


Mediciones Ambientales

Guía del Estudiante para Experimentos 1 al 6

Versión en Castellano 1.0

Sobre la precisión de este texto:

Se realizó un gran esfuerzo para asegurar la precisión de este texto y los experimentos, pero puede haber errores aún. Si usted encuentra errores o algún tema que requiera información adicional, por favor infórmelo a stampsinclass@parallaxinc.com, así podemos continuar mejorando la calidad de nuestra documentación.

PARALLAX 

Garantía

Parallax garantiza sus productos contra defectos en sus materiales o debidos a la fabricación por un periodo de 90 días. Si usted descubre un defecto, Parallax según corresponda, reparará, reemplazará o regresará el valor de la compra. Simplemente pida un número de autorización de regreso de mercadería (Return Merchandise Authorization, RMA), escriba el número en el exterior de la caja y envíela a Parallax. Por favor incluya su nombre, número telefónico, dirección, y una descripción del problema. Nosotros le regresaremos su producto o el reemplazo, usando el mismo método de correo que usted usó para enviar el producto a Parallax.

Garantía de 14 días de regreso de dinero

Si dentro de los 14 días en que usted recibió su producto, encuentra que no es conveniente para sus necesidades, puede regresarlo, recibiendo un reembolso. Parallax regresará el precio de compra del producto, excluyendo los costos de manipuleo y correo. Esto no se aplica si el producto a sido alterado o dañado.

Derechos de Copia y Marcas Registradas

Esta documentación tiene derechos de copia Copyright 1999 por Parallax, Inc. BASIC Stamp (Estampilla BASIC) es una marca registrada de Parallax, Inc. Si usted decide usar el nombre BASIC Stamp en su página web o en material impreso, debe agregar la aclaración: BASIC Stamp es una marca registrada de Parallax, Inc. Otros nombres de productos son marcas registradas de sus respectivos dueños.

Desvinculación de Responsabilidad

Parallax, Inc. no es responsable de daños por consecuencias, incidentes o daños especiales que resulten de cualquier violación de la garantía, bajo cualquier teoría legal, incluyendo pérdida de beneficio, tiempo, daño o reemplazo de equipo o propiedad y cualquier costo, recuperando, reprogramando o reproduciendo cualquier dato guardado o usado dentro de los productos Parallax. Parallax tampoco es responsable de cualquier daño personal, incluyendo vida o muerte, resultado del uso de cualquiera de nuestros productos. Usted tiene absoluta responsabilidad por la aplicación que desarrolle con el BASIC Stamp.

Acceso en Internet

Mantenemos sistemas de Internet para su uso. Estos pueden ser usados para obtener software, comunicarse con miembros de Parallax, y comunicarse con otros clientes. Las rutas de acceso a la información se muestran a continuación:

E-mail: stampsinclass@parallaxinc.com
Ftp: [ftp.parallaxinc.com](ftp://ftp.parallaxinc.com) y [ftp.stampsinclass.com](ftp://ftp.stampsinclass.com)
Web: <http://www.parallaxinc.com> y <http://www.stampsinclass.com>

Lista de Discusión de BASIC Stamp en Internet (En Inglés)

Mantenemos dos listas de discusión por e-mail para gente interesada en el BASIC Stamp. La lista trabaja así: mucha gente se suscribe a la lista y luego todas las preguntas y respuestas son distribuidas a todos los suscriptos. Es una forma rápida, divertida y gratis de discutir temas sobre el BASIC Stamp y obtener respuestas a preguntas técnicas. Para suscribirse a la lista de BASIC Stamp mande un e-mail a majordomo@parallaxinc.com y escriba *subscribe stamps* en el cuerpo del mensaje. Esta lista genera aproximadamente 40 mensajes diarios.

También mantenemos una lista exclusiva para educadores que usan el BASIC Stamp en el aula. Usted puede unirse a esta lista en el sitio web <http://www.stampsinclass.com>. Esta lista genera aproximadamente 5 mensajes diarios.

Contenido

Prefacio.....	3
Destinatarios y Guías para Profesores.....	3
Derechos de Copia y Reproducción.....	4
Agradecimientos Especiales.....	4
Experimento 1: Transductores de Temperatura y de Sonido	7
Partes Requeridas	8
¡Constrúyalo!	8
¡Prográmelo!	10
Transductor de Temperatura	10
Código Morse	12
¡Desafío!.....	24
Experimento 2: Adquisición de Datos	25
Partes Requeridas	26
¡Constrúyalo!	26
¡Prográmelo!	27
Tema Avanzado: Detectando un Doble-Click	32
Aprendiendo lo Básico de READ y WRITE.....	34
Termómetro que Habla, Revisión del Código Morse	38
¡Desafío!.....	46
Experimento 3: Punta de Temperatura para Micro-Ambientes.....	49
Partes Requeridas	49
Sensor de Temperatura Analógico.....	50
Pines del BASIC Stamp, Capacitores, Revisión de BASIC.....	51
Detector de Resistencia Simple.....	53
Sensor de Resistencia Usando RTime	57
Punta de Sensado de Temperatura Usando el AD592 y RTime.....	59
Calibración del AD592.....	62
Revisión del Termómetro que Habla, Dos Canales	66
Calibración Automática (Tema Avanzado).....	68
Algunos Experimentos de Medición de la Temperatura Ambiental	71
¡Desafío!.....	74
Experimento 4: Luz en la Tierra y Adquisición de Datos	77
Hágase la luz	77
Partes Requeridas	78
¡Constrúyalo!	79
Fotodiodo como Transductor de Luz	79
Fotodiodo y BASIC Stamp como un Medidor de Luz Digital	88

Medidor de Luz y Temperatura.....	93
Almacenamiento de Datos de Temperatura y Luz, Usando Memoria RAM.....	94
Experimentos con el Data Logger	99
¡Desafío!.....	102
Experimento 5: El Ambiente Líquido.....	103
Introducción	103
Partes Requeridas	104
¡Constrúyalo!	104
Alarma de Humedad	104
Medición de Conductancia Usando RCTime	108
Medición de Conductancia Usando el CI Temporizador 555.....	110
Conductancia en el Agua.....	116
Continuación de Almacenamiento de Datos: Secado de Suelos.....	119
Experimentos Adicionales para Intentar.....	122
¡Desafío!.....	125
Experimento 6: Medición y Control	127
Introducción	127
Partes Requeridas	128
¡Constrúyalo!	128
Control de Encendido-Apagado de Bombas	129
Control de Bombas con Realimentación.....	135
Memoria en el BASIC Stamp, Revisión	137
Almacenador de Datos.....	143
Otras Investigaciones	153
¡Desafío!.....	154
Apéndice A: Lista de componentes y Suministros.....	155
Apéndice B: Construcción de la Punta de Temperatura AD592	161
Apéndice C: Código de Colores de Resistores	163
Apéndice D: Hojas de Datos	165

Prefacio

El tema de estos seis experimentos es Mediciones Ambientales. Imagine ser un geólogo, intentando saber más sobre El Niño, el famoso efecto en las aguas costeras de Sudamérica que cambia los patrones climáticos del mundo. Va a necesitar realizar muchas mediciones. O imagine ser un operador de una planta de tratamiento de agua, donde una ciudad llena de personas cuenta con que usted les suministre agua pura noche y día. Va a tener que monitorear el agua y operar una planta computarizada para bombearla a través de toda la ciudad. O imagine ser el responsable de una plantación de manzanos. Va a necesitar seguir de cerca los cambios climáticos para controlar el riego, control de plagas y llevar su producción al mercado. Estos son algunos ejemplos de lo que queremos decir cuando nos referimos a Mediciones Ambientales.

Por supuesto las Mediciones Ambientales fueron de interés para los humanos mucho antes de que las microcomputadoras fueran incluso imaginadas. Aunque se sitúe en el pasado 50 años o 500, o incluso 5.000 años, usted puede imaginarse humanos controlando los cambios en el viento. Las computadoras, especialmente las pequeñas, algunas especializadas, permiten realizar mediciones donde ningún humano puede ir, y más importante, permiten realizar muchas mediciones. Las computadoras pueden almacenar todos esos datos e incluso ponerlos a disposición de la red mundial de computadoras. También pueden ser programadas para hacer cosas automáticamente como encender una bomba de agua cuando un campo necesita riego. Estas cosas difícilmente pudieran haber sido imaginadas 50 años atrás.

Las Mediciones Ambientales no difieren mucho de las mediciones en otros campos. Por ejemplo, electrodomésticos como secarropas, hornos a microondas y termostatos usan microcontroladores para medición y control, al igual que los instrumentos en la industria, el laboratorio, el hospital, y más allá de la tierra en el espacio. Las técnicas de medición en estos sitios tan diferentes son las mismas. Lo que aprenda en este curso se generaliza a muchos campos fuera de las mediciones ambientales. Pero reconozcamos que la salud de nuestro planeta depende en gran medida del conocimiento que se obtiene a través de las mediciones. Hay muchas carreras interesantes que combinan el conocimiento en electrónica con la pasión por el medio ambiente, como los científicos, ingenieros, o agricultores modernos.

Destinatarios y Guías para Profesores

El curriculum Mediciones Ambientales fue creado para edades de 17 años en adelante, como continuación de la guía ¿Qué es un Microcontrolador?. Como todos los curriculums de Stamps en Clase, éste enseña nuevas técnicas y circuitos con una mínima superposición con los otros textos. Los temas introducidos son: sistema de control por realimentación de bucle cerrado, comunicación serial, uso de la EEPROM del BASIC Stamp, calibración de sensores, conductividad en el agua, y el uso de un transductor de sonido para realimentación con los humanos.

La profundidad y disponibilidad de la Guía para Profesores (en inglés) varía entre los curriculums de Stamps en Clase. Debido a que expertos en cada campo escriben cada juego de experimentos, los mismos son provistos en formatos muy diversos. La Earth Measurement Teacher's Guide estará disponible por pedido por e-mail a stampsinclass@parallaxinc.com.

Derechos de Copia y Reproducción

El curriculum Stamps en Clase tiene derecho de copia © Parallax 1999. Parallax le garantiza a cada persona derechos condicionales de descargar, duplicar y distribuir este texto sin nuestro permiso. La condición es que este texto o cualquier parte de él, no debería ser duplicada para uso comercial, resultando en gastos para el usuario, más allá del costo de la impresión. Es decir, *nadie* deberá lucrar por la duplicación de este texto. Preferentemente, la duplicación no tendrá costo para el estudiante. Cualquier institución educativa que desee producir duplicados para los estudiantes, puede hacerlo sin nuestro permiso. Este texto también está disponible en formato impreso de Parallax. Debido a que imprimimos el texto en volumen, el precio al cliente es a menudo menor que el de una típica duplicación xerográfica. Este texto puede ser traducido a cualquier otro idioma, con el permiso de Parallax, Inc.

Agradecimientos Especiales

Tracy Allen Ph.D. de Electronically Monitored Ecosystems, ubicado en Berkeley, California escribió este curriculum (<http://www.emesystems.com>). EME Systems diseña y fabrica instrumentos para ciencia ambiental. Algunos de sus productos son genéricos, y otros son sistemas personalizados para clientes individuales. El comercialmente disponible OWL2C usa un microcontrolador BASIC Stamp II o IISX, proveyéndole programabilidad a los clientes que no desean usar el programador. Dr. Allen tiene un interés particular en programas que traten el manejo de plagas en el campo, uso eficiente de los recursos naturales, y relevamiento de especies o ecosistemas en peligro. Un proyecto reciente del Dr. Allen consiste en la medición de la temperatura corporal externa de las vacas lecheras para evaluar su productividad. Dr. Allen es un contribuyente frecuente a las listas de Parallax BASIC Stamp y Stamps en Clase. Parallax esta muy agradecido por su contribución con el programa Stamps en Clase.

Agradecemos especialmente también al grupo de Parallax que nos proveyó de ideas y contenidos para el programa. Los miembros de Parallax que diseñan, fabrican, reciben los pedidos y envían los productos de Stamps en Clase, son una parte clave del programa Stamps en Clase.

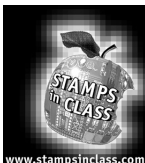
Traducción

La Versión en Castellano 1.0 de Mediciones Ambientales , es la traducción correspondiente a Earth Measurements , Version 1.1.

Traducido y adaptado al castellano por Arístides Alvarez y Ana Lusi de Alvarez. Si encuentra errores en el texto, contáctese con nosotros, para poder mejorar la calidad de la documentación en castellano.

e-mail: aristides@copetel.com.ar
Mar del Plata
Argentina

Experimento 1: Transductores de Temperatura y de Sonido



Experimento 1: Transductores de Temperatura y de sonido

La temperatura es la variable más importante en Mediciones Ambientales. Solamente necesitamos un transductor para medir la temperatura, y otro transductor para interpretar la lectura mediante nuestro ojos u oídos.

En el primer experimento, usted comenzará comprobando con un transductor que convierte los impulsos eléctricos del BASIC Stamp en tonos musicales. Este efecto auditivo le proveerá una información útil sobre la operación del BASIC Stamp a lo largo de esta serie de experimentos. La parte más importante de este experimento será instalar un sensor de temperatura digital en su Plaqueta de Educación. El sensor también es un transductor que convierte el valor de la temperatura en un código que el BASIC Stamp puede entender. El BASIC Stamp tomará estos valores y los mostrará en la pantalla de la computadora. En esta serie de experimentos, vamos a medir la temperatura con dos métodos diferentes, para demostrar la versatilidad del BASIC Stamp para realizar estas tareas.

Vamos ahora a medir temperatura. Usted probablemente este sentado en una habitación confortable, con una temperatura de 17 a 30 grados Celsius (63 a 86 grados Fahrenheit). Puede haber un termostato en la habitación que mantenga este valor agradable, usando un calefactor o un aire acondicionado (o tal vez no!?) ¿Piensa que la temperatura en la habitación es la correcta? ¿Y en el exterior? Si no tiene un termómetro, no se preocupe, tendrá uno antes de finalizar este experimento. Sabemos a través de nuestra experiencia personal que la temperatura es muy importante para nuestro bienestar.

Vivimos en un planeta que está a la distancia correcta del sol y tiene el tipo de atmósfera correcto para producir un rango de temperaturas favorables para el desarrollo de la vida (como la que conocemos). Mediante la industria y la tecnología humanas, desde ropa y vivienda hasta modernos controles de ambiente electrónicos, hemos extendido la superficie habitable del planeta.

No es muy exagerado decir que cada proceso en el planeta depende de la temperatura en alguna medida. Piense en la erosión de las montañas. Cada año el agua se introduce por las grietas de las rocas, se congela, se expande, y la rompe en pedazos. Casi todo fenómeno climático depende críticamente de la temperatura (nieve, lluvia, nubes, viento). Un cambio de unas décimas de grado en la temperatura del agua en el sur del Océano Pacífico (El Niño) puede afectar el clima de todo el mundo. Como crecen las manzanas en los árboles, como se alimentan los gusanos en las manzanas, como prosperan los mosquitos en las aguas estancadas, como sobreviven los renacuajos para comerse a los mosquitos, todo lo relativo a la agricultura y la biología depende de la temperatura. Agregue a esto el ambiente en fábricas, hospitales, laboratorios, escuelas, casas, museos, etc. Basta decir que si va a seguir cualquier carrera relacionada con el ambiente o los microcontroladores, va a tener que saber como medir temperatura.

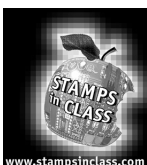
Experimento 1: Transductores de Temperatura y de Sonido



Partes Requeridas

En este experimento necesitará los siguientes componentes, además de un BASIC Stamp II y una Plaqueta de Educación:

- (1) transductor piezoeléctrico
- (1) Sensor de Temperatura DS1620
- (1) resistor de $1K \Omega$ (marrón negro rojo)
- (1) capacitor de $0.1 \mu F$
- (varios) cables de interconexión



¡Constrúyalo!

Siempre es bueno comenzar con un proyecto simple, para entrar en tema. Esto será una constante en esta serie de experimentos. Comenzaremos con un proyecto de precalentamiento, y luego nos centraremos en el foco principal de este experimento. El ejercicio de precalentamiento para este experimento es simplemente un zumbador, un dispositivo emisor de sonido. En

¿Qué es Unim?

Transductor piezoeléctrico:

Piezo deriva del griego y significa presionar, y eléctrico deriva de la palabra griega ámbar (mineral que acumula una carga de electricidad estática cuando es frotado). Algunos cristales, como el cuarzo y materiales plásticos y cerámicos, generan electricidad cuando se les aplica una torsión. Este es el llamado efecto piezoeléctrico. Cables conectados a la superficie del material pueden recoger esta electricidad. Esta es la base de funcionamiento de algunos tipos de micrófonos. Un micrófono es un transductor (Latín de "transformador") que transforma sonido en electricidad. El efecto piezoeléctrico también trabaja en el sentido inverso. Si se aplica electricidad sobre materiales piezoeléctricos, ellos vibran. Pueden ser fabricados con la forma de discos delgados, con conexiones eléctricas en ambas caras, y cables conectados. El disco es como un pequeño tambor. Cuando se conecta a una tensión alterna que varía rápidamente, vibra comprimiendo el aire y emitiendo ondas sonoras, pasando a llamarse transductor piezoeléctrico. Convierte la electricidad en sonido. La tensión alterna debe estar en el rango de frecuencia correcto para resonar con el tono natural del pequeño disco. También se produce un transductor piezoeléctrico con circuitería adicional, de forma que al conectarle la alimentación, suena a un tono prefijado en la fabricación. Este componente se denomina zumbador piezoeléctrico. El dispositivo que usaremos es un simple transductor piezoeléctrico. Solo emitirá sonido cuando le entreguemos pulsos eléctricos en frecuencia de audio desde el BASIC Stamp.

términos técnicos, es un indicador, o un **transductor piezoeléctrico**. Será el componente principal de interfase del usuario en todos los proyectos de este libro. Además, podremos ver también resultados de las mediciones en la computadora, siempre y cuando la Plaqueta de Educación esté conectada con su cordón umbilical. Tener un indicador auditivo nos permitirá alejarnos de la computadora, quedar a oscuras o expuestos a los rayos del sol, y aún ser capaces de escuchar qué está sucediendo.

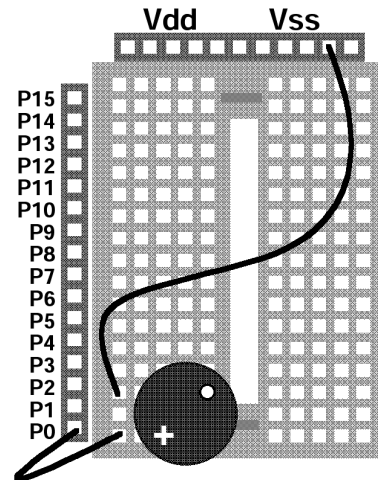
El transductor piezoeléctrico que encontrará en su kit de componentes es un cilindro plástico negro con dos pines (conectores) que salen por la base y un hueco en la cara superior.

Experimento 1: Transductores de Temperatura y de Sonido

Figura 1.1: Transductor piezoeléctrico

- Insértelo con el ángulo que se muestra en la figura
- EL pin con el signo $+$ se conecta a P0
- El otro pin se conecta a Vss

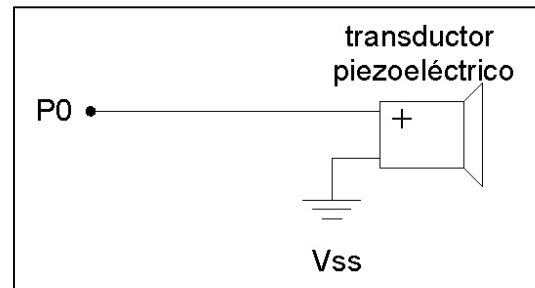
Nota: Los seis experimentos del libro agregarán componentes de unidad en unidad, en la Plaqueta de Educación. Para evitar rearmar los circuitos, siga la distribución de componentes sugerida en las figuras.



Uno de los pines está rotulado con un signo $+$. Conéctelo a la Plaqueta de Educación como se muestra en la Figura 1.1. El circuito se muestra en la Figura 1.2.

Figure 1.2: Esquema eléctrico del Transductor Piezoeléctrico

Circuito eléctrico de la imagen mostrada en la Figura 1.1



Experimento 1: Transductores de Temperatura y de Sonido



¡Prográmelo!

Este experimento consta de tres secciones más pequeñas: transductor piezoeléctrico, código Morse, y medición de temperatura. El proyecto es progresivo.

Transductor de Temperatura

Para emitir sonido con el transductor piezoeléctrico, el BASIC Stamp debe suministrar una señal de alta frecuencia en P0. El comando PBASIC para hacer esto es `freqout`. Es una contracción de "frequency output" (frecuencia de salida). Ejecute el editor del BASIC Stamp en la PC y escriba el siguiente programa PBASIC:

```
freqout 0,1000,1900
```

En caso de problemas en la descarga (download):

En las versiones de software para Windows y DOS, si RUN le da el mensaje "hardware not found" o "communication error", asegúrese de que el cable que conecta la PC a la Plaqueta de Educación esté bien. También asegúrese de que la Plaqueta de Educación esté bien alimentada y que el indicador luminoso de alimentación de la plaqueta esté encendido.

Si ve un mensaje que indica un error en el programa, busque errores de escritura. Si el programa está bien y ALT-R es aceptado sin mensajes de error, pero no se escucha ningún sonido, entonces revise el cableado de la Plaqueta de Educación. Compárelo con el cableado que aparece en la Figura 1.1.

Listo. Un programa de una línea.

Ahora revise el cable de conexión entre la Plaqueta de Educación y la PC. Mantenga presionada la tecla "ALT" y presione la letra "R", por "run" (ejecutar) en la versión DOS y ALT/R y "enter" en la versión de Windows. Esto le debe ser familiar si realizó los experimentos de ¿Qué es un Microcontrolador?. Si todo está bien, debería escuchar un sonido agudo. Cada vez que presione el botón reset en la Plaqueta de Educación, lo escuchará nuevamente. El botón reset está en la Plaqueta de Educación cerca de una esquina, y claramente rotulado Reset. Puede presionarlo todas las veces que quiera, sin problemas. Presionar el botón reinicia el programa desde la primera línea sin borrarlo.

Hay tres **parámetros** en el comando `freqout`:

```
freqout 0,1000,1900
      ^^^^--determina la frecuencia de salida de 1900 hertz
      ^^^^-----hace durar el tono 1 segundo 1000 milisegundos
      ^^-----usa la línea de señal P0 para el tono
```

Si pudiéramos observar el voltaje en P0 durante el comando `freqout`, encontraríamos que sube y baja de 0 a 5 volts muy rápidamente, obteniéndose una senoide de 1900-hertz durante 1 segundo. Si tiene un frecuencímetro intente medir la frecuencia. Para más información sobre el funcionamiento de los comandos `freqout` y `PWM`, ver el BASIC Stamp Manual Version 1.9 (en Inglés).

¿Qué es un...

Parámetro:

Un parámetro es un número que gobierna el comportamiento de un comando o un proceso. En el comando `freqout`, los parámetros del comando especifican qué pin usar, cuánto durará el sonido, y de qué frecuencia será. La palabra parámetro se emplea a menudo en la ciencia y en la ingeniería. Por ejemplo: la temperatura del aire es un parámetro que determina la velocidad de secado de una pintura. O, la temperatura media global de la atmósfera terrestre es un parámetro que determina cuánta agua se encuentra en el océano en relación a la que se encuentra en las capas de hielo, determinando el área costera que queda expuesta o sumergida.

Llegó el momento de experimentar. Modifique el programa reemplazando 1900 por 3800 para obtener un tono más agudo:

```
freqout 0,1000,3800
```

Descargue el programa en el BASIC Stamp. Observe que escucha un tono más alto. Pruebe un par de veces, con valores mayores y menores. No tema dañar el BASIC Stamp por reprogramarlo muchas veces. Puede reprogramar el BASIC Stamp más de un millón de veces. Pruebe variar la duración:

```
freqout 0,2000,3800
```

Y, para hacer que el comando `freqout` toque dos tonos a la vez:

```
freqout 0,2000,1900,2533
```

El número 2533 es igual a $\frac{4}{3}$ de 1900, la cuarta musical. El BASIC Stamp puede tocar dos tonos a la vez, pero no más de dos.

Además, intente lo siguiente:

```
freqout 0,2000,1900,1903
```

¿Cómo explica lo que oye?

Intente una duración tan pequeña como 2ms:

```
freqout 0,2,1900,3804
```

Siéntase libre de experimentar. Experimentando con los comandos del BASIC Stamp, puede descubrir cualidades que le serán útiles en programas futuros.

Experimento 1: Transductores de Temperatura y de Sonido

Código Morse

Un indicador sonoro es un dispositivo que indica acústicamente lo que está sucediendo en un sistema. Disponer de un indicador sonoro en la Plaqueta de Educación será de gran utilidad en el resto de los experimentos de Mediciones Ambientales. En el Experimento 2, haremos un programa para enviar números utilizando el código Morse, y usaremos el código para indicar las lecturas de temperatura. El código Morse es una forma excelente de transmisión de mensajes utilizando sonido. La tabla muestra los números del 0 al 9 en código Morse:

Número:	Código Morse	Binario
0	dah dah dah dah dah	11111
1	dit dah dah dah dah	01111
2	dit dit dah dah dah	00111
3	dit dit dit dah dah	00011
4	dit dit dit dit dah	00001
5	dit dit dit dit dit	00000
6	dah dit dit dit dit	10000
7	dah dah dit dit dit	11000
8	dah dah dah dit dit	11000
9	dah dah dah dah dit	11110

El código Morse se basa en enviar patrones de sonidos cortos y largos. El sonido largo tiene una duración de tres veces el sonido corto. Llamamos "dit" al sonido corto y "dah" al sonido largo. Todos los números constan de cinco dits y dahs. Las letras del alfabeto tienen de uno a cuatro sonidos, y las letras más comunes tienen los patrones más cortos (por ejemplo, e dit, t dah, s dit dit dit, q dah dah dit dah). Los signos de puntuación tienen seis sonidos, por ejemplo punto dit dit dah dah dit dit. Entre los sonidos de una letra o un número, debe haber una pausa igual a dit. Y la separación entre diferentes letras o dígitos, como en "50", debe tener una duración igual a dah. La columna "Binario" está para mostrar al código Morse como un número binario.

En estos experimentos, usaremos solamente los números. Pruebe este programa simple que envía el número de dos dígitos "50" en código Morse. No es necesario que copie los comentarios. Recuerde que se llama comentario al texto que se encuentra a continuación del apóstrofe '.

```
dit          con 70      ' definición de un período corto de tiempo
dah          con 3*dit   ' definición de un tiempo mayor, 3 veces el anterior
i            var nib     ' variable para contar
for i=1 to 5          ' repite 5 veces las dos líneas siguientes
  freqout 0,dit,1900   ' emite un dit
  pause dit           ' silencio corto
next
pause dah            ' silencio largo entre dígitos
```

Experimento 1: Transductores de Temperatura y de Sonido

```
for i=1 to 5          ' repite 5 veces las dos líneas siguientes
  freqout 0,dah,1900  ' emite un dah
  pause dit           ' silencio corto
next
```

Ejecute el programa. Presione el botón Reset de la Plaqueta de Educación si desea escuchar nuevamente el número 50. ¿Cómo modificaría el programa anterior para enviar el más famoso código Morse, SOS?

A esta altura ya debería estar familiarizado con el bucle `for . . . next` visto en el libro ¿Qué es un Microcontrolador? . Piense la forma en la que el programa incorpora las reglas del código Morse. Observe como comienza por definir la constante `dit` en milisegundos, y a continuación `dah` se define como tres veces `dit`. PBASIC le permite hacer esto, definir una constante matemáticamente en función de otra. Esto es muy conveniente, debido a que le permite modificar el comportamiento de todo el programa, modificando solamente el valor de la constante `dit`, y `dah` seguirá estos cambios.

En el programa, cambie la constante `dit` a un valor distinto de 70, por ejemplo el doble o la mitad, y escuche el efecto sobre la velocidad de todo el programa. El único aspecto importante es que la relación de duración entre `dit` y `dah` esté siempre en 1:3.

Esto es solamente una introducción. Escribiremos un programa de código Morse más serio en el experimento dos, para indicar las lecturas de temperaturas.

Lectura de Temperaturas con el DS1620

Ahora cambiemos completamente el tema. Centrémonos en el tema principal, adquirir algunas lecturas de temperatura. En ingeniería, normalmente usamos la palabra *adquirir* en lugar de *obtener*, cuando nos referimos a datos o lecturas. La Plaqueta de Educación se convertirá en nuestro sistema de adquisición de datos.

El DS1620 es un moderno transductor de temperatura (parte de las hojas de datos del DS1620, en inglés, se incluyen en el Apéndice D). Nos encontramos nuevamente con la palabra **transductor**. En este caso, se refiere a un dispositivo que transforma la temperatura en una señal eléctrica. El DS1620 toma la temperatura como una entrada, y transforma ese valor en un código digital que el BASIC Stamp puede interpretar. El código digital representa la temperatura del CI DS1620.

El DS1620 viene en un encapsulado plástico de 8-pines. Póngalo en la Plaqueta de Educación y conéctelo como se muestra en la imagen de la Figura 1.3. Aclaración importante Cuando modifique el conexionado de la plaqueta de Educación, es una buena idea desconectar la batería o la fuente de alimentación. Es muy fácil que mientras realiza un cambio, un cable toque en un lugar incorrecto y queme algún componente. Revise dos veces el cableado, o mejor aún, haga que alguien lo revise antes de conectar la alimentación. El esquema del DS1620 se muestra en la Figura 1.4.

Experimento 1: Transductores de Temperatura y de Sonido

Figura 1.3: Esquema del DS1620

Ubique el DS1620 en el extremo de la Plaqueta de Educación. Observe que hay una marca en una punta del encapsulado del DS1620 que indica la distribución de pines. Asegúrese de no invertir la polaridad de la alimentación.

- capacitor de 0.1 uF de Vdd a Vss
- pin 4 del DS1620 conectado a Vss.
- pin 8 del DS1620 conectado a Vdd
- pin 1 del DS1620 conectado a través de un resistor de 1K ohm a P15 del BASIC Stamp
- pin 2 del DS1620 conectado a P14 del BASIC Stamp

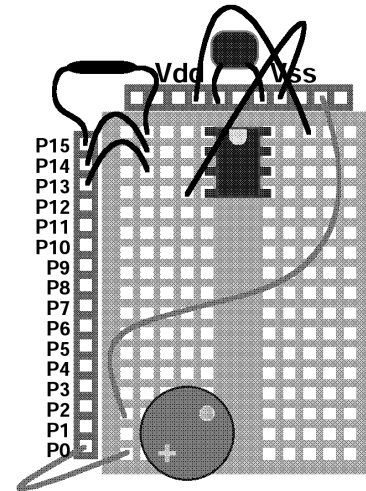
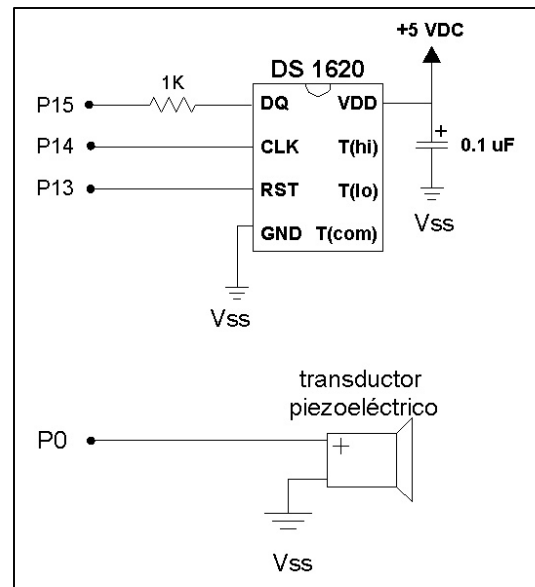


Figura 1.4: Circuito del DS1620

Circuito eléctrico del esquema anterior. Recuerde que la parte del transductor piezoeléctrico había sido montada con anterioridad.



Experimento 1: Transductores de Temperatura y de Sonido

Inserte el DS1620 en el extremo de la Plaqueta de Educación. Observe que hay una **indicador** en un extremo del encapsulado del DS1620, que indica cual es el pin 1. ¡Asegúrese de no invertir las conexiones de la fuente de alimentación!

Ahora llegó el momento de programar el DS1620, literalmente. El DS1620 es una pequeña computadora. Más precisamente, es un sensor inteligente. Puede memorizar ciertas configuraciones y realizar tareas bastante interesantes actuando solo. Los sensores inteligentes se están usando más y más en electrónica y en el campo del control y monitoreo ambiental.

Ingresa el siguiente programa. Nuevamente, no es necesario que escriba los comentarios.

```
low 13           ' Pone al DS1620 en espera
freqout 0,1000,3800 ' sonido indicador de ejecución del programa
high 13          ' Le avisa al DS1620 que viene una orden
shiftout 15,14,lsbfirst,[12,2] ' Comando para fijar al DS1620 en configuración 2
low 13           ' Completa el ciclo del comando
end              ' fin del programa
```

Revise la sintaxis del programa. Descargue el programa.



Escuchará un tono de 1 segundo. ¡Listo!. Parece simple pero muchas cosas han pasado. El comando `shiftout` envía dos bytes, 12 y 2, al DS1620. El 12 es una orden para que el DS1620 reciba una nueva configuración, y el 2 es la nueva configuración. Estas son las cuatro posibles configuraciones:

- 0: Sin CPU, conversión continua
- 1: Sin CPU, una sola conversión
- 2: Con CPU, conversión continua
- 3: Con CPU, una sola conversión

¿Qué? Seleccionando la configuración 2, le decimos al DS1620 que queremos que envíe sus lecturas a una CPU (Central Processing Unit- Unidad de Procesamiento Central-el BASIC Stamp). La otra opción es que controle la temperatura por su cuenta, sin comunicar ninguna lectura. ¿Para qué sirven estas opciones? Dijimos que el DS1620 es un sensor inteligente. Los pines que no usamos del DS1620 podrían ser conectados a un calentador o a un ventilador, y emplearse para regular la temperatura de una habitación o un invernadero. El DS1620 tiene también un comando que le permite fijar una temperatura. Profundizaremos sobre regulación de temperatura en el Experimento 6.

Experimento 1: Transductores de Temperatura y de Sonido

Seleccionando la opción CPU, el DS1620 enviará datos a través de la línea serial cuando reciba los comandos.

Conversión continua significa que leerá la temperatura una y otra vez, y siempre tendrá disponible el valor actual. Una sola conversión (opción que no usamos) significa que leerá la temperatura una vez y esperará a recibir una nueva instrucción. Este modo es usado cuando un ingeniero necesita minimizar el consumo de energía.

Una vez que le hemos enviado la configuración, el DS1620 no la olvidará. Es almacenada en la memoria del DS1620 (EEPROM, como la memoria de programa del BASIC Stamp) y no se borra al interrumpir la alimentación.

El corazón del programa PBASIC es el comando `shiftout`. La secuencia es un ejemplo de comunicación serial sincrónica. Puede costarle un poco entender como funciona esta comunicación. La mayoría de los componentes electrónicos modernos que se encuentran desde en agendas hasta satélites, usan este sistema. La razón principal de esta popularidad es que los componentes que usan comunicación serial pueden reducir su tamaño, debido a que no necesitan muchos pines de conexión. Estos son los parámetros del comando.

```

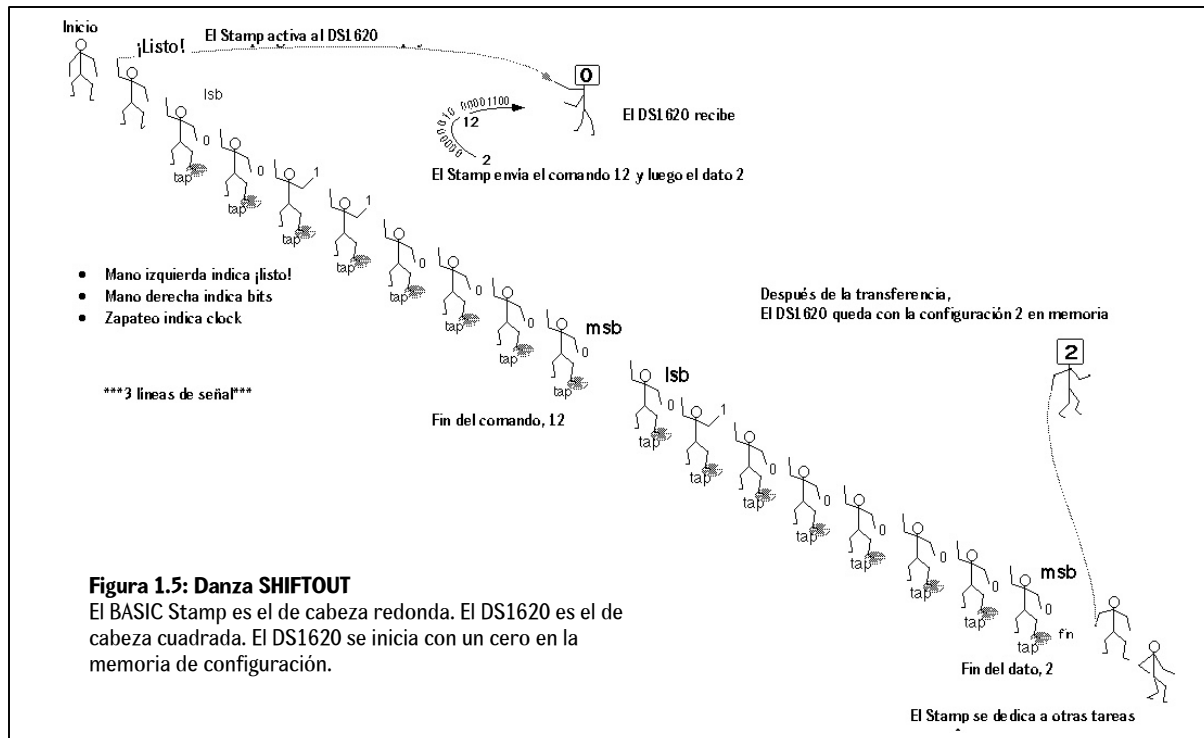
high 13          ---   inicia la sesión
shiftout 15,14,lsbfirst,[12,2]
                ^^^^--- dos bytes que se envían del Stamp al DS1620
                ^^^^^^^----- se envía el bit menos significativo primero
                ^^----- P14 del BASIC Stamp es el clock (reloj)
                ^^----- P15 del BASIC Stamp es usado para enviar los datos
low 13          <---   finaliza la sesión

```

Para explicar como funciona, intentaremos con una analogía con la danza de la figura. Por favor mire la Figura 1.5. El BASIC Stamp es el de cabeza redonda. El DS1620 es el de cabeza cuadrada. El DS1620 inicia con un cero en la memoria de configuración.

El BASIC Stamp inicia la danza SHIFTOUT levantando la mano izquierda. Esta señal activa el DS1620, y lo prepara para recibir el mensaje. Luego el BASIC Stamp golpea 8 veces en el pin del clock, que es el pie. Con cada golpe, el BASIC Stamp levanta la mano derecha para enviar un uno, o la baja para enviar un cero. Estos son los dígitos del número binario que se está enviando, comenzando por el menos significativo (lsb), por el pin de datos.

El DS1620 mira la mano derecha del BASIC Stamp en cada golpe. Después de ocho golpes, el DS1620 tiene el número binario 12 y lo reconoce como una instrucción. El BASIC Stamp sabe de antemano que el DS1620 interpretará al número 12 como una instrucción. (Las instrucciones son determinadas por los ingenieros de Dallas Semiconductor, el fabricante de este componente).



No hemos terminado aún. El DS1620 espera otro número binario a continuación del 12. El BASIC Stamp golpea 8 veces más, y el DS1620 mira la mano derecha del BASIC Stamp en cada golpe. Esta vez obtiene el número 2. El DS1620 almacena el 2 en su memoria EEPROM. Ahora el DS1620 está configurado. El BASIC Stamp baja la mano izquierda para indicar que terminó la secuencia. A partir de este momento el BASIC Stamp y el DS1620 no están comunicados. Toda esta comunicación se realiza automáticamente en menos de una milésima de segundo (1/1000), con el comando `shiftout`. La Figura 1.6 muestra esto mismo pero como lo representaría un ingeniero.

Figura 1.6: Diagrama de Tiempos

Diagrama de tiempos de la ejecución del comando SHIFTOUT desde el BASIC Stamp (en volts).

[illegible]

Experimento 1: Transductores de Temperatura y de Sonido

Observe que 12 decimal 00001100 binario, y 2 decimal es 00000010 binario.

Al revisar el diagrama de tiempos de la Figura 1.6 considere lo siguiente:

- P13 inicia el intercambio yendo de 0 a 5 volts. Esta orden finaliza cuando P13 vuelve de 5 a 0 volts. P13 se denomina chip select o chip enable (selector de chip o habilitación de chip).
- P14 es el clock (reloj) y emite una serie de 16 pulsos de 0 a 5 volts, en dos grupos de 8.
- P15 es la línea de datos y fija 0 ó 5 volts, en forma sincronizada con los pulsos de reloj de P14. El primer grupo forma el 12 (00001100 en binario), y el segundo grupo forma 2 (00000010 en binario).
- Observe el parámetro "lsbfirst" del comando `shiftout`. El bit menos significativo, el de menor peso, se emite primero en la secuencia.
- La transmisión completa de los 16 ciclos de reloj dura aproximadamente 1 milisegundo, 1/1000 de un segundo, y todo se realiza automáticamente mediante el comando `shiftout`.

Si desea una explicación más detallada, búsquela en el BASIC Stamp Manual Version 1.9 (en inglés), donde se describe el funcionamiento del comando `shiftout`, y además, hay una nota de aplicación.

Este sistema se llama comunicación serial sincronizada, debido a que los datos están sincronizados con los pulsos de reloj que salen del BASIC Stamp. El BASIC Stamp es comúnmente llamado maestro y el DS1620 esclavo. Esto es debido a que los pulsos de reloj son controlados por el BASIC Stamp.

Ahora el evento principal: leer la temperatura del DS1620. Ingrese el siguiente programa.

```
x      var byte      ' define una variable de propósito general, byte
C      var byte      ' define una variable para los grados Celsius
                        ' nota: el DS1620 está preprogramado en modo 2.

outs=%0000000000000000      ' define el estado inicial de todos los pines
      'fedcba9876543210
dirs=%1111111111111101      ' como salidas en estado bajo

freqout 0,20,3800      ' sonido indicador de inicio de ejecución
high 13      ' habilita el DS1620
  shiftout 15,14,lsbfirst,[238]      ' envía el comando "iniciar conversiones"
low 13      ' fin de la orden
  bucle:      ' subrutina que lee y muestra la temperatura
    high 13      ' habilita el DS1620
    shiftout 15,14,lsbfirst,[170]      ' envía el comando "obtener datos"
    shiftin 15,14,lsbpre,[x]      ' obtiene los datos
    low 13      ' deshabilita el DS1620
    C=x/2      ' convierte los datos en grados C
    debug ? C      ' muestra los datos en la pantalla de la PC
```

Experimento 1: Transductores de Temperatura y de Sonido

```

    pause 1000          ' pausa de un segundo
goto bucle             ' regresa a la etiqueta bucle

```

Descargue el programa en el BASIC Stamp.

La pantalla debug debería aparecer, mostrando las lecturas de temperatura actual, una por segundo. Las lecturas están en grados Celsius. Si mantiene su dedo apoyado sobre el DS1620, debería ver un aumento en la temperatura.

En caso de errores en su programa:

Si obtiene un mensaje de error en su programa, puede ser por una palabra mal escrita. El programa editor del Stamp posicionaré el cursor cercano a la ubicación del error. No tome literalmente el mensaje de error. Algunas veces, el mensaje de error no se relaciona con el error real. Busque errores cerca del cursor. Si el error que aparece es "hardware not found" o "communication error", asegúrese que la Plaqueta de Educación esté encendida (luz verde en la Plaqueta de Educación) y que el cable de la PC esté bien conectado. Si todo está bien pero el programa no funciona, entonces usted debe decidir si el problema está en el programa o en el conexionado del DS1620.

Ahora, ¿cuál es la temperatura de la habitación donde se encuentra?

Observe que cuando pone su dedo sobre el DS1620, o se expone al sol o a otra fuente de calor, es necesario un cierto tiempo para que se caliente, como así también para que se enfrie. Una vez que lo calienta, observe que puede enfriarlo más rápidamente soplándolo. ¿Cuál es la temperatura cerca del piso? ¿Y sobre la PC? ¿Cerca de su cuerpo? ¿Son diferentes? Intente tomar esas mediciones.

¿Cuál de todas las temperaturas (si son diferentes) será la llamada temperatura ambiente? Normalmente, ingenieros HVAC (Heating, Ventilation and Air Conditioning, Acondicionamiento de aire, ventilación y calefacción) prefieren las temperaturas tomadas a la sombra, lejos de fuentes de calor, como cuerpos y computadoras. Se llama temperatura representativa. En el mundo real, puede haber grandes variaciones en pequeñas distancias y cortos períodos de tiempo. Usted siempre debe elegir donde y cuando realizar una medición.

¿Cómo funciona el programa? Primero observemos las instrucciones outs y dirs:

```

outs=%000000000000000000    ' define el estado inicial de todos los pines
    ' fedcba9876543210
dirs=%1111111111111111      ' como salidas en estado bajo

```

Experimento 1: Transductores de Temperatura y de Sonido

Cuando usa el BASIC Stamp, o cualquier microcontrolador, habrá pines conectados al mundo exterior, y esos pines pueden estar configurados como entradas o salidas, y si son salidas se pueden fijar en estado alto o bajo. Usted ya está familiarizado con las variables `out` y `dir` del libro *¿Qué es un Microcontrolador?*. En este caso, con una "s" al final, la instrucción controla los 16 pines de E/S (entrada salida o I/O), numerados de 0 a f (observe el apóstrofe delante de la "f" esa línea es un comentario, y está para referencia.) Los pines de E/S del BASIC Stamp están numerados de P0 a P15 (donde a 10, b 11, c 12,...,f 15, en hexadecimal).

Es una buena costumbre en la programación, comenzar todo programa importante, fijando todos los pines del microcontrolador, en un estado conveniente. Cuando el BASIC Stamp es encendido o reiniciado (reset), todos los pines se configuran como entradas, por defecto. Este es un estado bastante conveniente para inicializar un microcontrolador. Usted, el diseñador, es el encargado de convertir en salidas los pines necesarios. Por otro lado, si un pin está desconectado del resto del circuito, no es una buena idea dejarlo como entrada. Entradas desconectadas pueden causar que el microcontrolador se comporte erráticamente o que consuma mucha energía de la batería. Las instrucciones anteriores convierten a todos los pines del BASIC Stamp en salidas en estado bajo. Esto es conveniente para el transductor piezoeléctrico y para el DS1620. Los otros pines se hacen salidas en estado bajo, por una cuestión de uniformidad. Esta condición irá variando al avanzar con los experimentos. Para más información sobre los comandos `dirs` y `outs`, lea el BASIC Stamp Manual Version 1.9, página 216 (en inglés).

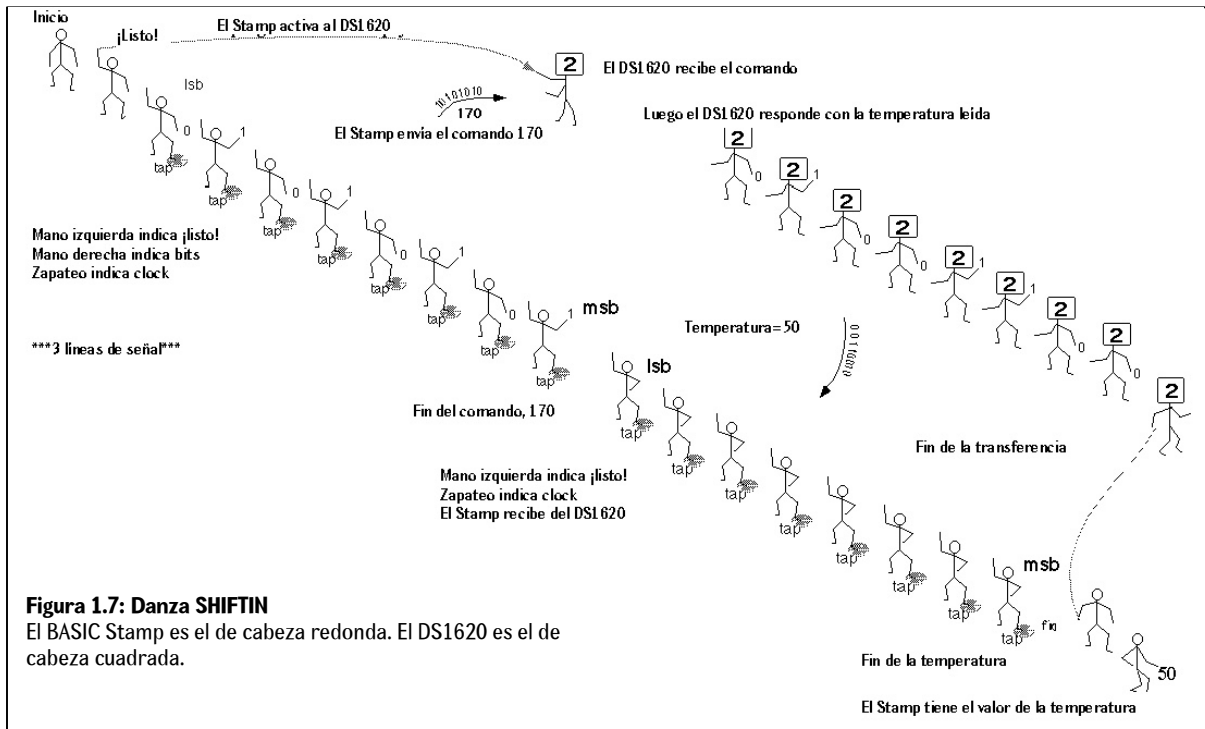
La acción más importante del programa de temperatura viene de los comandos `SHIFTOUT` y `SHIFTIN`.

El primer `shiftout` debería parecerle familiar. La secuencia familiar es: pone un alto en P13, luego envía un byte, 238, al DS1620, y luego baja P13 para dar por terminada la secuencia. Dentro del DS1620, el 238 es un comando que le ordena comenzar a convertir la temperatura en código digital. El comando 238 debe ser enviado al menos una vez, luego de encender el DS1620. A diferencia del comando de configuración, éste no es almacenado en la memoria permanente del chip.

A continuación viene el corazón de la rutina, que lee la temperatura del DS1620. Nuevamente debe reconocer la secuencia: pone un alto en P13, luego envía un byte, 170, al DS1620. Nada más y nada menos. El DS1620 interpreta el 170 como un comando que le ordena enviar las lecturas de temperatura actuales al BASIC Stamp. Ahora las cosas se ponen interesantes. El DS1620, en respuesta al comando 170, toma el control de la línea de datos. El BASIC Stamp ejecuta entonces el comando `shiftin`. Estos son los parámetros:

```
shiftin 15,14,lsbpre,[x]
           ^----- variable que recibe los datos
           ^^^^^^----- se recibe el bit menos significativo primero
           ^^----- P14 del BASIC Stamp es el clock (reloj)
           ^^----- P15 del BASIC Stamp es la línea de datos
low 13      <-- fin del comando
```

Experimento 1: Transductores de Temperatura y de Sonido



P15 del BASIC Stamp es ahora una entrada, mientras que para *shifout* fue una salida. El BASIC Stamp está listo para recibir datos del DS1620. Esto está representado en la Figura 1.7, y como diagrama de tiempos en la Figura 1.8.

Observe que el BASIC Stamp se sigue encargando del clock. El BASIC Stamp es aún el maestro y el DS1620 el esclavo. Este es el diagrama de tiempos:

Figura 1.8: Diagrama de Tiempos de SHIFTIN

[illegible]

Cada vez que el BASIC Stamp manda un pulso en la línea de reloj P14 (golpea su zapato), el DS1620 envía el siguiente bit del byte de la temperatura. Comienza por el bit menos significativo. El parámetro `lsbpre` significa que el BASIC Stamp busca el bit menos significativo antes de enviar el primer pulso de reloj. Funciona así, obtiene el primer bit, pulso de reloj, obtiene el segundo bit, pulso de reloj, y continúa hasta obtener los 8 bits. El BASIC Stamp almacena los datos recibidos del DS1620 en la variable `x`.

Si la temperatura es de 25 grados Celsius, el DS1620 envía el valor 50, que es dos veces la temperatura. En binario, 50 es 00110010. El byte que el DS1620 envía, siempre es el doble de la temperatura. Si la temperatura es de 25.5 grados C, entonces el byte que el DS1620 envía será 51. Cada aumento de x representa medio grado C. Esta es la resolución, el cambio más pequeño de temperatura que detecta el sensor.

El programa luego convierte el valor de x en temperatura:

$C=x/2$ ' convierte los datos en grados C

El BASIC Stamp usa aritmética entera. Elimina el 0,5 del resultado del cociente. Tanto 50/2 y 51/2 producirán C 25, y 52 y 53 darán C 26. Hay formas de mantener la resolución de medio grado, pero no la trataremos ahora. (¡Usted lo hará en el desafío!)

La temperatura se envía a la pantalla de debug por este comando:

```
debug ? C          ' muestra el resultado
```

El "?" hace que el BASIC Stamp muestre "c=" y luego el valor actual de la variable c en la pantalla de debug, en un renglón distinto cada vez.

Experimento 1: Transductores de Temperatura y de Sonido

¿Qué es Un

Límite operacional:

El DS1620 es perfectamente capaz de medir temperaturas bajo cero (hasta -25). Esto será importante si va a investigar la nieve en Alaska, o si está diseñando un sistema de control para un freezer. El problema es que el programa que hemos escrito no maneja temperaturas negativas correctamente. Por ejemplo, cuando la temperatura sea -1 grado C, nuestra lectura será C 127 en lugar de C -1. Para leer temperaturas negativas, se deben realizar un par de pasos, que complicarían el programa más de lo necesario para este caso. Como está planteado el programa, el límite operacional inferior es de cero grados. Los límites operacionales se encuentran muy frecuentemente en la ingeniería, y aparecen por muy variadas razones, tanto por software o hardware o las propiedades de los materiales. Este límite en particular se origina en la escritura simplificada del software. Esto se justifica siempre y cuando la temperatura se mantenga por encima del punto de congelamiento, pero será un error ("bug") si intentamos bajar más la temperatura. Un límite operacional famoso de software es (¡fue!) el problema del Y2K (año 2000), donde una simplificación del software de la segunda mitad del siglo XX, produjo problemas a partir del año 2000.

Es conveniente que guarde el programa que ha escrito y ejecutado. En esta serie de experimentos, armaremos un gran programa, una pieza a la vez. Esta es la primer pieza que reutilizará.

Si aún no lo hizo, es conveniente que agregue los comentarios al programa. Esto le facilitará la comprensión del funcionamiento del programa, y lo hará más simple de modificar, como lo probaremos en el Experimento 2.

Elija un nombre para el programa. Su profesor dará instrucciones, dependiendo de como el curso comparta las PC. El programa tendrá una extensión "BS2". Digamos que decide llamar al programa "DS1620.BS2"

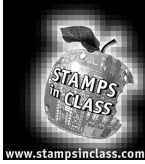
Esta es la forma de grabar el programa en las versiones de DOS y Windows del editor del BASIC Stamp de Parallax:

STAMP2.EXE (DOS):

Ingrese ALT-S, teniendo presionada la tecla ALT mientras oprime "S". Un recuadro de diálogo aparecerá, para permitirle ingresar o modificar el nombre del programa. Escriba el nombre, presione ENTER. ¡Listo!.

STAMPW.EXE (Windows):

Vaya a File/Save (Archivo/Guardar), luego navegue por el directorio donde desea grabar el programa, escriba el nombre, y presione enter o haga click en Save (Guardar).



¡Desafío!

1. Escriba un programa que use una secuencia de comandos `freqout` para tocar una melodía simple. Busque información del comando `freqout` en el BASIC Stamp Manual Version 1.9 (en inglés). Encontrará un ejemplo de como tocar Mary Had a Little Lamb (Mary tenía un corderito). Bueno, puede intentar con Stairway to Heaven, o la Quinta de Beethoven, si lo prefiere. Con esto descubrirá algunas de las limitaciones de alta fidelidad del transductor piezoeléctrico.
2. Defina una variable F para grados Fahrenheit. Muestre grados Celsius y Fahrenheit en la pantalla debug. Use cualquier fórmula:

$$F = C * 9 / 5 + 32$$

o

$$F = x * 9 / 10 + 32$$

¿Una fórmula es mejor que la otra? ¿Por qué? Observe como cambian las lecturas a medida que modifica la temperatura del DS1620.

3. Muestre grados Celsius con una resolución de 0,5 grados. Recuerde que el valor que se obtiene del DS1620 es un número binario, donde cada bit representa 0,5 grados. Para obtener la lectura, habíamos dividido por 2 y perdido una parte de la información. Usted puede mostrar el resultado como 205 para representar 20,5 grados C. Pista: multiplique por 5 en lugar de dividir por 2.
4. Si la temperatura es mayor de (elija un valor), toque un tono de alarma en el piezoeléctrico. Apague la alarma cuando la temperatura baje. Luego modifique el programa para que la alarma continúe sonando, aunque la temperatura baje. ¿Bajo que circunstancias será más conveniente cada alarma?



Experimento 2: Adquisición de Datos

El tema de la adquisición de datos se entiende más fácilmente al responder esta pregunta: ¿Qué es adquisición de datos y por qué es tan importante en las mediciones ambientales? Las actividades de este experimento son: (1) Diseñar una interfase de usuario agregando un pulsador al circuito existente en la Plaqueta de

Educación, que realice distintas acciones si se presiona una vez, dos veces seguidas (doble click), o una vez durante más tiempo (2) Aprender las bases de los comandos `read` y `write` con la EEPROM del BASIC Stamp y (3) Realizar un "termómetro audible (código Morse)".

La imagen que caracteriza al mundo natural, es la de cambio constante. Para comprender y predecir eventos, necesitamos llevar un registro de las variables involucradas. En el campo de las mediciones ambientales, un sistema de adquisición de datos (Data loggers o DAQ), es una herramienta esencial. Es un dispositivo que toma lecturas automáticamente y las almacena en su memoria, en periodos de tiempo constantes (o con alguna base conocida), para ser recuperados posteriormente.

Los datos se almacenan en un archivo (log). El termino inglés data logger viene de la historia naval, donde las lecturas de posición y profundidad se almacenaban en la bitácora (log-book). Por ejemplo, la velocidad se calculaba tomando el tiempo que tardaba un objeto (log) arrojado al agua por la parte delantera de la embarcación (proa), en alcanzar la parte trasera (popa).

En estos días, la adquisición de datos se realiza con sensores conectados a computadoras. Las computadoras son muy convenientes para la adquisición de datos, nunca se aburren ni se cansan, y pueden trabajar eficiente y rápidamente. Puede ser difícil, aburrido, o directamente imposible que una persona se encuentre en el lugar y el momento en que las lecturas deben ser tomadas. Los data loggers pueden ser encontrados en boyas flotando en el océano, en picos ventosos de montañas, en el espacio, en collares de osos grizzly, en estómagos de ballenas, en huertas y viñedos, y en innumerables procesos industriales.

Otra palabra relacionada que se usa en la actualidad es SCADA, por Small Computer Aided Data Acquisition (Adquisición de datos con ayuda de pequeñas computadoras). Esto se refiere a una red de sensores y computadoras, pero la idea general es la misma. Los Data loggers pueden incluso comunicarse a un concentrador (hub) central y mediante conexiones TCP/IP a Internet, o también a radios de larga distancia.

En este experimento aprenderá detalles importantes de la memoria EEPROM del BASIC Stamp II. Esto es una preparación para almacenar lecturas de temperatura, luz y nivel de agua en los próximos experimentos. Además, mejorará el termómetro DS1620 del experimento anterior, haciéndolo hablar (en código Morse). Y como precalentamiento, trabajará con un pulsador y el piezoeléctrico, para hacer una interfaz de usuario.

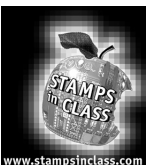
Todos los que manejaron una computadora saben lo que es un mouse (ratón), y las acciones de click, doble-click, y click-and-hold (mantener presionado). Estas acciones son muy usadas en cualquier interfaz de usuario moderna. ¿Alguna vez se preguntó cómo un programa implementa esas acciones? ¿Qué tan difícil será implementarlas en un BASIC Stamp? Bien, no es tan difícil, y vamos a hacerlo logrando que un pulsador en la Plaqueta de Educación realice múltiples tareas. En muchas aplicaciones, como en los próximos experimentos, no tendremos lugar disponible para muchos pulsadores. Un pulsador, con la ayuda del piezoeléctrico, será nuestra interfaz de usuario, cuando la Plaqueta de Educación no esté conectada a la PC.



Partes Requeridas

Los experimentos de Mediciones Ambientales son progresivos y se construyen usando como base los proyectos previos. Por lo tanto, deberá agregar componentes a la Plaqueta de Educación. Este experimento requiere los siguientes componentes:

- pulsador
- resistor de 10K ohm
- cables de interconexión



¡Constrúyalo!

En el Experimento 2 de ¿Qué es un Microcontrolador?, "Detectando el Mundo Exterior", aprendió a usar dos botones para tomar decisiones, y controlar dos LEDs. En este experimento nos basamos en esos conceptos y en los del experimento anterior de Mediciones Ambientales. Usted ya cuenta con un dispositivo de salida audible. Ahora, instale un pulsador como entrada según la Figura 2.1. El circuito eléctrico se muestra en la Figura 2.2.

Figura 2.1: Gráfico

Instale un pulsador(PB) en el extremo de la Plaqueta de Educación, del lado del piezoeléctrico. Dos de los pines quedarán por fuera de la protoboard, logrando así lugar para realizar un par de conexiones. Si acomoda correctamente los pines del pulsador, debe entrar en un cuadrado de 3x3 huecos de conexión, como se muestra en el gráfico. Sujete los terminales externos del pulsador con cinta adhesiva.

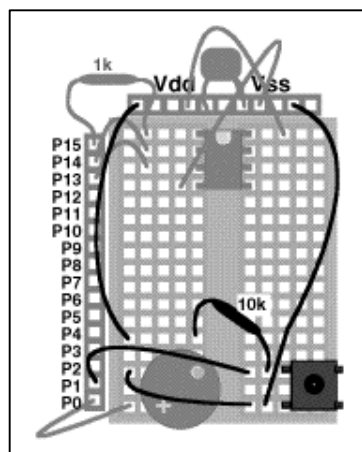
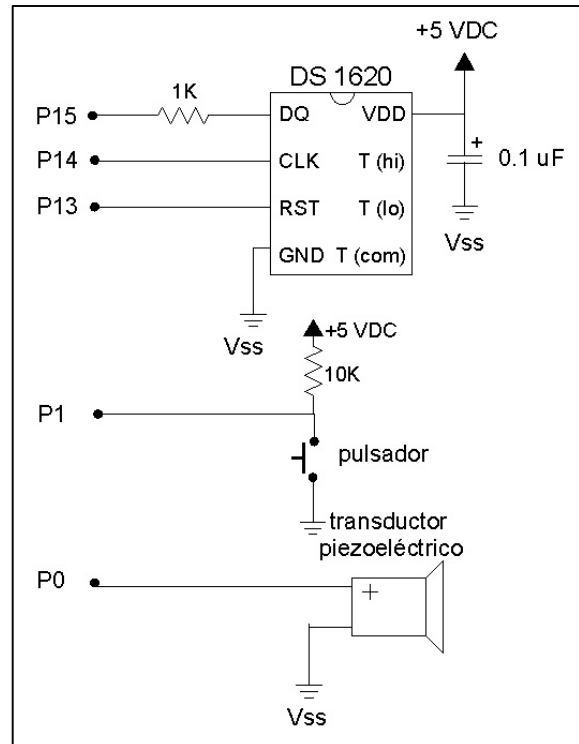


Figura 2.2: Esquema eléctrico

Realice el conexionado como sigue.

- Vss al pulsador (quitándolo del piezoeléctrico -)
- Conectar del pulsador al piezoeléctrico (-)
- Vdd (5 volts) conectado al lado del piezoeléctrico
- 10k ohm de Vdd al pulsador ()
- pulsador () a P1.

Nota: Si endereza los terminales del pulsador, no será necesario salirse de la protoboard.



¡Prográmelo!

El circuito tiene un pulsador conectado a un resistor de pull-up (conectado a Vdd), y la conexión entre pulsador y resistor conectada a P1 en el BASIC Stamp. Cuando el pulsador no está presionado, la tensión en el pin del BASIC Stamp es 5 volts (Vdd), a través del resistor de pull-up. Pero cuando se presiona el pulsador, la tensión en el pin del BASIC Stamp es baja, cero volts (Vss). Pruebe el siguiente programa.

```
' Mediciones Ambientales programa 2.1
' prueba de pulsador
bucle:
debug bin in1
goto bucle
```

¿Qué es DEBUG?

En estos experimentos, la instrucción DEBUG (corregir, depurar) se verá muy a menudo, enviando datos a la pantalla de la computadora. Usted puede enviar información a la pantalla que le ayude a controlar lo que sucede en el programa, pudiendo así corregir o depurar errores. Además puede pedirle al comando DEBUG que envíe datos o un mensaje a la pantalla al igual que con el comando SEROUT. No necesariamente se usa para corregir errores.

El comando debug le permite observar los valores con diferentes formatos, usando modificadores y comandos de control de pantalla. En el experimento 1, usamos este comando para mostrar la temperatura:

debug ? C

Esta es una instrucción combinada que realiza 3 tareas: imprime el nombre de la variable y un signo igual imprime el valor decimal de la variable y mueve el cursor a la línea siguiente. El resultado será:

C=25

El programa que acabamos de hacer tiene una instrucción debug diferente:

debug bin in1

Esta imprime el valor binario de la variable in1. Si, in1 es una variable, el estado del pin P1 como entrada es, bajo o alto, 0 ó 1. Esta forma del comando debug muestra el "0" o el "1", y no el nombre "in1", ni el " ", ni ningún espacio entre unos y ceros, ni se mueve al renglón siguiente (hasta que complete el ancho de la pantalla). El resultado se verá así:

11111110000000000011111111111100000000

000000111111111111111111000000011111

...

A medida que utilicemos nuevas formas, las describiremos brevemente. Referirse al BASIC Stamp manual, v1.9 Pág. 253-256 (en inglés).

Ejecútelo y observe la ventana debug mientras presiona y suelta el pulsador. El programa se repite indefinidamente, mostrando el valor que encuentra en la entrada. El estado de la variable in1 es bajo 0 o alto 1. La lectura será cero inmediatamente después de presionar el pulsador, y valdrá uno al soltarlo. ¿Sí? Pase al próximo paso. ¿No? Hay un problema en el programa, la conexión del BASIC Stamp, o en el cableado del pulsador.

Ahora vamos a producir un sonido continuo mientras el pulsador esté presionado.

```
' Mediciones Ambientales programa 2.2
' zumbador/pulsador
klik:      ' etiqueta de inicio
  if in1=1 then klik ' decide si está presionado
    freqout 0,8,2500 ' tono mientras está
                  ' presionado
goto klik
```

Ejecute el programa. Al presionar el pulsador, debería oír un sonido parecido al de un grillo. ¿Qué está sucediendo? Sin presionar el botón, no sucede nada, debido a que la instrucción if ve un 1 en el pin de entrada y simplemente envía al programa nuevamente al inicio (klik). Si el pulsador está presionado, la instrucción if ve un cero en el pin de entrada. El programa continúa, más allá del if, y ejecuta la instrucción freqout. Luego regresa al inicio del programa (klik). Mientras el pulsador esté presionado, se repetirá el bucle ejecutando la instrucción freqout.

Observe que el parámetro 8 en el comando freqout es la duración del tono en milisegundos. La frecuencia del tono es de 2500 hertz, así que en 8 milisegundos, hay 20 ciclos del tono (0.008 segundos 2500 ciclos por segundo 20 ciclos). Luego el tono se detiene brevemente, mientras el programa regresa al inicio y verifica el estado del pin P1 nuevamente. Durante ese tiempo no se produce el tono, debido a que el BASIC Stamp sólo puede ejecutar un comando a la vez (¡Es importante recordar esto!). Si el pin continúa en estado bajo, se ejecuta nuevamente el comando freqout.

Entonces el sonido se emitirá así: Lo que escucha no es un tono puro de 2500 hertz, sino un tono repetido con pequeñas interrupciones. Esto agrega un subtono al sonido, a aproximadamente 110 hertz (los 9 milisegundos que demora el bucle, $1/0.009 \approx 111$). Esto lo hace más parecido al sonido del grillo, que es producido cuando el insecto frota las alas, emitiendo un sonido agudo, con pequeñas pausas mientras reacomoda las alas.

Como variación del programa anterior, cambie los valores del parámetro duración por: 1, 4, 50, 500 y 5000. Ejecute el programa con cada cambio y escuche las variaciones. En los intervalos largos, 500 y especialmente 5000, observe que el tono continúa después de haber soltado el pulsador. ¿Por qué ocurre esto? ¿Por qué no se detiene inmediatamente después de liberar el pulsador?

Regrese el valor de la duración a 8 y teclee con el pulsador, en código Morse, el número "50" o "SOS". Revea el experimento 1 si no recuerda el código. dit dit dit dit dit "5" y dah dah dah dah dah "0", dit dit dit "S", dah dah dah "O". Es un programa útil, ¡un teclado de código Morse!

Inserte la instrucción "pause 6" en la línea posterior al comando `freqout`. Se obtiene un patrón que se parece más aún a un grillo. Los grillos, además del "transductor de salida", las alas, poseen un "transductor de entrada", un oído. ¡Es una membrana ubicada en las patas delanteras! Los grillos son muy sensitivos a patrones repetitivos y sonidos pulsantes. Es su " código Morse". Los sonidos son parte de los cortejos y disputas de machos. Los entomólogos han estudiado el sonido de los insectos reproduciéndolos en altavoces, y observando qué parámetros del sonido afectaban el comportamiento de los grillos.

Algunas veces no se desea que una acción se repita todo el tiempo durante el cual el pulsador esté presionado. Por ejemplo, puede querer que se ejecute sólo una vez. Modifique el programa como se muestra a continuación. (Hay una nueva convención para simplificar las cosas las líneas modificadas están marcadas con ▲, y las nuevas con □. El resto de las líneas permanece sin cambios.)

```
' Mediciones Ambientales programa 2.3
' click en el pulsador, acción con botón presionado
klik:      ' regresa aquí cuando es liberado el pulsador
  if in1=1 then klik      ' decide si el pulsador está presionado(0) o no (1)
freqout 0,100,3800      ' ▲ toca el tono si el botón fue presionado
klik1:      ' □ regresa aquí si el botón permanece bajo
  if in1=0 then klik1      ' □ decide si el pulsador está presionado(0) o no (1)
goto klik
```

Como en el programa anterior, nada sucede hasta que se presiona el pulsador. Luego el tono suena durante 100 milisegundos. Luego hay un segundo bucle, donde el programa se detiene hasta que el pulsador sea liberado. Cuando esto ocurre, el programa regresa al inicio, esperando que se vuelva a presionar el pulsador. Cada vez que se presiona, se realiza una acción.

Está bien, pero piense como funciona el click del mouse. El click del mouse no realiza ninguna acción hasta que es liberado. Es fácil. Mueva `freqout` al bucle `clik1`:

```
' Mediciones Ambientales programa 2.4
' click en el pulsador, acción al liberar el botón
clik:                                ' Espera aquí mientras no se presiona el botón
  if in1=1 then clik                 ' decide si está alto(1) o bajo(0)
clik1:                              ' espera aquí hasta que se libere el botón
  if in1=0 then clik1               ' decide si está alto(1) o bajo(0)
  freqout 0,50,1900                 ' ▲ toca el sonido al liberar el botón
  freqout 0,100,3800                ' ▲ toca este sonido a continuación
goto clik                          ' regresa al inicio
```

Cuando se libera el botón, se debe escuchar un sonido que aumenta de tono. Lógico ¿no? Asegúrese de comprender totalmente como funciona este programa.

Ahora hagamos que el programa realice acciones distintas, una si presionamos una vez el botón, y otra si lo mantenemos presionado por un tiempo. Esto es similar a lo que pasa en algunos programas de computación donde un menú sólo aparece si mantenemos presionado el botón del mouse un tiempo. O lo puede haber visto en un auto estéreo, donde si presiona brevemente un botón selecciona una estación, pero si mantiene presionado el botón un tiempo (hasta que escuche un beep), se grabará en la memoria la estación que está escuchando en ese momento. Estos trucos son muy usados en instrumental científico así como en electrodomésticos.

El programa necesita una variable para registrar el tiempo que el botón permanece presionado. Pruebe esto: (Código nuevo marcado con □)

```
' Mediciones Ambientales programa 2.5
' pulsador, acción al presionar (click) y al mantener presionado (hold)
n var word                          ' □ variable para registrar el tiempo
clik:                                ' espera aquí mientras no se presiona el botón
  if in1=1 then clik                 ' decide si está alto(1) o bajo(0)
  n=0                               ' □ pone en cero el temporizador (timer)
clik1:                              ' espera aquí hasta que se libere el botón
  n=n+1                             ' □ incrementa el timer
  if n>500 then cliklargo            ' □ salta después de cierto tiempo
  if in1=0 then clik1               ' o se repite hasta que se libera el botón
  freqout 0,50,1900                  ' toca el sonido una vez al
  freqout 0,100,3800                 ' liberar el botón
goto clik                          ' vuelve al inicio

cliklargo:                          ' □ inicia aquí si se mantuvo presionado
```



```

    freqout 0,5,3800,2533      ' ☐ toca un sonido para identificar pulsado largo
    cliklargo1:                ' ☐ espera aquí hasta que se libere el botón
    if in1=0 then cliklargo1    ' ☐ decide si está alto(1) o bajo(0)
    goto clik                  ' ☐ vuelve al inicio

```

El programa va a `clik1` cuando presiona el botón. Mientras el botón está presionado, el programa repite el bucle `clik1`. La instrucción con `in1=0` hace que se regrese a `clik1` mientras el botón esté presionado. Cada vez que se ejecuta el bucle, la variable `n` aumenta una unidad. Es una competencia para ver que se cumple primero. ¿Se soltará el botón primero, o antes el temporizador llegará a 500? Si se suelta primero el botón, bien, el programa actúa igual que el anterior. El programa toca el tono y regresa al inicio esperando una nueva acción. Pero si el temporizador `n` llega a 500 antes de liberar el botón, el programa salta a la rutina `cliklargo`. Luego toca un sonido distinto para hacerle saber que ha llegado a esta parte del programa, y espera a que libere el botón. Luego regresa al inicio.

¿De dónde apareció el número mágico 500? La respuesta es simple: "prueba y error". El programador (¡Usted!) prueba diferentes números hasta que obtiene el resultado esperado. ¿Aproximadamente cuánto tiempo (en milisegundos) debe mantener presionado el botón hasta que salta a la rutina `cliklargo`? Experimente con valores distintos a 500.

Razone sobre el orden de estas dos instrucciones del programa 2.5:

```

    if n>500 then cliklargo      ' ☐ salta a la subrutina después de un tiempo
    if in1=0 then clik1          ' repite hasta que se libere el botón

```

¿Qué pasaría si invertimos el orden de estas instrucciones? Si no está seguro, inténtelo.

Tema Avanzado: Detectando un Doble-click con el BASIC Stamp

¿Puede el BASIC Stamp detectar un doble click? Seguro, no es muy difícil. Al final del primer click, el programa debe esperar una fracción de segundo para ver si vuelve a presionar el botón. Si lo hace, entonces es un doble click. Si no lo hace, es un click solo. El intervalo de tiempo es tan corto que usted ni siquiera lo notará. El intervalo es determinado por prueba y error, a "gusto del usuario".

Esto también necesita una variable de temporización (timer). Reciclaremos la misma variable de temporización *n*, de la última rutina. Pruebe esto: (las líneas con ☐ son nuevas). Sólo por diversión, también modificamos la rutina *cliklargo*, de forma que toca un sonido constante hasta que se libera el botón.

```
' Mediciones Ambientales programa 2.6
' pulsador, acción con doble click
n var word
clik:
  if in1=1 then clik
  n=0
clik1:
  n=n+1
  if n=500 then cliklargo
  if in1=0 then clik1
  n=0
clik2:
  n=n+1
  if in1=0 then dobleklik
  if n<150 then clik2
  freqout 0,50,1900
  freqout 0,100,3800
goto clik
end

dobleklik:
  if in1=0 then dobleklik
  freqout 0,50,3800
  freqout 0,50,2533
  freqout 0,50,1900
goto clik

cliklargo:
  freqout 0,5,3800,2533
  if in1=0 then cliklargo
goto clik
```

' variable para registrar el tiempo
' espera aquí mientras no se presiona el botón
' decide si está alto(1) o bajo(0)
' pone en cero el temporizador (timer)
' espera aquí hasta que se libere el botón
' incrementa el timer
' se bifurca si n llega a 500
' o se repite hasta que se libera el botón
' ☐ pone en cero el temporizador (timer)
' ☐ espera hasta que n llega a 150
' ☐ incrementa el timer
' ☐ se bifurca si el botón es presionado
' ☐ repite clik2 hasta que n llega a 150
' sonido para un click simple
'
' vuelve al inicio

' ☐ espera aquí hasta que
' ☐ se libere el botón
' ☐ toca un sonido descendente

' ☐ vuelve al inicio

' llega aquí si se mantuvo presionado el botón
' ☐ emite un sonido
' ☐ se repite hasta que se libera el botón
' vuelve al inicio

Si usted rápidamente presiona y libera una vez el pulsador, el programa llega a `clik2`. Ahora hay otra carrera entre el timer y el botón que comienza cuando se libera el pulsador. Si rápidamente presiona el pulsador por segunda vez, antes que el contador llegue a 150, se interpretará como un doble click. Pero si el timer llega primero a 150, el programa interpretará que se presionó el botón una sola vez (single click), o usted tiene dedos muy lentos y necesita modificar el valor del timer de 150 a 200, por ejemplo.

Las rutinas `clik1` y `clik2` son similares, pero observe que no son idénticas. ¿Qué pasaría si invertimos el orden de estas dos instrucciones en el programa? Si no le parece obvio, hágalo y razone el resultado.

```
if in1=0 then dobleclik      ' ☐ salta a subrutina si se presiona el botón
if n<150 then clik2          ' ☐ repite el bucle hasta que n sea 150
```

¿Qué es Una

Subrutina (Snippet):

Usted puede extraer una parte de un programa para utilizarla en otro, con o sin modificaciones. Las partes de programas que realizan tareas específicas se llaman subrutinas. Cada subrutina no es un programa independiente. Los programadores, a menudo, intercambian ideas en forma de subrutinas.

Si usted quiere, puede extender esta lógica para hacer una subrutina que responda a un triple click, como algunos procesadores de texto que lo usan para seleccionar un párrafo completo. ¡Lo haremos como desafío!

Ahora continuemos.

Guarde el programa que acaba de ejecutar en un disco. Lo usará para realizarle modificaciones en la sección de desafíos. Usaremos **subrutinas (snippets)** de estos programas, en los próximos ejercicios. Use el nombre "cliks.bs2", o el nombre sugerido por su instructor.

Aprendiendo lo Básico de READ y WRITE.

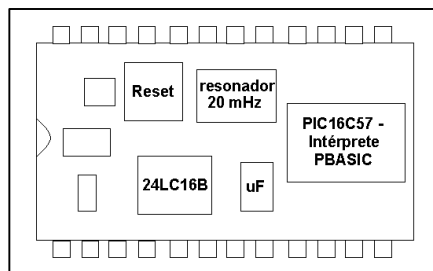
En esta serie de experimentos, vamos a programar el BASIC Stamp para almacenar lecturas de temperatura y otras variables. Vamos a almacenarlos a intervalos regulares de tiempo, en un archivo, para leerlos posteriormente. Haremos esto un paso a la vez. Primero, es importante comprender cómo esta organizada la memoria en el BASIC Stamp.

Sabe de "¿Qué es un Microcontrolador?" que el BASIC Stamp II tiene dos tipos de memorias, RAM y EEPROM.

Puede ayudarle a trabajar con estas memorias, saber dónde se ubican físicamente. Mire la Figura 2.3, que muestra una vista de arriba del BASIC Stamp II.

Figure 2.3: Memoria del BASIC Stamp

El PIC16C57 es la RAM y el procesador central del BASIC Stamp. El 24LC16B es la EEPROM, que almacena el programa PBASIC y los datos.



Las variables se crean en la RAM (Random Access Memory, Memoria de Acceso Aleatorio). Usted reserva lugar y almacena números en la RAM con estas instrucciones:

```
x      var      byte
x=76
```

Las variables son muy versátiles. Se las puede sumar, restar, y usar en cálculos aritméticos, y pueden ser parámetros en muchos de los comandos que se describen en el BASIC Stamp Manual Version 1.9 (en inglés). Los datos de la RAM se pueden manipular a gran velocidad (200 microsegundos por operación), y la RAM no se gasta con el uso. El problema es que no hay mucha RAM disponible en el BASIC Stamp, solamente 26 bytes. No es conveniente para almacenar muchos datos. Además, el contenido de la RAM se pierde cuando el BASIC Stamp se apaga, o cuando se presiona el botón reset (reiniciar). La RAM no es aconsejable para almacenar datos "invaluables", si se pretende que sobrevivan cuando se desconecta la alimentación.

También hay una EEPROM. Una gran cantidad de memoria EEPROM está disponible en el BASIC Stamp, 2048 bytes. Aunque parte de la EEPROM es usada por su programa PBASIC, quedará algo de espacio para los datos. Una gran ventaja de la EEPROM es que es casi permanente. La memoria EEPROM retiene su contenido sin alimentación y sobrevive al reset. Las desventajas de la EEPROM son: que es relativamente lenta (10 milisegundos para guardar un byte de datos), y que se gasta después de 1.000.000 de cambios en un punto. Por ejemplo, si cierto lugar de la EEPROM es reprogramado repetidamente, una vez por segundo, en 11 días estaremos cerca de 1.000.000 de ciclos de escritura. Por otro lado, a una vez por hora, se necesitarán 114 años para llegar al máximo de ciclos. Es algo a tener en cuenta en la planificación del programa. En Mediciones Ambientales escribiremos en un sector de la memoria unas cien veces, como máximo, cifra bastante lejana de 1.000.000.

La última desventaja de la EEPROM es que sólo dos instrucciones pueden manipular los datos almacenados en ella. `Read` lee un byte, y `write` almacena un byte. Eso es todo. No puede hacer cálculos directamente sobre los datos almacenados en la EEPROM, ni usarlos directamente como parámetros de un comando. Debe primero almacenarlos en una variable en la RAM, y luego manipularlos. Una vez obtenido el resultado, puede escribir (`write`) el valor de la variable en la EEPROM. Sabiendo esto, la principal razón por la que usamos la EEPROM es para almacenar grandes cantidades de datos, permanentemente (si no debemos cambiarlos a menudo).

En PBASIC, la instrucción `data` reserva un sector de la EEPROM, y le da un nombre:

```
log      data  7
          ----- el valor 7 es cargado en la EEPROM en la dirección "log"
          ----- nombre de la dirección en la EEPROM donde se ubica el dato.
```

`Read` lee un byte desde una dirección en la EEPROM, y copia este valor en una variable (en la RAM). El valor del byte en la EEPROM no es modificado por la lectura.

```
read     log, x
          ----- variable de RAM que recibe el dato
          ----- lugar de la EEPROM donde obtener el dato
```

`write` puede ser usado en un programa para cambiar el byte almacenado en la EEPROM.

```
write    log, 25
          ---- constante a almacenar
          ----- lugar de la EEPROM donde ubicarlo
```

O, con una variable,

```
write log, x
      ----- variable RAM
      ----- lugar de la EEPROM donde ubicarla
```

Los datos de la EEPROM se almacenan como bytes solamente. (Tema avanzado: La variable RAM en la instrucción anterior puede ser word, byte, nib o bit, pero los bits que sobran de la izquierda son truncados, y en el caso de faltar bits se agregan hasta completar la celda de un byte de la EEPROM.)

```
' Mediciones Ambientales programa 2.7
' diferenciación de constante, dato y variable
dit      con      70          ' define una constante
log      data      7          ' reserva un byte en eeprom, valor inicial 7
worm     data     240         ' reserva un byte en eeprom, valor inicial 240
x        var      byte        ' define dos variables
y        var      byte
read log,x          ' lee datos de la eeprom
read worm,y         ' y los asigna a las variables
debug ? dit, ? log, ? x, ? worm, ? y ' muestra los valores
```

El valor de **dit** es 70, una constante común. El nombre **dit** se refiere al valor en si. Los valores de **log** y **worm** son constantes, pero tienen valor de 0 y 1, no 7 y 240. Los nombres **log** y **worm** se refieren indirectamente a los datos. Para leer el 7 y el 240, hay dos instrucciones de lectura en el programa. Una lectura obtiene el 7 de la dirección **log=0** de la EEPROM y la ubica en la variable **x** de la RAM, y la segunda lectura obtiene el 240 de la dirección **worm=1** de la EEPROM y la ubica en la variable **y** de la RAM. Los rótulos **log** y **worm** tienen las direcciones 0 y 1 debido a que PBASIC asigna direcciones de memoria para datos comenzando por 0.

Ahora modifique el programa anterior, agregándole cuatro líneas al final.

```
' Mediciones Ambientales programa 2.8
' escribiendo una variable
dit      con      70
log      data      7
worm     data     240
x        var      byte
y        var      byte
read log,x
read worm,y
debug ? dit, ? log, ? x, ? worm, ? y
x=x+1      ' □ cambia el valor de x
y=y/2      ' □ cambia el valor de y
```

```
write log,x           ' ☐ cambia el valor almacenado en log
write worm,y         ' ☐ cambia el valor almacenado en worm
```

Ejecute este programa y presione RESET en la Plaqueta de Educación un par de veces, con la ventana de debug activa. Debería ver los valores de *x* incrementarse en 1 cada vez, y el valor de *y* reducirse a la mitad. Luego desconecte la alimentación, y reconéctela. El primer valor que verá en la pantalla debería ser el siguiente de la serie, demostrando que la EEPROM retiene los datos cuando se desconecta la alimentación. ¿Qué pasó con el 7 y el 240 que se cargaron junto con el programa? Se perdieron. La instrucción *write* cambió esos valores. La única forma de recuperar la condición inicial es ejecutar (RUN) el programa nuevamente desde la PC. Hágalo.

Hay información adicional de la instrucción *data* en las páginas 228-230 del BASIC Stamp Manual Version 1.9 (en inglés), y también de las instrucciones *read* (p. 302) y *write* (p. 341). También veremos más en los próximos experimentos.

La EEPROM es usada a menudo para almacenar ajustes y constantes de calibración que deban ser modificadas ocasionalmente. Puede ser un parámetro que indique la temperatura a la que se debe prender un ventilador, o cuántos segundos deben transcurrir antes de almacenar datos en un archivo. Este es un divertido programa de demostración que toca una escala musical cuando presiona el pulsador (single-click). Cuántas notas toca, depende de un parámetro que está almacenado en la EEPROM. Si mantiene presionado el botón, el programa entra en una rutina de calibración, donde escuchará una serie de pulsos. Suelte el botón después de unos cuantos pulsos, y luego presione nuevamente el botón (single click).

```
' Mediciones Ambientales programa 2.9
' almacenando una configuración en la eeprom
dit          con    70          ' longitud de sonido, milisegundos
cuantos      data   1          ' cantidad inicial de sonidos
cantidad     var    word       ' variable RAM para la cantidad de sonidos
n            var    word       ' variable multipropósito
tono         var    word       ' frecuencia del sonido

klik:        ' espera a que se presione el botón
  if in1=1 then klik
  n=500      ' decide si está alto(1) o bajo(0)
klik1:      ' inicializa el timer para kliklargo
  n=n-1     ' espera aquí hasta que se libere el botón
  if n=0 then kliklargo
  if in1=0 then klik1
  tono=4519 ' decrementa el contador de kliklargo
  read cuantos, cantidad
  for n=1 to cantidad
  freqout 0,dit,tono
  pause dit ' si n=0 salta a kliklargo
            ' decide si está alto(1) o bajo(0)
            ' este es el primer tono
            ' lee cuantos tocar de la eeprom
            ' sonido, duración dit, frecuencia tono
            ' silencio breve
```

```

    tono = tono**61858          ' siguiente nota de la escala cromática
next                          ' regresa hasta que se ejecute el último
goto clik                     ' vuelve al inicio
end

cliklargo:                   ' ingresa con n=0
    freqout 0,2,3800          ' pulso corto
    pause 400                 ' pausa corta (tiempo para responder)
    n=n+1                     ' se incrementa n
    if in1=0 then cliklargo    ' espera aquí hasta que se libere el botón
    write cuantos,n           ' almacena el nuevo parámetro
goto clik                     ' vuelve al inicio
end

```

Trate de imaginarse como funciona, en detalle. Está basado en subrutinas extraídas de los programas anteriores. (la fórmula matemática, $\text{tono} = \text{tono} \times 61858$, genera la escala cromática, pero no es necesario que lo comprenda ahora.) Es necesario que comprenda la función de `read` y `write`. Hay un comando `read` para obtener la cantidad de notas a tocar, y un comando `write` para almacenar el número seleccionado por el usuario.

Para verificar si comprendió todo, modifique el programa con las siguientes pautas:

1. Agregue una instrucción `data` con la etiqueta "dur" con un valor inicial de 70 milisegundos.
2. Cambie "dit" de constante a variable `byte`.
3. Al comienzo del programa guarde el valor de "dur" en la variable "dit". En este punto, el programa debería funcionar igual que antes.
4. Al final de la rutina `cliklargo`, antes de que regrese a "clik", espere a que presione y libere el botón por segunda vez.
5. Mientras que el botón este presionado por segunda vez, incremente el valor de "n" cada vez que ejecute el bucle.
6. Cuando se libere el botón, escriba el valor de `n` en la dirección "dur".
7. Verifique que el programa funciona, y que la rutina `cliklargo` le permite cambiar el número de notas y también la duración de las mismas.

Termómetro que Habla, Revisión del Código Morse

Ahora abra el programa que guardó en el Experimento 1. Para hacer esto, presione ALT-L si está usando la versión de DOS de STAMP2.EXE, o presione CTRL-O o use el mouse si está usando la versión de Windows, STAMPW.EXE.

El programa del Experimento 1 lee la temperatura del DS1620 y la muestra en la pantalla de debug. Después de abrir el programa, ejecútelo para controlar que funcione. Nunca se puede estar seguro, Tal vez movió accidentalmente un cable de la Plaqueta de Educación, o tal vez alguien modificó su programa. Es una buena costumbre controlar que todo funciona correctamente, en cada paso de la construcción de un sistema complejo.

En su versión original, el programa muestra la temperatura en la pantalla de debug, una vez por segundo. Modifiquémoslo, para hacer que el piezoeléctrico emita la temperatura usando código Morse. El código Morse en el primer experimento de Mediciones Ambientales fue una introducción, sólo enviaba el número 50. Necesitamos una subrutina que emita los sonidos correspondientes a cualquier número de dos dígitos que le entregemos. Y cambiaremos el programa de forma que el pulsador dé comienzo a la lectura de temperatura. Partiendo del programa del Experimento 1, las líneas nuevas están marcadas con \square , y las modificadas con \blacktriangle .

```
' Mediciones Ambientales programa 2.10
' termómetro que habla, usando código morse.
dit      con    70          '  $\square$  milisegundos de dit del código Morse
dit2     con    2*dit       '  $\square$  constantes dependientes del
dah      con    3*dit       '  $\square$  valor de dit
mc       var    byte        '  $\square$  variable temporal para patrón de Morse
xm       var    byte        '  $\square$  variable de entrada Morse
j        var    nib         '  $\square$  contador de dígitos a enviar
i        var    nib         '  $\square$  contador de dits y dahs
x        var    byte        ' define una variable multipropósito, byte
C        var    byte        ' define una variable para retener grados Celsius
                                ' nota: DS1620 preprogramado para el modo 2.
                                ' high 13:shiftout 15,14,[12,2]:low 13

outs=%000000000000000000    ' define el estado inicial de todos los pines
    'fedcba9876543210
dirs=%11111111111111101    '  $\square$  como salidas en estado bajo
                                '  $\square$  excepto P1, como entrada para pulsador

freqout 0,20,3800            ' sonido de inicio
high 13                      ' selecciona el DS1620
shiftout 15,14,lsbfirst,[238] ' envía el comando "comenzar conversión"
low 13                       ' finaliza el comando
klik:                         '  $\square$  espera a que se presione el botón
    if in1=1 then klik        '  $\square$  decide si está alto(1) o bajo(0)
klik1:                        '  $\square$  espera a que se libere el botón
    if in1=0 then klik1       '  $\square$  decide si está alto(1) o bajo(0)
high 13                      ' selecciona el DS1620
    shiftout 15,14,lsbfirst,[170] ' envía el comando "obtener datos"
    shiftin 15,14,lsbpre,[x]    ' obtiene los datos
```

```

low 13          ' fin del comando
C=x/2          ' convierte los datos en grados C
debug ? C      ' muestra la temperatura en la pantalla de debug
xm=C          '  subrutina morse espera datos en variable xm
gosub morse    '  salta a la subrutina
goto clik      '  vuelve al inicio (***)

morse:         '  emite el byte xm en código morse
  for j=1 to 0 '  emite 2 dígitos, decenas primero
    mc = xm dig j '  toma el dígito(j+1)
    mc = %11110000011111 >> mc '  fija el patrón para el código morse
    for i=4 to 0 '  5 dits y dahs
      freqout 0,dit2*mc.bit0(i)+dit,1900 '  emite el patrón de bits de mc
      pause dit '  silencio corto
    next i '  next i, fin de los cinco dit o dah
    pause dah '  silencio entre dígitos
  next j '  next j, fin de los dígitos
return '  vuelve al inicio (mediante ***)
end

```

Descargue el programa y presione el botón. Escuche el código Morse mientras hace subir y bajar la temperatura. Si usted no es un radioaficionado, puede necesitar un poco de práctica para identificar los números del código Morse. Pero no le llevará mucho tiempo. Puede leerlos en la pantalla a medida que los escucha. Puede calentar el sensor de temperatura DS1620 con su dedo, o colocándolo bajo una lámpara, o exponiéndolo al sol.

Este termómetro que habla, es un instrumento útil. Un invidente podría usarlo. ¿O tal vez un biólogo realizando una investigación sobre murciélagos en una cueva oscura? (Si usa auriculares, porque los murciélagos son muy sensibles a los sonidos de alta frecuencia.) ¿Puede imaginar otras situaciones donde este dispositivo sea útil?

Por favor guarde este programa en el disco. ¿Nombre del programa? (Cmorse.bs2)

Ahora analicemos el programa paso a paso. (En el resto del experimento se verá una explicación detallada de la rutina del código Morse, no realizará más programas hasta los desafíos).

Muchas variables y constantes se definieron al principio del programa. Algunas las reconocerá del Experimento 1, donde aparecieron en la rutina que enviaba el número 50 en código Morse. Está la longitud de dit en milisegundos, y la de dah, que se define como tres veces dit, y una nueva, dit2, que se define como dos veces la longitud de dit. Hay también un par de variables nuevas, `xm` y `mc`, de las que hablaremos en relación a la subrutina del código Morse que explicaremos más adelante.

P1 es una entrada, para el pulsador. P1 es fijada como entrada al hacer igual a cero el bit correspondiente en `dirs`. Las siguientes instrucciones fijan como entradas o como salidas en estado bajo, a los 16 pines del BASIC Stamp.

```
outs=%0000000000000000      ' define el estado inicial de todos los pines
    ' fedcba9876543210
dirs=%1111111111111101      ' como salidas en estado bajo
    ----- esta es fijada como entrada para el pulsador
```

Observe el pequeño cambio sobre el programa original. Si no fijamos a cero ese bit en la instrucción `dirs`, el programa no podrá leer el pulsador. Si no lo cree, inténtelo y vea que pasa. Usted debe preguntarse cómo lograban leer el estado del botón los primeros programas de este experimento, sin usar las instrucciones `dirs` y `outs`. La razón es que el BASIC Stamp siempre inicia con los pines fijados como entradas. Como técnica de buena programación, hacemos todos los pines salidas, excepto aquellos que se necesiten como entradas. Cuando definimos como entrada un pin como P1, deja de tener importancia el estado asignado por el comando `outs`. El estado de `outs` no tiene efecto sobre un pin que se define como entrada.

La idea central de la subrutina Morse reside en el patrón binario, 11110000011111. El signo , indica número binario. Así es como se almacenan efectivamente los números, en el cerebro digital del BASIC Stamp. Este número binario corresponde al decimal 15391, pero en este caso el valor del número carece de significado. A menudo en computación, se debe pensar sobre los datos, como algo independiente de su valor numérico decimal. Por ejemplo si usted detectara los primeros 5 bits moviéndose a la derecha, obtendría 11110. Esto se traduce a código Morse obteniendo dah dah dah dah dit, número nueve. (No es el número binario nueve, que sería 1001. En lugar de eso, es el patrón del número 9 en código Morse. ¡Hay muchas formas de representar números!) Dependiendo de dónde comienza a observar el patrón binario del código Morse, se obtendrá un número distinto. Los números están ordenados, tomados de a cinco bits, en orden decreciente. Es un truco.

```
11110000011111
^^^^^-----> 11110, dah dah dah dah dit  nueve (tomando de cinco bits)
  ^^^^^-----> 11100, dah dah dah dit dit  ocho
    ^^^^^-----> 11000, dah dah dit dit dit  siete
      ^^^^^-----> 10000, dah dit dit dit dit  seis
        ^^^^^-----> 00000, dit dit dit dit dit  cinco
          ^^^^^-----> 00001, dit dit dit dit dah  cuatro
            ^^^^^-----> 00011, dit dit dit dah dah  tres
              ^^^^^-----> 00111, dit dit dah dah dah  dos
                ^^^^^-----> 01111, dit dah dah dah dah  uno
                  ^^^^^-----> 11111, dah dah dah dah dah  cero
```

Ahora analicemos la subrutina del código Morse paso a paso. Primero, debe reconocer que es una subrutina, que comienza con la etiqueta "morse:", y finaliza con la instrucción "return". La rutina principal, después de obtener la lectura de la temperatura en grados C del sensor DS1620, y ponerla en la variable xm ejecuta el comando gosub morse. La subrutina morse realiza su función y el programa se continúa ejecutando a partir de la línea posterior a la instrucción gosub morse, que es "goto klik". Escribiendo el segmento de programa morse como una subrutina, nos permitirá llamarla en diferentes puntos de nuestro programa, a medida que aumenta su tamaño, sin tener que rescribir el segmento cada vez.

La variable xm es el dígito que se emitirá en código Morse. En la subrutina morse hay dos bucles for..next, anidados (uno dentro del otro). El bucle exterior tiene el índice j:

```
for j=1 to 0                                ' emite dos dígitos, primero decenas
  mc = xm dig j                             ' extrae el dígito j+1
  mc = %11110000011111 >> mc              ' configura el patrón para el código
  ...                                       ' más código aquí
next                                         ' siguiente dígito de dos
return                                     ' 
```

¿Qué es un

Índice y puntero:

Un índice es una variable que pasa por una secuencia de valores. Por ejemplo, "j" en el bucle for-next pasa por los valores 1 y 0. Un puntero es una variable que especifica en qué lugar de la memoria, o en qué lugar de un conjunto de datos, obtener información. Por ejemplo, la variable "j" es índice y puntero. Apunta a un dígito en la variable xm. El índice "j" en el mismo programa es un puntero a los bits de la variable mc. En próximos experimentos, usaremos índices y punteros para trabajar con los datos almacenados en la EEPROM, como 1^{er} lectura, 2^{da} lectura, etc.

Cuando el programa llega por primera vez a la rutina morse, le asigna a j el valor 1, y luego continúa con j 1 el resto del bucle, incluyendo la parte que dice "más código aquí". La palabra reservada, next, es el punto de regreso para el bucle for-next, y en este punto el programa salta de regreso a su correspondiente for, fija j 0, y ejecuta todo otra vez, hasta el next. ¡Observe que el BASIC Stamp sabe contar para atrás! Después que j ha tomado los valores 1 y 0, el bucle finaliza, y el programa regresa a la rutina principal, y de ahí al inicio (klik).

Hay dos instrucciones matemáticas en este bucle externo. La primera es:

```
mc = xm dig j.
```

Este "dig" es un operador, al igual que "más" o "dividido por". Se emplea entre dos números, xm y j, y entrega el dígito (j 1) de xm. Es fácil de demostrar con un ejemplo específico. Suponga que el valor es xm 25. En la primer pasada por el bucle, el valor de j es 1, y el resultado de (mc = 25 dig 1) será (mc=2), debido a que 2 es el segundo dígito de 25. En la segunda pasada, el resultado de (mc = 25 dig 0) será mc=5, debido a que 5 es el primer dígito de 25.

25

```
j 1 puntero de decenas-----
j 0 puntero de unidades-----
```

Esta lógica se puede extender a números más grandes, por ejemplo, `j 3` apuntará a millares. Sin embargo, en ésta rutina de código Morse sólo necesitaremos 2 dígitos.

Ahora tenemos un número entre 0 y 9 en la variable `mc`. La siguiente instrucción prepara el patrón del código Morse.

```
mc = %11110000011111 >> mc ' ' configura el patrón para el código
```

El símbolo `>>` es otro operador que se usa entre dos números. La constante, `%11110000011111`, es el patrón binario del que hablamos anteriormente. El operador `>>` trabaja exclusivamente con números binarios. Es llamado operador de desplazamiento. (Los desplazamientos de bits son muy importantes en la computación.) Desplaza el patrón binario hacia la derecha, agregando ceros por la izquierda, cierto número de lugares (`mc` lugares) y elimina la misma cantidad de bits que han salido por la derecha. Nuevamente, ejemplifiquemos con 25, la primera pasada por el bucle, el dígito es 2 cuando el programa llega a este comando:

```
BEFORE    mc= 11110000011111 >> 2
AFTER      00111100000111      ' patrón desplazado 2 bits a la derecha
                        \11      ' dos bits perdidos
                        ^^^^^----- 5 últimos bits son el patrón morse "2"
```

Y en la segunda pasada por el bucle, el dígito es 5:

```
BEFORE    mc= 11110000011111 >> 5
AFTER      00000111100000      ' patrón desplazado cinco bits a la derecha
                        \11111    ' cinco bits perdidos
                        ^^^^^----- 5 últimos bits son el patrón morse "5"
```

Lo que se logra con este método es modificar el patrón del código Morse reasignándolo a la variable `mc`, siendo de interés los últimos cinco bits (del 4 al 0). En el ejemplo, 00111 representa 2 en código Morse, y 00000 representa 5. Ahora veamos como emitir los últimos cinco bits de la variable `mc`.

Ahora el patrón del código Morse está en posición, y entramos en el bucle `for-next` interior:

```

for i=4 to 0                                ' 5 dits y dahs
  freqout 0,dit2*mc.bit0(i)+dit,1900        ' emite el patrón de bits de md
  pause dit                                  ' silencio corto
next                                          ' siguiente dit o dah de cinco
pause dah                                    ' silencio entre dígitos

```

El índice en este caso es `i`, y pasa por 5 valores, contando de 4 a cero (cuenta descendente). El comando `freqout` toca un dit o un dah cada vez que se ejecuta el bucle interno. Entre sonidos, hay una pausa corta con la misma duración de dit. Después que los 5 dits y dahs de un dígito son emitidos, hay una pausa más larga, con la misma duración de dah, y luego el programa se repite, emitiendo los cinco bits de las unidades, de la misma forma.

El comando `freqout` es conocido, excepto que la duración no es ni una constante ni una variable simple. Es una expresión. PBASIC le permite hacer esto. La expresión es:

```

dit2*mc.bit0(i)+dit
^^^^^^^^^^-----esto puede valer 0 o 1.

```

Comencemos diciendo que `mc.bit0(i)` es una variable que puede valer cero o uno solamente. Entonces los casos posibles son,

```

dit2 * 0 + dit ==> dit
o
dit2 * 1 + dit ==> 3*dit ==> dah

```

El comando `freqout` toca un dit o un dah, dependiendo del valor de la misteriosa variable.

¿Qué es exactamente `mc.bit0(i)`? Una característica poderosa del PBASIC es que permite el acceso a bits individuales de un byte. El byte, `mc`, tiene 8 bits. La notación, `mc.bit` es llamada modificador de la variable byte `mc`. Es la forma de llamar al bit menos significativo de ese byte. El segundo bit es `mc.bit1`, y así hasta `mc.bit4`, que es el quinto bit. Es simplemente una forma de llamar a los bits, una sintaxis que es reconocida por el lenguaje PBASIC.

Hay otra forma de referirse a los mismos bits, usando una variable como **puntero (o apuntador)** a los bits del byte. Esta notación es `md.bit0(i)`. Por ejemplo, `md.bit0(4)` y `md.bit4` se refieren al mismo bit. Literalmente significa, "el cuarto bit desde `md.bit0`". Vea el BASIC Stamp Manual, v1.9 pp. 221-224 (en inglés) para obtener más información.

Esta es la forma en la que trabaja:

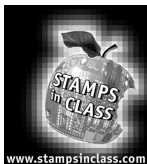
```
00111    <-- estos son los últimos cuatro bits de la variable mc
    ^---- mc.bit0    o    mc.bit0(0)    diferentes nombres para el mismo bit
    ^---- mc.bit1    o    mc.bit0(1)
    ^----- mc.bit2    o    mc.bit0(2)
    ^----- mc.bit3    o    mc.bit0(3)
    ^----- mc.bit4    o    mc.bit0(4)
```

La variable `i` es el puntero. La utilidad de este sistema vectorial, es que el programa se repite (para `i` 4 a 0) tomando todos los valores binarios de los bits de la variable byte `mc`. Esos son los bits que deben ser emitidos, 0 `dit` y 1 `dah`. Esta es otra forma de tocar los cinco `dits` y `dahs`, sin usar un bucle `for-next`:

```
freqout 0,dit2*mc.bit0+dit,1900    ' ☐ primer bit
pause dit                          ' ☐ silencio corto
freqout 0,dit2*mc.bit1+dit,1900    ' ☐ segundo bit
pause dit                          ' ☐ silencio corto
freqout 0,dit2*mc.bit2+dit,1900    ' ☐ tercer bit
pause dit                          ' ☐ silencio corto
freqout 0,dit2*mc.bit3+dit,1900    ' ☐ cuarto bit
pause dit                          ' ☐ silencio corto
freqout 0,dit2*mc.bit4+dit,1900    ' ☐ quinto bit
pause dit                          ' ☐ silencio corto
```

Como se observa, se hace referencia a cada bit, uno a la vez. Pero es más corto, y más elegante (?) usar el bucle `for-next` y el índice como puntero a los bits.

¡Bien! Fue mucha explicación para un segmento muy corto de programa. Pero contiene algunas ideas importantes. Cómo interpretar un número como patrón. **Índice y puntero**. Cómo extraer dígitos decimales. Los operadores `dig` y `shift`, cómo usan una expresión como parámetro. Cómo usar modificadores vectoriales de variables PBASIC. De esto se trata la programación de microcontroladores.



¡Desafío!

Conecte un led a P5, de forma que se encienda con `high 5`. Escriba un programa que encienda el led con un click del botón, y lo apague con el próximo click. Pista: aunque hay varias formas de hacer esto, el comando `toggle` puede ayudar. Ver página 339 del BASIC Stamp Manual Version 1.9 (en inglés).

- (A) Haga un programa BS2 que imprima "trabajando" en la pantalla de debug, y toque un sonido, cada vez que presiona el botón. Pista: imprima un mensaje en la pantalla usando comandos como `debug "trabajando"`, `CR` CR significa salto de línea
- (B) Luego prográmelo de forma que si mantiene presionado el pulsador, mientras presiona y libera RESET en la Plaqueta de Educación, no entre directamente en la rutina "trabajando". En lugar de eso, que imprima en la pantalla debug "Espero sus instrucciones", toque un sonido diferente, y espere hasta que se presione nuevamente el botón. (Piense en las impresoras, cómo algunas imprimen una "test page" página de prueba, si mantiene presionado algún botón del panel frontal mientras la enciende.)

El programa 2.10 mide la temperatura en grados Celsius.

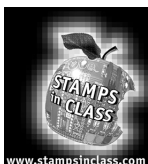
- (A) Modifique el programa de forma que muestre grados Fahrenheit, y los emita en código Morse.
- (B) Modifique la rutina del código Morse para que toque tres dígitos en lugar de dos, en caso de que la temperatura sobrepase los 99 Fahrenheit.
- (C) (Avanzado) Si quiere divertirse, haga el programa del inciso B pero que no toque los ceros de la izquierda, es decir, si la lectura es de 76 grados F, tocará "7", "6", y no "0", "7", "6".

Luego intente esto:

- (A) Comience con un byte de datos, iniciado en cero, almacenado en la EEPROM. Cada vez que se presiona el botón, se incrementa una unidad el valor almacenado en la EEPROM (`read`, `incremento`, `write`), y muestra el valor actual en la pantalla debug.
- (B) Cuando el valor llega a 7, pone el mensaje "acceso denegado" en la pantalla debug, toca una alarma y hace parpadear el led. En este punto, si resetea (reinicia) el Stamp o interrumpe momentáneamente la alimentación, la "alarma" sonará de todas formas (`read` e `if`, al comienzo del programa.).

- (C) (Avanzado) Piense en una forma, usando una acción especial del botón, como tener presionado durante mucho tiempo (long click), de resetear (reiniciar) el valor almacenado en la EEPROM a cero. Esto le permitirá el acceso, de forma que puede presionar el botón 7 veces más antes de que la alarma se active.

Escriba un programa que toque un tono si hace un triple click en el pulsador.



Experimento 3: Punta de Temperatura para Micro-Ambientes

El objetivo del Experimento 3 es conectar una sonda de temperatura en el extremo de un cable que pueda proyectarse más allá de la Plaqueta de Educación, para monitorear micro ambientes. Un sensor bien calibrado, con buena resolución, obtendrá resultados muy precisos. Las actividades específicas de este

experimento constan de: (1) Colocar un capacitor en una entrada del BASIC Stamp, y usar el comando `rctime` (2a) Medición de temperatura usando la punta AD592, con calibración en un termo con hielo y (2b) Comparación de calibración con el DS1620 a temperatura ambiente (3) Calibración automática usando la EEPROM del BASIC Stamp y (4) Experimentos de temperatura en código Morse, radiación solar, sensor húmedo/sensor seco, viento frío.



Partes Requeridas

Para este experimento, dejaremos los componentes de los experimentos anteriores en su ubicación, en la Plaqueta de Educación. Son necesarios los siguientes componentes:

- (1) punta de temperatura AD592. Vea el apéndice B si prefiere construir su propia punta, en lugar de usar la que se incluye en el Kit de Componentes de Mediciones Ambientales.
- (1) precintos de nylon (para sujetar la punta).
- (1) capacitor de 0.1 F monolítico
- (2) capacitor de 0.22 F (film de poliéster)
- (2) resistor 100Ω
- (2) resistor 100KΩ
- (3) cables de interconexión
- (1) punta de continuidad (dispositivo con dos tornillos de 5 cm, separados 1 cm, montados en un recorte de circuito impreso con cables de conexión)
- (1) Cuba con hielo. Se logran mejores resultados con hielo molido y agua dentro de un termo. Si no tiene un termo, use un recipiente de polietileno expandido (telgopor), y envuélvalo con film de aluminio.



¡Constrúyalo!

Sensor de Temperatura Analógico

3

dos momentos distintos. Por ejemplo, la temperatura sobre la hoja de una planta expuesta al sol, puede ser significativamente diferente a la temperatura ambiente. La hoja crea su propio micro ambiente. Y a medida que crecen las plantas, crean un único micro ambiente debajo suyo. A menudo las mediciones se deben realizar en varios lugares a la vez, enviándose a un instrumento central. Por ejemplo, una estación climática agrícola medirá la velocidad del viento a gran altura, humedad del suelo, y otros parámetros con ubicaciones intermedias. Esto significa que los sensores deben ser montados en cables para alcanzar todos los micro ambientes separados.

En el Experimento 1 aprendió sobre el sensor inteligente de temperatura DS1620. Una ventaja de este sensor es que entrega las lecturas directamente en números digitales. Pero una desventaja es que es un integrado con 8 pines, difícil de convertir en una punta que pueda ser usada separada del circuito impreso. En este experimento aprenderá sobre un sensor de temperatura distinto, el AD592. Es fácil de incorporar a una punta

debido a que sólo necesita dos cables. El AD592 es un **sensor de temperatura analógico**. Analógico significa que la señal es un valor eléctrico continuo (microamperes), proporcional a la temperatura. Analógico es lo opuesto a digital, que significa que las lecturas son obtenidas como un código de valores discretos (ceros y unos). El AD592 es de tecnología "clásica" que ha sido usado durante muchos años. Muchas de las señales que encontrará en la ciencia de las Mediciones Ambientales, o en muchos campos de la ingeniería, son señales analógicas. Integrados como el DS1620 tienen sensores analógicos en su interior, y los ingenieros han trabajado mucho para darle al DS1620 su inteligencia digital.

Los sensores analógicos requieren una interfaz distinta con el BASIC Stamp II. En este experimento aprenderá sobre el comando `rcTime`. Puede que sepa algo de conversores analógico-digital, un tipo de integrado que se dedica a hacer esas conversiones. El comando `rcTime` es un rudimentario conversor analógico-digital interno del BASIC Stamp II. Como introducción al comando `rcTime`, conectará un capacitor a la entrada del BASIC Stamp y repasará las propiedades de los capacitores. Una vez que tenga a `rcTime` leyendo el sensor de temperatura, aprenderá a calibrarlo, para que lea correctamente la temperatura, a pesar de la dispersión de los valores de los componentes que forman el circuito.

¿Qué es Un

Sensor de temperatura analógico:

La corriente (microamperes) producida por el AD592 es lo que se llama una "analogía" de la temperatura. Microamperes no es lo mismo que temperatura, así como las naranjas no son manzanas. Se puede hacer una analogía entre un capacitor y un tanque de agua, así como entre temperatura y corriente eléctrica. Esta es la base de los sensores "analógicos". Otros transductores de temperatura pueden traducir la temperatura en tensión o resistencia o capacitancia. Las señales de los dos lados de la analogía son de naturaleza continua, con infinidad de valores intermedios. Analógico es lo opuesto a digital, donde las señales se transmiten en códigos digitales de ceros y unos.

Una vez que tiene la punta con cable, puede medir la temperatura de los micro ambientes aledaños.

Pines del BASIC Stamp, Capacitores, Revisión de BASIC

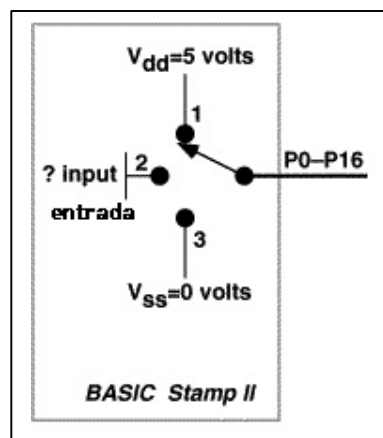
Usted probablemente ya sabe que los 16 pines E/S de propósito general del BASIC Stamp, pueden estar en tres estados distintos, en diferentes momentos. Como se muestra en la Figura 3.1, es como un interruptor de tres posiciones:

- (1) Interruptor conectado a V_{dd} 5 volts, como se muestra aquí, salida HIGH (alto). Puede circular corriente por el pin. (entrega corriente de la fuente de alimentación de 5 volt, V_{dd}).
- (2) Interruptor conectado como entrada (`input`). No circula corriente en ningún sentido por el pin. Como entrada, la circuitería interna del BASIC Stamp controla la tensión en el pin de entrada. Menos de 1,3 volts se ven como bajo (0). Más de 1,3 volts se ven como alto (1).
- (3) Interruptor conectado a V_{ss} 0 volts, salida LOW (bajo). Puede circular corriente por el pin (absorbe corriente a masa, V_{ss}).

Figura 3.1: Pines E/S del BASIC Stamp

Hay tres posiciones en este interruptor:

- (1) V_{dd} 5V
- (2) Entrada baja o alta (`input`)
- (3) V_{ss} 0V.



Comandos simples como `high 5` o `low 5` o `input 5`, ponen a ese pin instantáneamente en uno de estos tres estados. Muchos de los comandos del lenguaje PBASIC trabajan jugando con estos estados. Por ejemplo, el comando `freqout` hace un sonido conmutando rápidamente el estado de salida de un pin entre `high` y `low`. Los comandos `shiftin` y `shiftout` trabajan coordinando la actividad de varios pines a la vez, algo como salidas saltando de alto a bajo, con las entradas sincronizadas con esta acción. Ahora presentaremos el comando `ctime`, que conmuta un pin de salida a entrada y cronometra el tiempo que tarda la tensión del pin, en atravesar el umbral de 1.3 volt.

Experimento 3: Punta de Temperatura para Micro-Ambientes

El cambio en la tensión es debido a un circuito externo, normalmente un resistor (R) y un capacitor (C). El punto importante que queremos enfatizar aquí, es que el flujo de corriente por el pin de entrada, es prácticamente nulo. Sólo observa el cambio en la entrada.

3

Primero, un breve repaso sobre capacitores. Ténganos paciencia si ya sabe cómo trabajan los capacitores. La analogía es un tanque de agua, con un caño de entrada y otro de salida. El tanque almacena agua, análogamente al capacitor que almacena cargas eléctricas. Figuras 3.2 y 3.3 demuestran este punto.

Figura 3.2: Analogía, capacitor cargándose.

El agua fluye (amperes) dentro del tanque y el nivel (volts) aumenta. A mayor caudal, más rápidamente se llena. El flujo por el caño está limitado por la resistencia (ohms) del caño, o por la presión del agua en el otro extremo del caño.

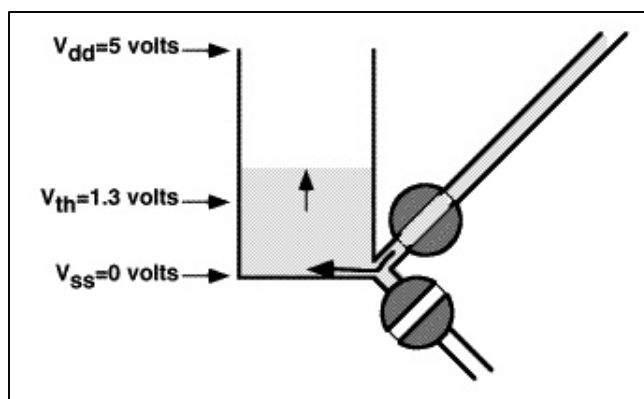
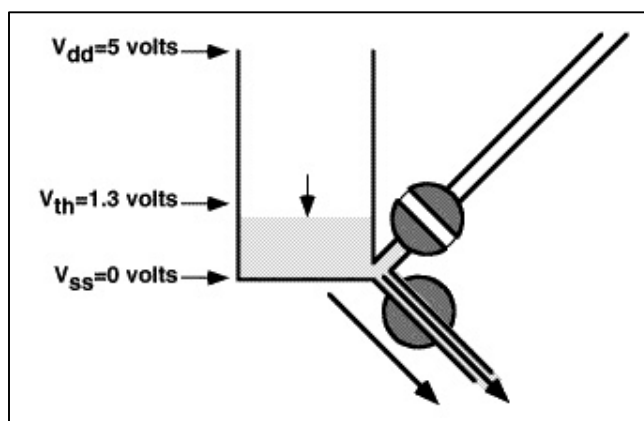


Figure 3.3: Analogía, capacitor descargándose.

El flujo (amperes) descarga el tanque y el nivel (volts) baja. El flujo puede ser escaso, un hilo de agua, o abundante, a borbotones. Si el flujo es cero (de entrada o salida), entonces el nivel permanece constante. Puede haber flujos no intencionales, llamados fugas (amperes).



Experimento 3: Punta de Temperatura para Micro-Ambientes

Los capacitores vienen en gran variedad de tamaños, y se miden desde picofaradios hasta faradios. Esto no se refiere al tamaño físico, sino a la capacidad de almacenar carga, que depende del material del que está hecho el capacitor. Dos capacitores del mismo tamaño, pueden tener capacidades muy diferentes. En este experimento usaremos valores entre 0.01 y 0.22 microfaradios (μF).

3

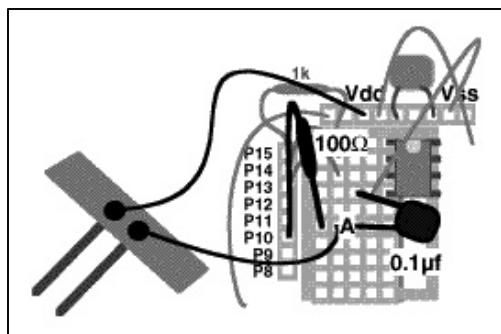
Detector de Resistencia Simple

Conecte un capacitor de 0.1 μF , un resistor de 100 Ω , y un sensor de conductividad como en la Figura 3.4. El circuito eléctrico se muestra en la Figura 3.5.

Figura 3.4: Capacitor y sensor en P10

El rótulo del capacitor probablemente será "104" o ".1" en letras pequeñas. La orientación de estos capacitores no tiene importancia.

- 0.1 μF desde el pin 4 del DS1620 al nodo A
- resistor 100 Ω desde nodo A a P10
- sensor de conductividad desde nodo A a Vdd (5 volts)



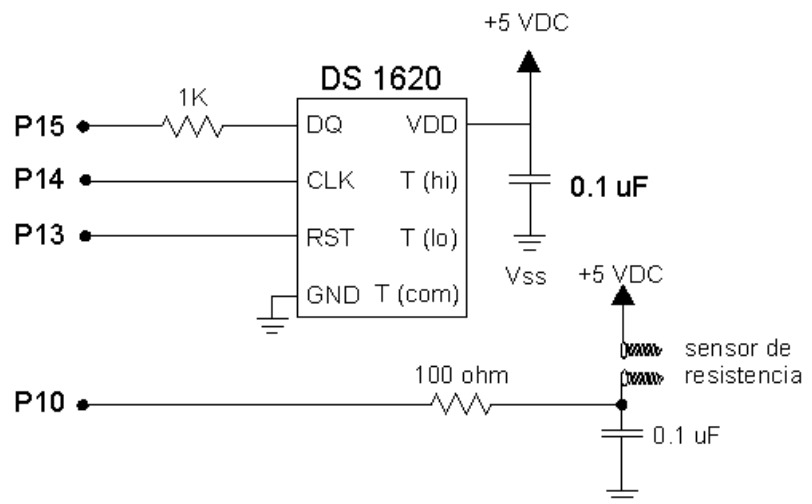
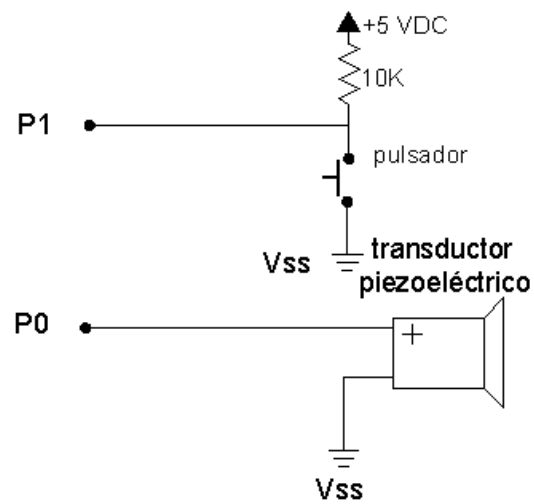


Figura 3.5: Esquema Eléctrico del Detector de Resistencia Simple



Una vez construido el circuito, cargue el siguiente programa:

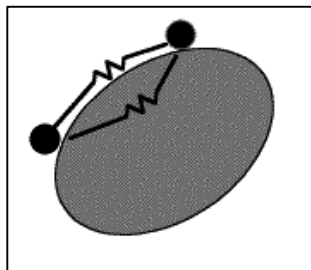
```
' Mediciones Ambientales programa 3.1
' simple demo de un capacitor en un pin del BS2.
v var bit ' variable tamaño bit, estado entrada
bucle1: ' viene aquí para descargar el capacitor
  low 10 ' descarga el capacitor a 0 volts
  freqout 0,5,3500 ' emite sonido
  debug CR ' nueva línea en la pantalla
  input 10 ' hace el pin una entrada
bucle2: ' espera a que la entrada sea >1.3 volts
  v=in10 ' lee la entrada
  debug bin v ' la muestra en la pantalla
  pause 99 ' pausa de 0.1 segundos
branch v,[bucle2,bucle1] ' vuelve a bucle1 si la tensión es >1.3 volts
```

La primera instrucción del programa descarga el capacitor a cero volts. El capacitor se descarga muy rápido, como un caño grande que arroja el agua al piso. La corriente del microcontrolador PIC del BASIC Stamp puede descargar el capacitor a través de un resistor de 100 ohm, en aproximadamente 25 microsegundos, que es mucho menos que la duración de 10 milisegundos, del comando `freqout`. Luego aparece la instrucción `input 10`. El pin instantáneamente se convierte en entrada. Déjelo un minuto o dos, sin tocar nada. ¿escucha algún sonido o ve algún 1 en la pantalla? ¿No? No se sorprenda si el capacitor permanece descargado, porque no hay fuente de corriente para cargarlo. Todos esos ceros en la pantalla significan que el capacitor sigue descargado.

Ahora toque los dos extremos del sensor de conductividad con sus dedos. ¡Experimente! El resultado dependerá de qué tan húmedos están sus dedos, o qué tan fuerte presiona (¿Un detector de mentiras?). Hay fugas de corriente a través de la humedad de sus dedos, y de la piel. Pruebe sumergiendo el extremo del sensor en agua, o tocar un papel mojado, o tocar una línea gruesa dibujada sobre un papel con un lápiz. También puede probar con un resistor de 100K ohm. Su dedo cortocircuita el capacitor como se muestra en la Figura 3.6.

Figura 3.6: Cortocircuito

Toque los dos tornillos de la punta de continuidad. Su dedo cortocircuitará el capacitor.



Experimento 3: Punta de Temperatura para Micro-Ambientes

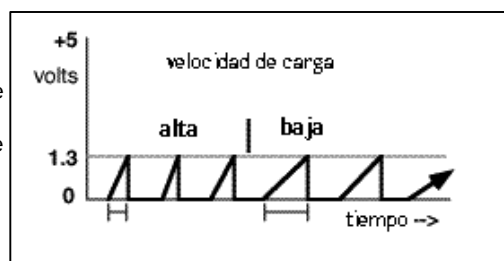
3

Debemos hacer una aclaración sobre seguridad. La tensión y la corriente de este circuito son muy pequeñas, cinco volts y unos pocos microamperes. Si desconfía de un circuito, siempre proceda con cautela.

El pin de entrada del BASIC Stamp está actuando como un "comparador". Este es un término técnico para un dispositivo que da como respuesta si o no, 1 o 0, a la pregunta, "¿es la tensión en P10 mayor que 1,3?" Este umbral de 1,3 volt es fijado por el microcontrolador PIC del BASIC Stamp II, y no podemos hacer nada para cambiarlo. La Figura 3.7 muestra como trabaja.

Figure 3.7: Descarga del Capacitor

Una y otra vez el capacitor es descargado a cero volts, y luego, más o menos rápidamente se recarga al umbral de 1,3 volt. Variando la resistencia de la punta, se afecta la velocidad de carga.



El programa usa la instrucción `branch`, que puede ser nueva para usted. Le dice al BASIC Stamp que vaya a uno de dos posibles destinos.

```
branch v,[bucle2,bucle1]  '
                        ^^^^^-----
                        ^^^^-----
                        ^-----
                        va aquí si v=1 (voltaje capacitor >1.3 volts)
                        va aquí si v=0 (voltaje capacitor <1.3 volts)
                        bifurca según esta variable
```

Si el nivel de voltaje del capacitor alcanza los 1,3 volts, entonces la variable `v` será igual a 1, el programa regresará a `bucle1` para descargar el capacitor, emitir un sonido, y saltar de renglón en la pantalla. De otra forma, el programa queda en `bucle2`, donde continúa leyendo la entrada e imprimiendo ceros en la pantalla. La variable `v` es un bit 0 ó 1, de esta forma la instrucción `branch` cubre todas las posibilidades, `bucle1` o `bucle2`. Otra forma de escribir esto podría ser,

```
if v=1 then bucle1      ' regresa a descargar el capacitor
goto bucle2             ' o sigue controlando y esperando
```

`Branch` es más conciso, y hace más prolijos a los programas. Para más información sobre la instrucción `branch`, vea el BASIC Stamp Manual Version 1.9, pg. 247 (en inglés).

Sensor de Resistencia Usando Rctime

3

Ahora escriba y descargue el siguiente programa:

```
' Mediciones Ambientales programa 3.2
' simple demo del comando Rctime.
rct var word          ' una variable word
n var byte            ' variable para el gráfico de barras
low 10                ' descarga el capacitor
bucle1:              ' repite a partir de aquí
  Rctime 10,0,rct      ' tiempo para que la tensión llegue a 1,3 volts
  low 10               ' descarga el capacitor a 0 volts
  debug ? rct          ' muestra el tiempo
  n=(rct-1)/2048+1      ' calcula la longitud del gráfico de barras
  debug rep "*" \n,cr   ' muestra el gráfico de barras
goto bucle1
```

Vuelva a experimentar, con diferentes humedades y presiones. ¿Qué clase de valores `rct` observa?

Estos son los parámetros del comando `Rctime`:

```
Rctime 10,0,rct
      ^^^----- variable que almacena resultado(unidades de 2 us)
      ^----- el comando inicia con in10=0, finaliza con in10=1
                  (en otros casos, inicia con in10=1 y finaliza in10=0)
      ^^----- use pin 10 para este comando Rctime
```

El comando `Rctime` mide el tiempo que tarda el capacitor en cargarse, de cero al umbral de 1,3 volt. El programa hace bajo al pin 10 al comienzo, y descarga el capacitor a cero volts. El comando `Rctime` luego hace a P10 una entrada, e inmediatamente empieza a controlar el estado del pin, hasta que atraviese el umbral de 1,3 volts, a la vez que cronometra el tiempo transcurrido en unidades de dos microsegundos. Si la tensión del pin cursa el umbral de 1,3 volt, entonces el comando `Rctime` finaliza y pone el tiempo transcurrido en la variable `rct`, y el programa continúa en la instrucción siguiente a `Rctime`. En este caso `low 10`, que descarga nuevamente el capacitor a cero. Si la tensión del pin no cruza el umbral de 1,3 volt dentro de una décima de segundo (0,13107 segundos para ser exactos), el comando `Rctime` finaliza, poniendo cero en variable `rct` (para indicar desbordamiento), y el programa continúa en la instrucción siguiente a `Rctime`.

`Rctime` cuenta en unidades de 2 us (microsegundos), y el valor máximo es 65535 (el valor máximo que entra en 16 bits). De esto se desprende que el tiempo máximo es de $2 \cdot 65535 = 131.070$ microsegundos, o sea 0,13107 segundos. Ver el BASIC Stamp manual Pág. 298 (en inglés), para más información sobre `Rctime`. Reiteramos, si no pasa nada dentro de 0,13107 segundos, `Rctime` pone un cero en la variable `rct`, para indicar el desbordamiento.

El comando `Rctime` es útil para medir muchas cosas diferentes. Eléctricamente, el circuito puede ser ordenado para que el tiempo dependa de la resistencia, capacidad, tensión o corriente. Muchos transductores emiten alguna de estas cantidades eléctricas. Por ejemplo, el sensor de temperatura que acabamos de ver, transforma la temperatura en corriente eléctrica. Una simple formula nos permitirá convertir el valor obtenido por `rctime`, en temperatura. Otro tipo de sensor de temperatura que se puede usar con `rctime` es el termistor. Tiene una resistencia que varía con la temperatura. No es tan conveniente, debido a que es difícil de calibrar.

Finalmente, una explicación del gráfico de barras ASCII del programa 3.2. Esto es para continuar demostrando las capacidades del comando `debug`. Antes del advenimiento de los gráficos en computadoras e impresoras, estos gráficos ASCII eran el único medio para obtener una salida gráfica.

```
n=(rct-1)/2048+1      ' calcula la longitud del gráfico de barras
debug rep "*" \n,cr    ' muestra el gráfico de barras ascii
```

Cuando `rct` tiene un valor entre 0 y 65535, el valor obtenido en `n` estará entre 1 y 32. Observe que $65535/2048 \approx 32$. Esto define el valor máximo y mínimo. Limitamos el valor máximo a 32 debido a que el gráfico de barras debe entrar en el ancho de la pantalla `debug` de `STAMP2.EXE`. Restarle 1 a `rct` es un refinamiento. Recuerde que `Rctime` sólo espera por 0,13107 segundos, y luego regresa `rct` 0 para mostrar que el tiempo fue mayor que el límite. Si graficamos eso, el tiempo más largo, el del desbordamiento, tendría la longitud más corta del gráfico. Restándole 1, `rct` 0 se convierte en $(rct-1) \approx 65535$. (Así trabaja la matemática binaria sin negativos en 16 bits, cero menos uno es igual a 65535). El gráfico tiene más sentido de ésta forma. El comando `debug` usa el modificador "rep" para imprimir `n` asteriscos en la pantalla, seguidos por un salto de renglón. Ver el BASIC Stamp Manual Version 1.9, página 256 (en inglés), para más información sobre el modificador `rep` y el comando `debug`.

Punta de Sensado de Temperatura Usando el AD592 y RCTime

3

Ahora, quite el capacitor y el sensor de conductividad de la Figura 3.4, y construya el circuito de la Figura 3.8. El circuito eléctrico se muestra en la Figura 3.9.

Figura 3.8: Sensor de Temperatura AD592 y RCTime

Punta de sensado de temperatura AD592. El AD592 está montado y aislado con espaguete termocontraíble, en el extremo de un par de cables de 40 cm. Vea el Apéndice B para detalles de construcción, si piensa hacerlo usted mismo. Pasos de conexonado:

- Cable rojo del AD592 () conectado al lado del botón, nodo A.
- Cable negro del AD592 (-) una fila más arriba que el rojo (), nodo B.
- Otro extremo del nodo A, conectado al nodo Vdd (5 volts), cerca del piezoeléctrico.
- Otro extremo del nodo B, a través de un resistor de 100Ω a P5.
- capacitor monolítico de $0.22\mu\text{F}$ (rotulado 224) de nodo B a dos filas arriba, nodo C
- modifique el cable Vss del botón (-) al nodo C, y cable nuevo del nodo C a Vss (0 volts).

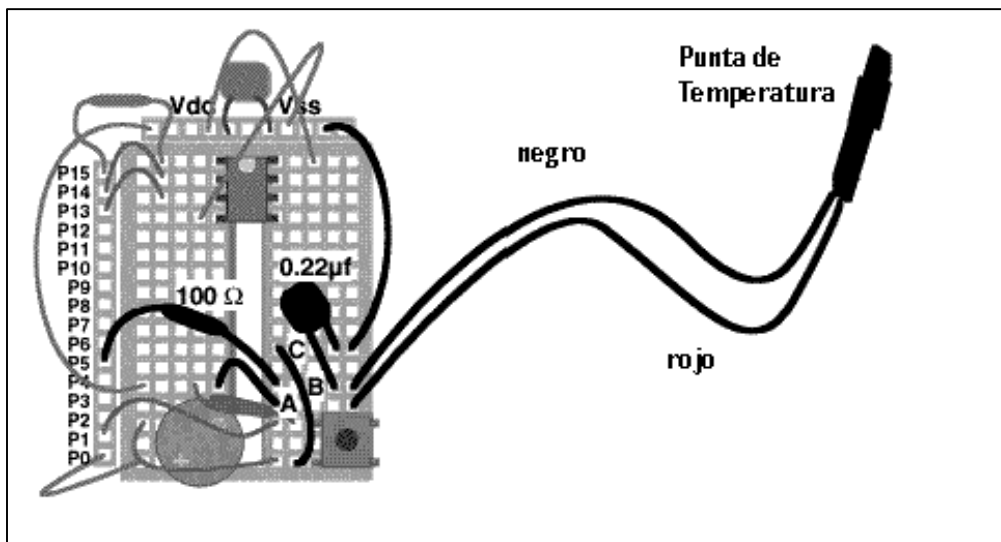
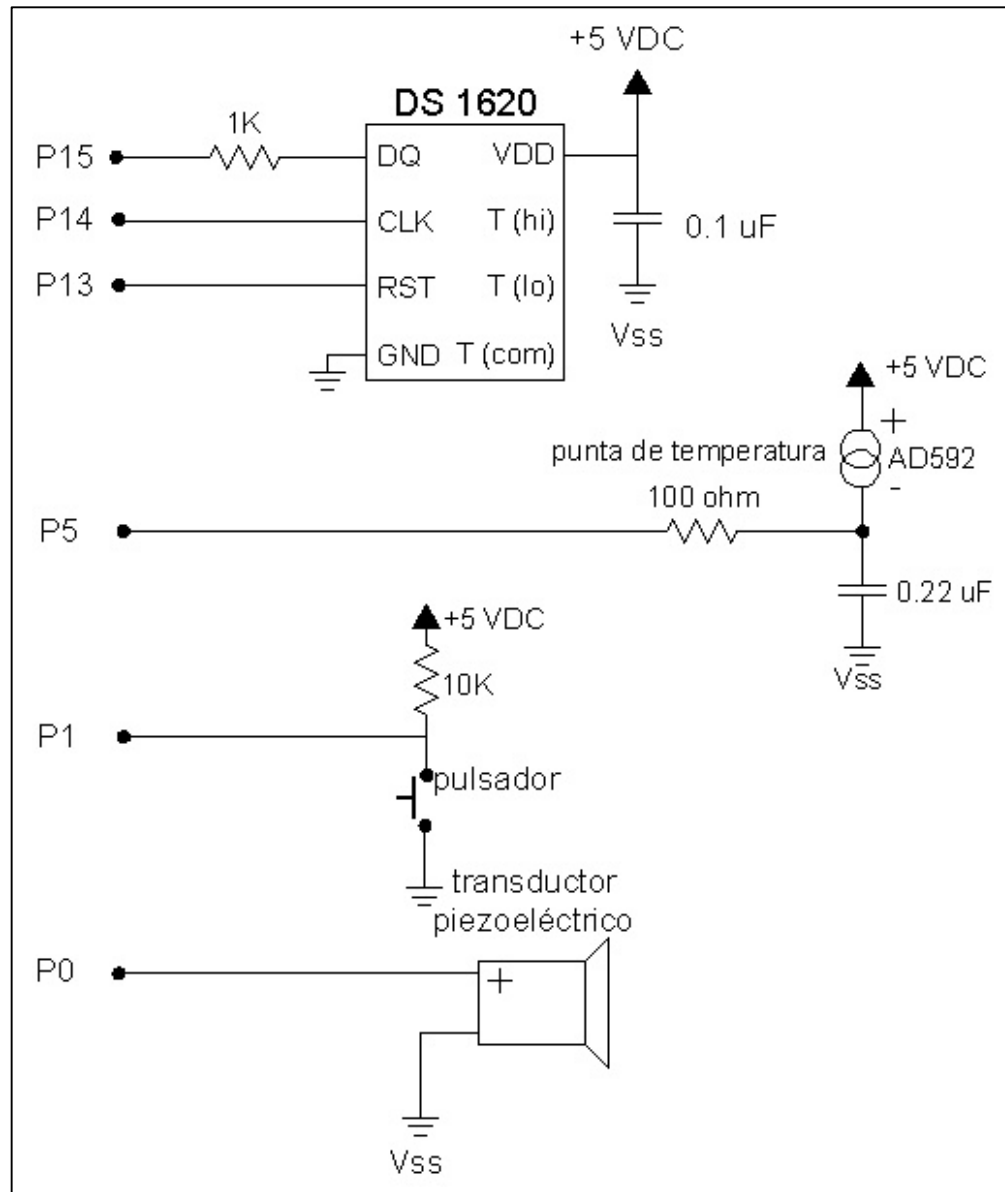


Figura 3.9: Sensor de Temperatura AD592 y Circuito RCTime

3



Pruebe el siguiente programa:

```
' Mediciones Ambientales programa 3.3
' Lecturas del sensor de temperatura AD592 usando Rctime.
Kal    con 15300                ' constante a ser determinada
rct    var word                 ' una variable word
TK     var word                 ' temperatura Kelvin
TC     var word                 ' grados Celsius
bucle:
  low 5                          ' descarga el capacitor
  Rctime 5,0,rct                 ' tiempo para alcanzar 1,3 volts
  TK = Kal/rct*10 + (Kal//rct*10/rct) ' calcula Kelvin
  TC = TK-273                    ' y Celsius
  debug dec rct,tab,dec TK,tab,sdec TC,CR ' muestra resultados
goto bucle
```

La ventana debug debería mostrar tres columnas, la cuenta directa (en unidades de dos microsegundos) de Rctime, y las temperaturas calculadas en Kelvin y Celsius. Caliente la punta de temperatura en su mano o mediante otro método no destructivo, y verifique que la lectura de rct disminuye a medida que la temperatura aumenta. Las lecturas de TK y TC deben subir con la temperatura, pero todavía no le preste atención a los valores exactos. Primero debe "calibrar" el sensor.

Nota: esta forma del comando debug separa los valores decimales de las variables con caracteres "tab", para ponerlos en columnas. El modificador "sdec" le permite mostrar números negativos.

Aclaremos que el transductor AD592 es eléctricamente, una fuente de corriente. La ecuación que gobierna su comportamiento es bastante simple:

Salida 1 microamper/ Kelvin

Es decir, a 273 Kelvin (fusión, 0 C), produce 273 microamperes. a 373 Kelvin (ebullición, 100 C), produce 373 microamperes. A cero absoluto, producirá cero microamperes, aunque esto se encuentra lejos del límite operacional de -40 grados Celsius.

Si quiere hacer una analogía entre el AD592 y el tanque de agua, es como un regulador de caudal del caño de entrada. El caudal no depende del nivel del tanque, sino que depende de la presión (tensión) que suministra la corriente del otro lado del regulador. Esto es muy diferente de un resistor o dedos húmedos, donde la corriente depende de varios factores. El nombre "Rctime" viene de: R por resistencia, C por capacitancia, y time por el tiempo que tarda el resistor en cargar al capacitor. Una fuente de corriente es un resistor regulado muy especial, que afortunadamente para nosotros, hace las cuentas más fáciles.

Calibración del AD592

La formula que relaciona la temperatura con el tiempo medido por RTime es un cociente: en este caso TK es temperatura Kelvin.

$$rct = \text{constante} / TK \quad \text{o} \quad TK = \text{constante} / rct$$

Vea el recuadro sobre la teoría de **velocidad de cambio de la tensión en un capacitor**. La constante será aproximadamente 153000 cuando el capacitor es de 0.22 uF. Pero puede variar este valor debido a la dispersión de los componentes. Por esto es necesaria la calibración.

¿Qué es la

Teoría de la velocidad de cambio de la tensión del capacitor:

La ecuación que gobierna la velocidad de cambio de la tensión sobre el capacitor es:

$$dV/dt = I/C$$

donde **I** es la corriente y **C** es la capacidad. Si sabe cálculo diferencial, y asume que **I** y **C** son constantes, puede resolver fácilmente el tiempo en función del cambio en la tensión, la capacidad y la corriente:

$$t = C * V / I$$

donde **t** está en segundos, **C** en faradios, **V** en volts, e **I** en amperes. Si sustituimos TK en Kelvin por microamperes, 0.22 f por C, 1,3 volts por V, y 2 rct por el tiempo en microsegundos, y teniendo en cuenta las unidades, obtenemos la fórmula del texto:

$$rct = \text{constante} / TK$$

La constante es 153000, cuando se reemplazan los valores ideales en la fórmula. En realidad, el capacitor no será exactamente de 0.22uF, el umbral no será exactamente 1,3 volts, y el AD592 no tendrá una salida de exactamente 1 microamper por Kelvin. Por lo tanto, como sólo hay una constante, necesitaremos un único punto de calibración.

Para calibrar el sensor, debemos hallar la constante para este caso en particular. Para lograr esto, el sensor AD592 debe ser puesto a una temperatura que usted conozca con certeza. Una buena elección es un recipiente con cubitos de hielo y agua a 0° C, 273 K. Con esta referencia, TK 273, la constante será (despejando la ecuación anterior):

$$\text{constante} = 273 * rct$$

Debemos poner la punta en el recipiente a 0° C, dejar que se estabilice, leer el valor de rct, y multiplicarlo por 273, para encontrar la constante.

¡Inténtelo! Ponga la punta AD592 en el **estándar de calibración o calibrador**, y ejecute el programa para ver las lecturas de RTime en la ventana de debug. Espere a que las lecturas se equilibren (estado estable).

¿valor leído de rct? _____

Multiplique ese número por 273. Esta es su constante de calibración.

constante = _____

Experimento 3: Punta de Temperatura para Micro-Ambientes

3

Recuerde que esta constante es específica para este sensor, este BASIC Stamp, y este capacitor. Redondee el valor de la constante, y desprecie el último dígito. Con esto obtendrá un número de cinco dígitos. Este será el valor Kal que debe reemplazar en el programa.

Kal = _____

Ponga este valor en su programa 3.3, reemplazando el valor por defecto 15300. Cuando ejecute el programa, TK y TC deberían mostrar la temperatura de calibración, 273 Kelvin, 0 Celsius.

Ahora explicaremos la fórmula para calcular TK. A diferencia de las grandes computadoras, donde los

lenguajes de programación tienen muchas funciones matemáticas disponibles, usted necesitará ajustarse a las limitaciones matemáticas del BASIC Stamp. La razón por la que hacemos ese truco con la constante, es porque este número 153000 (o el valor que encontró), es mayor que el número máximo con el que puede trabajar el BASIC Stamp en 16 bits (2^{16} 65536).

Recuerde cómo hacía las divisiones en la escuela. Esto es lo mismo, pero con una notación un poco diferente. Estos son ejemplos de dos elementos esenciales en la matemática del BASIC Stamp:

Notación del BASIC Stamp: .. significado	
1432/524 2	Una barra significa DIVISION ENTERA (524 entra dos veces en 1432), nos entrega el cociente y queda un resto.
1432//524 384	Doble barra entrega el resto de la DIVISION ENTERA: 1432-(2 524) 384. El resto siempre es menor que el divisor, 384 524.

Observe que 2 524 384 1432, es decir, el cociente por el divisor más el resto, es igual al número original (dividendo). Esta es la definición de la división.

¿Qué es un

Estándar de calibración:

El punto de fusión del hielo de agua pura es una constante física. Cero grados Celsius, 32 grados Fahrenheit, 273 Kelvin (0 273,14 si quiere más precisión). Puede obtener mejores resultados si la mezcla de hielo y agua es:

- (1) hecho con hielo molido de agua destilada
- (2) puesto en un termo de boca estrecha
- (3) mezclado lentamente mientras se toma la lectura y
- (4) por lo menos 5 cm de cable son sumergidos en la mezcla.

Si no tiene un termo, puede reemplazarlo con algún recipiente de polietileno expandido bien aislado. Una preparación cuidadosa es muy importante si quiere obtener buenos resultados. Observe hasta que la lectura se fije en un valor estable, en equilibrio.

Los metrólogos (no meteorólogos), son científicos que estudian la ciencia de las mediciones precisas. Ellos piensan en todos los factores que podrían influenciar las mediciones.

Experimento 3: Punta de Temperatura para Micro-Ambientes

Valor conversión:	real	Kelvin	Celsius
143000/484	295.5	295	22
143000/485	294.8	294	21
143000/486	294.2	294	21
143000/487	293.6	293	20
143000/488	293.0	293	20
143000/488	292.4	292	19

3

La resolución real es de aproximadamente 0.6 Kelvin. Quiere decir que cada paso de `rect` representa un cambio de 0.6 Kelvin en la temperatura. Lo redondearemos a 1 Kelvin. (Perdiendo un poco de información.)

Si usáramos un capacitor más grande (digamos 0.33 F) en el circuito, la constante sería mayor, y se mejoraría la **resolución**. Por el contrario, con un capacitor mas chico (como 0.1 F), la resolución sería menor.

En el próximo experimento, controlará la calibración de la punta AD592, comparándola con el DS1620 de los Experimentos 1 y 2.

¿Qué es la

Resolución:

Supongamos que está midiendo una cantidad que puede tomar cualquier valor entre cero y 100. Si su instrumento sólo puede ver la diferencia entre "mayor de 50" y "menor de 50", entonces tiene una resolución de un bit, quiere decir que la medición es "si/no". Por otro lado, si su instrumento puede diferenciar entre 1, 2 y 3, y así hasta 100, la resolución es del 1, o aproximadamente 7 bits (7 bits, debido a que $2^7 = 128$). Resolución no es lo mismo que precisión. Si su instrumento lee 50 cuando el valor real es 52.3, entonces no es preciso, o por lo menos debería ser calibrado. Esto es cierto así tenga uno o 7 bits o más resolución.

Revisión del Termómetro que Habla, Dos Canales

Combinemos ahora este nuevo sensor con el termómetro que habla del DS1620. Cargue el programa 2.10, del Experimento 2, y agregue las líneas marcadas con el símbolo ☐. Use la constante `Kal` que usted calculó anteriormente. Mire como está hecho el programa. Simplemente se agregan unas variables y la rutina PBASIC para el AD592, al programa existente. Ponga la punta AD592 en contacto con el DS1620 en la Plaqueta de Educación, mientras escribe este programa. Lo primero que hará será comprobar si la punta AD592 y el DS1620 toman la misma lectura de la "temperatura ambiente", así que necesitará que estén a la misma temperatura.

```
' Mediciones Ambientales programa 3.4
' termómetro que habla, dos canales.
dit      con    70          ' milisegundos para el dit del código Morse
dit2     con    2*dit       ' constante dependiente de dit
dah      con    3*dit       ' ídem
mc       var    byte        ' variable temporal para patrón Morse
xm       var    byte        ' variable de entrada morse
j        var    nib         ' índice de los dígitos a enviar
i        var    nib         ' índice de dits y dahs
x        var    byte        ' define una variable de propósito general, byte
C        var    byte        ' define una variable para almacenar Celsius

Kal      con    15300       ' ☐ USE SU PROPIA CONSTANTE DE CALIBRACIÓN
TK       var    word        ' ☐ temperatura en Kelvin del AD592
TC       var    word        ' ☐ Celsius del AD592
rct      var    word        ' ☐ para el temporizador RC.

outs=%0000000000000000    ' define el estado inicial de todos los pines
'fedcba9876543210
dirs=%1111111111111101   ' como salidas en estado bajo
                           ' excepto P1, como entrada para el pulsador

freqout 0,20,3800         ' sonido de inicio
high 13                   ' selecciona el DS1620
shiftout 15,14,lsbfirst,[238] ' envía el comando "comenzar conversión"
low 13                    ' finaliza el comando
klik:                     ' espera a que se presione el botón
  if in1=1 then klik       ' decide si el botón está alto o bajo
klik1:                    ' espera a que se libere el botón
  if in1=0 then klik1      ' decide si el botón está alto o bajo

DS1620:                   ' ☐ rótulo de subrutina del sensor DS1620
  high 13                  ' selecciona el DS1620
```

Experimento 3: Punta de Temperatura para Micro-Ambientes

3

```
    shiftout 15,14,lsbfirst,[170]      ' envía el comando "obtener datos"
    shiftin 15,14,lsbpre,[x]          ' obtiene los datos
    low 13                             ' fin del comando
    C=x/2                              ' convierte los datos en grados C
    debug ? C                          ' muestra la temperatura en la ventana debug
    xm=C                               ' subrutina morse espera datos en la variable, xm
    gosub morse                         ' a la subrutina
    pause 100

AD592:                                ' ☐ rótulo de subrutina del sensor AD592
    rctime 5,0,rct                     ' ☐ obtiene dato del AD592
    low 5                              ' ☐ descarga el capacitor
    TK = Kal/rct*10 + (Kal//rct*10/rct) ' ☐ calcula Kelvin
                                         ' ☐ y convierte a grados C
    TC = TK-273                        ' ☐ muestra el resultado (si está conectado a PC)
    debug ? TC                         ' ☐ subrutina morse espera datos en la variable,xm
    xm=TC                              ' ☐ pausa para separar lecturas
    pause 500                          ' ☐ a la subrutina
    gosub morse

goto klik                             ' vuelve al inicio

morse:                                ' emite el byte xm en código morse
    for j=1 to 0                       ' emite dos dígitos, decenas primero
        mc = xm dig j                  ' toma el dígito (j+1)
        mc = %11110000011111 >> mc    ' fija el patrón para el código morse
        for i=4 to 0                   ' 5 dits y dahs
            freqout 0,dit2*mc.bit0(i)+dit,1900 ' emite el patrón de bits de mc
            pause dit                  ' silencio corto
        next i                         ' next i, fin de los cinco dit o dah
        pause dah                      ' silencio entre dígitos
    next j                             ' next j, fin de los dígitos
    return                             ' vuelve a la subrutina que lo llamó
end
```

Si el código Morse se vuelve una molestia para usted o para sus compañeros, simplemente desconecte el cable de P1 para apagar el piezoeléctrico, o ponga un apóstrofe delante de `gosub morse`, convirtiendo la instrucción en un comentario no ejecutable.

Experimento 3: Punta de Temperatura para Micro-Ambientes

3

Ejecute el programa con la punta AD592 en contacto directo con el DS1620 en la Plaqueta de Educación (y a la sombra). Si tiene un poco, puede usar grasa siliconada (conductor térmico pero no eléctrico), entre los dos para mejorar el contacto. Asegúrese de que las lecturas sean constantes, y anote los valores en grados Celsius:

DS1620 : _____

AD592 : _____

Los valores deben ser muy cercanos. Ya calibró el AD592 con el hielo, y la hoja de datos del DS1620 especifica que la lectura tiene una precisión de 0.5 grados.

Guarde esta versión del programa en el disco, siguiendo las indicaciones de su instructor.

Calibración Automática (Tema Avanzado)

Una característica de muchos instrumentos modernos es la calibración automática. Por ejemplo, como sabemos que el DS1620 tiene una precisión de medio grado, podríamos saltarnos la calibración con hielo, que, después de todo, requiere muchos materiales y esfuerzo para hacerlo bien. Podemos usar el DS1620, a temperatura ambiente, como patrón de calibración. Podría conectar una punta de temperatura AD592 nueva a la Plaqueta de Educación, ponerla en contacto con el DS1620, dejarlo reposar unos minutos, y presionar un botón para ingresar el valor de calibración. ¡Listo! El BASIC stamp calcula el valor correcto de calibración y lo almacena en la EEPROM por usted. El siguiente programa hace esto. El programa también le enseñará a almacenar y recuperar datos tamaño word (16 bits), en la EEPROM.

En el programa 3.4, la constante de calibración se introdujo como

```
Kal con 15300 ' USE SU PROPIA CONSTANTE DE CALIBRACIÓN
```

Pero ahora es necesario reemplazar esa línea por:

```
eKal data word 15300 ' ☐ constante a ser determinada, automáticamente  
Kal var word ' ☐ variable temporaria para manipular calibración
```

El valor eKal apunta a un lugar en la EEPROM, y el dato de ese lugar se transferirá hacia y desde la variable Kal, usando los comandos read y write del BASIC Stamp.

La constante de calibración tiene tamaño word (dos bytes, 16 bits), pero la EEPROM sólo almacena bytes. El truco es almacenar el valor word en dos bytes sucesivos de la EEPROM. Ingrese las dos líneas siguientes en el programa, antes de la instrucción Rctime.

Experimento 3: Punta de Temperatura para Micro-Ambientes

3

```
read eKal, Kal.byte0      ' ☐ lee el byte bajo de la constante de calib
read eKal+1, Kal.byte1    ' ☐ lee el byte alto de la const de calibración
    ^^^^^^-----        ' lee de la ubicación eKal, luego de eKal+1
          ^^^^^^-----    ' byte0, luego byte1, de la variable word Kal
```

Kal.byte0 es un modificador de la variable Kal que significa, "byte 0 de word Kal". Esto es como los modificadores que vio en la subrutina del código Morse, en el Experimento 2.

También modifique la subrutina clik1, de forma que si mantiene presionado el botón mucho tiempo, el programa salte a una subrutina especial de calibración. ¿Recuerda la subrutina cliklargo del Experimento 2?

```
    x=0                    ' ☐ pone en cero el contador
clik1:
    pause 100              ' ☐ pausa de 0,1 segundos
    x=x+1                  ' ☐ incrementa el valor de x
    if x>30 then calibrar   ' ☐ calibra si se presiona > 3 segundos
    if in1=0 then clik1     ' repetir hasta que libere el botón
```

Finalmente, agregue la siguiente subrutina al final del programa:

```
calibrar:
    freqout 0,5,3400        ' ☐ indica inicio de calibración
    debug "La punta debe estar en contacto",CR
    debug " con el DS1620",CR
    TK=C+273                ' temperatura Kelvin del DS1620
    Kal = TK/10*rct + (TK//10*rct+5/10) ' ☐ calcula el valor de Kal
    debug ? Kal              ' ☐ muestra el valor de Kal
    write eKal,Kal.byte0     ' ☐ guarda el byte bajo de Kal
    write eKal+1,Kal.byte1   ' ☐ guarda el byte alto de Kal
    freqout 0,5,1900        ' ☐ indica fin de calibración
    goto clik                ' ☐ vuelve al inicio
```

Pruébalo. Cuando ejecuta por primera vez el programa, la temperatura obtenida por el AD592 será incorrecta, debido al valor incorrecto de eKal. Ponga en contacto el AD592 con el DS1620. **IMPORTANTE:** haga buen contacto entre el AD592 y el DS1620, sujetándolos firmemente, y use grasa siliconada para mejorarlo. Controle que no haya fuentes de calor cercanas.

Presione el botón y observe las lecturas hasta que vea que dejan de cambiar. Luego, presione y retenga el botón hasta que escuche el sonido indicador de inicio de calibración. Cuando libere el botón, las lecturas serán correctas, comparadas con las del DS1620. La punta AD592 puede ahora, ser usada para tomar temperaturas remotas. La calibración automática es muy importante en instrumentos que leen conductividad o pH (acidez), donde los sensores necesitan recalibraciones frecuentes.

Experimento 3: Punta de Temperatura para Micro-Ambientes

3

Ponga la punta AD592 en el recipiente con hielo y agua. Debería leer un valor cercano a cero, dentro de la resolución de 1 grado, del sistema de medición de la Plaqueta de Educación. Recuerde, la subrutina de calibración depende del hecho de tener el AD592 a la misma temperatura que el DS1620. ¡No ejecute la subrutina de auto calibración mientras la punta está en el recipiente con hielo y agua!

La constante de calibración viene de la ecuación:

$$\text{constante} = (\text{temperatura verdadera en Kelvin}) * rct$$

Asumiremos que el DS1620 nos da la temperatura "verdadera". Suponga que el DS1620 está a 25 grados Celsius, 298 Kelvin. Suponga que el valor de `rct` es 591. Así,

$$\text{constante} = (298 * 591) = 176134,$$

... y el valor que se debe almacenar en la EEPROM son los primeros cinco dígitos de éste, redondeado como antes. El truco es almacenar este resultado en el BASIC Stamp sin sobrepasar 16 bits. Lleva dos pasos, describir la temperatura de referencia como...

$$298 = 29 * 10 + 8$$

o en la notación del BS2, para cualquier temperatura en Kelvin:

$$TK = (TK/10)*10 + (TK//10) \quad \text{' división entera, más resto.}$$

Y multiplicar ambos miembros por `rct`, dividiendo luego por diez (para obtener los primeros 5 dígitos del producto). Terminamos obteniendo la fórmula del programa:

$$Kal = TK/10*rct + (TK//10*rct+5/10) \quad \text{' } \square \text{ calcula y redondea Kal}$$

El 5 agregado antes de la última división, es por redondeo. Piénselo.

Las dos instrucciones `write` almacenan, el byte 0 de word `Kal` en el lugar de la memoria, `eKal`, y el byte1 de word `Kal` en el lugar siguiente. Encontramos nuevamente los modificadores `.byte0` y `.byte1` de la variable word, `Kal`. Posteriormente, el comando `read` recupera el valor de calibración, en exactamente el mismo orden. La constante de calibración permanece sin cambios en la EEPROM hasta que (1) presione nuevamente el botón de calibración, o (2) descargue otro programa (RUN) desde STAMP2.EXE en la PC.

Algunos Experimentos de Medición de la Temperatura Ambiental

Investigue las temperaturas del ambiente que lo rodea. Su instructor puede tener instrucciones específicas. Controle la temperatura en las canillas de agua fría y caliente, en el acuario, en el exterior, debajo de los árboles, bajo el suelo. Observe que es posible extender la longitud de la punta de temperatura, si lo desea, simplemente agregando más cable. ¿Busca micro ambientes? ¿Dónde están las fuentes de calor que causan las variaciones de temperatura en los micro ambientes?

Estos son algunos de los experimentos que puede realizar. Sólo ejemplifican cómo puede ser usada una punta de temperatura, para medir más que temperatura.

1) Medir la temperatura del aire en la sombra.

Temperatura con sensor seco:

Luego enrolle un papel absorbente, o un trapo, alrededor del sensor de temperatura y reténgalo en su lugar con bandas elásticas o alambre. Intente hacer la envoltura ajustada y compacta, no muy gruesa. Humedezca la punta. Observe la temperatura a medida que hace circular aire a través del sensor, o hace girar la punta sosteniéndola desde el cable. Bajarán a un valor estable rápidamente si realizó la envoltura correctamente. ¿Cuál es la temperatura final?

Temperatura con sensor mojado:

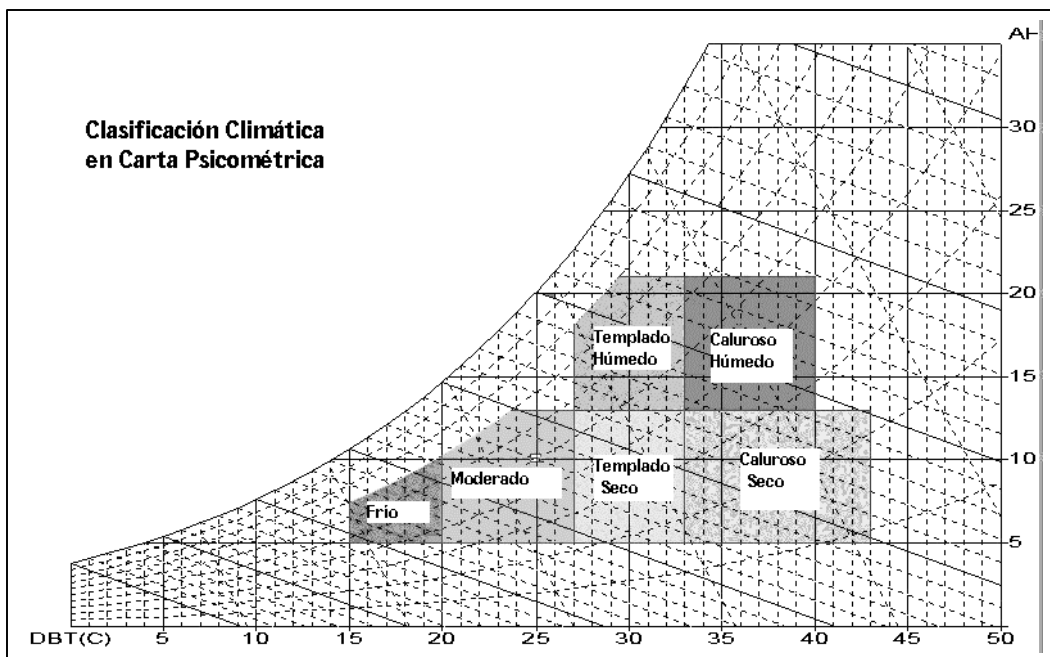
Es de esperar una reducción de 4 o 5 grados Celsius, con el sensor mojado, en una habitación con una humedad relativa del 50 %. Todos saben que un objeto mojado se enfría más rápido si hay brisa. El enfriamiento es mayor en aire seco. Depende principalmente de la humedad relativa del aire, y la velocidad del viento. Con vientos muy fuertes depende solamente de la humedad relativa. Un instrumento para medir la humedad, que usa un termómetro mojado y otro seco se llama psicrómetro (del griego, psychros, frío). La carta psicrométrica entrega el valor de la humedad ambiente como función de las temperaturas de sensor mojado y sensor seco. Pruebe esto en el exterior y en el interior. Si le interesa, se muestra un ejemplo de la carta psicrométrica en la Figura 3.10.

Figura 3.10: Ejemplo de Carta Psicrométrica

- El eje "y", sobre la derecha es la humedad relativa.
- El eje "x", es la temperatura de sensor seco (C)

Esta carta fue diseñada por la Hong Kong University para clasificar la habitabilidad de las zonas.

3



2) Corrimiento del punto de fusión en agua salada.

Ya usó un recipiente con hielo y agua para calibrar su punta de temperatura. ¿Qué sucede si le agrega sal de mesa a esta mezcla? Piense en la relación entre el agua salada y los fabricantes de helados, calles con hielo y icebergs. Puede hacer experimentos cuantitativos, variando la cantidad de sal de la mezcla, o probando con diferentes tipos de sal. Haga los experimentos en un recipiente bien aislado para obtener mejores resultados.

3) Intensidad de radiación solar con termómetros blanco y negro.

Envuelva el sensor con papel de aluminio, enroscado en la punta, de forma de poder sacarlo y ponerlo. Expóngalo al sol, reparado del viento, dentro de una jarra plástica o de vidrio. Tome la lectura cuando se estabilice

Lectura con aluminio:

Quite la envoltura de aluminio. Compare las lecturas con y sin el aluminio.

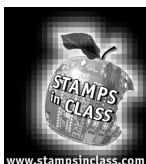
Lectura con sensor negro (sin aluminio) :

Todo el mundo sabe que los objetos oscuros se calientan al sol. Un dispositivo que mide la radiación mediante la diferencia de temperatura entre una superficie negra y una blanca se llama pirómetro "blanco y negro". (Pyr del griego "fuego").

4) Velocidad del viento con un anemómetro de punta caliente.

Haga que el sensor de temperatura negro, se caliente al sol. Luego soplelo o revoléelo sujetándolo por el cable. Todos saben que los objetos calientes se enfrían con la brisa. Esto muestra que la temperatura así obtenida, puede usarse para medir la velocidad del viento. "Anemómetros de cable caliente" usan un cable de platino, tanto como elemento sensor (su resistencia cambia con la temperatura), así como elemento calefactor (la corriente eléctrica que lo atraviesa, genera calor).

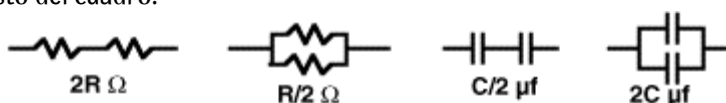
Cambio en la lectura:



¡Desafío!

3

- Arme el circuito de la figura 3.3 (o 3.4), pero use un resistor de 100K y un capacitor de 0.22 μF . Mida el valor τ_{CT} , e insértelo en la celda central de la tabla de abajo. Usted tiene dos capacitores de 0.22 μF en su kit, y dos resistores de 100K. Haciendo conexiones serie-paralelo con esos componentes, puede completar el resto del cuadro.

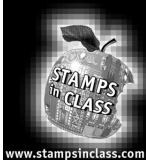


Rctime vs. R y C	50K Ω	100K Ω	200K Ω
0.11 μF			
0.22 μF			
0.44 μF			

- Modifique el programa 3.3 para que muestre los resultados en grados Fahrenheit y grados Rankine, en lugar de grados Celsius y Kelvin. (Rankine Kelvin 1.8.). No convierta Kelvin en grados Rankine con la ecuación. Calcule una nueva constante para Rankine constante/ τ_{CT} .
- Algunas veces el código Morse se vuelve molesto. Puede desconectar el cable de P0 para apagarlo. Pero el desafío en este caso, es encontrar la forma de apagarlo por software. Piense una forma de usar el botón para encender y apagar el sonido.
- Conecte un capacitor de 0.1 μF y un resistor de 100 ohm a P10, como en la figura 3.4, con la punta de continuidad. También conecte un LED y un resistor a P8, de forma que el BASIC Stamp pueda encenderlo y apagarlo. Luego (a) haga un programa que encienda el led cuando la punta se sumerge en agua (b) haga un programa que realice una pausa de 10 segundos, luego revise las entradas y titile el led si la punta estuvo sumergida en agua en algún momento de la pausa de 10 segundos. Luego recargue el capacitor, haga el pin una entrada, y regrese a la pausa y (c) Modifique el programa para que cuente cuántas veces se sumergió la punta en el agua, una posibilidad por pausa. Este programa muestra cómo el capacitor ayuda a monitorear instrumentos como pluviómetros y contadores de tráfico (interruptores que se activan brevemente, en forma impredecible). El capacitor es usado como una "memoria", recordándole al BASIC Stamp la información, cuando éste controla el pin de entrada.

Vocabulario

micro-environment micro ambiente	capacitor voltage level nivel de tensión de capacitor	debug rep " " n
analog vs digital analógico/digital	threshold umbral	integer division, remainder división entera, resto
analog as analogy analógico como analogía	rate of charging velocidad de carga	accuracy, resolution precisión, resolución
transducer sensor probe punta de sensor transductor	RCtime	automatic calibration calibración automática
sourcing current suministra corriente	Branch bifurca	wet bulb, dry bulb sensor seco, sensor mojado
sinking current absorbe corriente	calibration bath recipiente de calibración	psychrometer sicrómetro
273.14 Kelvin	ice bath recipiente con hielo y agua	



Experimento 4: Luz en la Tierra y Adquisición de Datos

El tema del experimento Luz en la Tierra y Adquisición de Datos es: "la luz, y su importancia para todo lo que se encuentra bajo el sol". Para demostrar esto, construiremos un fotómetro/data logger. Las actividades que realizaremos en este experimento son: (1) Fotodiodo sensor de luz usando `Rctime`, y escalas de magnitud de la intensidad

(2) Termómetro y fotómetro combinado (3) Un data logger (con pulsador), para temperatura y luz y (4) Experimentos usando este data logger.

"Hágase la luz"

Si la temperatura es la variable número uno en las mediciones ambientales, entonces la luz debería ser la número cero. El sol provee la energía para la mayoría de los procesos físicos y climáticos de la tierra. ¿dónde estaríamos sin la fotosíntesis? Los humanos han tomado mediciones del sol desde la prehistoria. En Stonehenge, en el Caracol de Chichén Itzá, y por todo el mundo, nuestros ancestros tomaron mediciones del ciclo solar de las estaciones en relación a la agricultura, y la vida espiritual y temporal.

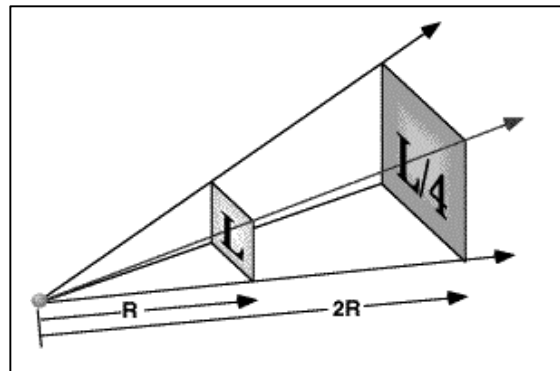
La temperatura es una variable relativamente simple comparada con la luz. La luz viene en un espectro de colores, visible e invisible, que se extiende hasta límites difusos de longitudes de onda. Tiene polarización y dirección. Muchos aspectos de la luz tienen significado especial. Ciertas longitudes de onda son responsables del bronceado otras son especiales para la maduración de los frutos. Hay patrones sutiles de luz. Por ejemplo, las abejas pueden ver patrones azul oscuro en las flores, que el ojo humano no puede percibir, y la visión del colibrí se extiende hacia el infrarrojo, que no llegamos a ver. La luz es importante para nosotros, en un gran rango de intensidades, desde energía solar para generar electricidad y calor, hasta la bioluminiscencia de las criaturas del fondo del océano.

Al igual que la temperatura, la luz se usa a menudo para medir otras cosas. Por ejemplo, instrumentos para detectar la calidad del aire y CO_2 , se basan en láser o en que los gases absorben longitudes de onda características. Los astrónomos usan el espectro para deducir la composición química de las estrellas y los gases interestelares. En el otro extremo de la escala de tamaño, la luz es usada para probar procesos químicos del ADN, y mecanismos de la célula viva. En el campo práctico, la luz es usada en detectores de movimiento, en indicadores, y por supuesto en iluminación, que es una especialidad de la ingeniería.

Una ley fundamental dice que la intensidad de luz, al alejarse del origen, disminuye con el cuadrado de la distancia. Es decir, al doble de distancia de una lámpara (o del sol), la intensidad de luz disminuirá a $1/4$ de su valor. La misma cantidad de energía se distribuye en un área 4 veces mayor. Usando el medidor de luz (fotómetro) que construirá en esta lección, tendrá una herramienta para investigar esa ley, así como también explorar la variación de luz en su ambiente. Este concepto se muestra en la Figura 4.1.

Figura 4.1: Atenuación de la luz

Al doble de distancia de una lámpara (o del sol), la intensidad de luz disminuirá a $1/4$ de su valor. Investigaremos esta ley con nuestro fotómetro microcontrolado.



4



Partes Requeridas

Para este experimento dejaremos los componentes instalados en el Experimento 3. Los siguientes componentes son necesarios para el Experimento 4:

- (1) fotodiodo (Photonic Detectors C113 o V113)
- (4) resistor de 100 ohm (marrón negro marrón)
- (2) capacitor de poliéster de 0.01 uF (103)
- (1) capacitor de poliéster de 0.22 uF (224)
- (2) capacitor cerámico de 100 pF (101)
- (1) led rojo
- (1) led verde
- (1) batería de 9 volts
- (1) lámpara spot de 50 wat R20 (para los experimentos)



¡Constrúyalo!

4

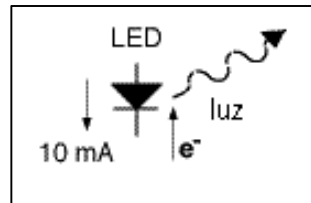
Fotodiodo como Transductor de Luz

En "¿Qué es un Microcontrolador?" usó un fotoresistor (también llamado elemento fotoconductor) para detectar personas imaginarias acercándose a la puerta de un supermercado, donde un servo motor abre la puerta automáticamente para dejarlos entrar. Un fotógrafo de vida silvestre o un biólogo puede usar un dispositivo similar para detectar un animal evasivo, y disparar la cámara. En esta lección, usaremos un tipo diferente de fotodetector, un fotodiodo. Un fotodiodo genera una corriente eléctrica (en amperes), que es directamente proporcional a la intensidad de luz. Esta característica lo hace ideal para mediciones cuantitativas.

Usted ya conoce al diodo emisor de luz, que convierte la corriente eléctrica en luz. El led emite luz cuando circula corriente en el sentido de su flecha. Ya debe saber que en realidad los electrones (cargas negativas, e^-) en realidad circulan en la dirección contraria. Pero por un accidente histórico de interpretación, tomamos a la corriente como si estuviera formada por cargas positivas. Resumiendo, el LED emite luz cuando fluye corriente (positiva) en la dirección de la flecha.

Figura 4.2: LED

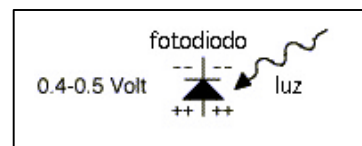
El LED emite luz cuando la corriente (positiva) circula en la dirección de la flecha.



También funciona a la inversa, como muchos transductores. La luz que llega al diodo produce electricidad. Si conecta un voltímetro a un diodo expuesto a la luz, medirá una fracción de un volt, con la polaridad indicada. Los electrones se acumulan del lado del cátodo.

Figura 4.3: Fotodiodo

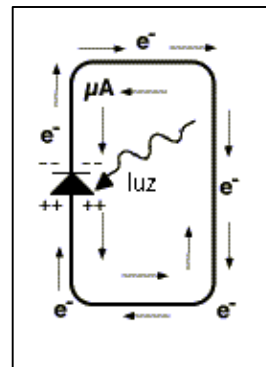
Un fotodiodo produce electricidad cuando es iluminado.



Si cortocircuita el diodo, circulará una corriente. La cantidad de corriente es proporcional a la intensidad de la luz. Esto es fundamental. La luz es la que genera los cambios en la corriente. Los electrones circulan por un circuito externo. Observe que fluyen por el circuito en sentido horario. La corriente convencional (cargas positivas) fluye en la dirección opuesta, en contra de la flecha del diodo. Esta se llama fotocorriente, y es una corriente inversa. Compare esto con la corriente que enciende al LED. Esto de la flecha puede ser confuso, pero en la electrónica, todo es relativo. En este circuito, la tensión sobre el diodo es cero (está cortocircuitado).

Figura 4.4: Fotocorriente

La fotocorriente se da cuando la corriente convencional (cargas positivas) fluye en sentido inverso, en contra de la dirección de la flecha del diodo.



La sensibilidad a la luz es una propiedad fundamental de diodos y transistores. En muchos casos, podría ser un efecto indeseable. Los transistores y circuitos integrados son normalmente recubiertos con plástico, cerámica, o encapsulados metálicos, y una de las razones para hacerlo, es precisamente alejarlo de la luz que afecta en gran medida su rendimiento. Los fotodiodos se fabrican especialmente, aumentando la sensibilidad a la luz. Mire el fotodiodo C113 de su kit. Tiene un encapsulado traslúcido, y en su interior puede ver un cubito de silicio, con conexiones en los lados. Las cargas eléctricas se generan en la cara superior e inferior. La diferencia entre panel solar y fotodiodo radica principalmente en la diferencia de superficies. Los paneles solares cubren grandes áreas, hasta metros cuadrados, para interceptar mucha luz y producir mucha corriente y potencia, medidas en amperes y wats. El fotodiodo está hecho de materiales especialmente purificados, para lograr precisión en medición, no para producir energía.

Lo que hace muy útiles a los fotodiodos en la medición, es que una ecuación muy simple gobierna su comportamiento como transductor:

$$i = \text{constante} \quad (\text{intensidad de luz})$$

Experimento 4: Luz en la Tierra y Adquisición de Datos

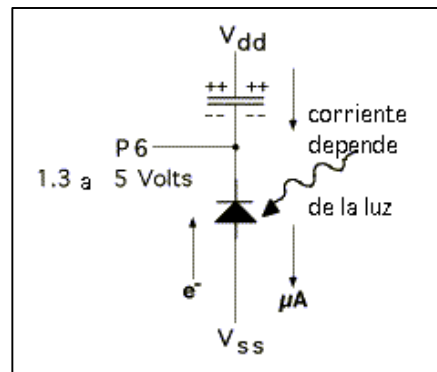
El sensor es lineal. Significa que si el nivel de luz aumenta, digamos, en un factor de 1000, entonces la corriente por el diodo, aumenta con el mismo factor. En el caso del fotodiodo, esto es cierto a través de varios órdenes de magnitud, sobre varias potencias de diez. Esta característica es la que lo hace tan útil en la medición.

La ecuación también es cierta cuando un diodo es conectado en inversa, como en la Figura 4.5. Con la misma intensidad de luz, la corriente que circula en este circuito será la misma que en corto circuito, como en la Figura 4.4. La corriente del fotodiodo carga el capacitor. La carga se acumula en el capacitor como se muestra, y la tensión aumenta gradualmente. El programa del BASIC Stamp II medirá el tiempo que tarda la tensión de P6 en caer de 5 V a 1,3 V, como se muestra en la Figura 4.5.

4

Figura 4.5: Circuito del Fotodiodo

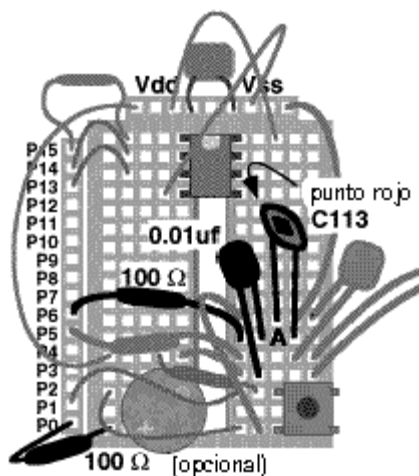
Con este circuito el BASIC Stamp puede medir el tiempo que tarda en caer de 5 V a 1,3 V.



Entonces ¡hágalo! La distribución se muestra en la Figura 4.6, y el circuito en la Figura 4.7.

Figura 4.6 Transductor de Luz (Fotodiodo)

- capacitor de poliéster de 0.01 μF (rotulado 103) del nodo Vdd (el mismo del cable rojo del AD592), al nodo A.
- ánodo del fotodiodo (punto rojo), al nodo Vss.
- Cátodo del fotodiodo al nodo A.
- resistor de 100 ohm de nodo A a P6
- opcional: si quiere bajar el volumen del piezoeléctrico, reemplace el cable de P0 con un resistor de 100 ohm.



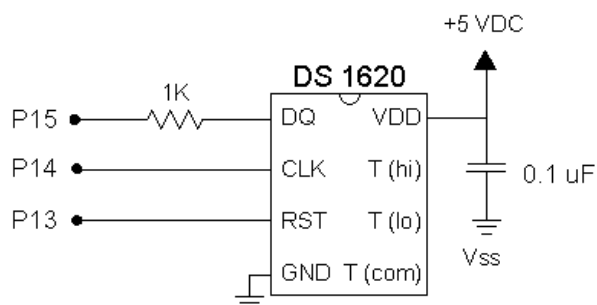
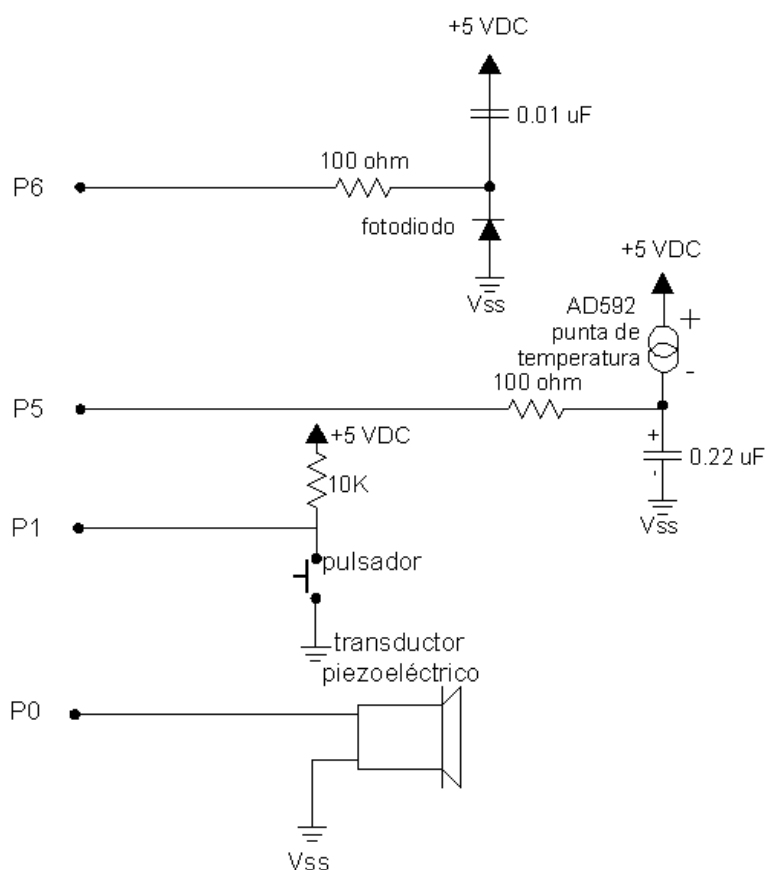


Figure 4.7: Circuito Eléctrico del Transductor de Luz (Fotodiodo)

Un resistor opcional puede ser instalado entre el piezoeléctrico y PO del BASIC Stamp, para bajar el volumen.



¿Qué es Un...

Nodo:

Nodo es el punto de un circuito donde se conectan dos o más componentes. Cada fila de 5 huecos en la Plaqueta de Educación es un nodo, debido a que todos los huecos están interconectados. Un nodo en la figura 4.6 es el punto donde se conectan el diodo, el capacitor y P6.

Ingrese el siguiente programa:

```
' mediciones Ambientales programa 4.1
' Sonido dependiente de los niveles de luz
rct var word          ' variable para Rctime
high 6                ' descarga el capacitor
bucle:
  Rctime 6,1,rct      ' tiempo para llegar a 1,3 V
  high 6              ' descarga el capacitor
  if rct=0 then bucle  ' no suena si Rctime desborda
  freqout 0,1,3400    ' emite sonido corto
goto bucle
```

Después de descargar el programa en el BASIC Stamp, exponga el sensor a variadas condiciones de luz, desde luces tenues, a lámparas y luz del sol, si está accesible. Lea los dos párrafos siguientes antes de continuar.

Los niveles de luz pueden variar ampliamente en el ambiente natural. Nuestros ojos tienen una sorprendente capacidad de acomodarse a la luz brillante así como a la penumbra. Sensitividad es la cantidad de luz necesaria para obtener una respuesta. Algunas veces necesitamos un ajuste en la sensibilidad, como un interruptor para sensibilidad "alta" y "baja". Cámaras y ojos tienen un iris que se abre o cierra para ajustar la cantidad de luz entrante, logrando extender el rango de sensibilidad.

Observe que el programa regresa al inicio sin emitir sonido, si el valor de `rct` es igual a cero. Es el extremo oscuro del rango. Puede que tenga que cubrir el sensor con una caja o algo similar, para lograr la oscuridad necesaria para ver este efecto. El capacitor tarda demasiado en descargarse y el comando `Rctime` no ve ninguna transición de 1 a 0 en P6 dentro del límite de 0.13107 segundos.

- Para lograr mejor respuesta con poca luz (más sensible), reemplace el capacitor de 0.01 uf con 100 pf (picofaradios, rotulado 101).

En el otro extremo, con luz brillante, los sonidos se amontonan, de forma que no se distinguen las diferencias.

- Para lograr mejor respuesta con mucha luz (menos sensible), cambie por un capacitor de 0.22 uf (rotulado 224), o ponga una capa de papel sobre el sensor, sujetándolo con una banda elástica, a modo de filtro, para disminuir la sensibilidad.

Ahora, alimente la Plaqueta de Educación con baterías y llévela por los alrededores. (Asegúrese que la batería esté en buen estado, caso contrario se obtienen malos resultados). Cambie los capacitores de 100pF y 0.22 uF para cambiar la sensibilidad, cuando lo crea necesario.

Experimento 4: Luz en la Tierra y Adquisición de Datos

- Apunte el sensor hacia arriba y abajo, para medir luz directa y reflejada y patrones de luz y sombra.
- Colóquelo cerca de los objetos sobre el escritorio, o cerca de los cambios de color en las cortina o en la ropa, o controle el parpadeo del monitor o del televisor.
- Mida lugares relativamente oscuros (usando el capacitor de 100pF), así como en el exterior al sol, si es posible (usando el capacitor de 0.22 uF).

4

¡Diviértase con él!

¿Qué es la luz?

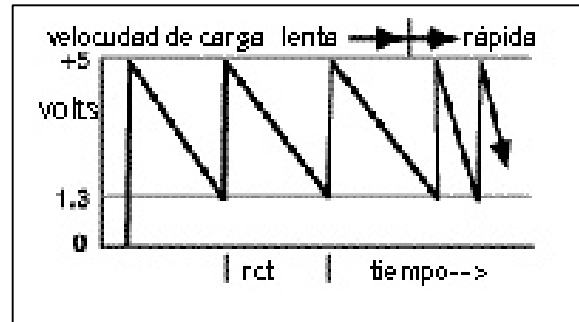
Intensidad de la luz:

- Un **Pirómetro** cuantifica la intensidad de la luz midiendo la energía luminica que golpea una superficie por unidad de tiempo. Esta es la medición correcta para quien diseña sistemas de paneles solares o un calentador de agua solar, o si es un arquitecto que calcula la eficiencia de energía de un edificio. Este tipo de intensidad luminica se mide en wats por metro cuadrado. La energía solar que golpea la superficie terrestre en un día despejado, está por encima de 1000 wats por metro cuadrado, o 75 wats por pie cuadrado. La energía solar antes de entrar a la atmósfera es de 1400 wats por metro cuadrado, que a menudo se da en otra unidad como 2 Langley por minuto. (1 Langley = 1 caloría por centímetro cuadrado). Algunas veces estamos interesados solamente en la energía de ciertas longitudes de onda. Por ejemplo, la luz ultravioleta de aproximadamente 300 a 320 nanómetros causa el bronceado, no sólo de la gente, sino también en otras formas vivientes, como el coral en el océano. Esta luz UV puede ser separada y medida. Llega a menos del 0.1% del total, menos de 1 wat por metro cuadrado. Pero es un 0.1% muy significativo. Más y más UV están llegando a la tierra, a medida que crece el agujero de ozono, aparentemente a raíz de la actividad humana, por el uso de ciertos químicos CFC.
- Un **Fotómetro** cuantifica la intensidad de luz como nuestros ojos. Esto se aplica en la ingeniería de iluminación y la fisiología. ¿Cómo ven las lechuzas? Nuestros ojos son más sensibles a la luz brillante en el amarillo-verde, disminuyendo la sensibilidad en el rojo y en el azul. La medición de la intensidad luminica sigue siendo unidad de energía por unidad de área por unidad de tiempo, pero ahora sólo incluye la energía de las longitudes de onda del espectro visible. La unidad internacional de medición es el lux. La intensidad luminica mirando al sol es de 110.000 lux, pero por supuesto, mirar directo al sol no es algo que acostumbremos hacer. Es demasiado intenso. En contraste, una lámpara de 100 wat, vista desde una distancia de un metro, tiene aproximadamente 100 lux. Esto también se percibe como muy brillante. La iluminación normal de un ambiente se mide en decenas de lux. Hay un rango muy grande de valores sobre el que los sensores, incluidos nuestros ojos, deben trabajar, aproximadamente de 7 u 8 órdenes de magnitud. Esto no es nada comparado con el rango de los niveles de luz que llegan del espacio, donde se detectan más de 20 órdenes de magnitud.
- Un **medidor PAR** cuantifica la intensidad de la luz según afecta el crecimiento de las plantas. Esto es de gran interés para los agricultores, acuaristas y botánicos. La fotosíntesis se produce en una banda especial de longitudes de onda, llamada espectro de acción fotosintética. PAR viene de Photosynthetically Active Radiation (Radiación Activa para Fotosíntesis). Mediciones de PAR le permiten a los botánicos estimar el máximo crecimiento posible de una planta, si la luz era el factor limitante. La luz viene en paquetes de energía, llamados fotones, y cada unidad de fotosíntesis toma un fotón de luz. Las unidades de PAR son micromoles de fotones por metro cuadrado por segundo. A pleno sol se obtienen aproximadamente 2000 moles por metro cuadrado por segundo. La biología trata mucho con las adaptaciones a los niveles de luz.
- Un **Espectrómetro** es la más versátil de todas. Le dice cuanta energía luminosa hay en cada longitud de onda del espectro. Por el contrario, la fotocorriente de su fotodiodo es debida al efecto total de muchas longitudes de onda diferentes. El espectrómetro puede ser usado para calibrar y caracterizar casi cualquier otro instrumento de medición de luz, pero, no es necesario decirlo, es un instrumento mucho más complicado y caro. Una versión económica de éste se llama colorímetro, que se puede encontrar en casa fotográficas, o en pinturerías para lograr colores.

Hablemos más sobre el programa del BASIC Stamp. Un pin del capacitor es conectado a Vdd (5) (en lugar de Vss, como en el circuito sensor de temperatura del AD592). En este caso, el programa comienza con HIGH 6, para descargar el capacitor. Esto puede parecer raro, hacer alto el pin para descargar el capacitor, pero observe que se dice que se "descarga" un capacitor cuando la diferencia de potencial entre los terminales (pines) es cero. Un pin del capacitor está conectado a 5 volts, así que HIGH 6 pone los dos pines a 5 volts, descargándolo. La Figura 4.8 muestra la tensión en función del tiempo en P6.

Figura 4.8: Descarga Resistor/Capacitor

La corriente que circula por el fotodiodo carga gradualmente el capacitor, disminuyendo la tensión en P6 hasta llegar al umbral de 1,3 V. El comando RCtime mantiene a P6 como entrada durante ese tiempo. Tan pronto como RCtime detecta el nivel de 1,3 volt, el programa pasa a HIGH 6 y rápidamente descarga el capacitor, y la tensión en P6 regresa rápidamente a 5 volts.



Mire cuidadosamente el comando `RCtime` del Programa 4.1. El segundo parámetro es 1. Ese parámetro era 0 en los programas del Experimento 3, con el transductor de temperatura AD592. El 1 le ordena al comando `RCtime` cronometrar el tiempo transcurrido mientras P6 sea igual a uno, deteniéndose cuando P6 haga la transición a cero. Pruebe de cambiar el segundo parámetro de 1 a 0. ¿No funciona, no? Es decir, ¿es sensible a la luz? ¿Escucha un tono alto, un tono bajo, o ningún tono? Para encontrar errores, es bueno plantearse este tipo de preguntas.

El BASIC Stamp acepta ambas formas de comando, con el segundo parámetro 0 ó 1, debido a que hay situaciones donde una será mejor que la otra, o será la única alternativa. Por ejemplo, el sensor de temperatura AD592 trabaja bien en el circuito del Experimento 3, pero no funcionará en este circuito del Experimento 4. Esto es debido a las limitaciones de tensión del AD592. No vamos a detallar todas las ventajas y desventajas de una configuración sobre otra, pero es bueno estar al tanto de estas opciones, a la hora de diseñar sus propios proyectos.

Observe que el tono del sonido aumenta (más agudo) cuando aumenta el brillo de la luz. ¿Cómo es esto posible, si el tiempo para descargar el capacitor, `rct`, se reduce al aumentar el brillo? Asegúrese de entender que con valores menores de `rct`, el bucle se repetirá más rápido, lo que hace un tono más agudo.

Fotodiodo y BASIC Stamp como un Medidor de Luz Digital

El programa anterior era un medidor analógico, debido a que la frecuencia de la salida era una analogía de la luz de entrada. Ambos aumentan y disminuyen de la misma manera. Los medidores analógicos son muy buenos para transmitir información directamente a nuestros sentidos. Mire los números de la ventana debug. Esta es una conexión digital. Modifique el programa como sigue (las líneas con ☐ deben ser agregadas o cambiadas):

```
' Mediciones Ambientales programa 4.2
' Niveles numéricos de luz del fotodiodo
rct var word           ' variable para Rctime
luz var word           ' ☐ variable para intensidad de luz
high 6                 ' descarga el capacitor
bucle:
  Rctime 6,1,rct       ' tiempo para llegar a 1,3 V
  high 6               ' descarga el capacitor
  luz=65535/rct        ' ☐ luz=constante/rct
  debug dec rct,tab,dec luz,tab, bin luz,cr ' ☐ muestra valores en pantalla
  pause 400            ' ☐ pausa para dar tiempo a leer
goto bucle
```

La constante 65535 es arbitraria. Lo importante es que la luz es proporcional a $1/rct$. El valor exacto de la constante de proporcionalidad será determinado cuando calibremos el sensor con una fuente de luz de intensidad conocida, como el sol, o una lámpara común.

Teoría:

Para los que estén interesados, la teoría es la misma del Experimento 3. La tensión sobre el capacitor debe disminuir 3,7 volts (ver Figura 4.8), en lugar de 1,3 volts (ver Figura 3.7). La fórmula es:

$$2 \cdot rct = C \cdot 3.700.000 / i$$

(2 rct) está en microsegundos, C en microfaradios, e i en microamperes. Esto muestra la relación inversa teórica entre la fotocorriente, i, y la variable rct que se obtiene del comando Rctime. La fotocorriente es proporcional a la intensidad de luz que alcanza al fotodiodo. Sin embargo, a diferencia de la punta de temperatura AD595, donde la constante de calibración es 1 uA/K, no hay una igualdad exacta entre unidades estándar de luz y fotocorriente. Pero no importa. Las constantes pueden ser agrupadas en una que puede ser determinada en la calibración:

$$rct = \text{constante} / (\text{nivel de luz}) \quad \text{o} \quad \text{nivel de luz} = \text{constante} / rct$$

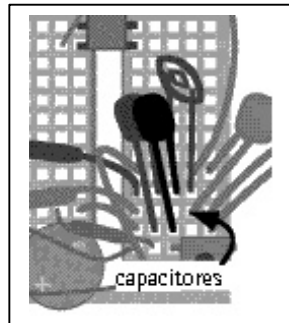
Cuando ejecute el programa, observe cómo los números de la segunda columna de la pantalla aumentan al aumentar el brillo. Y los números de la primera columna disminuyen su valor, debido a que el capacitor se carga más rápido, debido a una mayor fotocorriente.

La última columna se incluyó, no para que pueda ver el valor binario del nivel de luz, (BASIC Stamp Manual Version 1.9, página 256, en inglés), sino debido a que la longitud del número binario, de 1 a 15 dígitos binarios, es proporcional al logaritmo del nivel de luz. Un dígito se agrega cada vez que se duplica la intensidad de la luz. Pruebe tomando mediciones desde muy oscuro a muy brillante, para entender lo que queremos decir. Los logaritmos son útiles para trabajar con fenómenos que varían sobre un rango muy amplio. Como otro ejemplo, el vúmetro de un sistema de sonido estéreo muestra un gráfico logarítmico del nivel de sonido. Nuestros oídos, al igual que nuestros ojos, se pueden acomodar sobre un gran rango de niveles de sonido. En términos técnicos, la longitud del número binario muestra la parte entera del logaritmo base 2. Lo mismo es cierto para el número decimal de la segunda columna, que agrega un dígito cada vez que el nivel de luz se multiplica por 10, pero es más difícil de percibir. Puede probar reemplazando `bin luz` en la instrucción `debug` por `rep 42\ ncd luz`. Esto imprime una barra de asteriscos, en lugar del número binario real. El operador `ncd` (BASIC Stamp Manual Version 1.9, página 237, en inglés) es lo más cercano que tiene el BASIC Stamp a una función logarítmica.

Miremos el efecto del capacitor en la lectura. Con un capacitor de 0.01 μF en el circuito, coloque la Plaqueta de Educación donde la luz sea constante, y anote la lectura. Ahora encuentre el segundo capacitor de 0.01 μF en su kit, e instálelo en la Plaqueta de Educación, en paralelo con el primero, como en la Figura 4.9. Ahora (con el mismo nivel de luz que antes) observe las lecturas `rct` y nivel de luz. Las lecturas de `rct` deberían ser de aproximadamente el doble que antes, y las lecturas de nivel de luz deberían ser aproximadamente 1/2 de lo que eran. Esto es debido a que el capacitor entra en el cálculo de la constante de calibración, como se muestra en el recuadro de teoría. Si la corriente del fotodiodo es constante, entonces al duplicar el valor del capacitor también se duplica el tiempo necesario para cargarlo.

Figura 4.9: Paralelo de Capacitores

Agregando el segundo capacitor en el circuito en paralelo, la lectura de `rct` duplicará su valor anterior. El valor del capacitor entra en el cálculo de la constante de calibración.



Los capacitores que estamos usando son los llamados capacitores de "poliéster". Son convenientes por su estabilidad. Los cambios de temperatura no afectan su capacidad. Necesitamos un capacitor tan estable, para que el valor de R_{Ctime} sólo dependa de la corriente del fotosensor.

Ahora quite ambos capacitores, de forma que no quede ninguno en el circuito del fotodiodo. (Asegúrese de quitar el capacitor del fotodiodo, no el capacitor de temperatura). ¡Aún funciona! Los números que aparecen no tienen significado, pero se verá como si hubiese un capacitor en el circuito. Será sensible a niveles muy bajos de luz. Pruebe en la oscuridad. En realidad, hay un capacitor en el circuito. La compuerta de entrada del microcontrolador PIC 16C57 del BASIC Stamp tiene una capacidad interna de aproximadamente 50pf (picofaradios). Además, el cableado interno de la protoboard agrega capacidad. Recuerde, existe capacidad, intencional o no, en cualquier cable que esté cerca de otro. Esto se llama capacidad parásita, debido a que no se puso intencionalmente en el circuito. La combinación de la capacidad de entrada del PIC con la de la protoboard, suman un equivalente de aproximadamente 250 pF de capacidad parásita. Ponga un capacitor de 100 pF donde estaba originalmente el de 0.01 uF. Las lecturas no cambiarán mucho. La capacidad cambió de aproximadamente 250 a 350 pf, y no de cero a 100 pF como sería de esperar. Ponga un segundo capacitor de 100pf en paralelo con el primero. Las lecturas no cambiarán con un factor de 2, debido a que la capacidad ha cambiado de aproximadamente 350 a 450 pF, y no de 100 a 200 pF. A menudo cuando las cosas no funcionan como era de esperarse, es debido a elementos parásitos que no se están teniendo en cuenta.

Ahora quite los dos capacitores de 100 pF, y reubique el capacitor de 0.01 uF en su medidor de luz de la Plaqueta de Educación. Sostenga el medidor de luz cerca de una fuente de luz brillante. Si tiene, use una lámpara spot de 50 wat R20 (del tipo usada en reflectores). Si no tiene este tipo de lámpara, puede usar una común de 100 wat. La intensidad lograda con esta fuente de luz, apuntando al centro del haz, a un metro de distancia es de 425 lux (40 candelas). Escriba la lectura de la segunda columna de la pantalla.

luz (lectura directa segunda columna)

Esto es fácil de decir. Puede encontrar que las lecturas de la computadora fluctúan, subiendo y bajando bastante, haciendo difícil decidir cuál es la "lectura". Estas fluctuaciones vienen de un par de fuentes diferentes. Una de ellas es que el nivel de luz está realmente fluctuando, muy rápidamente, más rápido de lo que puede percibir. La intensidad de la lámpara depende de la tensión de la línea de alimentación, por eso debemos enfatizar que la intensidad es aproximadamente 425 lux a un metro. La línea de tensión de CA que alimenta la lámpara va de cero a 170 volts (ó 311 volts), 120 (ó 100) veces por segundo. A medida que esto ocurre, la intensidad varía. El filamento de la lámpara sigue brillando, debido a su inercia térmica, pero la salida varía en una escala de tiempo de 1/120 de segundo, aproximadamente 10 milisegundos. Con estas variaciones aparecen el fotodiodo y el BASIC Stamp, midiendo la luz en menos de un milisegundo. Algunas veces la encuentran en un máximo y otras en un mínimo.

Experimento 4: Luz en la Tierra y Adquisición de Datos

Hay formas de promediar todo esto, pero por ahora, tómelo como una lección. Estime su propio valor medio. Primero busque un valor mínimo, y luego el máximo. Luego tome el promedio.

¿mínimo?
¿máximo?
¿promedio?

4

Ahora, vea si puede encontrar un factor de escala de forma que el programa muestre el valor numérico en unidades estándar de 425 lux, en lugar del valor directo en unidades arbitrarias, cuando el sensor está en posición frente a la lámpara.

(lectura directa) (factor de escala) 425 lux

Por ejemplo, si la lectura directa es 168, con una calculadora encuentre el valor del factor de escala 2.53. Ese número se encuentra despejando la ecuación:

(factor de escala) $425/168$ 2.53

Una vez calibrado, usted puede mover el medidor de luz a un área desconocida y encontrar el nivel de luz actual, en lux.

(nivel nuevo de luz en lux) (lectura directa) 2.53

El problema es que el BASIC Stamp (como la mayoría de los microcontroladores) usa matemática entera. No tiene fracciones. Bien, no es tan cierto. El BASIC Stamp II tiene un operador matemático especial $*/$, llamado multiplicador fraccionario. La trampa es que la fracción debe ser uno de estos valores específicos: 0, $1/256$, $2/256$, y así hasta $256/256$ (unidad) y continuando: $257/256$ ($1 + 1/256$), hasta $65535/256$ ($255 + 255/256$). Todas las fracciones tienen el denominador 256. El factor que va del lado derecho del $*/$ es el numerador de la fracción, y el denominador 256 es implícito. Estos son algunos ejemplos:

$Y \ X / 256$ es lo mismo que $Y \ X$, porque $256/256 = 1$
 $Y \ X / 128$ es lo mismo que $Y \ X \ 1/2$, porque $128/256 = 1/2$
 $Y \ X / 384$ es lo mismo que $Y \ X \ 3/2$, porque $384/256 = 3/2$
 $Y \ X / 647$ es lo mismo que $Y \ X \ 647/256$...

Tenemos que $647/256$ está cerca de 2.53, que es el factor de escala que necesitamos. Pruébalo:

$647/256$ 168 $(647/256)$
¿Cerca de 2.53? ¿cerca de 425?

Experimento 4: Luz en la Tierra y Adquisición de Datos

Esta es la forma de encontrar el valor que va a la derecha de $*$ /. Usted tiene la lectura directa de la luz, digamos 168. También tiene el valor conocido de la intensidad de la luz, digamos 425 lux. Entonces el factor a poner a la derecha del $*$ / es $425\ 256/168\ 647$. Use el valor que obtuvo en la medida directa, segunda columna, dos páginas atrás, en el lugar de 168:

425 256 / (su medida directa)

La ecuación para el programa es:

```
luz = 65535/rct*/647      ' calcula intensidad de izquierda a derecha
```

(debe usar su propia constante en lugar de 647). Cuando ejecute el programa con este cambio, debería aparecer 425 en la segunda columna de la pantalla debug, cuando el sensor es colocado a un metro de la fuente de calibración. Ahora, a medida que mueve el medidor por la habitación, la lectura se mostrará en unidades estándar de lux. Esta es una calibración "aproximada". Pero demuestra la idea, y cómo el operador $*$ / puede ayudar con la matemática (BASIC Stamp Manual Version 1.9, página 242, en inglés). La calibración de sensores analógicos, a menudo involucra la multiplicación de los valores directos por una fracción, así que saber usar el operador $*$ / puede ser de gran ayuda.

4

Medidor de Luz y Temperatura

¿Aún tiene la punta del sensor de temperatura conectada? Esperamos que sí. Si no, reconéctela de acuerdo a la Figura 3.8, e ingrese el siguiente programa. Por favor use sus constantes de calibración del sensor de luz, y el sensor de temperatura.

4

```
' Mediciones Ambientales programa 4.3
' intensidad de luz y temperatura
kal      con 15068      ' ☐ USE SU CONSTANTE DE CALIBRACIÓN del AD592.
lical    con 647        ' ☐ USE SU CONSTANTE DE CALIBRACIÓN del fotodiodo

rct      var word       ' variable para Rctime
luz      var word       ' variable para intensidad de luz

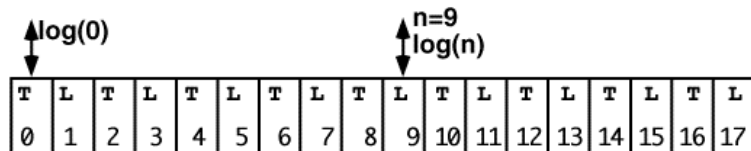
TC var word             ' ☐ para grados Celsius del AD592
low 5                   ' ☐ descarga el capacitor de temperatura
high 6                  ' ☐ descarga el capacitor de fotodiodo
bucle:
  Rctime 5,0,rct        ' ☐ lee la punta de temperatura
  low 5                  ' ☐ descarga el capacitor de temperatura
  TC=kcal/rct*10+(kal//rct*10/rct)-273      ' ☐ calcula Celsius
  Rctime 6,1,rct        ' lee el fotodiodo
  high 6                 ' descarga el capacitor de luz
  luz=65535/rct*/lical   ' ☐ calcula lux
  debug dec TC," C",tab,dec luz," lux",cr   ' ☐ muestra los valores
  pause 400              ' baja la velocidad del bucle
goto bucle
```

Ahora tiene las lecturas de temperatura (primer columna) y luz (segunda columna) en la pantalla, con unidades. ¡Qué progreso! Compare las dos instrucciones `Rctime` de temperatura y nivel de luz. Asegúrese de comprender por qué son diferentes, en relación a la forma del circuito eléctrico. Deje esto trabajando antes de pasar a la sección siguiente.

Almacenamiento de Datos de Temperatura y Luz, Usando Memoria RAM

Hagamos que el programa almacene un puñado de lecturas. Ya discutimos en lecciones anteriores por qué la adquisición de datos es importante. Es tiempo de llevarlo a la práctica. El objetivo final es almacenar datos en la memoria EEPROM del BASIC Stamp. Pero, para hacerlo simple, almacenaremos los datos en la memoria RAM. Recuerde del Experimento 1 que sólo hay 26 bytes de RAM disponibles para uso multipropósito en el BASIC Stamp II. Usaremos 18 bytes para almacenar los datos como se muestra en la Figura 4.10.

Figura 4.10: Ubicación en Memoria del Archivo de Almacenamiento (log)



La instrucción para reservar los 18 bytes en la RAM será,

```
log    var    byte(18)
```

Estos bytes son como 18 cajones en fila, numerados de 0 a 17, donde vamos a almacenar los valores de temperatura en los cajones pares y los de intensidad de luz en los impares. En el programa, nos referiremos a estos cajones usando un índice entre paréntesis:

```
log(0)=TC      ' almacena la temperatura en el primer cajón
luz=log(9)     ' recupera la intensidad de luz del décimo cajón.
luz=log(n)     ' el número de cajón es la variable, n.
               ' cuando n=9, recupera intensidad de luz del décimo cajón
```

Con una señal del pulsador, el programa tomará las lecturas de temperatura y nivel de luz, y almacenará los números en los próximos dos cajones disponibles. Cuando todos los cajones estén llenos, el programa hará un beep de protesta para indicar, "memoria completa". Cuando presione mucho tiempo el botón, los valores de los 18 bytes se mostrarán en la pantalla debug. Para borrar los datos y comenzar otra vez, todo lo que debe hacer es presionar el botón reset. Este vacía (reinicia) todos los cajones y el puntero.

Experimento 4: Luz en la Tierra y Adquisición de Datos

Modifique el programa 4.3 como sigue (líneas nuevas o modificadas marcadas con ☐):

```
' Mediciones Ambientales programa 4.4
' almacenamiento de luz y temperatura en la RAM
kal      con 15068      ' USE SU CONSTANTE DE CALIBRACIÓN del AD592
lical    con 647        ' USE SU CONSTANTE DE CALIBRACIÓN del fotodiodo

log      var byte(18)   ' ☐ 18 bytes reservados para almacenamiento
rct      var word       ' variable para RTime
luz      var word       ' variable para intensidad de luz
TC       var word       ' para grados Celsius del AD592
n        var byte       ' ☐ contador para el botón
ptr      var byte       ' ☐ puntero al archivo de almacenamiento

outs=%0000000001000000 ' ☐ ahora van las instrucciones outs y dirs.
      'fedcba9876543210
dirs=%1111111111111101 ' ☐ todas como salidas bajas
                        ' ☐ excepto P6, salida alta para descargar C
                        ' ☐ y P1 entrada para pulsador

debug cls,"¡Listo para almacenar datos",cr ' ☐
freqout 0,200,2550      ' ☐
freqout 0,400,3400      ' ☐
principal:              ' ☐
  if in1=1 then principal ' ☐ repite hasta que presiona
  n=0                   ' ☐ cronometra el tiempo
clik1:                  ' ☐
  pause 100             ' ☐ cuenta incrementos de 0,1 segundos
  if n>12 then reproducir ' ☐ salta a reproducir después de 1,2 seg.
  n=n+1                 ' ☐ incrementa contador
  if in1=0 then clik1    ' ☐ repite hasta que suelta el botón
leedato:                ' ☐
  if ptr>17 then protesta ' ☐ protesta si la memoria está llena
  freqout 0,10,1900      ' ☐ sonido indicador

RTime 5,0,rct           ' lee la punta de temperatura
low 5                   ' descarga el capacitor de temperatura
TC=kcal/rct*10+(kal//rct*10/rct)-273 ' ☐ calcula Celsius
log(ptr)=TC             ' ☐ almacena la temperatura
ptr=ptr+1               ' ☐ apunta al siguiente lugar de memoria

RTime 6,1,rct           ' lee el fotodiodo
high 6                  ' descarga el capacitor de fotodiodo
luz=65535/rct*/lical    ' calcula lux
log(ptr)=luz/2 max 255  ' ☐ almacena intensidad de luz /2
```

4

Experimento 4: Luz en la Tierra y Adquisición de Datos

```

ptr=ptr+1          ' □ apunta al siguiente lugar de memoria

debug dec TC," C",tab,dec luz," lux",cr ' muestra los valores
goto principal

reproducir:        ' □ muestra las 9 lecturas en pantalla
  freqout 0,50,2550 ' □ sonido indicador
  freqout 0,100,3400 ' □
  debug cls,"Datos almacenados",cr ' □ mensaje en la pantalla
  debug "C",tab,"Lux",cr ' □ imprime unidades de medición
  for n=0 to 16 step 2 ' □ barre los 9 registros
    TC=log(n) ' □ obtiene la temperatura
    luz=log(n+1)*2 ' □ obtiene la luz
    debug dec TC,tab,dec luz,cr ' □ muestra
  next ' □ siguiente registro de 9
pbl: ' □ después de mostrar los valores
  if in1=0 then pbl ' □ espera a que suelte el botón
  debug cr,"presione RESET para borrar los datos",cr ' □ imprime mensaje
  goto principal ' □ vuelve al inicio

protesta: ' □ viene aquí si la memoria está llena
  debug cr,"memoria llena" ' □ mensaje
  freqout 0,50,3400 ' □ ruido
  freqout 0,200,2000,2100 ' □
  goto principal ' □ vuelve al inicio

```

4

Cuando ejecute el programa, podría presionar 9 veces el botón para almacenar 9 registros de temperatura y nivel de luz. Después de eso, el programa emitirá el sonido "memoria llena". En cualquier momento, si mantiene presionado el botón por más de 1,2 segundos, entonces el programa mostrará los **9 registros** en la pantalla. Puede dejar la ventana de debug abierta en la pantalla, y salir a experimentar recolectando 9 registros, y luego regresar a la computadora y recuperarlos. Cuando quiera volver a comenzar, presione el botón reset de la Plaqueta de Educación.

¿Qué son?

Campos y registros:

Cada línea, o fila, de datos es un registro. Cada valor dentro del registro se llama campo. El primer campo en este caso es la temperatura en grados Celsius, el segundo es la luz en lux. Los campos se alinean en columnas. El archivo del ejemplo tiene 9 registros de 2 campos. Esta terminología es común en bases de datos y hojas de cálculo.

En la versión para Windows del software STAMP2W.EXE, puede abrir la ventana debug cuando quiera. En la versión de DOS, debe recargar el programa para reabrir la ventana. Si está usando la versión de DOS, deje abierta la ventana debug mientras sale a experimentar.

Experimento 4: Luz en la Tierra y Adquisición de Datos

Ahora explicaremos el programa, y luego realizará los experimentos.

Este programa comienza con las instrucciones `outs` y `dirs`. Reemplazan a `HIGH 5` y `LOW 6` del Programa 4.3. Observe que la sexta posición del comando `outs` es 1, y la posición 5 es cero. Esto hace P6 alto y P5 bajo, lo que era necesario para descargar ambos capacitores. La posición P1 de la instrucción `dirs` es un cero, lo que hace a P1 una entrada. Los otros pines del BASIC Stamp se fijan como salidas en estado bajo, como práctica de buena programación.

4

El programa comienza con las familiares rutinas `click` y `click largo` del pulsador. Los rótulos en este caso no son `"clik"` and `"clik1"`, pero reconocerá el mismo código. Cuando el botón es presionado, hay una carrera entre el cronómetro y el botón. Si el cronómetro llega a 1,2 segundos, entonces el programa salta a la subrutina `reproducir`.

Pero si el botón se suelta dentro de los 1,2 segundos, el programa continúa con la rutina `leedato`. Aquí, lee la punta de temperatura y la luz como en el programa 4.3. Luego pone la lectura de temperatura en el cajón que apunta la variable, `ptr`. Luego `ptr` se incrementa una unidad. Luego pone el valor de `luz/2` en el cajón siguiente. Luego `ptr` se incrementa nuevamente para apuntar al siguiente cajón vacío.

Al principio de la rutina `leedato`, el programa verifica el valor del puntero, y salta a la subrutina de protesta si el puntero es mayor de 17, para indicar memoria llena. El programa no permite recolectar más datos debido a que no hay espacio para almacenarlos. Usted podría intentar quitar esta línea, para observar qué error ocurrirá.

¿Por qué almacenar el valor de la luz dividido por 2? Los cajones sólo almacenan de a un byte, valores menores o iguales a 255. Esto está bien para los grados Celsius, que estarán en el rango de 0-100. Pero el nivel de luz puede ser mucho mayor. Fue calibrado a 425 lux. Al dividirlo por 2, se pueden almacenar valores de hasta 511 lux. La desventaja es una pérdida en la resolución, pero no es significativa por la baja precisión debida a la calibración "aproximada". Cuando recuperamos los valores de luz almacenados, los multiplicamos por dos para obtener los valores originales. Observe que `luz/2` está seguido por `max 255`. El 255 indica que su dato está fuera de rango.

La rutina `reproducir` usa un bucle `for-next` para barrer los 9 registros.

```
for n=0 to 16 step 2          ' □ barre 9 números pares
```

El "step 2" (paso 2) hace que el valor del índice, `n`, salte de a dos valores (los pares porque comienza en cero) , 0,2,4,6,...,16, en un total de 9 pasos. (Vea el BASIC Stamp II Manual, página 261, en inglés.) El primer valor de temperatura se lee de `log(0)`, y la luz de `log(1)`, y se muestran en la pantalla. Observe que la luz está multiplicada por 2 para recuperar el valor original. Luego el bucle `for-next` incrementa el valor del índice en 2 unidades y obtiene y muestra `log(2)` y `log(3)`, continuando así hasta `log(16)` y `log(17)`.

¿Por qué el programa usa a `n` como puntero, en lugar de `ptr`? Primero, comprenda que no importa qué variable se pone entre paréntesis después de `log()`. Lo único que importa es el valor numérico de lo que está entre paréntesis. Al usar "`n`" como puntero, el valor "`ptr`" no es perturbado. Digamos que tomó 5 lecturas, y luego hace el click largo para leer estas 5 lecturas. Luego puede continuar donde había quedado, tomando las lecturas 6 a 10, y hacer el click largo para leer todos los valores. Es un pequeño refinamiento.

Al final de la rutina reproducir, el programa le recuerda que debe presionar reset para borrar los valores y comenzar de cero.

Este programa usa todo el espacio de variable RAM disponible en el BASIC Stamp. Dos bytes son usados por cada variable word, `rct`, `TC` y `luz`, y un byte para el índice `n` y otro para `ptr`, y 18 bytes para el archivo de almacenamiento. Esto suma 26. Agregue una variable más al programa. Cuando intente ejecutarlo, obtendrá un mensaje de error, "out of variable space" (espacio de variable rebasado).

Experimentos con el Data Logger

1) Verificación de la reducción proporcional a $1/r^2$ (en función de la distancia).

Prepare una soga con nudos a un metro, 1,5 metros, 2 metros, 2,5 metros y así hasta 4 metros. Una la soga a la lámpara spot de 50 wat, R20. Reinicie su Plaqueta de Educación para limpiar los registros de almacenamiento. Tome los datos en cada distancia de la fuente de luz, teniendo cuidado de estar en el centro del haz. Vea y anote los datos de la pantalla debug de la PC. Grafique las lecturas en función de la posición. Verifique que la intensidad cae en forma cuadrática ($1/r^2$). Este es un experimento aproximado. Piense en los factores que perjudiquen los resultados de este experimento. ¡No olvide las fluctuaciones que ya vimos!

2) Investigación sobre la distribución de luz de la lámpara spot (o la que disponga).

Haga un semicírculo usando la misma soga, y un poco de ingenio, de forma que pueda sostener el sensor en 10 ángulos diferentes, alrededor del centro del haz del spot. Obtenga las lecturas y grafíquelas.

3) Velocidad de calentamiento y enfriamiento, adquisición temporal.

Sostenga la punta de temperatura a pocos centímetros de la lámpara, y presione regularmente el botón en intervalos de 15 segundos. Luego aleje el sensor y tome 4 lecturas más, con la misma secuencia. Descargue los datos y grafique la temperatura en función del tiempo. ¿no sería bueno que el data logger tome los datos, con estos intervalos, automáticamente? Es fácil de lograr. Cambie la subrutina principal como sigue:

```
principal:                                ' cambios al programa EM4.4
n=0.....                               ' ☐ inicia el contador
m11:                                     ' ☐ espera al botón o al contador
pause 1000                               ' ☐ pausa de un segundo
if n=15 and ptr<17 then leedato' ☐ obtiene datos a intervalos de 15 segundos
n=n+1                                    ' ☐ cuenta el tiempo
if in1=1 then m11                         ' ☐ puede presionar para obtener dato igual
n=0
clik1:                                   ' ... y continúa como el programa EM4.4
```

Puede poner el número de segundos que necesite. Presione reset para comenzar la ejecución. La rutina toma una lectura manualmente si presiona el botón. El número n puede ser de hasta 65535 segundos, más de 18 horas entre lecturas, si quiere hacer un experimento a largo plazo.

4

4) Escala alternativa del sensor de luz, para luz brillante, método simple.

Si quiere medir luz brillante, exteriores al sol por ejemplo, esta es una simple forma aproximada de conseguirlo. Ponga el capacitor de 0.22 uF en el lugar del capacitor de 0.01 uF en el circuito del fotodiodo. Agregue el factor 22 en las tres líneas que calculan la intensidad de luz:

```
luz=65535/rct*/lical*22          ' calcula lux
y
log(ptr)=luz/44 max 255          ' □ almacena intensidad de luz/44
y
luz=log(n+1)*44                  ' □ recupera el valor
```

Esto es porque la capacidad nueva es 22 veces la anterior. El valor de `rct` que se obtenía con 100 lux es el que producen ahora 2200 lux. Recuerde el efecto de duplicar la capacidad que vimos antes en esta lección. El rango anterior de mediciones era de 0 a 512 lux. Ahora es de 0 a 11264 lux.

5) Escala alternativa del sensor de luz, calibración a pleno sol.

sta le dará la lectura en PAR, en micromoles de fotones por metro cuadrado por segundo. Si lee el recuadro sobre intensidad de la luz, encontrará que es la medición usada para controlar el crecimiento de las plantas. Esta también es una calibración aproximada. Nuestro fotodiodo no tiene los filtros que limitarían las longitudes de onda de la medición, a aquellos valores que benefician el crecimiento de las plantas. Ponga el capacitor de 0.22 uF en lugar del capacitor de 0.01 uF. Momentáneamente haga la constante `lical` igual a 256, y quite el factor 44 de los cálculos de luz:

```
lical    con 256                  ' □ constante de calibración para el fotodiodo
log(ptr)=luz max 255              ' □ almacena intensidad de luz directa
luz=log(n+1)                      ' □ recupera el valor
```

Recuerde que al usar el operador `*/`, `*/256` es lo mismo que nada. Es la fracción 256/256, que es igual a uno. Ponga el sensor fotodiodo apuntando directamente al sol. Por supuesto, va a tener que salir al exterior con la Plaqueta de Educación a batería, en un día soleado para poder hacer esto. Puede tomar datos presionando el botón, o usando el modo automático. Baje los datos a la PC. (Nota: si las lecturas indican 255, significa que está fuera de rango, demasiado brillo. Ponga un pedazo o dos de papel tissue sobre el sensor, sujetado con una banda elástica, y pruebe otra vez.) Tome el valor directo de luz que se

Experimento 4: Luz en la Tierra y Adquisición de Datos

muestra en la pantalla en la segunda columna. Supongamos que sea 188, multiplíquelo por 256, y divídale por 200. Este es el nuevo valor de `lical`. Para el ejemplo, $2000 \cdot 256 / 188 = 2720$.

```
lical    con 272                                ' constante de calibración del fotodiodo
```

y

```
log(ptr)=luz/10 max 255                        ' ☐ almacena PAR/10
```

y

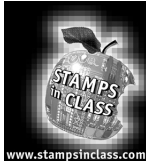
```
luz=log(n+1)*10                                ' ☐ recupera PAR
```

Además, cambie las unidades de medición de la instrucción `debug` de lux, a PAR. La lectura a pleno sol debería ser ahora de aproximadamente 2000.

6) Use su data logger para explorar las temperaturas y niveles de luz al aire libre.

Puede fijar el tiempo de medición automática en 2 horas (7200 segundos), y regresar en 18 horas para ver el comportamiento de la luz y la temperatura mientras usted no estaba.

4



¡Desafío!

¿Comprende el significado de energía por unidad de área? Un cierto láser entrega una energía total de un miliwat en un rayo con un área transversal de un milímetro cuadrado. ¿Cómo es esta intensidad comparada con la del sol, de aproximadamente 1000 wats por metro cuadrado?

4

Misión: Plutón Usted está planeando visitar el planeta Plutón, y desea saber que tan brillante será la luz. Pregunta: ¿Será la suficiente para leer cómodamente esta página? (a) Estime la iluminación de Plutón en lux. (la Tierra está a 149.500.000 kilómetros del Sol, mientras que Plutón está a un promedio de 5.920.000.000 kilómetros del Sol). En la Tierra, con el sensor apuntando al Sol, medimos aproximadamente 110.000 lux. ¿Qué valor aproximado en lux medirá cuando apunte el medidor directamente al Sol desde Plutón? (b) Usando su medidor de luz BS2 calibrado, encuentre un lugar en su ambiente donde el nivel de luz sea comparable a lo que experimentaría en un día soleado en Plutón.

Medidor de Tiempo de Reacción Instale un led y un resistor de 470 ohm en su Plaqueta de Educación de forma que la instrucción `HIGH` lo encienda, y `LOW` lo apague. Escriba un programa que haga lo siguiente para probar su velocidad de reacción. Cuando usted mantiene presionado el botón, el programa espera una cantidad aleatoria de tiempo entre 1 y 15 segundos, y luego enciende el LED. Entonces usted debe soltar el botón, tan rápido como pueda. El programa podría usar la instrucción `RCtime` para medir el tiempo que le lleva soltar el botón. Luego muestra su velocidad de reacción en milisegundos en la pantalla debug, apaga el LED, y vuelve a esperar otra ronda. El programa podría comprobar si usted suelta el botón antes de que el LED se apague, llamándolo "tramposo" en ese caso.

Colorímetro En su kit tiene un led rojo y otro verde. Estos diodos no sólo pueden emitir luz, sino que también pueden actuar como fotodiodos y recibirla. Es decir, la corriente inversa del LED es proporcional al nivel de luz que recibe. Responden mejor al mismo color que emiten. Así un LED rojo responde mejor a la luz roja, y el verde a la verde. Conecte los LEDs rojo y verde como se muestra en la figura 4.5, pero use los pines P8 y P9 del BASIC Stamp, y capacitores de 100pf. La corriente producida por los LEDs es muy pequeña. Puede invertir la posición de los diodos y los capacitores para tener más sensibilidad, si es necesario. Escriba un programa que lea la salida de ambos sensores sucesivamente y muestre los resultados en la pantalla debug. Con los sensores en luz blanca brillante, ambas lecturas serán diferentes, debido a que los diodos tienen sensibilidades distintas. Ajuste la cantidad de luz que llega a los diodos, o ajuste la escala del programa, para que ambas lecturas sean iguales en luz blanca. Luego ponga filtros rojo y verde frente a los diodos. ¿Ven los diodos los diferentes colores de un papel, o a través de diferentes filtros, o distinguen los colores de un prisma?



Experimento 5: El Ambiente Líquido

El tema del experimento del Ambiente Líquido es: "sensores de conductividad y nivel del ambiente líquido." Este es un tipo de sensor más difícil, pero necesario para los experimentos de adquisición de datos. Las actividades asociadas con este experimento son: (1) Conductividad usando entrada Encendido-Apagado o *Rctime*, y sus defectos (2a) Agregar

un oscilador 555 como entrada a la Plaqueta de Educación (2b) Usar el oscilador 555 para medir la conductividad del agua usando puntas de acero inoxidable y (3) Agregar medición de continuidad al data logger del Experimento 3.

5

Introducción

La vista de la Tierra desde la Luna de 1969 dejó claro de una vez y para todos, que vivimos en un planeta de agua. Los científicos, agricultores, doctores, meteorólogos, el público en general, todos necesitan saber alguna vez "cómo, cuándo, o por que" del agua. ¿Cuándo va a llover? Qué profundidad tiene. Qué tan fría, qué tan caliente, qué tan clara, qué tan limpia. Qué minerales o qué materiales orgánicos contiene. Qué tan rápido se mueve. ¿A qué profundidad se encuentra? ¿Cuánto tiempo estuvo ahí? ¿Cuánta agua hay en las capas de hielo, los océanos, los ríos, en el tejido vivo? ¿Cómo se forman las gotas de lluvia? ¿Por qué se produce El Niño? ¿Qué pasa dentro de una nube cuando nieva? ¿Hay peligro de deslizamientos, sequía, inundación, o escasez? ¿Puede vivir aquí un cactus, una rana, un ratón? ¿Puedo beberla? ¿Están desapareciendo las tierras húmedas? ¿Deberíamos preocuparnos?

Así que, el agua será la tercer variable que usaremos en Mediciones Ambientales. ¿Qué puede medir del agua? Estoy seguro que puede pensar cientos de ejemplos sin pestañar siquiera. Nos concentraremos en un par. El primero es detectar su presencia, o su nivel. Esta es la clase de medición que se necesita para monitorear o controlar el nivel de agua de un río, un acuario, o una planta de tratamiento de agua. El segundo tipo de medición será la conductividad eléctrica del agua. sta se usa para detectar la presencia de sal y minerales en el agua, y es también usada para evaluar la calidad del agua potable o para estudiar la mezcla de agua dulce con salada que se produce en los estuarios o desembocaduras de ríos. Hay muchos tipos de mediciones del agua que requieren diferentes sensores, como determinar la acidez o evaluar la claridad. Es un campo muy extenso, con mucha investigación en el desarrollo de sensores que puedan detectar la calidad del agua.

Las mediciones en un medio líquido son más problemáticas que las mediciones de luz o temperatura. Las puntas que detectan temperatura o luz no deben estar necesariamente en contacto eléctrico con el medio a medir. Por el contrario, los sensores de humedad deben estar en contacto directo y son sometidos a corrosión y todo tipo de interacciones eléctricas con metales, iones y corrientes en el medio líquido.

Siguiendo esa observación, queremos hacer una aclaración **IMPORTANTE**. El agua y la electricidad no se mezclan, normalmente, sin planificación. Nunca, repetimos, nunca por ningún motivo permita que se moje la Plaqueta de Educación. Siempre observe las normas de seguridad cuando trabaja con agua y electricidad.



Partes Requeridas

Los siguientes componentes son necesarios para este experimento:

5

- (1) temporizador cmos LMC555
- (4) cables de interconexión
- (2) capacitor de 0.1 uF
- (1) resistor de 100 ohm
- (2) resistor de 100K ohm
- (1) punta de continuidad (dispositivo con dos tornillos de 5 cm, separados 1 cm, montados en un recorte de circuito impreso con cables de conexión)
- (1) vaso
- (1) agua y sal



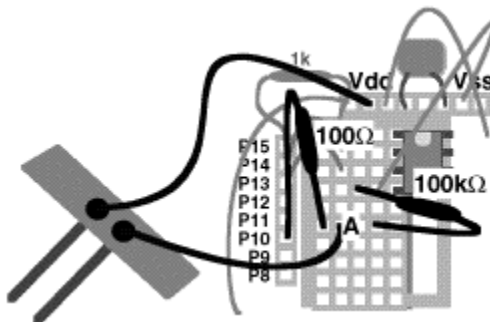
¡Constrúyalo!

Alarma de Humedad

En el Experimento 3 usó la punta de continuidad de su kit, como una introducción al comando `RCtime`. Ahora arme el circuito que se muestra en las Figuras 5.1 y 5.2.

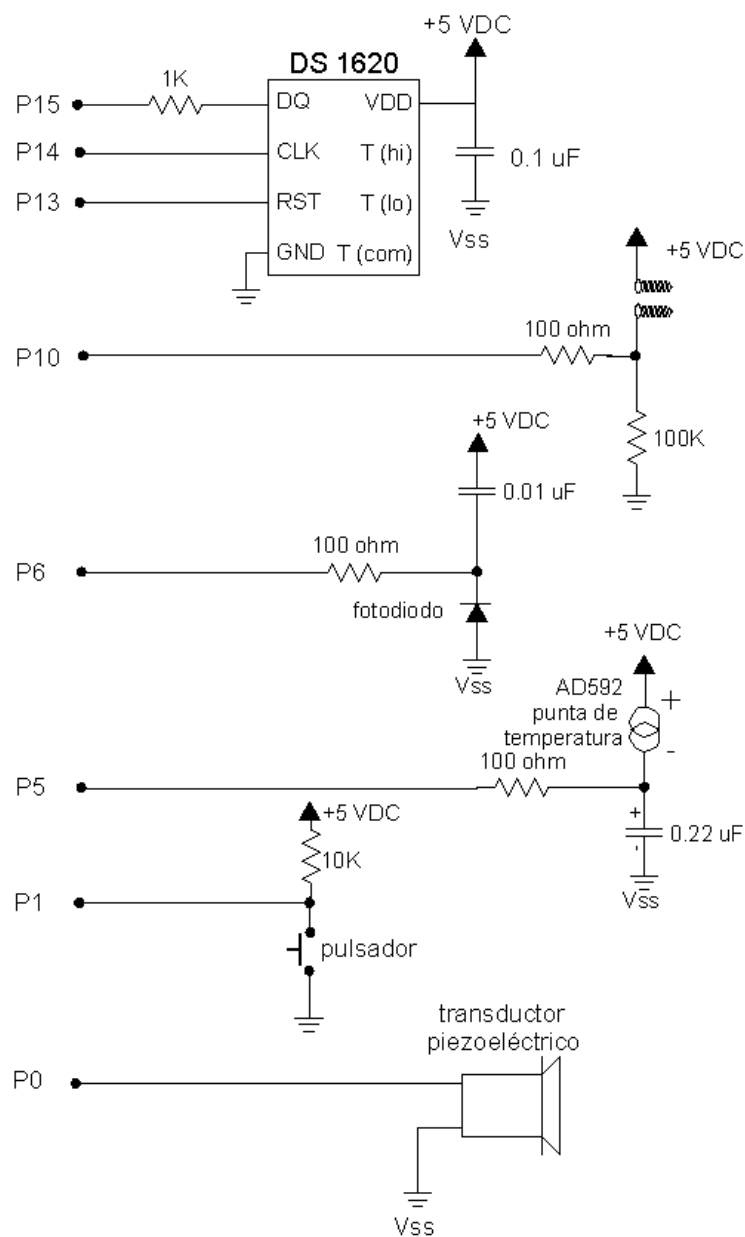
Figura 5.1 Distribución de Componentes de la Punta de Conductividad

- resistor de 100K de Vss (pin 4 del DS1620) al nodo A.
- resistor de 100 ohm del nodo A a P10.
- Punta de conductividad del nodo A a Vdd.



5

Figura 5.2: Circuito Eléctrico de la Punta de Continuidad



Ahora ingrese el siguiente programa:

```
' Mediciones Ambientales programa 5.1
' detector de agua con alarma
bucle:
  debug bin in10
  if in10=0 then bucle
  freqout 0,6,2550
  goto bucle
```

Tenga un vaso con agua a mano. Este es su detector básico de agua y alarma. Cuando ejecute el programa, no escuchará nada hasta que sumerja la punta en el agua. Este programa es igual a las subrutinas del pulsador estudiadas en el Experimento 2, donde pulsando el botón se emitía el sonido de un grillo. En este caso, la punta de continuidad en el agua reemplaza al pulsador. Recuerde la discusión donde se trató la punta de continuidad como un resistor variable, en el Experimento 3. Explique lo que sucede en el programa 5.1, agregándole comentarios.

¿Qué pasa si reemplaza `in10=0` con `in10=1`? Piense una situación donde ésta alarma sea útil.

¿Por qué se escogió un resistor de 100K para el circuito? El resistor fija la sensibilidad. Con valores de resistencia mayores, correremos el riesgo que el circuito diga "¡mojado!" con un poco de condensación en el cableado. Con valores más bajos, ese tipo de error es improbable, pero por otro lado, el sensor puede no indicar "mojado" y estarlo, en caso que el agua sea pura y no conductiva. El valor sale por prueba y error.

Pruebe un resistor de 1K en el lugar del de 100K. (Podría tomar prestado temporariamente 1K del pin 1 del sensor de temperatura DS1620.) Encontrará que debe sumergir la punta mucho más que antes para que suene la alarma. Este tipo de alarmas se usan para que los niños aprendan a no mojar la cama. Un acolchado absorbe la orina y suena la alarma. Un circuito similar es usado para sonar la alarma en plantas industriales en caso de derrames.

Imagine mejorar este circuito para controlar el nivel del agua. Si el agua se derrama, podríamos encender una bomba para limpiar. Luego, cuando el sensor indica que bajó el nivel, se apaga la bomba. Así trabajan las bombas de los sótanos, para mantener el agua lejos de los cimientos, o la bomba achicadora, para mantener el agua fuera del bote. Pero nos estamos adelantando. Este es el tema del Experimento 6. En este momento nos interesa realizar mediciones analógicas cuantitativas, no sólo "sí/no", sino "¿cuánta agua?", y "¿de qué calidad?"

5

Medición de Conductancia Usando Rctime

Vuelva a colocar el resistor de 100k en el circuito anterior con un capacitor de 0.1 F. Este es precisamente el circuito de la Figura 3.4. Ingrese el siguiente programa (es el programa 3.2):

```
' Mediciones Ambientales programa 5.2
' medición de conductividad con Rctime.
rct var word          ' variable word para Rctime
n var byte            ' variable para gráfico de barras
low 10
bucle:
  Rctime 10,0,rct      ' tiempo para subir a 1,3 V
  low 10               ' descarga el capacitor a 0 volts
  rct=rct-1           ' calcula la longitud de la barra
  debug dec rct,tab, rep "*" \ncd rct,cr ' muestra gráfico de barras ascii
  pause 1000          ' baja la velocidad a 1 por segundo.
goto bucle
```

5

Ahora tiene una salida digital que refleja la resistencia del agua entre los conectores de la punta. Observe cómo cambian las lecturas a medida que sumerge la punta en el agua.

Ubicación de la punta	Lectura rct
fuera del agua	
apenas tocando el agua	
1cm dentro del agua	
2cm dentro del agua	
3cm dentro del agua	

¿Ve una tendencia? Repita algunas veces las mediciones, y escriba los valores en el cuadro. ¿Se repiten las lecturas? Es decir, ¿obtiene el mismo resultado cada vez? Fije la punta a profundidad constante en el agua durante un minuto o dos.

Observaciones sobre el programa:

Primera, ¿por qué incluimos la fórmula, $rct=rct-1$? Es para que el gráfico se vea mejor. A medida que saca la punta del agua, el número rct se vuelve más y más grande, pero cuando la punta deja el agua, rct rápidamente se hace cero. Esto es debido al comando $Rctime$, que entrega un "0" como una especie de mensaje de error cuando rebasa su valor máximo.

Cuando a cero le resta 1 en aritmética entera, obtiene 65535. Cuando usa microcontroladores, o en realidad, cuando programa cualquier computadora, debe compensar las pequeñas peculiaridades de los comandos que tiene a su disposición.

¿Qué pasa con el gráfico? Usa un modificador de debug:

```
rep "*" \ncd rct
```

Usted sabe que `rct` es la variable. `rep` viene de "repeat" (repetir). Reimprime el carácter "*" en la ventana debug, las veces que determina el número detrás de `\`. Por ejemplo:

```
debug rep "*" \12
```

imprimirá 12 asteriscos en fila en la pantalla. También podría programarlo como:

```
debug "*****"
```

Pero usar "rep" es más conciso. Y el número detrás de "`\`" puede ser una variable, lo que puede resultar muy útil. En este caso la variable después de "`\`" es el resultado de una expresión. La expresión es "`ncd rct`", donde `ncd` es un operador matemático único del BASIC Stamp. El resultado es la longitud del número `rct` en formato binario. Por ejemplo, si `rct=35` en decimal, su forma binaria es `rct=%100011`. La longitud de ese número binario es 6 dígitos binarios. Usándolo detrás del operador `ncd`, con la forma `ncd rct` obtenemos el valor 6, y seis asteriscos se imprimen en la pantalla debug. ¡Tal vez nunca necesite usar ese comando! Pero ahí está, agréguelo a su valija de trucos. El número de asteriscos aumenta una unidad cada vez que se duplica el valor de `rct`.

Haga una o varias series de mediciones para la tabla anterior. Invierta las conexiones de la punta de conductividad en la Plaqueta de Educación. Es decir, saque el cable de la punta que está conectado al nodo A, y conéctelo a Vdd, y saque el cable que estaba conectado a Vdd y conéctelo al nodo A. Vuelva a hacer la serie de mediciones de la tabla anterior. Probablemente encontrará que los números son ligeramente diferentes.

La diferencia es debida a lo que sucede en el medio líquido, a medida que la electricidad pasa de uno a otro conector de la punta.

El efecto que está viendo se llama "polarización". Las reacciones químicas modifican el electrodo. Esto no es un gran problema para un sensor de tipo si/no, pero es desastroso en mediciones cuantitativas. La polarización se produce porque la corriente circula por la punta siempre en el mismo sentido. Es corriente continua, CC. La solución más simple es alimentar al sensor con una corriente primero en un sentido y luego en el opuesto. Esto es corriente alterna, CA. Muchas de estas reacciones químicas son reversibles, logrando con la corriente alterna lecturas más estables. El BASIC Stamp no puede suministrar la señal de CA necesaria.

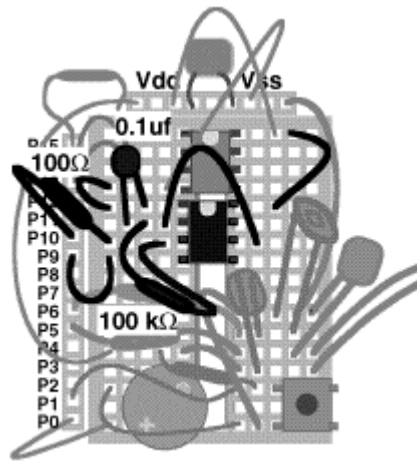
Un circuito integrado externo puede ayudar en este punto. Es uno que ya conoce del Experimento 5 de "¿Qué es un Microcontrolador?", el temporizador 555.

Medición de Conductancia Usando el CI Temporizador 555

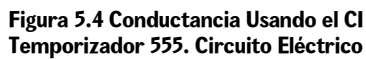
Quite los componentes del experimento anterior, e instale el temporizador (timer) 555 en la Plaqueta de Educación. Sea cuidadoso debido a que el cableado se está poniendo apretado. La distribución de componentes de este circuito se muestra en la Figura 5.3, y el circuito eléctrico en la Figura 5.4.

Figura 5.3 Conductancia Usando el CI Temporizador 555. Distribución de Componentes

- Enchufe el CI 555 en la Plaqueta de Educación, al lado del DS1620. La muesca que indica pin 1 va en el mismo sentido que el otro CI.
- pin 8 del 555 a Vdd (pin 8 del DS1620)
- pin 1 del 555 a Vss (pin 4 del DS1620)
- pin 2 del 555 al pin 6 del 555
- pin 4 del 555 a P9
- pin 3 del 555 a través de un resistor de 100 ohm a P10
- resistor de 100K entre pin 2 y 3 del 555
- capacitor de 0.1 f entre pin 1 y 2 del 555



5



El espacio en la protoboard se vuelve muy limitado. Siga la distribución de componentes de la figura anterior, para hacer entrar el proyecto en la protoboard.

Este circuito es similar al que armó en "¿Qué es un Microcontrolador?" Experimento 5. Es un multivibrador astable. Esta terminología viene desde los inicios de la electrónica. Quiere decir que la salida del circuito (pin 3 del 555) cambia de alto a bajo repetidamente por sí misma. El resistor del pin 3 al pin 2, junto con el capacitor del pin 2 al pin 1, determinan la frecuencia de oscilación. P10 del BASIC Stamp se configurará como entrada para poder monitorear la frecuencia producida por el 555. El pin 9 será configurado como salida para encender y apagar el 555. Cuando P9 es alta, el 555 está encendido. (Si revisa "¿Qué es un Microcontrolador?", notará que el circuito en este caso es diferente. Hay varias formas de conectar el 555, de hecho, hay libros enteros dedicados exclusivamente al 555.)

5

Ingresa y ejecute el programa siguiente:

```
' Mediciones Ambientales programa 5.3
' prueba del oscilador 555
cnt var word          ' variable word para el contador
high 9                 ' enciende el 555
bucle:
  count 10,1000,cnt ' cuenta durante un segundo
  debug dec cnt,cr ' valores
goto bucle
```

Estos son los parámetros del comando count del BASIC Stamp II:

```
count 10,1000,cnt          ' cuenta
      ^^^----- variable de RAM para almacenar el resultado de la cuenta
      ^^-----duración de la cuenta en milisegundos
      ^^-----pin a usar para contar, una entrada del stamp.
```

La lectura que ve en la pantalla cuando la duración es de 1000, debería ser de aproximadamente 75. Escriba su propia lectura.

lectura, cnt ?_____ cuando la duración es 1000, resistor 100K, capacitor 0.1uF.

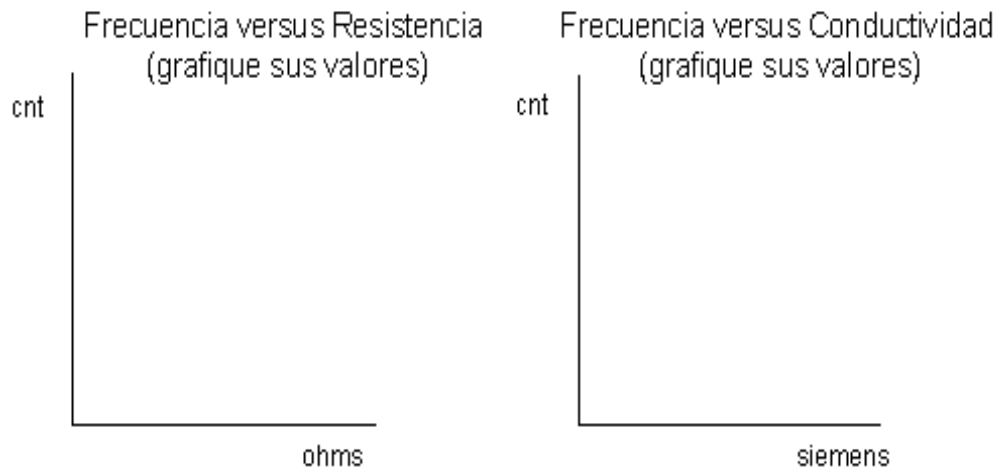
Ahora coloque un segundo resistor de 100K en paralelo con el primero, lado a lado en la Plaqueta de Educación. La combinación en paralelo de dos resistores de 100K es igual a 50K (la combinación en serie es 200K). La frecuencia debería ser el doble. Entre los valores en la tabla de abajo. Ahora ponga dos resistores de 100K en serie desde el pin 2 al pin 3 del 555. La frecuencia debería ser la mitad del valor original. También coloque estos datos en la tabla de abajo. Calcule el valor de $1/R$, que es llamado conductancia, y tiene unidades de siemens (un término más antiguo y más usado es el mho, u ohm dicho al revés: 1 siemen 1 mho).

Haga un gráfico rápido de la frecuencia en función de la resistencia, y otro de la frecuencia en función de la conductancia, en el espacio provisto a continuación.

5

R, resistencia, ohms	G, conductancia, mho (=1/R)	cnt, del comando COUNT del BASIC Stamp
50K		
100K		
200K		

Necesitará calcular $G = 1/R$, y medir cnt del programa 5.2



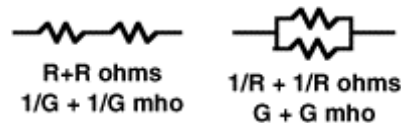
Observe qué gráfico es más lineal.

¿Por qué tenemos que hablar de resistencia y conductancia, si una es la inversa de la otra? Porque así se acostumbra. Hay un término para la inversa de todo en la electrónica. En electrónica es más común hablar de resistencia. Sin embargo, en la ciencia de materiales, química, e instrumentación ambiental, es más común escuchar el término conductancia. Tal vez es debido a que en el medio líquido hay muchísimos caminos diferentes entre dos puntos. Los caminos distintos son como muchos resistores en paralelo, y las modificaciones del medio líquido tienden a cambiar esos elementos en paralelo. Así es más fácil hablar de conductancia, porque la conductancia en paralelo se suma. La Figura 5.5 muestra como se colocan los resistores en serie y paralelo, para medir resistencia (R) y conductancia (G).

5

Figura 5.5: Fórmulas de Resistencia y Conductancia

Resistencia R, y conductancia G, de resistores en serie y paralelo. La fórmula de resistores en paralelo es más fácil en términos de conductancia.



Vuelva a poner solamente el resistor de 100K en el circuito. En su programa 5.3, cambie el parámetro de duración de 1000 milisegundos, a 500 milisegundos, o a 2000 milisegundos. Observe que las lecturas cambian con un factor cercano a 2 cada vez.

Digámoslo de otra forma. Anteriormente escribió un valor de `cnt`, valor que se obtuvo con un resistor de 100K y un capacitor de 0.1 F en el circuito, y un parámetro de duración de 1000 en el comando `count`. ¿Qué duración debería ponerle al comando `count` para que la lectura fuese 100 en lugar de 75 (o lo que usted haya leído ____)? Bien, debe hacer proporcionalmente más larga la duración. Una duración mayor le da una cuenta mayor, ¿no? Calculemos:

duración 1000 (100/75) 1333, pero usted use su propia lectura:

duración 1000 (100/____)
ponga su lectura aquí

Esta es su constante de calibración de duración. La necesitará más adelante.

Ponga ese nuevo valor de duración en el comando `count` en su programa, en lugar del 1000. Ahora cuando ejecute el programa con 10^{-5} siemens (100K) en el circuito, mostrará 100 en la pantalla, en lugar del valor anterior.

El punto es que el parámetro duración del comando count puede ser usado para escalar los resultados, de forma que aparezcan directamente en siemens. ¡Queremos que piense cuantitativamente!

Cambiamos el comando debug para que muestre las unidades. Y, ya que estamos, podríamos calcular la resistencia en ohms y mostrarla también.

```
' Mediciones Ambientales programa 5.3b
' Calibración del oscilador 555
cnt var word      ' variable word para contador
R var word        ' variable word para resistencia
high 9            ' enciende la oscilación del 555
bucle:
  count 10,1333,cnt ' cuenta por aproximadamente un segundo
  '^^^^-----   ;;;USE SU PROPIA CONSTANTE!!!
  R = 50000/cnt*2   ' calcula resistencia R=1/G
  debug dec cnt,"E-7",tab,dec R,"00",cr ' valores
goto bucle
```

5

¿Qué es la

Teoría del uso del temporizador 555 para medir conductividad.

Hay muchas referencias que explican el funcionamiento del circuito temporizador 555, y cómo se aplica. Incluso libros enteros que no tratan otra cosa que el 555. El punto importante para la medición de continuidad es que la corriente a través del resistor en este circuito es alterna, primero circula en una dirección y después en la opuesta, con la misma intensidad. Como dijimos antes, es lo más conveniente para una punta en un medio líquido. Se equilibra el flujo de corriente en cada dirección, para evitar la corrosión, el enchapado, y la polarización. La teoría del 555 es muy parecida a la de RCtime, pero no vamos a tratarla aquí. La ecuación de la frecuencia de salida es aproximadamente:
 $f = \frac{3}{4} R C$. Con R 100000 Ohm y C 0.1 f, se obtienen 75 Hertz.

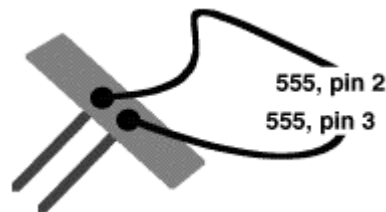
La pantalla debería mostrar $100E^{-7}$ (por $100 \cdot 10^{-7}$ siemens), y en la segunda columna debería mostrar 100000 (por ohms). Verifique la calibración agregándole un resistor en paralelo de 100K, para obtener 50K. La pantalla debería mostrar $200E^{-7}$ siemens y 50000 ohms. Observe que en la lectura de resistencia se agregan dos ceros al valor de R, para que salga en ohms.

Conductancia en el Agua

Es hora de probarla. Necesitará un vaso lleno de agua, una cuchara y unas pizcas de sal común. Reemplace el resistor de 100K por el sensor de conductividad. Deje el valor de la constante de calibración de duración que calculó para su circuito. La Figura 5.6 muestra cómo conectar la punta al temporizador 555 de su Plaqueta de Educación.

Figura 5.6: Sensor de Conductividad

Reemplace el resistor de 100K por el sensor de conductividad.



Con el sensor de conductividad en el circuito, ejecute el Programa 5.3b otra vez. Debería obtener una lectura en siemens y en ohms al poner sus dedos en la punta. Mire lo que pasa en la lectura si se humedece los dedos. ¿Cómo explica este resultado en términos de conductancia?

Sin tocar la punta, conecte un resistor de 100K a través de los terminales, para confirmar que el medidor esté calibrado. Debería leer 100E^{-7} siemens, 100K.

Ponga el sensor a 1 cm de profundidad, en el centro del vaso con agua de la canilla y tome la lectura. Repita la medición a 2, 3 y 4 cm de profundidad. Lea la conductancia de la pantalla de la PC. Debe poder determinar la profundidad en el agua. Tal vez poniendo marcas a los costados del vaso, o en el mismo sensor.

Agua destilada o de la Canilla, Conductancia en función de Profundidad

Nivel de agua	Conductancia
1 cm	
2 cm	
3 cm	
4 cm	

Manteniendo el sensor a profundidad constante, muévelo hasta que esté cerca del borde del vaso. ¿Qué pasa con la lectura? ¿Puede explicarlo en términos de conductancias en paralelo?

Regrese el sensor al centro del vaso. Mire las lecturas a medida que introduce un objeto metálico en el agua, como el mango de una cuchara, cerca del sensor de la punta. ¿Cómo afecta esto las lecturas? ¿Por qué es diferente a llevar la punta hasta el borde del vaso?

Disuelva una pizca de sal en el agua del vaso. Primero eche la sal, y mirando las lecturas, mézclela con el agua. Una vez disuelta la sal, tome lecturas a distintas profundidades:

Pizca de Sal Disuelta en Agua, Conductancia en Función de la Profundidad

Nivel de agua	Conductancia
1 cm	
2 cm	
3 cm	
4 cm	

5

La conductividad se usa a menudo para determinar la salinidad del agua (cuánta sal contiene por unidad de volumen), o más generalmente, cuánto mineral contiene. Si usó agua de la canilla, debería realizar el experimento nuevamente con agua destilada. Manteniendo la profundidad constante, podría usar esta punta para medir salinidad, que está muy relacionada, a través de una fórmula muy complicada, a la conductividad.

Note que en cada caso la conductividad es proporcional a la profundidad, tanto con agua de la canilla, como con agua salada. Usted puede usar este dispositivo para medir la profundidad del agua.

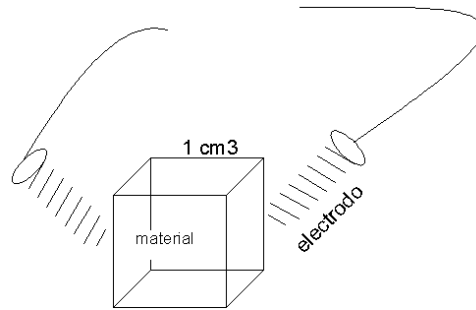
Sin embargo, las dos mediciones están entremezcladas. Si usa este dispositivo para medir profundidad, debe asegurarse que la cantidad de sal permanezca constante, o debe obtener una medición de conductancia separada, para compensar el error. Por otro lado, para medir conductancia, debe mantener el sensor a profundidad constante.

En el diseño de instrumental profesional se tiene muy en cuenta la compensación de los efectos de las variables entremezcladas. En mediciones de continuidad, se tiene la precaución de confinar la solución a un volumen fijo, usar electrodos de materiales estables, y controlar la temperatura en el punto de medición. Los medidores de profundidad del agua profesionales, rara vez se basan en el principio de la conductancia, debido a éstas dificultades.

La conductividad del agua en el ambiente natural barre muchos órdenes de magnitud. La continuidad se mide en unidades de Siemens por centímetro. El agua del Océano puede tener una conductividad de 50000 Siemens por cm, mientras que el agua destilada pura puede tener una conductividad de microsiemens por cm.

Figura 5.7: Medición de Conductividad

La conductividad es medida, en teoría, con un bloque de material, de 1 cm de lado. Los electrodos se sujetan a dos caras opuestas del bloque, y se mide la conductancia. Debido a que el bloque tiene un ancho de 1 cm, la conductividad tiene unidades de siemens por cm. Las mediciones de conductividad se supone que fueron obtenidas con esta configuración, pero la configuración real es mucho más complicada.



5

¿Cuál ES?

La diferencia entre conductancia y conductividad.

Conductividad es una propiedad de los materiales. Los materiales que conducen bien la electricidad, como los metales, tienen una conductividad alta, mientras que los aislantes tienen una conductividad baja. Si toma un cable fino de un metro de largo, tendrá cierta resistencia entre sus extremos, y su conductancia será simplemente la inversa de este valor. Un alambre más grueso del mismo material y el mismo largo, tendrá una resistencia más baja y una conductancia más alta. La conductividad es la misma en ambos casos. Es una propiedad del material

Para calibrar un instrumento de conductividad, necesitará una solución salina estándar, esto le permitirá hacer el salto de conductancia (medida por su Plaqueta de Educación con su sensor especial) a conductividad (una propiedad del agua que está siendo medida, independiente de las peculiaridades del instrumento de medición). Usted debe encontrar una constante llamada "constante del instrumento", que depende de la forma de la punta que está usando. Es una constante de proporcionalidad. Haremos la calibración en la sección de experimentos adicionales.

Continuación de Almacenamiento de Datos: Secado de Suelos

En la naturaleza, el fenómeno de la evaporación es muy importante. El agua se evapora del suelo y también se pierde en la transpiración de las plantas. La velocidad de evaporación y otros mecanismos de pérdida de agua dependen de la intensidad solar, la velocidad del viento, la temperatura, la humedad y el tipo de cobertura. Puede usar el data logger de la Plaqueta de Educación para estudiar la evaporación. Sobre una base práctica, puede hacer que su data logger le avise cuándo regar las plantas o el jardín.

Modifique el Programa 4.4 como se muestra a continuación. Esto le agrega conductividad a los datos. Ahora será capaz de almacenar 6 lecturas en la memoria, para ser leídas más tarde. El número total de bytes disponibles para almacenamiento es 18. Con 3 campos por registro, significa que estamos limitados a un total de 6 registros.

```
' Mediciones Ambientales programa 5.4
' almacenamiento de temperatura, luz y conductancia en la RAM
kal      con 15068      ' USE SU CONSTANTE DE CALIBRACIÓN DEL AD592.
lical    con 647        ' USE SU CONSTANTE DE CALIBRACIÓN DEL FOTODIODO
condcal  con 1333      ' ☐ USE SU CONSTANTE DE CALIBRACIÓN DE CONDUCTANCIA.

interval con 10        ' ☐ toma datos cada 10 segundos
                        ' puede elegir el intervalo que quiera
log      var byte(18)  ' 18 bytes reservados para archivo almacenamiento
rct      var word      ' variable para Rctime
luz      var word      ' variable para intensidad de luz
cnt      var rct       ' ☐ variable para conductividad ALIAS de rct
TC       var word      ' grados Celsius del AD592
n        var byte      ' contador del pulsador
ptr      var byte      ' puntero de almacenamiento

outs=%0000000001000000 ' ☐ ahora ponga todas las instrucciones outs y dirs.
      'fedcba9876543210
dirs=%111110111111101  ' ☐ todas como salidas en estado bajo, excepto:
                        ' P6 es salida alta para descargar C del fotodiodo
                        ' ☐ P5 es salida baja para descargar C del AD592
                        ' P1 es entrada para el pulsador
                        ' ☐ P10 es entrada para conductividad (555)
                        ' ☐ P9 controla encendido-apagado del 555
                        ' ☐ P0 es salida para el piezoeléctrico

debug cls,"listo para almacenar datos",cr
freqout 0,200,2550
freqout 0,400,3400
debug "C",tab,"lux",tab,"siemens",cr      ' ☐ muestra unidades
```

5

```

principal:          ' ☐
    n=0              ' ☐ inicializa el contador
m11:                ' ☐ espera botón o tiempo
    pause 1000       ' ☐ espera un segundo
    if n=interval and ptr<17 then leedato ' ☐ lee datos a intervalos
    n=n+1            ' ☐ cuenta el tiempo
    if in1=1 then m11 ' ☐ puede presionar botón para obtener datos también
    freqout 0,5,3400 ' ☐ tick al liberar el botón
    n=0              ' ☐ reinicia el contador para tiempo presionado
clik1:              '
    pause 100        ' escala el tiempo en incrementos de 0,1 segundos
    if n>12 then reproducir ' salta a la subrutina reproducir después de 1,2 seg.
    n=n+1            ' incrementa el tiempo
    if in1=0 then clik1 ' repite hasta que: suelta el botón o rebasa el tiempo
leedato:            '
    if ptr>17 then protesta ' protesta si la memoria está llena
    freqout 0,10,1900 ' sonido indicador

    Rctime 5,0,rct    ' lee la punta de temperatura
    low 5              ' descarga el capacitor de temperatura
    TC=kcal/rct*10+(kcal//rct*10/rct)-273 ' calcula Celsius
    log(ptr)=TC        ' almacena temperatura
    ptr=ptr+1          ' apunta al byte siguiente

    Rctime 6,1,rct    ' lee el fotodiodo
    high 6             ' descarga el capacitor del fotodiodo
    luz=65535/rct*/lical ' calcula lux
    log(ptr)=luz/2 max 255 ' almacena intensidad de luz/2
    ptr=ptr+1          ' apunta al siguiente byte

    high 9             ' ☐ enciende el 555
    pause 100          ' ☐ pausa para establecimiento del oscilador
    count 10,condcal,cnt ' ☐ cuenta la frecuencia
                        ' ☐ <-- ¡¡¡use su factor de escala!!!(condcal)
    low 9              ' ☐ apaga el 555
    log(ptr)=cnt        ' ☐ almacena la conductancia
    ptr=ptr+1          ' ☐ apunta al siguiente byte

    debug dec TC,tab,dec luz,tab dec cnt,"E-7",cr ' muestra datos
goto principal

reproducir:         ' ☐ muestra los 6 registros en la pantalla
    freqout 0,50,2550 ' sonido indicador
    freqout 0,100,3400.....'
    debug cls,"datos almacenados",cr ' mensaje en la pantalla
    debug "C",tab,"Lux",tab,"mho",cr ' ☐ imprime unidades
    for n=0 to 15 step 3 ' ☐ barre los 6 registros

```

5

```

TC=log(n)                ' recupera temperatura
luz=log(n+1)*2           ' recupera luz
cnt=log(n+2).....      ' □ recupera conductancia
debug dec TC,tab,dec luz,tab,dec cnt,cr ' □ muestra
next.....              ' siguiente registro
pbl:                     ' terminó de mostrar
if in1=0 then pbl        ' espera que se libere el botón
debug cr,"presione RESET para borrar datos",cr ' imprime mensaje
goto principal           ' vuelve al inicio

protesta:                ' viene aquí si la memoria está llena
debug cr,"memoria llena" ' mensaje
freqout 0,50,3400         ' sonido indicador
freqout 0,200,2000,2100  '
goto principal           ' vuelve al inicio

```

5

Notas sobre este programa:

Recuerde el Experimento 4 donde ya usamos este programa. Usábamos 18 bytes para el archivo de almacenamiento de datos, y el resto para las variables del programa. En este programa reutilizamos la variable `ret` para la función `count` de la conductancia. La llamamos `cnt`, y la definimos como un alias de la variable `ret`. Esto significa que `cnt` y `ret` en realidad es la misma variable. Cambiando el valor de una variable se modifica el de las dos, simplemente porque es la misma.

No hay nada inusual en este programa. Es una expansión directa del Experimento 4. Se está poniendo largo, pero cada subrutina tiene su función específica.

En este programa las instrucciones `outs` y `dirs` se modificaron para tener en cuenta los pines nuevos. P10 es una entrada para el comando `count`. P9 es una salida para encender y apagar el 555.

La constante nueva, `interval`, fija el intervalo de segundos entre lecturas (0-65535).

El programa nuevo tiene el código necesario para la punta de conductancia.

Tuvimos que modificar la subrutina `reproducir` para recuperar los datos de conductancia.

Ponga a funcionar este programa, con intervalos de medición de 10 segundos. Luego coloque la punta de temperatura y la de humedad en un vaso o un florero lleno de vermiculita u otro compuesto para macetas. Póngalo al sol, con el sensor de luz calibrado para exteriores. Déjelo por 6 horas. Mire los datos obtenidos. ¿Es apropiado el intervalo de 10 segundos?

Si está haciendo esto en una clase, diferentes grupos pueden introducir variaciones al experimento. Por ejemplo, algunos al sol, algunos a la sombra, algunos con ventilación, otros no. Use compost para macetas. Experimente, es la única forma de aprender a usar microcontroladores en la Ciencia Ambiental.

Como alternativa, para un experimento más rápido, cubra la punta del sensor de conductancia con una toalla de papel humedecida. Registre los valores de temperatura y humedad de la toalla de papel a medida que se seca.



Experimentos Adicionales para Intentar

1) Sensor de Condensación

Presione una pieza de plástico o vidrio, contra los tornillos del sensor de condensación. Cuando el vidrio está seco, es un aislante, y la conductividad es baja, así como la conductancia que mediremos. Pero si empaña el vidrio con su respiración, se depositará la condensación que conducirá la electricidad. Dependiendo de la temperatura y la humedad, puede tener que enfriar la superficie para poder lograr la condensación. Este tipo de sensor es útil en la agricultura, donde la condensación que se forma en las hojas de las plantas puede infectarlas con hongos y plagas.

2) Experimento de Sensor de Humedad

Busque un pedazo de paño de hilo o de algodón liviano. Remóelo en agua salada (una pizca de sal bastará) y enrósquelo alrededor de los tornillos de acero inoxidable de la punta de continuidad. Séquelo con un secador de pelo y observe la conductividad mientras hace esto. Luego expóngalo a la humedad ambiente. Si respira sobre éste, la conductancia aumentará. El NaCl tiene un punto de transición a aproximadamente 75 % de humedad, a partir del cual absorbe agua. Por debajo del 75 % de humedad, el NaCl tiende a entregar el agua a la atmósfera. Por encima del 75 % de humedad, el NaCl tiende a absorber humedad de la atmósfera. La conductancia sigue ese comportamiento. Sales diferentes responden a distintos niveles de humedad.

3) Explorador de Superficie:

- Haga una pileta de agua poco profunda con una gran fuente (de borde bajo) de plástico, vidrio o material descartable (no conductor). Ponga cuidadosamente unos granos de sal gruesa distribuidos por la fuente. Use la punta para explorar la difusión de la sal en su "laguna". Las incursiones del agua salada en ríos y arroyos es un gran problema en áreas donde el agua fresca se usa en la industria y en la agricultura.
- Haga unos garabatos sobre un papel usando un lápiz grueso o una carbonilla de artista. Mueva la punta sobre el papel, explorando el espesor y la resistencia de los trazos.

4) Dependencia Térmica de la Conductancia

La conductancia de soluciones salinas acuosas depende del tipo de sal, y también de la temperatura. Disuelva un poco de sal común en agua, y mida temperatura y conductancia a medida que calienta el agua. Recuerde que el sensor debe estar a profundidad constante. Grafique el resultado de su experimento. Realice el mismo experimento con otra sal (digamos KOH), o con ácido (vinagre). Encontrará que cada solución tiene su propia característica de dependencia de la temperatura. Los sensores de conductancia comerciales siempre miden la temperatura y la conductancia. Con estos datos calculan la concentración de la sal. ¿Puede imaginarse cómo hacen estos cálculos? Libros de química, como el Handbook of Chemistry and Physics, contienen este tipo de información. ¡Búsquela! Debe saber de antemano qué tipo de sal o mezcla de sales están disueltas en la solución.

5) Calibración Cuantitativa de la Punta de Conductividad, Usando una Solución Estándar.

Para calibrar este sensor con el objeto de medir conductividad (propiedad del material) en lugar de conductancia (una cantidad eléctrica), deberá preparar una solución estándar que tenga una conductividad conocida. Estas soluciones pueden ser compradas, o puede hacer la suya en clase, agregando una cantidad conocida de clorato de potasio (KCl) a una cantidad exacta de agua. Las tablas de conductividad se encuentran en manuales de química o sobre calidad del agua, o en referencias como el Handbook of Chemistry and Physics. Una vez que tenga la solución estándar, mida su conductancia con su instrumento de la Plaqueta de Educación. Así obtiene una constante de proporcionalidad entre su lectura de conductancia y la conductividad de la solución. Esta se llama constante del instrumento. Tiene que ver con la geometría de los electrodos y el vaso. También debe medir la temperatura de la solución. Con esta información en mano puede proceder a medir la conductividad (y la concentración) de muestras de agua desconocidas.

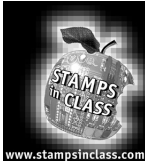
6) Error de Fuga a Tierra

Conecte un cable largo a Vss (cero volts) en la Plaqueta de Educación. Introduzca el extremo libre del cable en el vaso donde se encuentra operando, y mostrando sus lecturas en la pantalla, la punta de conductancia. Verá que las lecturas cambian. Esto es debido al camino a masa extra provisto por el cable. Esta situación es común en grandes sistemas de instrumentación, digamos, una planta industrial. Pero a veces es difícil rastrear el punto en el que se produce la interacción. Puede haber conexiones no planeadas entre puntos del sistema. Para evitar este problema, los ingenieros a menudo diseñan sensores "aislados", lo que significa que las señales pasan a través de un vínculo óptico u otra barrera de este tipo, de forma que no haya conexión eléctrica directa. Esto es extremadamente importante en situaciones donde se prioriza la seguridad o el riesgo eléctrico, como en la instrumentación médica.

7) Explorador Ambiental

Mida la conductividad de algunas muestras de agua. Pruebe agua destilada, agua de la canilla, agua de mar. ¿Cuánta sal debe agregarle a un litro de agua destilada, para hacerla tan salada como la del mar?

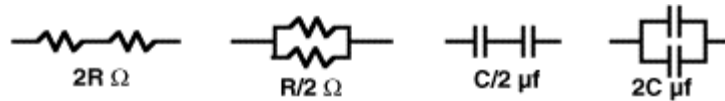




¡Desafío!

5

- 1) Escriba un programa que cuente la cantidad de veces que presiona un botón en 5 segundos. El BASIC Stamp debe emitir un sonido de "largada", luego contar el número de veces que presiona el botón, luego emitir sonido de "llegada", mostrar los resultados en pantalla, pausa de 3 segundos, y repetir.
- 2) Instale un led rojo y uno verde en la Plaqueta de Educación, de forma que P7 y P8 puedan encenderlos y apagarlos. Modifique el programa que mide temperatura, luz y conductividad como sigue. Encenderá el LED verde si las mediciones están dentro del rango operativo normal (elegido por usted). Si salen del rango normal, el LED verde se apaga, y se enciende el LED rojo. Si las mediciones regresan al rango normal, el LED verde se volverá a encender, pero la luz roja seguirá encendida para advertir que hubo un "problema". Si las lecturas se salen del rango, encienda la "sirena de alarma".
- 3) En el circuito de la figura 5.4, la frecuencia de la oscilación es proporcional a $1/RC$, donde R es la resistencia, y C es la capacidad. En su kit tiene 2 resistores de 100K y 2 capacitores 0.1 F. Puede poner los resistores en serie y obtener 200K, y en paralelo 50K. Puede poner dos capacitores en paralelo para hacer 0.05 F, y en paralelo para hacer 0.2 F. Escriba un programa que muestre la frecuencia en hertz para cada valor de resistencia y capacidad de la tabla siguiente.



Valores a completar son	0.05 uF	0.1 uF	0.2 uF
frecuencias del timer 555			
50K-20E ⁻⁶ mho			
100K-10E ⁻⁶ mho			
200K-5E ⁻⁶ mho			

Esto verificará si comprendió cómo usar la protoboard.

¿Parece cierto que frecuencia constante/RC? Grafique la frecuencia en función de la conductancia. Grafique frecuencia en función de la resistencia. ¿Cuál es lineal?



Experimento 6: Medición y Control

El tema del experimento Medición y Control es que los microcontroladores pueden hacer ambas cosas, cerrando el lazo de realimentación. Las actividades relacionadas con este experimento son: (1) Realimentación para controlar el nivel de agua de un recipiente usando una bomba alimentada a batería, y detector de nivel por conductancia. (No permita

que se moje la Plaqueta de Educación) y (2) Medición y control simultáneo de 4 variables. Los materiales usados en este experimento se improvisarán con lo que esté disponible en su salón de clases.

Introducción

6

La medición y adquisición de datos están a menudo relacionadas con el control. No satisfecha con sentarse a mirar, la Plaqueta de Educación puede afectar las condiciones del mundo exterior. Puede abrir una puerta en respuesta a un individuo que se aproxima, como usted vio en "¿Qué es un Microcontrolador?", Experimento 4. O puede funcionar como termostato, para controlar la calefacción o el aire acondicionado cuando la temperatura es muy baja o muy elevada. En la industria, en el campo, en oficinas públicas, en investigación científica, todos los procesos deben ser controlados y regulados basándose en la medición, para obtener el resultado deseado. Algunos instrumentos necesitan control y medición propios. Imagine qué sucede en una máquina como el explorador de Marte automatizado, donde brazos robóticos, laboratorios químicos e instrumentos de todo tipo, deben funcionar como un sistema de medición y control integrado, lejos de la interacción humana. Muchos instrumentos modernos, como analizadores de ADN o analizadores de la calidad del agua automatizados, son maravillas de la medición y el control.

En este experimento, encenderá una bomba para regular el nivel de agua en un recipiente, o para mantener la humedad de la tierra de una planta en una maceta. Puede pensar que está trabajando con una versión a escala de una quinta, una planta de tratamiento de agua, o el sistema de riego completo de un viñedo.

La realimentación es muy importante aquí. Es posible tener control sin realimentación. Si usted pone un vaso de agua por día en su maceta, sin importar la condición de la planta, no hay ninguna realimentación. Corre el riesgo de exceder la dosis de agua que la planta necesita, y desperdiciar agua y fertilizante. No será problema con una pequeña planta con un sustrato con buen drenaje, pero imagine que se trata de una quinta en el desierto, o la operación de un gran invernáculo. Si primero se mide la condición del suelo o de la planta, y en base a esto se decide regar o no, es lo que llamamos realimentación. El resultado puede ser una planta más feliz así como un uso más eficiente de los recursos. Esto es muy importante cuando el agua escasea. La realimentación puede tomar muchas formas, e involucrar una combinación de mediciones en el control de las decisiones.

El proyecto final en esta serie de Mediciones Ambientales será un data logger que combina los dos sensores de temperatura (el DS1620 y el AD592), el sensor de luz, el sensor de conductividad, y el control de la bomba en un programa. Los datos serán almacenados en la memoria EEPROM del BASIC Stamp. Este data logger puede ser usado en una gran variedad de experimentos, que usted puede emprender como proyecto de estudios, bajo su propia iniciativa. ¡Gracias por involucrarse! ¡Sea bueno con el planeta!



Partes Requeridas

Son necesarios los siguientes componentes en este experimento:

6

- (1) transistor TX1049A NPN "superbeta" (rotulado TX 104 9A)
- (1) resistor de 100 ohm
- (1) resistor de 10 ohm, 1 wat (marrón-negro-negro), construcción reforzada (este resistor es fabricado con materiales especiales con resistencia a altas temperaturas)
- (1) bomba (Edmund Scientific) con mangueras de 1/4"
- (2) 30 cm de cables rojo y negro para prolongar los cables de la bomba
- (1) vaso y bandeja, vaso con un agujero de 1/4" cerca de su base (el vaso y la bandeja no están incluidos en el Kit de Componentes de Mediciones Ambientales)

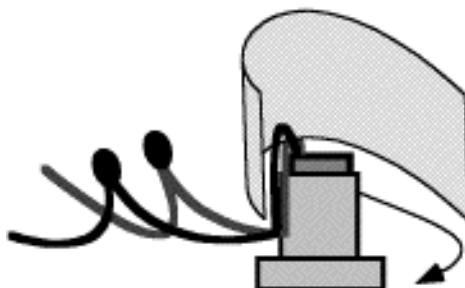


¡Constrúyalo!

La Figura 6.1 muestra la preparación de la bomba. Los cables de la bomba son frágiles. La bomba *no tolera agua dentro del motor*. Recomendamos que tome un trozo de cinta y lo enrosque alrededor del gabinete de la bomba para sujetar los cables. Cierre la cinta en la parte superior (figura 6.1, 6.2) para proteger la bomba de salpicaduras. Se proveen cables robustos para prolongar los finos y frágiles de la bomba.

Figura 6.1: Preparación de la Bomba

Enciente la parte superior de la bomba para proteger los cables y evitar que el agua ingrese a la bomba. Empalme y enciente los cables de la bomba con los cables rojo y negro incluidos en el kit.



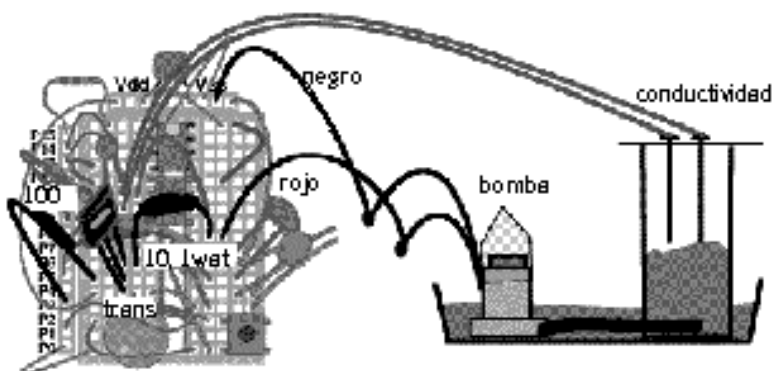
6

Control de Encendido-Apagado de la Bomba

Agregue el circuito del transistor a la Plaqueta de Educación, y conecte la bomba de modo que el programa PBASIC sea capaz de encenderla y apagarla. La distribución de componentes de la Figura 6.2 muestra al sensor de conductividad encima del vaso, pero con el tiempo pondremos al sensor sobre un costado. Regresaremos sobre el tema en un momento.

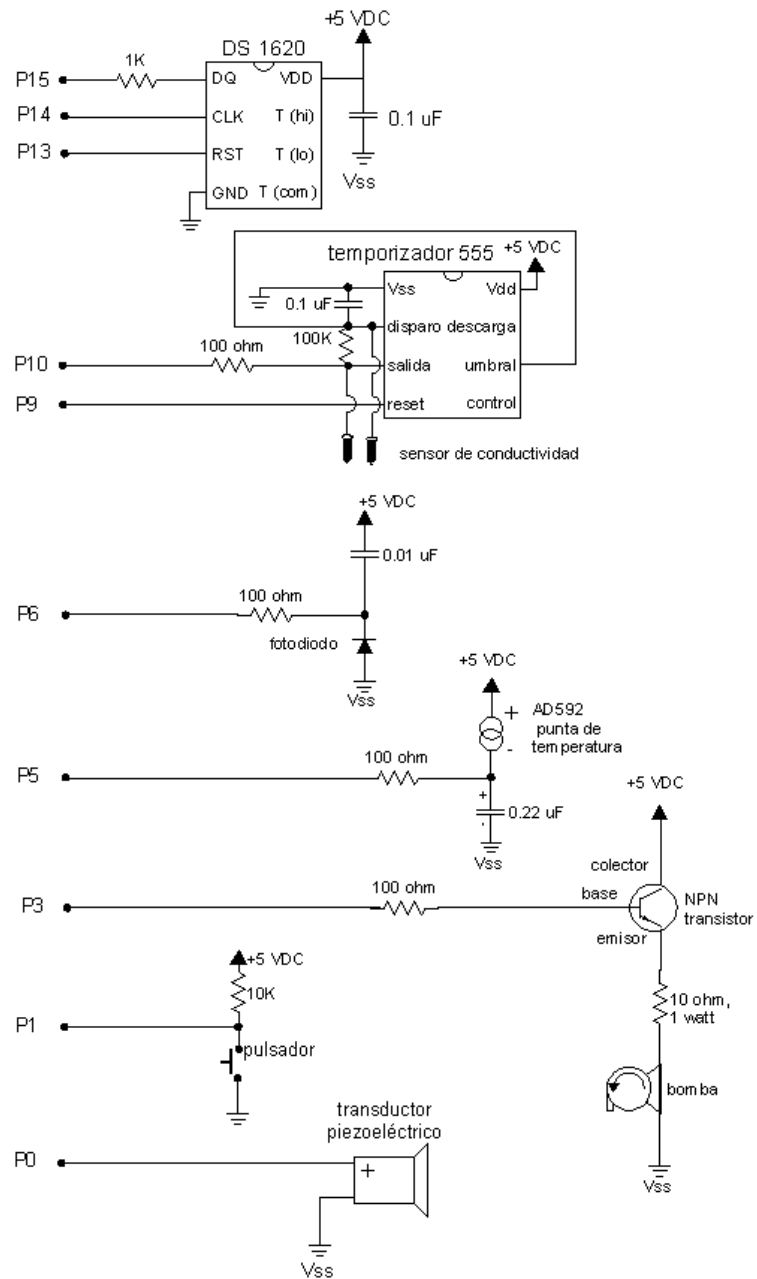
Figura 6.2: Control de Bomba con Transistor

- Colector del transistor al nodo Vdd
- base del transistor a través de un resistor de 100 ohm a P3
- emisor del transistor al resistor de 10 ohm, 1 wat
- resistor 10 ohm 1 wat al cable rojo de la bomba
- cable negro de la bomba a Vss



Los tres pines del transistor son: colector, base y emisor. La orientación está marcada en la cara impresa del transistor. Ponga la bomba con poca agua, suficiente para cubrir la unidad impulsora de 2 cm de profundidad. El circuito completo de este proyecto se muestra en la Figura 6.3.

Figura 6.3: Circuito Eléctrico de Control de la Bomba con Transistor



Ingrese el siguiente programa:

```
' Mediciones Ambientales programa 6.1
' prueba de la bomba
bucle:
  high 3
  pause 5000
  low 3
  pause 2000
goto bucle
```

La bomba debería funcionar durante 5 segundos y permanecer apagada por 2 segundos, y repetir. Por cierto, no use una batería de 9 volt para esto debido a que la bomba consume mucha corriente. Use la fuente de alimentación de 300 mA, 9 volt que viene con la Plaqueta de Educación.

6

Solución de problemas. Si funciona, ¡grandioso! Si no, éstas son un par de sugerencias.

- Golpee la bomba. No literalmente. Algunas veces la **hélice** se traba al secarse el agua, dejando depósitos de mineral en su interior. Mire la base de la bomba, y verá un hueco y las paletas del lado de la hélice. Puede destrabarlas con la punta de un clip para papeles. Además, controle que el tubo que sale de la bomba no esté puesto muy adentro, porque impide el giro de la hélice.
- Haga un puente con un cable entre colector y emisor del transistor de la Plaqueta de Educación. Esto puentea el control del programa sobre la bomba. Si la bomba sigue sin funcionar, desconéctela de la Plaqueta de Educación y conéctela a una batería de 1,5 volt. Si no funciona, la bomba puede estar rota, o los cables que la conectan pueden estar cortados.
- Si la bomba funciona, pero el circuito no responde al programa, revise el cableado. Asegúrese que el transistor está orientado correctamente en el circuito y que la base está conectada a P3. El colector debería estar en el nodo de 5 volt cercano al piezoeléctrico. Y uno de los extremos del resistor de 10 ohm debería estar en la misma fila que el emisor del transistor.
- El cableado se pone difícil. Debe controlar las conexiones cuidadosamente para asegurarse que no haya un cortocircuito con otro componente de la Plaqueta de Educación.

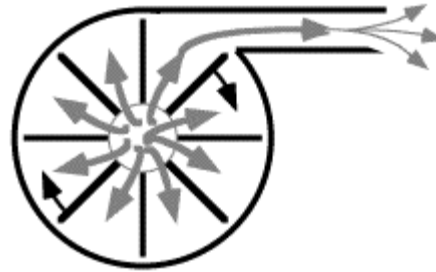
¿Qué es una...

Hélice:

Dentro de la carcasa plástica debajo del motor hay un rápido disco giratorio con paletas radiales. Las paletas toman el agua por un agujero en el centro, que se observa desde abajo, y la impulsan contra las paredes, forzándola a salir por el tubo de salida. El agua es empujada por la acción de la fuerza centrífuga. El disco giratorio con paletas se llama hélice.

La Figura 6.4 muestra una **hélice**, el disco que gira dentro de la bomba.

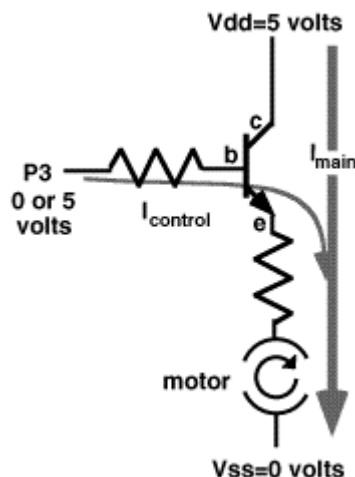
Figura 6.4: Hélice de la Bomba



6

El motor de la bomba consume aproximadamente 300 miliamperes de corriente a 3 volts. Esto es mucho más de lo que los pines del BASIC Stamp pueden suministrar. Es necesario usar un transistor para amplificar la corriente del BASIC Stamp. Hay muchas formas de usar transistores. Esta configuración se llama "seguidor emisor" o colector común. La tensión del emisor del transistor, "sigue" la tensión de la base. Cuando P3 es low, con cero volts, el emisor del transistor también está a cero volts y el motor está apagado. Pero cuando P3 es high, con 5 volts, el emisor lo sigue (4.4 volts cuando P3 está a 5 volts) y la bomba se enciende. Los 300 miliamperes que necesita la bomba, salen a través del transistor desde la fuente de alimentación, y no del pin P3. Sólo una pequeña corriente de 0,3 miliamperes es necesaria para obtener 300 miliamperes. El transistor está actuando como un amplificador de corriente. El resistor de 10 ohm 1 wat, limita la tensión aplicada a la bomba. La bomba opera como máximo con 3 volts. La Figura 6.5 muestra el funcionamiento del transistor.

Figura 6.5:
Funcionamiento del
Transistor
e emisor
b base
c colector



6

Ahora modifique el programa para que la bomba se encienda solamente mientras presionamos el pulsador.

```
' Mediciones Ambientales programa 6.1
' comprobación de la bomba
output 3
bucle:
    out3=~in1
goto bucle
```

¿Esperaba un programa más largo? Mire la instrucción `out3=~in1`. Lo que dice es, "el estado de la salida es el opuesto del de la entrada del pulsador." El símbolo `" "` significa "not". Si `in1` es cero, el botón está presionado, entonces el estado de la bomba será 1, encendida. Si `in1` es 1, pulsador liberado, entonces el estado de la bomba será 0, apagada.

Hágalo. Marque un nivel al costado del vaso, y presione el botón hasta que el nivel del agua alcanza la marca. Luego libere el botón. Intente presionar y liberar el botón de forma de mantener el nivel del agua cerca de la marca. Usted es ahora parte del lazo de realimentación, y pronto será reemplazado por la automatización. El sensor de conductividad controlará el nivel y encenderá y apagará la bomba. Pienso que este es uno de los trabajos que usted querría que se automatice pronto.

Una cosa más sobre el programa. Otra forma de escribirlo podría ser con instrucciones IF-THEN, quedando algo así:

```
' Mediciones Ambientales programa 6.1b
' comprobación de la bomba
bucle:
  if in1=0 then encender
  if in1=1 then apagar
goto bucle
encender:
  high 3
goto bucle
apagar:
  low 3
goto bucle
```

6

esta es una forma prolija de escribir el programa. Se ve muy claramente lo que sucede. Si se presiona el botón, la bomba se enciende. Si se suelta el botón, la bomba se apaga. El programa debe realizar una de las dos acciones, debido a que `in1` es 0 ó 1. Inténtelo.

El programa podría haber sido escrito usando la instrucción `branch`:

```
' Mediciones Ambientales programa 6.1c
' comprobación de la bomba
bucle:
  branch in1,[encender, apagar]
encender:
  high 3
goto bucle
apagar:
  low 3
goto bucle
```

Vea el BASIC Stamp Manual Version 1.9 (p. 247) en inglés, para mayor información sobre el comando `branch`. El programa salta a `encender` si el botón está presionado (`in1 = 0`) o a `apagar` si el botón está suelto (`in1 = 1`). Esta también es una forma clara de escribir el programa. Saltará a uno de los dos, debido a que `in1` solamente puede tomar los valores 0 ó 1.

Pruebe las diferentes formas de escribir el código. Siempre es bueno saber que hay diferentes formas de cumplir una tarea. El objetivo podría ser escribir el código tan compacto como sea posible, o lo más veloz que sea posible, o hacer el programa tan fácil de seguir en la documentación como sea posible.

6

```
' Mediciones Ambientales programa 6.2
' control de la bomba
cnt      var word
bucle:
  high 9
  count 10,133,cnt
  ^^^-----
  low 9
  debug dec cnt," umho",cr
  if cnt > 36 then apagar
  if cnt < 30 then encender
  goto bucle
apagar:
  low 3
  goto bucle
encender:
  high 3
  goto bucle
```

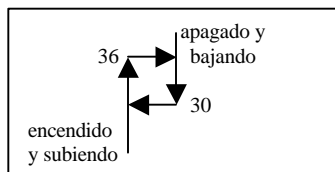
El agua subirá hasta el nivel del sensor, y luego la bomba se apagará. El nivel del agua baja hasta dejar de tocar el sensor, y entonces se enciende la bomba. Recuerde, la cuenta de count es mayor si la punta está más sumergida.

Observe la acción. ¿Qué tan seguido se enciende la bomba? ¿Cuál es la relación tiempo encendido/tiempo apagado? Este no es un sistema muy eficiente, debido a que el agua se escapa a través de la bomba mientras está apagada.

El nivel fijado para el agua, el punto donde la bomba cambia de encendida a apagada o viceversa, se llama "punto de ajuste" o umbral. Hay dos umbrales en este programa, uno para encender y otro para apagar. La bomba se enciende cuando el nivel es menor de 30, y se apaga cuando el nivel es mayor de 36. Piense lo que pasa en el código del programa cuando el nivel está entre 30 y 36. Ninguna de las instrucciones IF es verdadera, así que el programa repite el bucle sin tomar ninguna acción para cambiar el estado de la bomba. Si el motor estaba apagado, sigue apagado. Si el motor estaba encendido, sigue encendido. Esto le da 7 unidades de histéresis. Esta es una característica útil en algunos sistemas de realimentación. Por ejemplo, hay algunos tipos de motores y equipamientos que se estropean si se los enciende y apaga muy seguido. Es mejor dejar que el líquido alcance el umbral superior, y permitir que el motor descansa hasta que el líquido caiga por debajo del umbral inferior, antes de encender nuevamente el motor. Esto evita el desgaste y las roturas mecánicas. Otras veces es un requerimiento del sistema, digamos, dejar que la tierra de una planta se seque antes de volver a regar, o para lograr una crecida en el nivel de agua de una fuente. Este tipo de control, donde los umbrales están separados, es llamado histéresis. Este programa tiene 7 unidades de histéresis, de 30 a 36 inclusive.

El eje vertical es el nivel de agua, o la conductancia, debido a que a mayor nivel -- mayor conductancia. El eje horizontal es el estado de la bomba, encendida o apagada. Sobre la izquierda, el nivel es bajo, y la bomba se enciende. Sobre la derecha, el nivel es alto, y la bomba se apaga. En operación, el sistema se pasa todo el tiempo dando la vuelta al rectángulo de la histéresis. Encender hasta el umbral superior, luego apagar, y dejar caer hasta el umbral inferior.

Figura 6.6: Histéresis



Experimente con los valores del ajuste alto y bajo. Aumente el límite superior para hacer que el agua llegue más arriba en el sensor de conductividad, sin rebalsar el vaso. Observe el valor de `cnt` en la pantalla debug. Observe qué tan seguido se enciende y apaga la bomba, cuando los puntos de ajuste se juntan o cuando se separan.

¿Qué pasará con el nivel actual si cambia la conductividad del agua? Hágalo agregando una pizca de sal al agua y observe. ¡Este no es un sensor de nivel profesional!

Hay una forma alternativa del programa que evita el uso de las instrucciones `if. . then`. Es un asunto de estilo de programación. ¡Inténtelo!

```
' Mediciones Ambientales programa 6.2b
' control de la bomba
cnt    var word          ' variable para contador
low 3
bucle:
  high 9                  ' enciende el 555
  count 10,200,cnt        ' cuenta los pulsos
  low 9                   ' apaga el 555
  debug dec cnt,cr
  out3=~(cnt/36 max 1)
  goto bucle              ' vuelve al inicio
```

Cuando el valor de `count` es menor de 36, el valor de `cnt/36` será cero. Recuerde, es matemática entera, y el resultado de `cnt/36` es siempre un entero. Cuando `cnt` es mayor o igual a 36, entonces el valor de `cnt/36` será 1 o mayor. La operación adicional, `max 1`, limita el valor máximo a 1. Hay un operador "not", " " en frente de toda la expresión entre paréntesis. El resultado es tal que cuando `cnt` es menor que el punto de ajuste, `out3` es HIGH, y la bomba se enciende. Pero cuando `cnt` es mayor o igual al punto de ajuste, entonces `out3` es LOW y la bomba se apaga. Note que este programa comienza con el comando, `low 3`, que convierte a P3 en salida en estado bajo (LOW). De otra forma P3 sería una entrada.

Este programa 6.2b no tiene histéresis. La bomba está encendida para todos los valores de `count` menores de 36, y apagada para todos los valores mayores o iguales a 36. Esta es una instrucción que agrega un poco de histéresis:

```
out3  (cnt/(30 (out3 6)) max 1)
```

La bomba permanece encendida hasta que el nivel del agua sobrepasa 36, pero una vez que se apaga, no se vuelve a encender hasta que el nivel cae por debajo de 30. La histéresis se agrega mezclando el estado de la salida en el lado derecho de la fórmula. El BASIC Stamp puede hacer eso. `Out3` es una variable como cualquier otra, y su programa puede leer o fijar su valor. Piénselo un poco. Es un truco y no es tan claro como hacerlo con `if` y `goto`. Pero es la forma de compactar el código. `if` y `goto` le dan su propia forma al programa. La fórmula es una técnica avanzada para su valija de trucos.

Memoria en el BASIC Stamp, Revisión

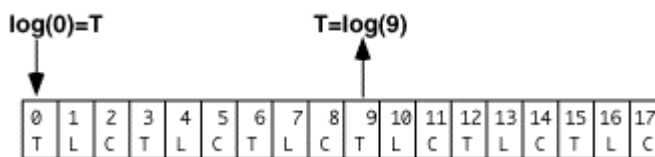
El data logger que desarrollamos en los Experimentos 4 y 5 tiene sólo 18 bytes de memoria RAM. Solo pudimos almacenar 9 registros con 2 campos en el Experimento 4, o 6 registros con 3 campos en el Experimento 5. No sólo eso, los datos desaparecían si desconectábamos la alimentación o si presionábamos el botón reset. Este es un serio inconveniente para un data logger que va a ser usado en la ciencia ambiental.

Un científico o un ingeniero puede necesitar obtener más datos, y puede pretender que no se borren prematuramente. Se deben poder almacenar y recuperar de un archivo antes de ser borrados del data logger.

Así que vamos a almacenar los datos en la memoria EEPROM en lugar de la RAM. Recuerde del Experimento 2 que hay 2048 bytes de EEPROM en el BASIC Stamp. Reservaremos 250 bytes para almacenar datos. Esto es mucho más que lo que teníamos disponible en la RAM. También podríamos reservar más, si fuera necesario. Y lo mejor de todo, nuestros datos en la EEPROM sobrevivirán aunque se interrumpa la alimentación o se presione reset.

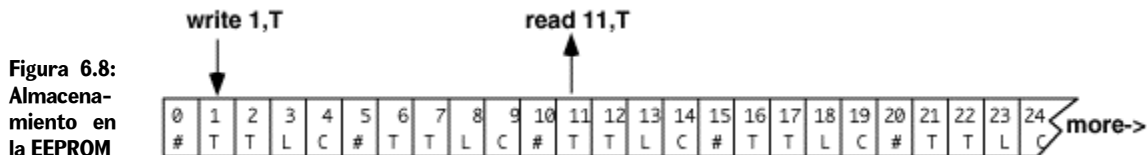
La forma de almacenar datos en la RAM es diferente que en la EEPROM. Así es en la RAM:

Figura 6.7: Almacenamiento en la RAM



Es un vector de 18 variables de un byte, desde $\log(0)$ a $\log(17)$. Ponemos los datos en cada lugar de memoria con la instrucción `log(i) TC`, y recuperamos los datos con la instrucción `TC log(i)`. El valor entre paréntesis es una variable, un puntero, como se hizo en los programas 4.4 y 5.4.

Esta es la diferencia con la EEPROM. Recuerde el Experimento 2:



Para escribir la temperatura en el lugar de memoria 1 de la EEPROM, usamos

```
write 1,TC
```

y para recuperar el valor de la posición 11 en la EEPROM, usamos:

```
read 11,TC
```

TC tiene un tamaño de un byte. Como en la RAM, podemos usar una variable puntero para ubicar el byte a leer o escribir. El siguiente programa demuestra un par de formas de reservar lugar en la EEPROM.

Inténtelo:

```
' Mediciones Ambientales programa 6.3
' reservar lugar en la eeprom
pad    data (32)          ' reserva 32 bytes, sin valor específico
log    data 1(60)         ' reserva 60 bytes, todos = 1
hey    data 72,101,121,33,32,66,83,50 ' reserva 8 bytes con sus valores
ptr    var byte           ' byte para el puntero
x      var byte           ' byte para datos de la eeprom
for ptr=0 to 7             ' apunta a 8 lugares del vector memoria
  read ptr+hey,x           ' lee el dato de la celda ptr
  debug dec x, " "        ' lo muestra con un espacio
next
debug cr
```

6

Cuando ejecute este programa, (o presione reset de la Plaqueta de Educación), debería ver los 8 números del vector "hey", aparecer en la ventana de debug.

Este programa reserva bastante espacio de la EEPROM para datos, 100 bytes para ser exactos. Hay 32 bytes de datos indefinidos (no se fija a un valor específico) comenzando en la dirección cero, luego 60 bytes de datos definidos (todos valen uno) comenzando en la dirección 32, y 8 bytes de datos numéricos entre las posiciones 92 y 99. Cada uno de estos 100 lugares de la EEPROM contienen un patrón de datos de 8 bit. Este puede representar un número como la temperatura o el nivel de luz, o puede representar una letra a imprimir en la pantalla, o podría ser el patrón del código Morse, o cualquier cosa que pueda imaginar, que entre en un patrón digital de bits.

El bucle `for. . next` lee e imprime los 8 números comenzando en la dirección de memoria "hey" de la EEPROM. Podríamos haberlo escrito así:

```
for ptr=92 to 99          ' <-- valores explícitos del puntero
  read ptr,x              ' <-- lee en esas ubicaciones
' y continúa
```

Pero lo mejor es dejar que el software del BASIC Stamp siga los detalles de los números de las direcciones de memoria. Eso hace que las modificaciones sean más fáciles.

Ahora hagamos un cambio simple a la instrucción `debug`, como sigue:

```
for ptr=0 to 7
  read ptr+hey,x
  debug x                  ' □ <-- cambie esto, quite dec y , " "
' y continúa
```

Ahora cuando ejecute el programa, el BASIC Stamp lee los mismos 8 bytes de la memoria EEPROM. Pero la instrucción `debug` imprime de a un byte en la pantalla de la PC. La pantalla los interpreta como caracteres imprimibles. Por ejemplo "72" enviado como simple byte es el código ASCII para la letra "H" (ASCII-American Standard Code for Information Interchange, Código Estándar Americano para el Intercambio de Información). Con el modificador `dec`, la instrucción `debug` interpreta al byte con el valor numérico 72, y lo envía a la pantalla como dos códigos ASCII, primero el "7" y después el "2". Si cambia los modificadores numéricos, puede ver los números en binario (72 1001000) o en hexadecimal (72 48):

```
debug x           ' muestra como texto ascii
debug dec x," "   ' muestra decimal, con un espacio
debug bin x," "   ' muestra binario, con un espacio
debug hex x," "   ' muestra hexadecimal, con un espacio
```

6

¡Inténtelo! El punto es que el patrón binario que es almacenado en la EEPROM es el mismo en cada caso. Solamente es diferente la interpretación del comando `debug` y la pantalla de la PC. Sea paciente con nosotros si ya sabe esto. Este es un punto que confunde a muchos estudiantes. Vamos a usar la EEPROM para almacenar datos numéricos, pero igualmente usaremos `debug` con el modificador decimal.

Almacenaremos cada dato como un byte en la EEPROM. Cada byte puede representar un número decimal de 0 a 255. Nuestro data logger no almacenará valores mayores. Es posible hacerlo, pero llevaría dos lugares de la EEPROM por cada valor.

Las declaraciones:

```
pad    data (32)      ' reserva 32 bytes, indefinidos
log    data 1(60)     ' reserva 64 bytes, todos=1
```

son otras dos formas de reservar espacio para datos en la EEPROM. La segunda inicializa los 60 bytes en el valor 1, mientras que la primera sólo reserva los bytes sin especificar ningún valor a almacenar.

Ahora modifique la parte central del Programa 6.3 una vez más, para mostrar el valor decimal de los 100 lugares de memoria de la EEPROM, como sigue:

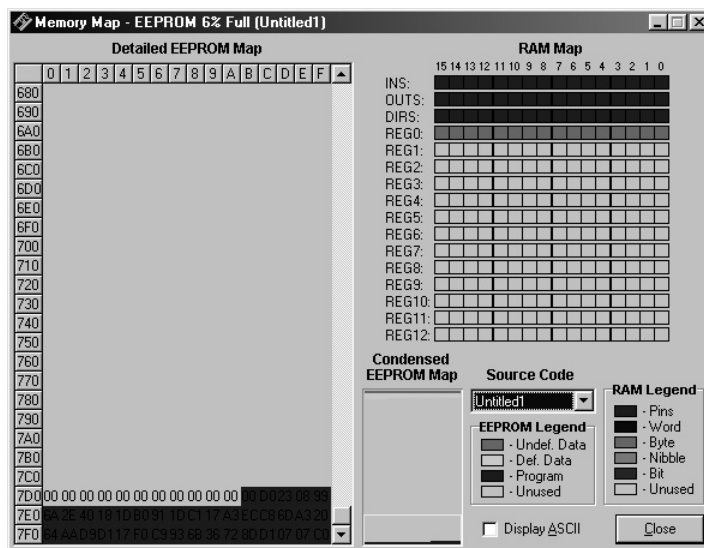
```
for ptr=0 to 99      ' lee los 100 bytes de datos
  read ptr+pad,x      ' comenzando en pad (pad=0)
  debug dec x," "     ' muestra los datos con formato decimal
next
```

Ahora debería ver 32 bytes de basura, seguidos de 60 unos, seguidos por 8 bytes de números que tienen un significado especial como texto ASCII. ¿Por qué decimos "basura"? Es debido a que los valores de los 32 bytes que usted ve, son los restos dejados por otros programas anteriores del BASIC Stamp. El programa reserva el espacio, pero no le asigna ningún valor al BASIC Stamp en esos lugares.

El editor del BASIC Stamp II tiene una característica muy útil que le permite ver directamente la distribución de la memoria. Es una herramienta invaluable para el desarrollo del programa. Cierre la ventana de debug si se encuentra activa. Si está usando el editor de DOS, STAMP2.EXE, presione ALT-M. Si está usando la versión de Windows, STAMP2W.EXE, presione CTRL-M (o seleccione mapa de memoria del menú o de la barra de herramientas). Lo que verá dependerá de la versión de editor que esté usando, pero en todos los casos hay tres cosas para ver. En la versión de Windows, las tres vistas aparecen en la misma ventana, y se llaman: "RAM Map" (mapa de RAM), "Condensed EEPROM Map" (Mapa resumido de la EEPROM), y "Detailed EEPROM Map" (Mapa detallado de la EEPROM). En la versión de DOS, aparecen en tres ventanas independientes, y pasa a través de ellas presionando la barra espaciadora.

Figura 6.9: Mapa de Memoria.

A la izquierda se ve la EEPROM, el código fuente de su BASIC Stamp y espacio extra de la EEPROM. A la derecha se ve la distribución del lugar de variables, RAM. Abajo a la derecha aparece la referencia de colores de los datos almacenados en la EEPROM y la RAM.



Mire el "RAM Map". Es la primer ventana que verá en la versión de DOS, en cambio en la versión de Windows, estará arriba a la derecha. Recuerde que la RAM está ubicada dentro del microcontrolador PIC, y almacena las variables del programa. Hay 32 bytes, (16 words, 256 bits). Los primeros 6 bytes (3 words) están dedicados a los pines de e/s del BASIC Stamp. Estas variables tienen los nombres preasignados, ins, outs y dirs. En el "RAM map" estas variables aparecen en rojo. Esto deja 13 words, 26 bytes, para las variables de nuestro programa. El Programa 6.3 tiene solamente dos variables, ambas definidas como bytes. El "RAM map" las muestra de color celeste por debajo de las variables de los pines. El resto de la memoria RAM no es usado en este programa, y se muestra en blanco (gris en Windows). Observe que "RAM Map" no muestra el valor de las variables, eso sólo ocurre cuando ejecuta el programa.

Ahora mire el "Condensed EEPROM Map". Está abajo en el centro de la ventana de la versión de Windows, pero en la versión de DOS debe presionar la barra espaciadora una vez, para poder verla. Recuerde que la memoria EEPROM está en un chip separado del microprocesador PIC. Hay 2048 bytes de EEPROM. En la parte superior están los datos en tonos de azul, y en la inferior está el código del programa en rojo. Entre el programa y los datos hay un espacio vacío que se irá llenando a medida que escribamos programas más largos y reservemos más espacio para los datos. Se preguntará qué pasa si las dos zonas se encuentran en el medio. Es simple, obtendrá el mensaje de error "out of memory" (fuera de memoria). Observe que el área de datos tiene dos sombreados. Los primeros 32 bytes en azul son los datos "indefinidos", o "vacíos" declarados con la instrucción:

```
pad    data (32)
```

Cuando ejecuta (run) su programa en el BASIC Stamp, el proceso de carga no toca esos bytes, y esa será la "basura" que vio cuando ejecutó el Programa 6.3. Por el contrario, las siguientes instrucciones crean lo que se llama "datos definidos".

```
log    data 1(60)
hey    data 72,101,121,33,32,66,83,50
```

Cuando ejecuta el programa en el BASIC Stamp, esos bytes especificados se cargan en la EEPROM junto con el programa mismo.

Ahora mire el "Detailed EEPROM Map". Si está usando la versión de DOS, presione nuevamente la barra espaciadora para ver la ventana. Esta muestra el contenido de la EEPROM byte por byte. Primero verá 32 guiones (versión DOS) o 32 ceros azul oscuro (versión Windows), seguidos de 60 ceros, seguidos por los 8 bytes específicos. La pantalla muestra números hexadecimales (de 00 a FF). En la versión de DOS, puede ver el texto ASCII (¡cuando tiene sentido!) a la derecha del mapa, y en la versión de Windows, puede presionar ALT-A para ver los datos como texto ASCII.

Note que el editor del BASIC Stamp no le muestra la "basura". Sólo la verá cuando ejecute el programa en el BASIC Stamp. Como ve, el BASIC Stamp le da muchas opciones para manipular los recursos de la EEPROM.

Ahora use la barra de desplazamiento, o las flechas (ALT-flecha en la versión de Windows) para explorar el "Detailed EEPROM Map". Cuando mira la parte inferior del mapa de memoria, verá los bytes reales del programa, como es almacenado en la EEPROM. El Programa 6.3 ocupa aproximadamente 34 bytes de memoria. El código de programa es almacenado en una forma muy comprimida, así que no busque una sencilla correspondencia entre los bytes de la EEPROM y el texto del programa.

Presione ESCAPE en el teclado (DOS) o cierre la ventana (ALT-C Windows), para regresar a la pantalla del editor. El propósito de ésta explicación fue ayudarlo a comprender la organización de la memoria del BASIC Stamp, y también demostrar una característica muy útil del software de programación del BASIC Stamp.

Almacenador de Datos (Data Logger)

Bien, manos a la obra. Hay varias cuestiones que deben ser solucionadas para obtener un data logger. En lugar de resolverlo por partes, las juntaremos para formar los objetivos de diseño.

- El data logger también controlará la bomba, manteniendo el nivel de agua del vaso. Así que realizará medición y control.
- Presionar el botón una vez, cargará los siguientes 5 bytes en la EEPROM.

ordinal 1,2,3, . . . 50	temperatura del DS1620	temperatura del AD592	luz del fotodiodo	conductancia de la punta	→ más
----------------------------	---------------------------	--------------------------	----------------------	-----------------------------	-------

- Tendrá la capacidad de adquirir datos automáticamente, tomando las lecturas con un intervalo programado desde segundos a horas. Por ejemplo, con una medición por hora y un total de 50 registros, la unidad necesitaría dos días para llenarse de datos. El intervalo es fijado en el momento de programar el BASIC Stamp.
- Debido a que vamos a usar 250 bytes para almacenar datos, y debido a que cada registro tiene cinco campos, habrá lugar para 50 registros en el archivo. El archivo puede agrandarse o achicarse de acuerdo a las necesidades de diferentes proyectos.
- El programa puede recuperar la posición en la que estaba grabando datos, después de un RESET o una falla en la alimentación. Esto se logra revisando el archivo de datos, donde el siguiente espacio vacío será marcado con un cero.
- Si presiona el botón por 1,2 segundos se introduce en la subrutina de reproducción de los datos almacenados. Esto es como en el data logger de RAM de los Experimentos 4 y 5. Después de reproducir los datos, puede continuar tomando datos en donde quedó.
- Para borrar los datos y comenzar otra vez, mantenga presionado el pulsador mientras presiona RESET.

6

- Realice toda la interacción con el usuario con el piezoeléctrico. Muestre los datos en la pantalla debug. Código Morse es opcional.

El punto de inicio para este programa es el Programa 3.4, que debería tener guardado en un disco. Este programa también usa código del Programa 5.4. Si tiene la versión Windows del software del BASIC Stamp, puede ahorrarse un poco de trabajo cortando y pegando. El programa se vuelve bastante largo, pero queremos enfatizar que se construye con un montón de piezas que usted ya conoce. Este programa simplemente las junta todas. El objetivo aquí es hacer un programa que pueda usar para los experimentos avanzados de Mediciones Ambientales.

Ingresa este programa y hágalo funcionar. Una estrategia es escribir todos los cambios, y pelearse con todos los errores después. No es una estrategia mala debido a que tiene razones para creer que el programa funcionará bien (¡y esperamos que así sea!). Otra estrategia es copiar pequeños segmentos e ir probándolos a medida que avanza. Esta es normalmente la mejor si no conoce el funcionamiento de las subrutinas. Primero verifique su Programa 3.4, luego agregue las variables adicionales, las constantes y las declaraciones de datos, luego los sensores de conductividad y de luz, luego las rutinas de control de la bomba, luego la rutina de almacenamiento de datos en la memoria, luego la adquisición de datos automática, y finalmente la subrutina para leer los datos de la memoria.

6

```
' Mediciones Ambientales programa 6.4
' data logger para 2 temperaturas, luz y conductancia
' con control simultáneo del nivel del agua

' código morse constantes y variables
dit    con    50          ' milisegundos para el dit de Morse
dit2   con    2*dit       ' constantes relacionadas a dit
dah    con    3*dit       ' ídem
mc     var    byte        ' temporaria para patrón Morse
j      var    nib         ' índice para dígitos a enviar
i      var    nib         ' índice para dits y dahs

' variables de propósito general
xm     var    byte        ' variable morse y de entrada de eeprom
x      var    byte        ' variable multipropósito
n      var    word        ' variable para el cronómetro
                        ' nota: DS1620 preprogramado para modo 2.
                        ' high 13:shiftout 15,14,lsbfirst,[12,2]:low 13

' constantes de calibración de sensores. USE SUS PROPIAS CONSTANTES
Kal    con    16428        ' para el AD592 en Kelvin con 0.22uF
lical  con    647          ' para el fotodiodo en lux con .01uF
cntcal con    1333/10      ' para conductancia en umho con 0.1uF.
```

```

' variables de los sensores
C      var word      ' para temperatura Celsius del DS1620
TK     var word      ' para temperatura Kelvin del AD592
TC     var word      ' Celsius del AD592
rct    var word      ' para el temporizador RC.
luz    var word      ' nivel de luz del fotodiodo
cnt    var word      ' para la punta de conductancia
umho   var byte      ' conductancia
mhomax var byte      ' máximo valor de la conductancia

' constantes y variables de almacenamiento
interval con 600      ' intervalo de almacenamiento en décimas de seg.
nfls   con 5          ' número de campos por registro
nrecs  con 50         ' número de registros en el archivo
logsiz con nfls*nrecs ' tamaño del archivo en bytes
pad    data (16)      ' relleno para evitar desgaste de la memoria
log    data 0(logsiz) ' bytes reservados en la eeprom para datos
ptr    var byte       ' puntero a los datos del archivo
                        ' rutina que usa también la variable xm

outs=%00000000001000000      ' ahora especifica outs y dirs.
      'fedcba9876543210
dirs=%1111101111111101

P0 es salida para el piezoeléctrico
P1 es entrada para el pulsador
P3 bajo para la bomba
P5 es salida baja para descargar C del AD592
P6 es salida alta para descargar C del fotodiodo
P9 enciende y apaga el 555
P10 es entrada para conductividad (555)
P13-15 salida para DS1620 SPI
los pines sin usar son salidas bajas

inicio:                      ' programa inicia aquí

      ptr=-5                ' puntero=-5 para prepararse para subrutina
findptr:                     ' busca la siguiente ubicación libre de la eeprom
      ptr=ptr+5              ' apunta a un registro
      read ptr+log,x         ' lee un byte
      if x>0 AND ptr<logsiz then findptr' si x no es cero, no es un registro libre
                                ' también comprueba si se llenó, ptr=logsiz
                                ' continúa si encuentra un registro libre
                                ' ptr apunta a ese lugar vacío
                                ' borra los datos con botón+RESET presionados
If in1=0 then borrar         ' muestra el puntero y la dirección base
debug ? ptr                 ' con este mensaje
debug "RESET+botón=borrar",cr
freqout 0,20,1900          ' sonido indica que está funcionando

```

```

principal:      ' programa principal
  n=0           ' inicia el cronómetro
clik:          ' repite por botón o tiempo
  if n=interval then leedato ' toma datos a intervalos
  gosub bomba   ' actualiza el estado de la bomba
  n=n+1         ' cuenta el tiempo
  if in1=1 then clik ' puede presionar el botón para tomar datos
  freqout 0,5,2550 ' indicador toma de datos por botón
  n=0          ' reinicia el cronómetro para click largo
clik1:         ' incrementos de 0,1 segundos para click largo
  pause 100    ' salta a subrutina reproducir después de 1,2 seg
  if n>12 then reproducir ' incrementa cronómetro
  n=n+1        ' repite hasta que suelta botón o rebasa tiempo
  if in1=0 then clik1 ' continúa si liberó el botón

leedato:       ' lee y almacena dato
freqout 0,20,3400 ' sonido indicador
low 3          ' apaga la bomba, incondicional, mientras lee
xm=ptr/5+1     ' pone el puntero en memoria
gosub writedata ' escribe en la eeprom
debug dec xm," " ' imprime en la pantalla

DS1620:        ' código de sensor de temperatura DS1620
  high 13      ' selecciona el DS1620
  shiftout 15,14,lsbfirst,[238] ' comando "iniciar conversión"
  low 13       ' finaliza el comando
  pause 450    ' retardo para la conversión
  high 13     ' selecciona el DS1620
  shiftout 15,14,lsbfirst,[170] ' comando "obtener datos"
  shiftin 15,14,lsbpre,[x]      ' obtiene los datos
  low 13                       ' fin del comando
  C=x/2                        ' convierte datos en grados C
  xm=C                         ' morse espera datos en variable, xm
  gosub writedata              ' escribe el dato C
  debug dec xm,tab             ' lo muestra en la pantalla debug
  'gosub morse                  ' y como código morse (opcional)

AD592:         ' código del sensor de temperatura AD592
  rctime 5,0,rct ' lee AD592
  low 5          ' pone pin en bajo, descarga el capacitor
  TK = Kal/rct*10 + (Kal//rct*10/rct)
  ' calcula Kelvin
  TC = TK-273    ' y convierte en grados C
  xm=TC          ' morse espera datos en variable, xm
  gosub writedata ' escribe datos en la eeprom
  debug dec xm,tab ' los muestra en la pantalla debug

```

```

' gosub morse          ' y en código morse (opcional)

Fotodiodo:
  Rctime 6,1,rct      ' lee el fotodiodo
  high 6              ' descarga el capacitor
  luz=65535/rct*/lical ' calcula lux
  xm=luz/2 max 255    ' listo para almacenar en eeprom
  gosub writedata     ' almacena en la eeprom
  debug dec luz,tab    ' muestra en la pantalla debug

Conductancia:
  xm=mhomax           ' almacena el máximo valor para la rutina bomba
  gosub writedata     ' escribe en eeprom
  debug dec xm,cr      ' muestra máxima conductancia en umho
  mhomax=0            ' reinicia el acumulador

goto principal        ' vuelve al inicio
end                   ' fin del programa principal

'----- más rutinas goto -----

borrar:
  freqout 0,400,2550,1900 ' sonido indicador
  for x=0 to ptr step 5   ' barre los registros
  write x+log,0           ' haciéndolos cero
  next
  debug cls,"datos borrados",cr ' limpia la pantalla y muestra el mensaje
  ell:                    ' espera a que se libere el botón
  if in1=0 then ell
  goto inicio

reproducir:
  low 3                  ' apaga la bomba incondicional
  freqout 0,50,2550      ' sonidos indicadores
  freqout 0,100,3400
  debug cls,"datos almacenados",cr ' mensaje y unidades
  debug "#",32,"C",tab,"C",tab,"lux",tab,"umho",cr
  ptr=0                  ' puntero en cero
pb0:
  read ptr+log,x         ' lee registro
  if x=0 then pb1        ' si es cero, es un registro vacío
  debug dec x," "        ' muestra número de registro
  read ptr+1+log,C       ' lee temperatura (DS1620)
  read ptr+2+log,TC      ' lee temperatura (AD592)
  read ptr+3+log,luz     ' lee luz
  read ptr+4+log,umho    ' lee conductancia
  luz=luz*2              ' restaura unidades de luz

```

```

debug dec C,tab,dec TC,tab,dec luz,tab,dec umho,cr
ptr=ptr+5          ' apunta al siguiente registro
goto pb0           ' regresa a up
pbl:               ' espera a que se libere botón
if in1=0 then pbl
  debug rep "-"\31,cr  ' línea horizontal
goto principal     ' regresa al bucle principal

' ----- subrutinas -----

morse:             ' envía un byte xm en código morse
for j=1 to 0       ' envía 2 dígitos, decenas y luego unidades.
  mc = xm dig j    ' extrae el dígito (j+1)
  mc = %11110000011111 >> mc  ' patrón para código morse
  for i=4 to 0     ' 5 dits y dahs
    freqout 0,dit2*mc.bit0(i)+dit,1900 ' emite patrón de mc
    pause dit      ' silencio corto
  next            ' siguiente i, dit o dah de cinco
  pause dah       ' silencio entre dígitos
next             ' siguiente j, dígito de dos
return           ' regresa al programa

Bomba:
  high 9          ' enciende el 555
  count 10,100,cnt  ' cuenta la frecuencia
  low 9           ' apaga el 555
  umho=cnt*cntcal/100 max 255  ' calcula umho
  mhomax=umho min mhomax  ' valor máximo de umho
  if umho>99 then encender  ' umbral para apagar la bomba
  if umho<50 then apagar    ' umbral para encenderla
return              ' llega aquí si umho está entre los umbrales
encender:
  high 3          ' enciende
return
apagar:
  low 3           ' apaga
return
  out3=~(umho/(out3*49+50) max 1)  ' controla la bomba, opción

writedata:
  if ptr=>logsiz then writeout  ' controla fin de archivo
  write ptr+log,xm  ' escribe este campo
  ptr=ptr+1        ' apunta siguiente campo
writeout:
  return

```

Una vez que tiene el programa funcionando, adquiera algunos datos para asegurarse que funcione. Debe funcionar igual que el data logger de RAM que hizo en los Experimentos 4 y 5. Revise los objetivos de diseño para saber cómo se supone que trabaja. El intervalo de medición está fijado inicialmente en 1 minuto (interval con 600) en décimas de segundo. Si está trabajando en un curso, su profesor puede tener otras sugerencias para el intervalo y por el tamaño y ubicación del archivo de almacenamiento.

Notas y Soluciones de Problemas

- ¿Comportamiento extraño? Si el programa reinicia frecuentemente, ejecútelo con la bomba de agua desconectada.
- ¿Plaqueta de Educación caliente? La bomba consume bastante potencia. El resistor de 10 ohm que está en serie con el motor en la Plaqueta de Educación se entibiará, como así también el regulador de tensión que alimenta todo el circuito desde la Plaqueta de Educación. Es de esperar que la temperatura que marca el sensor de temperatura DS1620 aumente cuando se alimenta durante mucho tiempo la bomba. Tóquelos cuidadosamente y vea.
- ¿DS1620 muerto? Si el DS1620 deja de responder (ve solamente ceros en la segunda columna de la pantalla debug), aumente el retardo de la subrutina del DS1620 de 450 a un valor mayor. Un retardo es necesario después de enviar la orden de comenzar la conversión analógica-digital del DS1620. Si se fija atentamente en el programa 3.4 verá que el código "comenzar las conversiones" se envía solamente una vez, al principio del programa. Desafortunadamente, el DS1620 es muy sensible al ruido generado por la bomba. Como solución rápida, apagamos el motor, y luego enviamos el comando "iniciar conversiones". En un proyecto real de ingeniería, este comportamiento sería un problema, y se realizaría un esfuerzo extra para aislar y resolver el problema.
- ¿Problemas de calibración? Recuerde que si cambia los capacitores, también debe revisar la calibración. Asegúrese de tener un capacitor de 0.22 F para el sensor de temperatura AD592, y el de 0.01 F para el fotodiodo, y 0.1 para el sensor de conductancia. Usted puede, por ejemplo, querer usar el sensor de luz con la luz del sol, de modo que debe cambiar el capacitor de 0.22 f y también la constante de calibración.

6

Las constantes son:

Kal	del Experimento 3	valor	
lical	del Experimento 4	valor	interior
lical	del Experimento 4	valor	exterior
cntcal	del Experimento 5	valor	

- ¿Tareas múltiples en tiempo crítico? Note que la subrutina de la bomba del final del programa es llamada desde un par de lugares. En particular, es llamada repetidamente desde los bucles del pulsador. Esto es debido a que la operación de la bomba es una tarea de tiempo crítico. Esto es lo que deben hacer muchos programas complicados. Deben realizar múltiples tareas prácticamente a la vez. En este caso controlar el botón y mantener el nivel de agua. El programador debe asegurarse que ambas tareas se realizan en tiempo y forma. La lectura de conductancia de la subrutina de la bomba tarda 1/10 de segundo, y retarda todo el proceso. El BASIC Stamp es una computadora relativamente lenta, y no puede hacer verdadera "multitarea". El BASIC Stamp en este caso está revisando el pulsador y la bomba aproximadamente 10 veces por segundo. Puede ver bajar el nivel del agua en el vaso cuando el programa toma las lecturas de los sensores. Tal vez uno o dos segundos de retardo es aceptable aquí. Pero en otros sistemas, puede ser un gran problema y usted puede necesitar un microcontrolador más rápido.
- ¿La potencia de la bomba arruina las lecturas de los sensores? Note el comando `low 3` cerca del inicio de la subrutina `leedato`. La bomba es apagada incondicionalmente mientras se leen los sensores. De otra forma, el ruido o el consumo de la bomba podrían afectar las lecturas. Intente ver lo que queremos decir. Comente el comando `low 3` (ponga un apóstrofe al inicio de la línea), y ejecute el programa nuevamente. Durante un intervalo que la bomba esté encendida, presione el botón y observe las lecturas en la pantalla debug. Luego presione el pulsador pero cuando la bomba está apagada, y compare ésta lectura con la anterior.
- ¿Por qué `mhomax`? La lectura de conductancia requiere una explicación. La conductancia es usada para controlar el nivel de agua. Así que la subrutina de la bomba lee el valor de la conductancia a menudo. Esa subrutina almacena el valor máximo de la conductancia que ha detectado.

```
mhomax=umho min mhomax
```

Que quiere decir, "asígnale a `mhomax` el valor mayor, la conductancia (`umho`) o el valor actual de `mhomax`" (min, debido a que `mhomax` es el valor mínimo garantizado). Por ejemplo, si el valor actual de `umho` es 67, y el valor actual de `mhomax` es 65, el nuevo valor de `mhomax` será 67. Es `mhomax` el valor que se almacenará en la rutina de conductancia. Luego `mhomax` es puesto a cero para que pueda acumular un nuevo valor máximo en el intervalo siguiente.

- ¿5 y el siguiente libre? La subrutina `writedata` es llamada desde varias partes de la rutina `leedato`. Primero es llamada para almacenar el número del apuntador, y luego una vez para cada sensor. Al final de la rutina `leedato`, el puntero es dejado apuntando al siguiente byte libre, donde comienza el siguiente registro que almacenará los datos una vez cumplido el intervalo de tiempo, o al presionar el botón.

Figura 5.10: Escribiendo Datos



6

En un data logger profesional, se debería almacenar el tiempo junto con los datos (que cumpla con Y2K, por supuesto).

- ¿Buscar y encontrar? La subrutina `findptr` busca por todos los registros del archivo de datos, buscando solamente en los lugares reservados para registros numéricos: (ptr 0, 5, 10, 15, ... ,245). Si encuentra un cero en uno de esos lugares, entonces ese es el siguiente registro disponible donde se guardarán los datos. Esto sucederá cada vez que presione RESET en la Plaqueta de Educación, o cuando se encienda el circuito. Poniendo marcas en el archivo de datos, puede reconstruir dónde estaba. Note que después de esto, está la instrucción que se fija si el pulsador está siendo presionado, justo después de RESET. Si es así el BASIC Stamp salta a la subrutina que borra el archivo de datos. Lo que realmente hace es poner ceros en todos los campos del primer registro. De esta forma la subrutina `findptr` comienza desde el principio. La rutina `borrar` no borra realmente todos los datos, sino los cinco primeros.
- ¿Reproducir? La rutina que reproduce los datos es la inversa de la que los pone en la memoria.

Figura 5.11: Leyendo Datos



La subrutina lee 5 campos, y luego incrementa el apuntador al siguiente grupo de cinco. Al final, el puntero se deja apuntando al siguiente registro libre.

- ¿Pulsador? El código que detecta cuando presiona y libera el pulsador debería serle muy familiar a esta altura. Aquí, cada paso por el bucle, mientras espera que suceda algo, el programa revisa la bomba y la enciende o apaga de ser necesario. La rutina que lee la conductividad tarda 0,1 segundos, lo que también retarda el bucle del pulsador.
- ¿Tiempo? El intervalo de tiempo no es muy preciso. Puede intentar calibrarlo cambiando las constantes.

10 segundos
1 minuto
10 minutos
18000, 30 minutos
36000, 1 hora

6

Para calibrar el tiempo, necesita un cronómetro. Fije el intervalo en, por ejemplo, 1 minuto, y contrólole (usando el sonido indicador) para ver como se comporta con interval 6000. Si el valor real se excedió un 1 por ejemplo, compense este error con interval 5940.

- ¿Arruinar la EEPROM? Note que se reservaron 16 bytes al principio de la memoria. El propósito de esto es evitar fatigar estos bytes. Recuerde que la EEPROM tiene una vida finita en términos del número de veces que puede ser escrita. Soporta un millón o más de reescrituras. Cuando usa la EEPROM para almacenar datos, debe recordar esto. Lleva mucho tiempo llegar a un millón de veces, pero tenga en cuenta que el programa puede rescribir muy rápidamente.
- ¿Qué hay en la memoria? En el editor de la PC, presione ALT-M (DOS) o CTRL-M (Windows). Recuerde la discusión sobre el "RAM Map" y el "EEPROM Map". Esto le ayuda a visualizar cómo están siendo usadas las variables y el espacio de la EEPROM. ¿Qué fracción de la EEPROM ocupan el programa y los datos?

Otras Investigaciones

- 1) Estudie el caudal de la bomba, y la altura de columna de agua que es capaz de soportar.
- 2) Haga un calentador solar poniendo un tubo de cobre negro bajo un vidrio o en una botella, expuesto al sol. Encienda la bomba para hacer circular el agua hasta un tanque. Controle la temperatura del agua y la intensidad solar. Use la temperatura del agua y la intensidad solar para decidir cuándo encender y apagar la bomba automáticamente.
- 3) Intente usar PWM (modulación de ancho de pulso) con la bomba, es decir, enciéndala y apáguela rápidamente:

```
x con 5
bucle:
high 3
pause x
low 3
pause 10-x
goto bucle
```

La constante 5 hace que la bomba esté la mitad del tiempo encendida y el resto apagada. Así que la bomba trabaja a la mitad de la velocidad. Cambia tan rápido de encendido a apagado que no lo percibirá. Cambie la constante para ver qué pasa con otras relaciones de encendido-apagado. También puede probar el comando propio del BASIC Stamp II llamado PWM.

El data logger puede ser usado en los experimentos de laboratorio. Modifique el programa, de forma que haga lo que usted quiera.



¡Desafío!

- 1) Escriba un programa simple que encienda la bomba cuando el botón es presionado una vez, y quede encendida hasta que lo presione nuevamente. (Push on, push off presiona enciende, presiona apaga).
- 2) Usted está a cargo de una fuente pública que se supone que funciona solamente de día, y solamente cuando el día está despejado y la temperatura es mayor de 70 grados Fahrenheit. Escriba un programa que controle la fuente.
- 3) Haga un programa que dibuje un gráfico de conductancia en la pantalla debug. Deje que el programa haga rebalsar el vaso, y luego deje caer el nivel por debajo de las puntas del sensor, antes de comenzar a bombear otra vez. Todo mientras grafica las lecturas de conductividad en la pantalla debug.
- 4) Una quinta ictícola debe mantener alto el nivel de agua en un tanque, filtrarla, y mantenerla aireada, y controlar las condiciones que sean peligrosas para los peces. Escriba un programa que mantenga el nivel del agua del vaso subiendo y bajando, pero que suene una alarma si la temperatura del agua excede los 80 grados C, o si el agua deja de fluir por alguna razón, o si la conductividad del agua cambia drásticamente.
- 5) Algunas veces en configuraciones del mundo real, es recomendable saber cuánto tiempo (o que porcentaje) un motor está encendido y cuánto apagado. Esto ayuda con el mantenimiento y el planeamiento de la eficiencia de energía. Modifique el programa 6.2 de forma que muestre el porcentaje de tiempo que la bomba permanece encendida. Podría agregar este dato a su programa de adquisición de datos, como una indicación de cuánta agua fue usada.

6