UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Sistema de detección, registro y despliegue de tiempos para pruebas de capacidad físico atlética

TESIS

Que para obtener el título de

Ingeniero Eléctrico Electrónico

P R E S E N T A

Santiago Ogando Justo

DIRECTOR DE TESIS

Ing. Yukihiro Minami Koyama

Ciudad Universitaria, Cd. Mx., 2003

A mis padres, Eliseo y Mary Carmen,
por su inagotable apoyo,
confianza y paciencia

A mi hermana Ana Belén,
por ser una inmejorable
referencia en mi vida

A mis abuelos, Venancio y Dulcina,
por todo el amor que me han dado

A mi familia,
de la que me siento tan afortunado

A mis amigos,
el tesoro más grande
que un hombre puede tener

A la vida, que me ha dado tanto…

ÍNDICE

CAPÍTULO 1

INTRODUCCIÓN

1.1 INTRODUCCIÓN GENERAL

El deporte es una actividad que ha sido practicada por el ser humano desde hace cientos de años con fines recreativos, así como para fomentar la competencia y el desarrollo de distintas capacidades y habilidades físicas. Una persona dedicada al deporte realiza día con día diversos ejercicios para mejorar su rendimiento físico, esperando alcanzar un desempeño óptimo en el deporte que practica, y sin duda alguna, a la persona dedicada a esta actividad le gustaría llevar un control cuantitativo de la evolución de su cuerpo para determinar la eficiencia de estos ejercicios.

Asimismo, existen instituciones deportivas que se dedican a conformar grupos de personas con características y habilidades particulares para fomentar el deporte de conjunto. En este caso, los entrenadores realizan pruebas de selección en las cuales los jugadores son sometidos a distintas pruebas de resistencia, fuerza y velocidad, a fin de obtener resultados significativos que permitan realizar una selección de los jugadores físicamente más aptos.

El Club Universidad Nacional, A. C., se ha destacado a lo largo de muchos años en la preparación de jóvenes futbolistas, para lo cual cuenta con varias filiales y equipos de fuerzas básicas en los que se encuentran deportistas que fueron seleccionados por sus habilidades y capacidades físico atléticas. Acostumbrado a realizar numerosas pruebas de rendimiento deportivo, el Club cobró conciencia de que la comodidad, confiabilidad y rapidez con que se realicen dichas pruebas, se verán reflejadas en un mejor control de los resultados obtenidos, y por lo tanto en una mejor selección de los jugadores.

La ingeniería es una actividad profesional que tiene como objetivo mejorar el nivel de vida de la sociedad, mediante la implementación de soluciones tecnológicas que permitan al ser humano realizar sus distintas actividades de una manera más fácil y eficiente.

De esta forma, la ingeniería encuentra en el deporte un campo más en el cual puede introducir innovaciones tecnológicas que permitan a los deportistas realizar un mejor

seguimiento de su evolución física, siendo la medición, automatización, adquisición y manejo estadístico de datos, algunos de los aspectos en los cuales puede visualizarse más claramente su aportación.

Por todo lo anterior, el Club Universidad Nacional, A. C., se acercó a la Facultad de Ingeniería buscando una solución tecnológica para automatizar sus pruebas de capacidad física. Como producto de dicho encuentro surge el presente proyecto, cuyo fin es realizar la medición, registro y despliegue de los resultados obtenidos en los ejercicios de salto vertical, carrera corta y frecuencia de paso.

Pero la importancia de este proyecto va más allá de la medición de pruebas físicas. Puede puntualizarse que el acercamiento que existió entre una institución deportiva y una educativa es un hecho a señalar, ya que es una muestra clara de que la sociedad considera a las universidades como una fuente de conocimiento capaz de brindar soluciones reales a problemas reales.

1.2 OBJETIVO

Se pretende diseñar y construir un sistema que permita estimar la altura que salta un jugador, el tiempo que emplea en realizar una carrera corta y su frecuencia de paso. Con el sistema a construir, se busca aumentar la rapidez, comodidad, fiabilidad y control con que se realizan dichas pruebas. Además, es importante que este sistema calcule, despliegue y almacene los resultados de las pruebas de manera que el entrenador pueda analizarlos y obtener conclusiones a partir de ellos de forma eficiente.

1.3 DESCRIPCIÓN DE LA TESIS POR CAPÍTULOS

El presente trabajo consta de cinco capítulos y un apéndice. En este capítulo, se presenta una introducción general al problema planteado, junto con los objetivos del proyecto.

En el segundo capítulo, se hace una revisión teórica de los elementos que sustentan al presente proyecto. Como temas principales se encuentran la teoría relacionada con las pruebas física a realizar, sensores, microcontroladores, transmisión de información y dispositivos de despliegue.

En el tercer capítulo se muestran las distintas opciones para la solución de cada una de las etapas del proyecto, así como los elementos que pueden constituir cada una de ellas. Al final del capítulo se hace una justificación del diseño y de los dispositivos finalmente elegidos, y se adquiere una visión de la constitución global del sistema.

En el cuarto capítulo se hace una descripción detallada del prototipo construido junto con cada una de las etapas que conforman el sistema. Asimismo, se muestra el funcionamiento y las características de cada uno de los elementos que constituyen el dispositivo tanto de hardware como de software, y de las interfaces empleadas.

En el último capítulo se muestran los resultados obtenidos, tanto del funcionamiento del prototipo construido como del diseño realizado, las conclusiones del proyecto, y finalmente una recomendación sobre el diseño final, que si bien va más allá de los objetivos plateados, se considera una buena alternativa para mejorar el dispositivo.

Por último, se presenta un apéndice en donde se muestran los listados de los programas realizados junto con información más específica sobre algunos tópicos.

CAPÍTULO 2

GENERALIDADES

2.1 INTRODUCCIÓN

En el presente capítulo se hace una revisión teórica de los elementos principales que componen cada una de las etapas del sistema. Antes que todo, se lleva a cabo la caracterización del sistema, y se explica la teoría relacionada con la realización de las pruebas físicas, junto con las consideraciones necesarias para obtener los resultados de las mismas por medio de un dispositivo como el que se diseña en el presente proyecto. Posteriormente, se describen algunas de las propiedades de los sensores que pueden definir las variables de entrada al sistema. A continuación se exponen los conceptos y características más importantes que involucran a un microcontrolador, y se hace una revisión de los conceptos y formatos relacionados con la transmisión de datos. Finalmente, se muestran los elementos, tanto de software como de hardware, que pueden utilizarse en el despliegue y almacenamiento de los resultados obtenidos en las pruebas.

2.2 DEFINICIÓN DEL PROBLEMA

En el diseño del dispositivo que va a emplearse para realizar la medición, despliegue y almacenamiento de los resultados de las pruebas físicas ya mencionadas, debe tomarse en cuenta características importantes como: movilidad, para que las pruebas físicas puedan realizarse tanto en espacio abierto como en espacio cerrado; robustez, para que soporte el trato duro que los deportistas puedan darle; versatilidad, para que un mismo dispositivo sirva para las tres actividades; fiabilidad, para que las mediciones obtenidas correspondan a la realidad; y economía, para obtener los resultados esperados al menor costo posible.

Para llevar a cabo la medición de las pruebas físicas se pretende utilizar una plataforma de medición, que mediante sensores permita determinar los instantes en que una persona hace contacto con ella, incluso de forma ligera, y deja de hacerlo. Para poder obtener un funcionamiento óptimo del dispositivo, será de gran importancia la elección de los sensores

que se colocarán en la plataforma, atendiendo aspectos como velocidad de respuesta, resolución, vida media y costo.

Para la prueba de carrera corta, se colocarán dos plataformas idénticas separadas por una distancia determinada, alrededor de 20 m. La prueba consiste en que una persona corra de una plataforma a otra a alta velocidad, y que el dispositivo calcule, con base en la detección del instante en que hace contacto con cada una de ellas, el tiempo que la persona emplea en cada recorrido. La figura 2.1 muestra el ejercicio descrito.

Figura 2.1 Prueba de carrera corta.

Para el caso de la prueba de salto vertical, el objetivo es hallar el desplazamiento del centro de gravedad de la persona al realizar el salto. En uno de los métodos convencionales para medir dicho desplazamiento, se mide la altura que alcanza una persona con la punta del dedo de la mano, manteniendo el brazo y las piernas completamente estirados y haciendo contacto apenas con la punta de los dedos de los pies, tal y como si se simulara el inicio de un salto vertical; posteriormente, cuando la persona realiza el salto, se mide la altura máxima que alcanza nuevamente con la punta del dedo y con el brazo estirado. La diferencia entre ambas medidas se considera como el desplazamiento del centro de gravedad, y por lo tanto, la altura saltada.

Ahora bien, en el presente proyecto se pretende estimar el desplazamiento del centro de gravedad mencionado a partir del tiempo en que la persona se mantiene en el aire. Para ello, se realiza el salto vertical sobre la plataforma de medición, y se detectan los instantes en que la persona la abandona al iniciar el salto y en el que regresa a ella al finalizar el mismo, como se muestra en la figura 2.2.

Si se calcula el tiempo transcurrido entre esos dos instantes, y se considera que el tiempo necesario para que la persona alcance la máxima altura es la mitad del tiempo total, puede calcularse la altura alcanzada con base en las ecuaciones de movimiento uniformemente acelerado. Cabe mencionar que dicha consideración no es completamente exacta. Si bien el

tiempo comienza a contarse a partir del instante en que las puntas de los pies dejan de tocar la plataforma con las piernas estiradas, tal y como corresponde a la teoría del salto vertical, el fin del salto se realiza con una pequeña flexión de las rodillas para amortiguar la caída, lo que significa que el tiempo de caída será ligeramente mayor al tiempo de ascenso, y que la diferencia no sólo es distinta para cada persona, sino que es distinta para cada uno de los saltos de una misma persona. Sin embargo, se espera que dicho error no sea significativo, y que pueda compensarse para obtener resultados confiables.

Figura 2.2 Prueba de salto vertical.

Para la prueba de frecuencia de paso se utiliza nuevamente la plataforma de medición, sobre la cual la persona debe colocar ambos pies, y comenzar el ejercicio levantando y bajando alternadamente cada una de las piernas a la máxima velocidad posible, como se muestra en la figura 2.3. Así, la plataforma debe contar el número de veces que la persona hace contacto con cada uno de los pies, y partir del tiempo durante el cual se realizó el ejercicio, se calcula la frecuencia de paso.

En una segunda etapa del sistema, se lleva a cabo el procesamiento de la información detectada por los sensores. Como variables de entrada se pretende emplear los instantes en que la persona hace contacto con la plataforma, y deja de hacerlo, mientras que como variables de salida se obtienen los parámetros que permiten determinar los resultados de las pruebas.

Adicionalmente, se requerirá un dispositivo de despliegue portátil para tener acceso a los datos provenientes del procesador; esta información deberá de almacenarse y desplegarse de manera clara y cómoda, para que el entrenador pueda realizar una primera evaluación en el momento de llevar a cabo la prueba.

Una vez finalizada la adquisición de datos en dicho dispositivo, éstos podrán transferirse a una computadora personal (PC) para ser almacenados y analizados posteriormente. Asimismo, los resultados guardados en la *PC* deberán visualizarse de manera clara y

estructurada, para que el entrenador pueda analizarlos y obtener conclusiones a partir de ellos.

Figura 2.3 Prueba de frecuencia de paso.

2.3 SENSORES

Un sensor es un dispositivo capaz de convertir un fenómeno físico en una señal eléctrica. Los sensores obtienen información del mundo exterior y la convierten en señales de entrada para un sistema de automatización o de control, y por esto, un correcto funcionamiento del sistema dependerá en gran medida de una adecuada elección de estos dispositivos. Los aspectos que son necesarios cuidar para la elección de los sensores son: inmunidad al ruido, seguridad, vida media, tamaño, consumo de energía, costo, comportamiento bajo condiciones ambientales y la compatibilidad de la señal de salida con los otros elementos del sistema, entre otros.

Dado que en la primera etapa del sistema se busca detectar los instantes en que una persona está situada sobre la plataforma y se retira de ella durante la realización de las pruebas, es necesario diseñar un arreglo de sensores que permita obtener una señal eléctrica que diferencie a dichos instantes. Como se desea que la variable de entrada al sistema únicamente cuente con dos estados (contacto – no contacto), la señal entregada por el sensor debe de ser una señal digital representada por unos y ceros lógicos.

Se consideró que para los fines del presente proyecto, los sensores más adecuados eran aquéllos capaces de detectar presencia y ausencia, una presión ejercida o un desplazamiento. A continuación se presentan algunos de los sensores que existen para determinar dichos parámetros, acompañados de una breve descripción de su principio físico. El propósito no es realizar una descripción exhaustiva de los sensores y su funcionamiento, sino mostrar algunas de las opciones para la elección final de los mismos.

1 Dispositivo potenciométrico. El posicionamiento de un cursor por medio de una fuerza eléctrica varía la resistencia eléctrica de un potenciómetro.

2 Galga extensiométrica resistiva. La resistencia de un alambre o de un semiconductor se modifica por la elongación debida a esfuerzos aplicados externamente.

3 Galga de presión de capacitancia variable. La distancia entre dos capas paralelas se varía por la aplicación de una fuerza externa.

4 Transductor de circuito magnético. La autoinductancia de una bobina excitada con corriente alterna se varía cambiando su circuito magnético.

5 Detector de reluctancia. La reluctancia de un circuito magnético se cambia variando la posición del núcleo de hierro de una bobina.

6 Transformador diferencial. El voltaje diferencial de dos devanados secundarios de un transformador se cambia variando la posición de un núcleo magnético por medio de una fuerza aplicada externamente.

7 Galga magnetométrica. Las propiedades magnéticas se varían por presión y esfuerzos mecánicos.

8 Emisor láser – fototransistor. El láser[1] (amplificación de luz por emisión estimulada de radiación) es un dispositivo emisor de luz cuyo haz cuenta con propiedades particulares que lo diferencia de otras fuentes luminosas: intensidad, que implica potencia por unidad de superficie; dirección, el haz es estrecho y su dispersión es mínima; coherencia, las ondas luminosas viajan en la misma dirección (coherencia espacial), en la misma frecuencia y en la misma fase (coherencia temporal); y finalmente, el haz emitido es monocromático, es decir, de un solo color. El fototransistor es un transistor con la base abierta en el que se presenta una pequeña corriente inversa de colector-base formada por portadores minoritarios. La corriente de colector es directamente proporcional a dicha corriente inversa con un factor β. Cuando la luz incide en la base, se produce un incremento de la corriente inversa, y por tanto un incremento mucho mayor en la corriente de colector.

9 Microinterruptor. El desplazamiento de un componente mecánico provoca un cambio en el estado de encendido y apagado del microinterruptor.

2.4 MICROCONTROLADOR

Un microcontrolador es un circuito integrado que contiene 5 unidades básicas conectadas entre sí por medio de grupos de conductores que permiten la transferencia de palabras en paralelo (buses): unidad de memoria, unidad lógico aritmética (ALU), unidad de entrada, unidad de salida y unidad de control. Figura 2.4.

[1] Light Amplification by Stimulated Emission of Radiation

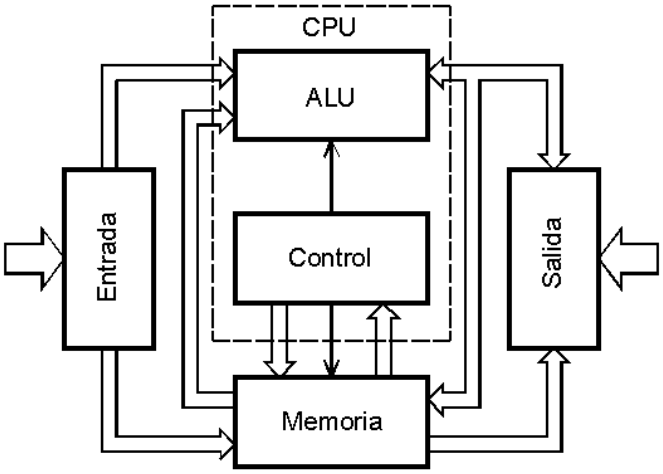


Figura 2.4 Diagrama general de un microcontrolador.

La unidad de memoria almacena conjuntos de bits (palabras) que representan las instrucciones que el microcontrolador ejecuta, o bien datos que son utilizados por el programa. La memoria también almacena datos temporales que son el resultado de las operaciones ejecutadas por la ALU.

La ALU lleva a cabo las operaciones aritméticas y lógicas sobre los datos. La unidad de control envía una señal a la ALU que determina la operación que va a ser ejecutada sobre los datos. Los resultados de las operaciones pueden ser enviados a la memoria para almacenarse, o a la unidad de salida.

Las unidades de entrada y salida consisten en una serie de dispositivos que permiten que la información proveniente del mundo exterior pueda ser recibida por la ALU o almacenada en la unidad de memoria, así como enviar los datos al mundo exterior después de que fueron procesados por la ALU. Algunos de estos dispositivos pueden ser puertos de entrada y salida, serie o paralelo, unidireccionales o bidireccionales, convertidores analógico-digitales, convertidores digital-analógicos, moduladores por ancho de pulso, etc.

La unidad de control es el cerebro del microcontrolador y se encarga de determinar la secuencia de las instrucciones, así como de mantener en correcta sincronización a todas las unidades.

En muchas ocasiones la ALU y la unidad de control se consideran como una sola unidad, que se denomina Unidad Central de Procesamiento (CPU). Cuando la CPU se construye en un solo chip se denomina microprocesador, y las unidades de memoria, de entrada y de salida, se convierten en unidades externas.

Los microcontroladores pueden clasificarse en los de propósito general y de propósito específico. Los microcontroladores de propósito general, son aquéllos que incorporan una gran cantidad de recursos. Este tipo de microcontroladores puede ser utilizado en cualquier aplicación, pero muchas veces cuenta con más recursos de los que se necesitan, y el elevado costo que hay que pagar por ellos es innecesario. Para ello, existen los

microcontroladores de propósito específico, más baratos, con menos capacidad de memoria, menos líneas de entrada y salida, menor velocidad de operación o de ancho de palabra, pero que se adaptan perfectamente a una aplicación específica. Para utilizar este tipo de microcontroladores, es necesario analizar muy bien la aplicación que va a desarrollarse para asegurar que el microcontrolador cuenta con las características requeridas.

La consideración de los recursos comunes a todos los microcontroladores, permite realizar una efectiva selección del más adecuado para una aplicación específica. A continuación se proporciona una breve explicación de estos recursos.

1 Arquitectura básica. Existen dos tipos de arquitectura en los microcontroladores: von Neumann y Harvard. La primera, se utilizó en todos los primeros microcontroladores, y se caracteriza por utilizar una misma memoria para almacenar indistintamente datos e instrucciones, y un único sistema de buses para transmitir información de datos, direcciones y control. La segunda, más utilizada en los últimos microcontroladores, tienen dos memorias separadas, una para los datos y otra para las instrucciones; ambas cuentan con un sistema de buses independiente, y pueden realizarse operaciones de lectura y escritura simultáneamente.

2 Procesador. Se encarga de direccionar la memoria de instrucciones, recibir y decodificar el código de la instrucción en curso y ejecutar la operación correspondiente, así como la búsqueda del operando y el almacenamiento del resultado. Los procesadores actuales pueden clasificarse en tres tipos:
CISC[2] (computadora de juego de instrucciones complejo). Cuenta con más de 80 instrucciones muy sofisticadas y potentes que requieren de muchos ciclos para su ejecución. Muy útil para aplicaciones que requieran rutinas complejas.
RISC[3] (computadoras de juego de instrucciones reducido). Cuenta con pocas y sencillas instrucciones que usualmente se ejecutan en un solo ciclo. Ideal para optimizar una aplicación.
SISC[4] (computadoras de juego de instrucciones específico). Este tipo de procesador, además de contar con un número de instrucciones reducido, éstas se adaptan a las necesidades de la aplicación.

3 Memoria. La memoria que utilizan los microcontroladores es de dos tipos: memoria ROM[5] (memoria de solo lectura) no vólátil, que se utiliza para almacenar el programa que gobierna la aplicación; memoria RAM[6] (memoria de acceso aleatorio), volátil, en donde se almacenan los datos y las variables que se producen en la ejecución del programa.
Existen cinco tipos de memoria ROM:
ROM con máscara. Memoria no volátil de sólo lectura cuyo contenido se graba durante la fabricación del chip. El elevado costo del diseño de la máscara sólo hace

[2] Complex Instruction Set Computer
[3] Reduced Instruction Set Computer
[4] Specific Instruction Set Computer
[5] Read Only Memory
[6] Random Access Memory

aconsejable el empleo de los microcontroladores con este tipo de memoria cuando se precisan cantidades superiores a varios miles de unidades.

PROM[7] (memoria programable una sola vez). El microcontrolador contiene una memoria no volátil de sólo lectura, programable una sola vez por el usuario mediante un grabador controlado por un programa desde una *PC*. Esta versión es recomendable cuando se construyen prototipos y series muy pequeñas. Tanto en este tipo de memoria como en la EPROM, suele protegerse el código contenido mediante fusibles.

EPROM[8] (memoria borrable y programable una sola vez). Los microcontroladores que disponen de memoria EPROM pueden borrarse y grabarse muchas veces con un grabador gobernado desde una *PC*. Para borrar el contenido, disponen de una ventana de cristal en su superficie por la que se somete a rayos ultravioleta durante varios minutos.

EEPROM[9] (memoria eléctricamente borrable y programable una sola vez). Se trata de memorias de sólo lectura, programables y borrables eléctricamente desde el propio grabador y bajo el control programado de una *PC*. Las operaciones de grabado y de borrado son muy cómodas y rápidas. Los microcontroladores dotados de memoria EEPROM una vez instalados en el circuito, pueden grabarse y borrarse cuantas veces se quiera sin ser retirados de dicho circuito. El número de veces que puede grabarse y borrarse una memoria EEPROM es finito. Este tipo de memoria es relativamente lenta.

FLASH Se trata de una memoria no volátil, de bajo consumo, que se puede escribir y borrar. Funciona como una ROM y una RAM pero consume menos energía y es más pequeña. Es programable en el circuito. Es más rápida, tolera más ciclos de escritura y borrado, y es de mayor densidad que la EEPROM.

3 Puertos de entrada y salida. Los puertos de entrada y salida consisten en ciertos registros asociados a las terminales del microcontrolador, por medio de las cuales se transmiten datos e información del mundo exterior hacia la memoria o hacia la ALU, y viceversa.

4 Reloj principal. Todos los microcontroladores disponen de un circuito oscilador, generalmente compuesto por un cristal de cuarzo junto a elementos pasivos, que genera una onda cuadrada de alta frecuencia que configura los impulsos de reloj usados en la sincronización de todas las operaciones del sistema.

Algunos microcontroladores poseen recursos especiales para aumentar su potencial; sin embargo, si la aplicación no requiere de alguno de ellos, puede seleccionarse un microcontrolador que no los tenga a fin de disminuir el costo. Entre los principales de estos recursos se encuentran:

1 Temporizador o *Timer*. Se emplea para controlar periodos de tiempo (temporizador) o para llevar la cuenta de acontecimientos que suceden en el exterior (contadores). Consta básicamente de un registro que se incrementa o decrementa según los impulsos de un reloj o debido a acontecimientos externos, hasta que se desborda provocando un aviso.

[7] Programmable Read Only Memory
[8] Erasable Programmable Read Only Memory
[9] Electrically Erasable Programmable Read Only Memory

2 Perro guardián o *Watchdog*. El Perro guardián consiste en un temporizador que cuando se desborda y pasa por 0, provoca automáticamente el reinicio del microcontrolador. El programa de trabajo que controla la tarea a ejecutar, debe realizarse de forma que inicialice al Perro guardián antes de que provoque el reinicio. Si el programa falla o se bloquea, no se refrescará al Perro guardián y se provocará el reinicio del sistema.

3 Protección ante fallo de alimentación o *Brownout.* Se trata de un circuito que reinicia al microcontrolador cuando el voltaje de alimentación (V_{DD}) es inferior a un voltaje mínimo "brownout". El microcontrolador comenzará a funcionar nuevamente hasta que se sobrepase dicho nivel de voltaje.

4 Estado de reposo o de bajo consumo. Para ahorrar energía, los microcontroladores disponen de una instrucción especial que los transfiere al estado de reposo o de bajo consumo, en el cual los requerimientos de potencia son mínimos. En dicho estado se detiene el reloj principal y se "congelan" sus circuitos asociados. Al activarse una interrupción ocasionada por el acontecimiento esperado, el microcontrolador se "despierta" y reanuda la ejecución del programa.

5 Convertidor A/D (CAD). Los microcontroladores que incorporan un convertidor A/D (analógico-digital) pueden procesar señales analógicas provenientes del exterior y convertirlas a señales digitales, de tal forma que los valores de voltaje de la señal analógica quedan representados por un conjunto de datos.

6 Convertidor D/A (CDA). Un convertidor D/A (digital-analógico) genera una señal analógica a partir datos digitales obtenidos mediante el procesador.

7 Modulador de ancho de pulso (PWM). Son circuitos que proporcionan en la salida impulsos de anchura variable, que se envían al exterior a través de las terminales del chip.

8 Puertos de entrada/salida digitales. Todos los microcontroladores destinan algunas de sus terminales a soportar líneas de entrada-salida digitales. Por lo general, estas líneas se agrupan de ocho en ocho formando puertos. Las líneas digitales de los puertos pueden configurarse como entrada o como salida.

9 Puertos de comunicación. Para que el microcontrolador pueda comunicarse con otros dispositivos externos, otros buses de microprocesadores, de sistemas o de redes, y poder adaptarlos con otros elementos bajo otras normas y protocolos específicos, algunos modelos disponen de recursos que permiten realizar directamente esta tarea, entre los que destacan:
UART[10]: receptor y transmisor universal asíncrono.
USART[11]: transmisor y receptor universal síncrono y asíncrono.

[10] Universal Asynchronous Receiver Transmitter
[11] Universal Synchronous Asynchronous Receiver Transmitter

USB[12]: un moderno bus serie universal para las *PC*.
Bus I^2C[13]: circuito inter-integrado que funciona como interfaz serie de dos hilos desarrollado por Philips.

Algunos de los fabricantes más importantes de microcontroladores son: Intel, Motorola, Toshiba, Texas Instruments, Philips, National y MicroChip.

En cuanto a técnicas de fabricación, la mayoría de los microcontroladores se fabrican con tecnología CMOS (Complementary Metal Oxide Semiconductor), con propiedades sobresalientes en bajo consumo de energía e inmunidad al ruido.

Por último, es necesario mencionar las herramientas de software que permiten desarrollar aplicaciones con un microcontrolador.

- Ensamblador. La programación en lenguaje ensamblador es un tanto ardua; sin embargo, permite tener un mayor control sobre los elementos del sistema y obtener aplicaciones muy eficientes. El ensamblador transforma un programa escrito con instrucciones simbólicas (lenguaje ensamblador) a lenguaje de máquina (unos y ceros) con el que opera el microcontrolador.

- Compilador. La programación en un lenguaje de alto nivel permite crear aplicaciones en menos tiempo; no obstante, los programas pueden resultar poco eficientes por la gran cantidad de código que generan. El compilador transforma un programa de lenguaje de alto nivel a lenguaje de máquina.

- Simulador. Un simulador es un software que se instala en una *PC*, y que permite depurar y ejecutar línea por línea un programa desarrollado en un microcontrolador, al mismo tiempo que se visualiza el estado de los registros y de la memoria, entre otras cosas; sin embargo, es difícil simular la entrada y salida de datos, así como el ruido en las entradas.

- Placas de evaluación. Se trata de pequeños sistemas con un microcontrolador ya montado y que suelen conectarse a una *PC* desde el que se cargan los programas que se ejecutan en el microcontrolador. Las placas suelen incluir pantallas, teclados, LED's, entre otras.

- Emuladores en circuito. Se trata de un instrumento que se coloca entre una *PC* y la base del chip de la tarjeta de circuito impreso donde se alojará el microcontrolador definitivo. El programa es ejecutado desde la *PC*, pero para la tarjeta de aplicación es como si lo hiciese el mismo microcontrolador que luego se instalará en la base. Además, presenta en pantalla toda la información tal y como luego sucederá cuando se coloque el circuito integrado.

[12] Universal Serial Bus
[13] Inter-Integrated Circuit

2.5 TRANSMISIÓN SERIAL DE DATOS

Existen dos tipos de comunicación serial de datos entre dispositivos: comunicación síncrona y asíncrona. En la comunicación serial asíncrona, el transmisor puede enviar los datos al receptor en cualquier momento y con un retraso indeterminado entre cada palabra enviada. En otras palabras, en una comunicación asíncrona la señal de reloj del transmisor no tiene que estar sincronizada con la señal de reloj del receptor.

En una comunicación serial síncrona, el transmisor continuamente está transmitiendo datos al receptor. Las palabras de datos usualmente se agrupan en bloques, y cada bloque está separado por palabras especiales, llamadas caracteres de sincronización. Estos caracteres son utilizados por el receptor para sincronizar su señal de reloj con la del transmisor. Así, cuando el transmisor no tiene ningún dato que enviar, envía estos caracteres especiales.

La mayor parte de la comunicación serial de los microcontroladores se realiza de forma asíncrona, que es en la que se profundizará a continuación.

La figura 2.5 muestra un arreglo básico mediante el cual se lleva a cabo la comunicación serial del microcontrolador con un dispositivo. El objetivo principal de la interfaz, es convertir los datos en paralelo, provenientes del microcontrolador a través del bus de datos, a su forma serial para ser transmitidos al dispositivo externo; de igual forma, recibe un dato del exterior de forma serial, y lo transforma a un arreglo en paralelo que se envía al microcontrolador. La transmisión serial de los datos realizada entre el dispositivo externo y la interfaz, sigue un formato que a continuación se describe.

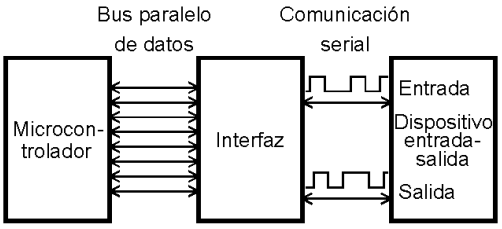


Figura 2.5 Comunicación serial entre un microcontrolador y un dispositivo externo.

La señal de un dato transmitido está dividida en intervalos de tiempo llamados tiempo de bit. Durante cada tiempo de bit (Tb), la señal puede valer un 0 ó 1 lógico, valor que sólo puede cambiar al empezar un nuevo intervalo. Cada dato transmitido serialmente sigue un formato estándar que consiste en cuatro partes:

1 Un bit de inicio, cuyo valor siempre es un 0 lógico.
2 De cinco a ocho bits de datos que representan la información transmitida. Normalmente se transmite primero el bit menos significativo del dato (LSB) y al final el bit más significativo (MSB).
3 Opcionalmente se transmite un bit de paridad para detección de errores.
4 Uno o dos bits de paro, que siempre son unos lógicos.

La figura 2.6 ilustra un ejemplo de una transmisión serial de un dato de siete bits con un bit de paridad y dos bits de paro, formato en el que usualmente se transmiten los caracteres en código ASCII.

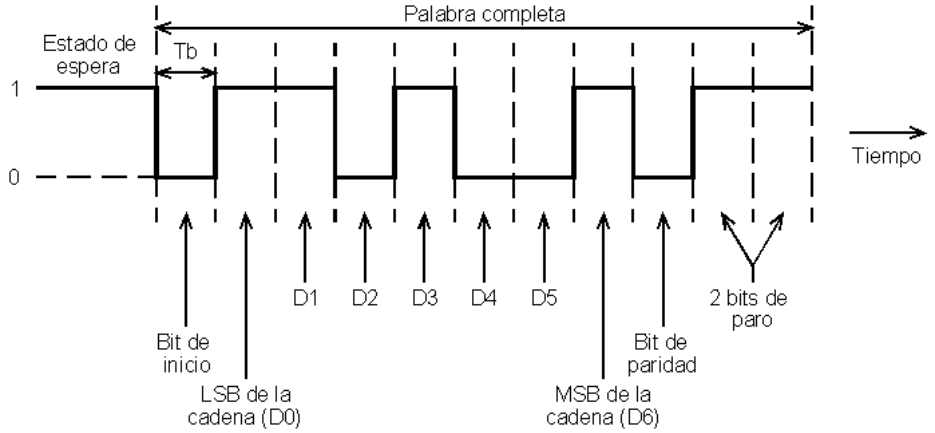


Figura 2.6 Transmisión serial de datos.

Cuando ningún dato es transmitido hacia el receptor, la señal transmitida se encuentra en estado alto (1 lógico) permanentemente, conocido también como estado de espera. Cuando varias palabras son transmitidas consecutivamente, debe existir al menos dos tiempos de bit entre el último bit de dato transmitido y el bit de inicio de la siguiente cadena.

Uno de los estándares de comunicación serial más utilizados para transmitir información de una *PC* a un dispositivo externo mediante una terminal es el RS-232-C. Este estándar establece niveles de voltaje para las señales transmitidas mediante una lógica negativa. Así, un 1 lógico está representado por valores de voltaje entre –12 V y –3 V, mientras que al 0 lógico le corresponden valores de entre 3 V y 12 V. Los valores de voltaje entre 3 V y –3 V no definen ningún nivel lógico.

En los sistemas convencionales que emplean microcontroladores, se emplean los niveles de voltajes definidos por la tecnología TTL, que asigna 0 V al 0 lógico y 5 V al 1 lógico. Por lo tanto, para que una transmisión serial pueda realizarse entre un microcontrolador y una *PC*, debe utilizarse una interfaz que permita acoplar los valores de voltaje empleados en la tecnología TTL y los estándares RS-232-C.

2.6 DESPLIEGUE DE INFORMACIÓN: HARDWARE Y SOFTWARE

Existen varios dispositivos que permiten recibir información de otros dispositivos y desplegarla. Por lo general, para poder manipular dicha información y crear un ambiente que permita al usuario tener acceso a ella, es necesario utilizar alguna herramienta de programación. A continuación se mencionan algunos de los dispositivos que pueden emplearse útiles para desempeñar dicha tarea, junto con el software para el desarrollo de aplicaciones.

Mediante la utilización de una *PC* y empleando un lenguaje de alto nivel de programación, como puede ser Quick Basic, Turbo Pascal o C/C++, entre otros, puede recibirse información a través de sus puertos de comunicación, procesarla, almacenarla y desplegarla. Para ello, es necesario considerar los formatos de transmisión y recepción que utilizan tanto los puertos de la *PC* como los puertos de los dispositivos con los cuales se va a establecer la comunicación, a fin de desarrollar la interfaz necesaria si éstos no llegaran a ser iguales. Asimismo, habría que emplear una herramienta como Visual Basic o Visual C, que permita a un usuario, sin conocimientos de programación, visualizar y manipular los resultados obtenidos en un ambiente interactivo, claro y fácil de usar.

Una pantalla de cristal líquido (LCD[14]), permite visualizar caracteres con distintos formatos. Existen distintos modelos de *LCD's*, los cuales se diferencian principalmente en el tamaño de la pantalla en la cual despliegan la información. Poseen una amplia gama de caracteres, y bastantes recursos en cuestión de formatos para desplegar la información; sin embargo, dicho dispositivo es incapaz de almacenar y desplegar información por sí mismo, y requiere de un dispositivo como un microcontrolador, para controlar sus funciones.

Un organizador personal (Palm), es un dispositivo de pequeñas dimensiones diseñado para organizar y almacenar información relacionada con horarios, direcciones, teléfonos y citas importantes, entre otras cosas. Sin embargo, al contar este dispositivo con distintas aplicaciones, una capacidad de memoria considerable y formatos de comunicación, entre otros aspectos que se mencionarán posteriormente, la idea de utilizar este dispositivo con otros fines se convierte en una posibilidad a tomarse en cuenta. Existe una gran variedad de modelos de *Palm*, los cuales presentan distintas características en la versión del sistema operativo, memoria, opciones de pantalla y sonido, conexión con dispositivos externos, por citar algunas.

El sistema operativo empleado por estos dispositivos es el Palm OS. Este es un sistema operativo que fue desarrollado en lenguaje C, y por lo tanto, el software para el desarrollo de aplicaciones más flexible y eficiente para las *Palms* también está escrito en lenguaje C.

La memoria interna de la *Palm* es rápida y no volátil, lo que significa que puede guardar valores almacenados en ella aún cuando se encuentre apagada. Las *Palms* más recientes tienen una capacidad de memoria de 14 MB, la cual puede ampliarse con tarjetas de expansión de hasta 32 MB. Además, muchas de ellas están equipadas con una memoria Flash, lo que significa que la versión de su Palm OS puede ser actualizada.

La mayoría de los modelos de *Palm* cuentan con un puerto de comunicación serial que funciona bajo el protocolo RS-232-C, aunque los más modernos ya incluyen un conector USB. La base en donde se coloca la *Palm*, que se conecta al puerto serie de la *PC*, incluye un botón que al oprimirse permite sincronizar la *Palm* con la *PC* y así transferir datos y programas entre ambas. También puede conectarse a este puerto un módem o algún otro dispositivo, como un lector de código de barras, por ejemplo. Además, algunas de las *Palms* más recientes cuentan con un puerto de comunicación infrarroja que permite, con la

[14] Liquid Cristal Display

programación apropiada, conectar el dispositivo con otros accesorios que utilicen el mismo tipo de comunicación.

Los últimos modelos de *Palm* cuentan con conexión a Internet una vez que se ha contratado el servicio adecuado de comunicación.

A continuación se presenta una lista de algunas de las herramientas más importantes en el desarrollo de aplicaciones para la *Palm*.

1 Code Warrior. Esta es una de las herramientas de desarrollo de aplicaciones para Palm OS más completas que existen. Esta herramienta utiliza el lenguaje C para crear sus aplicaciones. La mayor ventaja de *Code Warrior*, radica en la utilización de un ambiente de programación, llamado Integrated Development Environment (IDE) que permite editar, compilar, ligar y depurar el código de programa conjuntamente.

2 GCC/PRC. Éste es un compilador que se encuentra disponible sin costo alguno, y que permite crear códigos de una manera rápida. Sin embargo, su instalación requiere de mucho tiempo y esfuerzo si no se está familiarizado con la aplicación. Existen ciertas restricciones en el uso de este compilador, por lo que es recomendable una minuciosa consulta de la licencia de este producto antes de invertir mucho tiempo en ella.

3 Emulador Palm OS. Esta herramienta permite probar la aplicación creada sin necesidad de usar una *Palm* directamente. Es ideal para depurar, revisar el código de programación paso a paso y encontrar problemas en la ejecución de la aplicación que de otra manera serían muy difíciles de hallar.

4 Pendragon Forms. Es una de las mejores alternativas para la adquisición y manejo de datos. *Pendragon Forms*, permite crear una base de datos en la *Palm* para recaudar datos y transferirlos posteriormente a otra base de datos creada en la *PC* que trabaja conjuntamente con Microsoft Access. Está limitado para otras aplicaciones que no estén relacionadas con el manejo de datos.

5 Satellite Forms. Esta es una de las herramientas más útiles para los usuarios sin conocimiento de programación. Utiliza un ambiente interactivo que permite crear aplicaciones básicas de manera rápida, aunque no es una herramienta muy flexible. Como restricción, es necesario comprar una licencia para cada *Palm* en la cual se instale este software.

CAPÍTULO 3

BÚSQUEDA DE SOLUCIONES

3.1 INTRODUCCIÓN

El presente proyecto puede clasificarse como un sistema de instrumentación electrónico que consta de distintos componentes para realizar una medición y desplegar un resultado. Como todo sistema de instrumentación, consta de tres dispositivos básicos: un dispositivo de entrada, un dispositivo de procesamiento y un dispositivo de salida. Mediante el uso de sensores, el dispositivo de entrada nos permite obtener del mundo exterior las variables de entrada al sistema. El dispositivo de procesamiento, se encarga de manipular la información proporcionada por los sensores y obtener los resultados de las pruebas. Finalmente, el dispositivo de salida almacena y despliega los valores calculados por el procesador para que un usuario pueda acceder a ellos.

En el presente capítulo, se muestran las ventajas y desventajas de las diferentes opciones planteadas para resolver cada una de las etapas del sistema, a fin de justificar el diseño definitivo del mismo. Al finalizar el capítulo, se tendrá una idea global del funcionamiento y de los elementos que constituyen el sistema.

3.2 PLATAFORMA DE MEDICIÓN

En esta primera etapa, se considera el diseño de una plataforma de medición en donde se realizarán las pruebas, de tal forma que se obtengan los parámetros provenientes del mundo exterior que permitan calcular los resultados de las mismas. Para llevar a cabo dicho diseño deben tomarse en cuenta las siguientes consideraciones:

- La misma plataforma de medición debe permitir obtener los resultados de las tres pruebas físicas.

- Para el caso de la prueba de salto vertical, la plataforma debe detectar los instantes en que la persona despega el pie de la plataforma al iniciar el salto, y en el que cae sobre ella al finalizar el mismo. Según resultados obtenidos en el Laboratorio de Biomecánica de la Dirección de Medicina del Deporte, la fuerza que ejerce una persona sobre el suelo al caer y al impulsarse cuando lleva a cabo un salto vertical, tiene un valor aproximado de dos veces el peso de la persona; por ello, la plataforma debe ser lo suficientemente resistente para soportar al menos tres veces el peso de una persona normal, para asegurar que no va a sufrir deformaciones que afecten el funcionamiento del sistema durante la realización de las pruebas.

- Para el caso de la prueba de frecuencia de paso, la plataforma debe ser capaz de detectar el instante en que la persona la pisa con cada uno de los pies.

- Por último, para la prueba de carrera corta, se necesita que la plataforma detecte el instante en que la persona inicia y finaliza el recorrido. Para ello, la persona despegará su pie de una plataforma cuando inicie la carrera, y cuando llegue al final de la misma, tocará con el pie otra plataforma. Por lo anterior, se requiere que la plataforma sea lo suficientemente sensible para detectar contactos ligeros.

Con base en lo anterior, se consideran dos principios de operación bajo los cuales la plataforma puede detectar los eventos mencionados: detección de contacto con base en la presión ejercida sobre la plataforma, y detección de contacto con base en la presencia y ausencia sobre la plataforma. A partir de dichos principios de operación, se contempla el diseño físico de la plataforma, así como la elección y distribución de los sensores que se van a emplear.

3.2.1 Detección de contacto con base en la presión ejercida sobre la plataforma

Bajo este principio, se pretende transformar en una señal eléctrica la fuerza que ejerce una persona sobre la plataforma cuando hace contacto con ella. Para ello se consideró el uso de tres tipos de sensores: microinterruptor, sensor de presión y sensor de flexión. Para los tres tipos de sensores se pensó en emplear un diseño basado en una plataforma cuya base superior se desplace hacia abajo al ejercer una fuerza sobre ella, y cuyo desplazamiento fuese detectado por alguno de estos sensores. La figura 3.1 muestra el diseño de una plataforma con esta estructura.

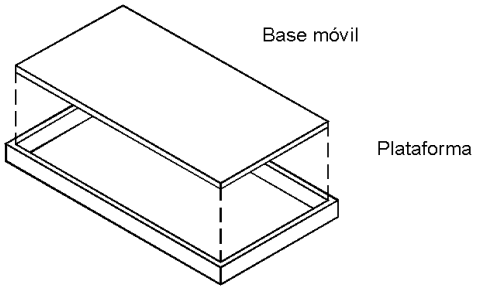


Figura 3.1 Plataforma con base superior móvil.

- **Plataforma con microinterruptor**

El microinterruptor que se muestra en la figura 3.2 (http://www.jameco.com), es un dispositivo que cambia de estado al cambiar de posición su parte móvil.

Figura 3.2 Microinterruptor.

Una forma de conseguir el objetivo buscado, sería instalando varios microinterruptores en la parte interna de las paredes de la plataforma, de manera que cuando la persona hiciera contacto sobre ella, la parte móvil se desplace hacia el interior una distancia suficiente para accionar alguno de los microinterruptores. Con este tipo de sensor, puede obtenerse una señal digital tal que cuando la persona no haga contacto con la plataforma, el microinterruptor esté apagado y se envíe un estado bajo, y cuando la persona se coloque sobre ella, el microinterruptor se accione y envíe un estado alto.

La distribución de los microinterruptores y la distancia que recorra la base móvil para activarlos, debe ser tal que al menos uno de ellos se encienda cuando se ejerza la fuerza mínima bajo la cual se considera que la persona está haciendo contacto con la plataforma, sin importar el lugar en donde se realice dicho contacto. Asimismo, deben instalarse resortes que restablezcan la posición original de la base móvil en el instante en que la persona deje de ejercer presión sobre ella, para que el microinterruptor regrese a su estado de apagado. Así, la distancia que recorra la base móvil estará determinada por la fuerza (F) ejercida por la persona sobre la plataforma, y por la constante del resorte utilizado. En la figura 3.3 se muestra el corte de una plataforma hipotética que utilizara los elementos que se describieron.

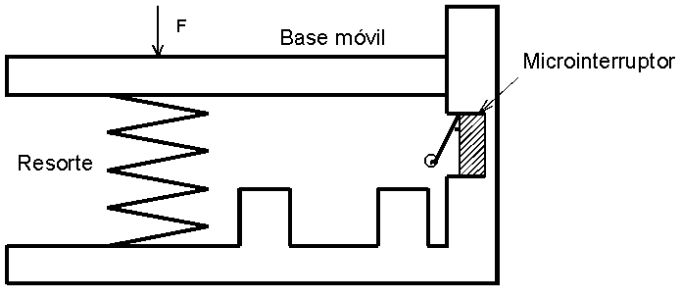


Figura 3.3 Plataforma con microinterruptor.

Los microinterruptores son comerciales y de bajo costo; sin embargo, es difícil determinar con exactitud el instante en que aquéllos pasan de estado encendido a estado apagado, además de que dicho cambio de posición es distinto para cada uno de ellos. También habría que realizar distintas pruebas para determinar la distancia de recorrido de la base móvil

dependiendo de las características de los resortes usados, o bien, adquirir los resortes bajo pedido con las características físicas exactas requeridas.

- **Plataforma con sensor de flexión**

El sensor de flexión funciona bajo el principio de una galga extensiométrica resistiva, en la que la resistencia del material depende de su resistividad, longitud, y área de la sección transversal. Cuando el sensor se encuentra totalmente distendido aparece una resistencia nominal en sus terminales; en cambio, cuando se somete a un esfuerzo provocando su flexión, la resistencia del mismo cambia proporcionalmente a la deformación sufrida. Este sensor se muestra en la figura 3.4 (http://www.jameco.com).

Figura 3.4 Sensor de flexión.

Con base en esto, puede diseñarse un sistema tal que una fuerza ejercida sobre la plataforma provoque un desplazamiento de la base móvil, la cual produzca a su vez la flexión del sensor. Igualmente, con la ayuda de resortes restituidores, podrían obtenerse distintos grados de flexión del sensor, dependiendo de la distancia que se haya desplazado la base móvil, la cual dependerá tanto de la fuerza ejercida sobre la plataforma, como de las características de los resortes. Asimismo, es necesario cuidar la distribución de los sensores, de tal forma que cuando se ejerza la fuerza mínima sobre la plataforma bajo la cual se considera que existió contacto, se produzca la flexión necesaria para generar un voltaje de salida alto, sin importar el lugar exacto en donde se llevó a cabo el contacto. La figura 3.5 muestra el corte de una plataforma con el diseño descrito.

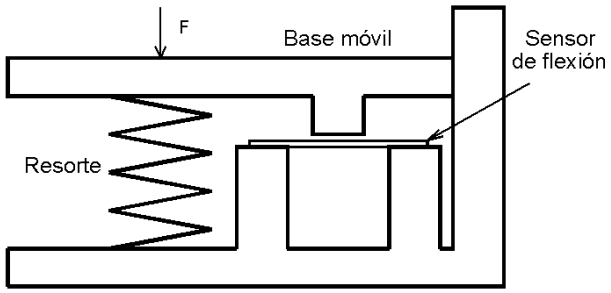


Figura 3.5 Plataforma con sensor de flexión.

A partir de lo anterior, podría diseñarse un circuito tal que con una fuente constante de corriente, se obtengan distintos valores de voltaje a partir de los diferentes valores de resistencia que pudieran presentarse en el sensor. Debido a que la señal de salida de dicho circuito sería una señal analógica, se necesitaría utilizar un convertidor análogo-digital para digitalizar la señal, o bien utilizar un comparador cuya señal de salida fuera un 1 lógico a

partir del instante en que se presente el desplazamiento mínimo, y un 0 lógico para todo desplazamiento menor.

- **Plataforma con sensor de presión**

Siguiendo la misma idea para el diseño de la plataforma, se plantea el uso de un sensor de presión como el que se muestra en la figura 3.6 (http://www.jameco.com). Este sensor es un circuito integrado que cuenta con un pequeño tubo por el cual se introduce un líquido o un gas, cuya presión es medida. Dicho sensor genera un voltaje como señal de salida directamente proporcional a la presión del fluido. Para emplear dicho sensor en la plataforma, podría construirse un compartimento que contuviera un fluido, y el cual se acoplara al tubo del circuito integrado.

Figura 3.6 Sensor de presión.

Tomando en cuenta el principio de Pascal que establece que "la presión aplicada a un fluido encerrado se transmite sin disminución a cada punto del fluido y a las paredes del recipiente", si se colocara dicho compartimento dentro de la plataforma, por debajo de la base móvil, la fuerza ejercida sobre ésta cuando una persona hiciera contacto con ella provocaría un incremento de la presión del fluido, la misma que podría ser detectada por el sensor y traducido un voltaje de salida del mismo. Gracias a dicho principio, sería indistinto el lugar en donde la persona hiciera contacto sobre la plataforma siempre y cuando éste fuera sobre la parte móvil. Un sistema como el anteriormente descrito se muestra esquemáticamente en la figura 3.7.

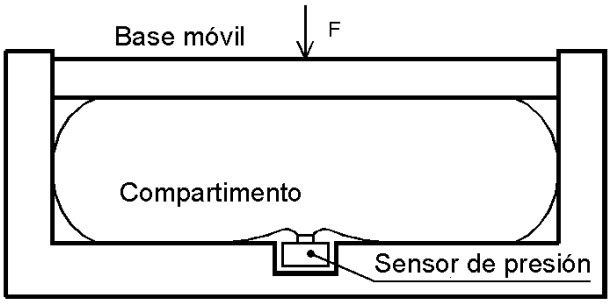


Figura 3.7 Plataforma con sensor de presión.

Al igual que en el caso del sensor de flexión, la señal de salida es de tipo analógico, por lo que también habría que utilizar un convertidor análogo-digital o un comparador para digitalizar la señal. Quizá la parte más compleja de este sistema es la caracterización del sensor y la construcción del compartimento. Se tendría que construir una bolsa lo suficientemente resistente para que soportara la presión ejercida sin que presentase fugas ni

elongaciones, para lo cual no sólo el material tendría que ser muy resistente, sino que tendría que acoplarse perfectamente al sensor. Además, habría que realizar un análisis de compresibilidad de fluidos, y su variación bajo distintas condiciones ambientales, como altura y temperatura, por ejemplo. También habría que llevar a cabo un análisis detallado de las condiciones de operación del sensor, tales como rango de presión y carga máxima, de tal forma que pudiera ser utilizado bajo las condiciones de operación del sistema.

3.2.2 Detección de contacto con base en la presencia sobre la plataforma

El diseño de una plataforma que opere bajo este principio, se basa en el uso de sensores optoelectrónicos para detectar la presencia de un individuo que esté colocado sobre la plataforma. Cuando la luz incide en la base de un fototransistor (figura 3.8, http://sharp-world.com) polarizado adecuadamente, se produce una corriente de colector que es directamente proporcional a la intensidad de luz incidente. Si se utiliza una fuente emisora de luz direccionable y lo suficientemente intensa para generar dicha corriente de colector en el fototransistor, como lo puede ser un láser (figura 3.9, http://www.calpaclasers.com), entonces puede diseñarse una distribución de este arreglo emisor-receptor de luz para construir un sensor de presencia. Así, cuando no haya ninguna persona situada en la plataforma, puede lograrse que la luz de un láser incida en el fototransistor provocando una corriente de colector que encienda al mismo; cuando la persona se coloque sobre la plataforma, el haz se verá interrumpido por la presencia de la misma, impidiendo que incida sobre el fototransistor, provocando un cambio de estado en él.

Figura 3.8 Fototransistor.

Figura 3.9 Láser.

Bajo este principio se consideraron dos diseños para colocar los sensores de manera que pudieran lograrse dichas detecciones de manera efectiva: en uno, se utilizan varios emisores láser y varios fototransistores; en el otro, se utiliza sólo un emisor láser y un fototransistor. En ambos casos, se elimina la parte móvil de la plataforma utilizada en los diseños basados en la presión ejercida sobre la misma, y únicamente se trabaja sobre la distribución del haz láser que permita, dentro de un área horizontal determinada, detectar que la persona hizo contacto con la plataforma.

- **Plataforma con sistema multi-láser**

En este caso, se considera una plataforma en uno de cuyos extremos se coloca, espaciada y paralelamente, una serie de emisores láser cuyo haz apunta hacia el extremo opuesto de la misma, en donde a su vez se coloca igual número de fototransistores espaciados a la misma distancia que los emisores, de tal forma que el haz emitido por cada láser incida en cada uno de los fototransistores. De esta manera, varios haces recorrerán paralelamente la plataforma de extremo a extremo, cubriendo el área horizontal de la misma, como se muestra en la figura 3.10.

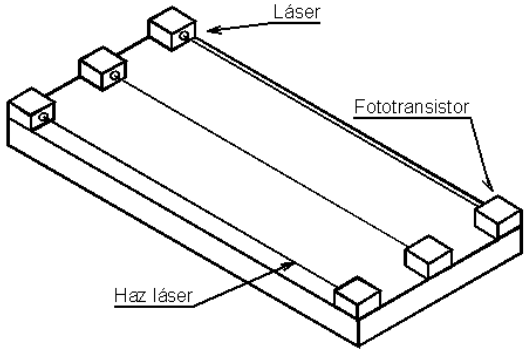


Figura 3.10 Plataforma con sistema multi-láser.

La distancia de separación entre los emisores y los fototransistores debe de ser tal, que si una persona coloca su pie paralelamente a la trayectoria de los haces, al menos uno de ellos se vea interrumpido. De esta manera se asegura que en cualquier parte de la plataforma en donde se coloque la persona, al menos uno de los fototransistores detectará la presencia de la misma. La salida del fototransistor será una señal digital cuyas características dependerán de la polarización del fototransistor.

- **Plataforma con sistema mono-láser**

Con la misma idea de utilizar dispositivos optoelectrónicos para el diseño de la plataforma, se pensó en diseñar un arreglo que utilice un solo emisor láser y un fototransistor. Para que pueda detectarse la presencia de la persona en cualquier parte de la plataforma, debe lograrse que el haz emitido por ese único emisor cubra con su trayectoria la mayor parte de la superficie horizontal de aquélla, antes de incidir sobre el fototransistor. Para ello, se hace uso de un par de espejos que se colocan en los extremos de la plataforma. El emisor láser se coloca en una esquina de la misma, a un lado de uno de los espejos, de tal forma que el haz emitido incida sobre el espejo que se encuentra en el extremo opuesto. Si al láser se le da una inclinación oblicua hacia el interior de la plataforma, se logra que el haz reflejado en el espejo opuesto se dirija directamente al espejo que se encuentra junto al emisor. De esta forma el haz se reflejará varias veces en los espejos hasta que finalmente llegue al fototransistor que se encuentra en la otra esquina, en el mismo extremo de la plataforma que el láser. La figura 3.11 ilustra el sistema descrito.

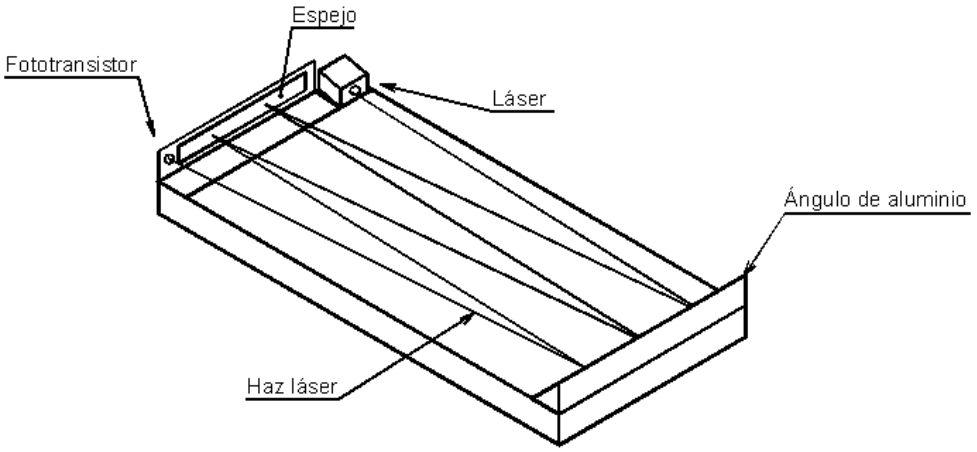


Figura 3.11 Plataforma con sistema mono-láser.

Si se varía el ángulo con que incide el haz láser en los espejos, puede incrementarse o decrementarse el número de haces que recorren de lado a lado la plataforma. Con una correcta elección de este número de haces, no habrá ningún lugar en la plataforma en el que una persona coloque su pie sin que corte la trayectoria de alguno de ellos, y por lo tanto, sin que sea detectada por el fototransistor.

3.2.3 Elección final del diseño

A continuación se comentan las implicaciones, ventajas y desventajas de cada uno de los diseños planteados en esta sección, a fin de encontrar el diseño que mejor se ajuste a las necesidades del presente proyecto.

En los diseños basados en la detección de contacto con base en la presión ejercida sobre la plataforma, se encuentra que la principal desventaja radica en la utilización de los elementos mecánicos que intervienen en ellos. La inclusión de una parte móvil en la plataforma, exige la búsqueda de un mecanismo que combine el perfecto desplazamiento de dicho elemento, sin que se presenten atascos, con la resistencia y robustez que se necesita en el mismo.

En cualquiera de los tres diseños comentados, es necesario determinar un rango de medición en el que se establezca la mínima fuerza que se va a ejercer sobre la plataforma, a partir de la cual se considera que existió contacto con la misma.

Para el caso del diseño que utiliza el microinterruptor y el sensor de flexión, dicho límite depende directamente de las propiedades del desplazamiento de la base móvil y por lo tanto del resorte utilizado. Es importante señalar que el cambio de estado de un microinterruptor de este tipo, sólo puede determinarse con base en prueba y error para cada uno de los microinterruptores utilizados, lo cual no asegura ni precisión ni exactitud. De cualquier forma, en ambos casos el hecho de que siempre que exista contacto éste sea detectado,

depende directamente de la distribución de los sensores y de la calidad en la construcción de la plataforma.

En resumen, aunque los sensores de flexión y los microinterruptores no tienen un costo elevado, el inconveniente radica principalmente en la utilización de elementos mecánicos que generalmente tienen una exactitud, precisión, vida media y velocidad de respuesta, menor al de los elementos electrónicos.

Para el caso del diseño que utiliza el sensor de presión, también involucra una parte móvil en la plataforma, y en este caso también es muy importante que ésta se desplace libremente, a fin de no obtener lecturas de presión erróneas. Aunque con este sensor puede mejorarse notablemente la exactitud y precisión de las lecturas, la construcción del compartimento que almacena al fluido y su acoplamiento al circuito integrado no resulta una labor fácil, ya que se requiere considerar factores como deformaciones, resistencia y durabilidad.

En los diseños de la plataforma basados en la detección de contacto con base en la presencia, se solucionan muchos de los inconvenientes que se plantearon anteriormente.

Por un lado, los sensores son puramente electrónicos, y se disminuye la complejidad de los elementos mecánicos, lo que en general le confiere mayor confiabilidad al sistema. En este caso no existen elementos móviles en la plataforma que sean determinantes para generar la señal de salida. La plataforma debe de soportar el peso de una persona cuando cae al realizar un salto vertical, por lo que tiene que ser muy rígida e indeformable, ya que una pequeña deformación de la misma puede cambiar la trayectoria del haz láser, y evitar que incida sobre el fototransistor cuando debe hacerlo. Sin embargo, siempre es más fácil construir una estructura con materiales resistentes cuando es completamente estática que cuando contiene elementos que se desplazan. Además, en este caso, no es necesario establecer un mínimo de fuerza aplicada sobre la plataforma para determinar que existió contacto, ya que siempre que la persona toque la plataforma, por ligero que sea el contacto, su presencia será detectada.

No obstante, los diseños con elementos optoelectrónicos deben de incluir un sistema de calibración bastante preciso que permita ajustar la posición horizontal y vertical con que incide el haz del láser sobre el fototransistor. Además, deben contemplarse los efectos que la luz ambiental puede tener sobre este último, a fin de no generar señales erróneas.

El sistema multi-láser utiliza varios sensores independientes, cada uno de los cuales consume energía y requiere de un mecanismo de calibración propio. Con el sistema mono-láser se reduce el número de sensores a utilizar, lo que implica una reducción del consumo general de energía y del número de elementos electrónicos susceptibles de sufrir algún tipo de fallo. Además, con este último diseño puede incrementarse la densidad de haces sobre la plataforma con solo cambiar el ángulo de incidencia del haz sobre los espejos, sin necesidad de incrementar los emisores ni los receptores. Sin embargo, debido a que se incluyen elementos muy frágiles como los espejos, debe cuidarse la robustez de los elementos de protección de los mismos. Aunque en el sistema mono-láser sólo se requiere ajustar la posición vertical y horizontal de un haz y no de varios como en el diseño multi-

láser, el mecanismo de calibración debe ser mucho más exacto, dado que el haz recorre varias veces la plataforma de lado a lado.

Así, considerando todo lo anteriormente mencionado, y tomando en cuenta características como exactitud, precisión, vida media, velocidad de respuesta, costo, durabilidad, y viabilidad de la construcción de la plataforma, se considera al sistema mono-láser como el que mejor resuelve la etapa de entrada al sistema, y por lo tanto es el que se elige para desarrollar el presente proyecto.

3.3 DESPLIEGUE Y ALMACENAMIENTO

En el capítulo anterior se presentaron tres dispositivos que permiten visualizar los resultados obtenidos de las pruebas.

La utilización de una *PC*, puede ser una opción para procesar la señal digital proveniente de los sensores, así como para desplegar los resultados. Mediante una herramienta de programación con ambiente gráfico como Visual Basic o Visual C, los resultados podrían desplegarse de forma clara y comprensible para cualquier usuario. Sin embargo, una de las características más importantes del dispositivo, es que debe ser fácilmente transportable, y las pruebas deben poder realizarse en espacios abiertos. Incluso considerando la utilización de una Laptop, se hace deseable la búsqueda de un dispositivo portátil más cómodo que permita procesar y desplegar los resultados de las pruebas en cualquier lugar. Sin embargo, la utilización de una *PC* no queda descartada para almacenar información, una vez que ésta fue recabada por otro dispositivo en el lugar en donde se llevó a cabo la prueba.

Otra opción para desplegar los resultados puede ser un *LCD*. El *LCD* maneja voltajes TTL que pueden conectarse a los puertos de un microcontrolador. Como una característica importante, puede mencionarse que este dispositivo cuenta con líneas de datos y líneas de control para manejar sus funciones. Si la información recibida a través de las líneas de datos, es la información que va a visualizarse en la pantalla, ésta se guarda en el registro de datos; si en cambio es la información que determina el formato que va a presentarse en la pantalla, se guarda en el registro de control. Por medio de las líneas de control, puede determinarse si la información que va a ser enviada por las líneas de datos es de datos o de control, así como seleccionar las funciones de lectura y escritura o de habilitación del despliegue. Aunque existen varias opciones para el formato con que se despliegan los datos, este dispositivo no permite manipular ni almacenar los resultados obtenidos. Por ello, es necesaria la búsqueda de otro dispositivo de despliegue, que además de mostrar la información, permita almacenar los valores.

Con base en lo anterior, la utilización de una *Palm*, en la que se puedan recibir y desplegar los resultados de las pruebas de manera organizada en el momento en que éstas se realizan, que permita almacenar los resultados y transferirlos a una *PC* para su análisis posterior, y que sea lo suficientemente pequeña para poder llevarse cómodamente a cualquier lugar, cumple con los requisitos que se necesitan para llevar a cabo la última etapa del proyecto.

Existen varias herramientas que permiten desarrollar aplicaciones en la *Palm*, y aunque muchas de ellas son muy interesantes y cuentan con diversos recursos, debe seleccionarse aquélla que mejor se ajuste a las necesidades del proyecto. En lo particular, se requiere que la aplicación desarrollada sea capaz de recibir los resultados calculados por el procesador, desplegarlos en el mismo momento en que éstos fueron recibidos, y almacenarlos y transferirlos a una base de datos de una *PC* para su posterior análisis.

La herramienta *GCC/PRC* se muestra como una opción interesante para el desarrollo de aplicaciones. Sin embargo, es una herramienta que presenta algunos problemas de aprendizaje, por lo cual, sería deseable encontrar otra herramienta que pudiera ser empleada rápidamente.

Code Warrior es una herramienta que utiliza lenguaje C para crear aplicaciones y que por lo mismo cuenta con mucho potencial. Permite manipular el puerto serie de la *Palm*, y posee una amplia gama de recursos para diseñar la aplicación. Es una herramienta en la que tienen que ser programados cada uno de los elementos que van a necesitarse.

Satelite Forms, es una herramienta bastante poderosa a pesar de no requerir muchos conocimientos en cuestiones de programación. Es capaz de manipular los puertos para recibir información, y crear enlaces entre una base de datos creada en la *Palm* y una base de datos creada en la *PC*. Sin embargo, se requiere una licencia distinta para cada *Palm* que se utiliza.

Por último, *Pendragon Forms* es una herramienta con pocos recursos y pocas variantes para la creación de aplicaciones; sin embargo, está creada específicamente para la adquisición de datos. Con esta herramienta, por cada forma descargada en una *Palm*, se crea una base de datos en la *PC* basada en Microsoft Access, en el que cada vez que se hace una sincronización entre la *Palm* y la *PC*, los registros se actualizan tanto en una como en la otra. A pesar de no ser una herramienta muy poderosa, se ajusta perfectamente a las necesidades específicas del proyecto, y por lo tanto es la plataforma elegida para crear la aplicación en la *Palm* que permita recibir los datos provenientes del procesador.

Una aplicación creada en la *Palm* mediante este software, consiste en un conjunto de "formas" en las que se encuentran contenidos "campos" que almacenan información. Cada una de las "formas" representa una base de datos distinta enfocada a objetivos particulares. Cada uno de los "campos" puede tener distintas características dependiendo de la información que manipule. Además, pueden crearse varios registros para una misma "forma", en los que la distribución y características de los "campos" no cambian, pero la información que se guarda en ellos sí.

Cada uno de los "campos" de las "formas" contiene un área de programación en la que pueden realizarse distintas operaciones que involucren a otros "campos" o a los recursos de la *Palm*, y que se ejecutan en el momento en que se accede a ellos.

La *Palm* utiliza un puerto serial para comunicarse con el mundo exterior. *Pendragon Forms* permite recibir y enviar valores bajo el código ASCII a través de dicho puerto. Por lo tanto, el dispositivo de procesamiento utilizado en el proyecto, debe ser capaz de entablar una

comunicación serial asíncrona bajo el estándar RS-232-C, además de interpretar correctamente un número expresado en código ASCII.

3.4 PROCESAMIENTO

En las dos últimas secciones de este capítulo se definieron los dispositivos de entrada y de salida del sistema. Con base en ellos, debe elegirse el elemento de procesamiento que mejor se acople a dichos dispositivos, y que permita dar una solución eficiente al cálculo de los resultados de las pruebas. A continuación, se presentan algunas opciones para la solución de dicha etapa.

Como se mencionó en la sección anterior, una *PC* podría funcionar como dispositivo de procesamiento y de salida. La señal digital proveniente del sensor podría introducirse a ella mediante alguno de sus puertos de comunicación, y con el uso de algún lenguaje de programación de alto nivel, la señal podría recibirse y procesarse, a fin de obtener los resultados requeridos. Sin embargo, se requiere que el dispositivo sea transportable, por lo que conviene buscar otra solución para dicha etapa.

También podría pensarse en utilizar la *Palm* como el dispositivo que directamente reciba la señal proveniente de los sensores y calcule los resultados de las pruebas. Sin embargo, la mayor parte de la literatura para el desarrollo de aplicaciones para la *Palm*, está enfocada a utilizar la *Palm* como despliegue y almacenamiento de datos, no como unidad de procesamiento. Además, para recibir la señal de entrada proveniente de los sensores, ésta tendría que ser convertida al formato de comunicación serial asíncrona y bajo el estándar RS-232-C para que pudiera ser introducida a la *Palm*, lo que le restaría eficiencia a la aplicación.

Por todo lo anterior, se considera que un microcontrolador es el dispositivo ideal para realizar esta etapa del sistema, ya que, a diferencia de la *Palm*, sus recursos son transparentes al programador, y está enfocado al procesamiento de señales, lo que permite desarrollar programas eficientes y tener un control completo sobre la tarea que se está llevando a cabo. El microcontrolador puede colocarse en una tarjeta la cual se instalaría en la misma plataforma. La señal proveniente del sensor podría introducirse sin modificaciones relevantes al microcontrolador, y los resultados podrían transmitirse a la *Palm* mediante una comunicación serial asíncrona.

Para elegir el microcontrolador más conveniente para el proyecto planteado, es necesario realizar un análisis de las necesidades y de las características del sistema. Inicialmente, el microcontrolador recibirá la señal digital de entrada proveniente del sensor, calculará con base en ella la frecuencia de paso y el tiempo transcurrido entre contactos, y enviará los resultados a la *Palm*, y para entonces ya debe estar preparado para recibir la siguiente señal de los sensores. Por lo tanto, puede considerarse que es un sistema que trabaja en tiempo real, lo que significa que el tiempo para llevar a cabo ciertas rutinas es limitado, y que por lo tanto la velocidad con que se ejecuten las instrucciones es muy importante. Además, se considera que el programa no es de considerable complejidad, y que con algunas

instrucciones básicas en lenguaje ensamblador, el programa puede desarrollarse eficientemente.

Los recursos que el microcontrolador debe tener según las necesidades del proyecto, se analizan a continuación. Como la señal que se recibe de los sensores es de tipo digital, ésta puede introducirse al microcontrolador directamente, siempre y cuando los valores de corriente y voltaje sean los adecuados para recibir la señal, lo cual puede lograrse con algunos componentes electrónicos. Así, es necesario considerar uno o dos bits de entrada para recibir la señal proveniente de las dos plataformas, y un par de bits para comunicar a la *Palm* con el microcontrolador, por lo que basta con un puerto de comunicación para realizar estas funciones.

Por otro lado, como se requiere contabilizar tiempo y número de pasos, es deseable que el microcontrolador incluya un contador y un temporizador con Preescalador (ver Apéndice, sección A.2.2), a fin de ajustar la frecuencia de reloj a la más conveniente.

Ya que no se almacenarán valores permanentemente, los requerimientos de memoria tan solo deben ser los suficientes para albergar al programa y almacenar las variables temporales que éste utilice.

Por último, ya que se comunicará al microcontrolador con la *Palm* por medio de una comunicación serial asíncrona, sería recomendable contar con un puerto de comunicación que permita transmitir y recibir valores con ese formato.

El PIC 16F84A, en adelante PIC, es un microcontrolador con arquitectura Harvard y procesador RISC. Cuenta con 35 instrucciones, cada una de las cuales se ejecuta en un ciclo de instrucción que dura 1μs, excepto las que involucran saltos, que emplean 2 ciclos de instrucción. Cuenta con 1024 palabras de memoria de programa, 68 bytes de memoria de datos en RAM, 64 bytes de memoria de datos EEPROM, 13 líneas de entrada/salida, y un Temporizador/Contador de 8 bits, con preescalador programable de 8 bits.

Se considera que las características de este microcontrolador son suficientes para dar solución a la etapa de procesamiento del sistema. Quizá pudo pensarse en utilizar un microcontrolador que incluyera un módulo para la comunicación serial asíncrona, pero debido a que se contaba con bastante experiencia en la utilización de este PIC, con la literatura correspondiente, con las tarjetas de programación y de prueba, y dado que el formato de transmisión serial asíncrona puede realizarse con una cuantas líneas de programación, se consideró conveniente utilizar este PIC, y realizar la programación manual de la transmisión y recepción de datos.

CAPÍTULO 4

DISEÑO DEL SISTEMA

4.1 INTRODUCCIÓN

En el capítulo anterior se describieron de manera general los elementos que constituyen el sistema, y las razones por las cuales se eligieron cada uno de ellos, gracias a lo cual ahora se cuenta con un panorama global del funcionamiento del dispositivo.

En el presente capítulo se describirá a detalle el diseño y la construcción de cada una de las etapas del sistema, además de las interfaces que conectan al dispositivo de procesamiento con los dispositivos de entrada y de salida.

4.2 PLATAFORMA DE MEDICIÓN

A continuación se describe el diseño de la plataforma de medición junto con los materiales que la conforman. También se detallan los elementos que constituyen el sensor de presencia, y cómo estos se encuentran instalados en la plataforma para detectar el instante en que una persona hace contacto con ella. Finalmente se puntualiza sobre el funcionamiento electrónico del láser y del fototransistor, y sobre las características de la señal de entrada que este último genera, además de cómo ésta debe de ser modificada para que pueda ser interpretada correctamente por el PIC.

La plataforma de medición consta de dos bases de metal idénticas. En el caso de la prueba de carrera corta, se colocan cada una de ellas en los extremos del recorrido. Para el caso de la prueba de salto vertical y frecuencia de paso, se juntan ambas plataformas longitudinalmente conformando una sola, duplicando el área en donde la persona puede realizar la prueba y permitiendo que, si la persona coloca un pie sobre cada una de las plataformas, pueda llevarse la cuenta independiente del número de pisadas para cada pie. Cada una de las plataformas independientes (figura 4.1) están conformadas por un cuadro de solera de acero de sección rectangular, y en cuyo interior se suelda

longitudinalmente una barra del mismo material para darle rigidez a la estructura. Una lámina de 0.002 m de espesor se suelda sobre dicho cuadro, y sobre ella se pega asimismo un material plástico antideslizante.

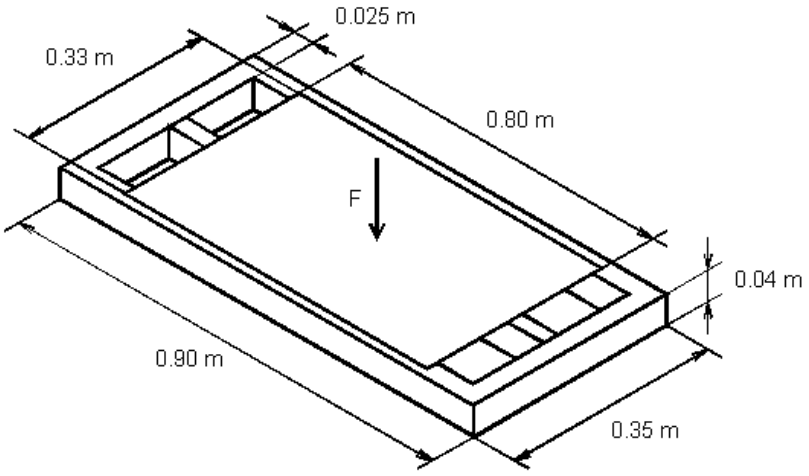


Figura 4.1 Estructura de la plataforma de medición.

Cada una de las pruebas se lleva a cabo sobre el área definida por la lámina soldada al cuadro. Con la estructura descrita, se busca que no existan deformaciones plásticas en el cuadro construido con la solera, que lleguen a modificar permanentemente la posición con que el haz incide en el fototransistor cuando una persona realice alguna de las pruebas. En el caso de que existiera una deformación elástica de la plataforma, ésta no tendría mayores consecuencias, ya que se presentaría únicamente al caer la persona sobre la misma, momento en que la luz del láser está bloqueada por el pie del individuo; al retirarse la persona, la luz láser recobraría su posición original, e incidiría nuevamente en el fototransistor.

Para comprobar que se cumplen las condiciones anteriormente mencionadas, se lleva a cabo el siguiente análisis. Si se supone, en primera instancia, que la plataforma se coloca sobre un terreno uniforme, de tal manera que el área inferior de la misma está completamente en contacto con el suelo, y se aplica una fuerza vertical (F) sobre la plataforma como se indica en la figura 4.1, entonces, como reacción a dicha fuerza, existirá un conjunto de fuerzas ejercidas por el suelo sobre la plataforma, en la misma dirección y en sentido opuesto a la fuerza aplicada. En este caso, la única posibilidad de deformación de la solera, es a causa de la compresión que en un momento dado pudiera generarse en el material a causa de la fuerza aplicada.

No obstante, supóngase el caso extremo en el que se coloca la plataforma de medición, o en particular, una de las barras longitudinales de la plataforma, sobre dos puntos de apoyo, y se ejerce una fuerza (F) sobre el punto medio de la barra, como se muestra en la figura 4.2.

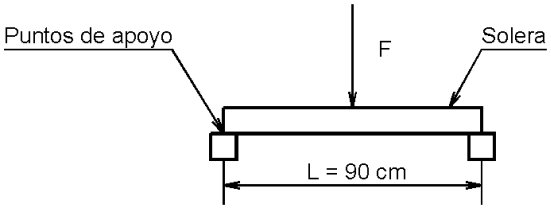


Figura 4.2 Barra de solera con dos puntos de apoyo sometida a una fuerza central.

El esfuerzo σ, al cual se encuentra sometida la barra debido a la fuerza F, se obtiene mediante la expresión:

$$\sigma = \frac{M\left(\frac{h}{2}\right)}{I}$$

en el que h es la altura de la solera, M el momento flexionante máximo provocado por la fuerza F y los puntos de apoyo, e I el momento de inercia de la sección transversal de la solera, que pueden calcularse de la siguiente manera:

$$M = \frac{F \cdot L}{8}$$

$$I = 2\frac{e\,h^3}{12} + 2\left[\frac{(b-2e)e^3}{12} + \left(h-\frac{e}{2}\right)^2 (b-2e)e\right]$$

en el que e, h y b, corresponden a las medidas de la sección transversal de la solera que se muestra en la figura 4.3.

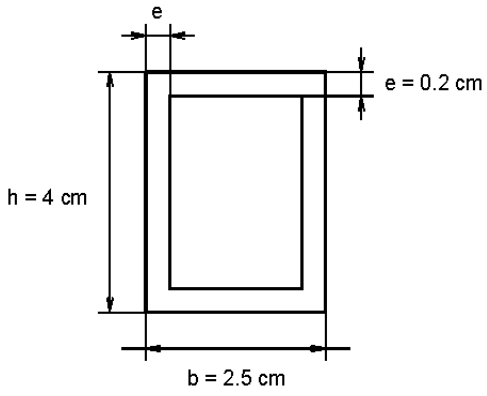


Figura 4.3 Sección transversal de la solera.

Si se supone que sobre la plataforma salta una persona con un peso de 200 kgf, y que al impulsarse y al caer ejerce una fuerza de tres veces su peso, entonces puede

considerarse que F = 600 kgf. Sustituyendo los valores en las correspondientes ecuaciones, se encuentra que I = 14.91 cm^4, el momento flexionante máximo M = 6750 kgf·cm; y por lo tanto, el valor del esfuerzo ejercido sobre la barra en sus extremos es de $\sigma = 905.28\ kgf/cm^2$.

Ahora bien, según estudios realizados, el mínimo esfuerzo necesario que se necesita ejercer sobre el acero para que éste sufra una deformación plástica, tiene un valor aproximado de $\sigma = 2500\ kgf/cm^2$, y como se observa, el esfuerzo al que se somete la plataforma bajo condiciones extremas, es notablemente inferior a ese valor. Cabe resaltar, que analizar una sola de las barras y sacar conclusiones de la plataforma completa a partir de ella, es una aproximación bastante burda, ya que existen otros elementos que contribuyen a soportar el peso de la persona. Sin embargo, en cuestión de análisis de materiales, se considera como regla empírica que los resultados obtenidos a partir de aproximaciones burdas brindan resultados conservadores de la estructura real o del material analizado. Por lo tanto, puede asegurarse que si la barra de solera no sufre deformaciones según el análisis realizado, la plataforma es lo suficientemente resistente para llevar a cabo las pruebas físicas sin que el sistema sufra alteraciones en su funcionamiento.

Finalmente, en una de las paredes exteriores de la solera, se practica en ambas plataformas tres perforaciones que van de lado a lado. En los orificios de una de las plataformas se insertan tres tornillos, a fin de que al juntarlas longitudinalmente para realizar la prueba de frecuencia de paso y salto vertical, se introduzcan los mismos por los orificios practicados en la otra, sujetándolas así mutuamente y apretándolas con una "mariposa", que es una especie de tuerca que puede aflojarse y apretarse fácilmente. Adicionalmente, se instala una agarradera en la pared exterior de la solera opuesta a la que se practicaron las perforaciones para que la estructura pueda transportarse fácilmente.

También es necesario diseñar una estructura especial para realizar la prueba de carrera corta. Dado que en dicha prueba la persona va de una plataforma a otra a gran velocidad, se espera que utilice a la plataforma como apoyo para frenarse e impulsarse para iniciar el recorrido de regreso. Por ello, es recomendable dotar a la plataforma de una cierta inclinación a fin de que la persona pueda apoyarse en ella, y que se encuentre fija para que no se desplace bajo la acción del contacto. Así, se diseñan tres estructuras triangulares de solera de acero que, por un lado, se acoplan a la plataforma por medio de tornillos de tal forma que brinden a la misma un ángulo de 30°, y por otro, se anclan al suelo para evitar que la plataforma se desplace cuando la persona ejerza fuerza sobre ella. La estructura descrita se muestra en la figura 4.4.

Cada una de las plataformas tiene un arreglo formado por dos espejos, un emisor láser y un fototransistor. Cada espejo se coloca en la parte vertical de un ángulo de aluminio, de tal forma que la parte horizontal del mismo se fija sobre la solera que conforma cada uno de los lados menores de la base de metal, quedando colocada la parte pulida de cada espejo hacia el interior de la plataforma.

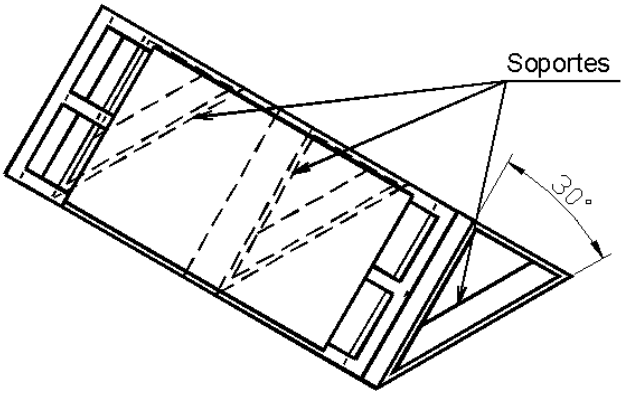


Figura 4.4 Estructura de soporte.

Uno de los ángulos de aluminio es de menor longitud ya que a su lado se coloca el emisor láser. Éste último, se encuentra instalado en una base cúbica de aluminio en posición horizontal, apuntando al espejo de mayor longitud, con una incidencia ligeramente oblicua al interior de la plataforma, de tal forma que, si se mantienen los espejos perfectamente en posición vertical, el haz de luz se refleje consecutivamente de lado a lado de la plataforma, cubriendo con su trayectoria un plano horizontal. La base cúbica cuenta con una perforación cilíndrica de lado a lado, en donde se introduce y fija el emisor láser. Como la base cúbica mencionada abarca un área horizontal mayor que el ancho de la solera, para poder instalarla sobre la plataforma se utiliza una lámina que se fija al cuadro, y sobre la cual se coloca la base. Cabe mencionar que en el prototipo realizado se utilizó madera para construir la base que contiene al láser, a fin de reducir costos.

En el extremo opuesto del emisor láser, al otro lado del espejo de menor longitud, se hace una perforación en el ángulo de aluminio en donde se introduce un tubo de plástico negro, en cuyo interior se coloca el fototransistor. La trayectoria final del láser, después de varias reflexiones en los espejos, incide sobre el tubo de plástico negro y se introduce en él hasta llegar al fototransistor. Dicho tubo de plástico tiene la finalidad de impedir que la luz exterior incida sobre el fototransistor y provoque un cambio de estado en el mismo sin que la luz del láser haya incidido en él. La figura 4.5 muestra el dispositivo descrito.

La plataforma cuenta con un mecanismo de calibración con dos grados de libertad para lograr que el haz láser incida directamente en el fototransistor. Por un lado, permite variar la inclinación de cada espejo para mantener horizontal la trayectoria del haz en todo su recorrido, evitando que en algún momento una de las reflexiones no incida sobre alguno de los espejos, y lograr así ajustar la posición vertical final con la que el haz incide sobre el fototransistor. Por otro lado, la base que contiene al emisor láser puede girar sobre un plano horizontal, modificando el ángulo de incidencia del rayo sobre los espejos, lo que permite modificar la densidad de haces que cubren un plano horizontal sobre la plataforma, al igual que la posición horizontal con que el rayo incide en el fototransistor.

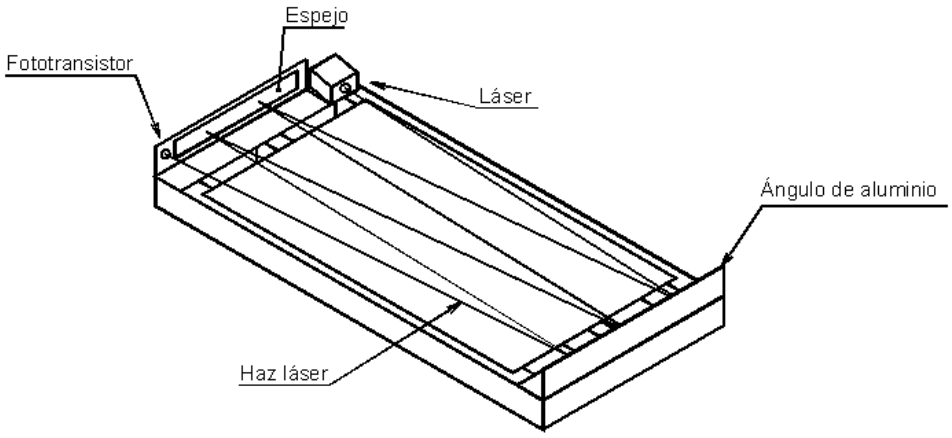


Figura 4.5 Imagen completa de la plataforma diseñada.

Para ajustar la inclinación de los espejos, se sujeta el ángulo de aluminio a la plataforma mediante dos hileras de tornillos a lo largo del mismo. Una hilera de tornillos, se coloca lo más cerca posible de la parte vertical del ángulo de aluminio, y la otra hilera lo más lejos posible. En el sitio donde está colocado cada tornillo, se instala una rondana de presión entre el ángulo de aluminio y la plataforma, lo que permite aumentar o reducir la separación que existe entre ellos, sin perder rigidez en la fijación entre los dos elementos. Así, si se aprietan los tornillos que se encuentran más alejados al espejo y se aflojan los que se encuentran más cercanos, el ángulo de aluminio, y por lo tanto el espejo, si inclinará hacia adelante, haciendo que el haz de luz incidente se refleje con cierto ángulo hacia abajo con respecto a la trayectoria de incidencia. Inversamente, si se aflojan los tornillos que se encuentran más alejados del espejo, y se aprietan los que están más cercanos, el haz se reflejará hacia arriba con respecto a la línea de incidencia. La figura 4.6 muestra el sistema descrito.

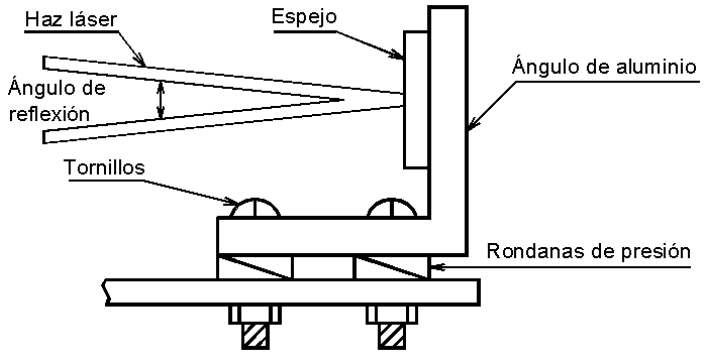


Figura 4.6 Mecanismo de calibración de la posición vertical de incidencia del haz láser.

Por otra parte, la base cúbica que contiene al emisor láser, cuenta con un eje vertical que lo atraviesa de arriba abajo en una de sus esquinas y que se fija en la plataforma. Al hacer girar la base alrededor de dicho eje, puede ajustarse la posición horizontal de

incidencia del haz sobre el fototransistor. En la esquina opuesta al eje vertical mencionado, se coloca un tornillo que también atraviesa la base de arriba a abajo, pero que además atraviesa por medio de una ranura la lámina sobre la cual está instalada la base. Esta ranura permite desplazar el tornillo a lo largo de ésta cuando se está ajustando la posición de la base, y una vez hallado el ángulo de inclinación correcto, se utiliza una tuerca para sujetar la base contra la lámina, evitando que se mueva de la posición final determinada.

Así, el emisor láser se ajusta para que el haz emitido se refleje 3 veces en el espejo de mayor longitud y dos veces en el espejo de menor longitud antes de incidir sobre el fototransistor, logrando así un total de 6 haces que recorren de lado a lado la plataforma, lo que hace que la separación máxima entre dos haces de luz contiguos sea de 0.11 m en la zona cercana a los espejos. Tomando en cuenta que dicha distancia de separación decrece al alejarse de los espejos, se considera que la densidad de haces elegida es suficiente para que, aún en el peor de los casos en el que la persona coloque el pie longitudinalmente a un lado de los espejos, alguno de los haces se vea interrumpido siempre que la persona toque la plataforma. Figura 4.7.

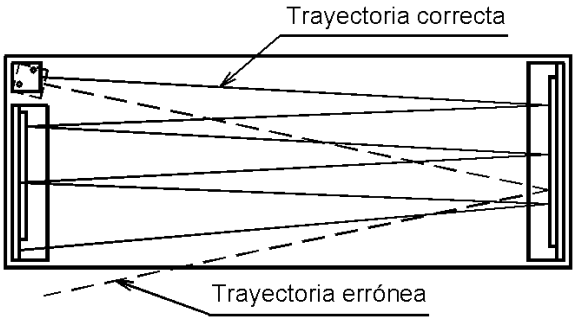


Figura 4.7 Mecanismo de calibración de la posición horizontal de incidencia del haz láser.

La unidad de procesamiento encargada de calcular los resultados de las pruebas, se instala en una tableta de madera que se sujeta en la parte inferior de la plataforma, en el espacio que existe entre la lámina y el suelo. Dicha tableta de madera, se sujeta a las paredes internas de la solera, sin que toque la lámina, a fin de que no se transmitan vibraciones directamente de la lámina, ni se transmita calor de la plataforma de metal a la estructura. Asimismo, circuito electrónico se cubre con una protección de acrílico para aislarla del polvo y de la humedad del suelo, más aún si se utiliza sobre hierba.

Con fines de protección, se diseña una cubierta de aluminio que se coloca sobre los ángulos de aluminio y que se atornilla a la plataforma. En las paredes de dichas cubiertas que dan hacia el interior de la plataforma, se practican unos pequeños orificios a fin de que el haz láser pueda reflejarse en los espejos. Esta estructura de protección, tiene la finalidad de evitar que accidentalmente puedan desajustarse o incluso dañarse seriamente los espejos o el emisor láser durante la realización de las pruebas, además de evitar que la luz exterior incida sobre los espejos, provocando reflexiones de luz no

deseadas que puedan incidir sobre el fototransistor. En la figura 4.8 se muestra la plataforma de medición con dichas piezas de protección.

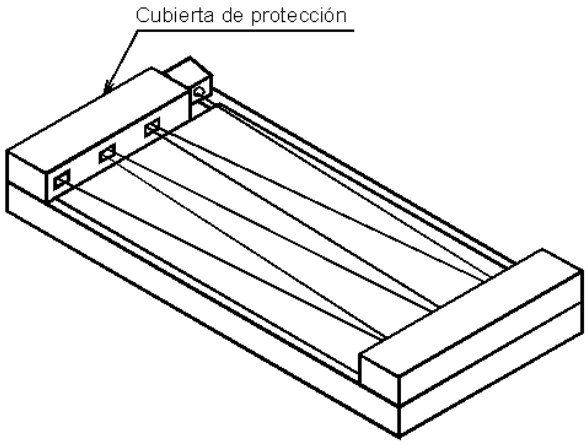


Figura 4.8 Plataforma de medición con la cubierta de protección.

Por último, el plano W01 muestra el ensamble del ángulo de aluminio de menor longitud con su respectivo espejo, el láser, y el fototransistor, a la base de metal previamente descrita.

4.3 PROCESAMIENTO

En la presente sección se hará una descripción del dispositivo de procesamiento del sistema. Por un lado, se comentará el programa desarrollado para procesar la señal de entrada proveniente de los sensores y que permite obtener los resultados de las pruebas; por otro, se describirán los dispositivos electrónicos que acompañan al PIC y que en conjunto conforman la unidad de procesamiento.

4.3.1 Software

Como se mencionó en el capítulo anterior, se decidió utilizar un PIC 16F84A para llevar a cabo la etapa de procesamiento del sistema. A continuación se enumeran las funciones específicas que el PIC debe realizar:

- Con el uso del Temporizador propio del PIC y la conexión de la salida del sensor a una terminal de uno de sus puertos, un registro se incrementa cada 500 μs a partir del instante en que el fototransistor cambia su estado de apagado a encendido (instante en que el pie de la persona se separa de la plataforma), y se detiene cuando el fototransistor pasa de encendido a apagado (instante en que el pie se apoya sobre la plataforma). El valor almacenado en el registro representa el tiempo en el que un jugador fue de una plataforma a otra para el caso de la prueba de la carrera corta, o el tiempo que se mantuvo en el aire para el caso de la prueba de salto vertical.

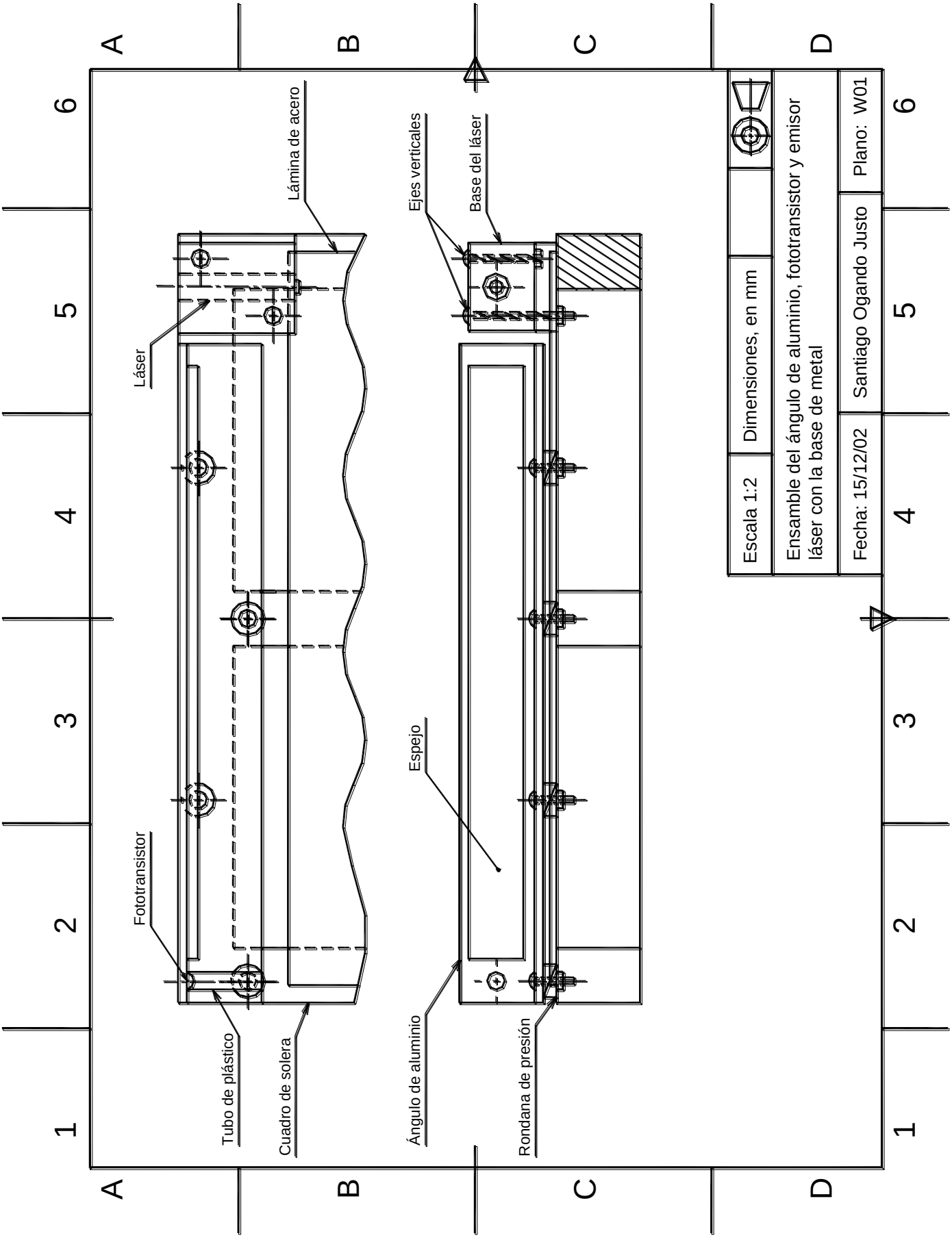


Lámina de acero
Ejes verticales
Base del láser
Láser
Espejo
Fototransistor
Tubo de plástico
Cuadro de solera
Ángulo de aluminio
Rondana de presión
Escala 1:2
Dimensiones, en mm
Ensamble del ángulo de aluminio, fototransistor y emisor láser con la base de metal
Fecha: 15/12/02
Santiago Ogando Justo
Plano: W01

- Por medio de una recepción serial asíncrona, el PIC recibe por otra de las terminales de uno de sus puertos, el tiempo durante el cual va a realizarse la prueba de frecuencia de paso, introducido a través de la *Palm*.
- Para el caso de la prueba de frecuencia de paso, se incrementa un registro cada vez que uno de los fototransistores pasa de estado encendido a estado apagado durante el tiempo recibido de la *Palm*, a partir del instante en que ambos fototransistores se encienden por primera vez (instante en que la persona levanta los pies para iniciar la prueba). El valor almacenado en el registro representa el número de pasos que la persona dio en el tiempo establecido.
- Por medio de otra terminal de sus puertos, envía hacia la *Palm* de forma serial asíncrona, el valor de los registros que representan el tiempo calculado y los pasos contabilizados, para su despliegue.

A continuación se describe la secuencia del programa desarrollado en el PIC 16F84A para calcular los resultados obtenidos por las pruebas.

En primer lugar, el programa recibe un valor proveniente de la *Palm* que indica si la prueba que se va a realizar es la de salto de altura y carrera corta o de frecuencia de paso. Si se va a realizar la prueba de frecuencia de paso, el PIC se prepara nuevamente para recibir otro valor de la *Palm*, el cual representa esta vez el tiempo durante el cual se va a realizar la prueba. Como todos los valores provenientes de la *Palm* están bajo el código ASCII, se desarrolló un algoritmo que permite transformar un número en dicho código a un valor hexadecimal, el cual se describe en el apartado A.2.4 del Apéndice del presente trabajo. Posteriormente, el programa se mantiene en estado de espera hasta detectarse el instante en que la persona se sube a la plataforma, después de lo cual se queda nuevamente en estado de espera hasta que el sensor indica que la persona ya no está situada sobre ella.

Cabe señalar, que cada vez que el PIC detecta un cambio de estado en la línea por donde entra la señal proveniente del sensor, vuelve a verificar después de 1 ms que realmente existió dicho cambio de estado, a fin de eliminar falsas detecciones del sensor. Dicho valor se determinó experimentalmente y se comprobó que, por un lado, efectivamente se detecta un cambio en el estado del sensor sólo cuando verdaderamente alguien se coloca o se retira de la plataforma, y por otro, que no es un retraso de tiempo lo suficientemente grande para que afecte el funcionamiento global del sistema.

Cuando el fototransistor envía la señal que indica que la persona ha retirado ambos pies de la plataforma, se habilita el Temporizador, con lo que empezará a decrementarse cada 500 μs la variable que contiene el tiempo durante el cual se va a realizar la prueba. A partir de ese mismo instante y hasta que la variable que contiene el tiempo de realización de la prueba llegue a cero, cada vez que exista un contacto sobre la plataforma, se incrementará otra variable. Al final de la prueba, se envía a la *Palm* el valor almacenado en la variable que contiene el número de contactos que se realizaron sobre la plataforma, e inmediatamente el programa regresa al estado en que espera que la persona se coloque sobre la plataforma para iniciar otra prueba. Si se quiere realizar la prueba con otro tiempo de prueba, es necesario reiniciar el programa.

Si la prueba a llevarse a cabo es la de carrera corta o la de salto vertical, una vez que se reconoce que la prueba a realizar es una de estas dos, el programa pasa directamente al estado de espera hasta que se detecta cuándo la persona se coloca sobre la plataforma; ya que lo hizo, espera hasta que deja de haber contacto, que es el instante en el que se inicia el salto o la carrera. Una vez que eso sucede, se habilita el Temporizador con lo que se comenzará a incrementar una variable cada 500 μs. Cuando el sensor indica que la persona ha regresado a la plataforma, entonces se inhabilita el Temporizador y se transmite hacia la *Palm* el valor almacenado en la variable que se incrementó y que representa el tiempo que la persona estuvo en el aire o que tardó en llegar a la siguiente plataforma. Una vez transmitido el valor, el programa regresa al estado de espera en que se detecta el instante en que la persona se sitúa sobre la plataforma para comenzar una nueva rutina; con ello, se logra que puedan realizarse varios saltos o varias carreras de ida y vuelta consecutivas, y que el tiempo registrado se envíe a la *Palm* cuando finalice cada una de ellas, quedando el programa listo para registrar el siguiente intervalo de tiempo. Para dar comienzo a una nueva prueba, se debe reiniciar el PIC, y pasar nuevamente por cada una de las etapas descritas.

El programa completo y algunas especificaciones del mismo, se mostrarán a detalle en el apartado A.2 del Apéndice del presente trabajo.

4.3.2 Hardware

La figura 4.9 muestra el diagrama general del circuito electrónico que conforma la unidad de procesamiento.

Como se observa, el diagrama general puede dividirse en cuatro bloques principales: microcontrolador, transmisión y recepción entre el PIC y la *Palm*, sensores y circuito de reinicio o *reset*. A continuación se describirán cada uno de estos bloques.

En el bloque que contiene al PIC se muestran las líneas de entrada y salida, y la conexión con los elementos externos necesarios para que el PIC funcione. Como se observa, la señal digital de entrada proveniente del sensor entra por la terminal 10 del chip, que se refiere al bit 4 del puerto B (RB4); la transmisión hacia la *Palm* de los valores calculados se realiza a través el bit 3 del puerto A (RA3) que se encuentra en la terminal 2 del encapsulado; y la recepción de los valores también provenientes de la *Palm* se hace mediante la terminal 1 en donde se encuentra el bit 2 del puerto A (RA2). En la terminal 4 se encuentra el *reset* externo del PIC, que está conectado a la salida de un contador, y el cual provoca que se reinicie el PIC cada vez que se genera una transición de alto a bajo. Por último, mediante la utilización de un cristal de 4 MHz y dos condensadores de 18 pF, se genera una señal cuadrada de igual frecuencia que el cristal, a fin de proveer al PIC de una frecuencia de oscilación.

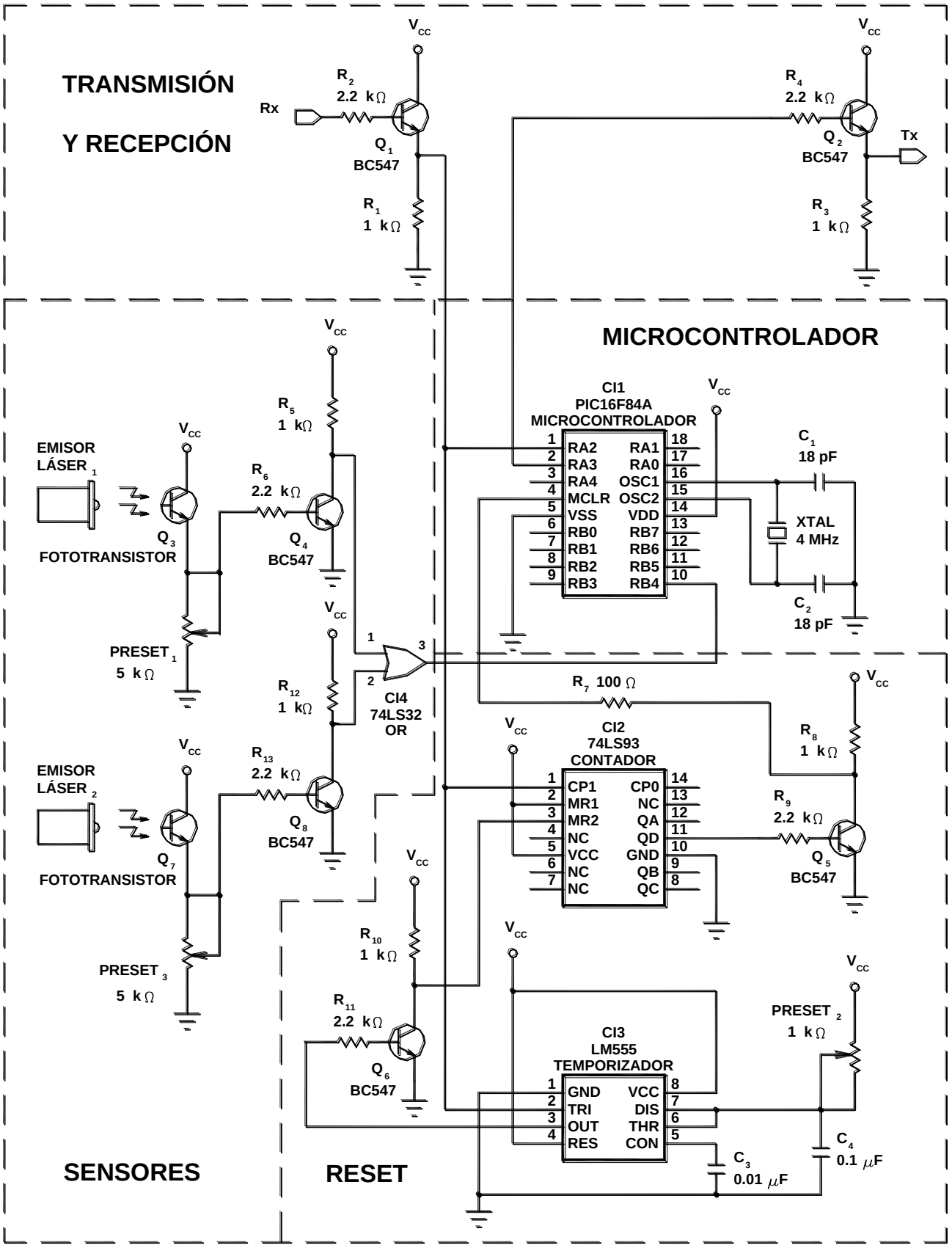


Figura 4.9 Diagrama general del circuito electrónico.

En el bloque referente a la transmisión y recepción de valores entre la *Palm* y el PIC, se muestran un par de transistores (Q_1 y Q_2) BC547 polarizados como transistores de conmutación con configuración de colector común. Con los valores de resistencia seleccionados, se asegura que para un voltaje de base mayor a 0.9 V, se alcancen los voltajes de saturación de colector-emisor (V_{CEsat}) y de base-emisor (V_{BEsat}), lo que permite tener a la salida un voltaje de 5 V; cuando el voltaje de base es menor a 0.9 V, el transistor entra en corte y el voltaje en la salida es 0 V. Con esto siempre se logra tener una señal digital cuadrada, tanto en la recepción como en la transmisión, a fin de evitar falsas lecturas.

En la etapa de los sensores puede observarse la conexión de los fototransistores con el fin de generar una señal digital que se introduce en el PIC. El fototransistor Q_3, por ejemplo, está conectado bajo la configuración de colector común. Cuando la luz incide en el fototransistor se genera una corriente de emisor que genera un voltaje en las terminales del *preset*$_1$. El transistor Q_4 está polarizado como transistor de conmutación como el descrito anteriormente, salvo que esta vez tiene configuración de emisor común, lo que significa que la señal de salida está invertida con respecto a la señal de entrada; cuando el voltaje en las terminales del *preset*$_1$ es superior a los 0.9 V, el voltaje de salida en Q_4 es de 0 V, y cuando es inferior a los 0.9 V, el voltaje en la salida de Q_4 es de 5 V. Así, el *preset*$_1$ se ajusta a 1.8 kΩ de tal forma que bajo la luz ambiental el nivel de voltaje en las terminales del mismo está entre 0 V y 0.5 V, en cambio cuando incide la luz del láser se alcanza un nivel de voltaje de salida alrededor de los 2 V. Con esto, logra generarse una señal digital con niveles TTL de manera que pueda ser interpretada por el PIC correctamente.

Sin duda alguna, el hecho de que los sensores puedan generar la señal digital adecuada sin errores, depende directamente de que los estados de encendido y apagado estén perfectamente bien definidos, es decir, que la diferencia de voltaje a la salida del fototransistor sea grande entre los instantes en que se tiene la presencia del láser y cuando únicamente se tiene la presencia de la luz ambiental. Para ello, juega un papel muy importante el tubo de plástico negro que contiene al fototransistor en la plataforma, ya que evita que la luz ambiental incida directamente en el fototransistor, disminuyendo el voltaje de salida, al mismo tiempo que permite que la luz láser llegue hasta el fototransistor sin problema alguno. Un aumento de la potencia nominal y una disminución de la dispersión del láser, ayuda a incrementar dicha diferencia de voltaje y por lo tanto mejora el funcionamiento general del dispositivo. Por ello, se recomienda emplear el láser VLM-650-06-LPA, sobre el cual se hablará más a detalle en la sección A.1 del Apéndice.

Posteriormente se utiliza una compuerta OR 74LS32 en cuya entrada se introducen las señales enviadas por los sensores, y cuya salida se envía a la terminal RB4 del PIC. Esta compuerta OR, se utiliza para que el PIC detecte la presencia de una persona sobre la plataforma sin importar en cual de las dos se realizó el contacto. Esto implica que, en el caso de la prueba vertical, el inicio del salto se detectará hasta que la persona haya despegado ambos pies de la plataforma, y finalizará cuando el primer pie haga contacto con la misma en la caída; en el caso de la prueba de frecuencia de paso, se llevará la cuenta del momento en que cada pie haga contacto con la plataforma, teniendo que

levantar forzosamente uno de los dos pies antes de bajar el otro. Obviamente para la prueba de carrera corta, se detectará el instante en que la persona llega a cada una de las plataformas.

En el último bloque se muestra un circuito de reinicio o *reset* externo, que reinicia al PIC desde la *Palm* cuando ésta envía un valor específico, y que permite al usuario finalizar una prueba y comenzar una nueva. Para lograr reiniciar el PIC, es necesario generar un pulso bajo con una duración mayor de 2 μs, lo cual se logra utilizando un temporizador LM555 y un contador binario de cuatro bits, el 74LS93.

El principio de funcionamiento del circuito de *reset* se basa principalmente en el código ASCII con que son enviados los valores de la *Palm* al PIC. Toda la información que la *Palm* transmite hacia el PIC, está conformada únicamente por números que van del 0 al 9, y que se transmiten con un bit de inicio bajo, una cadena de 8 bits que representan el dígito enviado, y un bit de paro alto. Si se cuenta el número de transiciones de estado alto a bajo en cada una de las cadenas de bits que representan estos números, se encuentra que existen desde 1 hasta 3, al contar a partir del bit de inicio y hasta el bit de paro; ahora bien, si se cuenta el número de estas transiciones que hay en la cadena de bits que representan la letra "U" se verifica que son 4, lo que nos permite poseer un parámetro distintivo entre el código de la "U" y el código de los otros números. Lo anterior se aprecia en la tabla 4.1.

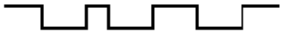
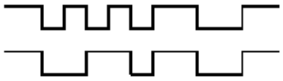
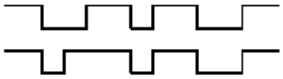
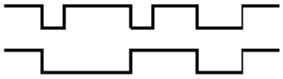
Número decimal	Código ASCII	Cadena transmitida	Número de transiciones
0	30		1
1	31		2
2	32		2
3	33		2
4	34		2
5	35		3
6	36		2
7	37		2
8	38		1
9	39		2
U	55		4

Tabla 4.1 Cadenas que se transmiten entre el microcontrolador y la Palm.

Todo dato enviado por la *Palm*, además de ser enviado directamente al bit RA2, es introducido, por un lado, al reloj del contador, y por otro a la entrada del temporizador, que al detectar un bit de inicio de una cadena de bits, genera un pulso monoestable de una duración igual al tiempo de byte de la cadena recibida. Dicho pulso monoestable habilita al contador durante el tiempo que dure la cadena de bits, tiempo en que

contabilizará el número de veces que exista una transición de estado alto a bajo en la cadena de bits recibida. Un transistor de conmutación (Q_6) es colocado entre la salida del temporizador y el habilitador del contador, a fin de mantener una señal cuadrada. Cuando se termina el tiempo de byte, el contador se limpia y se inhabilita hasta que otro bit de inicio es detectado.

Como el contador se configuró en modo de cuenta de tres bits, el bit más significativo de éste sólo se encontrará en estado alto cuando el valor enviado por la *Palm* sea una "U", ya que es el único caso de entre todos los valores enviados en el que existe un código con cuatro transiciones. Si además se conecta dicho bit del contador a la base de un transistor de conmutación (Q_5) con configuración emisor común, ya descrito anteriormente, en el momento de presentarse la cuarta transición se generará un pulso bajo que durará hasta que se deshabilite el contador. De esta forma, si se conecta la salida del transistor a la terminal de *reset* del PIC, entonces éste se reiniciará cada vez, y únicamente, cuando la *Palm* transmita una "U" hacia el PIC.

El consumo total de energía del sistema, incluyendo los bloques anteriormente descritos, el consumo del láser, y el consumo de una interfaz entre el PIC y la *Palm* que se explicará posteriormente, es de 70 mA. Para suministrar energía al sistema, se utiliza una pila recargable de 7 V con una potencia de 1200 A/h. Para obtener el voltaje de operación de los circuitos integrados empleados, se utiliza un regulador LM7805, que proporciona 5 V a la salida, que es el voltaje de alimentación (Vcc) utilizado.

4.4 DESPLIEGUE Y ALMACENAMIENTO

El almacenamiento y despliegue de los resultados calculados por el PIC es la última etapa del sistema. A continuación se describirán las características más importantes del modelo de *Palm* que se empleó en el proyecto, así como la base de datos que se desarrolló para recibir y desplegar los valores provenientes del PIC. Asimismo, se mostrará la interfaz empleada para llevar a cabo la comunicación entre la *Palm* y el PIC.

4.4.1 Software

Para que la *Palm* pudiera recibir los datos provenientes del PIC, fue necesario desarrollar una base de datos con *Pendragon Forms*. Dicha base de datos permite llevar un control sobre la prueba que se está realizando, además de almacenar y desplegar la información de manera clara y cómoda, para que el entrenador pueda realizar una primera evaluación en el momento de realizar la prueba.

Los requerimientos para que *Pendragon Forms* pueda instalarse en una *PC* son: Windows 95/ Windows 98/ Windows NT 4.0, 32 MB en RAM y 25 MB de espacio libre en disco duro. Aunque Microsoft Access no es indispensable para instalar el software en una *PC*, si es necesario para este proyecto.

Los requerimientos de la *Palm* necesarios son: Versión de *Palm* OS 3.0 o mayor, 2 MB en RAM y controlador *HotSync* 3.0 o mayor. Los modelos de *Palm* que cumplen con estas características son: *Palm* III, *Palm* IIIx, *Palm* V, *Palm* Vx, *Palm* VII, Symbol SPT-1500 e IBM WorkPad y posteriores.

El programa realizado en la *Palm* lleva a cabo las siguientes funciones:

- Recepción de los valores provenientes del PIC.
- Conversión de los valores recibidos a tiempo transcurrido, altura saltada y frecuencia de paso.
- Almacenamiento de los valores recibidos en una base de datos y presentación de la misma de forma ordenada y clara.
- Cálculo de parámetros estadísticos con base en los resultados de las pruebas.
- Envío al PIC, mediante una transmisión serial asíncrona, del valor del tiempo durante el cual se realizará la prueba de frecuencia de paso.
- Selección de la prueba a realizar.
- Control sobre el reinicio del PIC.

Con el software ya mencionado se diseñó una base de datos en una *PC*. Dicha base de datos consiste en un conjunto de "formas" que contienen una serie de "campos" para almacenar información. Para cada una de las pruebas se diseñó una "forma" distinta, según los parámetros que cada una de ellas requiere manejar. Por ejemplo, una de las "formas" puede ser la prueba de salto vertical, dentro de la cual se encuentran "campos" que guardan información, como puede ser el nombre del jugador, su edad, su peso, y la altura que saltó al realizar una prueba. Existen distintos tipos de "campos" según el tipo de información que manejan, y cada uno de ellos contiene un área de programación, conocida como "script", enfocada a realizar una operación específica que involucre a otros "campos" o a los recursos propios de la *Palm*, y que se ejecutan en el momento en que se accede a ellos.

Después de que se ha diseñado la base de datos, las "formas" y su contenido se descargan en la *Palm*, en donde los "campos" de cada una pueden ser llenados o modificados. Además, pueden crearse varios registros para una misma "forma", de tal manera que el diseño y los "campos" que contiene la misma no cambien, pero que la información almacenada en ellos sea diferente. Una vez recolectada la información en uno o varios registros, puede sincronizarse la *Palm* con la *PC* mediante el puerto serie, haciendo que la información recolectada en la primera se transfiera a una base de datos tipo Access en la segunda. De esta forma, se tendrá una base de datos contenida en una *Palm* cuyos valores se pueden recopilar fácilmente en cualquier lugar, y una vez terminada la adquisición, dichos valores pueden actualizarse en la *PC* para ser analizados posteriormente. En la figura 4.10 se muestran las "formas" diseñadas para cada una de las pruebas tal y como aparecen en la *Palm*, las cuales se describirán a continuación.

Figura 4.10 "Formas" de las pruebas de carrera corta, salto vertical y frecuencia de paso.

En el primer "campo" de cualquiera de las "formas" se muestra el nombre del deportista al cual se le está aplicando la prueba. Para introducir el nombre de la persona, éste puede escribirse directamente en la zona de escritura de la pantalla de la *Palm*, o bien hacer clic sobre ese "campo", con lo que se despliega una lista previamente almacenada del nombre de varias personas. Dicho "campo" es de tipo "Búsqueda en lista" (look up list), especial para desplegar listas previamente realizadas.

En el caso de la "forma" de carrera corta, el siguiente "campo" almacena el valor de la distancia que va a recorrerse, para posteriormente calcular la velocidad media del recorrido. Al igual que en el "campo" anterior, el valor de la distancia puede introducirse escribiendo directamente en la *Palm*, o bien eligiendo entre las que ya se encuentran previamente definidas en una lista.

En el caso de la "forma" de frecuencia de paso, el siguiente "campo" permite introducir el tiempo durante el cual va a realizarse la prueba, el cual también puede llenarse de las dos maneras ya explicadas. El siguiente "campo" de esta "forma", cuenta con un botón de inicio de la prueba, que al oprimirse, reinicia el PIC y le envía el tiempo introducido en el "campo" anterior. A partir de este momento, todas las pruebas de frecuencia de paso deberán realizarse en el tiempo ya establecido, ya que si este cambia, las estadísticas finales de las pruebas arrojarán valores erróneos. Si se quiere realizar la prueba con otro tiempo de duración, debe de crearse un registro nuevo.

Los siguientes "campos" son comunes a todas las pruebas y son aquéllos que reciben y almacenan los resultados. Originalmente están diseñados para recibir valores provenientes de un lector de código de barras conectado al puerto serie de la *Palm*, pero de igual forma sirven para conectarse a cualquier dispositivo que utilice una comunicación serial asíncrona a una velocidad 9600 bits/s, sin bit de paridad, 8 bits de datos y 1 bit de paro. Cuando se hace clic sobre alguno de estos "campos", automáticamente se abre el puerto serie de la *Palm* y se prepara para recibir los valores transmitidos por el PIC y almacenarlos en dicho "campo".

En el caso de las pruebas de carrera corta y salto vertical, después de que se ha recibido el conjunto de palabras que representan el resultado de una prueba, el PIC envía un carácter especial que al ser detectado por la *Palm* provoca que el próximo resultado enviado se almacene en el siguiente "campo". Esto hace que puedan realizarse varias pruebas consecutivas y que el resultado de cada una de ellas se almacene en un "campo" distinto. Aunque en las figuras de las tres "formas" mostradas sólo aparecen dos "campos" de almacenamiento de datos, la aplicación fue diseñada para recibir un máximo de 10 valores, los cuales aparecen al irse recibiendo los resultados correspondientes a dicho "campo".

Es importante señalar que los datos recibidos por la *Palm* sufren un último procesamiento antes de ser presentados como los resultados finales de las pruebas. Si bien los datos en código ASCII son convertidos automáticamente por la *Palm* en los caracteres que equivalen, los datos recibidos provenientes del PIC están en sistema hexadecimal, aunque estén representados por caracteres decimales. Por lo tanto, debe llevarse a cabo tanto la conversión de sistema hexadecimal a decimal para procesar los resultados recibidos, como la conversión de sistema decimal a hexadecimal, para transformar el tiempo durante el cual va a realizarse la prueba de frecuencia de paso, antes de que sea enviado al PIC. Ambas conversiones se explican detalladamente en la sección A.3.2 del Apéndice.

También hay que recordar que en el caso de las pruebas de carrera corta y salto vertical, la cuenta de tiempo en los registros del PIC se realizó cada 500 μs, por lo que es necesario multiplicar el valor del tiempo recibido por 0.0005 para obtener el tiempo real en segundos. Asimismo, antes de enviar al PIC el tiempo durante el cual va a llevarse a cabo la prueba de frecuencia de paso, dicho valor de tiempo debe dividirse entre 0.0005.

Además, para la prueba de salto vertical, aún es necesario calcular la altura a partir del tiempo obtenido. Si se parte de la suposición que el movimiento de ascenso es muy similar al movimiento de descenso, se considera que el tiempo empleado para que la persona alcance la máxima altura, es la mitad del tiempo total transcurrido durante el salto. Por tanto, la altura máxima (h) alcanzada se obtiene a partir de:

$$h = (t/2)^2 * g/2$$

En la que "t" es el tiempo total de salto y "g" la aceleración de la gravedad en la Ciudad de México. Además, debe considerarse que la trayectoria del haz láser se encuentra en un plano horizontal a 0.025 m sobre la superficie de la plataforma, por lo que debe agregarse esa distancia a la altura calculada.

Por último, como producto de un proceso de calibración, será necesario introducir una última modificación a los resultados de las pruebas, la cual se describirá en la sección 4.5 del presente capítulo.

Todos estos cálculos se programan en el *script* de los "campos" Variable1 y Variable2 de cada una de las "formas", "campos" que permanecen ocultos al usuario.

En los siguientes dos “campos” de cada una de las tres “formas” se despliega la fecha y la hora en que se creó el registro sobre el que se está trabajando. Esto brindará al usuario una herramienta para organizar los registros cuando éstos sean transferidos a la *PC*.

A diferencia de las “formas” de salto vertical y de carrera corta, en la de frecuencia de paso el programa no se prepara para recibir un nuevo valor en el “campo” siguiente, sino que da por finalizada la prueba después de recibir un resultado. Esta “forma” se diseñó según lo anterior, porque no se espera que se realice este tipo de prueba consecutivamente. Si se desea realizar otra prueba en este mismo registro, puede oprimirse el botón rotulado como “Prueba nueva”, lo que hará que aparezca una nueva entrada de almacenamiento de resultado, y que permitirá realizar otra prueba con el tiempo establecido en la prueba anterior y sin necesidad de oprimir el botón de inicio.

Los siguientes cuatro “campos” son similares para las tres “formas”. En el primero se muestra un botón que despliega las estadísticas de la prueba. Esto tiene efecto cuando se han realizado varios ejercicios para una misma prueba. En los tres casos, al presionar el mencionado botón, se despliega el promedio, el mínimo y el máximo del total de valores obtenidos correspondientes a cada prueba. Para la prueba de carrera corta, además de estos tres parámetros, muestra los correspondientes para la velocidad media alcanzada. Es importante señalar que no deben quedar “campos” de adquisición de datos vacíos, ya que éstos se incluirán en el cálculo de las estadísticas.

Los siguientes dos “campos” también son botones. El primero sirve para borrar todos los valores almacenados durante la realización de la prueba, y para volver a realizarla sin necesidad de crear un nuevo registro; el segundo sirve para crear un nuevo registro de la misma “forma”.

Todos los registros creados quedan almacenados en la *Palm* y pueden consultarse o modificarse en cualquier momento. Posteriormente, si se conecta la *Palm* al puerto serie de la computadora, y se oprime el botón de sincronización de la misma, todos los registros y su contenido se transferirán a la *PC*, y quedarán almacenados en una base de datos distinta para cada una de las “formas”. En dicha base de datos, las columnas corresponden a cada uno de los “campos” creados, y los renglones a cada uno de los registros. En la figura 4.11 se muestra una sección de dicha base de datos.

Viewer for FORM_ID_217369141 : Formulario

RecordID	UnitID	UserName	TimeStamp	Jugador	Altura1	Altura2
0	0	unam	12:25:42 p.m.	Joaquín Beltrán	21.5	27.2
0	0	unam	12:27:02 p.m.	Miguel España	23.3	26.1
0	0	unam	12:30:40 p.m.	Horacio Sánch	24.7	24.2

Registro: 2 de 3

Figura 4.11 Base de datos en la PC.

Es importante mencionar, que gracias a que *Pendragon Forms* trabaja conjuntamente con Microsoft Access, pueden utilizarse las herramientas propias de este programa para cambiar el formato de la base de datos, crear filtros, y ordenar los valores con base en la información contenida en un "campo".

En la sección A.3 del Apéndice del presente trabajo, se muestra la programación realizada en los "campos" de cada una de las "formas".

4.4.2 Hardware

En esta sección se puntualizará sobre las características más importantes relacionadas con la *Palm* desde el punto de vista del hardware. Además, se mostrará la interfaz utilizada para conectar la *Palm* con el PIC.

La *Palm Vx* es el modelo utilizado en el presente proyecto, ya que cumple con los requerimientos del sistema y está dentro del conjunto de modelos de *Palm* en el cual puede instalarse *Pendragon Forms*. A continuación se mostrarán los principales recursos de la *Palm Vx* y se detallará sobre aquéllos que son importantes para el dispositivo.

- Versión de Palm OS: 3.5.0
- Memoria: 8 MB
- Memoria dinámica: 128 kB
- Pantalla: Negra y gris de 160 x 160 pixeles
- Puerto de comunicación infrarroja
- Software "HotSync"
- Batería de litio recargable
- Procesador: "DragonBall EZ"

A pesar de que en la actualidad existen *Palms* con memoria de 16 MB, y que la memoria de este modelo no puede ampliarse con tarjetas de expansión, se considera que 8 MB es una capacidad suficiente de memoria para cubrir las necesidades del presente proyecto. La aplicación completa de *Pendragon Forms* instalada en la *Palm*, sin contenido alguno, ocupa por sí misma 201 kB; cada una de las tres "formas" diseñadas, 11 kB; y cada registro, con información en cada uno de sus "campos", ocupa aproximadamente 1 kB.

"DeskTop", es un software que permite extender las funciones de la *Palm* a una *PC*, además de hacer copias de seguridad de los datos almacenados en el organizador. Utilizando la tecnología *HotSync*, todos los datos almacenados en la *Palm* pueden transferirse a la *PC*. Para llevar a cabo la sincronización de ambos dispositivos, se coloca la *Palm* en una base que contiene un cable con un conector DB-9. Dicho conector se conecta a un puerto serie libre de una *PC*, y se aprieta el botón de *HotSync* que se encuentra en la base mencionada.

En el caso de este proyecto, al llevar a cabo la sincronización de la *Palm* con la *PC*, se transferirá tanto la información general contenida en la misma, como las "formas" diseñadas para las pruebas junto con todos sus registros. Para que esto pueda lograrse, debe

de estar instalada la aplicación de *HotSync* en la *PC* por medio de la instalación de *DeskTop*, cuyos requerimientos mínimos son: Windows 95/ Windows 98/ Windows NT 4.0, computadora IBM-compatible 486 o superior, 8 MB de RAM, aunque se recomiendan 16 MB (requerida para Windows NT 4.0), 20 MB libres en disco duro, monitor VGA o superior, CD-ROM para la instalación del software, aunque puede descargarse de http://www.palm.com, o solicitarlo en discos de 3.5" en la compañía 3Com, y un puerto serial disponible.

La *Palm* utiliza una batería recargable que sólo necesita ser conectada unos cuantos minutos cada día para mantenerla cargada. Para recargar la *Palm*, únicamente debe de colocarse en la base, a la cual se le conecta un adaptador de AC, que a su vez se conecta al contacto eléctrico.

La *Palm Vx* utiliza el procesador "DragonBall EZ" que es un procesador 68EZ328 de Motorola, con recursos como temporizadores, puertos paralelos, moduladores de ancho de pulso, y controlador de LCD, entre otras. Sin embargo, dichos recursos no son transparentes al usuario, y no pueden usarse directamente.

Otro aspecto a destacar en esta sección, es la interfaz realizada entre la *Palm* y el PIC. Como ya se explicó, si bien la *Palm* y el PIC transmiten y reciben bajo el mismo formato de comunicación (comunicación serial asíncrona a una velocidad 9600 bits/s, sin bit de paridad, 8 bits de datos y 1 bit de paro), no manejan los mismos niveles de voltaje, ya que la primera lo hace bajo el estándar RS-232-C, y el segundo con niveles TTL, cuyas características se explicaron en el segundo capítulo del presente trabajo.

Para acoplar dichos niveles de voltaje, se emplea una interfaz MAX232C. Dicho circuito integrado maneja una línea bidireccional para la transmisión y otra para la recepción; un extremo de cada línea se conecta al PIC y otro lado a la *Palm*. Al transmitir un valor del PIC a la *Palm*, cuando a la entrada del MAX232C se tiene un voltaje de 5 V, a la salida se obtiene un valor de voltaje entre –12 V y –3 V; cuando a la entrada se tiene 0 V, a la salida se obtiene un valor de voltaje entre 3 V y 12 V. De igual forma, cuando se transmite un valor de la *Palm* al PIC, se lleva a cabo la misma conversión de valores de voltaje pero en sentido inverso. En la figura 4.12 se muestra la interfaz descrita.

Por otro lado, la base de sincronización de la *Palm* utiliza un cable con un conector DB-9 para comunicar su puerto serie con dispositivos externos, en el que cada línea tiene las siguientes funciones para la conexión de módem nulo, en relación con el diagrama mostrado en la figura 4.13.

Terminal 1: RLSD, detección de portador
Terminal 2: Rx, recepción de datos
Terminal 3: Tx, transmisión de datos
Terminal 4: DTR, la terminal de datos está lista
Terminal 5: GND, tierra
Terminal 6: DSR, dato preparado
Terminal 7: RTS, solicitud de transmisión
Terminal 8: CTS, listo para enviar

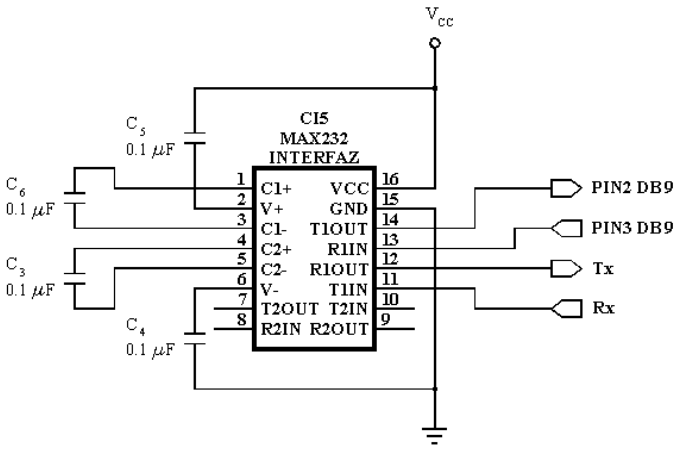


Figura 4.12 Interfaz MAX232C.

Como en este caso se está realizando una comunicación asíncrona, únicamente se emplean las líneas de transmisión, recepción y tierra, las cuales se conectan al MAX232C tal como se indica en la figura 4.12, para llevar a cabo la conversión a voltajes TTL.

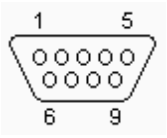


Figura 4.13 Conector DB-9.

4.5 CALIBRACIÓN

En esta última sección se describen una serie de experimentos que se llevaron a cabo una vez que se finalizó el prototipo del dispositivo diseñado. La idea es obtener muestras de los resultados obtenidos en cada una de las pruebas realizadas con el sistema construido, y compararlos con resultados considerados como reales obtenidos a partir de otro dispositivo de medición. Finalmente, a partir de dicha comparación se llevará a cabo un proceso de calibración para mejorar, cuando sea necesario, la exactitud de los resultados.

4.5.1 Experimento 1

El primero de los experimentos se realizó para determinar la exactitud con que el dispositivo mide el tiempo. Para ello, se utilizó la interfaz *Pasco Scientific*, modelo CI-6510 *signal interface*, y el software *Precision Timer* de Pasco, los cuales sirven para medir intervalos de tiempo. De esta forma, se conectó dicha interfaz a la plataforma de medición a fin de obtener lecturas de tiempos simultáneas en ambos dispositivos, y comparar así resultados.

Dado que el programa desarrollado en el PIC mide intervalos de tiempo entre un flanco de subida y un flanco de bajada (señal E), y el software mencionado los mide entre dos flancos de bajada, fue necesario diseñar una interfaz (figura 4.14) que generara un pulso bajo de 100 μs por cada cambio de nivel de la señal proveniente del sensor de la plataforma (señal S), y así lograr que ambos dispositivos inicien y finalicen los intervalos de tiempo en el mismo instante. Las señales descritas se muestran en la figura 4.15.

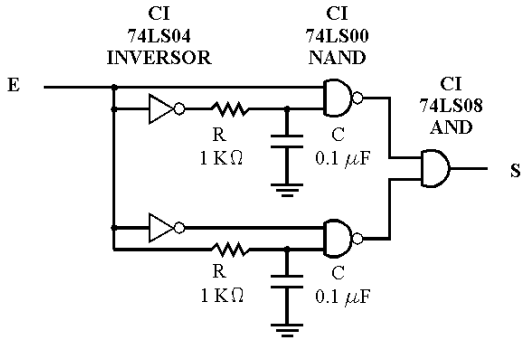


Figura 4.14 Interfaz para adecuar la señal de entrada del Precision Timer.

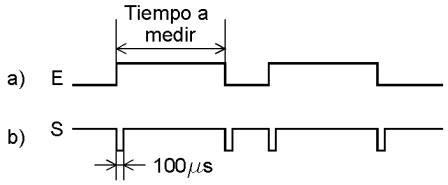


Figura 4.15 a) Señal proveniente de los sensores, b) Señal de salida de la interfaz generadora de pulsos.

Los resultados del experimento se muestran en la tabla 4.2.

Palm	0.281	0.230	0.214	0.188	0.192
Precision Timer	0.279	0.228	0.212	0.186	0.190
Palm	0.471	0.576	0.571	0.559	0.596
Precision Timer	0.470	0.575	0.570	0.558	0.596
Palm	1.051	1.094	0.945	0.901	0.977
Precision Timer	1.052	1.095	.0946	0.902	0.978
Palm	1.513	1.553	2.018	2.036	1.912
Precision Timer	1.516	1.557	2.022	2.041	1.916
Palm	5.851	6.045	6.536	5.503	5.707
Precision Timer	5.869	6.063	6.557	5.520	5.724
Palm	9.994	10.273	10.770	10.177	10.224
Precision Timer	10.026	10.307	10.806	10.150	10.258

Tabla 4.2 Resultados del experimento 1, en segundos.

Dado que el error obtenido en las mediciones es sistemático, y además puede representarse por medio de una función lineal, con base en los resultados se realiza un ajuste por el método de mínimos cuadrados, en donde se considera como variable independiente (x) los resultados obtenidos con el *Precision Timer*, y como variable dependiente (y) los resultados obtenidos con la plataforma de medición. La función resultante de dicho ajuste es:

$$y = 1.00127 + 0.99746\ x$$

Finalmente, con el fin de ajustar los resultados obtenidos por el PIC, se despeja la variable independiente, y se programa la función resultante en las "formas" de las pruebas de carrera corta y salto vertical.

4.5.2 Experimento 2

El segundo experimento que se realizó fue para verificar que el tiempo indicado en la *Palm* para realizar la prueba de frecuencia de paso, corresponde realmente al tiempo durante el cual el PIC cuenta los pasos dados sobre la plataforma. En el caso particular de esta prueba, dado que van a medirse tiempos más largos, no es necesaria tanta exactitud en los resultados como en las otras dos pruebas, por lo que la prueba para verificar el tiempo se realiza con un cronómetro manual.

La prueba consiste en indicar distintos valores de realización de prueba en la *Palm*, y después, con ayuda de un cronómetro manual, medir el tiempo desde que se retira un objeto de la plataforma, hasta que llega el resultado a la *Palm*. Los resultados que se obtuvieron se muestran en la tabla 4.3.

Palm	5	5	5
Cronómetro	5.55	5.75	5.74
Palm	10	10	10
Cronómetro	10.77	10.89	10.77
Palm	15	15	15
Cronómetro	15.75	15.8	15.77
Palm	20	20	20
Cronómetro	20.67	20.65	20.94
Palm	25	25	25
Cronómetro	25.94	25.78	25.84
Palm	30	30	30
Cronómetro	30.75	30.55	30.87

Tabla 4.3 Resultados del experimento 2, en segundos.

Como se observa en los resultados, se presenta un error sistemático cuya magnitud se mantiene prácticamente constante sin importar el tamaño del intervalo de tiempo medido. Es necesario tomar en cuenta que el cronómetro manual se accionó al escucharse un sonido que se produce cada vez que un resultado ha sido recibido por la *Palm*, por lo que hay que

considerar, por un lado, un retraso de tiempo desde que el PIC deja de contar los eventos ocurridos sobre la plataforma hasta que se recibe el resultado y se emite el sonido, y por otro lado, el tiempo en que la persona reacciona desde que escuchó el sonido hasta que detiene el cronómetro, que al menos es de 0.24 segundos, según la Asociación Internacional de Federaciones de Atletismo (IAAF).

Tomando en cuenta todo lo anterior, se considera que las condiciones de operación de esta prueba referente al tiempo de realización de las mismas, cumplen con las características adecuadas para obtener resultados confiables, y se hace innecesario algún tipo de ajuste.

4.5.3 Experimento 3

El último experimento es relativo a la prueba de salto vertical. Para evaluar los resultados obtenidos en esta prueba, se empleó un dispositivo mecánico para medir la altura máxima que alcanza una persona en un salto. Dicho dispositivo consta de un tubo vertical graduado cada centímetro, con una serie de aspas móviles a lo largo del mismo. Las aspas tienen una separación entre sí de 0.01m, y pueden girar una distancia alrededor del tubo si son empujadas. El dispositivo descrito se muestra en la figura 4.16.

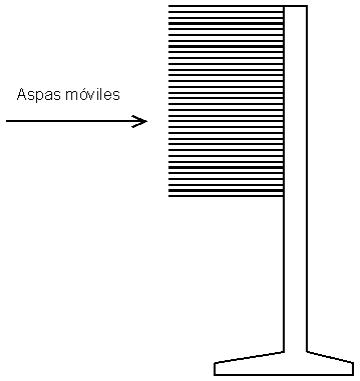


Figura 4.16 Dispositivo mecánico para medir la altura que salta una persona.

Para medir la altura que alcanza una persona con dicho aparato, primero debe medirse la altura que alcanza la persona, desplazando las aspas con la punta de los dedos de una mano, con las piernas y el brazo extendidos, y estirando el cuerpo de manera que las puntas de los pies estén a punto de despegarse del suelo pero sin hacerlo completamente. Posteriormente, la persona debe saltar y desplazar con los dedos de la mano, las aspas móviles del dispositivo, a fin de marcar la máxima altura alcanzada. Así, la altura saltada será la diferencia entre ambas medidas.

De esta forma, la medición de la altura alcanzada se realizó con el aparato descrito pero realizando el salto sobre la plataforma de medición, a fin de comparar los resultados obtenidos en ambos dispositivos. Se realizaron 9 mediciones con 2 personas de distinta altura y peso, y los resultados obtenidos fueron los que se muestran en las tablas 4.4 y 4.5.

Persona 1		
h_p	h_m	e
0.278	0.27	0.008
0.293	0.28	0.013
0.255	0.25	0.005
0.284	0.28	0.004
0.276	0.26	0.016
0.293	0.28	0.013
0.269	0.26	0.033
0.293	0.28	0.013
0.266	0.29	0.024
$\bar{e}_1$=0.0143, R=[0.004,0.033]		

Persona 2		
h_p	h_m	e
0.185	0.15	0.035
0.272	0.25	0.022
0.247	0.19	0.057
0.306	0.29	0.016
0.244	0.21	0.034
0.339	0.30	0.039
0.335	0.33	0.005
0.361	0.34	0.021
0.358	0.34	0.018
$\bar{e}_2$= 0.0274, R=[0.005,0.057]		

Tablas 4.4 y 4.5 Resultados del experimento 3, en metros.

En las tablas anteriores, h_p es la altura desplegada en la *Palm*, h_m la altura medida por el aparato mecánico, el error en las mediciones $e = h_p - h_m$, R el rango del conjunto de errores obtenidos, y $\bar{e}_i$ la media de los errores de las pruebas de la persona i, con i = 1, 2, y la cual se obtuvo mediante la expresión:

$$\bar{e}_i = \frac{1}{n}\sum_{r=1}^{n} e_r$$

en el que n=9 representa el total de resultados obtenidos para cada persona.

Como puede observarse, se presenta una serie de errores en las mediciones realizadas, cuyas fuentes se analizan a continuación. La principal fuente de error está relacionada con la técnica de salto vertical. Lo primero que hay que considerar es que se partió del hecho de que el movimiento con que una persona inicia el salto es el mismo con el que lo finaliza. Esto no es totalmente cierto, ya que el inicio del salto se lleva a cabo con las piernas totalmente extendidas, mientras que la caída se lleva a cabo con una pequeña flexión de las rodillas. Esto se refleja en el tiempo registrado, ya que el tiempo en el que la persona tarda en alcanzar su altura máxima, es menor al tiempo que tarda en descender. Dado que el tiempo de ascenso se considera la mitad del tiempo total, el tiempo de ascenso empleado para realizar los cálculos será un poco más grande que el tiempo de ascenso real, lo que se verá reflejado en la medición de una altura ligeramente mayor a la que realmente se alcanzó, situación que se presentó en todas las mediciones.

Además, puede observarse que las medias de los errores son distintas para cada persona. Cada individuo realiza movimientos distintos para realizar un salto, los cuales pueden contribuir a que el tiempo total de salto sea menor o mayor. Existen métodos para mejorar las técnicas de salto vertical, las cuales ayudarían considerablemente a que una persona obtuviera resultados más precisos con la utilización de este dispositivo.

No obstante, también hay que considerar que el dispositivo mecánico que se utiliza para obtener los resultados reales de las pruebas también tiene fuentes de error. En primer lugar, considerando el ancho de las aspas y la separación entre ellas de 0.005 m, se genera una incertidumbre en las mediciones del mismo valor.

Por otro lado, la determinación inicial de la altura que tiene la persona con las piernas y el brazo extendido simulando el momento de despegue del salto, y la determinación de la altura máxima que alcanza en el salto, moviendo con la mano el aspa más alta que alcanza, también puede considerarse como una fuente de error, ya que dependen de apreciaciones por parte de la persona: la persona "siente" que realiza la marca inicial cuando le "parece" que las puntas de sus pies casi se separan de la plataforma sin hacerlo, la persona "siente" que se estira para marcar la distancia inicial lo mismo que se estira cuando realiza el salto, y por último la persona "siente" que desplaza el aspa al realizar el salto cuando se encuentra a la máxima altura del mismo. Todos estos "sentimientos", pueden generar errores en los resultados, los cuales obviamente pueden tratar de reducirse con la supervisión de una persona ajena que verifique la prueba.

De esta forma, hemos encontrado distintas fuentes que pueden dar origen a los errores aleatorios encontrados. Aún sabiendo que dichos errores pueden disminuirse tratando los aspectos ya mencionados, se realizará un ajuste en la programación del dispositivo para que los resultados del mismo sean similares a los resultados obtenidos con el dispositivo mecánico. Así, dado que la media total de los errores obtenidos es $\bar{e} = .021$ m, se le resta dicho valor al resultado de la altura calculado por la *Palm*, antes de ser desplegado. Luego de realizar la operación anterior y observar las magnitudes de los errores establecidos, se puede esperar que los resultados de la prueba de salto vertical se encuentren dentro de un intervalo de ± 0.02 m con respecto al valor real.

4.6 COSTOS

A continuación se presenta un presupuesto del costo aproximado del dispositivo desarrollado en este trabajo. Hay que tomar en cuenta que en algunos elementos del dispositivo existe un rango muy amplio de modelos con diferentes características que pueden ser igualmente empleados en el dispositivo, y cuya elección puede encarecer o abaratar el costo total. Por dicha razón, en el presupuesto calculado, se considera el precio medio de estos dispositivos. El costo de la plataforma de medición y de los dispositivos electrónicos incluye el costo de las dos plataformas requeridas. El presupuesto no incluye el costo de la *PC*, ya que el uso de ésta no es indispensable.

Plataforma de medición	$ 2000.00
Dispositivos electrónicos	$ 1500.00
Software	$ 1500.00
Palm	$ 2000.00
Total	$ 7000.00

CAPÍTULO 5

RESULTADOS Y CONCLUSIONES

5.1 RESULTADOS

Con el prototipo construido, se logró evaluar el funcionamiento general del sistema, tomando en cuenta aspectos relacionados con los materiales empleados, la respuesta del dispositivo bajo condiciones normales de operación y la confiabilidad de los resultados obtenidos.

En primer lugar, se encontró un inconveniente importante relacionado con la construcción de la plataforma, ya que se comprobó que el material empleado en la base que contiene al láser, no es el más adecuado. Para calibrar el dispositivo, se requiere girar dicha base para ajustar el ángulo horizontal de incidencia del haz, y una vez que se encontró dicha posición, debe de apretarse fuertemente para que no se mueva. Como dicha base se construyó de madera, en el momento de apretarla una vez encontrada la posición correcta, el material sufría ciertas deformaciones que modificaban la posición ya determinada. Por eso, se consideró en el diseño final del dispositivo utilizar aluminio para la construcción de dicha base, a fin de evitar dichas deformaciones.

Con lo que respecta al resto de los elementos involucrados en la construcción de la plataforma de medición, se encontró una respuesta favorable. El cuadro de acero empleado para dar forma y rigidez a la plataforma, no sufre deformaciones que provoquen un cambio importante en la trayectoria del haz. Por otro lado, el sistema de calibración diseñado permite ajustar con precisión la trayectoria del láser, y no presenta desajustes causados por los impactos bajo los cuales se somete a la plataforma.

En lo que se refiere al sistema de sensores diseñado, hay varios aspectos sobre los cuales se debe puntualizar. Aunque el láser empleado permitió determinar la presencia y la ausencia de una persona sobre la plataforma, hay varios aspectos del mismo que no ofrecían un comportamiento eficiente. Por un lado, la dispersión que presenta es demasiado grande, lo que provoca un aumento gradual considerable del diámetro del haz a lo largo de su

trayectoria. Esta es una condición indeseable de operación, ya que se produce una considerable pérdida de potencia por unidad de área cuando el haz incide en el fototransistor. Dicha dispersión no fue tan grande como para que impidiera generar un cambio de estado en el fototransistor, pero para asegurar un mejor funcionamiento, se eligió para el diseño final del dispositivo un láser con un grado de dispersión pequeño.

Otro aspecto del láser cuya característica no arrojó resultados satisfactorios, fue su respuesta a la temperatura. Pudo comprobarse que la pérdida de potencia del mismo era tan grande cuando se ponía a funcionar bajo la luz del sol, que el fototransistor prácticamente no detectaba la presencia del haz. En relación con este hecho, se comprobó que emplear una estructura de protección para cubrir los ángulos de aluminio y el emisor láser, también evita la incidencia directa del sol sobre los mismos, reduciendo la temperatura y mejorando considerablemente su funcionamiento. Además, si se pinta dicha estructura de blanco, entonces casi la totalidad de la luz que incida sobre ella será reflejada. Aún así, en el diseño final del dispositivo, se sugiere la utilización de un láser con un rango amplio de temperatura, a fin de evitar fallos de operación por dichas condiciones.

Otro parámetro que afecta considerablemente el funcionamiento del dispositivo es la luminosidad ambiental. En muchas ocasiones, se comprobó que la luz del sol era capaz de provocar un cambio de estado en el fototransistor, aún cuando no hubiera presencia del haz láser. Por ello, se utilizaron distintos recursos para evitar lo más posible la incidencia de luz ambiental sobre el fototransistor. Por un lado, el empleo del tubo negro en cuyo interior se coloca el fototransistor, permite bloquear considerablemente la luz solar, permitiendo únicamente la penetración del haz láser que incide frontalmente sobre el orificio del tubo. Por otro lado, el pintar la plataforma y los ángulos de aluminio de negro, a fin de absorber la luz solar, y el uso de las estructuras de protección, evitan la incidencia de la luz solar sobre los espejos, anulando así una posible luz reflejada en los mismos en dirección al fototransistor. Con esto, se logra una diferencia lo suficientemente grande en el voltaje de salida del fototransistor cuando incide el haz del láser y cuando sólo incide la luz ambiental. Además, con el aumento de la potencia y la reducción de la dispersión del haz láser, dicha diferencia de voltaje será más amplia, y el funcionamiento del sistema mucho mejor.

Sin embargo, tomando en cuenta las especificaciones eléctricas del láser sugerido, en especial la referente a la potencia y a la dispersión, se hace necesario establecer medidas de seguridad en el uso del dispositivo, con el fin de evitar la incidencia directa del haz en los ojos de una persona. Para ello, tendrá que evitarse el uso de materiales que reflejen la luz al realizar las pruebas, a fin de impedir que haz alcance al ojo humano.

Tomando en cuenta todas las consideraciones anteriores, puede afirmarse que el sistema de sensores elegido fue adecuado, ya que permite obtener la información requerida del mundo exterior, sin ambigüedades y con una velocidad de respuesta alta, como lo requiere la aplicación.

El hecho de colocar la tableta con los elementos electrónicos en la parte inferior de la plataforma, sujeta a las paredes inferiores de la solera y cubierta por una caja, contribuye a evitar fallos en el funcionamiento relacionados con el calor, impactos y las condiciones de humedad y polvo de la superficie sobre la cual se realicen las pruebas.

En lo relacionado con las unidades de procesamiento y de despliegue, puede considerarse que los elementos elegidos para llevar a cabo estas etapas fueron los correctos. Con los recursos del PIC 16F84A pudieron solucionarse todas las exigencias del sistema. También puede afirmarse que la utilización de la *Palm* como dispositivo portátil de despliegue resultó ideal para la aplicación. El hecho de que *Pendragon Forms* fuera un software diseñado especialmente para la adquisición y despliegue de datos, hizo que pudiera enviarse y recibirse información correctamente, además de presentar los resultados obtenidos de una forma clara y útil. Asimismo, este software permitió almacenar la información en una *PC*, de forma que los datos pudieran ordenarse y manipularse al gusto del usuario.

En lo referente a los resultados obtenidos, como se mencionó en la sección de "Calibración" del capítulo anterior, la exactitud con que se realiza el cálculo del tiempo en cualquiera de las tres pruebas, es lo suficientemente grande para arrojar resultados confiables.

En el caso de la prueba de frecuencia de paso, se comprobó que el número de pasos que se dan sobre la plataforma corresponde efectivamente al número de pasos contados por el PIC y que la velocidad con que dichos pasos se realicen, no afecta la cuenta de los mismos.

En el caso de la prueba de carrera corta, se comprobó que el tiempo que la persona emplea en tocar la plataforma y desprenderse posteriormente de ella para iniciar el recorrido de regreso, es suficiente para que el PIC envíe a la *Palm* el resultado de la carrera anterior. Este hecho es de gran importancia, ya que el PIC debe de estar preparado para comenzar a contar el tiempo a partir del instante en que la persona despegue su pie de la plataforma, por lo que, para ese entonces, ya debió de haber enviado el resultado de la prueba anterior.

En el caso de la prueba de salto vertical, se encontró un rango de error en los resultados obtenidos. Como se analizó anteriormente, dicho error está relacionado con condiciones aleatorias más que al funcionamiento del dispositivo. Para mejorar los resultados obtenidos por el dispositivo, puede trabajarse en las técnicas del salto vertical, a fin de que coincida la teoría en la cual se basó el diseño del dispositivo con la forma real en que la persona está realizando este ejercicio.

Sin embargo, a pesar de dicho error, puede considerarse que los resultados son bastante buenos para obtener la información que necesita el entrenador. Hay que recordar que el dispositivo no fue diseñado para un centro especializado de pruebas físicas, sino para un equipo de fútbol, en el que la medida de la altura que alcanza una persona en un salto brinda información adicional sobre las características físicas de la persona, pero que no es tan determinante para la realización de este deporte, como para pensar que el error obtenido pueda llegar a ser decisivo.

Finalmente, pudo comprobarse que el dispositivo diseñado cumple con los objetivos planteados al inicio del presente trabajo. El hecho de que con un mismo dispositivo puedan llevarse a cabo los tres tipos de pruebas planteados, que dicho dispositivo pueda llevarse a cualquier lugar para realizar las pruebas sin necesidad de instalaciones o ajustes, y que además sea resistente y duradero al mismo tiempo que exacto, sin duda cumple con las

características de ser un dispositivo versátil, robusto y fácilmente transportable, como se esperaba.

Al ser un dispositivo que almacena automáticamente los resultados obtenidos en una *Palm* de forma clara y ordenada, y que permite transferir dichos resultados a una *PC* para su posterior análisis, sin duda contribuye a mejorar la rapidez y comodidad de realización de las pruebas con respecto a las formas tradicionales, además de que los resultados entregados son confiables.

5.2 CONCLUSIONES

Sin duda alguna, la incursión de la tecnología en el deporte es un fenómeno que se ha intensificado con el transcurso de los años. El ser humano siempre ha intentado dar un valor numérico a las actividades que realiza, de tal forma que pueda efectuar una valoración de manera más objetiva y justa de las mismas.

De esta manera, las pruebas de capacidad física no podían escapar a ese conjunto de actividades susceptibles de ser medidas. El deseo de evaluar la evolución física del cuerpo humano cuando se realiza un ejercicio, la capacidad para determinar numéricamente la respuesta de una persona a distintas pruebas, y la necesidad de establecer un patrón de comparación objetivo entre varios individuos que realizan un mismo tipo de prueba, es sin duda un reto interesante para la ingeniería.

Conforme más avances tecnológicos puedan incluirse en las pruebas de capacidad física, más fácil será le evaluación que un entrenador puede realizar de sus jugadores, y más seguro se sentirá al establecer juicios de valor a partir de los resultados que obtenga. Además, la aportación de la ingeniería no sólo se restringe a la capacidad de un dispositivo para brindar un resultado exacto, sino que además debe lograr que el ejercicio se desarrolle de tal forma que sea cada vez más cómodo para los entrenadores realizar las pruebas, y que sea cada vez mayor la rapidez con que puedan sacar sus conclusiones.

Como se observa en los resultados presentados, sin duda el dispositivo desarrollado brinda una buena solución a la necesidad de automatización de las pruebas de salto vertical, carrera corta y frecuencia de paso realizadas en el Club Universidad Nacional, A.C. Pero quizá la consecuencia más importante de la realización de este proyecto, no sea específicamente la solución tecnológica, sino el hecho de que se lograra una vinculación entre dos instituciones, una deportiva y otra académica, que aparentemente desarrollan actividades completamente distintas, pero que se unen por medio de una mutua necesidad. Porque si bien el deporte recurre a la ingeniería para resolver un problema, una institución académica que se dedica a la formación de ingenieros, necesita situaciones reales en las que los alumnos puedan aplicar sus conocimientos y desarrollar habilidades esenciales para su formación.

Sin vacilación, puede afirmarse que una creciente vinculación entre las instituciones académicas y los distintos sectores de la sociedad, trae, por un lado, que la sociedad se enriquezca con propuestas e ideas diversas y novedosas, y por otro, que los estudiantes en

plena formación encuentren una situación propicia que les permita desarrollar sus habilidades y aplicar sus conocimientos, y así conseguir una formación integral. Y finalmente, señalar que dichas relaciones entre los diversos sectores de la sociedad seguramente contribuirán a formar una nación más fuerte y más unida, que sea capaz de encontrar internamente las soluciones a sus problemas, apoyándose en las instituciones que la conforman.

5.3 RECOMENDACIONES

A pesar de que con el sistema desarrollado se cumplieron con los objetivos planteados al inicio del proyecto, podría hacerse una modificación al mismo con el fin de aumentar la comodidad en el momento en que un entrenador está manipulando la *Palm* cuando los jugadores están realizando la prueba. Dicha modificación consiste en dotar al dispositivo de una comunicación inalámbrica entre el PIC y la *Palm*, con el fin de que exista una independencia física entre el usuario de ésta última y la plataforma de medición, y evitar que el entrenador tenga que estar pendiente del cableado entre ambos. En el presente capítulo, se describe una interfaz inalámbrica instalada entre la Palm y el PIC, mencionando sus ventajas e inconvenientes, además de los aspectos a mejorar en el caso de que se quisiera incluir definitivamente en el diseño final del dispositivo.

Para lograr dicha comunicación inalámbrica, se utilizaron unos módulos de radio frecuencia Parallax (figura 5.1, http://www.parallax.com) capaces de recibir y transmitir información (transceiver) bajo un formato de comunicación serial asíncrona, a una frecuencia de 433 MHz y con un voltaje de alimentación de 5 V. Los datos se envían y se reciben a una velocidad de 9600 bps, sin bit de paridad, 8 bits de datos y un bit de paro, y con niveles de voltaje de 5 y 0 V. Es importante resaltar que para que una cadena sea enviada a través del *Transceiver*, la señal debe invertirse con respecto a la señal original, de tal forma que el estado de espera, el bit de paro, y los unos lógicos contenidos en la cadena se convierten a ceros lógicos, mientras que el bit de inicio y los ceros lógicos contenidos en la cadena se convierten en unos lógicos. También es importante señalar que para transmitir un conjunto de bytes, estos deben estar espaciados por lo menos 15 ms.

Figura 5.1 Transceiver.

En la figura 5.2 se muestra la conexión entre el *Transceiver* y el PIC. Cabe señalar que el módulo únicamente necesita un voltaje de alimentación (+VDC), un nivel de voltaje alto en la terminal 3 (MODE) para seleccionar el modo de comunicación serial, y la conexión de la línea de transmisión y de recepción de las terminales 4 (TXD) y 6 (RXD) respectivamente con el PIC. Para respetar el tiempo de 15 ms que debe existir entre cada byte enviado, puede realizarse de dos maneras: la primera es utilizando la terminal 5 (TXFLO) cuya salida se encuentra en estado bajo siempre que el *Transceiver* está preparado para transmitir información; si se conecta dicha terminal a una terminal libre de uno de los puertos del PIC, como lo puede ser la RA1, se puede llevar un control mediante software, en el que sólo se realice la transmisión de datos cuando exista un voltaje de entrada bajo en ese bit del puerto, y lo que aseguraría una máxima velocidad de transmisión. Otra forma de mantener ese tiempo entre bytes, es creando un retraso manual en el programa del PIC de 15 ms entre cada byte que se desee enviar. Como ya se mencionó con anterioridad, para poder transmitir las señales a través del *transceiver*, éstas deben estar invertidas. Por dicha razón, y como se observa en la figura 5.2, los transistores de conmutación Q_1 y Q_2, que antes tenían la configuración de colector común, ahora se configuran como emisor común, para que la señal de salida esté invertida con respecto a la señal de entrada.

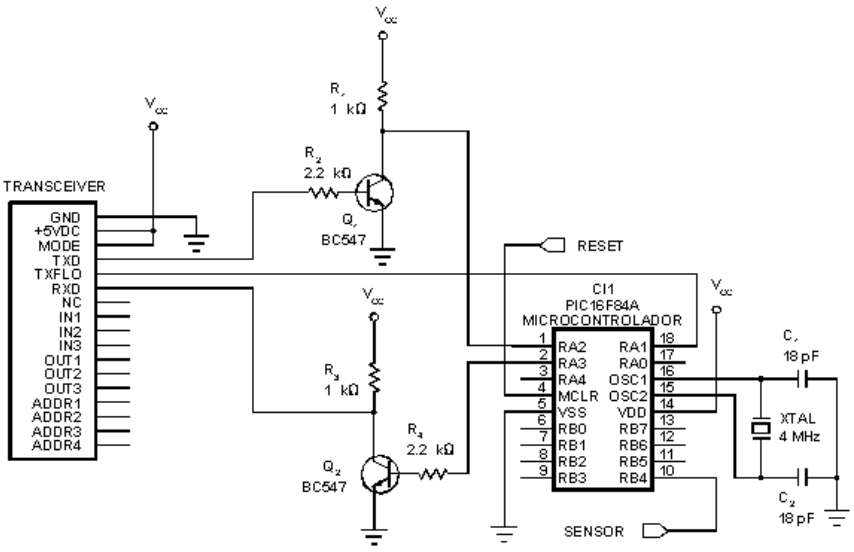


Figura 5.2 Conexión entre el PIC y el transceiver.

Para la conexión entre el radio y Palm, se realiza de una manera similar. El módulo debe ser alimentado con 5 V, además de conectar la terminal MODE a 5 V, y las líneas de transmisión y recepción a las líneas correspondientes de la Palm. Como el *Transceiver* maneja niveles TTL de voltaje, las líneas de transmisión y recepción del mismo, deben de conectarse del lado de la interfaz MAX232 que maneja dichos niveles de voltaje. Además, es necesario incluir un circuito integrado inversor 74LS04 entre el radio y la interfaz, a fin de invertir las señales mientras éstas tienen voltajes TTL. En este caso, no puede utilizarse la terminar TXFLO para controlar el envío de datos de la *Palm* hacia el PIC, ya que dicha línea no puede introducirse a la *Palm*, por lo que es necesario generar dicho retraso en el programa de la *Palm*. La figura 5.3 muestra la modificación a la interfaz con la inclusión del *Transceiver*.

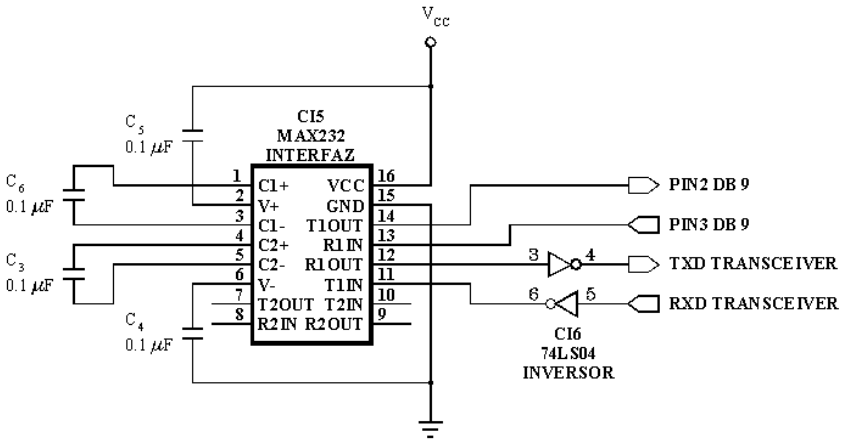


Figura 5.3 Interfaz MAX232 con el transceiver.

Aunque la comunicación inalámbrica descrita se probó en el dispositivo con resultados favorables en general, existieron ciertos aspectos que restaron eficiencia al sistema. La primera de ellas está relacionada con la velocidad de transmisión. En primer lugar, a pesar de que los módulos reciben señales con una velocidad de 9600 bps, la velocidad real con la que se transmiten los datos entre ellos es de tan solo 1200 bps, lo que hace más lento el intercambio de datos. Si a esto se suma el retraso obligatorio de 15 ms que debe existir entre cada byte enviado, entonces la lentitud que adquiere el sistema se vuelve significativa. Según las pruebas realizadas, se comprobó que para el caso de las pruebas de salto vertical y frecuencia de paso, no existió un cambio significativo en la dinámica de las pruebas, ya que la tardanza con que llegan los resultados a la *Palm*, no afecta a los mismos directamente. Sin embargo, en el caso de la prueba de carrera corta, dado que el tiempo transcurrido entre que un jugador va de una plataforma a la otra debe ser enviado durante el tiempo que está haciendo contacto con una de ellas, es decir, entre el instante en que llega a una plataforma y el instante en que la abandona para iniciar un nuevo recorrido, se comprobó que no era suficiente para enviar los datos a la *Palm*, y en muchas ocasiones se registraron lecturas erróneas por ese motivo.

Para solucionar dicho problema, se consideraron dos soluciones: por un lado, puede llevarse a cabo la búsqueda de otro tipo de radios que transmitan información a una velocidad más elevada, o bien puede llevarse a cabo una modificación en el manejo de los resultados para la prueba de carrera corta, haciendo que los resultados de cada recorrido se almacenen en la memoria de la PIC, y se envíen todos juntos al finalizar la prueba.

Otro problema que surgió con dichos módulos fue una reducción considerable del alcance nominal de comunicación. Según las especificaciones, los módulos están diseñados para transmitir a una distancia de hasta 75 m en espacio abierto. Sin embargo, se realizaron pruebas con los módulos instalados en el dispositivo, y la distancia máxima de comunicación alcanzada en espacio libre fue de 15 m. La reducción de dicha distancia de alcance, se debió a que no se respetaron algunas condiciones de operación que aseguran el correcto funcionamiento de los módulos, como lo fue un excesivo acercamiento entre el módulo con metales, baterías, y microcontroladores.

Se piensa que incluir una transmisión inalámbrica entre el PIC y la *Palm* es una opción que aún debe considerarse seriamente para la realización física del dispositivo, pero sin duda alguna debe ahondarse aún más en temas relacionados con transmisión inalámbrica de información y módulos de radio frecuencia.

APÉNDICE

A continuación, se tratarán aspectos técnicos relacionados con algunos de los dispositivos empleados en el proyecto. En primer lugar, se revisarán conceptos relacionados con los diodo-láseres. En segundo término, se presentarán los listados de los programas desarrollados en el PIC 16F84A y en la *Palm*, junto con la explicación de algunas particularidades relacionadas con ellos.

A.1 LÁSER

En el segundo capítulo del presente trabajo se mencionaron algunas características relacionadas con los láseres como intensidad, dirección, coherencia y color. El láser utilizado en este proyecto, es un diodo-láser o láser semiconductor, lo que quiere decir que, al igual que los leds, está formado por la unión de dos materiales semiconductores tipo n y tipo p, con la peculiaridad de que la luz que emiten es coherente. Al igual que los leds, es muy importante regular la corriente que circula a través del diodo para evitar posibles daños en el mismo, lo cual puede lograrse colocando una resistencia en serie del valor apropiado según la corriente máxima que pueda circular a través del diodo.

Para la elección de un láser, es necesario tomar en cuenta distintos aspectos. Aunque es importante considerar la potencia nominal, es necesario conocer la divergencia y las dimensiones del haz emitido. La divergencia es una medida del esparcimiento del haz cuando deja el módulo emisor. Un láser con una divergencia alta, disminuye la intensidad luminosa por unidad de área, a pesar de tener una alta potencia de emisión.

El color de un láser está determinado por los materiales semiconductores que lo conforman, los cuales definen la longitud de onda del haz emitido. Los láseres pueden emitir longitudes de onda que se encuentren en el rango de la luz visible o del infrarrojo.

La temperatura es un parámetro importante a tomarse en cuenta, ya que afecta directamente el funcionamiento del láser. Por un lado, un incremento en la temperatura provoca una variación de la longitud de onda del haz del orden de 0.2 nm/°C a 0.5 nm/°C, dependiendo

del tipo de diodo. Por otro lado, si se hace funcionar el láser fuera del rango de temperatura de operación, se detecta una reducción importante en la potencia.

El láser empleado en el prototipo construido es un láser semiconductor de propósito general con una potencia menor a 1 mW, con una longitud de onda de 630 nm a 680 nm, Clase II, que se emplea como apuntador láser. Con los resultados obtenidos, se llegó a la conclusión de que el funcionamiento del sistema puede mejorarse empleando un láser con menor divergencia, mayor potencia, y mayor rango de temperatura de operación.

Otro aspecto que conviene tomar en cuenta, es el de la seguridad en la utilización de este tipo de dispositivos. Existe una clasificación del láser de acuerdo con los estándares de seguridad, con el fin de asegurar el mínimo riesgo de exposición accidental al mismo, cuando éste está siendo utilizado en dispositivos de ingeniería. Según el Estándar Federal de Operación para Productos láser de Estados Unidos de América, los láseres pueden clasificarse en cuatro tipos: Clase I, Clase II, Clase IIIB, y Clase IV. El número de clase mayor, corresponde a los láseres de más alta potencia. Dicha clasificación, está realizada con base en el concepto de Límite de Emisión Accesible (AEL). El AEL es el máximo valor de radiación láser al cual puede ser expuesto un individuo durante su operación. A su vez, el AEL está basado en el nivel de Exposición Máxima Permisible (MPE). Un MPE es un nivel de la exposición a la luz láser a la cual se cree que un individuo puede ser expuesto sin sufrir daño alguno, el cual es un parámetro aceptado internacionalmente. Así, un láser se asigna a una clase, cuando su nivel de emisión excede al AEL de las clases anteriores, pero no excede al AEL de la clase asignada.

El láser empleado en el prototipo realizado pertenece a la Clase II, lo que significa que su potencia máxima es menor a 1 mW, y que el ojo está suficientemente protegido contra una exposición accidental o deliberada, por el pestañeo producto del reflejo natural al incidir el haz sobre el ojo.

El láser sugerido para realizar este proyecto pertenece a la Clase IIIB. Los láseres pertenecientes a esta clase, tienen una potencia menor a 5 mW, lo que implica que tienen la suficiente potencia para causar daño en los ojos si el rayo emitido incide directamente en ellos, y el cual está en función del tiempo y la potencia por unidad de área a la cual fue expuesto el ojo. Por dicha razón, siempre que un láser de este tipo se emplee en algún proyecto o aplicación, deben de mencionarse las características del láser empleado para evitar condiciones de uso inapropiado y evitar daños físicos en las personas circundantes.

A.2 PROGRAMACIÓN DEL PIC 16F84A

A continuación se muestra el programa desarrollado en el PIC 16F84A para calcular los resultados de las pruebas. El programa desarrollado ocupa un total de 278 palabras de memoria, un 30% del total de la memoria del PIC.

Para simular y ensamblar el programa realizado, se utilizó el ambiente de desarrollo MPLab para Windows/16, v.4.12.00 de Microchip Technology, Inc. Para descargar el programa en el PIC, se utilizó el programador PICALL & AVR, modo P16PRO, v 0.10a.

Cabe señalar que las líneas de programación que se encuentran en negritas, son aquéllas deben añadirse al programa para que el sistema pueda utilizar los módulos de radio frecuencia descritos en la sección de “Recomendaciones” del presente proyecto, cuya única modificación consiste en añadir un retraso de 15 ms entre cada byte que se transmite del PIC a la *Palm*.

Después del listado presentado, se describen algunas partes del programa que requieren de una explicación más detallada para su comprensión.

A.2.1 Listado del programa desarrollado en el PIC 16F84A

```
;Declaración de registros, variables y banderas

TMR0        EQU     0X01        ;Temporizador
INTCON_REG  EQU     0X0B        ;Registro INTCON
OPTION_REG  EQU     0X81        ;Registro OPTION
TRISA       EQU     0X85        ;Registro de programación del puerto A
TRISB       EQU     0X86        ;Registro de programación del puerto B
STATUS      EQU     0X03        ;Registro STATUS
PORTA       EQU     0X05        ;Puerto A
PORTB       EQU     0X06        ;Puerto B
RP0         EQU     0X05        ;Bit de selección de registros de bancos
RP1         EQU     0X06        ;Bit de selección de registros de bancos
Z           EQU     0X02        ;Bandera de cero
C           EQU     0X01        ;Bandera de acarreo
TOIF        EQU     0X02        ;Bandera de sobreflujo de interrupción por
                                ;Temporizador
GIE         EQU     0X07        ;Bit de habilitación general de las
                                ;interrupciones
RB4         EQU     0X04        ;Bit 0 del puerto B, entrada de la señal
                                ;proveniente del sensor
RA2         EQU     0X02        ;Bit 2 del puerto A, entrada de la señal
                                ;proveniente de la Palm
RA3         EQU     0X03        ;Bit 3 del puerto A, salida de la señal hacia
                                ;la Palm
TEMP0       EQU     0X11        ;Byte menos significativo de la variable que se
                                ;incrementa cada 500 µs durante la carrera y el
                                ;salto
TEMP1       EQU     0X12        ;Byte más significativo de la variable que se
                                ;incrementa cada 500 µs durante la carrera y el
                                ;salto
TIEMP0      EQU     0X1B        ;Byte menos significativo de la variable que
                                ;contiene el tiempo recibido en hexadecimal
TIEMP1      EQU     0X1C        ;Byte más significativo de la variable que
                                ;contiene el tiempo recibido en hexadecimal
DATO        EQU     0X16        ;Variable que contiene el dato a ASCII que
                                ;se va enviar
PDATO       EQU     0X17        ;Variable que contiene el dato en
                                ;hexadecimal que se transformará en ASCII
TIEM        EQU     0X18        ;Variable que almacena el valor ASCII
                                ;recibido
PASO        EQU     0X14        ;Variable que almacena el número de pasos
                                ;durante la prueba de frecuencia de paso
DECR0       EQU     0X1E        ;Variable para guardar TIEMP0
DECR1       EQU     0X1F        ;Variable para guardar TIEMP1
CUENTA      EQU     0X15        ;Variable utilizada en transmisión serie
TEMP        EQU     0X19        ;Variable utilizada en recepción de datos
BAND        EQU     0X10        ;Variable utilizada en transmisión de datos
```

```
BAND2       EQU     0X1A          ;Variable utilizada en transmisión de datos
SELEC       EQU     0X1D          ;Variable utilizada en identificación de
                                  ;pruebas
RET         EQU     0X20          ;Variable utilizada en retrasos
RET1        EQU     0X21          ;Variable utilizada en retrasos
RET2        EQU     0x22          ;Variable utilizada en retrasos

;Inicialización de registros

          org     0X0000
          BSF     STATUS,RP0
          BCF     STATUS,RP1
          MOVLW   0X10
          GOTO    SALTA
          GOTO    INTER
SALTA     MOVWF   TRISB
          MOVLW   0X04
          MOVWF   TRISA
          MOVLW   0X80
          MOVWF   OPTION_REG
          BCF     STATUS,RP0
          CLRF    INTCON_REG
          CLRF    PORTA
          CLRF    PORTB
          CLRF    SELEC
          BSF     PORTA,RA3

;Identificación de la prueba que se va a realizar

          CLRF    TIEMP0
          CLRF    TIEMP1
          CALL    RECIVDAT
          BTFSS   TIEM,0X00     ;Si el valor recibido es un uno, entonces se
                                ;realizará la prueba de carrera corta o salto
                                ;vertical; si se recibe un cero, se realizará
                                ;la prueba de frecuencia de paso
          GOTO    REG17
          BSF     SELEC,0X00
          GOTO    REG

;Recepción del tiempo proveniente de la Palm durante el cual se realizará la
;prueba de frecuencia de paso

REG17     BCF     SELEC,0X00
          CLRF    TIEMP0
          CLRF    TIEMP1
          MOVLW   0X01
          MOVWF   BAND2
          MOVLW   0X02
          MOVWF   BAND
          CALL    RECIVDAT
          CALL    RECIVE
          CALL    RECIVDAT
          CALL    RECIVE
          DECF    BAND2
          MOVLW   0X02
          MOVWF   BAND
          CALL    RECIVDAT
          CALL    RECIVE
          CALL    RECIVDAT
          CALL    RECIVE
          MOVF    TIEMP0,0
          MOVWF   DECR0
```

```
          MOVF    TIEMP1,0
          MOVWF   DECR1

;Inicio de la prueba

REG       BTFSS   PORTB,RB4     ;Detección del momento en que la persona se
                                ;coloca sobre la plataforma
          GOTO    REG
          CALL    RETRASO6
          BTFSS   PORTB,RB4
          GOTO    REG
          CLRF    PASO
          CLRF    TEMP0
          CLRF    TEMP1
REG1      BTFSC   PORTB,RB4     ;Detección del momento en que la persona deja
                                ;de hacer contacto con la plataforma iniciando
                                ;el salto, la carrera o los pasos
          GOTO    REG1
          CALL    RETRASO6
          BTFSC   PORTB,RB4
          GOTO    REG1
          MOVLW   0X0E
          MOVWF   TMR0
          MOVLW   0XA0
          MOVWF   INTCON_REG
          BTFSC   SELEC,0X00    ;Si la prueba a realizar es la de carrera o la
                                ;de salto, llama a subrutina; si es la de
                                ;frecuencia de paso, continúa
          CALL    CARRE
          MOVF    DECR0,0       ;Inicia programa de frecuencia de paso
          MOVWF   TIEMP0
          MOVF    DECR1,0
          MOVWF   TIEMP1
          CLRF    PASO
          BCF     SELEC,0X01
          BSF     PORTA,RA3
REG22     BTFSS   PORTB,RB4
          GOTO    REG22
          CALL    RETRASO6
          BTFSS   PORTB,RB4
          GOTO    REG22
          CALL    CONTAR
          GOTO    REG22
REG16     BCF     INTCON_REG,GIE
          MOVF    PASO,0        ;Envío de número de pasos registrado
          MOVWF   PDATO
          CALL    ENVIA
          MOVLW   0X0D          ;Código ASCII de retorno de carro
          MOVWF   DATO
          CALL    ENVDATO
          GOTO    REG

;Subrutina que cuenta el número de veces que la persona pisa la plataforma para
la prueba de frecuencia de paso

CONTAR    INCF    PASO
REG14     BTFSC   PORTB,RB4
          GOTO    REG14
          CALL    RETRASO6
          BTFSC   PORTB,RB4
          GOTO    REG14
          RETURN
```

```
;Inicio de la subrutina para contar el tiempo entre contactos con la plataforma
para las pruebas de carrera corta y de salto vertical

CARRE     BTFSS   PORTB,RB4     ;Detección del momento en que la persona
                                ;regresa a la plataforma
          GOTO    CARRE
          CALL    RETRASO6
          BTFSS   PORTB,RB4
          GOTO    CARRE
          BCF     INTCON_REG,GIE
          MOVF    TEMP0,0       ;Byte menos significativo del tiempo calculado
          MOVWF   PDATO
          CALL    ENVIA
          MOVF    TEMP1,0       ;Byte más significativo del tiempo calculado
          MOVWF   PDATO
          CALL    ENVIA
          MOVLW   0X0D          ;Código ASCII de retorno de carro
          MOVWF   DATO
          CALL    ENVDATO
          GOTO    REG

;Rutina que transforma el valor en ASCII recibido de la Palm, a hexadecimal

RECIVE    MOVF    TIEM,0
          BCF     STATUS,Z
          XORLW   0X30
          BTFSS   STATUS,Z
          GOTO    LETRAR
          CALL    RECIVDAT      ;El valor en ASCII que se va a recibir
                                ;corresponde a un número en hexadecimal
REG5      MOVLW   0X0F
          ANDWF   TIEM,0
          DECFSZ  BAND
          GOTO    PRIM
          RLF     TIEM
          RLF     TIEM
          RLF     TIEM
          RLF     TIEM
          MOVLW   0XF0
          ANDWF   TIEM,0
PRIM      BTFSC   BAND2,0X00
          GOTO    REG12
          ADDWF   TIEMP1
          RETURN
REG12     ADDWF   TIEMP0
          RETURN
LETRAR    CALL    RECIVDAT      ;El valor en ASCII que se va a recibir
                                ;corresponde a una letra en hexadecimal
          INCF    TIEM
          INCF    TIEM
          MOVLW   0X08
          IORWF   TIEM
          GOTO    REG5

;Subrutina para la recepción serial asíncrona de datos provenientes de la Palm

RECIVDAT  BTFSC   PORTA,RA2
          GOTO    RECIVDAT
          CLRF    TIEM
          MOVLW   0X08
          MOVWF   CUENTA        ;Se va a recibir una palabra de 8 bits
          CALL    RETRASO3
OTRA      CALL    RETRASO1
```

```
          BTFSS   PORTA,RA2
          GOTO    CER
          MOVLW   0XFF
          MOVWF   TEMP
          GOTO    UNO
CER       CLRF    TEMP
UNO       MOVLW   0X01
          ADDWF   TEMP
          RRF     TIEM
          DECFSZ  CUENTA,1
          GOTO    OTRA
          CALL    RETRASO1
          RETURN

;Rutina que transforma un valor hexadecimal a código ASCII para ser
;posteriormente enviado a la Palm

ENVIA     BSF     PORTA,RA3
          MOVLW   0X02
          MOVWF   BAND
REG4      BCF     STATUS,C
          MOVF    PDATO,0
          ANDLW   0X0F
          SUBLW   0X09
          BTFSC   STATUS,C
          GOTO    NUM
          GOTO    LETRA
NUM       MOVLW   0X30          ;El nibble a enviar es un número
          MOVWF   DATO
          CALL    ENVDATO
          MOVF    PDATO,0
          ANDLW   0X0F
          IORLW   0X30
          MOVWF   DATO
          CALL    ENVDATO
REG3      RRF     PDATO
          RRF     PDATO
          RRF     PDATO
          RRF     PDATO
          DECFSZ  BAND
          GOTO    REG4
          RETURN
LETRA     MOVLW   0X31          ;El nibble a enviar es una letra
          MOVWF   DATO
          CALL    ENVDATO
          MOVF    PDATO,0
          ANDLW   0X0F
          MOVWF   DATO
          DECF    DATO
          DECF    DATO
          MOVFW   DATO
          ANDLW   0X07
          IORLW   0X30
          MOVWF   DATO
          CALL    ENVDATO
          GOTO    REG3

;Subrutina que transmite valores en ASCII hacia la Palm

ENVDATO   BSF     PORTA,RA3
          MOVLW   0X08
          MOVWF   CUENTA
          BCF     PORTA,RA3
```

```
          CALL    RETRASO
REG9      MOVF    DATO,0
          BCF     STATUS,Z
          ANDLW   0X01
          BTFSC   STATUS,Z
          GOTO    CERO
          BSF     PORTA,RA3     ;Se envía un uno
REG10     CALL    RETRASO1
          RRF     DATO
          DECFSZ  CUENTA
          GOTO    REG9
          BSF     PORTA,RA3
          CALL    RETRASO
          CALL    RETRASO
          CALL    RETRASO
          CALL    RETRASO4
          RETURN
CERO      BCF     PORTA,RA3     ;Se envía un uno
          GOTO    REG10

;Retrasos de tiempo

RETRASO   CLRF    INTCON_REG    ;Retraso de 105 µs (tiempo de bit)
          MOVLW   0XD1
          MOVWF   TMR0
REG8      BTFSS   INTCON_REG,TOIF
          GOTO    REG8
          RETURN

RETRASO1  CLRF    INTCON_REG    ;Retraso de 93 µs
          MOVLW   0XD7
          MOVWF   TMR0
REG7      BTFSS   INTCON_REG,TOIF
          GOTO    REG7
          RETURN

RETRASO3  CLRF    INTCON_REG    ;Retraso de 63 µs
          MOVLW   0XE6
          MOVWF   TMR0
REG13     BTFSS   INTCON_REG,TOIF
          GOTO    REG13
          RETURN

RETRASO4  MOVLW   0XA0
          MOVWF   RET
REG19     CLRF    INTCON_REG    ;Retraso de 15 ms
          MOVLW   0XD6
          MOVWF   TMR0
REG18     BTFSS   INTCON_REG,TOIF
          GOTO    REG18
          DECFSZ  RET
          GOTO    REG19
          RETURN

RETRASO6  MOVLW   0XFF          ;Retraso de 1 ms, para evitar rebotes en la
                                ;lectura del sensor
          MOVWF   RET1
REG23     NOP
          DECFSZ  RET1
          GOTO    REG23
          RETURN

;Interrupción por Temporizador cada 500 µs
```

```
INTER     MOVLW   0X09
          MOVWF   TMR0
          MOVLW   0XA0
          MOVWF   INTCON_REG
          BTFSS   SELEC,0X00
          GOTO    INTER2
          INCFSZ  TEMP0         ;Los registros se incrementan para las pruebas
                                ;de carrera y salto
          RETFIE
          INCFSZ  TEMP1
          RETFIE
INTER2    DECFSZ  TIEMP0        ;Los registros se decrementan para la prueba
                                ;de frecuencia de paso
          RETFIE
          BTFSC   SELEC,0X01
          GOTO    REG16
          DECFSZ  TIEMP1
          RETFIE
          BSF     SELEC,0X01
          MOVLW   0XFF
          MOVWF   TIEMP0
          RETFIE

FIN       NOP
          END
```

A.2.2 Cálculo de tiempo

Como se observa en el programa realizado, cuando la persona se coloca sobre la plataforma, se encuentra en estado de espera hasta que se detecta que la persona ha abandonado la plataforma para iniciar cualquiera de las tres pruebas físicas, instante en que se debe empezar a contar el tiempo. En dicho instante, se habilita la interrupción por temporizador (TOIF) y la habilitación global (GIE), lo que hará que al presentarse un sobreflujo en el Temporizador (TMR0), se genere una interrupción y el programa salte a la dirección 0004h de la memoria de programa. Cada vez que exista una interrupción por sobreflujo del Temporizador, para el caso de las pruebas de carrera y salto, se incrementarán las variables TEMP0 y TEMP1, que al final contendrán la cuenta del tiempo total registrado, y para la prueba de frecuencia de paso, se decrementarán las variables TIEMP0 y TIEMP1 que contienen el tiempo durante el cual se va a realizar la prueba.

Para saber cada cuanto tiempo se quiere que se genere una interrupción por Temporizador, es necesario tomar en cuenta el tamaño de una localidad de memoria, la resolución con que se desean los resultados, y el tiempo máximo esperado que se va a contabilizar en cada prueba. En este caso, se determinó que el Temporizador generará una interrupción cada 500 μs. Con esto se tiene que el error máximo que se podrá generar en la cuenta del tiempo durante la realización de las pruebas es de 0.0005 s, lo que resulta insignificante para pruebas de capacidad física. Para poder conseguir que el Temporizador se interrumpa cada 500 μs, debe considerarse lo siguiente. El Temporizador se incrementa normalmente cada ciclo de instrucción, el cual equivale a 4 periodos del oscilador. Como se utiliza un cristal de 4 MHz, cada ciclo de instrucción se lleva a cabo en 1 μs. El PIC 16F84A cuenta con un Preescalador de 8 bits, que se programa asignándolo al módulo Temporizador colocando ceros en el bit de asignación (PSA), y en los bits de selección de proporción del

Preescalador (PS0, PS1, PS2), para que el Temporizador se incremente cada 2 ciclos de instrucción. Considerando que el Temporizador tiene 8 bits, entonces éste generará una interrupción por sobreflujo cada:

$$256 \times 2\ \mu s = 512\ \mu s$$

Si cada vez que se vaya a iniciar una cuenta con el Temporizador, éste se carga con un valor inicial, que en este caso fue de 0Eh, entonces se obtiene la cuenta exacta de 500 μs. Ahora bien, si una variable de 8 bits se incrementa cada interrupción por temporizador, entonces el tiempo total que será capaz de almacenar una sola localidad de memoria será de:

$$256 \times 500\ \mu s = 0.128\ s$$

Pero si se utilizan dos localidades de memoria, de tal forma que la segunda se incremente cada vez que se llena la primera, entonces se obtiene una capacidad de almacenamiento de:

$$256 \times 256 \times 500\ \mu s = 32.768\ s$$

Para la prueba de salto vertical, resulta más que obvio que una persona no va a permanecer en el aire más de 30 s; para la prueba de carrera corta, la cual se determinó de no más de 20 m, no se espera que una persona que está realizando su máximo esfuerzo para lograr el menor tiempo posible, emplee más de 30 s para completar el recorrido; finalmente, en el caso de la prueba de frecuencia de paso, dado que es una prueba de capacidad, y no de resistencia, no se utilizan tiempos tan largos para realizar la prueba.

Así, se concluye que dos localidades de memoria para almacenar los tiempos son más que suficientes para los fines perseguidos, y en el supuesto caso de que se necesitara contabilizar un tiempo mayor a los 32 s, basta con añadir más localidades de memoria.

Finalmente, para llevar la cuenta del número de pasos que la persona realiza en la plataforma, es suficiente con una sola localidad de memoria, ya que no se espera que la persona realice más de 256 pasos en tiempos tan cortos.

A.2.3 Transmisión y recepción serial asíncrona

Una de las etapas del programa que se desarrolló fue la de comunicación serial asíncrona entre el PIC y la *Palm*, cuya teoría se explicó en el segundo capítulo.

La *Palm* trabaja con un formato de transmisión de 9600 bps, 8 bits de datos, sin bit de paridad y un bit de paro. Esto quiere decir que cada cadena transferida entre la *Palm* y el PIC constará de 10 bits: 1 bit de inicio bajo, 8 bits de datos, y 1 bit de paro alto. El tiempo de bit, que es el inverso de la velocidad de transmisión, resulta de 104 μs.

La transmisión de datos del PIC a la *Palm* se realiza mediante la terminal RA3. Para lograrlo, antes de iniciar la transmisión RA3 debe presentar un estado alto para generar el

estado de espera. Cuando la transmisión va a comenzar, se genera el bit de inicio cambiando el estado de RA3 a cero durante un tiempo de bit. Posteriormente, se realiza una AND entre el dato que se va a transmitir y el 01h, para comprobar si el bit menos significativo del dato es cero o uno. Dependiendo del resultado, se colocará un 0 ó un 1 en RA3 durante un tiempo de bit; después, se recorre el dato que se está enviando un bit a la derecha, y se realiza la misma secuencia 8 veces, de tal forma que se envían todos los bits del dato, comenzando por el menos significativo. Una vez que se han enviado todos los bits del dato, se pone en alto la salida de RA3, al menos durante dos tiempos de bit, antes de que se transmita el siguiente valor.

Para realizar la recepción de datos, se sigue la misma idea. La recepción se realiza mediante RA2. La señal en RA2 está permanentemente en alto mientras no se recibe algún valor (estado de espera), y por ello, cuando el programa va a recibir un dato, se mantiene en espera hasta que se detecta un cambio de nivel de alto a bajo, lo que indica que se ha recibido el bit de inicio bajo de una cadena. Cuando esto sucede, se realiza la lectura del estado de RA2 después de un tiempo y medio de bit, lo cual permite tomar la lectura del primer bit recibido justo a la mitad del tiempo de transmisión de dicho bit, lo más alejado posible de los bits contiguos para evitar errores. A partir de este instante, se realizará 7 veces más la lectura del estado de RA2 cada tiempo de bit, hasta que se haya completado la lectura de la cadena recibida. Cuando el bit leído es un 1, se genera un sobreflujo, lo que provoca que la bandera de "acarreo" (C) se ponga en 1; posteriormente, se recorre la variable "TIEM" un lugar hacia la derecha, lo que hará que se coloque un 1 en el bit más significativo de dicha variable a causa del acarreo provocado; cuando el bit leído es un cero, entonces no se genera el sobreflujo, y al recorrer la variable un lugar hacia la derecha se colocará un cero en el bit más significativo. De esta forma, si esto se realiza cada vez que se lee RA2, se formará la cadena recibida en la variable "TIEM" que se va recorriendo, quedando el primer bit recibido en el bit menos significativo de la localidad de memoria.

A.2.4 Conversión de caracteres en sistema hexadecimal a su equivalente en sistema decimal en código ASCII, y viceversa

Como ya se mencionó, los resultados de las pruebas quedan almacenados en localidades de memoria del PIC. Sin embargo, dichos resultados se encuentran en sistema hexadecimal, y los datos que recibe la *Palm* son interpretados bajo el código ASCII, lo que significa que un número almacenado en la memoria del PIC, no representa el mismo número cuando es interpretado por la *Palm*. El objetivo es lograr que los dígitos hexadecimales contenidos en la memoria del PIC, aparezcan en la *Palm* en su equivalente decimal, sabiendo que ésta última interpretará los datos recibidos bajo el Código ASCII.

Para lograrlo, es necesario considerar dos equivalencias del sistema hexadecimal, una con el código ASCII y otra con el sistema decimal, las cuales se muestran en las tablas A.1 y A.2 respectivamente.

Para llevar a cabo dicha tarea, debe considerarse lo siguiente. Cada localidad de memoria del PIC está conformada por 8 bits, en el que cada nibble (conjunto de 4 bits) representa un dígito hexadecimal. La idea es transformar cada uno de estos nibbles en su equivalente

decimal en código ASCII, de tal forma que la *Palm* reciba el valor decimal que representa cada nibble. Dado que por cada dígito en el sistema hexadecimal se obtienen 2 dígitos en el sistema decimal, se enviarán un total de 4 bytes de valores en código ASCII por cada localidad de memoria que se desee transmitir. Recordando que fue necesario almacenar el valor de tiempo calculado en dos localidades de memoria, es importante señalar que se comenzará a transmitir el nibble menos significativo de los 4 que conforman el número total, mientras que el primer dígito decimal que se enviará, será el de las decenas.

Caracteres ASCII (código de 7 bits)								
MSN / LSN	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	'	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	'	<	L	v	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	\	n	~
F	SI	US	/	/	O	_	o	DEL

Tabla A.1 Conversión entre código ASCII y hexadecimal.
MSN, nibble más significativo;
LSN, nibble menos significativo.

Sistema hexadecimal	Sistema decimal
0	00
1	01
2	02
3	03
4	04
5	05
6	06
7	07
8	08
9	09
A	10
B	11
C	12
D	13
E	14
F	15

Tabla A.2 Conversión entre el sistema binario y el sistema hexadecimal.

Para lograr lo anterior, se desarrolló el algoritmo que se describe a continuación. Como el nibble está en hexadecimal, su valor puede estar representado por un "número" o una

"letra", lo cual se determina restándole el número 9; si el resultado es mayor que cero, el nibble es una "letra", si no, es un "número".

Si el nibble es una "letra", quiere decir que su valor decimal equivalente está entre el 10_{10} y el 15_{10}. Así, siempre que se detecte una "letra", se transmitirá en primer lugar 31_{16}, que es el código ASCII del 1_{10}, las decenas del número decimal. Para determinar el siguiente byte a transmitir, el de las unidades, se decrementa dos veces el nibble en cuestión, se realiza una AND con el 07_{16} y luego una OR con el 30_{16}. El resultado que se obtiene es el código ASCII del dígito decimal buscado.

Si el nibble es un "número", entonces se envía primero el 30_{16} (que representa un 0_{10}, que es el dígito de las decenas) y posteriormente se efectúa una OR entre el 30_{16} y el nibble para obtener el código ASCII del dígito de las unidades. Dicho proceso se realiza con cada nibble de cada localidad de memoria que se desea enviar. El siguiente ejemplo ilustra el procedimiento anterior.

Se toma al $4F_{16}$ como el valor hexadecimal que va a ser enviado a la *Palm*. Según el algoritmo, se toma el nibble menos significativo, que en este caso es F_{16}. Como es una "letra", el dígito de las decenas de su equivalente decimal es el 1_{10} y su código ASCII el 31_{16}. Para determinar el dígito de las unidades, se decrementa F_{16} en dos, quedando D_{16}, luego se realiza una AND entre el D_{16} y 7_{16}, que da como resultado 5_{16}, cuyo código ASCII es el 35_{16}. Como puede comprobarse en la tabla A.2, F_{16} efectivamente equivale a 15_{10}. De esta forma, los bytes que serán enviados para que la *Palm* reconozca al valor F_{16}, será el 31_{16} y el 35_{16}.

Continuando con el ejemplo, se observa que el nibble más significativo del valor planteado es el 4_{16}. Como es un "número", el dígito de las decenas del equivalente decimal es el 0_{10}, mientras que el de las unidades es el mismo que el número hexadecimal, el 4_{10}. Así, los bytes enviados serán el 30_{16} y 34_{16}, ya que son los códigos ASCII del 0_{10} y del 4_{10}, respectivamente. De esta manera, el número total que aparecerá en la *Palm*, en representación del $4F_{16}$, son los dígitos: 1, 5, 0 y 4.

Es necesario advertir, que para obtener el tiempo real transcurrido en la realización de las pruebas, la transformación de cada dígito hexadecimal a decimal es sólo el primer paso, pero el resto se verá más adelante cuando se muestre el programa desarrollado en la *Palm*.

Para la recepción de datos provenientes de la *Palm*, se realiza el mismo proceso que para la transmisión, pero de forma inversa. Cada byte recibido representa uno de los dos dígitos que conforman el número decimal representado en código ASCII. Con dos dígitos recibidos, debe conformarse cada uno de los nibbles que forman cada byte hexadecimal, de tal forma que será necesario recibir 4 bytes en código ASCII para completar una localidad de memoria. También es importante señalar que el primer byte recibido representa las decenas del número decimal que está siendo enviado; por otro lado, el primer par de bytes recibido representa el nibble menos significativo de los 4 necesarios para conformar las dos localidades de memoria empleadas para almacenar la totalidad del valor del tiempo.

Lo primero que se hace con el primer byte recibido es verificar si es un 30_{16} o un 31_{16}. Si es un 30_{16}, quiere decir que el valor decimal recibido representará un "número" luego de transformarlo al nibble correspondiente en hexadecimal. Así, al siguiente byte recibido, sólo habrá que efectuar una AND con un $0F_{16}$ para obtener el valor de nibble que se almacenará en la memoria.

En cambio, si el primer byte que se recibe es un 31_{16}, quiere decir que el dato recibido estará representado por una "letra" al transformarlo al nibble correspondiente en hexadecimal. Para hallar la correspondiente "letra", se toma el segundo byte recibido, se realiza una AND con $0F_{16}$, se incrementa dos veces, y se efectúa una OR con el 08_{16}, dando como resultado el nibble hexadecimal buscado. Nuevamente un ejemplo ilustrará el procedimiento anterior.

Suponiendo que la *Palm* envía la siguiente cadena de números: 1, 2, 0 y 9, y que por lo tanto el PIC recibe: 31_{16}, 32_{16}, 30_{16} y 39_{16}, en ese mismo orden.

Como el primer byte recibido es el 31_{16}, quiere decir que el dato enviado representará una "letra" al transformarlo a hexadecimal. Por lo tanto, se toma el siguiente byte, el 32_{16}, se realiza una AND con $0F_{16}$ quedando 02_{16}, luego se incrementa dos veces quedando 04_{16}, y por último al efectuar una OR con el 08_{16}, da como resultado $0C_{16}$. Como puede verificarse en la Tabla A.2, $0C_{16}$ efectivamente es el equivalente hexadecimal del 12_{10}. Posteriormente se detecta el 30_{16}, lo que indica que el dato enviado representará un "número" al transformarlo a hexadecimal. Así, si se realiza una AND entre 39_{16} y $0F_{16}$, da como resultado 09_{16}, que representa al 9_{10}. De esta forma, el byte hexadecimal que se almacenará en la memoria es $9C_{16}$, considerando que la primera pareja de bytes en código ASCII representa el byte menos significativo del byte hexadecimal.

A.3 PROGRAMACIÓN DE LA *PALM*

En esta sección se incluye el listado del programa desarrollado en la *Palm* y se explica el procedimiento que se empleó para convertir los resultados provenientes del PIC de sistema hexadecimal a su correspondiente en sistema decimal. Otros detalles importantes en la programación de la *Palm*, se describieron en la sección 4.4.1 del capítulo 4.

A.3.1 Listado de los programas realizados en la *Palm*

- **Carrera corta**

```
Nombre: Jugador
Tipo: lookup list
Nombre de columna: Jugador
Default: Jorge Campos
Propiedades: ninguna
Script:
       initialize:
       result = "9600-N-8-1"
       temp = U
       transmit serial temp
       result = "9600-N-8-1"
       temp=1
       transmit serial temp

Nombre: Distancia[m]
Tipo: lookup list
Nombre de columna: Distancia
Default: 5
Propiedades: ninguna
Script:
       Distancia[m]
       calculate:
```

```
          temp = $16 * 1000
          temp = temp + 0 .5
          temp =  integer temp
          $16 = temp/1000
          temp= $16/$2
          temp=temp*1000
          temp=temp+.5
          temp=integer temp
          temp = temp/1000
          $16 = $16 & "s, "
          $16 = $16 & temp
          $16 = $16 & " m/s"

          temp= $17/$2
          temp=temp*1000
          temp=temp+.5
          temp=integer temp
          temp = temp/1000
          $17 = $17 & "s  "
          $17 = $17 & temp
          $17 = $17 & " m/s"
          temp= $18/$2
          temp=temp*1000
          temp=temp+.5
          temp=integer temp
          temp = temp/1000
          $18 = $18 & "s,  "
          $18 = $18 & temp
          $18 = $18 & " m/s"
          $22 = 0

Nombre:  Tiempo1[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo1
Default:
Propiedades: ninguna
Script:
       exit:
          show 5
          beep
       scan:
              answer = scandata

Nombre: Tiempo2[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo2
Default:
Propiedades:ninguna
Script:
       enter:
       if $3 = null then
         goto 3
       endif
       scan:
          answer = scandata
       exit:
           show 6
          beep

Nombre: Tiempo3[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo3
Default:
Propiedades:oculto
Script:
       enter:
       if $4 = null then
         goto 4
       endif
       scan:
          answer = scandata
       exit:
           show 7
           beep

Nombre: Tiempo4[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo4
Default:
Propiedades: oculto
Script:
       enter:
       if $5 = null then
         goto 5
       endif
       scan:
          answer = scandata
       exit:
           show 8
           beep

Nombre: Tiempo5[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo5
Default:
Propiedades: oculto
Script:
       enter:
       if $6 = null then
         goto 6
       endif
       scan:
          answer = scandata
       exit:
           show 9
           beep

Nombre: Tiempo6[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo6
Default:
Propiedades: oculto
Script:
       enter:
       if $7 = null then
         goto 7
       endif
       scan:
          answer = scandata
       exit:
           show 10
           beep

Nombre: Tiempo7[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo7
```

```
Default:
Propiedades: oculto
Script:
       enter:
       if $8 = null then
         goto 8
       endif
       scan:
          answer = scandata
       exit:
           show 11
           beep

Nombre: Tiempo8[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo8
Default:
Propiedades: oculto
Script:
       enter:
       if $9 = null then
         goto 9
       endif
       scan:
          answer = scandata
       exit:
           show 12
           beep

Nombre: Tiempo9[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo9
Default:
Propiedades: oculto
Script:
       enter:
       if $10 = null then
         goto 10
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Tiempo10[seg]  [BARCODE]
Tipo: Freeform text
Nombre de columna: Tiempo10
Default:
Propiedades: oculto
Script:
       enter:
       if $11 = null then
         goto 11
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Hora
Tipo: time
Nombre de columna: Hora
Default:
Propiedades:
Script:
       initialize:
       answer = now

Nombre: Fecha
Tipo: date only
Nombre de columna: fecha
Default:
Propiedades:
Script:
       initialize:
       answer = now

Nombre: Estadísticas
Tipo: button
Nombre de columna: Estadisticas
Default: mostrar
Propiedades:
Script:
       click:
       show 17
       show 18
       show 16
       if $21 = 1 then
          msgbox "Ya realizó los
       cálculos"
          return
       endif
       $21 = 1
       $22 = 1
       $23 = 1
       if $3 = null then
          $16 = 0
          $17 = 0
          $18 = 0
          hide from 5 to 12
          return
       endif
       $17 = $3
       $16 = $3
       $18 = $3
       temp = $3
       if $4 = null then
           hide from 5 to 12
           return
       endif
       if $4 < $3 then
             $17 = $4
       else
             $18 = $4
       endif
       temp = temp + $4
       if $5 = null then
          $16 = temp/2
          hide from 5 to 12
          return
       endif
       if $5 < $17 then
             $17 = $5
       endif
       if $5 > $18 then
             $18 = $5
```

```
endif
temp = temp + $5
if $6 = null then
   $16 = temp/3
    hide from 6 to 12
    return
endif
if $6 < $17 then
      $17 = $6
endif
if $6 > $18 then
      $18 = $6
endif
temp = temp + $6
if $7 = null then
   $16 = temp/4
    hide from 7 to 12
   return
endif
if $7 < $17 then
      $17 = $7
endif
if $7 > $18 then
      $18 = $7
endif
temp = temp + $7
if $8 = null then
   $16 = temp/5
    hide from 8 to 12
   return
endif
if $8 < $17 then
      $17 = $8
endif
if $8 > $18 then
      $18 = $8
endif
temp = temp + $8
if $9 = null then
   $16 = temp/6
    hide from 9 to 12
    return
endif
if $9 < $17 then
      $17 = $9
endif
if $9 > $18 then
      $18 = $9
endif
temp = temp + $9
if $10 = null then
    $16 = temp/7
     hide from 10 to 12
    return
endif
if $10 < $17 then
      $17 = $10
endif
if $10 > $18 then
      $18 = $10
endif
temp = temp + $10
if $11 = null then
   $16 = temp/8
    hide from 11 to 12
   return
endif
if $11 < $17 then
      $17 = $11
endif
if $11 > $18 then
      $18 = $11
endif
temp = temp + $11
if $12 = null then
   $16 = temp/9
    hide 12
    return
else
temp = temp + $12
   $16 = temp/10
endif
if $12 < $17 then
      $17 = $12
endif
if $12 > $18 then
      $18 = $12
endif
```

```
Nombre: Promedio
Tipo: freeform text
Nombre de columna: Promedio
Default:
Propiedades: oculto
Script:

Nombre: Mínimo
Tipo: freeform text
Nombre de columna: Minimo
Default:
Propiedades: oculto
Script:

Nombre: Máximo
Tipo: freeform text
Nombre de columna: maximo
Default:
Propiedades: oculto
Script:

Nombre: Limpiar valores
Tipo: button
Nombre de columna: Limpiar
Default: Limpiar
Propiedades:
Script:
      click:
      $3 = null
      $4 = null
      $5 = null
      $6 = null
      $7 = null
      $8 = null
      $9 = null
      $10 = null
      $11 = null
```

```
      $12 = null
      $16 = null
      $17 = null
      $18 = null
      $23 = 0
      $22 = 0
      $21 = 0
      hide from 5 to 12
      hide from 16 to 18
      result = "9600-N-8-1"
      temp= U
      transmit serial temp
      result = "9600-N-8-1"
      temp=1
      transmit serial temp

Nombre: Registro nuevo
Tipo: button
Nombre de columna: NvoRegistro
Default: Crear
Propiedades:
Script:
      click:
      result = "9600-N-8-1"
      temp=U
      transmit serial temp
      result = "9600-N-8-1"
      temp=1
      transmit serial temp
      $3 = null
      $4 = null
      $5 = null
      $6 = null
      $7 = null
      $8 = null
      $9 = null
      $10 = null
      $11 = null
      $12 = null
      $16 = null
      $17 = null
      $18 = null
      $23 = 0
      $22 = 0
      $21 = 0
      hide from 5 to 12
      hide from 16 to 18
      clone

Nombre: Variable1
Tipo: freeform text
Nombre de columna: Variable1
Default: 0
Propiedades: oculto
Script:
      calculate:
      if $23 = 1 then
         return
      endif
      temp=0
      if $3 = null then
         hide 5
         return
      endif
      mid $3 1 2
      temp = result
      mid $3 3 2
      result = 16 * result
      temp= temp + result
      mid $3 5 2
      result = result * 16
      result = result * 16
      temp = temp + result
      mid $3 7 2
      result = result * 16
      result = result * 16
      result = result * 16
      temp = temp + result
      result = temp * .0005
      result = result * 1.00255
      result = result - .0027169
      result = result * 1000
      result = result + .5
      result = integer result
      $3 = result /1000
      if $4 = null then
         hide  from 5 to 12
         return
      endif
      mid $4 1 2
      temp = result
      mid $4 3 2
      result = 16 * result
      temp= temp + result
      mid $4 5 2
      result = result * 16
      result = result * 16
      temp = temp + result
      mid $4 7 2
      result = result * 16
      result = result * 16
      result = result * 16
      temp = temp + result
      result = temp * .0005
      result = result * 1.00255
      result = result - .0027169
      result = result * 1000
      result = result + .5
      result = integer result
      $4 = result /1000
      if $5 = null then
          hide from 5  to 12
          return
      endif
      mid $5 1 2
      temp = result
      mid $5 3 2
      result = 16 * result
      temp= temp + result
      mid $5 5 2
      result = result * 16
      result = result * 16
      temp = temp + result
      mid $5 7 2
      result = result * 16
      result = result * 16
```

```
        result = result * 16
        temp = temp + result
        result = temp * .0005
        result = result * 1.00255
        result = result - .0027169
        result = result * 1000
        result = result + .5
        result = integer result
        $5 = result /1000
        if $6 = null then
            hide from 6 to 12
            return
        endif
        mid $6 1 2
        temp = result
        mid $6 3 2
        result = 16 * result
        temp= temp + result
        mid $6 5 2
        result = result * 16
        result = result * 16
        temp = temp + result
        mid $6 7 2
        result = result * 16
        result = result * 16
        result = result * 16
        temp = temp + result
        result = temp * .0005
        result = result * 1.00255
        result = result - .0027169
        result = result * 1000
        result = result + .5
        result = integer result
        $6 = result /1000
        if $7 = null then
            hide from 7 to 12
         return
        endif
        mid $7 1 2
        temp = result
        mid $7 3 2
        result = 16 * result
        temp= temp + result
        mid $7 5 2
        result = result * 16
        result = result * 16
        temp = temp + result
        mid $7 7 2
        result = result * 16
        result = result * 16
        result = result * 16
        temp = temp + result
        result = temp * .0005
        result = result * 1.00255
        result = result - .0027169
        result = result * 1000
        result = result + .5
        result = integer result
        $7 = result /1000

Nombre: Variable2
Tipo: freeform text
Nombre de columna: Variable2
Default: 0
Propiedades: oculto
Script:
        calculate:
        if $23 = 1 then
           return
        endif

        if $8 = null then
            hide from 8 to 12
            return
        endif
        mid $8 1 2
        temp = result
        mid $8 3 2
        result = 16 * result
        temp= temp + result
        mid $8 5 2
        result = result * 16
        result = result * 16
        temp = temp + result
        mid $8 7 2
        result = result * 16
        result = result * 16
        result = result * 16
        temp = temp + result
        result = temp * .0005
        result = result * 1.00255
        result = result - .0027169
        result = result * 1000
        result = result + .5
        result = integer result
        $8 = result /1000
        if $9 = null then
            hide from 9 to 12
            return
        endif
        mid $9 1 2
        temp = result
        mid $9 3 2
        result = 16 * result
        temp= temp + result
        mid $9 5 2
        result = result * 16
        result = result * 16
        temp = temp + result
        mid $9 7 2
        result = result * 16
        result = result * 16
        result = result * 16
        temp = temp + result
        result = temp * .0005
        result = result * 1.00255
        result = result - .0027169
        result = result * 1000
        result = result + .5
        result = integer result
        $9 = result /1000
        if $10 = null then
            hide from 10 to 12
            return
        endif
        mid $10 1 2
```

```
temp = result
mid $10 3 2
result = 16 * result
temp= temp + result
mid $10 5 2
result = result * 16
result = result * 16
temp = temp + result
mid $10 7 2
result = result * 16
result = result * 16
result = result * 16
temp = temp + result
result = temp * .0005
result = result * 1.00255
result = result - .0027169
result = result * 1000
result = result + .5
result = integer result
$10 = result /1000
if $11 = null then
    hide from 11 to 12
    return
endif
mid $11 1 2
temp = result
mid $11 3 2
result = 16 * result
temp= temp + result
mid $11 5 2
result = result * 16
result = result * 16
temp = temp + result
mid $11 7 2
result = result * 16
result = result * 16
result = result * 16
temp = temp + result
result = temp * .0005
result = result * 1.00255
result = result - .0027169
result = result * 1000
result = result + .5
result = integer result
$11 = result /1000
if $12 = null then
    hide 12
   return
endif
mid $12 1 2
temp = result
mid $12 3 2
result = 16 * result
temp= temp + result
mid $12 5 2
result = result * 16
result = result * 16
temp = temp + result
mid $12 7 2
result = result * 16
result = result * 16
result = result * 16
temp = temp + result
result = temp * .0005
result = result * 1.00255
result = result - .0027169
result = result * 1000
result = result + .5
result = integer result
$12 = result /1000
```

```
Nombre: Variable3
Tipo: freeform text
Nombre de columna: Variable3
Default: 0
Propiedades: oculto
Script:
```

- **Salto Vertical**

```
Nombre: Jugador
Tipo: lookup list
Nombre de columna: Jugador
Default: Jorge Campos
Propiedades: ninguna
Script:
       initialize:
       result = "9600-N-8-1"
       temp=U
       transmit serial temp
       result = "9600-N-8-1"
       temp=1
       transmit serial temp

Nombre: Altura1[cm]   [BARCODE]
Tipo: freeform text
Nombre de columna: Altura1[cm]
Default:
Propiedades: ninguna
Script:
       exit:
          show 4
          beep
       scan:
              answer = scandata

Nombre: Altura2[cm]   [BARCODE]
Tipo: freeform text
Nombre de columna: Altura2[cm]
Default:
Propiedades: ninguna
Script:
       enter:
       if $2 = null then
         goto 2
       endif
       scan:
          answer = scandata
       exit:
           show 5
          beep

Nombre: Altura3[cm]   [BARCODE]
Tipo: freeform text
Nombre de columna: Altura3[cm]
Default:
Propiedades: oculto
Script:
       enter:
       if $3 = null then
         goto 3
       endif
       scan:
          answer = scandata
       exit:
           show 6
          beep

Nombre: Altura4[cm]   [BARCODE]
Tipo: freeform text
Nombre de columna: Altura4[cm]
Default:
Propiedades: oculto
Script:
       enter:
       if $4 = null then
         goto 4
       endif
       scan:
          answer = scandata
       exit:
           show 7
          beep

Nombre: Altura5[cm]   [BARCODE]
Tipo: freeform text
Nombre de columna: Altura5[cm]
Default:
Propiedades: oculto
Script:
       enter:
       if $5 = null then
         goto 5
       endif
       scan:
          answer = scandata
       exit:
           show 8
           beep

Nombre: Altura6[cm]   [BARCODE]
Tipo: freeform text
Nombre de columna: Altura6[cm]
Default:
Propiedades: oculto
Script:
       enter:
       if $6 = null then
         goto 6
       endif
       scan:
          answer = scandata
       exit:
           show 9
           beep

Nombre: Altura7[cm]   [BARCODE]
Tipo: freeform text
Nombre de columna: Altura7[cm]
Default:
Propiedades: oculto
Script:
       enter:
       if $7 = null then
         goto 7
       endif
       scan:
          answer = scandata
       exit:
           show 10
           beep

Nombre: Altura8[cm]   [BARCODE]
Tipo: freeform text
```

```
Nombre de columna: Altura8[cm]
Default:
Propiedades: oculto
Script:
enter:
       if $8 = null then
         goto 8
       endif
       scan:
          answer = scandata
       exit:
           show 11
           beep

Nombre: Altura9[cm]   [BARCODE]
Tipo: freeform text
Nombre de columna: Altura9[cm]
Default:
Propiedades: oculto
Script:
       enter:
       if $9 = null then
         goto 9
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Altura10[cm]   [BARCODE]
Tipo: freeform text
Nombre de columna: Altura10[cm]
Default:
Propiedades: oculto
Script:
       enter:
       if $10 = null then
         goto 10
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Hora
Tipo: time
Nombre de columna: hora
Default:
Propiedades:
Script:
       initialize:
       answer = now

Nombre: Fecha
Tipo: date only
Nombre de columna: Fecha
Default:
Propiedades:
Script:
       initialize:
       answer = now

Nombre: Estadísticas
Tipo: button
Nombre de columna: Estadisticas
Default: Mostrar
Propiedades:
Script:
       click:
       show from 15 to 17
       if $20 = 1 then
          msgbox "Ya realizó los
       cálculos"
          return
       endif
       $21 = 1
       $20 = 1
       if $2 = null then
          $15 = 0
          $16 = 0
          $17 = 0
          hide 4
          return
       endif
       temp = $2
       $16 = $2
       $17 = $2
       if $3 = null then
          $15 = temp
           hide 4
           return
       endif
       if $3 < $2 then
             $16 = $3
       else
             $17 = $3
       endif
       temp = temp + $3
       if $4 = null then
          $15 = temp/2
           hide from 4 to 11
           return
       endif
       if $4 < $16 then
             $16 = $4
       endif
       if $4 > $17 then
             $17 = $4
       endif
       temp = temp + $4
       if $5 = null then
          $15 = temp/3
           hide from 5 to 11
           return
       endif
       if $5 < $16 then
             $16 = $5
       endif
       if $5 > $17 then
             $17 = $5
       endif
       temp = temp + $5
       if $6 = null then
          $15 = temp/4
           hide from 6  to 11
          return
```

```
endif
if $6 < $16 then
      $16 = $6
endif
if $6 > $17 then
      $17 = $6
endif
temp = temp + $6
if $7 = null then
   $15 = temp/5
    hide from 7 to 11
    return
endif
if $7 < $16 then
      $16 = $7
endif
if $7 > $17 then
      $17 = $7
endif
temp = temp + $7
if $8 = null then
   $15 = temp/6
    hide from 8 to 11
    return
endif
if $8 < $16 then
      $16 = $8
endif
if $8 > $17 then
      $17 = $8
endif
temp = temp + $8
if $9 = null then
   $15 = temp/7
    hide from 9 to 11
    return
endif
if $9 < $16 then
      $16 = $9
endif
if $9 > $17 then
      $17 = $9
endif
temp = temp + $9
if $10 = null then
   $15 = temp/8
    hide from 10 to 11
    return
endif
if $10 < $16 then
      $16 = $10
endif
if $10 > $17 then
      $17 = $10
endif
temp = temp + $10
if $11 = null then
   $15 = temp/9
    hide 11
   return
else
temp = temp + $11
   $15 = temp/10
      endif
      if $11 < $16 then
            $16 = $11
      endif
      if $11 > $17 then
            $17 = $11
      endif
```

```
Nombre: Promedio[cm]
Tipo: freeform text
Nombre de columna: Promedio
Default:
Propiedades: oculto
Script:

Nombre: Mínimo[cm]
Tipo: freeform text
Nombre de columna: Minimo
Default:
Propiedades: oculto
Script:

Nombre: Máximo[cm]
Tipo: freeform text
Nombre de columna: Maximo
Default:
Propiedades: oculto
Script:

Nombre: Limpiar valores
Tipo: button
Nombre de columna: Limpiar
Default: Limpiar
Propiedades:
Script:
      click:
      $2 = null
      $3 = null
      $4 = null
      $5 = null
      $6 = null
      $7 = null
      $8 = null
      $9 = null
      $10 = null
      $11 = null
      $17 = null
      $15 = null
      $16 = null
      $21 = 0
      $20 = 0
      hide from 4 to 11
      hide from 15 to 17
      result = "9600-N-8-1"
      temp=U
      transmit serial temp
      result = "9600-N-8-1"
      temp=1
      transmit serial temp

Nombre: Registro nuevo
Tipo: button
Nombre de columna: NvoRegistro
```

```
Default: Crear
Propiedades:
Script:
       click:
       result = "9600-N-8-1"
       temp=U
       transmit serial temp
       result = "9600-N-8-1"
       temp=1
       transmit serial temp
       $2 = null
       $3 = null
       $4 = null
       $5 = null
       $6 = null
       $7 = null
       $8 = null
       $9 = null
       $10 = null
       $11 = null
       $17 = null
       $15 = null
       $16 = null
       $21 = 0
       $20 = 0
       hide from 4 to 11
       hide from 15 to 17
       clone

Nombre: Variable1
Tipo: freeform text
Nombre de columna: Variable1
Default: 0
Propiedades:  oculto
Script:
       calculate:
       if $21 = 1 then
          return
       endif
       temp=0
       if $2 = null then
          hide 4
          return
       endif
       mid $2 1 2
       temp = result
       mid $2 3 2
       result = 16 * result
       temp= temp + result
       mid $2 5 2
       result = result * 16
       result = result * 16
       temp = temp + result
       mid $2 7 2
       result = result * 16
       result = result * 16
       result = result * 16
       temp = temp + result
       temp = temp * .0005
       temp = temp * 1.00255
       temp = temp - .0027169
       temp = temp / 2
       temp = temp * temp
       temp = temp * 4890
       temp = temp + 0.5
       temp = integer temp
       temp = temp / 10
       $2 = temp + 0.4
       if $3 = null then
          hide  from 4 to 11
          return
       endif
       mid $3 1 2
       temp = result
       mid $3 3 2
       result = 16 * result
       temp= temp + result
       mid $3 5 2
       result = result * 16
       result = result * 16
       temp = temp + result
       mid $3 7 2
       result = result * 16
       result = result * 16
       result = result * 16
       temp = temp + result
       temp = temp * .0005
       temp = temp * 1.00255
       temp = temp - .0027169
       temp = temp / 2
       temp = temp * temp
       temp = temp * 4890
       temp = temp + 0.5
       temp = integer temp
       temp = temp / 10
       $3 = temp + 0.4
       if $4 = null then
           hide from 4 to 11
           return
       endif
       mid $4 1 2
       temp = result
       mid $4 3 2
       result = 16 * result
       temp= temp + result
       mid $4 5 2
       result = result * 16
       result = result * 16
       temp = temp + result
       mid $4 7 2
       result = result * 16
       result = result * 16
       result = result * 16
       temp = temp + result
       temp = temp * .0005
       temp = temp * 1.00255
       temp = temp - .0027169
       temp = temp / 2
       temp = temp * temp
       temp = temp * 4890
       temp = temp + 0.5
       temp = integer temp
       temp = temp / 10
       $4 = temp + 0.4
       if $5 = null then
          hide from 5 to 11
```

```
    return
endif
mid $5 1 2
temp = result
mid $5 3 2
result = 16 * result
temp= temp + result
mid $5 5 2
result = result * 16
result = result * 16
temp = temp + result
mid $5 7 2
result = result * 16
result = result * 16
result = result * 16
temp = temp + result
temp = temp * .0005
temp = temp * 1.00255
temp = temp - .0027169
temp = temp / 2
temp = temp * temp
temp = temp * 4890
temp = temp + 0.5
temp = integer temp
temp = temp / 10
$5 = temp + 0.4
if $6 = null then
   hide from 6 to 11
 return
endif
mid $6 1 2
temp = result
mid $6 3 2
result = 16 * result
temp= temp + result
mid $6 5 2
result = result * 16
result = result * 16
temp = temp + result
mid $6 7 2
result = result * 16
result = result * 16
result = result * 16
temp = temp + result
temp = temp * .0005
temp = temp * 1.00255
temp = temp - .0027169
temp = temp / 2
temp = temp * temp
temp = temp * 4890
temp = temp + 0.5
temp = integer temp
temp = temp / 10
$6 = temp + 0.4
if $7 = null then
   hide from 7 to 11
    return
endif
mid $7 1 2
temp = result
mid $7 3 2
result = 16 * result
temp= temp + result
mid $7 5 2
result = result * 16
result = result * 16
temp = temp + result
mid $7 7 2
result = result * 16
result = result * 16
result = result * 16
temp = temp + result
temp = temp * .0005
temp = temp * 1.00255
temp = temp - .0027169
temp = temp / 2
temp = temp * temp
temp = temp * 4890
temp = temp + 0.5
temp = integer temp
temp = temp / 10
$7 = temp + 0.4
answer = 0
```

```
Nombre: Variable2
Tipo: freeform text
Nombre de columna: Variable2
Default: 0
Propiedades:  oculto
Script:
```

```
       calculate:
       if $21 = 1 then
          return
       endif
       if $8 = null then
          hide from 8 to 11
           return
       endif
       mid $8 1 2
       temp = result
       mid $8 3 2
       result = 16 * result
       temp= temp + result
       mid $8 5 2
       result = result * 16
       result = result * 16
       temp = temp + result
       mid $8 7 2
       result = result * 16
       result = result * 16
       result = result * 16
       temp = temp + result
       temp = temp * .0005
       temp = temp * 1.00255
       temp = temp - .0027169
       temp = temp / 2
       temp = temp * temp
       temp = temp * 4890
       temp = temp + 0.5
       temp = integer temp
       temp = temp / 10
       $8 = temp + 0.4
       if $9 = null then
          hide from 9 to 11
           return
       endif
```

```
mid $9 1 2
temp = result
mid $9 3 2
result = 16 * result
temp= temp + result
mid $9 5 2
result = result * 16
result = result * 16
temp = temp + result
mid $9 7 2
result = result * 16
result = result * 16
result = result * 16
temp = temp + result
temp = temp * .0005
temp = temp * 1.00255
temp = temp - .0027169
temp = temp / 2
temp = temp * temp
temp = temp * 4890
temp = temp + 0.5
temp = integer temp
temp = temp / 10
$9 = temp + 0.49
if $10 = null then
   hide from 10 to 11
    return
endif
mid $10 1 2
temp = result
mid $10 3 2
result = 16 * result
temp= temp + result
mid $10 5 2
result = result * 16
result = result * 16
temp = temp + result
mid $10 7 2
result = result * 16
result = result * 16
result = result * 16
temp = temp + result
temp = temp * .0005
temp = temp * 1.00255
temp = temp - .0027169
temp = temp / 2
temp = temp * temp
temp = temp * 4890
temp = temp + 0.5
temp = integer temp
temp = temp / 10
$10 = temp + 0.4
if $11 = null then
   hide 11
   return
endif
mid $11 1 2
temp = result
mid $11 3 2
result = 16 * result
temp= temp + result
mid $11 5 2
result = result * 16
result = result * 16
temp = temp + result
mid $11 7 2
result = result * 16
result = result * 16
result = result * 16
temp = temp + result
temp = temp * .0005
temp = temp * 1.00255
temp = temp - .0027169
temp = temp / 2
temp = temp * temp
temp = temp * 4890
temp = temp + 0.5
temp = integer temp
temp = temp / 10
$11 = temp + 0.4
answer = 0
```

- **Frecuencia de paso**

```
Nombre: Jugador
Tipo: lookup list
Nombre de columna: Jugador
Default: Jorge Campos
Propiedades: ninguna
Script:
       initialize:
       result = "9600-N-8-1"
       temp = U
       transmit serial temp

Nombre: ¿Tiempo-prueba?   (seg)
Tipo: lookup list
Nombre de columna: TiempoPrueba
Default: 5
Propiedades: ninguna
Script:
       calculate:
       if answer > 30 then
            msgbox "El tiempo no
       puede ser mayor de 30
       segundos"
            $2=null
       endif

Nombre: Iniciar Prueba
Tipo: button
Nombre de columna: InicioPrueba
Default: Inicio
Propiedades: ninguna
Script:
       click:
       if $2 <> null then
           readonly 2
       endif
       if $2 = null then
           msgbox "Debe introducir
       el tiempo antes de realizar
       la prueba"
           return
       endif
       temp = $2 / .0005
       temp = temp/16
       temp = temp/16
       temp = temp/16
       result = integer temp
       if result < 10 then
           $24 = "0" & result
       else
           $24 = result
       endif
       temp = temp - result
       temp = temp*16
       result = integer temp
       if result < 10 then
           result = "0" & result
           $24 = result & $24
       else
           $24 = result & $24
       endif
       temp = temp - result
       temp = temp*16
       result = integer temp
       if result < 10 then
           result = "0" & result
           $24  = result & $24
       else
           $24 = result & $24
       endif
       temp = temp - result
       temp = temp*16
       result = integer temp
       if result < 10 then
           result = "0" & result
           $24 = result & $24
       else
           $24 = result & $24
       endif
       result = "9600-N-8-1"
       temp=U
       transmit serial temp
       result = "9600-N-8-1"
       temp = 0
       transmit serial temp
       result = "9600-N-8-1"
       temp = $24
       transmit serial temp

Nombre: Frec1[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec1
Default:
Propiedades: ninguna
Script:
       scan:
              answer = scandata
       exit:
       beep

Nombre: Frec2[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec2
Default:
Propiedades: oculto
Script:
       enter:
       if $4 = null then
         goto 4
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Frec3[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec3
Default:
Propiedades: oculto
Script:
       enter:
       if $5 = null then
         goto 5
       endif
```

```
       scan:
          answer = scandata
       exit:
           beep

Nombre: Frec4[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec4
Default:
Propiedades: oculto
Script:
       enter:
       if $6 = null then
         goto 6
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Frec5[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec5
Default:
Propiedades: oculto
Script:
       enter:
       if $7 = null then
         goto 7
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Frec6[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec6
Default:
Propiedades: oculto
Script:
enter:
       if $8 = null then
         goto 8
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Frec7[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec7
Default:
Propiedades: oculto
Script:
       enter:
       if $9 = null then
         goto 9
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Frec8[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec8
Default:
Propiedades: oculto
Script:
       enter:
       if $10 = null then
         goto 10
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Frec9[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec9
Default:
Propiedades: oculto
Script:
       enter:
       if $11 = null then
         goto 11
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Frec10[paso/s]   [BARCODE]
Tipo: freeform text
Nombre de columna: Frec10
Default:
Propiedades: oculto
Script:
       enter:
       if $12 = null then
         goto 12
       endif
       scan:
          answer = scandata
       exit:
           beep

Nombre: Hora
Tipo: time
Nombre de columna: hora
Default:
Propiedades:
Script:
       initialize:
       answer = now

Nombre: Fecha
Tipo: date only
Nombre de columna: fecha
Default:
Propiedades:
Script:
       initialize:
```

```
      answer = now

Nombre: Prueba nueva
Tipo: button
Nombre de columna: NvaPrueba
Default: Crear
Propiedades:
Script:
      click:
      if $4 = null  then
         msgbox "Realice la prueba
      anterior antes de crear una
      nueva"
         return
      endif
      result = "9600-N-8-1"
      temp=U
      transmit serial temp
      result = "9600-N-8-1"
      temp = 0
      transmit serial temp
      result = "9600-N-8-1"
      temp = $24
      transmit serial temp
      if $23 = 0 then
         show 5
         $23 = $23 + 1
          return
      endif
      if $5 = 0 then
         msgbox "Realice la prueba
      anterior antes de crear una
      nueva"
         return
      endif
      if $23 = 1 then
         show 6
         $23 = $23 + 1
          return
      endif
      if $6 = 0 then
         msgbox "Realice la prueba
      anterior antes de crear una
      nueva"
         return
      endif
      if $23 = 2 then
         show 7
         $23 = $23 + 1
           return
      endif
      if $7=0 then
         msgbox "Realice la prueba
      anterior antes de crear una
      nueva"
         return
      endif
      if $23 = 3 then
         show 8
         $23 = $23 + 1
          return
      endif
      if $8=0 then
         msgbox "Realice la prueba
      anterior antes de crear una
      nueva"
         return
      endif
      if $23 = 4 then
         show 9
         $23 = $23 + 1
          return
      endif
      if $9=0 then
         msgbox "Realice la prueba
      anterior antes de crear una
      nueva"
         return
      endif
      if $23 = 5 then
         show 10
         $23 = $23 + 1
          return
      endif
      if $10=0 then
         msgbox "Realice la prueba
      anterior antes de crear una
      nueva"
         return
      endif
      if $23 = 6 then
         show 11
         $23 = $23 + 1
          return
      endif
      if $11=0 then
         msgbox "Realice la prueba
      anterior antes de crear una
      nueva"
         return
      endif
      if $23 = 7 then
         show 12
         $23 = $23 + 1
         return
      endif
      if $12=0 then
         msgbox "Realice la prueba
      anterior antes de crear una
      nueva"
         return
      endif
      if $23 = 8 then
         show 13
         $23 = $23 + 1
         return
      endif

Nombre: Estadísticas
Tipo: button
Nombre de columna: Estadisticas
Default: Mostrar
Propiedades:
Script:
      click:
      if $25 = 1 then
```

```
   msgbox "Ya realizó los
cálculos"
   return
endif
$25 = 1
$26 = 1
show 19
show 18
show 20
if $4 = null then
   $20 = 0
   $18 = 0
   $19 = 0
   return
endif
temp = $4
$20 = $4
$19 = $4
if $5 = null then
   $18 = temp
    return
endif
if $5 < $4 then
      $19 = $5
else
      $20 = $5
endif
temp = temp + $5
if $6 = null then
   $18 = temp/2
    return
endif
if $6 < $19 then
      $19 = $6
endif
if $6 > $20 then
      $20 = $6
endif
temp = temp + $6
if $7 = null then
   $18 = temp/3
    return
endif
if $7 < $19 then
      $19 = $7
endif
if $7 > $20 then
      $20 = $7
endif
temp = temp + $7
if $8 = null then
   $18 = temp/4
   return
endif
if $8 < $19 then
      $19 = $8
endif
if $8 > $20 then
      $20 = $8
endif
temp = temp + $8
if $9 = null then
   $18 = temp/5
   return
endif
if $9 < $19 then
      $19 = $9
endif
if $9 > $20 then
      $20 = $9
endif
temp = temp + $9
if $10 = null then
   $18 = temp/6
   return
endif
if $10 < $19 then
      $19 = $10
endif
if $10 > $20 then
      $20 = $10
endif
temp = temp + $10
if $11 = null then
   $18 = temp/7
   return
endif
if $11 < $19 then
      $19 = $11
endif
if $11 > $20 then
      $20 = $11
endif
temp = temp + $11
if $12 = null then
   $18 = temp/8
   return
endif
if $12 < $19 then
      $19 = $12
endif
if $12 > $20 then
      $20 = $12
endif
temp = temp + $12
if $13 = null then
   $18 = temp/9
   return
endif
if $13 < $19 then
      $19 = $13
endif
if $13 > $20 then
      $20 = $13
endif
temp = temp + $13
$18 = temp/10

Nombre: Promedio[paso/s]
Tipo: freeform text
Nombre de columna: Promedio
Default:
Propiedades: oculto
Script:
Nombre: Mínimo[paso/s]
Tipo: freeform text
```

```
Nombre de columna: Minimo
Default:
Propiedades: oculto
Script:

Nombre: Máximo[paso/s]
Tipo: freeform text
Nombre de columna: Máximo
Default:
Propiedades: oculto

Nombre: Limpiar valores
Tipo: button
Nombre de columna: Limpiar
Default: limpiar
Poripedades:
Script:
       click:
       $2 = null
       $4 = null
       $5 = null
       $6 = null
       $7 = null
       $8 = null
       $9 = null
       $10 = null
       $11 = null
       $12 = null
       $13 = null
       $20 = null
       $18 = null
       $19 = null
       $24 = 0
       $25 = 0
       $26 = 0
       $23 = 0
       $27 = 0
       hide from 5 to 13
       hide from 18 to 20
       readwrite 2

Nombre: Registro nuevo
Tipo: button
Nombre de columna: NvoRegistro
Default: Crear
Poripedades:
Script:
       click:
       clone
       $2 = null
       $4 = null
       $5 = null
       $6 = null
       $7 = null
       $8 = null
       $9 = null
       $10 = null
       $11 = null
       $12 = null
       $13 = null
       $20 = null
       $18 = null
       $19 = null
       $24 = 0
       $25 = 0
       $26 = 0
       $23 = 0
       $27 = 0
       hide from 5 to 13
       hide from 18 to 20
       readwrite 2

Nombre: Variable1
Tipo: freeform text
Nombre de columna: Variable1
Default: 0
Propiedades: oculto
Script:
       calculate:
       if $26 = 20  then
            return
       endif
       temp=0
       if $4 = null then
           return
       endif
       mid $4 1 2
       temp = result
       mid $4 3 2
       result = 16 * result
       temp= temp + result
       if $26 < 1 then
           $4 = temp /$2
           $26 = $26 + 1
       endif
       if $5 = null then
              return
       endif
       mid $5 1 2
       temp = result
       mid $5 3 2
       result = 16 * result
       temp= temp + result
       if $26 < 2 then
           $5 = temp /$2
           $26 = $26 + 1
       endif
       if $6 = null then
           return
       endif
       mid $6 1 2
       temp = result
       mid $6 3 2
       result = 16 * result
       temp= temp + result
       if $26 < 3 then
           $6 = temp /$2
           $26 = $26 + 1
       endif
       if $7 = null then
           return
       endif
       mid $7 1 2
       temp = result
       mid $7 3 2
       result = 16 * result
```

```
        temp = temp + result
        if $26 < 4 then
            $7 = temp /$2
            $26 = $26 + 1
        endif
        if $8 = null then
         return
        endif
        mid $8 1 2
        temp = result
        mid $8 3 2
        result = 16 * result
        temp= temp + result
        if $26 < 5 then
            $8 = temp /$2
            $26 = $26 + 1
        endif

Nombre: Variable2
Tipo: freeform text
Nombre de columna: Variable2
Default: 0
Propiedades: oculto
Script:
        calculate:
        if $26 = 1 then
             return
        endif
        if $9 = null then
         return
        endif
        mid $9 1 2
        temp = result
        mid $9 3 2
        result = 16 * result
        temp= temp + result
        if $26 < 6 then
            $9 = temp /$2
            $26 = $26 + 1
        endif
        if $10 = null then
         return
        endif
        mid $10 1 2
        temp = result
        mid $10 3 2
        result = 16 * result
        temp= temp + result
        if $26 < 7 then
            $10 = temp /$2
            $26 = $26 + 1
        endif

        if $11 = null then
         return
        endif
        mid $11 1 2
        temp = result
        mid $11 3 2
        result = 16 * result
        temp= temp + result
        if $26 < 8 then
            $11 = temp /$2
            $26 = $26 + 1
        endif
        if $12 = null then
         return
        endif
        mid $12 1 2
        temp = result
        mid $12 3 2
        result = 16 * result
        temp= temp + result
        if $26 < 9 then
            $12 = temp /$2
            $26 = $26 + 1
        endif
        if $13 = null then
           return
        endif
        mid $13 1 2
        temp = result
        mid $13 3 2
        result = 16 * result
        temp= temp + result
        if $26 < 10 then
            $13 = temp /$2
            $26 = $26 + 1
        endif

Nombre: Variable3
Tipo: freeform text
Nombre de columna: Variable3
Default: 0
Propiedades: oculto
Script:

Nombre: Variable4
Tipo: freeform text
Nombre de columna: Variable4
Default: 0
Propiedades: oculto
Script:

Nombre: Variable5
Tipo: freeform text
Nombre de columna: Variable5
Default: 0
Propiedades: oculto
Script:
```

A.3.2 Conversión de sistema hexadecimal a sistema decimal

Como se analizó en la sección A.2, la *Palm* recibe un total de 8 dígitos que representan al tiempo calculado en las pruebas. Aunque en la programación del PIC se desarrolló un algoritmo para convertir los dígitos hexadecimales a su correspondiente valor decimal, el dato en su totalidad sigue estando en sistema hexadecimal.

Cada dígito hexadecimal está representado por dos dígitos decimales, uno para las unidades (u) y otro para las decenas (d), y recordando que el par de dígitos que están a la izquierda representan al valor menos significativo, el número recibido en ASCII, al transformarse tiene la siguiente forma:

$$D_{d0}D_{u0}\ D_{d1}D_{u1}D_{d2}D_{u2}\ D_{d3}D_{u3}$$

Atendiendo a la posición que ocupa cada dígito dentro del número, para obtener el número decimal (Nd) se realiza la siguiente operación:

$$16^0 \times D_{d0}D_{u0} + 16^1 \times D_{d1}D_{u1} + 16^2 \times D_{d2}D_{u2} + 16^3 \times D_{d3}D_{u3} = Nd$$

En el caso de la prueba de frecuencia de paso (tomando en cuenta que en este caso sólo se recibió $D_{d0}D_{u0}\ D_{d1}D_{u1}$) Nd representa el número de pasos que la persona realizó en el tiempo establecido, y para obtener la frecuencia de paso únicamente se divide ese valor entre el tiempo en el que se realizó la prueba.

Ahora bien, como se requiere que la *Palm* envíe el valor del tiempo durante el cual va a efectuarse la prueba de frecuencia de paso, también es necesario llevar a cabo la transformación del tiempo de sistema decimal a sistema hexadecimal, para lo que se realizan las siguientes operaciones:

$$Nd / 16^3 = D_{d3}D_{u3}.residuo1$$
$$residuo1 * 16 = D_{d2}D_{u2}.residuo2$$
$$residuo2 * 16 = D_{d1}D_{u1}.residuo3$$
$$residuo3 * 16 = D_{d0}D_{u0}$$

en el que se considera que "$D_{d3}D_{u3}$" representa la parte entera del resultado, y "residuo1" la parte fraccionaria, tomando como ejemplo la primera operación. Además debe tomarse en consideración que cuando el entero resultante de la operación es menor a 10, $D_d = 0$. El acomodo final del número que va a enviarse, queda de la siguiente forma:

$$D_{d0}D_{u0}\ D_{d1}D_{u1}D_{d2}D_{u2}\ D_{d3}D_{u3}$$

La programación de las operaciones anteriores se lleva a cabo en el "campo" de "Inicio de prueba" de la "forma" de frecuencia de paso, en el cual también se muestra la instrucción utilizada para enviar los datos a través del puerto serie de la *Palm* con el formato adecuado.

REFERENCIAS

1 Cooper, W, Instrumentación electrónica y mediciones, Prentice Hall, México, (1982), 501.

2 Gregory, B, Instrumentación electrónica y sistemas de medida, Gustavo Gili, Barcelona, (1984), 469.

3 Hegedüs, J, Técnicas Atléticas, Stadium, República de Argentina, 175.

4 Houvion, M, Tratado de atletismo: saltos, Hispano-Europea, (1986), 449.

5 Khambata, A, Microprocesadores/Microcomputadores, Gustavo Gili, México, (1987), 536.

6 Malvino, A, Principios de electrónica, McGraw-Hill, 2ª ed., México, (1989), 872.

7 Mykland, R, Palm OS programmer and developer, McGraw–Hill, USA, (2000), 507.

8 Pendragon Software Corporation, Pendragon Forms, Pendragon Software Corporation, USA, (2000), 242.

9 Tocci, R, Microprocessors and Microcomputers, Prentice Hall, 4ª ed., USA, 565.

10 Warren, C, Roark's Formulas for Stress & Strain, McGraw-Hill, 6ª ed., USA, (1989), 763.

11 http://www.digikey.com

12 http://www.fairchildsemi.com

13 http://www.jameco.com

14 http://www.jmengual.com/Manuales/Manual_PIC_02.pdf

15 http://www.laserfx.com/

16 http://www.laser2000.co.uk/lasers/modules/semicon.htm

17 http://www.monografias.com/trabajos/laser/laser.shtml

18 http://www.nrpb.org/press/information_sheets/laser_pointers.htm

19 http://www.nullmodem.com/

20 http://www.palm.com

21 http://www.parallaxinc.com