

Nombre	Valor	Descripción
HOR_LIM_MAX	4	Valor del límite de horas con que se regresa a 0
D2_LIM_MAX	30	Valor del límite de kilómetros con que se regresa a 0
FCMAX_MAX	211	Valor de la FCMax con que se regresa al mínimo
FCMAX_MIN	100	Mínimo valor que puede tomar FCMax
FCMIN	30	Mínimo valor que puede tomar la frecuencia cardiaca
ILUM_U1	30	Primer umbral entre niveles de iluminación 0 y 1
ILUM_U2	90	Segundo umbral entre niveles de iluminación 1 y 2
ILUM_U3	190	Tercer umbral entre niveles de iluminación 2 y 3

Tabla 6.5: Definición de nombres para valores constantes.

Terminales de entrada y salida

Las terminales a emplear como entradas y salidas son las siguientes:

- ***AN0 y AN1*:** Entradas analógicas de temperatura.
- ***AN3*:** Entrada analógica para el sensor de intensidad luminosa.
- ***TOCKI*:** Entrada de impulsos externos para el Timer 0.
- ***RE0 – RE1*:** Líneas de control para la pantalla LCD.
- ***RD0 – RD7*:** Líneas de transferencia de datos para la pantalla LCD.
- ***INT*:** Terminal de entrada para solicitud de interrupción externa conectada al sensor de giro.
- ***RB1*:** Salida para activación de la alarma auditiva.
- ***RB2*:** Salida para activación del indicador luminoso de seguridad.
- ***RB5 – RB7*:** Entradas digitales para los pulsadores.

6.4.2. Configuración de los recursos internos

En el archivo ***conf_recur.asm*** se especifica el modo de funcionamiento que tendrán los recursos internos empleados en el programa. Esto se hace asignando valores a determinados bits de los registros asociados con cada elemento.

Registro	Valor	Para configurar
INTCON	01010000b	Interrupciones
PIE1	00000001b	Interrupciones
PIE2	00000000b	Interrupciones
T2CON	01111011b	Timer 2
TRISA	00011011b	Puerto A
ADCON1	10000100b	Puerto A y convertidor A/D
ADCON0	10000000b	Convertidor A/D
OPTION_REG	10101000b	Timer 0 e interrupción externa
T1CON	00110000b	Timer 1
TRISB	11100001b	Puerto B
TRISC	00000000b	Puerto C
TRISD	00000000b	Puerto D
TRISE	00000000b	Puerto E

Tabla 6.6: Valores almacenados en los registros para configurar los recursos utilizados.

Interrupciones

Solo se emplean la interrupción externa y la interrupción por desborde en el Timer 1. El resto de las posibles interrupciones quedan inhibidas.

Timer 2

Este temporizador se usa para contar periodos de tiempo y así generar retardos de diferentes duraciones. Se configura para emplear el predivisor con lo que la frecuencia con que se incrementa su valor es 16 veces menor que la del sistema. El Timer 2 tiene asociado un comparador, que activa la bandera de interrupción cuando el temporizador alcanza el valor que contiene. Esto sucede aún cuando la interrupción del Timer 2 está deshabilitada y un predivisor permite ampliar el número de comparaciones con que se activa la bandera hasta. Para generar retardos de algunas decenas de milisegundos se activa en su valor máximo (16).

Puerto A, convertidor A/D, TMR0 e INT

Estos cuatro recursos se configuran al mismo tiempo pues comparten registros de control. En el puerto A se establecen como líneas de entrada las terminales RA0, RA1, RA3 y RA4. El convertidor A/D emplea las primeras tres como entradas analógicas para sus canales de conversión. El Timer 0 recibe impulsos externos a través de la terminal RA4/T0CKI detectando flancos ascendentes y no se le asocia el predivisor que comparte con el Watchdog. Finalmente, la interrupción externa trabaja con flancos descendentes por la terminal RB0/INT.

Timer 1

Este temporizador emplea la frecuencia interna del sistema, dividida entre 8 mediante el predivisor, para incrementar su valor.

Puertos B, C, D y E

Las terminales del puerto B RB0 y RB5 – RB7 se configuran como entradas digitales. Las demás funcionan como salidas. Los puertos C, D y E quedan configurados como salidas digitales. En algunos procedimientos de manejo de la pantalla LCD el puerto D se configura como entrada al ejecutar comandos de lectura.

6.4.3. Iniciación de variables y recuperación de parámetros

En el archivo ***ini_var.asm*** se inicializa el LCD llamando a sus subrutinas respectivas y se verifica que no se halla producido una reinicio del sistema por bajo voltaje de alimentación. La razón de hacerlo hasta este momento es que se deben configurar los puertos y el LCD para poder desplegar el mensaje que indica al usuario esta situación. El microcontrolador entrará, si es el caso, en el modo de bajo consumo, deteniendo por completo su funcionamiento hasta que se vuelva a encender con una nueva batería.

Si no se produce esta situación, se ejecutan las instrucciones para almacenar un valor inicial a las variables empleadas por el programa. Además se llama a una subrutina destinada a recuperar de la EEPROM el valor de la rodada, la frecuencia cardiaca máxima y el de los límites de tiempo y distancia para almacenarlos en sus respectivos registros de la RAM.

Ya guardado el valor de la rodada en RAM se llama a una subrutina que determina el valor de las constantes de tiempo y velocidad correspondientes a este valor. Al final recupera de la memoria EEPROM las lecturas de tiempo y distancia almacenadas.

6.5. Subrutinas

Para evitar repetir a lo largo del programa procedimientos similares se emplean subrutinas que ejecutan una tarea específica. En algunos casos utilizan el valor almacenado en el registro de trabajo W al momento de ser llamadas.

6.5.1. Subrutina *RETARDO*

El archivo que la contiene se llama ***sr_retardo.asm***. Se emplea para generar retardos de tiempo desde un milisegundo y hasta 120. Emplea el Timer 2 con su registro de comparación [4], en el que se almacena el valor guardado en W y que determina la duración del retardo de acuerdo a la tabla 6.7.

Retardo [ms]	Valor decimal en W al hacer la llamada	Retardo [ms]	Valor decimal en W al hacer la llamada
5	20	35	137
10	39	40	156
15	59	45	176
20	78	50	195
25	98	55	215
30	117	60	234

Tabla 6.7: Valor que debe almacenarse en W para generar un retardos por medio de la subrutina *RETARDO* con una frecuencia de reloj de 2.097152 MHz.

6.5.2. Subrutinas de manejo de memoria EEPROM

Son cuatro subrutinas ubicadas dentro del archivo ***sr_eeprom.asm*** y que están destinadas a almacenar y leer en la memoria de datos no volátil. Se emplean los procedimientos recomendados por el fabricante para escribir y leer en esta memoria [5].

LEER_EE

Toma la dirección de memoria EEPROM a leer de W. El valor recuperado lo coloca en W antes de retornar.

ESCRI_EE

Toma la dirección de destino de W y el dato a guardar del registro DAT_ESC_EE.

EE_LECT_DAT

Recupera las lecturas de tiempo y distancia almacenadas en memoria EEPROM y las guarda en sus respectivos registros en la memoria RAM.

EE_LECT_PAR

Recupera los límites de tiempo y distancia, el valor de la rodada y de la frecuencia cardiaca máxima de la memoria EEPROM y los almacena en sus respectivos registros en la memoria RAM.

6.5.3. Subrutina *ROD_AJUSTE* para asignación de constantes

A partir del valor almacenado en el registro RODADA almacena el valor correspondiente de las constantes de velocidad y distancia (tablas 6.1 y 6.3) en los registros destinados a este propósito. Los valores asociados a cada rodada forman parte del código de una instrucción, por lo que se ubican dentro de la memoria de programa. El archivo en que se encuentra esta subrutina se llama ***sr_rodajuste.asm***.

6.5.4. Subrutinas de control de la pantalla LCD

En el archivo ***sr_lcd.asm*** se ubican las subrutinas que configuran y envían los caracteres a la pantalla LCD mediante los puertos E y D.

LCD_OCUP

Esta rutina es la única que realiza lecturas en el módulo LCD. Se llama para esperar a que se desocupe el LCD antes de enviar otro comando. Lee la bandera BUSY del LCD que indica que aún está ejecutando el comando anterior. Además implementa un retardo para evitar que el microcontrolador se quede *congelado*, esperando a que el LCD termine, lo que no le permitiría realizar otros procedimientos como la activación de las alarmas. La subrutina concluye su ejecución cuando el LCD se desocupa, o antes, si el retardo concluye su cuenta.

LCD_INI

Ejecuta procedimiento para preparar la pantalla LCD para recibir caracteres [6], configurándola para trabajar con 8 líneas de datos y con dos líneas de 16 caracteres cada una.

LCD_BORRA y LCD_POS

La primera borra el contenido de la pantalla y regresa el cursor a la primera posición, mientras que la segunda solo ubica al cursor en la posición indicada por el valor en W.

LCD_TEXTO

Envía el código ASCII de un carácter para ser mostrado en la posición actual del cursor.

6.5.5. Subrutina de conversión BCD – ASCII

Esta rutina, que se ubica dentro del archivo ***sr_bcdascii.asm***, toma el valor que se encuentre en el registro de trabajo W y lo convierte a BCD extendido[1] almacenándolo en tres registros de la memoria RAM. Para realizar esto ejecuta el siguiente procedimiento:

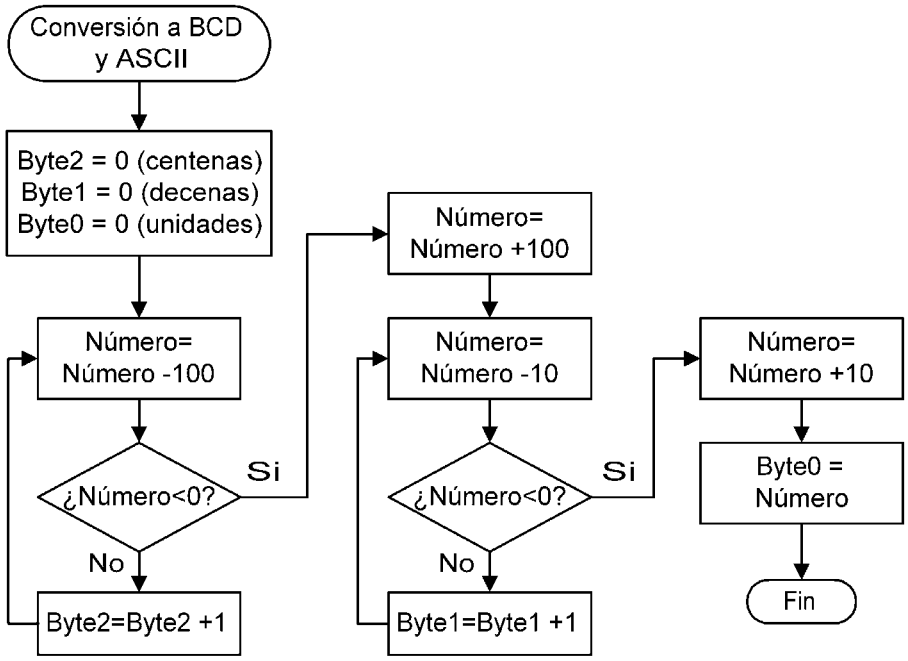


Figura 6.11: Diagrama de flujo de la conversión a BCD extendido.

Después de realizar la conversión envía el código ASCII de cada dígito de acuerdo al valor de algunos bits de su registro de control (BCD_CONT) definido en la memoria RAM. Su valor determina la forma en que la subrutina envía los datos a la pantalla LCD. Las opciones disponibles son las siguientes:

- Cuando el número es menor a 10 se puede enviar para las decenas el código ASCII del dígito cero o un espacio. Cuando el número es menor a 100 siempre se envía para las centenas un espacio en blanco en lugar del código ASCII del dígito cero.
- Se puede seleccionar que dígitos que se envían al LCD. Si no se envía ninguno únicamente se realizará la conversión a BCD.

[1] Este código representa un número decimal separando cada dígito en un byte.

- Se puede enviar el código ASCII de un punto para representar números no enteros de hasta tres dígitos.

6.5.6. Subrutina de envío de mensajes a la pantalla LCD

Despliega en la primera línea el mensaje "Guardando" (subrutina *MENSAJE_G*) al momento de almacenar datos en memoria EEPROM, o el mensaje "Leyendo" (subrutina *MENSAJE_L*) cuando se están leyendo datos desde esta memoria. Cuando el microcontrolador se ha reiniciado por un bajo nivel de voltaje, presenta el mensaje “Baterías” (subrutina *MENSAJE_B*).

6.6. Operación normal del sistema

El funcionamiento normal del dispositivo se controla mediante las instrucciones contenidas en el archivo ***op_normal.asm***, excepto la rutina de atención a interrupciones que se encuentra dentro de archivo ***interrupcion.asm***. Como puede observarse en el diagrama de flujo de la figura 6.12, después de realizar los procedimientos iniciales se entra en un ciclo que solo puede ser interrumpido por medio de un botón que determinará el cambio al modo de configuración (figura 2.7).

Los procedimientos iniciales consisten en borrar el valor del Timer 1 para iniciar la cuenta de tiempo y borrar las quince lecturas de pulsos cardiacos para reiniciar la medición de la frecuencia cardiaca.

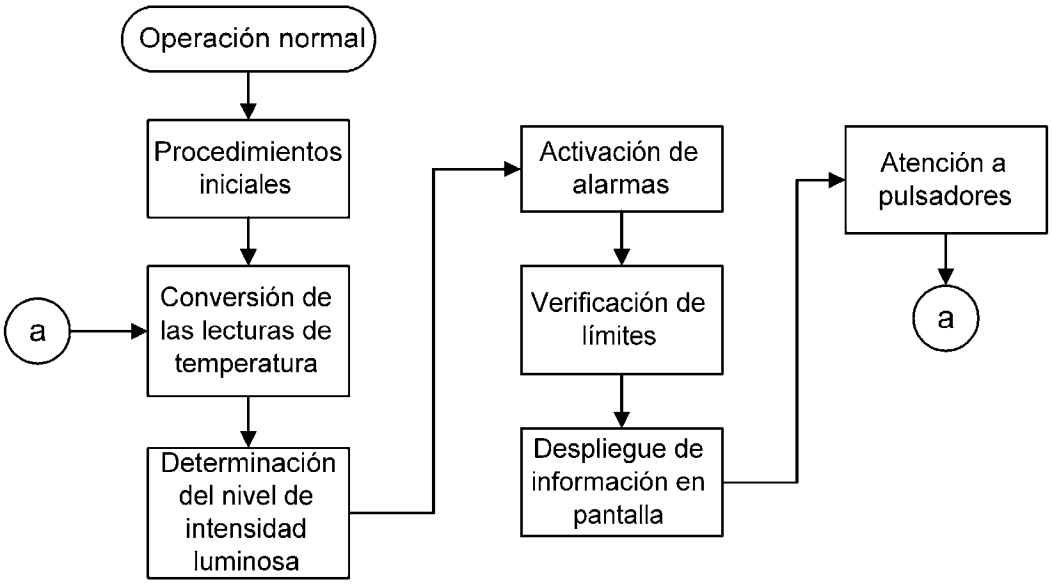


Figura 6.12: Diagrama de flujo de la operación normal del dispositivo.

6.6.1. Interrupciones

El funcionamiento del dispositivo está controlado básicamente por la base de tiempo de un segundo que está implementada mediante una interrupción. El sensor de giro también activa una interrupción para indicar que se ha cumplido una nueva vuelta de la rueda. La rutina de atención a interrupciones ejecuta los siguientes pasos.

Almacenamiento en memoria RAM del estado actual del sistema

Aunque el valor del contador de programa se salva automáticamente en la pila al responder a una solicitud de interrupción, debe guardarse al principio de la rutina de atención el valor del registro de trabajo W, del registro de estado STATUS y la parte alta del contador del programa PCLATH que sirve para saltos entre posiciones en distintas páginas de la memoria de programa. Se emplean instrucciones que no afecten a los bits del registro STATUS siguiendo el método indicado por el fabricante [7].

Identificación de la fuente de la interrupción

Mediante las banderas respectivas de cada fuente de interrupción se verifica cual de ellas ha sucedido. Como solo se emplean la interrupción del Timer 1 y la externa, solo estas banderas son verificadas, y en caso de que ninguna esté levantada se sale de la rutina de interrupción. Después de terminar la atención a cualquiera de las dos posibles fuentes, se debe bajar su bandera y regresar a este lugar por si se presentaron las dos al mismo tiempo. La jerarquía se determina por el orden de verificación de las banderas, siendo la principal la rutina del Timer 1 pues maneja la base de tiempo.

Interrupción del TIMER 1

En esta parte se ejecutan cuatro procedimientos, después de los cuales se baja la bandera de la interrupción y se regresa a la identificación de la fuente de interrupción.

1. Se resta un segundo al contador de tiempo para la lectura de las temperaturas, que se realiza cada diez segundos. Si éste llega a cero se activa el bit de permiso para la medición de temperatura y se reestablece el valor inicial del contador en 10.
2. Se verifica que el corono-odómetro esté funcionando para determinar si se ejecuta el incremento al cronómetro llevando a cabo el método respectivo (figura 6.4).
3. Cada tres segundos se realiza un nuevo cálculo de velocidad tomando la lectura de las vueltas transcurridas desde la ocasión anterior, por lo que se implementa un contador de tiempo. Con cada interrupción del Timer 1 se descuenta un segundo de él. Al llegar a cero se reestablece su valor inicial de 3 y se ejecuta el método de medición de la velocidad (figura 6.7).
4. El cálculo de la frecuencia cardiaca (figura 6.8) se lleva a cabo cada segundo, pero se emplea un contador de tiempo para indicar si las primeras quince lecturas han sido tomadas. Cuando el contador llega a cero el programa sabe que la lectura de la frecuencia es válida y se puede mostrar en la pantalla.

Interrupción externa

Incluye tres procedimientos que se ejecutan con cada giro de la rueda y posteriormente se baja la bandera de la interrupción y se retorna a la sección de identificación de fuentes de interrupción.

1. Se verifica que el corono-odómetro esté funcionando para determinar si se ejecuta el incremento en la distancia llevando a cabo el método respectivo (figura 6.6).

2. Se incrementa el contador de las vueltas para calcular velocidad. Esta cuenta es reiniciada cada tres segundos por el método de cálculo para la velocidad.
3. Se verifica si existe una solicitud de para arrancar el crono-odómetro, y de ser el caso, se pone a uno el bit que habilita los incrementos al tiempo y a la distancia. Esto se realiza después del incremento del odómetro debido a que se arranca con el primer giro de la rueda después de solicitarse, por lo que esa vuelta no debe considerarse para la distancia recorrida.

Recuperación de memoria RAM del estado actual del sistema

Los valores almacenados en memoria la principio de la rutina se recuperan y guardan en sus registros.

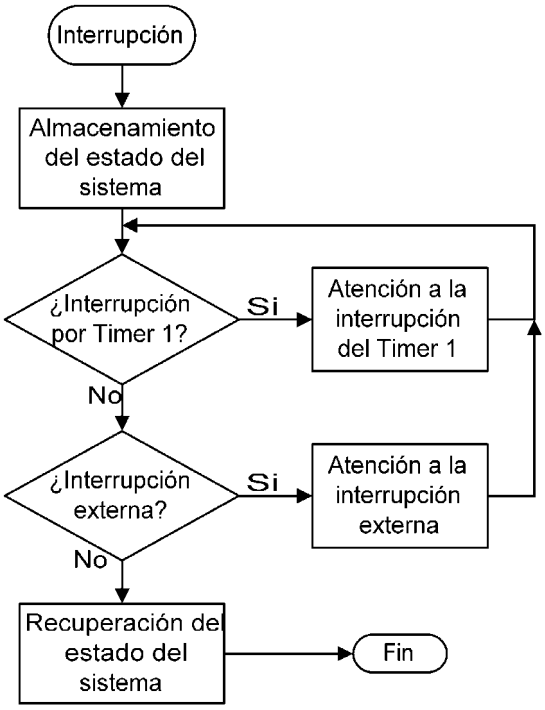


Figura 6.13: Diagrama de flujo de la rutina de atención a interrupciones.

6.6.2. Conversión de las lecturas de temperatura

Se verifica el bit de permiso para realizar estas conversiones, que a activa cada diez segundos con la finalidad de proporcionar lecturas estables haciendo un muestreo periódico. Se ejecuta la conversión de acuerdo a lo indicado en el apartado 4.8. Se toma el resultado de la conversión de los canales AN0 y AN1, el cual se coloca justificado a la derecha, y se almacena en el registro correspondiente a cada temperatura.

6.6.3. Determinación del nivel de intensidad

Se realiza la conversión del canal AN3 y se ajusta el resultado a la izquierda, pues los voltajes recibidos del sensor de intensidad de luz son mayores. La lectura es copiada en el registro destinado a almacenar este valor, por medio del cual se determina en cuál de los cuatro niveles

se encuentra la medición. Con base a esto se establece el carácter que indica dicho nivel en pantalla y se activa o desactiva el indicador luminoso.

6.6.4. Activación de alarmas

Se activan las alarmas visuales y auditivas mediante la comparación de la velocidad actual con la velocidad límite, y de la frecuencia cardiaca con la frecuencia cardiaca máxima guardada por el usuario. Se determina el carácter a mostrar en la pantalla en cada caso y la activación o desactivación del zumbador. Con la finalidad de distinguir el origen de la alarma auditiva, este último es accionado una vez por segundo para la alarma de velocidad y cuatro veces por segundo para la alarma de frecuencia cardiaca siendo esta última la que tiene preferencia.

6.6.5. Verificación de límites

Si el corono-odómetro está funcionando se verifica que el valor de tiempo y distancia no rebasen su límite respectivo. En caso de hacerlo se detiene su funcionamiento y no se puede volver a arrancar mientras se mantenga esta condición. Para volver a accionarlo se debe reiniciar su cuenta, desactivar los límites o cambiar el valor de éstos.

6.6.6. Despliegue de información en pantalla

Se envían al LCD los datos y caracteres de acuerdo a la figura 5.18. Se emplean para esto las rutinas para manejo del LCD y para conversión BCD-ASCII.

6.6.7. Atención a pulsadores

Se sigue el procedimiento indicado por el diagrama de flujo de la figura 5.8, pero para evitar que al mantener presionado un botón se detenga el funcionamiento del dispositivo y no se puedan realizar los demás procedimientos, se indicará mediante un bit cuando se esté esperando a que se suelte algún botón. Después de detectar una pulsación válida se verifica este bit y en caso de estar activo no se atiende ningún botón. El funcionamiento de los botones queda determinado por el estado de operación.

Crono-odómetro detenido

Si se presiona el botón 1 bajo estas condiciones se generará una solicitud de arranque, para que el crono-odómetro comience a funcionar con el siguiente giro de la rueda. El botón 2 sirve para establecer el valor de las lecturas de tiempo y distancia. Si se presiona por menos de dos segundos se pondrán en ceros, y si se mantiene por más tiempo se recuperaran las lecturas almacenadas en la memoria EEPROM. El tercer botón provocará que se salga del modo de operación normal para pasar al modo de configuración.

Crono-odómetro contando o en espera de arrancar

El botón 1 detiene el crono-odómetro o cancela la solicitud de arranque. Si se mantiene presionado por más de dos segundos, almacena las lecturas actuales de tiempo y distancia en memoria EEPROM sustituyendo a las que se hubieran guardado con anterioridad. El segundo botón sirve para incrementar el valor de la velocidad límite, y el tercero para habilitarlo o deshabilitarlo.

6.7. Modo de configuración

La configuración de los parámetros y límites se controla mediante las instrucciones contenidas en el archivo ***modo_conf.asm***. Como puede observarse en el diagrama de flujo de la figura 6.14, después de realizar los procedimientos iniciales se entra en un ciclo que solo puede ser interrumpido por medio del tercer botón. De acuerdo al diagrama de flujo de la figura 2.7, al concluir la ejecución del modo de configuración se regresa a la operación normal.

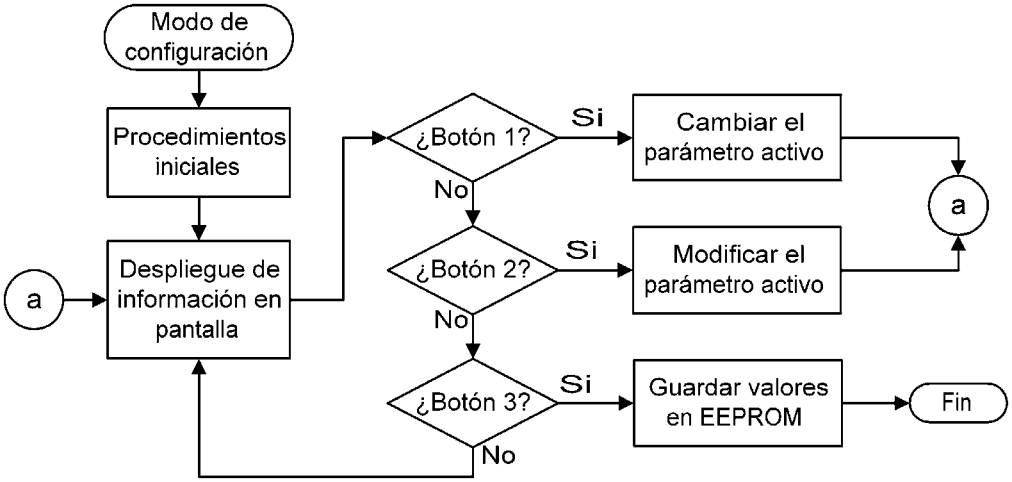


Figura 6.14: Diagrama de flujo del modo de configuración.

Los procedimientos iniciales consisten en deshabilitar las interrupciones, borrar el indicador de parámetro activo, borrar la pantalla y desactivar la alarma auditiva. Si en un lapso de 4 segundos no se ha presionado ningún botón, se regresa a modo normal de operación.

6.7.1. Despliegue de información en pantalla

Se envían al LCD los datos y caracteres de acuerdo a la figura 5.19. Se emplean para esto las rutinas para manejo del LCD y para conversión BCD-ASCII.

6.7.2. Cambio del parámetro activo

El parámetro activo determina el valor que será afectado al presionar el botón dos y se indica mediante una cadena de texto en pantalla. Al principio ninguno está activo y el orden en que se cambian es el siguiente (al llegar al final de la lista se vuelve al primer elemento):

- Límite de tiempo
- Límite de distancia
- Límites activos
- Rodada
- frecuencia cardiaca máxima (FCMax)
- Ninguno

6.7.3. Modificación del parámetro activo

El valor del parámetro que se encuentre activo se incrementa, excepto en el caso de los límites activos, en que con cada vez que se presione se cambiará entre estas tres opciones:

- Límite de tiempo activo
- Límite de distancia activo
- Ambos límites activos
- Ningún límite activo

Con cualquier modificación que se realice se activará el indicador de modificaciones para que los datos sean almacenados al salir.

6.7.4. Almacenamiento de parámetros

Antes de salir y volver a la operación normal del dispositivo se verifica si existe alguna modificación para guardar en memoria EEPROM los límites, la rodada y la FCMax. La información del límite activo no tiene asociada una localidad en memoria EEPROM, por lo que no se guarda.

6.8. Referencias

1 Lance A. Leventhal, *Z80: Assembly language programming*, “Problem definition and program design”, Berkeley: Ed. McGraw Hill, 1979, p. 13-1.

2 *Ibíd*., p. 13-16 – 13-49.

3 *Ibíd.*, pp. 13-31 – 13-34.

4 *PIC16F87X Data Sheet. 28/40-Pin 8-Bit CMOS FLASH Microcontrollers*, Microchip Technology Inc., USA: 2001, pp. 55-56.

5 *Ibíd*. p. 43.

6 *AN587. Interfacing PICmicros to an LCD Module*, Microchip Technology Inc., USA: 1997, p 3.

7 *PIC16F87X Data Sheet, Op. Cit.*, p. 130.

Capítulo VII

Implementación

7.1. Herramientas

El lenguaje empleado para la realización del programa es el lenguaje ensamblador del PIC16F877, debido a que permite manejar directamente los recursos del microcontrolador, se reduce el tamaño código y facilita la depuración. El emplear otro lenguaje, como el *C*, hubiera significado un mayor tamaño del código binario del programa además de requerirse un compilador, el cual tiene cierto costo, a diferencia del programa ensamblador que es gratuito.

A todas las instrucciones en el código fuente se les acompañó de una línea de comentario que permitiera identificar su propósito y se separaron en varios archivos para localizar los procedimientos más rápido y evitar que se volviera confuso el programa.

La escritura del programa se realizó mediante el entorno de desarrollo MPLAB® IDE, distribuido por Microchip a través de su página de Internet, en su versión 5.5 que soporta el microcontrolador utilizado y es en la que mayor experiencia se tenía. Ésta incluye diversas herramientas como ensamblador, editor de texto, simulador, administrador de proyectos y archivo de ayuda.

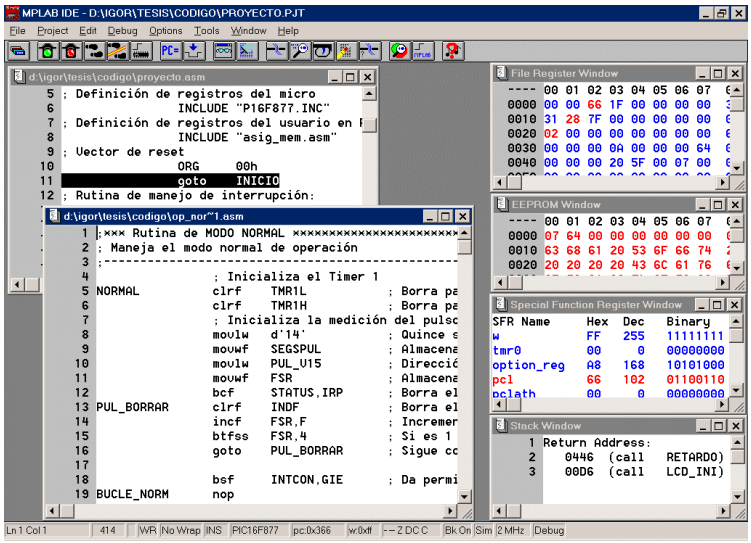


Figura 7.1: Vista del entorno de desarrollo MPLAB de Microchip.

El ensamblador que incluye este entorno es el MPASM, ensamblador universal para toda la línea de microcontroladores PIC de Microchip, en su versión 3.00. El simulador incluido permitió probar las subrutinas y varios procedimientos el programa sin tener que emplear el microcontrolador.

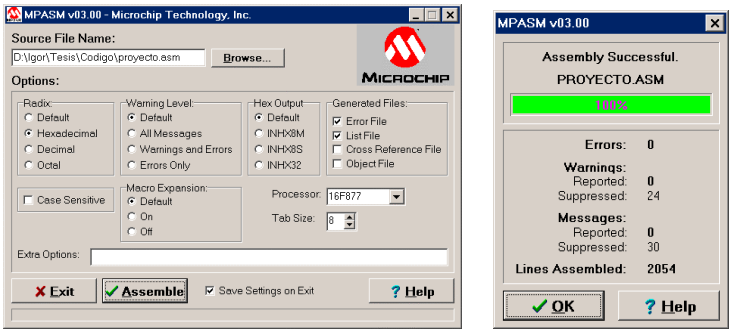


Figura 7.2: Vista del ensamblador MPASM 3.0 de Microchip.

El circuito programador que se utilizó es un programador de dominio público que se comunica con un programa de computadora llamado IC-Prog, y del que puede obtenerse información en la página de Internet *http://www.ic-prog.com*.

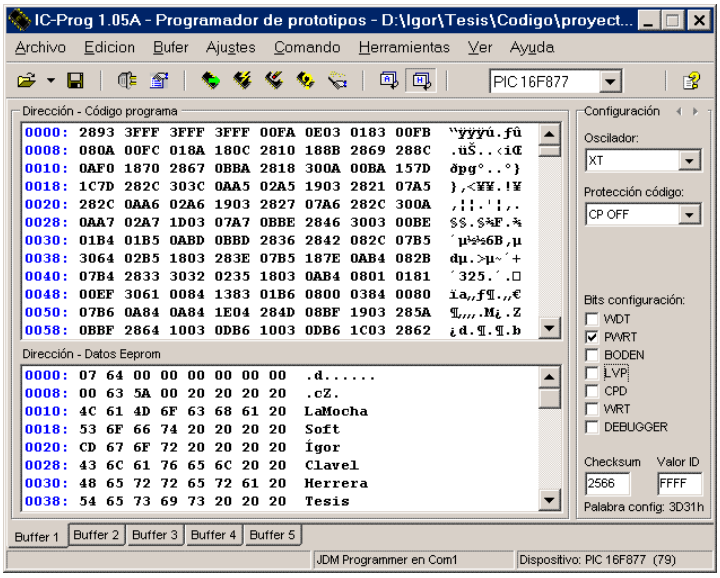


Figura 7.2: Vista del programa IC-Prog para grabación de microcontroladores PIC y otros dispositivos.

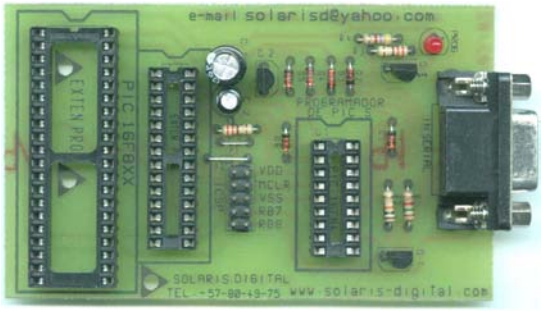


Figura 7.3: Vista del circuito programador de microcontroladores PIC de 8, 28 y 40 terminales.

7.2. Desarrollo y pruebas

El hardware se implementó en tabletas *protoboard*, que permiten una rápida conexión de los circuitos. Se seleccionó esta opción como primera etapa de la implementación ya que el número de conexiones no es significativamente grande, lo que permite una minuciosa revisión de los contactos entre terminales. Además, durante el desarrollo y las pruebas se fueron conectando los componentes de manera progresiva, por lo que el tiempo dedicado a esto se redujo. Sin embargo, el circuito no es completamente fiable, por lo que al estar definidos los circuitos se soldaron sobre una tarjeta perforada empleando alambre adecuado.

La implementación del circuito se realizó de manera progresiva, para que el desarrollo del hardware fuera a la par del desarrollo del programa. Esto permitió tener funcionando de forma correcta y al mismo tiempo los distintos sensores y circuitos junto con las rutinas dedicadas a manejarlos. Las pruebas realizadas tuvieron como finalidad depurar el funcionamiento tanto del hardware como del software.

Es importante hacer notar que a pesar de un buen funcionamiento del dispositivo y de sus componentes, éste solo no basta para procurarle beneficios al usuario al practicar su deporte pues tiene fines únicamente preventivos y de apoyo. Se deben tomar siempre en cuenta las consideraciones hechas en el capítulo I respecto a las características del entrenamiento (esfuerzo, intervalos de reposo y tanto condiciones ambientales como la frecuencia con que se realiza) y del cuidado de la salud física. Siempre se debe recordar que no es un instrumento de diagnóstico si no sólo de apoyo y prevención, por lo que ante cualquier signo de alerta deberá acudirse con un médico.

7.2.1. Desarrollo previo de software

Antes de comenzar la implementación física se desarrollaron las subrutinas y procedimientos que podían ser depurados mediante el uso del simulador del entorno de desarrollo. Esto permitió contar con programas de prueba confiables para verificar el funcionamiento los circuitos.

Se programaron y depuraron las subrutinas que se emplean en el programa. Para verificar que funcionan adecuadamente se revisó el resultado en los registros que modifican. De igual forma se implementaron los algoritmos de medición y se simularon mediante una ejecución cíclica, con lo que cada repetición representaba un evento (giro de la rueda, un segundo transcurrido o un pulso cardiaco.

Estos procedimientos se incluyeron después dentro de la rutina de interrupción, simulándose mediante software únicamente el desborde del Timer 1, por lo que los procedimientos correspondientes a la interrupción externa fueron ejecutados simulando una vuelta cada segundo. Estas pruebas dieron como resultado que la rutina de atención ejecutara correctamente.

7.2.2. Implementación del hardware

Durante este proceso se fueron realizando las pruebas de hardware sobre los circuitos por separado, algunas veces para verificar su funcionamiento y otras para comprobar su compatibilidad con el microcontrolador. A medida que se avanzaba en la conexión de los circuitos, se fueron escribiendo y probando también las rutinas que trabajarían con éstos. Esto

sirvió para ir probando los procedimientos que se iban escribiendo en el código del programa de acuerdo a las etapas del proceso de desarrollo de software.

El primer paso consistió en armar y verificar el circuito de la fuente de alimentación para que el voltaje entregado fuera el correcto. El valor medido fue bastante cercano al esperado gracias al empleo de resistencias de precisión. Se le aplicó una carga de aproximadamente 100 mA mediante una resistencia de aproximadamente 51 ohms para verificar su regulación. Ésta fue bastante buena, bajando únicamente en 20 mV su voltaje de salida con una corriente mucho mas grande que la que se esperaba para todo el dispositivo.

Ya con la fuente funcionando, se implementó el circuito básico del microcontrolador y se le cargó un programa para incrementar el puerto D, de manera que en sus terminales se pudiera observar una señal de distinta frecuencia mediante una punta lógica. Después se conectó la pantalla LCD a los puertos correspondientes para verificar su funcionamiento. Se realizó esto antes que nada debido a que representaría un medio de salida para observar el correcto funcionamiento del microcontrolador.

Las subrutinas ya programadas se cargaron junto con un programa destinado a enviar un mensaje de bienvenida. Se grabó el microcontrolador y se colocó sobre el circuito. Con este circuito base se realizó la prueba de la base de tiempo, mostrando el valor del cronómetro en la primera línea de la pantalla y comparando su incremento con el de un cronómetro de reloj. No se observó un desfase en las cuentas.

El cálculo de la distancia y la velocidad se probaron ejecutando la rutina cada segundo. Para la frecuencia cardiaca se incrementó internamente el valor del Timer 0 cada segundo. Los resultados fueron los esperados, y comprobaron los resultados de las simulaciones por software.

Con la finalidad de dejar lista la estructura general del programa, se implementaron los pulsadores. Con ello se pudieron realizar los procedimientos destinados a arrancar y detener el crono-odómetro y a cambiar entre modo de operación normal y de configuración. Se verificó el funcionamiento del algoritmo contra rebotes, y una vez que se comprobó su correcta operación se procedió a programar lo antes mencionado.

Posteriormente se probaron las subrutinas para el manejo de la memoria no volátil. Se leyeron algunas localidades de la memoria EEPROM con un valor establecido durante la programación, y se presentaron sus valores en pantalla para verificar que fueran los mismos. La rutina de prueba también grababa otros valores y se repetía la lectura para verificar la operación de escritura.

Antes de conectar el zumbador directamente al microcontrolador, se verificó que no consumiera más corriente que la que puede entregar una terminal de salida del PIC16F877. La corriente medida fue de 2.7 mA, prácticamente diez veces menor que la máxima que es capaz de entregar el microcontrolador por una terminal. Ya comprobada la compatibilidad se conectó a su terminal de salida correspondiente y se ejecutó un programa de prueba que lo activaba con dos distintas frecuencias.

Para concluir con los medios de salida, se probó que el optoacoplador, accionado por la terminal de salida correspondiente, fuera capaz de encender los tres leds, que consumen 60 mA trabajando a 3 V. No tuvo ningún problema para encender y apagar intermitentemente incluso bajándole su voltaje de encendido hasta 1.12 V con una resistencia de 1 kΩ.

El siguiente paso consistió en configurar correctamente el convertidor analógico-digital. Primero se probó la conversión de un voltaje dentro del rango que entrega el sensor LM35 mediante un divisor de voltaje implementado con un potenciómetro. Se presentaron las lecturas en pantalla y resultaron ser cinco veces menores a las lecturas mostradas por un voltímetro, lo cual es correcto porque este último tenía una resolución de 1 mV y la del convertidor es de 5 mV.

Se implementó el sensor de intensidad y se verificó que aumentara su voltaje de salida al aumentar la intensidad de luz presente. Esta prueba determinó los valores de voltaje de salida para los límites empleados según las condiciones de luz asociadas. Se establecieron en 0.6 V (30 para el programa), 1.8 V (90 para el programa) y 3.8 V (190 para el programa).

Se programó la rutina dedicada a este medio de entrada y se presentaron en pantalla los valores directos de la lectura mediante la rutina de conversión a BCD y ASCII. Además de esto se presentó el indicador de nivel para verificar que las comparaciones fueran correctas. En los casos en que era necesario se encendió el indicador luminoso para dejar terminada esta etapa.

Como el convertidor estaba ya funcionando correctamente, se conectaron los sensores de temperatura y se presentaron en pantalla sus lecturas. Se observó que el valor era inestable, por lo que se estableció el tiempo de muestreo para presentar lecturas legibles.

Mediante un pulsador se verificó que el microcontrolador atendiera la interrupción externa. Para esto se desplegó en la pantalla el número de vueltas que se iban contando para el cálculo de la velocidad. En esta prueba ya se pasó el procedimiento de cálculo de distancia a donde correspondía.

Se conectó el sensor de efecto Hall y se verificó su funcionamiento con imanes pequeños y medianos, siendo estos últimos los más adecuados para encenderlo a una distancia de cinco centímetros. No presentó problemas para hacer cambios rápidos en su salida. De hecho, el fabricante lo recomienda para aplicaciones de medición de velocidad rotativa en motores. Se conectó su salida a la terminal de la interrupción externa, sustituyendo el pulsador empleado para las pruebas.

El pulsador para pruebas se conectó entonces a la entrada de impulsos externos para el Timer 0. Esto sirvió para verificar que la detección de los flancos fuera la correcta. Posteriormente se introdujo una señal digital proveniente de una terminal de salida. En ésta se generó una señal de un pulso por segundo y dos por segundo para verificar el funcionamiento de la rutina de cálculo de la frecuencia cardiaca. En ambos casos la lectura fue la correcta (60 y 20 pulsos por minuto). Posteriormente se pudo conectar el sensor de pulso y realizar mediciones.

El sensor de pulso fue el componente que requirió más pruebas. Fueron básicas para su desarrollo ya que antes de comenzar el diseño de las etapas de adecuación de la señal era necesario saber como se comportaba la salida del sensor. Principalmente consistieron en lecturas de la señal de entrada del sensor y de sus diferentes etapas mediante un osciloscopio. Cada una de las etapas de este sensor se fueron desarrollando y probando de manera que fueran compatibles con la anterior y entregaran una señal adecuada para la siguiente.

Posteriormente se probó su funcionamiento y durante el esfuerzo, presentándose un mejor desempeño debido a que el sensor presenta una salida mayor en estas condiciones por haber mayor presión sanguínea. Se efectuó una prueba de esfuerzo tomándose la persona que efectuó el ejercicio el pulso en la muñeca mientras otra persona tomó las lecturas del dispositivo y coincidieron sus cuentas.

La tabla 7.1 presenta los valores de voltaje en su componente alterna y directa para las distintas señales identificadas en los diagramas en las figuras 5.28 a la 5.31. Se midió su consumo de corriente durante las pruebas siendo de 7.65 mA.

Señal	VAC	VDC
V0	1.8 mV	1.626 V
V1	1.8 mV	1.738 V
V2	1.8 mV	2 mV
V3	0.52 V	0.8 V
V4	0.5 V	2.3 V
Vref	-	2.3 V
Vhist	-	0.14 V

Tabla 7.1: Voltajes medidos durante las pruebas del sensor de pulso cardiaco.

Para comprobar que no existiera un error en el programa que generara un desborde de la pila, se dejó trabajando el dispositivo durante medio día, contando el tiempo que transcurría y sin que se detuviera su funcionamiento. También se midió su consumo de corriente, que fue tan solo de 12.5 mA, lo que permite determinar el tiempo de vida de una batería en uso continuo, conociendo su capacidad.

Al trabajar tanto tiempo, la batería se fue descargando, lo que permitió implementar y probar la rutina que avisa cuando el voltaje de alimentación no es adecuado, mostrándose en pantalla un mensaje y deteniendo todo su funcionamiento.

Conclusiones

El dispositivo diseñado puede satisfacer las necesidades tanto del ciclista que realiza solo paseos ocasionales como del deportista que entrena constantemente y busca lograr cada vez un mejor desempeño con la bicicleta, sin dejar de tomar en cuenta que es un elemento auxiliar y no de diagnóstico. En cualquier caso es necesario que el ciclista conozca sus capacidades y limitaciones, que se realice exámenes médicos y que nunca dude en acudir a un doctor ante el menor síntoma de malestar.

Desde el punto de vista de ingeniería de diseño, el desarrollo de este trabajo de tesis representó una aportación significativa a mi formación académica, pero también me permitió tener una visión más clara de lo que se me presentará al desempeñarme profesionalmente. Para buscar siempre la solución más óptima fue indispensable buscar opciones adecuadas a ciertas condiciones y determinar de entre ellas la que más se ajustara a las exigencias del diseño en cuanto a desempeño, eficiencia y costo. Esto permitió desarrollar mi capacidad de juicio y mi sensibilidad para generar criterios de comparación.

Es común que al realizar un proyecto en equipo las personas aprecien los beneficios de este método de trabajo, pero mi experiencia durante este trabajo individual también me permitió darme cuenta de esto. A pesar de que el proceso de toma de decisiones se simplifica considerablemente al trabajar de forma individual, por no tener opiniones opuestas, aumenta la posibilidad de tomar una mala decisión al no ser evaluadas las opciones desde distintos puntos de vista. Uno debe ser capaz de formular soluciones de manera individual, pero la colaboración con otras personas permitirá observar aspectos que uno pudiera haber pasado por alto durante el proceso de análisis.

Estoy consciente de que aunque se va haciendo cada vez más importante la especialización, durante mi desempeño profesional deberé aplicar no sólo los conocimientos relativos al diseño digital, sino también a otros aspectos de la ingeniería. Si bien este trabajo consistió básicamente en el diseño de un sistema digital de control, uno de los aspectos que más satisfacción y beneficios me dejaron fue la necesidad de desarrollar interfaces analógicas

Algunos componentes únicamente requirieron la aplicación de criterios de evaluación, pero en los casos en que no estaba disponible en el mercado un circuito que desempeñara una función específica fue necesario diseñarlos. El sensor de pulso cardiaco fue el que más aportó a mi formación. Es cierto que al no ser un elemento de diagnóstico se redujeron los requerimientos del diseño (que de cualquier forma debí contemplar al considerar diferentes opciones), pero aún así se debió interactuar con el cuerpo humano y lograr una interfaz que cumpliera el objetivo, de forma no invasiva y sin dañar al organismo.

Los objetivos planteados inicialmente para este trabajo fueron superadas. En un principio se había considerado el diseño de un dispositivo enfocado principalmente al apoyo de deportistas que se dedicaran al ciclismo y que no contaran con acceso a instalaciones adecuadas ni con un entrenador. Al avanzar en el desarrollo del trabajo se observó que era necesario incluir otros

aspectos que son también de interés para cualquier persona que practique el ciclismo aunque únicamente lo haga como actividad recreativa. Los riesgos presentados en este trabajo no son exclusivos de los deportistas, y es más probable que se presenten en un ciclista ocasional que no tiene una buena condición física que en uno que practica periódicamente.

Otra aportación de este trabajo fueron opciones adicionales que no están presentes en otros dispositivos disponibles en el mercado, y que son de bastante utilidad para el entrenamiento, como la posibilidad de establecer límites de distancia, tiempo y velocidad, y que el dispositivo avise al llegar a ellos. Además de que permite guardar los valores acumulados cuando lo desee el usuario, y no de forma automática.

Una conclusión que me deja el presente trabajo es que el costo del desarrollo de ingeniería tiene mucho que ver con los conocimientos, los procedimientos y el tiempo invertidos en el diseño de una solución. Este aspecto me hizo ver la importancia que tiene el saber valorar mi trabajo. Mientras mas experiencia y conocimientos tenga uno, se pueden hacer las cosas de forma eficiente y en menos tiempo, y si esto permite reducir el tiempo, los costos y los errores durante el proceso de diseño, entonces debe ser recompensado adecuadamente.

Pienso que el dispositivo desarrollado tendría un precio cercano a los $500 pesos, ubicado entre las opciones disponibles que únicamente cuentan con crono-odómetro, y los dispositivos que incluyen medición de la frecuencia cardiaca. Los primeros cuestan alrededor de $200 pesos y aunque son mas baratos, el sistema desarrollado tiene otras características adicionales que justificarían su compra. Los otros pasan de los mil pesos, además de que no incluyen opciones como la medición de temperatura. Este precio debe incluir tanto el costo de fabricación como el de desarrollo, de manera que con la venta de varias unidades se recupere la inversión y se comiencen a obtener ganancias.

A futuro puedo distinguir básicamente dos líneas de mejora al dispositivo. La primera en cuanto al diseño específico de elementos como la pantalla LCD o el gabinete, y por otro lado está la posibilidad de tener un solo circuito integrado componentes como el sensor de pulso. En ambos casos se reduciría el tamaño del dispositivo al optimizarse el espacio disponible. Esto representaría un costo importante que solo se justificaría con la fabricación de un número elevado de unidades y requeriría de especialistas en otras disciplinas, como diseñadores industriales. Esto me permite ver que el desarrollo de tecnología nacional no es únicamente responsabilidad de los Ingenieros, pero no por eso debemos rehuir a nuestra responsabilidad social con México.

Quiero resaltar que el beneficio de este trabajo no debe ser únicamente para mí. Es una síntesis de muchos de los conocimientos que adquirí a lo largo de mi carrera, por lo que me gustaría que en un futuro sirviera a otros estudiantes, que lo pudieran retomar y mejorar, en beneficio suyo pero principalmente de nuestra Universidad.

POR MI RAZA HABLARÁ EL ESPÍRITU
Junio 2004

ÍGOR CLAVEL HERRERA
igor@lamochasoft.com

Apéndice A

Código Fuente

A.1. Archivo *proyecto.asm*

```
                LIST    P=16F877
                __config 3D71h                  ; Bits de configuración
                RADIX   HEX
                INCLUDE "P16F877.INC"
                INCLUDE "asig_mem.asm"
                ORG     00h                     ; Vector de reset
                goto    INICIO
                ORG     04h                     ; Rutina de manejo de interrupción:
                INCLUDE "interrupcion.asm"
INICIO          ;                               ; Comienza en la próxima línea
                INCLUDE "conf_recur.asm"
                INCLUDE "ini_var.asm"
                INCLUDE "op_normal.asm"         ; Está en modo normal
                INCLUDE "modo_conf.asm"         ; Está en modo de configuración
FIN             goto    FIN                     ; Para evitar entradas a subrutinas
                INCLUDE "sr_retardo.asm"
                INCLUDE "sr_eeprom.asm"
                INCLUDE "sr_rodajuste.asm"
                INCLUDE "sr_lcd.asm"
                INCLUDE "sr_bcdascii.asm"
                INCLUDE "sr_mensaje.asm"
                END                             ; *** FIN DEL PROGRAMA PRINCIPAL
```

A.2. Archivo *asig_mem.asm*

```
                ORG     2100h           ;Inicialización de datos en EEPROM
EE_RODADA       DE      d'7'            ; Rodada
EE_FCMAX        DE      d'100'          ; Frecuencia cardiaca máxima
EE_HOR_LIM      DE      d'0'            ; Para el límite del cronómetro
EE_MIN_LIM      DE      d'0'
EE_D2_LIM       DE      d'0'            ; Para el límite de distancia
EE_D1_LIM       DE      d'0'
EE_SEG          DE      d'0'            ; Segundos almacenados
EE_MIN          DE      d'0'            ; Minutos almacenados
EE_HOR          DE      d'0'            ; Horas almacenados
EE_DIST0        DE      d'0'            ; Distancia 0 almacenada
EE_DIST1        DE      d'0'            ; Distancia 1 almacenada
EE_DIST2        DE      d'0'            ; Distancia 2 almacenada
TEMPO           EQU     h'70'           ; ----- Registros en RAM --------------
W_TEMP          EQU     h'7A'           ; Para almacenar W en interrupciones
STATUS_TEMP     EQU     h'7B'           ; Para almacenar STATUS en interrup.
PCLATH_TEMP     EQU     h'7C'           ; Para almacenar PCLATH en interrup.
ESTADO          EQU     h'7D'           ; Para controlar el estado del sistema
PUROS0          EQU     h'7E'           ; Para SKIP no condicionales
DAT_ESC_EE      EQU     h'7F'           ; Dato a escribirse en EEPROM
; Banco 0:
CONTEMP         EQU     h'20'           ; Para almacenar contadores temporales
BCD_CONT        EQU     h'21'           ; Para subrutina de LCD
BCD_2           EQU     h'22'
BCD_1           EQU     h'23'
BCD_0           EQU     h'24'
```

```
SEG             EQU     h'25'           ; Para el cronómetro
MIN             EQU     h'26'
HOR             EQU     h'27'
DIST0           EQU     h'28'           ; Metros y decenas de metros
DIST1           EQU     h'29'           ; Kilómetros y centenas de metros
DIST2           EQU     h'2A'           ; Centenas y decenas de kilómetros
CTTEV1          EQU     h'2B'           ; Constante de velocidad (entero)
CTTEV0          EQU     h'2C'           ; Constante de velocidad (fracción)
CTTED1          EQU     h'2D'           ; Constante de distancia (cientos de m.)
CTTED0          EQU     h'2E'           ; Constante de distancia (0 a 99 m.)
MIN_LIM         EQU     h'2F'           ; Para el límite del cronómetro
HOR_LIM         EQU     h'30'
D1_LIM          EQU     h'31'           ; Para el límite de distancia
D2_LIM          EQU     h'32'
VEL_LIM         EQU     h'33'           ; Velocidad límite
VEL_ACT         EQU     h'34'           ; Velocidad actual
VEL_PAR         EQU     h'35'           ; Parcial de la velocidad actual
PULSO           EQU     h'36'           ; Lectura actual del pulso
FCMAX           EQU     h'37'           ; Frecuencia cardiaca máxima
TEMP1           EQU     h'38'           ; Temperatura 1 por canal 0
TEMP2           EQU     h'39'           ; Temperatura 2 por canal 1
SEGSTEMP        EQU     h'3A'           ; Contador de seg. entre conv. de temp.
ILUMIN          EQU     h'3B'           ; Lectura de iluminación por canal 3
VUELDIST        EQU     h'3C'           ; Contador de las 50 vueltas para dist.
VUELVEL         EQU     h'3D'           ; Contador de las vueltas para velocidad
SEGSVEL         EQU     h'3E'           ; Contador de seg. para velocidad
SEGSPUL         EQU     h'3F'           ; Contador de segundos para pulso
SEP_REL         EQU     h'40'           ; Separador del reloj (":" o " ")
LIMITES         EQU     h'41'           ; Para indicar qué límites estan activos
ALARMAS         EQU     h'42'           ; Indicar la fuente de las alarmas
CARLIM          EQU     h'43'           ; Caracter indicador de limites activos
ILUM_NIV        EQU     h'44'           ; Nivel de iluminación
MODIF           EQU     h'45'           ; Para indicar q valor se está modif.
RODADA          EQU     h'46'           ; Rodada (2-7)
BASURA          EQU     h'60'           ; Destino final del recorrido
PUL_V15         EQU     h'61'           ; Valor parcial del pulso más viejo
PUL_V14         EQU     h'62'           ; Valor parcial del pulso
PUL_V13         EQU     h'63'           ; Valor parcial del pulso
PUL_V12         EQU     h'64'           ; Valor parcial del pulso
PUL_V11         EQU     h'65'           ; Valor parcial del pulso
PUL_V10         EQU     h'66'           ; Valor parcial del pulso
PUL_V9          EQU     h'67'           ; Valor parcial del pulso
PUL_V8          EQU     h'68'           ; Valor parcial del pulso
PUL_V7          EQU     h'69'           ; Valor parcial del pulso
PUL_V6          EQU     h'6A'           ; Valor parcial del pulso
PUL_V5          EQU     h'6B'           ; Valor parcial del pulso
PUL_V4          EQU     h'6C'           ; Valor parcial del pulso
PUL_V3          EQU     h'6D'           ; Valor parcial del pulso
PUL_V2          EQU     h'6E'           ; Valor parcial del pulso
PUL_V1          EQU     h'6F'           ; Valor parcial del pulso
; registro ESTADO
CRONO           EQU     h'00'           ; 0: Cronómetro detenido 1: Andando
ARRANQ          EQU     h'01'           ; 0: Nada 1: Se solicita arrancar
CONV_T          EQU     h'02'           ; 0: No convertir temperaturas 1: Sí
ACT_LV          EQU     h'03'           ; Activación del límite de velocidad
SOLT_BOT        EQU     h'04'           ; 0: Nada 1: Se espera que se suelten
DAT_MOD         EQU     h'05'           ; 1: Se modificó algún dato
; LIMITES
LIMTI           EQU     h'00'           ; Indica que se activa el lim. de tiempo
LIMDI           EQU     h'01'           ; Indica que se activa el lim. de dist.
; MODIF
M_T             EQU     h'00'           ; Indica que se está modificando tiemp.
M_D             EQU     h'01'           ; Indica que se está modificando dist.
M_L             EQU     h'02'           ; Indica que se está modificando lim.
M_R             EQU     h'03'           ; Indica que se está modificando rodada
```

```
M_F             EQU     h'04'           ; Indica que se está modificando FCM
; Alarmas
ALVEL           EQU     h'00'           ; Alarma por límite de velocidad
ALPUL           EQU     h'01'           ; Alarma por pulso
; Control del LCD (Puerto E)
EN              EQU     h'02'
RW              EQU     h'01'
RS              EQU     h'00'
; Del puerto B
GIRO            EQU     h'00'           ; Entrada del sensor de giro
ZUMB            EQU     h'01'           ; Salida al zumbador
ACTIV           EQU     h'02'           ; Salida de activación de la iluminación
BOTON3          EQU     h'05'           ; Entrada botón 3
BOTON2          EQU     h'06'           ; Entrada botón 2
BOTON1          EQU     h'07'           ; Entrada botón 1
; -- Constantes:
HOR_LIM_MAX     EQU     d'4'
D2_LIM_MAX      EQU     d'30'
FCMAX_MAX       EQU     d'211'
FCMAX_MIN       EQU     d'100'
FCMIN           EQU     d'30'
ILUM_U1         EQU     d'30'           ; Primer umbral entre niveles 0 y 1
ILUM_U2         EQU     d'90'           ; Segundo umbral entre niveles 1 y 2
ILUM_U3         EQU     d'190'          ; Tercer umbral entre niveles 2 y 3
```

A.3. Archivo *interrupcion.asm*

```
;*** Rutina de atención a interrupciones ************************************
INI_INT         movwf   W_TEMP          ; Copia W a registro temporal
                swapf   STATUS,W        ; Salva STATUS en W invertiendo nibbles
                clrf    STATUS          ; Cambia a banco 0 sin importar donde
                                        ; estaba antes de la interrupción
                movwf   STATUS_TEMP     ; Salva STATUS en su registro temporal
                movf    PCLATH,W        ; Salva PCLATH (solo si se usan páginas
                                        ; 1, 2 y 3) en W
                movwf   PCLATH_TEMP     ; Para guardarlo luego en su reg. tempo.
                clrf    PCLATH          ; Cambia a página 0 sin importar en cual
                                        ; estaba antes de la interrupción.
; 0) Verificar la fuente de la interrupción
INT_FUENTE      btfsc   PIR1,TMR1IF     ; Ve si la interrupción es del Timer 1
                goto    INT_TMR1        ; Atiende la interrupción del Timer 1
                btfsc   INTCON,INTF     ; Ve si la interrupción es de int. ext.
                goto    INC_DIST        ; Atiende la interrupción externa
                goto    FIN_INT         ; En cualquier otro caso sale
; 1) Interrupción del TIMER 1 que sucede cada segundo
        ; 1.0) Verificar permiso de conversión de temperaturas:
INT_TMR1        bsf     TMR1H,7         ; Pone a 1 bit 7 para mitad rango
                decfsz  SEGSTEMP,F      ; Decrementa y brinca si es cero
                goto    INC_CRONO       ; No es cero y aún no debe permitir
                movlw   d'10'           ; Pone 10 segundos
                movwf   SEGSTEMP        ; para el retardo entre conversiones
                bsf     ESTADO,CONV_T   ; Permite conv. de temp.
        ; 1.1) Incremento al cronómetro. Llega hasta 9:59:59 y de ahí a 0:00:00
INC_CRONO       btfss   ESTADO,CRONO    ; ve si el cronómetro esta andando
                goto    CALC_VEL        ; Salta a (1.2) si no está andando
                movlw   d'60'           ; Carga 60d en W para comparaciones
                incf    SEG,F           ; Incrementa segundos en 1
                subwf   SEG,F           ; Resta para ver si ya llegó a 60 segs.
                btfsc   STATUS,Z        ; Ve si la resta dio 0 y llegó a 60
                goto    INC_MIN         ; Si es 0, incrementa minutos
                addwf   SEG,F           ; No es 0, restaura valor
                goto    CALC_VEL        ; Salta al siguiente paso
INC_MIN         incf    MIN,F           ; Si es cero, incrementa minutos en 1
                subwf   MIN,F           ; Resta para ver si ya llegó a 60 min.
```

```
                btfsc   STATUS,Z        ; Ve si la resta dio 0 y llegó a 60
                goto    INC_HOR         ; Si es 0, incrementa horas
                addwf   MIN,F           ; No es 0, restaura valor
                goto    CALC_VEL        ; Salta al siguiente paso
INC_HOR         movlw   d'10'           ; Es 0, Carga 10d en W para comparación
                incf    HOR,F           ; incrementa horas en 1
                subwf   HOR,F           ; Resta para ver si ya llegó a 10 horas
                btfss   STATUS,Z        ; Ve si la resta dio 0 y llegó a 10
                addwf   HOR,F           ; No llegó, restaura valor
        ; 1.2) Calcular velocidad
CALC_VEL        decfsz  SEGSVEL,F       ; Decrementa la cuenta de segundos
                goto    PUL_MED         ; No ha llegado, salta el final
                movlw   d'03'           ; Reinicializa la cta de los 3 segundos
                movwf   SEGSVEL         ; a las que se hace calc. de la vel.
                clrf    VEL_ACT         ; Pone a ceros la velocidad actual
                clrf    VEL_PAR         ; Pone a ceros el parcial de vel. actual
                incf    VUELVEL,F       ; Incrementa 1 vta. para entrar al ciclo
VEL_VUELT       decfsz  VUELVEL,F       ; Decrementa para contar cuantas vueltas
                goto    VEL_OTRA        ; Aún no acaba y calcula otra vuelta
                goto    VEL_REDOND      ; Ya no hay vueltas y va al final
VEL_OTRA        movf    CTTEV0,W        ; Toma la constante del parcial
                addwf   VEL_PAR,F       ; La suma al parcial de la vel. actual
                movlw   d'100'          ; Pone 100 en W para ver si hay 1 km/h
                subwf   VEL_PAR,F       ; Resta los 100
                btfsc   STATUS,C        ; Ve si se pasó la centena
                goto    VEL_A           ; Sí se ha pasado y se sumará 1 km/h
                addwf   VEL_PAR,F       ; Reestablece el valor anterior
                btfsc   PUROS0,0        ; Salta la siguiente línea
VEL_A           incf    VEL_ACT,F       ; Hay un nuevo km/h
                movf    CTTEV1,W        ; Toma la constante de km/h
                addwf   VEL_ACT,F       ; La suma a la velocidad actual
                goto    VEL_VUELT       ; Continúa viendo si hay más vueltas
VEL_REDOND      movlw   d'50'           ; Pone 50 en W para ver si hay redondeo
                subwf   VEL_PAR,W       ; Resta los 50
                btfsc   STATUS,C        ; Ve si se pasó de medio km/h
                incf    VEL_ACT,F       ; Sí se ha pasado y se sumará 1 km/h
        ; 1.3) Calcular pulso
PUL_MED         movf    TMR0,W          ; Pasa lectura actual a W
                clrf    TMR0            ; Borra el acumulador de pulso
                movwf   PUL_V1          ; Guarda como valor actual
                movlw   PUL_V15         ; Dirección de inicio de los datos
                movwf   FSR             ; Almacenar en el apuntador
                bcf     STATUS,IRP      ; Borra el bit IRP
                clrf    PULSO           ; Borra el valor actual del pulso
PUL_SUMA        movf    INDF,W          ; Pasa el valor apuntado a W
                decf    FSR,F           ; Decrementa el apuntador
                movwf   INDF            ; Recorre a la localidad anterior
                addwf   PULSO,F         ; Suma y deja en PULSO
                incf    FSR,F           ; Incrementa el apuntador
                incf    FSR,F           ; Incrementa el apuntador
                btfss   FSR,4           ; Si es 1 deja del ciclo (llegó a 70h)
                goto    PUL_SUMA        ; Sigue con el ciclo
                movf    SEGSPUL,F       ; Para activar banderas
                btfsc   STATUS,Z        ; Brinca si no está en ceros
                goto    PUL_MULT        ; Está en ceros por lo que multiplica
                decfsz  SEGSPUL,F       ; Decrementa los segundos y brinca si 0
                goto    PUL_NOVAL       ; No hay valor válido aún
PUL_MULT        bcf     STATUS,C        ; Prepara corrimiento
                rlf     PULSO,F         ; Multiplica por 2
                bcf     STATUS,C        ; Prepara corrimiento
                rlf     PULSO,F         ; Multiplica por 2
                btfss   STATUS,C        ; Ve si hay acarreo
                goto    PUL_NOAC        ; No hay avarreo
                movlw   h'FF'           ; Hay acarreo
                movwf   PULSO           ; Valor máximo en PULSO
```

```
PUL_NOAC        decf    PULSO,F         ; Para saltar al final
                goto    PUL_FIN         ; Acaba
PUL_NOVAL       movlw   FCMIN           ; Carga la frec. min.
                movwf   PULSO           ; Y sobreescribe la suma
PUL_FIN         incf    PULSO,F         ; Incrementa en 1
FIN_TMR1        bcf     PIR1,TMR1IF     ; Borra la bandera de int. del Timer 1
                goto    INT_FUENTE      ; Va a ver si hay más interrupciones
        ; 2.1) Incremento de la distancia
INC_DIST        btfss   ESTADO,CRONO    ; ve si el cronómetro esta andando
                goto    INC_VEL         ; Salta a (2.2)  si no está andando
DISTANCIA       decfsz  VUELDIST,F      ; Decrementa la cuenta de vueltas
                goto    INC_VEL         ; No ha llegado, salta el inc. de dist.
DISTA_C         movlw   d'50'           ; Reinicializa la cta de las 50 vueltas
                movwf   VUELDIST        ; a las que se hace incremento de dist.
                movf    CTTED0,W        ; Toma la constante de unidades y dec.
                addwf   DIST0,F         ; La suma a la distancia actual en D0
                movlw   d'100'          ; Pone 100 en W para ver si hay centena
                subwf   DIST0,F         ; Resta los 100
                btfsc   STATUS,C        ; Ve si se pasó la decena
                goto    DISTA_A         ; Sí se ha pasado y se sumará 1 centena
                addwf   DIST0,F         ; Reestablece el valor anterior
                btfsc   PUROS0,0        ; Salta la siguiente línea
DISTA_A         incf    DIST1,F         ; Hay una nueva centena de metros
                movf    CTTED1,W        ; Toma la constante de centenas y miles
                addwf   DIST1,F         ; La suma a la distancia actual en D1
                movlw   d'100'          ; 100 en W para ver si hay decenas de km
                subwf   DIST1,F         ; Resta los 100
                btfsc   STATUS,C        ; Ve si se pasó la decena
                goto    DISTA_B         ; Sí se ha pasado y se sumarán 10 km
                addwf   DIST1,F         ; Reestablece el valor anterior
                btfsc   PUROS0,0        ; Salta la siguiente línea
DISTA_B         incf    DIST2,F         ; Hay una nueva decena de km
                movlw   d'100'          ; 100 en W para ver si hay miles de km
                subwf   DIST2,F         ; Resta los 100
                btfss   STATUS,C        ; Ve si se pasó
                addwf   DIST2,F         ; No se ha pasado y se reestablece
        ; 2.2) Incremento de las vueltas para calcular velocidad
INC_VEL         incf    VUELVEL,F       ; Agrega una vuelta más para cal. vel.
        ; 2.3) Arrancar el cronómetro si así se requiere
ARRANCAR        btfss   ESTADO,ARRANQ   ; Ve si es necesario arrancar el sistema
                goto    FIN_INTEXT      ; No es necesario y continúa
                bcf     ESTADO,ARRANQ   ; Borra la solicitud de arranque
                bsf     ESTADO,CRONO    ; Arranca el cronómetro
FIN_INTEXT      bcf     INTCON,INTF     ; Borra la bandera de int. externa
                goto    INT_FUENTE      ; Va a ver si hay más interrupciones
FIN_INT         movf    PCLATH_TEMP,W   ; Carga en W PCLATH temporal
                movwf   PCLATH          ; para restaurarlo
                swapf   STATUS_TEMP,W   ; Carga STATUS temporal en W
                movwf   STATUS          ; y reestablece STATUS anterior
                swapf   W_TEMP,F        ; Invierte nibbles de W temp. ahí mismo
                swapf   W_TEMP,W        ; y los reinvierte pero en W
                retfie                  ; y  regresa al programa principal
```

A.4. Archivo *conf_recur.asm*

```
; 1) Deshabilitar interrupciones.
                bsf     STATUS,RP0      ; Cambia al banco 1
                bcf     STATUS,RP1
                movlw   b'01010000'     ; Deshabilita el permiso global de int.
                movwf   INTCON          ; Habilita las int. extras e INT/RB0
                movlw   b'00000001'     ; Solo permite la interrupción extra
                movwf   PIE1            ; del TMR1
                clrf    PIE2            ; y las otras también las prohibe
; 2) Configurara el TMR2 para usarse en retardos
```

```
                bcf     STATUS,RP0      ; Cambia a banco 0
                movlw   b'01111011'     ; Activa ambos divisores a 16
                movwf   T2CON           ; y apaga TMR2
; 3) Configurar puerto A, conversor A/D, TMR0 e INT
                bsf     STATUS,RP0      ; Cambia al banco 1
                movlw   b'00011011'     ; Pines 0,1,3 y 4 son entradas
                movwf   TRISA
                movlw   b'10000100'     ; Pines 0,1,3 analógicos y el res. del
                movwf   ADCON1          ; convertidor se ajusta a la derecha
                movlw   b'10101000'     ; Se configura OPTION para flancos asc.
                movwf   OPTION_REG      ; en TMR0 y desc. en INT ext. Div al WDG
                bcf     STATUS,RP0      ; Cambia a banco 0
                movlw   b'10000000'     ; Se establece tiempo para conversión y
                movwf   ADCON0          ; se deja apagado el C A/D en el canal 0
                clrf    PORTA           ; Borra el puerto
                clrf    TMR0            ; Borra el Timer 0
; 4) Configurar TMR1
                movlw   b'00001110'     ; Con predivisor en 1 y reloj externo,
                movwf   T1CON           ; no se sincroniza y se deja apagado
; 5) Configurar puertos C, B, D y E
                bsf     STATUS,RP0      ; Cambia a banco 1
                movlw   b'11100001'     ; Bits 7,6 y 5 entradas. Bit 0 para la
                movwf   TRISB           ; interrupción externa y las demás sal.
                clrf    TRISC           ; Todo el puerto C es de salida
                clrf    TRISD           ; Todo el puerto D es de salida
                clrf    TRISE           ; Todo el puerto E es de salida
                bcf     STATUS,RP0      ; Cambia a banco 0
                clrf    PORTB           ; Borra los puertos
                clrf    PORTC
                clrf    PORTD
                clrf    PORTE
; 6) Retardo inicial
                movlw   h'FF'           ;cuenta máxima
                call    RETARDO         ;ejecuta el retardo
```

A.5. Archivo *ini_var.asm*

```
                call    LCD_INI         ; Inicializa el LCD
                clrf    PUROS0          ; Pone ceros para SKIPs no condicionales
                ; Verificar si hay reset por bajo voltaje
                bsf     STATUS,RP0      ; Cambia al banco 1
                btfsc   PCON,NOT_BOR    ; Salta si no está en cero
                goto    CONF_NORMAL     ; No hay problema con la batería
                btfss   PCON,NOT_POR    ; Salta si no está en uno
                goto    CONF_NORMAL     ; No hay problema con la batería
                call    MENSAJE_B       ; Mensaje para avisar baja batería
                clrf    INTCON          ; Deshabilita interrupciones
                sleep                   ; Operación de bajo consumo
                ; Operación con buen voltaje
CONF_NORMAL     bsf     PCON,NOT_POR    ; A 1 el bit /POR
                bsf     PCON,NOT_BOR    ; A 1 el bit /BOR
                bcf     STATUS,RP0      ; Cambia al banco 0
                clrf    ESTADO          ; Borra banderas de estado del sistema
                movlw   d'50'           ; Inicializa la cuenta de las 50 vueltas
                movwf   VUELDIST        ;
                clrf    VEL_ACT         ; Pone a ceros la velocidad actual
                clrf    VEL_PAR         ; Pone a ceros el parcial de vel. actual
                movlw   d'10'           ; Inicializa velocidad límite a 10 km/h.
                movwf   VEL_LIM         ;
                clrf    VUELVEL         ; Pone a ceros las vueltas para la vel.
                movlw   d'03'           ; Pone 3 para los seg. de la velocidad
                movwf   SEGSVEL         ;
                clrf    PULSO           ; Pone a ceros el pulso actual
                movlw   d'15'           ; Pone 15 para los seg. del pulso
```

```
                movwf   SEGSPUL         ;
                clrf    TEMP1           ; Pone a ceros la temperatura 1
                clrf    TEMP2           ; Pone a ceros la temperatura 2
                clrf    ILUMIN          ; Pone a ceros la iluminación
                movlw   '_'             ; Inicializa indicador nivel de ilum.
                movwf   ILUM_NIV        ;
                bsf     ESTADO,CONV_T   ; Convertir desde el inicio
                movlw   d'01'           ; Pone 1 segundo
                movwf   SEGSTEMP        ; para el retardo entre conversiones
                clrf    LIMITES         ; Desactiva los límites
                movlw   ' '             ; Pone espacio en caracter límites
                movwf   CARLIM          ;
                call    EE_LECT_DAT     ; Lee y guarda en RAM lect. almacenadas
                call    EE_LECT_PAR     ; Lee y guarda en RAM param. almacenados
                call    ROD_AJUSTE      ; Ajusta los valores de las constantes
                bsf     T1CON,TMR1ON    ; Enciende el Timer 1
```

A.6. *Archivo op_normal.asm*

```
NORMAL          clrf    TMR1L           ; Borra parte baja del timer 1
                clrf    TMR1H           ; Borra parte alta del timer 1
                bsf     TMR1H,7         ; Pone a 1 bit 7 para mitad de rango
                ; Inicializa la medición del pulso
                movlw   d'14'           ; Quince segundos para llenar los datos
                movwf   SEGSPUL         ; Almacenar en el registro del contador
                movlw   PUL_V15         ; Dirección de inicio de los datos
                movwf   FSR             ; Almacenar en el apuntador
                bcf     STATUS,IRP      ; Borra el bit IRP
PUL_BORRAR      clrf    INDF            ; Borra el valor apuntado
                incf    FSR,F           ; Incrementa el apuntador
                btfss   FSR,4           ; Si es 1 deja del ciclo (llegó a 70h)
                goto    PUL_BORRAR      ; Sigue con el ciclo
                bsf     INTCON,GIE      ; Da permiso global a interrupciones
                ; Conversión de la temperatura 1:
BUCLE_NORM      btfss   ESTADO,CONV_T   ; Ve si se permiten conv. de temp.
                goto    NORM_INTLUM     ; No se harán conversiones de temp.
                bcf     ESTADO,CONV_T   ; Borra pues ya se harán las conv.
                bcf     ADCON0,CHS2     ; Selecciona canal 0 del convertidor
                bcf     ADCON0,CHS1     ; para convertir temperatura 1
                bcf     ADCON0,CHS0     ;
                bsf     ADCON0,ADON     ; Enciende el convertidor
                movlw   d'9'            ; Para un retardo de 5ms
                call    RETARDO         ; que asegure la conversión
                bsf     ADCON0,GO_DONE  ; Inicia la conversión
N_T1ESP         btfsc   ADCON0,GO_DONE  ; Espera a que GO_DONE se ponga en 0
                goto    N_T1ESP         ; No se ha puesto y sigue en espera
                bcf     ADCON0,ADON     ; Apaga el convertidor
                bsf     STATUS,RP0      ; Cambia al banco 1
                movf    ADRESL,W        ; Pasa a W el resultado (8 bits - sig.)
                bcf     STATUS,RP0      ; Cambia al banco 0
                movwf   TEMP1           ; Pasa la lectura a su registro
                bcf     STATUS,C        ; Limpia el carry para rotación
                rrf     TEMP1,F         ; Pasa el bit - sig. al carry
                btfsc   STATUS,C        ; Ve si el bit rotado es 1, salta si 0
                incf    TEMP1,F         ; Incrementa en uno para redondear
                ; Conversión de la temperatura 2:
                movlw   d'9'            ; Para un retardo de 5ms
                call    RETARDO         ; antes de iniciar sig. conversión
                bsf     ADCON0,CHS0     ; Selecciona canal 1 del convertidor
                bsf     ADCON0,ADON     ; Enciende el convertidor
                movlw   d'9'            ; Para un retardo de 5ms
                call    RETARDO         ; que asegure la conversión
                bsf     ADCON0,GO_DONE  ; Inicia la conversión
N_T2ESP         btfsc   ADCON0,GO_DONE  ; Espera a que GO_DONE se ponga en 0
```

```
                goto    N_T2ESP         ; No se ha puesto y sigue en espera
                bcf     ADCON0,ADON     ; Apaga el convertidor
                bsf     STATUS,RP0      ; Cambia al banco 1
                movf    ADRESL,W        ; Pasa a W el resultado (8 bits - sig.)
                bcf     STATUS,RP0      ; Cambia al banco 0
                movwf   TEMP2           ; Pasa la lectura a su registro
                bcf     STATUS,C        ; Limpia el carry para rotación
                rrf     TEMP2,F         ; Pasa el bit - sig. al carry
                btfsc   STATUS,C        ; Ve si el bit rotado es 1, salta si 0
                incf    TEMP2,F         ; Incrementa en uno para redondear
                ; Conversión de intensidad luminosa:
NORM_INTLUM     bsf     ADCON0,CHS1     ; Selecciona canal 3 del convertidor
                bsf     STATUS,RP0      ; Cambia al banco 1
                bcf     ADCON1,ADFM     ; Coloca el resultado en ADRESH
                bcf     STATUS,RP0      ; Cambia al banco 0
                bsf     ADCON0,ADON     ; Enciende el convertidor
                movlw   d'9'            ; Para un retardo de 5ms
                call    RETARDO         ; que asegure la conversión
                bsf     ADCON0,GO_DONE  ; Inicia la conversión
N_ILESP         btfsc   ADCON0,GO_DONE  ; Espera a que GO_DONE se ponga en 0
                goto    N_ILESP         ; No se ha puesto y sigue en espera
                bcf     ADCON0,ADON     ; Apaga el convertidor
                bsf     STATUS,RP0      ; Cambia al banco 1
                bsf     ADCON1,ADFM     ; Coloca el resultado en ADRESL
                bcf     STATUS,RP0      ; Cambia al banco 0
                movf    ADRESH,W        ; Pasa a W el resultado (8 bits + sig.)
                movwf   ILUMIN          ; Pasa la lectura a su registro
                movlw   ILUM_U3         ; Tercer umbral entre niveles 2 y 3
                subwf   ILUMIN,W        ; Compara para ver si F>=W
                btfsc   STATUS,C        ; Si F>=W carry será 1
                goto    N_IL_N3         ; Nivel 3 de iluminación
                movlw   ILUM_U2         ; Segundo umbral entre niveles 1 y 2
                subwf   ILUMIN,W        ; Compara para ver si F>=W
                btfsc   STATUS,C        ; Si F>=W carry será 1
                goto    N_IL_N2         ; Nivel 2 de iluminación
                movlw   ILUM_U1         ; Primer umbral entre niveles 0 y 1
                subwf   ILUMIN,W        ; Compara para ver si F>=W
                btfsc   STATUS,C        ; Si F>=W carry será 1
                goto    N_IL_N1         ; Nivel 1 de iluminación
N_IL_N0         bcf     PORTB,ACTIV     ; Apaga indicador
                btfsc   TMR1H,4         ; Bit 4 del TMR1 esta en 0 durante 1/8 seg
                bsf     PORTB,ACTIV     ; Enciende indicador
                movlw   ' '             ; Indicador de nivel 0
                goto    N_IL_FIN        ; Y va a guardar
N_IL_N1         bcf     PORTB,ACTIV     ; Apaga indicador
                btfsc   TMR1H,6         ; Bit 6 del TMR1 esta en 0 durante 1/2 seg
                bsf     PORTB,ACTIV     ; Enciende indicador
                movlw   '_'             ; Indicador de nivel 1
                goto    N_IL_FIN        ; Y va a guardar
N_IL_N2         bcf     PORTB,ACTIV     ; Apaga indicador
                movlw   '-'             ; Indicador de nivel 2
                goto    N_IL_FIN        ; Y va a guardar
N_IL_N3         bcf     PORTB,ACTIV     ; Apaga indicador
                movlw   '='             ; Indicador de nivel 2
N_IL_FIN        movwf   ILUM_NIV        ; Guarda en el indicador
                ;Alarmas activas e inactivas
NORM_ALVEL      bcf     ALARMAS,ALVEL   ; Borra la alarma por velocidad
                btfss   ESTADO,ACT_LV   ; Ve si esta activado el límite de vel.
                goto    NORM_ALPUL      ; No esta activado y no hace nada
                movf    VEL_LIM,W       ; Carga velocidad límite en W
                subwf   VEL_ACT,W       ; Resta vel. lim. a la velocidad actual
                btfss   STATUS,C        ; Ve si esta en 1 o 0, salta en 1
                goto    NORM_ALPUL      ; No esta activado y no hace nada
                bsf     ALARMAS,ALVEL   ; Activa la alarma por velocidad
NORM_ALPUL      bcf     ALARMAS,ALPUL   ; Borra la alarma por pulso
```

```
                movf    FCMAX,W         ; Carga frec. cardiaca máxima en W
                subwf   PULSO,W         ; Resta al pulso actual
                btfss   STATUS,C        ; Ve si esta en 1 o 0, salta en 1
                btfsc   PUROS0,0        ; Salta siguiente línea
                bsf     ALARMAS,ALPUL   ; Activa alarma por pulso alto
                ; Activación del zumbador si hay alarmas
NORM_ZUMB       bcf     PORTB,ZUMB      ; Apaga el zumbador
                btfsc   ALARMAS,ALPUL   ; Esta activa la alarma del pulso
                goto    NORM_Z_PUL      ; Activa el zumbador para pulso
                btfsc   ALARMAS,ALVEL   ; Esta activa la alarma de velocidad
                goto    NORM_Z_VEL      ; Activa el zumbador para velocidad
                goto    NORM_SEP_REL    ; No activa ninguna
NORM_Z_PUL      btfsc   TMR1H,4         ; Bit 4 de TMR1 esta en 1 durante 1/8 seg
                bsf     PORTB,ZUMB      ; Zumbido cada 1/8 seg.
                goto    NORM_SEP_REL    ; Salta al final
NORM_Z_VEL      btfsc   TMR1H,6         ; Bit 6 de TMR1 esta en 1 durante 1/2 seg
                bsf     PORTB,ZUMB      ; Zumbido cada 1/2 seg.
                ; Caracteres para indicadores
NORM_SEP_REL    movlw   ':'             ; Dos puntos en W
                btfss   ESTADO,ARRANQ   ; Ve si está esperando arranque
                goto    NORM_SEP_FIN    ; No está esperando así que fija ':'
                btfsc   TMR1H,6         ; Bit 6 de TMR1 esta en 1 durante 1/2 seg
                movlw   ' '             ; Espacio cada 1/2 seg.
NORM_SEP_FIN    movwf   SEP_REL         ; Guarda el indicador de separación
                movlw   ' '             ; Pone espacio en blanco
NORM_IND_LIM    btfsc   LIMITES,LIMTI   ; Salta si lim. tiempo inactivo
                goto    N_I_L_T0        ; Va a ver si esta activo lim dist.
                btfsc   LIMITES,LIMDI   ; Salta si lim. dist inactivo
                movlw   'd'             ; Pone indicador de distancia activa
                goto    NORM_IND_LIMF   ; Va al final pues ambos estan en 0
N_I_L_T0        movlw   't'             ; Pone indicador de tiempo activo
                btfsc   LIMITES,LIMDI   ; Salta si lim. dist inactivo
                movlw   'a'             ; Pone indicador de tpo. Y dist. activos
NORM_IND_LIMF   movwf   CARLIM          ; Guarda caracter indicador de límites
                ;1) Primera línea del LCD
NORM_LCD_L1     movlw   h'00'           ; Coloca 00h en W para
                call    LCD_POS         ; colocar el LCD en home
                ; Reloj
                movlw   b'00000010'     ; Se enviara 1 dígito solamente
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                movf    HOR,W           ; carga horas en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movf    SEP_REL,W       ; Carga separador en W
                call    LCD_TEXTO       ; y lo imprime en LCD
                clrf    BCD_CONT        ; Se enviarán 2 dígitos sin espacio
                movf    MIN,W           ; carga minutos en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movf    SEP_REL,W       ; Carga separador en W
                call    LCD_TEXTO       ; y lo imprime en LCD
                movf    SEG,W           ; carga segundos en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movf    CARLIM,W        ; Carga indicador de límites en W
                call    LCD_TEXTO       ; y lo imprime en LCD
                ; Temperaturas
                movlw   ' '             ; Carga espacio en W
                call    LCD_TEXTO       ; y lo imprime en LCD
                movlw   b'00000001'     ; Se enviaran 2 dígitos con espacio
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                movf    TEMP1,W         ; Carga temperatura 1 en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movlw   ' '             ; Carga espacio en W
                call    LCD_TEXTO       ; y lo imprime en LCD
                movf    TEMP2,W         ; Carga temperatura 2 en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movlw   ' '             ; Carga espacio en W
```

```
                call    LCD_TEXTO       ; y lo imprime en LCD
                movf    ILUM_NIV,W      ; Carga nivel de iluminación en W
                call    LCD_TEXTO       ; y lo imprime en LCD
                ;2) Segunda línea del LCD
NORM_LCD_L2     movlw   h'40'           ; Coloca 40h en W para
                call    LCD_POS         ; colocar en segunda línea
                ; Distancia
                clrf    BCD_CONT        ; Se enviarán 2 dígitos sin espacio
                movf    DIST2,W         ; Carga cent. y dec. de km. en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movlw   b'00001000'     ; Se enviaran 2 dígitos con pto. decimal
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                movf    DIST1,W         ; Carga km y centenas de m. en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                ; Velocidad
                movlw   ' '             ; Carga espacio en W
                call    LCD_TEXTO       ; y lo imprime en LCD
                movlw   b'00000101'     ; Se enviaran 3 dígitos con espacios
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                movf    VEL_ACT,W       ; Carga la velocidad actual
                call    BCD_ASCII       ; Convierte y manda al LCD
                movlw   ' '             ; Carga espacio en W
                btfss   ESTADO,ACT_LV   ; Ve si esta activo el límite de vel.
                goto    NORM_LCD_0      ; No está activo
                movlw   '<'             ; Indicador de que esta activo el lim.
                btfss   ALARMAS,ALVEL   ; Ve si esta activa la alarma del lim.
                goto    NORM_LCD_0      ; No está activa la alarma
                movlw   '>'             ; Indicador de que la vel. es > q v. lim
                btfsc   TMR1H,6         ; Bit 6 de TMR1 esta en 1 durante 1/2 seg
                movlw   ' '             ; Espacio cada 1/2 seg.
NORM_LCD_0      call    LCD_TEXTO       ; y lo imprime en LCD
                movlw   b'00000001'     ; Se enviaran 2 dígitos con espacio
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                movf    VEL_LIM,W       ; Carga el lim. vel.
                call    BCD_ASCII       ; Convierte y manda al LCD
                ; Pulso actual
                movlw   h'A5'           ; Caracter sin pulso presente
                btfsc   PORTA,4         ; Verifica la entrada de pulso
                movlw   h'DB'           ; Caracter de pulso presente
                call    LCD_TEXTO       ; y lo imprime en LCD
                movf    SEGSPUL,F       ; Para ver si la carga finalizó
                btfsc   STATUS,Z        ; Ve si está en ceros
                goto    NORM_LCD_1      ; Está en 0 y se puede presentar valor
                movlw   '-'             ; No está en ceros y no se presenta val.
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   '-'             ; El guión así lo indica
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   '-'             ; Se imprimen 3 para abarcar los dígitos
                call    LCD_TEXTO       ; Imprime en LCD
                goto    NORM_LIMS       ; Sigue con lo próximo
NORM_LCD_1      movlw   b'00000101'     ; Se enviaran 3 dígitos con espacios
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                btfss   ALARMAS,ALPUL   ; Ve si se activa alarma por pulso
                goto    NORM_LCD_2      ; No está activa la alarma
                btfsc   TMR1H,5         ; Bit 5 del TMR1 esta en 0 durante 1/4 seg
                goto    NORM_LCD_2      ; No está en 0 el bit
                movlw   ' '             ; Está en cero y no se presenta val.
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ; El espacio así lo indica
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ; Se imprimen 3 para abarcar los dígitos
                call    LCD_TEXTO       ; Imprime en LCD
                goto    NORM_LIMS       ; Sigue con lo próximo
NORM_LCD_2      movf    PULSO,W         ; Carga el pulso actual
                call    BCD_ASCII       ; Convierte y manda al LCD
```

```
                ; Manejo de límites de distancia y tiempo
NORM_LIMS       btfsc   ESTADO,CRONO    ; Ve si está corriendo o parado
                goto    N_LIM_EST       ; Está corriendo
                btfsc   ESTADO,ARRANQ   ; Ve si está esperando arranque
                goto    N_LIM_EST       ; Está esperando arranque
                goto    NORM_BOT        ; Esta parado y sin esperar arranque
N_LIM_EST       btfss   LIMITES,LIMTI   ; Ve si esta activo el límite de tiempo
                goto    N_LIM_ESD       ; No esta activo y va a ver el de dist.
                movf    HOR_LIM,W       ; Pone el límite de hora en W
                subwf   HOR,W           ; Hace comparación y ve si HOR>=HOR_LIM
                btfss   STATUS,C        ; Si carry es 0 no ha pasado el límite
                goto    N_LIM_ESD       ; 0: Va a verificar lim. de distancia
                movf    MIN_LIM,W       ; Pone el límite de minutos en W
                subwf   MIN,W           ; Hace comparación y ve si MIN>=MIN_LIM
                btfss   STATUS,C        ; Si carry es 0 no ha pasado el límite
                goto    N_LIM_ESD       ; 0: Va a verificar lim. de distancia
                goto    N_LIM_STOP      ; Se han pasado ambos valores
N_LIM_ESD       btfss   LIMITES,LIMDI   ; Ve si esta activo el límite de disat.
                goto    NORM_BOT        ; No esta activo y sale
                movf    D2_LIM,W        ; Pone el límite de dist. 2 en W
                subwf   DIST2,W         ; Hace comparación y ve si DIST2>=D2_LIM
                btfss   STATUS,C        ; Si carry es 0 no ha pasado el límite
                goto    NORM_BOT        ; 0: No ha pasado y sale
                movf    D1_LIM,W        ; Pone el límite de dist. 1 en W
                subwf   DIST1,W         ; Hace comparación y ve si DIST1>=D1_LIM
                btfss   STATUS,C        ; Si carry es 0 no ha pasado el límite
                goto    NORM_BOT        ; 0: No ha pasado y sale
N_LIM_STOP      bcf     ESTADO,CRONO    ; Se pasó un límite y para el cronómetro
                bcf     ESTADO,ARRANQ   ; y una posible solicitud de arranque.
                clrf    TMR1H           ; Borra el Timer 1 para dejar 1/2
                clrf    TMR1L           ; segundo encendido el zumbador.
                bsf     TMR1H,7         ; Pone a 1 bit 7 para mitad de rango
                bsf     PORTB,ZUMB      ; Activa zumbido
N_LIM_1         btfss   TMR1H,6         ; Bit 6 de TMR1 esta en 1 durante 1/2 seg
                goto    N_LIM_1         ; Espera medio segundo
                bcf     PORTB,ZUMB      ; Desactiva zumbido
                ; Manejo de botones
NORM_BOT        btfss   ESTADO,SOLT_BOT; Ver si se espera a que se desactiven
                goto    NORM_ACTIVO     ; No hay que esperar
                movf    PORTB,W         ; Pasa el puerto donde están los botones
                andlw   b'11100000'     ; Enmascara los tres botones
                btfss   STATUS,Z        ; Ve si estan en ceros
                goto    BUCLE_NORM      ; No estan en ceros y no hace nada
                bcf     ESTADO,SOLT_BOT; Ya estan en ceros y atiende botones
NORM_ACTIVO     btfsc   PORTB,BOTON1    ; Ve si está presionado el botón 1
                goto    N_BOT1          ; Si está presionado y va a atenderlo
                btfsc   PORTB,BOTON2    ; Ve si está presionado el botón 2
                goto    N_BOT2          ; Si está presionado y va a atenderlo
                btfsc   PORTB,BOTON3    ; Ve si está presionado el botón 3
                goto    N_BOT3          ; Si está presionado y va a atenderlo
                goto    BUCLE_NORM      ; Ningún botón está presionado
                ; Atención al botón 1
N_BOT1          movlw   d'81'           ; Pide retardo de 40 ms
                call    RETARDO         ; de la rutina de retardos
                btfsc   ESTADO,CRONO    ; Ve si está corriendo o parado
                goto    N_PARAR         ; Va a parar el cronómetro
                btfsc   ESTADO,ARRANQ   ; Ve si hay una solicitud anterior
                goto    N_PARAR         ; Va a parar solicitud anterior
                bsf     ESTADO,SOLT_BOT; Esperar a que se suelte el botón
                bsf     ESTADO,ARRANQ   ; Solicita el arranque del cronometro
                goto    BUCLE_NORM      ; Fin de atención al botón 1
N_PARAR         bcf     ESTADO,ARRANQ   ; Cancela la solicitud anterior
                bcf     ESTADO,CRONO    ; Para el cronómetro
                clrf    TMR1H           ; Borra el Timer 1 para dejar 1/2
                clrf    TMR1L           ; segundo encendido el zumbador.
```

```
                bsf     TMR1H,7         ; Pone a 1 bit 7 para mitad de rango
                movlw   d'3'            ; Valor 3 en W
                movwf   CONTEMP         ; Inicializa contador temporal
N_COFF          btfss   PORTB,BOTON1    ; Espera a que se suelte el botón
                goto    BUCLE_NORM      ; Fin de atención al botón 1
                btfss   TMR1H,6         ; Bit 6 de TMR1 esta en 1 cada seg.
                goto    N_COFF          ; Sigue esperando
                bcf     TMR1H,6         ; Para volver a contar
                decfsz  CONTEMP,F       ; Ve si ya transcurrieron 3 seg.
                goto    N_COFF          ; No ha transcurrido, sigue esperando
N_GUARDAR       call    MENSAJE_G       ; Presenta mensaje de guardado
                ; Guardar distancia actual
                movf    DIST0,W         ; Pasa la dist. 0 actual
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_DIST0        ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                movf    DIST1,W         ; Pasa la dist. 1 actual
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_DIST1        ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                movf    DIST2,W         ; Pasa la dist. 2 actual
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_DIST2        ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                ; Guardar tiempo actual
                movf    SEG,W           ; Pasa los segundos actuales
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_SEG          ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                movf    MIN,W           ; Pasa los minutos acuales
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_MIN          ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                movf    HOR,W           ; Pasa las horas actuales
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_HOR          ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                bsf     ESTADO,SOLT_BOT ; Esperar a que se suelte el botón
                goto    BUCLE_NORM      ; Fin de atención al botón 1
                ; Atención al botón 2
N_BOT2          movlw   d'81'           ; Pide retardo de 40 ms
                call    RETARDO         ; de la rutina de retardos
                btfsc   ESTADO,CRONO    ; Ve si está corriendo o parado
                goto    N_VL            ; Está corriendo, mod. vel. lim
                btfsc   ESTADO,ARRANQ   ; Ve si espera arranque
                goto    N_VL            ; Está esperando, mod. vel. lim
                clrf    SEG             ; Pone a ceros los segundos
                clrf    MIN             ; Pone a ceros los minutos
                clrf    HOR             ; Pone a ceros las horas
                clrf    DIST0           ; Pone a ceros metros y decenas de m.
                clrf    DIST1           ; Pone a ceros centeas de m y km.
                clrf    DIST2           ; Pone a ceros decenas y centenas de km.
                movlw   d'50'           ; Inicializa la cuenta de las 50 vueltas
                movwf   VUELDIST        ; a las que se hace incremento de dist.
                clrf    TMR1H           ; Borra el Timer 1 para dejar 1/2
                clrf    TMR1L           ; segundo encendido el zumbador.
                bsf     TMR1H,7         ; Pone a 1 bit 7 para mitad de rango
                movlw   d'3'            ; Valor 3 en W
                movwf   CONTEMP         ; Inicializa contador temporal
N_C0            btfss   PORTB,BOTON2    ; Espera a que se suelte el botón
                goto    BUCLE_NORM      ; Fin de atención al botón 2
                btfss   TMR1H,6         ; Bit 7 de TMR1 esta en 1 durante 1/2 seg
                goto    N_C0            ; Sigue esperando
                bcf     TMR1H,6         ; Para volver a contar
                decfsz  CONTEMP,F       ; Ve si ya transcurrieron 3 seg.
```

```
                goto    N_C0            ; No ha transcurrido, sigue esperando
N_LEER          call    MENSAJE_L       ; Presenta mensaje de lectura
                call    EE_LECT_DAT     ; Lee y guarda en RAM
                bsf     ESTADO,SOLT_BOT ; Esperar a que se suelte el botón
                goto    BUCLE_NORM      ; Fin atención a botón 2
N_VL            movlw   d'05'           ; Valor de incremento del lim. de vel.
                addwf   VEL_LIM,F       ; Suma el incremento
                movlw   d'100'          ; Carga el límite del valor
                subwf   VEL_LIM,F       ; Resta para saber si se llegó
                btfsc   STATUS,C        ; Salta si no se pasó
                goto    N_VL0           ; Se ha pasado y debe ajustar val. min.
                addwf   VEL_LIM,F       ; Reestablece el valor anterior
                goto    N_VL_FIN        ; Continua
N_VL0           movlw   d'10'           ; Valor mínimo del lim. de vel.
                movwf   VEL_LIM         ; Guarda en el registro
N_VL_FIN        bsf     ESTADO,SOLT_BOT ; Espera a que se suelte el botón
                goto    BUCLE_NORM      ; Fin atención a botón 2 cuando el
                ; Atención al botón 3
N_BOT3          movlw   d'81'           ; Pide retardo de 40 ms
                call    RETARDO         ; de la rutina de retardos
                btfsc   ESTADO,CRONO    ; Ve si está corriendo o parado
                goto    N_AVL           ; Está corriendo, activa vel. lim
                btfsc   ESTADO,ARRANQ   ; Ve si espera arranque
                goto    N_AVL           ; Está esperando, activa vel. lim
                bsf     ESTADO,SOLT_BOT ; Esperar a que se suelte el botón
                goto    CONFIGU         ; Va a modo de configuración
N_AVL           btfsc   ESTADO,ACT_LV   ; Ve si esta activo
                goto    N_LVOFF         ; Esta activo y da a apagar
                bsf     ESTADO,ACT_LV   ; Activa el lim. vel.
                goto    N_LVFIN         ; Y va a esperar
N_LVOFF         bcf     ESTADO,ACT_LV   ; Desactiva el lim. vel.
N_LVFIN         bsf     ESTADO,SOLT_BOT ; Espera a que se suelte el botón
                goto    BUCLE_NORM      ; Fin atención a botón 3
```

A.7. Archivo *modo_conf.asm*

```
CONFIGU         bcf     INTCON,GIE      ; Detiene las interrupciones
                call    LCD_BORRA       ; Borra el LCD
                clrf    MODIF           ; Borra para indicar que ningún valor se
                bcf     ESTADO,DAT_MOD  ; está modificando.
                bcf     PORTB,ZUMB      ; Apaga cualquier alarma anterior
                ; Para iniciar cuenta de seguridad
CONF_TEMPO      movlw   h'0A'           ; Para cuenta de 4 segundos sin botones
                movwf   TEMPO           ; Pasa a valor temporal
CONF_CLRT1      clrf    TMR1H           ; Borra el Timer 1
                clrf    TMR1L           ; Borra el Timer 1
                ; Lo primero es verificar si ya pasó un segundo
BUCLE_CONF      btfss   TMR1H,7         ; Salta si bit 7 en 1, cada segundo
                goto    CONF_LCD_L1     ; Si sigue en 0 imprime en LCD
                decfsz  TEMPO,F         ; Decrementa TEMPO y si es 0 salta
                goto    CONF_CLRT1      ; No 0 y borra T1 para 1 seg más
                goto    NORMAL          ; Ya es 0 y sale de modo config.
                ;1) Primera línea
CONF_LCD_L1     clrw                    ; Coloca 00h en W para
                call    LCD_POS         ; colocar el LCD en home
                ; Texto del Modo
                movlw   'C'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'o'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'n'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'f'             ;
                call    LCD_TEXTO       ; Imprime en LCD
```

```
                movlw   '.'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                ; Límite del reloj
                movlw   ' '             ; Espacio después del texto
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   't'             ; Para indicar límite de tiempo
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ; Espacio para límite de tiempo inactivo
                btfsc   LIMITES,LIMTI   ; Ve si está activo el límite de tiempo
                movlw   '<'             ; Indicar límite de tiempo activo
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   b'00000010'     ; Se enviara 1 dígito solamente
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                movf    HOR_LIM,W       ; Carga límite de horas en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movlw   ':'             ; Carga separador en W
                call    LCD_TEXTO       ; y lo imprime en LCD
                clrf    BCD_CONT        ; Se enviarán 2 dígitos sin espacio
                movf    MIN_LIM,W       ; Carga límite de minutos en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movlw   ' '             ; Espacio
                call    LCD_TEXTO       ; Imprime en LCD
                ; Diámetro de la rueda
                movlw   'R'             ; Etiqueta
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   '2'             ; Todas las rodadas van con un 2 antes
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   b'00000010'     ; Se enviara 1 dígito solamente
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                movf    RODADA,W        ; Carga rodada en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                ;2) Segunda línea
CONF_LCD_L2     movlw   h'40'           ; Coloca 40h en W para
                call    LCD_POS         ; colocar el LCD en segunda línea
                ; Etiqueta que indica qué se está modificando
                btfsc   MODIF,M_T       ; Ve si se está modificando tiempo
                goto    C_LCD_TIEMP     ; Etiqueta de tiempo
                btfsc   MODIF,M_D       ; Ve si se está modificando distancia
                goto    C_LCD_DIST      ; Etiqueta de distancia
                btfsc   MODIF,M_L       ; Ve si se está modificando límite
                goto    C_LCD_LIMS      ; Etiqueta de límite
                btfsc   MODIF,M_R       ; Ve si se está modificando rodada
                goto    C_LCD_RODA      ; Etiqueta de rodada
                btfsc   MODIF,M_F       ; Ve si se está modificando FCM
                goto    C_LCD_FCM       ; Etiqueta de FCM
                movlw   ' '             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ;
                call    LCD_TEXTO       ; Imprime en LCD
                goto    C_LCD_ETFIN     ;
C_LCD_TIEMP     movlw   'T'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'i'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'e'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'm'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'p'             ;
```

```
                call    LCD_TEXTO       ; Imprime en LCD
                goto    C_LCD_ETFIN     ;
C_LCD_DIST      movlw   'D'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'i'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   's'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   't'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ;
                call    LCD_TEXTO       ; Imprime en LCD
                goto    C_LCD_ETFIN     ;
C_LCD_LIMS      movlw   'L'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'i'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'm'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'i'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   't'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                goto    C_LCD_ETFIN     ;
C_LCD_RODA      movlw   'R'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'o'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'd'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'a'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'd'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                goto    C_LCD_ETFIN     ;
C_LCD_FCM       movlw   'F'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'C'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'M'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'a'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'x'             ;
                call    LCD_TEXTO       ; Imprime en LCD
C_LCD_ETFIN     movlw   ' '             ; Espacio después del texto
                call    LCD_TEXTO       ; Imprime en LCD
                ; Límite de la distancia
                movlw   'd'             ; Para indicar límite de distancia
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ; Espacio para límite de dist. inactivo
                btfsc   LIMITES,LIMDI   ; Ve si está activo el límite de dist.
                movlw   '<'             ; Indicar límite de dist. activo
                call    LCD_TEXTO       ; Imprime en LCD
                clrf    BCD_CONT        ; Se enviarán 2 dígitos sin espacio
                movf    D2_LIM,W        ; Carga cent. y dec. de km. en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movlw   b'00010000'     ; Se enviará sólo el dígito 1
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                movf    D1_LIM,W        ; Carga km y centenas de m. en W
                call    BCD_ASCII       ; Convierte y manda al LCD
                movlw   ' '             ; Espacio
                call    LCD_TEXTO       ; Imprime en LCD
                ; Frecuencia cardiaca máxima
                movlw   'F'             ; Etiqueta
```

```
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   b'00000101'     ; Se enviaran 3 dígitos con espacios
                movwf   BCD_CONT        ; y se coloca en el reg. de cont. de LCD
                movf    FCMAX,W         ; Carga la frec card max para mostrarse
                call    BCD_ASCII       ; Convierte y manda al LCD
                ; Manejo de los botones
CONF_LCD_FIN    btfss   ESTADO,SOLT_BOT ; Ver si se espera a que se desactiven
                goto    CONF_ACTIVO     ; No hay que esperar
                movf    PORTB,W         ; Pasa el puerto donde están los botones
                andlw   b'11100000'     ; Enmascara los tres botones
                btfss   STATUS,Z        ; Ve si estan en ceros
                goto    CONF_TEMPO      ; No estan en ceros y no hace nada
                bcf     ESTADO,SOLT_BOT ; Ya estan en ceros y atiende botones
CONF_ACTIVO     btfsc   PORTB,BOTON1    ; Ve si está presionado el botón 1
                goto    C_BOT1          ; Si está presionado y va a atenderlo
                btfsc   PORTB,BOTON2    ; Ve si está presionado el botón 2
                goto    C_BOT2          ; Si está presionado y va a atenderlo
                btfsc   PORTB,BOTON3    ; Ve si está presionado el botón 3
                goto    C_BOT3          ; Si está presionado y va a atenderlo
                goto    BUCLE_CONF      ; Ningún botón está presionado
                ; Atención al botón 1
C_BOT1          bcf     STATUS,C        ; Limpia carry para rotar
                movf    MODIF,F         ; Para ver si está en ceros
                btfsc   STATUS,Z        ; Ve si es cero el registro
                bsf     STATUS,C        ; Siguiente valor en entrar será un 1
                rlf     MODIF,F         ; Rota a la izquierda para cambiar
                movlw   b'00011111'     ; Enmascara sólo los bits de utilidad
                andwf   MODIF,F         ; Guarda en el registro
                clrf    TMR1H           ; Borra el Timer 1
                clrf    TMR1L           ; Borra el Timer 1
C_BOT1_ESP      btfss   TMR1H,6         ; Bit 6 en 1 durante 1/2 seg
                goto    C_BOT1_ESP      ; Espera antes de salir por si se deja
                goto    CONF_TEMPO      ; Repite el ciclo
                ; Atención al botón 2
C_BOT2          btfsc   MODIF,M_T       ; Ve si se está modificando tiempo
                goto    C_BOT2_TIEMP    ; Rutina de cambio de tiempo
                btfsc   MODIF,M_D       ; Ve si se está modificando distancia
                goto    C_BOT2_DIST     ; Rutina de cambio de distancia
                btfsc   MODIF,M_L       ; Ve si se está modificando límite
                goto    C_BOT2_LIMS     ; Rutina de cambio de límite
                btfsc   MODIF,M_R       ; Ve si se está modificando rodada
                goto    C_BOT2_RODA     ; Rutina de cambio de rodada
                btfsc   MODIF,M_F       ; Ve si se está modificando FCM
                goto    C_BOT2_FCM      ; Rutina de cambio de FCM
                goto    C_BOT2_NADA     ; No se está modificando nada
C_BOT2_TIEMP    movlw   d'5'            ; Incremento de 5 minutos
                addwf   MIN_LIM,F       ; Suma al límite de minutos
                movlw   d'60'           ; Carga 60d en W para comparaciones
                subwf   MIN_LIM,F       ; Resta para ver si ya llegó a 60 min.
                btfsc   STATUS,C        ; Si F<W Carry es 0 y no llegó a 60
                goto    INC_HOR_LIM     ; Si es 1, incrementa horas
                addwf   MIN_LIM,F       ; No es 1, restaura valor
                goto    C_BOT2_FIN      ; Ir al final de atención al botón 2
INC_HOR_LIM     movlw   HOR_LIM_MAX     ; Carga máximo en W para comparación
                incf    HOR_LIM,F       ; Incrementa hora límite en 1
                subwf   HOR_LIM,F       ; Resta para ver si ya llegó a 4 horas
                btfss   STATUS,C        ; Si F<W Carry es 0 y no llegó a 10
                addwf   HOR_LIM,F       ; No llegó, restaura valor
                goto    C_BOT2_FIN      ; Ir al final de atención al botón 2
C_BOT2_DIST     movlw   d'50'           ; Incremento de 5 km
                addwf   D1_LIM,F        ; Suma al límite de distancia 1
                movlw   d'100'          ; Carga 100d en W para comparaciones
                subwf   D1_LIM,F        ; Resta para ver si ya llegó a 10 km.
                btfsc   STATUS,C        ; Si F<W Carry es 0 y no llegó a 100
                goto    INC_D2_LIM      ; Es 1, F>=W incrementa distancia 2
```

```
                addwf   D1_LIM,F        ; No es 1, restaura valor
                goto    C_BOT2_FIN      ; Ir al final de atención al botón 2
INC_D2_LIM      movlw   D2_LIM_MAX      ; Carga máximo en W para comparación
                incf    D2_LIM,F        ; Incrementa distancia límite 2
                subwf   D2_LIM,F        ; Resta para ver si ya llegó a 1000 km
                btfss   STATUS,C        ; Si F<W Carry es 0 y no llegó a 100
                addwf   D2_LIM,F        ; No llegó, restaura valor
                goto    C_BOT2_FIN      ; Ir al final de atención al botón 2
C_BOT2_LIMS     incf    LIMITES,F       ; Incrementa para cambiar lim. activos
                movlw   b'00000011'     ; Solo interesan los dos bits - sig.
                andwf   LIMITES,F       ; Los enmascara y guarda
                goto    C_BOT2_FIN      ; Ir al final de atención al botón 2
C_BOT2_RODA     incf    RODADA,F        ; Incrementa rodada
                movlw   b'00000111'     ; Solo se cuenta hasta 7 (3 bits)
                andwf   RODADA,F        ; Los enmascara y guarda
                btfsc   STATUS,Z        ; Ve si dio cero
                bsf     RODADA,1        ; Si es cero debe poner 2 como mínimo
                call    ROD_AJUSTE      ; Ajusta los valores de las constantes
                goto    C_BOT2_FIN      ; Ir al final de atención al botón 2
C_BOT2_FCM      movlw   d'5'            ; Incrementos de 5 en 5
                addwf   FCMAX,F         ; Incrementa la frec. max. card. en 5
                movlw   FCMAX_MAX       ; Valor máximo
                subwf   FCMAX,F         ; Resta para comparar
                btfsc   STATUS,C        ; Si F<W Carry es 0 y no pasó de 210
                goto    C_B2_FCMIN      ; Si llegó y hay que poner mímimo valor
                addwf   FCMAX,F         ; Restaura el valor
                goto    C_BOT2_FIN      ; Ir al final de atención al botón 2
C_B2_FCMIN      movlw   FCMAX_MIN       ; El valor mínimo
                movwf   FCMAX           ; Permitirá bajar el límite
C_BOT2_FIN      bsf     ESTADO,DAT_MOD  ; Se ha modificado algún valor
C_BOT2_NADA     clrf    TMR1H           ; Borra el Timer 1
                clrf    TMR1L           ; Borra el Timer 1
C_BOT2_ESP      btfss   TMR1H,5         ; Bit 5 en 1 durante 1/4 seg
                goto    C_BOT2_ESP      ; Espera antes de salir por si se deja
                goto    CONF_TEMPO      ; Repite el ciclo
                ; Atención al botón 3
C_BOT3          movlw   d'81'           ; Pide retardo de 40 ms
                call    RETARDO         ; de la rutina de retardos
                btfss   ESTADO,DAT_MOD  ; Salta si hay dato modificado
                goto    C_BOT3_ESP      ; Sale sin guardar datos en EEPROM
C_GUARDAR       call    MENSAJE_G       ; Presenta mensaje de guardado
                ; Guardar constante de rodada y frec. max.
                movf    RODADA,W        ; Pasa la rodada actual
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_RODADA       ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                movf    FCMAX,W         ; Pasa la frec. card. max. actual
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_FCMAX        ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                ; Guardar tiempo y distancia límites
                movf    HOR_LIM,W       ; Pasa la hora límite actual
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_HOR_LIM      ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                movf    MIN_LIM,W       ; Pasa los minutos límite actual
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_MIN_LIM      ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                movf    D1_LIM,W        ; Pasa la dist. 1 límite actual
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
                movlw   EE_D1_LIM       ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
                movf    D2_LIM,W        ; Pasa la dist. 2 límite actual
                movwf   DAT_ESC_EE      ; al registro del dato a escribir en EE.
```

```
                movlw   EE_D2_LIM       ; Se guardará en esta dirección de EE
                call    ESCRI_EE        ; Escribe en EE
C_BOT3_ESP      bsf     ESTADO,SOLT_BOT ; Esperar a que se suelte el botón
                goto    NORMAL          ; Va a modo de operación normal
```

A.8. Archivo *sr_retardo.asm*

```
RETARDO         bcf     STATUS,RP0      ; Cambia al banco 0
                bcf     STATUS,RP1
                clrf    TMR2            ; Limpia TMR2
                bsf     STATUS,RP0      ; Cambia al banco 1
                movwf   PR2             ; Establece el límite para el TMR2
                bcf     STATUS,RP0      ; Cambia al banco 0
                bsf     T2CON,TMR2ON    ; Enciende el TMR2
RETARDO0        btfss   PIR1,TMR2IF     ; Ve bandera para ver si TMR2 ya acabo
                goto    RETARDO0        ; Es 0: no ha acabado
                bcf     T2CON,TMR2ON    ; Ya acabo. Apaga el TMR2
                bcf     PIR1,TMR2IF     ; Baja la bandera indicadora
                return                  ; Fin de la subrutina
```

A.9. Archivo *sr_eeprom.asm*

```
LEER_EE         bcf     STATUS,RP0      ; Cambia al banco 2
                bsf     STATUS,RP1
                movwf   EEADR           ; Selecciona la dirección a leer
                bsf     STATUS,RP0      ; Cambia al banco 3
                bcf     EECON1,EEPGD    ; Accesar a EEPROM
                bsf     EECON1,RD       ; Ejecutar la lectura
                bcf     STATUS,RP0      ; Cambia al banco 2
                movf    EEDATA,W        ; Se toma el valor leido y se deja en W
                bcf     STATUS,RP1      ; Cambia al banco 0
                return                  ; Regresa con el valor en W
ESCRI_EE        bcf     INTCON,GIE      ; Deshabilitar interrupciones
                bcf     STATUS,RP0      ; Cambia al banco 2
                bsf     STATUS,RP1
                movwf   EEADR           ; Selecciona la dirección a escribir
                movf    DAT_ESC_EE,W    ; Toma el valor a escribir del registro
                movwf   EEDATA          ; que será escrito en EEPROM
                bsf     STATUS,RP0      ; Cambia al banco 3
                bcf     EECON1,EEPGD    ; Accesar a EEPROM
                bsf     EECON1,WREN     ; Permitir la escritura en EEPROM
                movlw   h'55'           ; Instrucciones del fabricante para
                movwf   EECON2          ; escritura en EEPROM escribiendo 55h y
                movlw   h'AA'           ; luego AAh en el registro 2 de control
                movwf   EECON2          ; de la EEPROM.
                bsf     EECON1,WR       ; Iniciar la escritura
                bcf     EECON1,WREN     ; No permitir más la escritura en EEPROM
ESCRI_EE_ESP    btfsc   EECON1,WR       ; Espera a que termine la escritura
                goto    ESCRI_EE_ESP    ; No ha terminado
                bcf     STATUS,RP0      ; Cambia al banco 0
                bcf     STATUS,RP1
                return                  ; Ha finalizado
EE_LECT_DAT     movlw   EE_SEG          ; Segundos almacenados
                call    LEER_EE         ; Ejecuta la lectura
                movwf   SEG             ; Y guarda en el registro de RAM
                movlw   d'60'           ; Para ver si es dato válido
                subwf   SEG,W           ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    SEG             ; Carry es 1 y es >= y no es válido
                movlw   EE_MIN          ; Minutos almacenados
                call    LEER_EE         ; Ejecuta la lectura
                movwf   MIN             ; Y guarda en el registro de RAM
                movlw   d'60'           ; Para ver si es dato válido
```

```
                subwf   MIN,W           ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    MIN             ; Carry es 1 y es >= y no es válido
                movlw   EE_HOR          ; Horas almacenadas
                call    LEER_EE         ; Ejecuta la lectura
                movwf   HOR             ; Y guarda en el registro de RAM
                movlw   d'10'           ; Para ver si es dato válido
                subwf   HOR,W           ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    HOR             ; Carry es 1 y es >= y no es válido
                movlw   EE_DIST0        ; Distancia 0 almacenada
                call    LEER_EE         ; Ejecuta la lectura
                movwf   DIST0           ; Y guarda en el registro de RAM
                movlw   d'100'          ; Para ver si es dato válido
                subwf   DIST0,W         ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    DIST0           ; Carry es 1 y es >= y no es válido
                movlw   EE_DIST1        ; Distancia 1 almacenada
                call    LEER_EE         ; Ejecuta la lectura
                movwf   DIST1           ; Y guarda en el registro de RAM
                movlw   d'100'          ; Para ver si es dato válido
                subwf   DIST1,W         ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    DIST1           ; Carry es 1 y es >= y no es válido
                movlw   EE_DIST2        ; Distancia 2 almacenada
                call    LEER_EE         ; Ejecuta la lectura
                movwf   DIST2           ; Y guarda en el registro de RAM
                movlw   d'100'          ; Para ver si es dato válido
                subwf   DIST2,W         ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    DIST2           ; Carry es 1 y es >= y no es válido
                return                  ; Final
EE_LECT_PAR     movlw   EE_RODADA       ; Dir. del valor almacenado de la rodada
                call    LEER_EE         ; Ejecuta la lectura
                movwf   RODADA          ; Y guarda en el registro de RAM
                movlw   EE_MIN_LIM      ; Dir. límite de minutos almacenado
                call    LEER_EE         ; Ejecuta la lectura
                movwf   MIN_LIM         ; Y guarda en el registro de RAM
                movlw   d'60'           ; Para ver si es dato válido
                subwf   MIN_LIM,W       ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    MIN_LIM         ; Carry es 1 y es >= y no es válido
                movlw   EE_HOR_LIM      ; Dir. límite de horas almacenado
                call    LEER_EE         ; Ejecuta la lectura
                movwf   HOR_LIM         ; Y guarda en el registro de RAM
                movlw   HOR_LIM_MAX     ; Para ver si es dato válido
                subwf   HOR_LIM,W       ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    HOR_LIM         ; Carry es 1 y es >= y no es válido
                movlw   EE_D1_LIM       ; Dir. límite 1 de distancia almacenado
                call    LEER_EE         ; Ejecuta la lectura
                movwf   D1_LIM          ; Y guarda en el registro de RAM
                movlw   d'100'          ; Para ver si es dato válido
                subwf   D1_LIM,W        ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    D1_LIM          ; Carry es 1 y es >= y no es válido
                movlw   EE_D2_LIM       ; Dir. límite 0 de distancia almacenado
                call    LEER_EE         ; Ejecuta la lectura
                movwf   D2_LIM          ; Y guarda en el registro de RAM
                movlw   D2_LIM_MAX      ; Para ver si es dato válido
                subwf   D2_LIM,W        ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                clrf    D2_LIM          ; Carry es 1 y es >= y no es válido
                movlw   EE_FCMAX        ; Dir. frec. max. cardiaca almacenada
                call    LEER_EE         ; Ejecuta la lectura
```

```
                movwf   FCMAX           ; Y guarda en el registro de RAM
                movlw   FCMAX_MAX       ; Para ver si es dato válido
                subwf   FCMAX,W         ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                goto    INI_FCM_NV      ; No es válido
                movlw   FCMAX_MIN       ; Para ver si es dato válido
                subwf   FCMAX,W         ; Lo resta para ver si es mayor o igual
                btfsc   STATUS,C        ; Si carry es 0 es menor
                goto    INI_FCM_V       ; Si es válido
INI_FCM_NV      movlw   FCMAX_MIN       ; Valor mínimo de FCM
                movwf   FCMAX           ; Se guarda en RAM
INI_FCM_V       return                  ; Final
```

A.10. Archivo *sr_rodajuste.asm*

```
ROD_AJUSTE      movlw   d'02'           ; Constante vel. para todas las rodadas
                movwf   CTTEV1          ; Guardar en su registro de RAM
                movlw   d'7'            ; Para ver si es rodada 7
                subwf   RODADA,W        ; Compara
                btfsc   STATUS,Z        ; Si da 0 es que son iguales
                goto    ROD_AJ_CTTE_7   ; Cambiar las constantes a rodada 7
                movlw   d'6'            ; Para ver si es rodada 6
                subwf   RODADA,W        ; Compara
                btfsc   STATUS,Z        ; Si da 0 es que son iguales
                goto    ROD_AJ_CTTE_6   ; Cambiar las constantes a rodada 6
                movlw   d'5'            ; Para ver si es rodada 5
                subwf   RODADA,W        ; Compara
                btfsc   STATUS,Z        ; Si da 0 es que son iguales
                goto    ROD_AJ_CTTE_5   ; Cambiar las constantes a rodada 5
                movlw   d'4'            ; Para ver si es rodada 4
                subwf   RODADA,W        ; Compara
                btfsc   STATUS,Z        ; Si da 0 es que son iguales
                goto    ROD_AJ_CTTE_4   ; Cambiar las constantes a rodada 4
                movlw   d'3'            ; Para ver si es rodada 3
                subwf   RODADA,W        ; Compara
                btfsc   STATUS,Z        ; Si da 0 es que son iguales
                goto    ROD_AJ_CTTE_3   ; Cambiar las constantes a rodada 3
                movlw   d'2'            ; Para ver si es rodada 2
                subwf   RODADA,W        ; Compara
                btfsc   STATUS,Z        ; Si da 0 es que son iguales
                goto    ROD_AJ_CTTE_2   ; Cambiar las constantes a rodada 2
                movlw   d'2'            ; No fue ninguna
                movwf   RODADA          ; Asigna valor de default
                goto    ROD_AJ_CTTE_2   ; Cambiar las constantes a rodada 2
ROD_AJ_CTTE_7   movlw   h'01'           ; Constante 1 distancia
                movwf   CTTED1          ; para rodadas 25-27
                movlw   d'8'            ; Constante 0 dist. para rodada 27
                movwf   CTTED0          ; Guardar en su registro de RAM
                movlw   d'58'           ; Constante par. vel. para rodada 27
                movwf   CTTEV0          ; Guardar en su registro de RAM
                goto    ROD_AJ_FIN      ; Ir al final de subrutina
ROD_AJ_CTTE_6   movlw   h'01'           ; Constante 1 distancia
                movwf   CTTED1          ; para rodadas 25-27
                movlw   d'4'            ; Constante 0 dist. para rodada 26
                movwf   CTTED0          ; Guardar en su registro de RAM
                movlw   d'48'           ; Constante par. vel. para rodada 26
                movwf   CTTEV0          ; Guardar en su registro de RAM
                goto    ROD_AJ_FIN      ; Ir al final de subrutina
ROD_AJ_CTTE_5   movlw   h'01'           ; Constante 1 distancia
                movwf   CTTED1          ; para rodadas 25-27
                clrf    CTTED0          ; Constante 0 dist. para rodada 25
                movlw   d'39'           ; Constante par. vel. para rodada 25
                movwf   CTTEV0          ; Guardar en su registro de RAM
                goto    ROD_AJ_FIN      ; Ir al final de subrutina
```

```
ROD_AJ_CTTE_4  clrf    CTTED1          ; Constante 1 dist. para rodadas 22-24
               movlw   d'96'           ; Constante 0 dist. para rodada 24
               movwf   CTTED0          ; Guardar en su registro de RAM
               movlw   d'30'           ; Constante par. vel. para rodada 24
               movwf   CTTEV0          ; Guardar en su registro de RAM
               goto    ROD_AJ_FIN      ; Ir al final de subrutina
ROD_AJ_CTTE_3  clrf    CTTED1          ; Constante 1 dist. para rodadas 22-24
               movlw   d'92'           ; Constante 0 dist. para rodada 23
               movwf   CTTED0          ; Guardar en su registro de RAM
               movlw   d'21'           ; Constante par. vel. para rodada 23
               movwf   CTTEV0          ; Guardar en su registro de RAM
               goto    ROD_AJ_FIN      ; Ir al final de subrutina
ROD_AJ_CTTE_2  clrf    CTTED1          ; Constante 1 dist. para rodadas 22-24
               movlw   d'88'           ; Constante 0 dist. para rodada 22
               movwf   CTTED0          ; Guardar en su registro de RAM
               movlw   d'11'           ; Constante par. vel. para rodada 22
               movwf   CTTEV0          ; Guardar en su registro de RAM
ROD_AJ_FIN     return                  ; Final
```

A.11. Archivo *sr_lcd.asm*

```
; 1) Rutina para esperar a que se desocupe el LCD
LCD_OCUP       bsf     STATUS,RP0      ; Cambia a banco 1
               bcf     STATUS,RP1
               movlw   h'FF'           ; Configura PORTD como entrada
               movwf   TRISD
               movlw   d'09'           ; Prepara retardo de 5 ms con el TMR
               movwf   PR2             ; Establece el límite para el TMR2
               bcf     STATUS,RP0      ; Cambia a banco 0
               clrf    TMR2
               bsf     T2CON,TMR2ON    ; Enciende el TMR2
               bsf     PORTE,EN        ; Activa el LCD
               bcf     PORTE,RS        ; Envía un comando
               bsf     PORTE,RW        ; Va a hacer una lectura
LCD_OCUP0      btfss   PORTD,7         ; Ve si sigue ocupado el LCD
               goto    LCD_OCUP1       ; Está en 0: ya se desocupó
               btfss   PIR1,TMR2IF     ; Ve bandera para ver si TMR2 ya acabo
               goto    LCD_OCUP0       ; Está en 0: No ha transcurrido
LCD_OCUP1      bcf     PORTE,EN        ; Ya terminó o ya paso el retardo
               bcf     PORTE,RW        ; Deja en escritura
               bcf     T2CON,TMR2ON    ; Apaga el TMR2
               bcf     PIR1,TMR2IF     ; Baja la bandera indicadora
               bsf     STATUS,RP0      ; Cambia a banco 1
               clrf    TRISD           ; Configura PORTD como salida
               bcf     STATUS,RP0      ; Cambia a banco 0
               return                  ; Fin de la subrutina
; 2) Rutina para inicializar el LCD
; 2.1.- Ejecutar Function Set 3 veces
LCD_INI        bcf     STATUS,RP0      ; Cambia a banco 0
               bcf     STATUS,RP1
               movlw   h'03'           ; Para realizarse 3 veces según indica
               movwf   CONTEMP         ; el fabricante
LCD_INI0       bsf     PORTE,EN        ; Activa el LCD
               bcf     PORTE,RS        ; Va a enviar un comando
               movlw   b'00111000'     ; Function Set al LCD
               movwf   PORTD
               bcf     PORTE,EN        ; Desactiva el LCD
               movlw   d'09'           ; Prepara la llamada al retardo de 5 ms
               call    RETARDO         ; Hace la llamada
               decf    CONTEMP,F       ; Decrementa el contador
               btfss   STATUS,Z        ; Verifica el bit Z para ver si ya acabó
               goto    LCD_INI0        ; Es 0: no ha terminado
; 2.2.- Encender display
               bsf     PORTE,EN        ; Activa el LCD
```

```
                bcf     PORTE,RS        ; Va a enviar un comando
                movlw   b'00001100'     ; Encender LCD
                movwf   PORTD
                bcf     PORTE,EN        ; Desactiva el LCD
                call    LCD_OCUP        ; Espera a que termine de ejecutar
; 2.3.- Entry mode y limpiar LCD
                bsf     PORTE,EN        ; Activa el LCD
                bcf     PORTE,RS        ; Va a enviar un comando
                movlw   b'00000110'     ; Determina como se escribe en el LCD
                movwf   PORTD
                bcf     PORTE,EN        ; Desactiva el LCD
                call    LCD_OCUP        ; Espera a que termine de ejecutar
; 3) Rutina para borrar el LCD
LCD_BORRA       bcf     STATUS,RP0      ; Cambia a banco 0. Se redunda por si
                bcf     STATUS,RP1      ; la subrutina se llama por separado
                bsf     PORTE,EN        ; Activa el LCD
                bcf     PORTE,RS        ; Va a enviar un comando
                movlw   b'00000001'     ; Clear
                movwf   PORTD
                bcf     PORTE,EN        ; Desactiva el LCD
                call    LCD_OCUP        ; Espera a que termine de ejecutar
                return                  ; Fin de la subrutina
; 4) Rutina para escribir texto
LCD_TEXTO       bcf     STATUS,RP0      ; Cambia a banco 0
                bcf     STATUS,RP1
                bsf     PORTE,EN        ; Activa el LCD
                bsf     PORTE,RS        ; Va a enviar un dato
                movwf   PORTD           ; El parámetro enviado por W al puerto
                bcf     PORTE,EN        ; Desactiva el LCD
                call    LCD_OCUP        ; Espera a que termine de ejecutar
                return                  ; Fin de la subrutina
; 5) Rutina para establecer la posición del cursor
LCD_POS         bcf     STATUS,RP0      ; Cambia a banco 0
                bcf     STATUS,RP1
                bsf     PORTE,EN        ; Activa el LCD
                bcf     PORTE,RS        ; Va a enviar un comando
                iorlw   h'80'           ; Agrega el bit 7 en 1 para el comando
                movwf   PORTD
                bcf     PORTE,EN        ; Desactiva el LCD
                call    LCD_OCUP        ; Espera a que termine de ejecutar
                return                  ; Fin de la subrutina
```

A.12. Archivo *sr_bcdascii.asm*

```
BCD_ASCII       bcf     STATUS,RP0      ; Cambia al banco 0
                bcf     STATUS,RP1
                movwf   BCD_0           ; Guarda el valor a convertir en BCD_0
                clrf    BCD_1           ; Borra BCD_1 para iniciar la conversión
                clrf    BCD_2           ; Borra BCD_0 para iniciar la conversión
                movlw   d'100'          ; Pone 100 en W para contar centenas
BCD_BYTE2       subwf   BCD_0,F         ; Resta 100 para ver si hay centena
                btfss   STATUS,C        ; Ve si ocurrió acarreo en resta ¿C=0?
                goto    BCD_DECE        ; C=0 Si hay acarreo. Ya no hay centenas
                incf    BCD_2,F         ; Añade una centena
                goto    BCD_BYTE2       ; Continua restando centenas
BCD_DECE        addwf   BCD_0,F         ; Restaura valor orig. ant. a la resta
                movlw   d'10'           ; Pone 10 en W para contar decenas
BCD_BYTE1       subwf   BCD_0,F         ; resta 10 para ver si hay decena
                btfss   STATUS,C        ; Ve si ocurrió acarreo en resta ¿C=0?
                goto    BCD_UNID        ; C=0 Si hay acarreo. Ya no hay decenas
                incf    BCD_1,F         ; Añade una decena
                goto    BCD_BYTE1       ; Continua restando decenas
BCD_UNID        addwf   BCD_0,F         ; Restaura valor orig. ant. a la resta
                movlw   h'30'           ; Carga 30h para pasar a sus ascii
```

```
                addwf   BCD_0,F         ; Suma al dig 0 30h para ascii
                btfsc   BCD_CONT,0      ; ve si se desean espacios en dig 1 o 2
                goto    BCD_PONESP      ; Es uno: si se desean espacios
BCD_XX          movlw   h'30'           ; Carga 30h para pasar a sus ascii
                addwf   BCD_1,F         ; Suma 30h a dig 1 para obt. su ascii
                addwf   BCD_2,F         ; Suma 30h a dig 2 para obt. su ascii
                goto    BCD_LCD         ; Va a enviar al LCD
BCD_PONESP      swapf   BCD_2,W         ; Lleva dig. 2 a parte alta de W
                addwf   BCD_1,W         ; Le suma dig 1 y deja en W
                btfss   STATUS,Z        ; ¿La composición de dígitos es 00?
                goto    BCD_VER         ; Z=0 y no es 00. Ver si es 0X O XX
                movlw   h'20'           ; Si fue 00 y carga 20h para espacios
                movwf   BCD_2           ; tanto en BCD_2 como
                movwf   BCD_1           ; en BCD_1
                goto    BCD_LCD         ; Va a enviar al LCD
BCD_VER         movf    BCD_2,W         ; Carga dig. 2 en W para ver si es 0
                btfss   STATUS,Z        ; Ve si es 0X (dig 2=0)
                goto    BCD_XX          ; Salto pues es XX
                movlw   h'20'           ; Es 0X y carga 20h para espacio
                movwf   BCD_2           ; Hace espacio en dig. 2
                movlw   h'30'           ; Carga 30h para número en ascii
                addwf   BCD_1,F         ; Suma 30h para tener el dig. 1
BCD_LCD         movf    BCD_2,W         ; Carga el dígito 2 para ser escrito
                btfsc   BCD_CONT,2      ; Si 0 no se imp. y salta la escritura
                call    LCD_TEXTO       ; Escribe en LCD dígito 2
                movf    BCD_1,W         ; Carga dígito 1 para ser escrito
                btfss   BCD_CONT,1      ; Si 1 no se imp. y salta la escritura
                call    LCD_TEXTO       ; Escribe en LCD dígito 1
                movlw   '.'             ; Carga punto deci. para ser esc. en LCD
                btfsc   BCD_CONT,3      ; Si 0 no se imp. y salta la escritura
                call    LCD_TEXTO       ; Escribe en LCD punto decimal
                movf    BCD_0,W         ; Carga dígito 0 para ser escrito
                btfss   BCD_CONT,4      ; Si 1 no se imp. y salta la escritura
                call    LCD_TEXTO       ; Escribe en LCD dígito 0
                return                  ; FIN DE LA SUBRUTINA
```

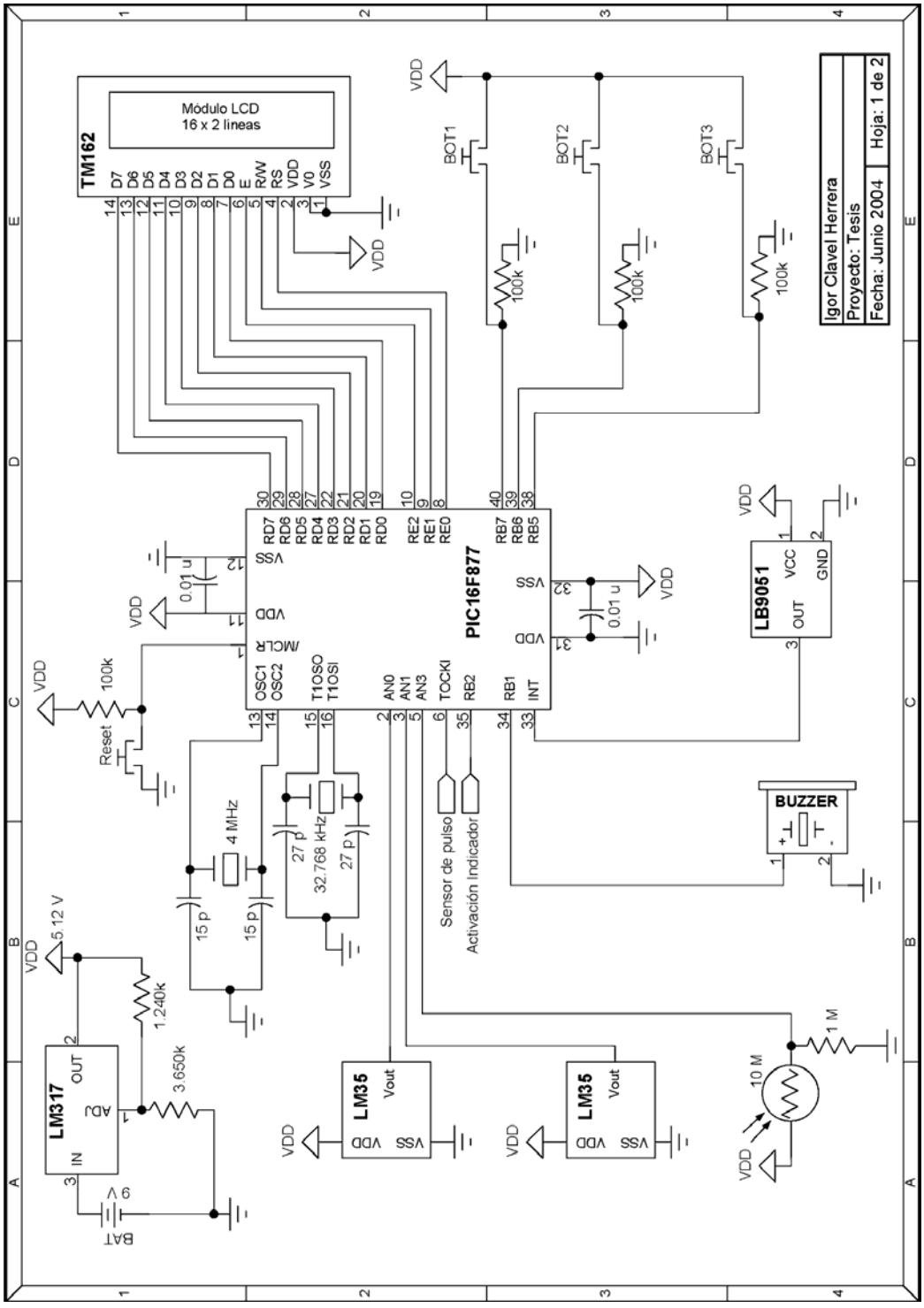
A.13. Archivo *sr_mensaje.asm*

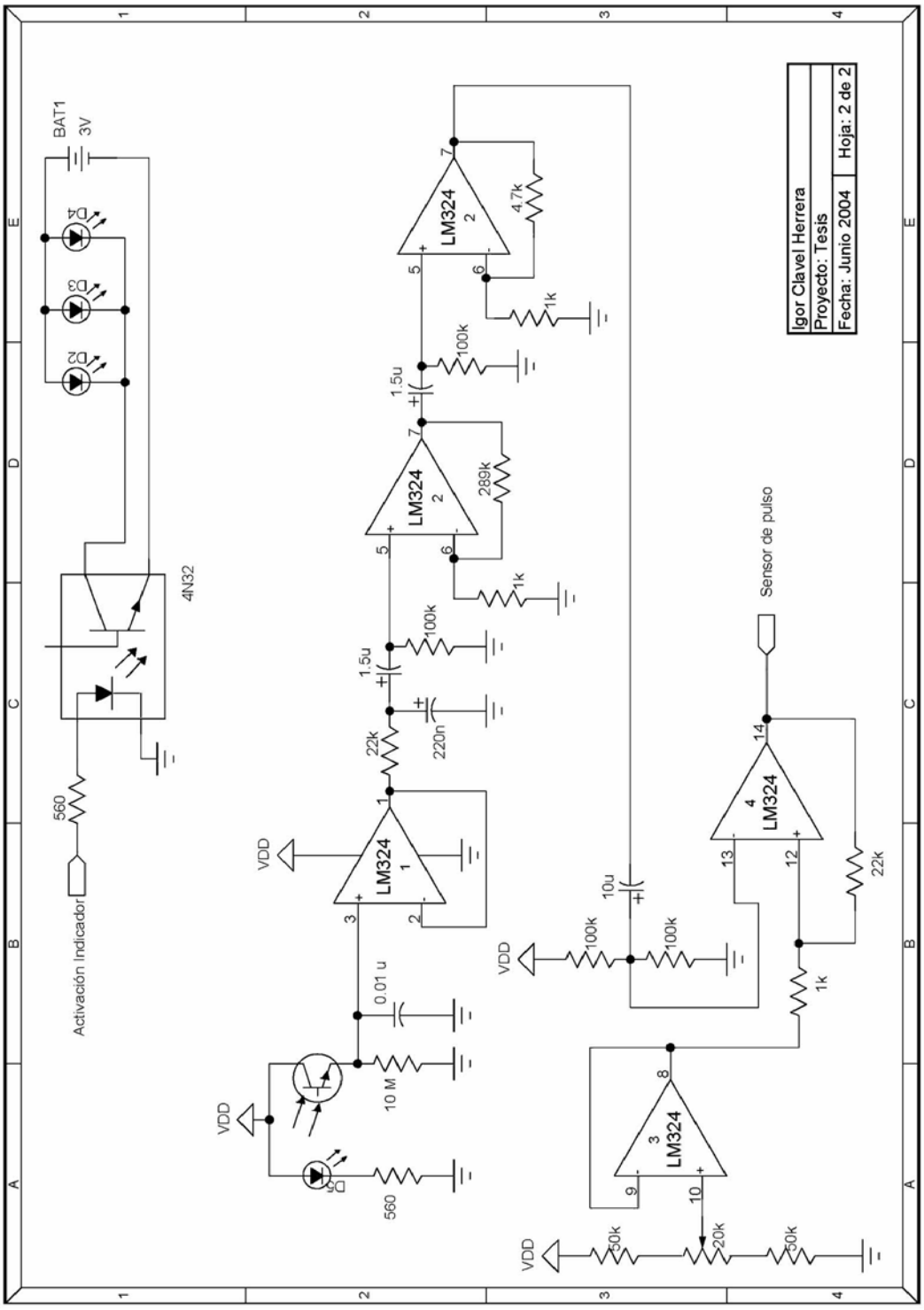
```
MENSAJE_G       clrf    TMR1H           ; Borra el Timer 1
                clrf    TMR1L           ;
                bsf     TMR1H,7         ; Pone a 1 bit 7 para mitad de rango
                bsf     PORTB,ZUMB      ; Activa el zumbador
MENSAJE_ESPG    btfss   TMR1H,5         ; Bit 5 de TMR1 en 1 durante 1/4 seg
                goto    MENSAJE_ESPG    ; Retardo de 1 segundo
                bcf     PORTB,ZUMB      ; Apaga el zumbador
                call    LCD_BORRA       ; Borra el LCD
                movlw   'G'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'u'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'a'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'r'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'd'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'a'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'n'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'd'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'o'             ;
                call    LCD_TEXTO       ; Imprime en LCD
```

```
                goto    MENSAJE_FIN     ; Va al final
MENSAJE_L       clrf    TMR1H           ; Borra el Timer 1
                clrf    TMR1L           ;
                bsf     TMR1H,7         ; Pone a 1 bit 7 para mitad de rango
                bsf     PORTB,ZUMB      ; Activa el zumbador
MENSAJE_ESPL    btfss   TMR1H,5         ; Bit 5 de TMR1 en 1 durante 1/4 seg
                goto    MENSAJE_ESPL    ; Retardo de 1 segundo
                bcf     PORTB,ZUMB      ; Apaga el zumbador
                call    LCD_BORRA       ; Borra el LCD
                movlw   'L'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'e'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'y'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'e'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'n'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'd'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'o'             ;
                call    LCD_TEXTO       ; Imprime en LCD
MENSAJE_FIN     clrf    TMR1H           ; Borra el Timer 1
                clrf    TMR1L           ;
MENSAJE_ESP2    btfss   TMR1H,7         ; Bit 7 de TMR1 en 0 durante 1 seg
                goto    MENSAJE_ESP2    ; Retardo de 1 segundo
                bcf     TMR1H,7         ; Reinicia cuenta de 1 segundo
MENSAJE_ESP3    btfss   TMR1H,7         ; Bit 7 de TMR1 en 0 durante 1 seg
                goto    MENSAJE_ESP3    ; Retardo de 1 segundo
                return                  ; Regresa
MENSAJE_B       call    LCD_BORRA       ; Borra el LCD
                movlw   'B'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'a'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'j'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'a'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   ' '             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'b'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   'a'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   't'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                movlw   '.'             ;
                call    LCD_TEXTO       ; Imprime en LCD
                return                  ; Regresa
```

Apéndice B

Diagrama Electrónico




BAT1
3V
D4
D3
D2
4N32
560
Activación Indicador
VDD
LM324
1
22k
220n
1.5u
100k
LM324
2
289k
1k
LM324
2
4.7k
1k
0.01 u
10 M
D5
560
LM324
3
LM324
4
Sensor de pulso
10u
100k
100k
22k
1k
50k
20k
50k
Igor Clavel Herrera
Proyecto: Tesis
Fecha: Junio 2004
Hoja: 2 de 2

Bibliografía

- ANGULO, José Ma., ANGULO, Ignacio, *Microcontroladores PIC. Diseño práctico de aplicaciones*, 2a. ed., Madrid: Ed. McGraw Hill, 1999, 295 pp.
- ANGULO, José Ma., CUENCA, E. Martín, *Microcontroladores PIC. La solución en un chip*, Barcelona: Ed. Paraninfo, 1998, pp. 5-53.
- ANGULO, José Ma.,ROMERO, Susana, ANGULO, Ignacio, *Microcontroladores PIC. Diseño práctico de aplicaciones. Segunda parte: PIC16F87x*, Madrid: Ed. McGraw Hill, 2000, 232 pp.
- BALL, Stuart R., *Analog interfacing to embedded microprocessors. Real world design*, Boston: Ed. Newnes, 2001, 271 pp.
- BOYLESTAD, Robert, NASHELSKY, Louis, *Electrónica. Teoría de circuitos*, trad., Carlos A. Franco, México: Ed. Prentice-Hall Hispanoamericana,1983, 782 pp.
- CROMWELL, Leslie (...) y otros, *Instrumentación y medidas biomédicas*, trad., Ramón Pallás, Barcelona: Ed. Marcombo, 1980,427 pp.
- DELORE, Michel, *Preparación y entrenamiento del ciclista. Guía práctica para todas las especialidades*, Barcelona: Ed. Hispano Europea, 1998, 124 pp.
- FAULKENBERRY, Luces M., *Introducción a los amplificadores operacionales con aplicaciones a CI lineales*, México: Ed. Limusa, 1996, 662 pp.
- FUELLA GARCÍA, José, *Acumuladores electroquímicos. Fundamentos, nuevos desarrollos y aplicaciones*, España: Ed. McGraw Hill, 1994, 248 pp.
- HALL, Douglas V., *Microprocessors and interfacing. Programming and hardware*, 3a. ed., USA: Ed. McGraw Hill, 1992, 624 pp.
- HODGES, David A., JACKSON, Horace G., *Análisis y diseño de circuitos integrados digitales*, "Memorias de semiconductores", Barcelona: Ed. Gustavo Gili, 1988, pp. 376-380.
- KRICK, E. V., *Introducción a la Ingeniería y al Diseño en la Ingeniería*, 2ª. ed., México: Ed. Limusa, 1980, 240 pp.
- LEVENTHAL, Lance A., *Z80: Assembly language programming*, “Problem definition and program design”, Berkeley: Ed. McGraw Hill, 1979, pp. 13-1 – 13-49.
- MANO, M. Morris, KIME, Charles R., *Fundamentos de diseño lógico y computadoras*, México: Ed. Prentice Hall Hispanoamericana, 1998, 604 pp.

- MILLER, Gene H., *Microcompter engineering*, USA: Ed. Prentice-Hall, 1999, 539 pp.
- NORET, André, BAILLY, Lucien, *El Ciclismo. Aspectos técnicos y médicos*, Barcelona: Ed. Hispano Europea, 1991, 430 pp.
- OGATA, Katsuhiko, *Ingeniería de control moderna*, trad., Bartolomé Fabian-Frankel, México: Ed. Prentice-Hall Hispanoamericana, 1980, pp. 675-679.
- PORTE, Gérard, *Guía general del ciclismo*, Madrid: Ed. Tutor, 1996, 377 pp.
- RODRÍGUEZ, Jorge, *Sistema de biomonitoreo médico personal*, Universidad Nacional Autónoma de México, Tesis de Maestría, México: El autor, 1994, 93 pp.
- RUEDA, Moisés E., PADILLA, Francisco J., MEDINA, Tomás, *Diseño e implementación de un sistema de control electrónico para un equipo médico de diálisis peritoneal*, Universidad Nacional Autónoma de México, Tesis de Licenciatura, México: Los autores, Tesis de Licenciatura, Universidad Nacional Autónoma de México, 1990, 193 pp.
- WILLIS, Neil, *Fundamentos de arquitectura de ordenadores y comunicaciones de datos*, España: Anaya, 1990, 303 pp.
- WRAY, William C., GREENFIELD, Joseph D., *Using microprocessors and microcomputers. The Motorola Family*, USA: Ed. Prentice-Hall, 1994, pp. 1-7.
- *AN587. Interfacing PICmicros™ to an LCD Module*, Microchip Technology Inc., USA: 1997, 33 pp.
- *Estatutos de la Unión Ciclista Internacional*, trad., Real Federación Española de Ciclismo, Homepage: Real Federación Española de Ciclismo, bajado el 13 de mayo de 2003, http://www.rfec.com.
- *National Analog and Interface Productos Databook* (Libro y CD-ROM), National Semiconductor, USA: 2001, 1672 pp.
- *PIC16F87X Data Sheet. 28/40-Pin 8-Bit CMOS FLASH Microcontrollers*, Microchip Technology Inc., USA: 2001, 218 pp.
- *Reglamento Generales del Deporte Ciclista*, trad., Real Federación Española de Ciclismo, Homepage: Real Federación Española de Ciclismo, bajado el 13 de mayo de 2003, http://www.rfec.com.