

# **Analógico y Digital Básicos**

**Guía del Estudiante**

Version 1.2

PARALLAX 

## **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 [aalvarez@parallax.com](mailto:aalvarez@parallax.com), así podemos continuar mejorando la calidad de nuestra documentación.

## **GARANTÍA**

Parallax garantiza sus productos contra defectos en sus materiales o debidos a la fabricación por un período 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. Clientes fuera de EEUU contactar primero Parallax a la dirección [stampsinclass@parallax.com](mailto:stampsinclass@parallax.com).

## **GARANTÍA DE 14 DÍAS DE REGRESO DEL 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 ha sido alterado o dañado, o en compras realizadas fuera de EEUU.

## **DERECHOS DE COPIA Y MARCAS REGISTRADAS**

Esta documentación tiene derechos de copia Copyright 1999/2003 por Parallax, Inc. Al descargar este libro por internet usted acepta los derechos de copia y se compromete a usar este material solamente con productos comercializados por Parallax, Inc. Cualquier otro uso no es permitido y representa una violación de los derechos de copia y propiedad intelectual de Parallax, Inc. No se permite la reproducción con fines comerciales. Se permite la reproducción con fines educativos, siempre y cuando los estudiantes no deban abonar más que el costo de la duplicación. BASIC Stamp 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: [aalvarez@parallax.com](mailto:aalvarez@parallax.com)  
Web: <http://www.parallax.com>

## SITIOS WEB Y LISTAS DE DISCUSIÓN

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 ParallaxenEspanol dirijase a [www.parallax.com](http://www.parallax.com), o directamente en: <http://espanol.groups.yahoo.com/group/ParallaxenEspanol/>

También mantenemos una lista exclusiva para educadores que usan el BASIC Stamp en el aula. Los docentes pueden unirse a esta lista. Debido a que es una lista internacional, toda la información se intercambia en esta lista es en inglés.

También mantenemos muchas listas de discusión en inglés que son un recurso muy importante de soporte técnico. Como los foros intercambian mensajes en inglés, dejamos la explicación en el idioma original:

- BASIC Stamps – Con más de 3800 suscriptos, esta lista es extensamente utilizada por Ingenieros, hobbistas y estudiantes, quienes comparten sus proyectos y preguntas acerca del BASIC Stamp.
- Stamps in Class – Creada por educadores y estudiantes, esta lista tiene 650 suscriptos, quienes discuten el uso del material de Stamps in Class en sus cursos. La lista provee la oportunidad a los estudiantes de hacer preguntas a los profesores también.
- Parallax Educators – Este grupo de 200 miembros consta exclusivamente en educadores y aquellos quienes contribuyen con el desarrollo del curriculum de Stamps in Class. Parallax creo este grupo con el fin de obtener soporte en el desarrollo de nuestro curriculum y proveer un foro para los educadores que desarrollan las Guías para Profesores.
- Toddler Robot – Un cliente creó esta lista de discusión para compartir las aplicaciones y programación del robot Toddler de Parallax .
- SX Tech – Lista de discusión para programar el microcontrolador SX microcontroller de Parallax con las herramientas del lenguaje assembly, compiladores (BASIC y C). Cuenta aproximadamente con 600 miembros.



## Índice

<b>Prefacio.....</b>	<b>v</b>
Destinatarios y Guías Para Profesores .....	v
Derechos de Copia y Reproducción .....	vi
Traducciones .....	vi
Traducción al Español .....	vii
Colaboradores especiales .....	vii
<b>Capítulo 1: Tensión Analógica y Estados Binarios .....</b>	<b>1</b>
Introducción a Analógico y Digital.....	1
Componentes Requeridos .....	2
Armando el Comparador.....	7
Programando el Proyecto .....	9
¿Qué Aprendí? .....	16
Preguntas.....	17
Desafío.....	17
¿Por qué aprendí esto? .....	18
¿Cómo puedo aplicarlo? .....	18
<b>Capítulo 2: Introducción al Proceso de Bits.....</b>	<b>19</b>
Comunicación básica.....	19
Componentes Requeridos .....	19
Construyendo el circuito .....	20
Programando el Proyecto .....	22
Transmisión Serie y Paralelo .....	31
Programación para Enviar Datos en Serie.....	32
¿Qué aprendí? .....	38
Preguntas.....	39
Desafío.....	39
¿Por qué aprendí esto? .....	40
¿Cómo puedo aplicarlo? .....	40
<b>Capítulo 3: Conversión Analógica a Digital Básica .....</b>	<b>41</b>
Construya su Propio Voltímetro Digital de CC.....	41
Componentes Requeridos .....	42
El Potenciómetro, una Fuente de Tensión Variable.....	42
El Circuito Integrado ADC0831. Un Conversor Analógico Digital de 8-bits .....	43
Constrúyalo .....	45
Prográmelo.....	46
Repaso de Conversión Binaria a Decimal .....	53
Cálculo de la Tensión.....	56
Resolución .....	63

Calibración .....	64
¿Qué aprendí? .....	65
Preguntas.....	66
Desafío.....	66
¿Por qué aprendí esto? .....	67
¿Cómo puedo aplicarlo? .....	67
<b>Capítulo 4: Conversión Digital a Analógica Básica.....</b>	<b>69</b>
Construcción de una Red Resistiva en Escalera .....	69
Componentes Requeridos .....	70
Constrúyalo.....	71
Prográmelo .....	72
Direccionamiento .....	78
El Seguidor de Tensión.....	83
¿Qué aprendí? .....	88
Preguntas.....	89
Desafío.....	89
¿Por qué aprendí esto? .....	90
¿Cómo puedo aplicarlo? .....	90
<b>Capítulo 5: Señales que Varían en el Tiempo .....</b>	<b>91</b>
Componentes Requeridos .....	92
Prográmelo .....	93
La Onda Cuadrada .....	99
La Sinusoide y la Modulación de Ancho de Pulso (PWM).....	103
Programa de Notas Musicales.....	104
¿Qué aprendí? .....	106
Preguntas.....	107
Desafío.....	107
¿Por qué aprendí esto? .....	107
¿Cómo puedo aplicarlo? .....	108
<b>Capítulo 6: Capturando Datos sobre Frecuencia .....</b>	<b>109</b>
Componentes Requeridos .....	109
Constrúyalo.....	111
La Salida .....	113
Prográmelo .....	116
¿Qué aprendí? .....	121
Preguntas.....	122
Desafío.....	122
¿Por qué aprendí esto? .....	123
¿Cómo puedo aplicarlo? .....	123
<b>Capítulo 7: Digital a Analógico Fácil con PWM.....</b>	<b>125</b>

Componentes Requeridos .....	128
Constrúyalo .....	129
Programelo .....	131
¿Qué aprendí? .....	139
Preguntas .....	140
Desafío .....	140
¿Por qué aprendí esto? .....	141
¿Cómo puedo aplicarlo? .....	141
<b>Capítulo 8: Fotómetro con constantes de Tiempo RC .....</b>	<b>143</b>
Componentes Requeridos .....	146
Constrúyalo .....	146
Programelo .....	148
Matemática involucrada .....	152
¿Qué aprendí? .....	154
Preguntas .....	155
Desafío .....	155
¿Por qué aprendí esto? .....	156
¿Cómo puedo aplicarlo? .....	156
<b>Apéndice A: Listado de Componentes .....</b>	<b>159</b>
<b>Apéndice B: Código de Color de Resistores.....</b>	<b>161</b>
<b>Índice Alfabético .....</b>	<b>163</b>





## Prefacio

---

La computadora personal nos llevó a una nueva era de la sofisticación electrónica. Ahora poseemos una inmensa cantidad de poder de computación digital localizado sobre nuestro escritorio. Las computadoras trabajan bien cuando están conectadas a alguna otra computadora y los datos pueden ser transferidos completa y eficazmente de una máquina a otra.

Sin embargo, en el instante en que nosotros queremos conectar la computadora digital con algún dispositivo del “mundo real”, vamos a necesitar diseñar un circuito, llamado interfase, para transmitir la información proveniente del dispositivo analógico a la computadora digital. En muchos casos, esto implica la conversión de tensión analógica a la representación de ese valor como tensión digital

Este libro de experimentos de Stamps en Clase, explorará muchos de los principios básicos de interfases entre dispositivos analógicos y microcontroladores digitales. Muchas veces eso implica el uso de comandos de fácil uso, contruidos dentro del BASIC Stamp, y otras veces requiere el uso de un “conversor analógico a digital”.

¿Por qué deberíamos estar interesados en convertir de analógico a digital? Muchos aspectos diferentes de nuestras vidas dependen de éste proceso de conversión. Algunos de ellos no son muy críticos para nuestra supervivencia, como los reproductores de CDs, los sistemas telefónicos y de música. Otros, sin embargo, son esenciales. El equipamiento médico frecuentemente requiere una conversión de analógico a digital y de digital a analógico.

El manual Analógico y Digital Básicos será revisado y actualizado continuamente basándose en correcciones enviadas por estudiantes y profesores. Si usted quisiera colaborar con el autor o tiene ideas para mejorar el libro, entonces envíelas a [stampsinclass@parallax.com](mailto:stampsinclass@parallax.com). Nosotros haremos lo mejor para incluir sus ideas y asistirlo en todo lo que sea posible.

### DESTINATARIOS Y GUÍAS PARA PROFESORES

Este libro fue creado para mayores de 17 años. Esta guía puede ser usada completamente como un libro de introducción a Analógico y Digital en el laboratorio o como referencia para obtener explicaciones detalladas acerca del hardware y técnicas usadas en otras guías para alumnos del programa Stamps en Clase.

Las respuestas de los experimentos no presentan ninguna dificultad técnica y pueden ser resueltos con un poco de paciencia. Para evacuar dudas en la resolución de los ejercicios recomendamos a particulares, docentes y alumnos suscribirse a la lista de discusión ParallaxenEspanol:

(<http://espanol.groups.yahoo.com/group/ParallaxenEspanol>).

Los instructores además podrían participar en el Forum para Educadores para obtener soporte (solamente en inglés). Para enrolarse en este foro hay que enviar un email a [stampsinclass@parallax.com](mailto:stampsinclass@parallax.com).

Las Guías para Docentes se cargarán en ese foro exclusivo para educadores a medida que se encuentren disponibles. Las Guías para Docentes solamente estarán disponibles en inglés.

## **DERECHOS DE COPIA Y REPRODUCCIÓN**

Parallax le garantiza a cada persona derechos condicionales de descarga, duplicación y distribución de este texto sin nuestro permiso, para ser usado con productos Parallax. 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 solicitar nuestro permiso.

## **TRADUCCIONES**

Los textos educativos de Parallax pueden ser traducidos a otros idiomas con nuestro previo permiso (diríjase a nuestro e-mail: [stampsinclass@parallax.com](mailto:stampsinclass@parallax.com)). Si usted planea hacer alguna traducción, por favor contáctese con nosotros para que podamos proveerle los documentos en MS Word, imágenes, etc. También mantenemos un grupo de discusión para traductores de Parallax, al que usted se puede unir. De esta forma nos aseguramos de mantenerlo al tanto de las actualizaciones.

## **TRADUCCIÓN AL ESPAÑOL**

Traducido al español por Ana Lusi de Alvarez. Si encuentra errores en el texto, por favor contáctese a ésta dirección: [anamlusi@yahoo.com.ar](mailto:anamlusi@yahoo.com.ar) , para poder mejorar la calidad de la documentación en español.

## **COLABORADORES ESPECIALES**

La versión original de este texto fue escrita en su mayor parte por Andy Lindsay, basado en un manuscrito de Matt Gilliland, autor original de “¿Qué es un Microcontrolador?” y de los populares Microcontroller Application Cookbooks. Andy escribió este texto durante su último año en la Universidad del Estado de California, en Sacramento, donde estudió Ingeniería Eléctrica y Electrónica. Este fue el primero de tres libros de Stamps en Clase que él revisó y/o escribió. Cuando no está escribiendo material de educación, Andy realiza ingeniería de productos para Parallax.

La versión 1.2 contiene imágenes mejoradas por Rich Allred. Los programas están actualizados a PBASIC 2.5 y la edición general fue realizada por Aristides A. Alvarez.



# Capítulo 1: Tensión Analógica y Estados Binarios

---

Esta serie de experimentos presentan la electrónica analógica y digital. ¿Qué significa esto? En “¿Qué es un Microcontrolador?” aprendimos que analógico es un “valor que varía en forma continua”. También se puede interpretar a la electrónica analógica como una analogía de la naturaleza.

Hay muchos valores que varían en forma continua en la naturaleza, tales como movimiento, nivel de luz, y sonido. La posición de una puerta a medida que se abre es un buen ejemplo de un valor que varía de forma continua (sin saltos). A medida que la puerta pasa de completamente cerrada a completamente abierta, recorre todos los valores intermedios. En un instante del recorrido, estará abierta a 1/3 de su recorrido. En otro momento, estará abierta a la mitad, y así hasta abrirse completamente.

## INTRODUCCIÓN A ANALÓGICO Y DIGITAL

Digital simplemente significa representado por dígitos. Piense la cantidad de veces en el día que encuentra valores analógicos que están representados por dígitos. La temperatura es de 27,8 grados. El límite de velocidad es de 45 kilómetros por hora, etc. Como era de esperar, la electrónica digital representa los valores con dígitos.

El término digital también se usa cuando nos referimos a dispositivos binarios tales como los circuitos que hacen funcionar una calculadora, el microprocesador de una computadora, o el microcontrolador BASIC Stamp. Todos son dispositivos digitales. El sistema binario es un tipo de sistema digital que usa dos dígitos, 0 y 1.

Los experimentos de “¿Qué es un Microcontrolador?” trataron principalmente el uso del BASIC Stamp en aplicaciones binarias. Este libro se enfocará principalmente en el uso del BASIC Stamp en aplicaciones analógicas. El primer experimento introduce el concepto de tensión analógica.

En este primer Capítulo, construiremos un circuito que produce una tensión analógica en su salida. Recuerde que la tensión analógica varía en forma continua. El circuito tendrá salida regulable entre 0 y 5 Volts. También construiremos un circuito llamado seguidor de tensión que emplea esta tensión analógica para alimentar el circuito de un LED.



**Tensión:** El Volt es una unidad fundamental de medición eléctrica que debe su nombre al científico del siglo XVIII, Alessandro Volta, siendo la tensión el valor medido en Volts.

Encontramos esta unidad de medición cuando compramos baterías, como las de 9 Volt (CC) que puede ser usada para alimentar la Plaqueta de Educación. Dentro de una batería hay dos reacciones químicas que están separadas por una barrera. Una de las reacciones crea un excedente de electrones, mientras que la otra crea la falta de estos.

Ambos lados de la barrera están conectados a los terminales positivo y negativo de la batería. Si se les facilita un camino o circuito a través de la barrera, los electrones tienen el potencial de realizar trabajo entre los terminales. El Volt indica este potencial para realizar trabajo. El Volt también es conocido como unidad de potencial eléctrico.

La tensión analógica también estará conectada a uno de los pines de E/S del BASIC Stamp, configurado como entrada. Esta entrada binaria puede ser usada para medir las variaciones de la tensión analógica. El lenguaje PBASIC se usará para programar el BASIC Stamp de forma que controle un circuito de LED binario, que indicará cuándo se detecten variaciones en la entrada.

La ventana Debug también es una herramienta útil para mostrar los datos que el BASIC Stamp recibe y envía. Será usada para monitorear el valor binario que recibe el pin de entrada, a medida que varía la tensión analógica.

### **Componentes Requeridos**

En cada experimento necesitará un BASIC Stamp 2 y una Plaqueta de Educación o simplemente una HomeWork Board, conectada a una PC IBM-compatible Win95/98/NT4.0/XP con el último software editor del BASIC Stamp disponible ya instalado. Además, en este experimento necesitará los siguientes componentes:

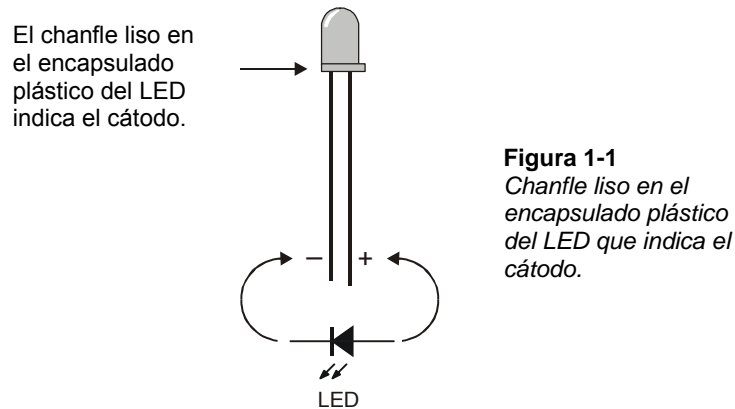
- (2) Resistores de 470 Ohm
- (2) LEDs rojos
- (1) Potenciómetro de 100 k $\Omega$ .
- (6) Cables de interconexión.
- (1) Amplificador Operacional LM358

En todos los experimentos, armaremos circuitos basándonos en diagramas de circuitos. Una de las claves para aprender a leer diagramas de circuitos, es conocer el significado de cada símbolo. También es importante aprender a conectar cada componente del Kit a la Plaqueta de Educación o a la HomeWork Board, basándose en el esquema.

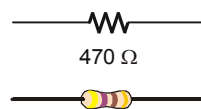


**Diagrama de circuito:** A menudo llamado simplemente esquema, es un mapa que usa símbolos para mostrar los componentes de un circuito y el modo en que se conectan. Los componentes se representan por símbolos como el que muestra la Figura 1-1 del LED.

La Figura 1-1 muestra el símbolo esquemático de un LED a la izquierda y el dibujo físico de un LED del kit de componentes, a la derecha. También muestra la correspondencia entre los pines de un LED y los terminales del símbolo esquemático.



La Figura 1-2 muestra el dibujo de un resistor debajo de su símbolo esquemático. Este símbolo normalmente tiene el valor de la resistencia escrito a su lado. Las bandas de colores mostradas en el dibujo indican su valor, que es medido en Ohms. El símbolo omega ( $\Omega$ ) se usa para indicar Ohm. Puede usar el Apéndice B para convertir el código de color de un resistor en valores de resistencia.



**Figura 1-2**  
*Símbolo esquemático de un Resistor y componente correspondiente*



**Corriente/Amperes:** La corriente se produce cuando los electrones se desplazan desde el punto A al B. La corriente continua se produce cuando se provee al excedente de electrones del terminal negativo de una batería, un camino para llegar hasta el terminal positivo. Los Amperes surgen de la medición de la cantidad de electrones por segundo que se desplazan por ese camino.



**Resistencia/Ohm:** Resistencia es una propiedad del material que se coloca en el camino de los electrones. A mayor dificultad de los electrones para atravesar el material, mayor es la resistencia. La resistencia se mide en Ohms ( $\Omega$ ).



**Ley de Ohm:** Cuando se usa un resistor para proveer un camino para la circulación de corriente entre los terminales positivo y negativo de una batería, obtenemos un circuito eléctrico con tensión, resistencia y corriente. La Ley de Ohm relaciona estas tres cantidades con la fórmula:

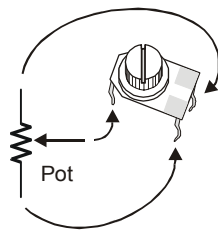
$$V = I \times R$$

V es la tensión medida en Volts o Voltios, I es la corriente medida en Amperes, y R es la resistencia medida en Ohms.

**Las otras personas:** ¿Se fijó que Volts, Amperes y Ohms comienzan con mayúsculas? Esto es debido a que recibieron el nombre de personas que realizaron descubrimientos significativos sobre la electricidad. Ya sabemos de donde salió la unidad Voltio pero, ¿qué hay sobre los otros? Los Amperes recibieron su nombre del físico del siglo XVIII André Marie Ampère. Los Ohms surgen por el físico del siglo XIX George Simon Ohm.

## El Potenciómetro – Una Fuente de Tensión Variable

El potenciómetro (pot) tiene 3 pines en su cara inferior, que se enchufan en la Plaqueta de Educación. En la cara superior, tiene una perilla mediante la cual puede ser ajustado su valor. En este experimento, usaremos la resistencia variable para obtener una salida de tensión variable. Figura 1-3 muestra la correspondencia entre los pines del potenciómetro y su símbolo esquemático.



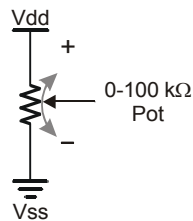
**Figura 1-3**  
Potenciómetro con su correspondiente símbolo esquemático

La Figura 1-4 muestra lo que sucede en el interior del potenciómetro, a medida que es ajustado. La línea zigzagante representa un elemento resistivo, normalmente hecho de carbón. Un extremo del elemento resistivo es conectado a Vdd en la Board of Education (Plaqueta de Educación) o HomeWork Board, y el otro a Vss. El terminal del medio está



conectado al “cursor” y es donde la tensión de salida variable es medida. El cursor permanece en contacto con el elemento de carbón, a medida que se desplaza.

A medida que el cursor se acerca a Vdd, la tensión medida en el terminal de éste se aproximará al valor de Vdd, que es 5 Volts. Igualmente cuando el cursor se acerca a Vss, la tensión del terminal se acercará a Vss, que es 0 Volts. Cuando el cursor se desplaza entre Vdd y Vss, la salida varía entre estos valores, en forma análoga a una puerta que se abre y cierra.



**Figura 1-4**  
Cursor del  
potenciómetro

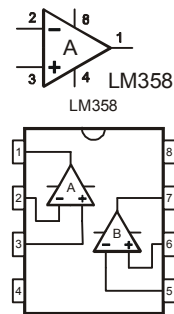
*Mostrando cómo se  
desplaza sobre la  
superficie del elemento  
resistivo a medida que  
es ajustado.*

### **El Amplificador Operacional LM358**

Un op-amp (amplificador operacional) es un bloque de construcción usado comúnmente en circuitos analógicos. La Figura 1-5 muestra el símbolo esquemático y el diagrama en bloques del amplificador operacional LM358 usado en este experimento. El circuito usado se denomina seguidor de tensión, debido a que la tensión de salida es igual a la tensión de entrada. En otras palabras, la tensión de salida "sigue" a la tensión de entrada. La razón para usar el seguidor de tensión es separar eléctricamente el circuito del potenciómetro del circuito del LED. Aprenderemos más sobre la utilidad del seguidor de tensión en el Capítulo 4.

**Figura 1-5**

Amplificador operacional LM358



*El símbolo esquemático tiene números en cada uno de sus terminales que corresponden a los números del diagrama en bloques.*

*El diagrama en bloques es una vista de arriba del dispositivo, mostrando los símbolos esquemáticos en su interior.*

*Asegúrese de identificar correctamente la ubicación del pin 1 y la marca indicadora, cuando coloque el LM358 en la protoboard. Un conexionado incorrecto podría dañar el amplificador operacional.*



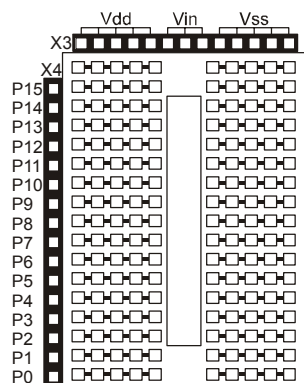
**IMPORTANTE:** Desconecte la fuente de alimentación de la Plaqueta de Educación mientras arma el circuito.

## La Plaqueta de Educación

Figura 1-6 muestra el resto de los símbolos esquemáticos empleados en el primer experimento y dónde están ubicados en la plaqueta. El símbolo Vdd es la fuente de 5 Volts del BASIC Stamp y la Plaqueta de Educación. Hay 4 conectores en la parte superior izquierda de la protoboard para Vdd.

El símbolo de masa representa a Vss. Este es el terminal de referencia para tomar mediciones y se considera que su potencial es de 0 Volts comparado con cualquiera otra tensión de la plaqueta. Los cuatro conectores para Vss están en la parte superior derecha de la protoboard.

Hay una tira de dieciséis conectores al costado izquierdo de la protoboard, para los pines de E/S del BASIC Stamp. Cada pin de E/S tiene un rótulo. El pin P0 se accede por el conector inferior. El pin P1 es el conector siguiente, hasta llegar al pin P15 que se encuentra en la parte superior.



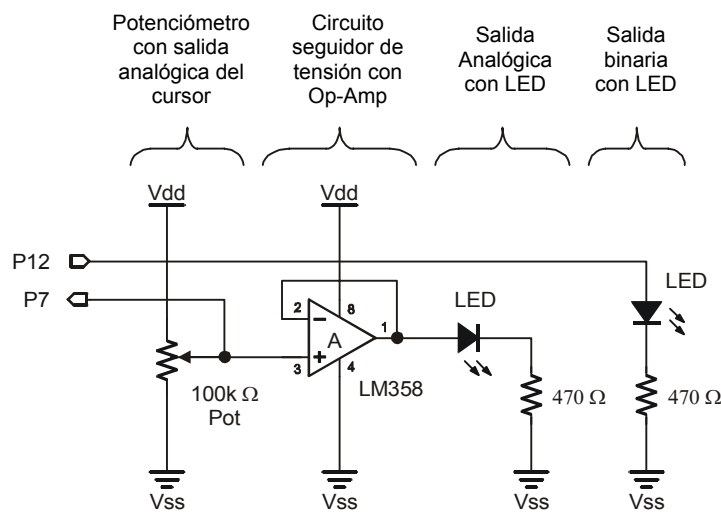
**Figura 1-6**  
Plaqueta de Educación o  
HomeWork Board

*Símbolos sobre la Plaqueta  
de Educación y su ubicación  
real.  
La HomeWork Board tiene  
la misma distribución.*

La Figura 1-6 también muestra cómo se encuentran unidos los 5 conectores de cada fila de la protoboard. Hay 34 de estos grupos de 5 conectores unidos, agrupados en dos columnas. Si quiere unir eléctricamente dos dispositivos, simplemente debe enchufarlos en la misma fila de 5 conectores. Los terminales estarán eléctricamente conectados.

### **Armando el Comparador**

- Arme el circuito de acuerdo al esquema de la Figura 1-7. Este diagrama es como una lista de conexiones entre los dispositivos. Intente usar esta lista para armar el circuito. Esta es una lista parcial de las conexiones que muestra el esquema:
- El cursor del potenciómetro de 100 k $\Omega$  se conecta al pin 3 del op-Amp LM358.
- El pin 2 del LM358 se conecta al pin 1 del LM358.
- El pin P7 del BASIC Stamp se conecta al cursor del potenciómetro.
- El pin 8 del LM358 se conecta a Vdd de la Plaqueta de Educación.
- El pin 4 del LM358 se conecta a Vss de la Plaqueta de Educación.
- Siga el diagrama como si fuera una lista hasta que termine de montar el circuito.



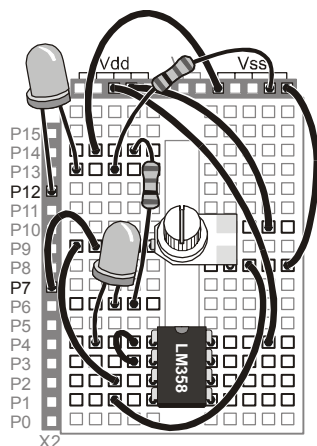
**Figura 1-7**  
Diagrama  
esquemático.

*Recuerde tratar a este diagrama como una lista de conexiones para construir su circuito.*

*Aunque este circuito tiene pocos componentes, en realidad consta de 4 sub circuitos separados, teniendo cada uno una función diferente.*

El potenciómetro es el que genera la salida analógica. El op-amp se configura como seguidor de tensión. Este alimenta a su salida el circuito analógico con LED. Hay otro circuito separado que usa un pin del BASIC Stamp para controlar un LED.

La Figura 1-8 muestra una configuración posible de protoboard para el diagrama de la Figura 1-7. Si necesita ayuda adicional para montar circuitos en la protoboard, consulte “¿Qué es un Microcontrolador?”.

**Figura 1-8**

Ejemplo de Protoboard:

*Compare esta distribución con la Figura 1-6.**¿Está conectado correctamente el LM358?**¿Vdd va conectado al pin 8 y Vss al pin 4?**Verificando todo obtendremos respuestas afirmativas. Vss está conectado por un cable al terminal izquierdo del potenciómetro. Otro cable conecta el terminal izquierdo del potenciómetro al pin 4 del LM358.**Dado que puede seguir un cable durante todo el camino entre el pin 4 del LM358 y Vss, esto significa que el pin 4 está conectado directamente a Vss.*

**IMPORTANTE:** Preste mucha atención al colocar el LM358, de forma que la marca indicadora quede hacia la derecha, como se muestra en el ejemplo de la protoboard. Si lo coloca al revés, el op-amp se arruinará luego de que le conecte la batería o fuente de alimentación a la Plaqueta de Educación.

### **Programando el Proyecto**

El Estado del Programa\_de\_P7.bs2, muestra como el PBASIC puede ser usado para que el BASIC Stamp realice varias tareas. Primero, el BASIC Stamp controla el estado del pin P7, que se configura como entrada. Recuerde, P7 está conectado al cursor del potenciómetro.

Dependiendo del nivel de tensión analógica sobre el pin P7, el BASIC Stamp ve a la entrada en estado alto o bajo (binario 0 o 1). Tan pronto como la entrada P7 recibe un estado alto, el BASIC Stamp envía un estado alto al circuito del LED por el pin P12. Cuando la entrada está en estado bajo, el pin de salida se pone en estado bajo. La Ventana DEBUG también será usada para monitorear el estado del pin P7.

Ingresa el Estado del Programa\_de\_P7.bs2 en el Stamp Editor y guárdelo con un nombre conveniente, tal como Estado\_de\_P7\_1\_R0.bs2. El nombre sale de Programa 1.1, Revisión 0. Asegúrese de que el cable de programación esté correctamente conectado a su plaqueta y al puerto serial de su computadora. Luego de asegurarse que la batería o

fuelle de alimentación esté conectada, ejecute el programa presionando Ctrl-R o haciendo clic en el botón Run.

```
' Analógico y Digital Básicos - Estado de P7.bs2
' Comprueba el estado de P7 y muestra éste en la Ventana Debug.
' {$STAMP BS2}
' {$PBASIC 2.5}

DEBUG CLS
INPUT 7
OUTPUT 12

DO
  OUT12 = IN7
  DEBUG HOME, "El estado de P7 es ", BIN IN7
LOOP
```

## Explicación del Programa

Las primeras líneas comienzan con un apóstrofe. Esto significa que son comentarios y no comandos PBASIC. La primera línea indica el libro y el archivo, para futuras referencias.

```
' Analógico y Digital Básicos - Estado_de_P7.bs2
```

El Segundo comentario es una descripción del programa. ¿Que hace el programa?

```
' Comprueba el estado de P7 y muestra éste en la Ventana Debug.
```

Las dos líneas siguientes, son comentarios especiales, no para futuras referencias. Nosotros las llamamos instrucciones compiladas y su propósito es identificar al BASIC Stamp y la versión PBASIC que nosotros estamos usando. Por ejemplo, si queremos seguir este manual con el BASIC Stamp 2 SX, podemos reemplazar las instrucciones compiladas.

“‘{\$STAMP BS2}” con “‘{\$STAMP BS2SX}”.

```
' {$STAMP BS2}
' {$PBASIC 2.5}
```

Es bueno inicializar la ventana debug y limpiarla antes de mostrar datos. De esta forma se evita mostrar datos erróneos procedentes de programas anteriores que estuvieron en la memoria del BASIC Stamp. La ventana debug se abre automáticamente la primera vez que encuentra el comando **DEBUG** en un programa en PBASIC. Este comando **DEBUG** limpia la ventana debug luego de abrirla:

```
DEBUG CLS
```

El BASIC Stamp necesita conocer cuántos pines de entrada-salida están conectados al circuito. Es decir, ellos pueden establecer si la función es de entrada o salida. Este comando del PBASIC puede establecer si el pin P7 es un pin de entrada.

```
INPUT 7
```

De la misma manera, el pin P12 de entrada-salida, funciona como de salida con este comando.

```
OUTPUT 12
```

El resto del programa debería repetirse una y otra vez, así que este es un buen lugar para introducir un bucle **DO-LOOP**. Por lo tanto, en este punto, nosotros necesitamos comenzar a repetir el código que pusimos

```
DO
```

Más adelante en el programa, aparecerá la instrucción **DO-LOOP**. Cada vez que el programa encuentra el comando **DO-LOOP**, regresa al bucle **DO-LOOP** y comienza a ejecutar nuevamente las instrucciones.

La siguiente tarea es lograr que el LED conectado al pin P12 se encienda cuando la tensión en P7 es suficientemente alta como para ser considerada como señal binaria de estado alto. En otras palabras, si el valor de entrada medido en P7 es un 1 binario, la salida en P12 deberá ser un 1 binario. Aunque hay varias formas de llevar esto a cabo, la más simple es igualar el valor de salida binario del pin P12 al valor de entrada binario del pin P7.

```
OUT12 = IN7
```

Se pueden usar comandos **DEBUG** para mostrar los niveles de señal recibidos por un pin de E/S que esté funcionando como entrada, en la ventana debug. El comando **DEBUG** de abajo imprime tres datos diferentes. Cuando se imprime más de un dato con el comando **DEBUG**, éstos deben separarse con comas.

```
DEBUG HOME, "El estado de P7 es ", BIN IN7
```

Lo primero que aparece después del comando **DEBUG** es **HOME**. Esto envía el cursor a la esquina superior izquierda (conocida como "**HOME**" o "**INICIO**") de la ventana debug. Note que **HOME** está seguida por una coma para separar el siguiente dato a imprimir. El siguiente dato está entre comillas: "El estado de pin P7 es ". Cada vez que quiera mostrar un mensaje de texto en la ventana **DEBUG**, use comillas. El tercer dato es **BIN IN7**, quien le dice a la ventana **DEBUG** que muestre el valor de entrada binario del pin P7.

Queremos que el BASIC Stamp siga controlando el valor en P7 una y otra vez. También queremos que el BASIC Stamp actualice automáticamente el LED y la ventana debug con la información obtenida de P7. Esto se lleva a cabo repitiendo indefinidamente el programa desde el comando **DO-LOOP**, la etiqueta bucle que creamos anteriormente.

Para enviar el programa hacia el comando **DO** y empezar todo el proceso nuevamente, usamos la instrucción:

LOOP



## **Solución de Problemas**

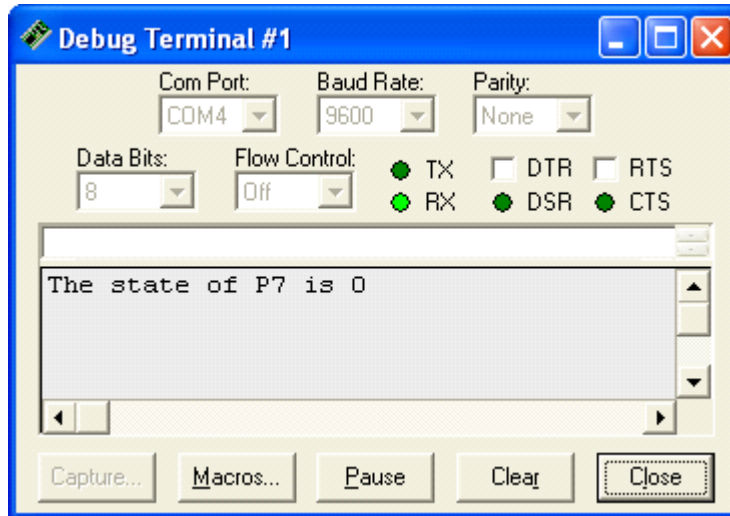
Estos son algunos temas a considerar si su programa no funciona como se esperaba.

- A menudo se necesitan varios intentos para corregir todos los errores en el cableado y el programa. Los más comunes se producen al teclear el programa. En algunos casos el Stamp Editor le avisará que hay un error. Por ejemplo, si escribe incorrectamente un comando el Stamp Editor no lo entenderá y le indicará la existencia del error resaltando la palabra y mostrando un mensaje en inglés.
- En otros casos el programa se ejecutará aunque una línea de código esté escrita incorrectamente. Por ejemplo, podría haber escrito out13 en lugar de out12. Con un error como este, cuando ejecute el programa, el LED conectado al pin P12 no se encenderá cuando se esperaba, debido a que la señal de estado alto se intenta enviar al pin P13.
- Otro error común es conectar un cable en una posición equivocada en la protoboard. Si un LED no se enciende y apaga como debiera y no hay errores de programación, controle el cableado. También controle que el cátodo y el ánodo del LED estén conectados correctamente. Cuando un LED se conecta al revés, no se enciende.
- Si la información de la ventana **DEBUG** aparece con ruido o información sin sentido, intente solucionarlo cerrando la ventana **DEBUG** y ejecutando el programa nuevamente.

## La Salida

A medida que mueve el potenciómetro, note como el LED de la salida del seguidor de tensión analógico varía su brillo. Mientras tanto el circuito del LED alimentado por P12 solamente se enciende o apaga. Esta es la diferencia característica entre tensión analógica y tensión digital (binaria).

La salida mostrada en la ventana debug será similar a la de la Figura 1-9. El estado de P7 podría ser 0 ó 1. De acuerdo con este valor, el LED alimentado por P12 estará apagado o encendido.



**Figura 1-9**  
Salida de Debug  
para el Programa  
1.1.

Ajuste el potenciómetro hasta que encuentre la tensión de umbral. Usted sabrá que encontró el umbral cuando la ventana debug indique que el estado de P7 cambia de 0 a 1 y viceversa, con un mínimo movimiento del potenciómetro. Observe la posición del potenciómetro. Cuando montemos un Voltímetro de CC en el Capítulo 3, podremos ver que la tensión está cerca de 1.4 Volts, que es el umbral de tensión para un pin de E/S del BASIC Stamp cuando funciona como entrada.

## **Sobre el Comparador**

Usando PBASIC, programamos al BASIC Stamp para que funcione como comparador. Un comparador es un circuito que compara su tensión de entrada con una tensión específica, conocida como tensión umbral. Si la tensión de entrada es mayor que la tensión de umbral, el comparador envía una señal de estado alto a su salida. Si la entrada está por debajo del umbral, envía un estado bajo.

En nuestro caso, cuando la tensión analógica en el pin P7 está por debajo de 1,4 Volts, el BASIC Stamp pone en estado bajo (0 Volts) el pin P12. Cuando la tensión analógica del pin P7 está por encima de 1,4 Volts se pone en estado alto (5 Volts) el pin P12. Como puede observar en la ventana debug, el BASIC Stamp interpreta las entradas analógicas inferiores a 1,4 Volts como estado bajo (0) y las superiores a 1,4 Volts como estado alto (1).

Hacer funcionar un comparador cerca de la tensión de umbral es interesante, debido a que con un pequeño cambio en la tensión en un pin de entrada del BASIC Stamp, por ejemplo de 1,3 a 1,5 Volts, obtenemos un cambio extremo, de 0 a 5 Volts en la salida.

### ¿Qué Aprendí?

Complete las oraciones de abajo con las palabras de la lista de la izquierda.

carbón	Un _____ muestra los símbolos esquemáticos interconectados por líneas. Cada símbolo corresponde a un componente y las líneas que los conectan nos guían en la construcción del circuito en una _____.
op-amp	Un _____ es un bloque de construcción analógico que se usó en este experimento como seguidor de tensión.
analógica	En este experimento, el potenciómetro se usó como fuente de tensión _____.
umbral	La tensión en el cursor del potenciómetro puede variarse de acuerdo a la posición de éste sobre el elemento de _____.
salida	Un comparador es un dispositivo que genera una salida binaria que varía si la tensión de entrada está por encima o debajo de un cierto _____ de tensión. Un comparador puede reaccionar a una pequeña variación en la tensión de entrada con un gran salto en la tensión de _____.
diagrama	
protoboard	

**Preguntas**

1. Marque la palabra que haga verdadera la expresión: La entrada del seguidor de tensión es por el terminal ( inversor / no inversor ) del op-amp, en este experimento.
2. ¿Cómo diferencia el cátodo del ánodo del LED?
3. Si el umbral de un comparador es de 2,5 Volts y la entrada es de 1,5 Volts. ¿Cuál sería la salida?
4. Explique lo que hace el comando **DEBUG home**. ¿Qué debe hacerse para mostrar más de un dato con una misma instrucción **DEBUG**?
5. ¿Qué instrucción usaría para configurar al pin P8 como entrada?

**Desafío**

1. Agregue un segundo LED al circuito de la Plaqueta de Educación y use el pin P11 para alimentarlo en sentido inverso. En otras palabras, cuando un LED esté encendido, el otro estará apagado. Pista: agregue un valor de 1 a la salida en P11 para invertirlo
2. Modifique el código del Programa 1.1 para que el LED titile mientras la salida del potenciómetro esté por encima de la tensión umbral del pin de entrada del BASIC Stamp. Pista: Puede usar el comando `pause 500` para lograr una pausa de medio segundo.
3. Modifique el Programa 1.1 para que un LED titile mientras la entrada de tensión al pin P7 esté por encima de la tensión umbral y el otro LED titile cuando la tensión del pin P7 esté por debajo de la tensión umbral.

### **¿Por qué aprendí esto?**

En este capítulo, comparamos una salida de LED binaria con una analógica. La única información que podemos anticipar con el cambio digital en la salida binaria es si el cursor del potenciómetro pasó por el umbral de tensión, sin poder determinar el brillo del LED. Por otro lado, cerca del umbral de tensión podríamos detectar pequeñas variaciones de la tensión analógica.

Incluso con la limitada cantidad de información analógica provista por la entrada binaria, podemos desarrollar un dispositivo llamado comparador, que tiene muchas aplicaciones en la electrónica. Como descubriremos en el último capítulo, el temporizador 555 puede hacer cosas muy interesantes. Esto se debe en parte, a dos comparadores microscópicos que se encuentran dentro del chip.

### **¿Cómo puedo aplicarlo?**

En próximos capítulos, usaremos el umbral de tensión para medir la frecuencia de sonido para aplicaciones de grabación y reproducción. También podemos construir otro tipo de conversor analógico digital usando un circuito muy simple, el BASIC Stamp y el concepto de umbral de tensión. Usaremos esta técnica para medir intensidad de luz, así como también valores de capacitores.

## Capítulo 2: Introducción al Proceso de Bits

## 2

Un paso importante en el aprendizaje de cómo lograr que el BASIC Stamp procese datos analógicos, es aprender cómo hacer para enviar y recibir números binarios. También es importante entender cómo trabajan los números binarios y cómo se realiza la conversión de un número binario al sistema decimal.

### COMUNICACIÓN BÁSICA

Este capítulo introduce algunas técnicas para transmitir y recibir números binarios con el BASIC Stamp. En este capítulo, armaremos un teclado binario para transmitir números binarios al BASIC Stamp. El BASIC Stamp también será programado para procesar y mostrar los números binarios que reciba. Los números binarios se mostrarán con LEDs así como también con la ventana **DEBUG**. La ventana **DEBUG** también será útil para mostrar los números binarios en formato decimal.

En “¿Qué es un Microcontrolador?”, aprendimos que binario es un sistema numérico usado por los microcontroladores, que emplea solamente dos dígitos, 0 y 1. El BASIC Stamp es uno de muchos dispositivos electrónicos digitales que pueden interpretar a 0 Volts como un 0 binario y a 5 Volts como un 1 binario.

El sistema binario es bueno para describir estados y números. En términos de estados, los dos dígitos del sistema binario (0 y 1) pueden ser usados para representar encendido / apagado, abierto / cerrado, no / si, alto / bajo, etc. Combinaciones de dígitos binarios pueden ser usadas para describir números. Por ejemplo, los números binarios 101, 110 y 111 representan a los números decimales 5, 6 y 7. Estos números pueden ser usados para describir información analógica, como la posición de una puerta a medida que se abre o cierra.

### Componentes Requeridos

Separe estos componentes antes de comenzar.

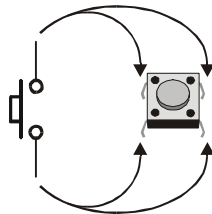
- (2) Resistores de 470 Ohm
- (2) Resistores de 220 Ohm
- (2) Resistores de 10 K Ohm
- (2) Pulsadores

(2) LEDs rojos  
(varios) Cables de interconexión

### El botón pulsador

En este capítulo introducimos el pulsador y su símbolo esquemático, que mostramos en la Figura 2-1. Note que a cada terminal del símbolo esquemático le corresponden dos pines en el componente. Si quiere realizar una conexión a uno de los terminales del símbolo esquemático, puede conectar cualquiera (o ambos) de los dos pines correspondientes.

El espacio libre en el símbolo esquemático indica que el interruptor está normalmente abierto. Cuando los dos terminales de un interruptor no están conectados, se obtiene un circuito abierto. En circunstancias normales (cuando el pulsador no está presionado) el circuito está abierto, dándole el nombre de normalmente abierto.

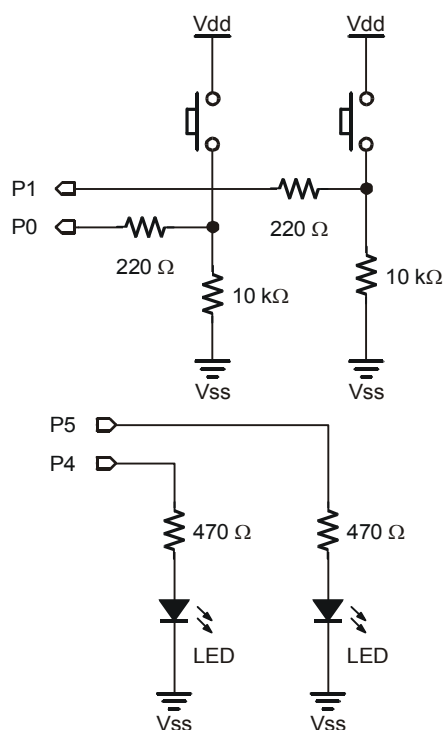


**Figura 2-1**  
Símbolo del Pulsador  
Comparado con el  
Componente

### Construyendo el circuito

La Figura 2-2 muestra el esquema para este experimento. Recuerde interpretarlo como una lista de componentes y conexiones. Por ejemplo, el ánodo del LED de la derecha está conectado al terminal P5 de la Plaqueta de Educación. El cátodo está conectado a un terminal de un resistor de 470  $\Omega$ . El otro pin del mismo resistor está conectado a Vss de la Plaqueta de Educación y continúa así hasta completar el montaje. Siga al pie de la letra los esquemas cuando esté armando los circuitos.





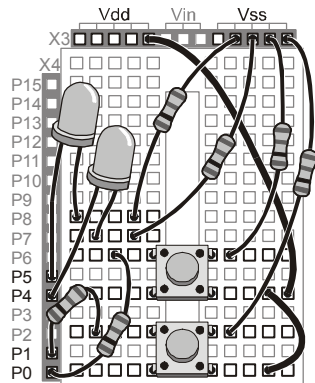
**Figura 2-2**  
Esquema que muestra  
dos circuitos de  
pulsadores y dos  
circuitos de LEDs.

Antes de hacer un programa en PBASIC que le diga al BASIC Stamp como controlar este circuito, es esencial entender cómo funciona. Los LEDs son muy simples de comprender. Ponga a P4 en estado alto y el LED se encenderá; ponga a P4 en estado bajo y el LED se apagará. El circuito del LED conectado a P5 trabaja de la misma manera.

Ahora, ¿qué sabemos sobre los pulsadores? Observemos qué es lo que ve el pin P0 cuando el pulsador está presionado o no. Cuando el pulsador está presionado, P0 queda conectado directamente a Vdd, que son 5 Volts. P0 ve un estado alto. Cuando el pulsador no está presionado, P0 está conectado a Vss (0 Volts) a través del resistor de 10 kΩ, por lo tanto P0 ve un estado bajo. Este concepto se aplica a los dos pulsadores mostrados en la Figura 2-2

La Figura 2-3 muestra una distribución posible sobre la protoboard. De los dos pines de E/S del BASIC Stamp usados para los pulsadores, el pin de la izquierda (P0) está conectado al pulsador de la derecha. El pin de la derecha (P1) está conectado al pulsador

de la izquierda. La razón por la que se cruzaron los cables de los pulsadores está relacionada a la forma en que se escriben los números binarios, que explicaremos más adelante en este mismo capítulo.



**Figura 2-3**

Ejemplo de montaje en protoboard.

*Será más fácil introducir números binarios con los pulsadores si orienta la Plaqueta de Educación como se muestra en la figura.*

*Note que el botón de la izquierda está conectado al pin P1 y el de la derecha a P0.*

El Programa 2.1 hace que el LED de la izquierda de la Figura 2-3 se encienda cuando el pulsador de la izquierda es presionado. El LED de la derecha se enciende cuando se presiona el pulsador de la derecha. El programa muestra además, la actividad de los pulsadores en la ventana **DEBUG**.

### Programando el Proyecto

Esta es una descripción más precisa de las especificaciones del programa para los pulsadores y LEDs.

- Cuando P0 recibe un estado bajo, P5 debería emitir un 0.
- Cuando P0 recibe un estado alto, P5 debería emitir un 1.
- Cuando P1 recibe un 0, P4 debería emitir un 0.
- Cuando P1 recibe un 1, P4 debería emitir un 1

La ventana **DEBUG** puede ser usada para mostrar lo que el BASIC Stamp recibe en los pines P0 y P1. Se usan comandos PBASIC para mostrar los valores que el BASIC Stamp recibe, así como también sus equivalentes decimales.

Veamos cómo se puede llevar a cabo esto con PBASIC. Ingrese el Programa 2.1 en el BASIC Stamp Editor, guárdelo con el nombre PL2\_1R0.bs2. El nombre sale de

Programa 2.1 Revisión 0. Asegúrese que la Plaqueta de Educación reciba alimentación y que los cables estén correctamente conectados. Luego ejecute el programa.

```
' Analógico y Digital Básicos - PL2 1R0.bs2
' Programa 2.1 Revisión 0.
' {$STAMP BS2}
' {$PBASIC 2.5}

a      VAR      Bit
b      VAR      Bit
d      VAR      Nib

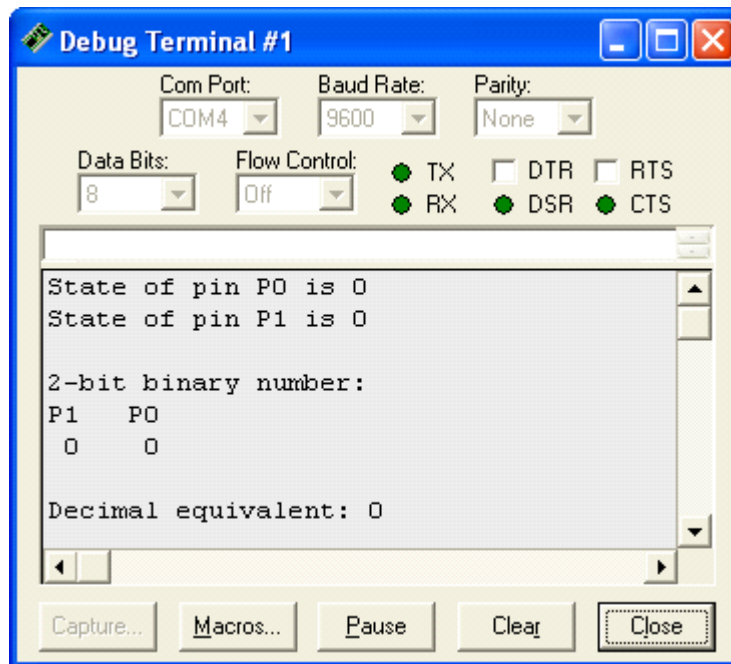
INPUT 0
INPUT 1
OUTPUT 4
OUTPUT 5

DEBUG CLS

DO
  a = IN0
  b = IN1
  OUT4 = b
  OUT5 = a
  d = (2*b) + (1*a)
  DEBUG HOME, "El estado del pin P0 es ", BIN a, CR
  DEBUG "El estado del pin P1 es ", BIN b, CR, CR
  DEBUG "número binario de 2-bit: ", CR
  DEBUG "P1  P0", CR
  DEBUG " ", BIN b, " ", BIN a, CR, CR
  DEBUG "Equivalente decimal: ", DEC1 d, CR
LOOP
```

## La Salida

Así es como debería trabajar el programa: mientras no se presionen pulsadores, la salida de **DEBUG** debería ser igual que la de la Figura 2-4 y ambos LEDs deberían estar apagados. Presione el botón de la derecha (en la Figura 2-3). ¿Se encendió el LED de la derecha? ¿Cambió a 1 el estado del pin P0 en la ventana debug? ¿El equivalente decimal es 1? Si es así, parece que su circuito y su programa están funcionando correctamente.



**Figura 2-4**  
Salida de Debug  
para el Programa  
2.1.

¿Cómo se cuenta en decimal de 0 a 3 usando los pulsadores binarios? El equivalente binario de 2 bits del decimal 0 es 00. Cuando no presiona ninguno de los pulsadores, la salida decimal es 0 en la ventana debug. Cuando presiona el botón de la derecha obtiene 01, que tiene un equivalente decimal de 1. Cuando presiona el pulsador de la izquierda obtiene 10, que tiene un equivalente decimal de 2. Cuando presiona ambos pulsadores obtiene 11, que tiene un equivalente decimal de 3.


### Explicación del Programa

Como en el capítulo previo, las primeras líneas comienzan con un apóstrofe, así que se trata de comentarios e instrucciones compiladas que el BASIC Stamp ignorará.

```
' Analógico y Digital Básicos - PL2_1R0.bs2
' Programa 2.1 Revision 0.
' {$STAMP BS2}
' {$PBASIC 2.5}
```

Luego se definen tres variables. Las variables pueden ser usadas para almacenar valores mientras que se ejecuta el programa. Las letras **a** y **b** se definen como variables que pueden almacenar 1 bit cada una. Así que la variable **a** puede almacenar un único dígito binario, al igual que la variable **b**. La letra **d** es definida como una variable que almacena un "nibble" (4 bits) de información binaria.

a	VAR	Bit
b	VAR	Bit
d	VAR	Nib



**Tamaño de memoria**

- Un bit de memoria puede almacenar un dígito binario, o sea 0 ó 1.
- Un nibble de memoria almacena 4 bits.
- Un byte almacena 8 bits.
- Una word almacena 16 bits.

Este segmento del código usa comandos presentados en el capítulo anterior. Primero, dos pines de entrada-salida son declarados entradas y otros dos pines son declarados salidas. Entonces la ventana **DEBUG** es cerrada y borrada.

```
INPUT 0
INPUT 1
OUTPUT 4
OUTPUT 5
DEBUG CLS
```

Nosotros queremos que el BASIC Stamp revise constantemente las entradas. También queremos que el BASIC Stamp actualice automáticamente los LEDs y la ventana debug con la última información de los pulsadores. De esta manera, podremos hacer que el programa se repita constantemente al llegar al comando **DO-LOOP**. Para definir el punto de inicio del comando usamos el comando **DO**, y para hacer que el programa vuelva atrás, usamos el comando **LOOP**.

**DO**

Seguidamente, necesitamos revisar el estado de los pulsadores para comprobar el estado de los pines P0 y P1. El primero de esos dos comandos hace que la variable **a** sea igual al estado medido en el pin P0. El segundo comando hace que la variable **b** sea igual al estado medido en el pin P1.

```
a = IN0
b = IN1
```

A continuación necesitamos hacer el estado de salida del pin P4 igual a la entrada tomada del pin P1. El LED de la izquierda que está conectado a P4 se encenderá cuando se presione el botón de la izquierda que está conectado a P1. Del mismo modo, la salida en el pin P5 se igualará al valor medido en el pin P0.

Dado que los valores de entrada fueron asignados a las variables **a** y **b**, podemos usarlas para fijar los valores de salida en los pines P4 y P5.

```
OUT4 = b
OUT5 = a
```

Podríamos haber usado simplemente comandos como OUT4=IN1 y OUT5=IN0; sin embargo, usar variables para almacenar valores en memoria tiene ventajas cuando los programas se vuelven más complicados. En el siguiente experimento, será necesario usar variables para almacenar valores.

La razón por la que las usamos en este programa es porque pueden ser manipuladas aritméticamente, lo que haremos en la siguiente línea para convertir de binario a decimal. Para lograr esto, multiplicamos la variable **b** por 2 y la variable **a** por 1 y las sumamos. La variable **d** es usada para almacenar el resultado. Éste es el método para convertir un número binario de 2 bits en un número decimal. La próxima sección muestra cómo hacer esto para números binarios de cualquier tamaño.

```
d = (2*b) + (1*a)
```



#### **BASIC Memoria del BASIC Stamp:**

**RAM:** El BASIC Stamp tiene 26 bytes de RAM (random access memory = memoria de acceso aleatorio) que pueden ser usados para almacenar valores variables. Otros 6 bytes de RAM se usan para administrar los pines de E/S del BASIC Stamp.

**EEPROM:** Es la sigla de electrically erasable programmable read only memory, que en castellano es memoria solo de lectura borrrable eléctricamente. Es usada principalmente para almacenar programas en PBASIC. La EEPROM también puede usarse para almacenar datos que no varían frecuentemente.

Como el cálculo se realiza en PBASIC, los paréntesis son necesarios para mantener las reglas algebraicas de las operaciones. Esto se debe a que el BASIC Stamp realiza los

cálculos comenzando desde la izquierda. Luego, realiza cada operación dentro de los paréntesis que encuentra, mientras avanza de izquierda a derecha.

Sin los paréntesis,  $d$  podría ser igual al valor  $((2 \times b + 1) \times a)$  debido a que este es el orden en el que se encuentran los operadores (+, -, \*, /, etc.).

Seis comandos **DEBUG** son usados para mostrar todos los estados medidos y los valores binarios calculados en la ventana debug. El primer comando **DEBUG** realiza cuatro acciones diferentes. Recuerde que cada parámetro independiente de un comando **DEBUG** debe estar separado por comas.

La instrucción **DEBUG HOME** envía el cursor a la posición superior izquierda (conocida como "**HOME**") de la ventana debug. Note que luego se coloca una coma para separar el siguiente dato. Luego aparece un mensaje entre comillas: "El estado del pin P0 es ". Cada vez que quiera imprimir un mensaje de texto en la ventana debug, use comillas. El tercer dato que aparece es **BIN a**, que le dice a la ventana debug que imprima en formato binario el valor de la variable **a**. El cuarto parámetro es **CR**, que hace que en la ventana debug se produzca un salto de línea.

```
DEBUG HOME, "El estado del pin P0 es ", BIN a, CR
```

Se imprime un mensaje similar para la variable **b**, sin el comando **HOME**. El comando **HOME** funciona bien cuando se utiliza una sola vez en un bucle. Recuerde que **DEBUG HOME** envía el cursor a la esquina superior izquierda de la ventana debug. Si usáramos **HOME** más de una vez en el mismo bucle, la información mostrada luego del primer comando **HOME** sería tapada o sobrescrita por el segundo comando **HOME**.

```
DEBUG "El estado del pin P1 es ", BIN B, CR, CR
```

Luego, dos comandos **DEBUG** son usados. Cada uno muestra un mensaje entre comillas

```
DEBUG "número binario de 2-bit: ", CR
DEBUG "P1    P0", CR
```

En el siguiente comando, el mensaje entre comillas consta de espacios en blanco. El primero contiene un espacio (la barra espaciadora se presionó una sola vez). Luego se imprime el valor binario de **b**, seguido por dos espacios entre comillas. Luego se imprime el valor binario de **a** y dos saltos de línea.

```
DEBUG " ", BIN b, "  ", BIN a, CR, CR
```

Aquí hay algo nuevo. El modificador **DEC** se usa para imprimir el valor decimal de la variable **d**. Queremos que el BASIC Stamp controle continuamente el estado de las entradas. También queremos que el BASIC Stamp actualice el estado de los LEDs y la ventana **DEBUG** con la última información proveniente de los pulsadores. Para lograr esto solamente debemos repetir indefinidamente el programa a partir de la etiqueta bucle: que creamos con anterioridad. Para enviar el programa hacia bucle:, se usa el comando **LOOP**

```
DEBUG "Equivalente decimal: ", DEC1 d, CR
LOOP
```

## Contando en Binario

La Tabla 2.1 muestra como contar de 0 a 3 usando números binarios de 2 bits y como contar de 0 a 7 usando números binarios de 3 bits.

Observe que se pueden representar cuatro números (decimales 0 al 3) con un número binario de 2 bits. Ocho números (0 al 7) con un número binario de 3 bits. Con 4 bits se pueden representar 16 números diferentes, con 5 bits se pueden representar 32 números diferentes y así sucesivamente.

Table 2-1: Contando con Números Binarios		
Número Decimal	Representación binaria de 2 bits	Representación binaria de 3 bits
0	00	000
1	01	001
2	10	010
3	11	011
4		100
5		101
6		110
7		111

Se puede determinar fácilmente la cantidad máxima de números que se pueden obtener (combinaciones de 0 y 1) de una cantidad específica de bits mediante la siguiente fórmula:

$$\text{combinaciones} = 2^{\text{bits}}$$



Esto significa que la cantidad de combinaciones posibles es igual a dos elevado al número de bits. Para 2 bits, se pueden obtener  $2^2 = 4$  números. Para 3-bits, el número de combinaciones es  $2^3 = 8$ , y así sucesivamente.

Para convertir un número de binario a decimal se necesitan dos pasos. El primero es multiplicar cada bit por su potencia de dos. La Tabla 2.2 muestra las potencias de dos para hasta 8 bits. Cuando multiplica cada bit por su valor de la Tabla 2.2, obtiene una serie de valores decimales. El segundo paso es realizar la suma de todos estos valores decimales obtenidos.

Table 2-2: Factores para Números Binarios de 8 bit s								
Bit	7	6	5	4	3	2	1	0
Multiplicador	128	64	32	16	8	4	2	1



**Factor de cada Bit y Potencias de dos** El bit-0 es el bit menos significativo o de menor peso (least significant bit = LSB, en inglés) y el bit-7 es el de mayor peso o más significativo (most significant bit = MSB, en inglés). Esto se debe a que el bit-0 hace la contribución más pequeña al valor del número y el bit-7 hace la contribución más grande. Los números binarios se arman comenzando por el bit-7 a la izquierda y finalizando con el bit-0 a la derecha, lo que permite convertirlos utilizando potencias de dos.

Ejemplos:

El multiplicador para el bit-0 es 1, que es igual a 2<sup>0</sup>.

El multiplicador para el bit-1 es 2, que es igual a 2<sup>1</sup>.

El multiplicado para el bit-7 es 128, que es igual a 2<sup>7</sup>.

---Puede usar potencias de dos para extender la Tabla 2.2 para cualquier cantidad de bits.--

A modo de ejemplo, convirtamos el número binario 1011 a decimal. Primero, multipliquemos cada bit por su potencia de dos de la Tabla 2.2.

$$\begin{aligned} 8 \times 1 &= 8 \\ 4 \times 0 &= 0 \\ 2 \times 1 &= 2 \\ 1 \times 1 &= 1 \end{aligned}$$

Segundo, sumemos los 4 valores decimales:

$$8+0+2+1 = 11$$

Ahora sabemos que el número binario 1011 es igual al número decimal 11.

### **Transmisión Serie y Paralelo**

El Programa 2.1 repite la rutina que verifica e informa el estado de los pulsadores una y otra vez. Dado que el BASIC Stamp controla el estado de las entradas sin esperar por alguna señal que avise que se están enviando números binarios, este tipo de comunicación se denomina **asincrónica**.



**Asincrónico:** Asincrónico significa no sincronizado. En el caso de nuestro teclado binario, significa que podemos cambiar los valores binarios sin tener que esperar una habilitación del BASIC Stamp. El BASIC Stamp controla las señales en P0 y P1 tan rápido como puede, sin esperar una señal de que el dato está disponible para ser leído.

Estamos enviando los bits binarios a través de dos líneas de datos separadas a la vez. Esto significa que estamos enviando los bits de datos al BASIC Stamp en **paralelo**.

El BASIC Stamp tiene 16 pines de E/S. Podríamos enviar un número binario tamaño word al BASIC Stamp en paralelo. El problema sería que no nos quedarían más pines para realizar otras tareas. Cuando se trabaja con números binarios grandes, puede ser recomendable enviar los datos en **serie** en lugar de **paralelo** debido a que reduce la cantidad de pines del BASIC Stamp empleados.

Cuando envía datos en serie, debe haber alguna forma de que el BASIC Stamp sepa cuando leer cada nuevo bit. El BASIC Stamp tiene funciones internas para enviar datos en serie en forma asincrónica o **sincrónica**.

En el próximo ejemplo, los mismos pulsadores se emplean para enviar un dato en forma sincrónica al BASIC Stamp tamaño nibble (4-bits). El resultado se muestra en la ventana **DEBUG**.



**Paralelo:** Paralelo significa que los bits de datos se envían por más de una línea al mismo tiempo. Acabamos de usar los pulsadores para enviar dos bits en paralelo.

**Serie:** En lugar de enviar los datos en paralelo por varias líneas de datos, puede usarse una sola línea de datos por la que se envían los bits uno después de otro.

**Sincrónica:** Enviar los datos sincrónicamente significa que estamos enviando los datos con cierta coordinación temporal (sincronía). Técnicamente, significa que el emisor y el receptor de los bits de datos trabajarán con la misma señal de reloj.

### Programación para Enviar Datos en Serie

Ingrese el Programa 2.2 en el Editor del BASIC Stamp y guárdelo con el nombre PL2\_2R0.bs2.

```
' Analógico y Digital Básicos - PL2 2R0.bs2
' Programa 2.2 Revision 0.
' {$STAMP BS2}
' {$PBASIC 2.5}

n      VAR      Nib
d      VAR      Nib

INPUT 0
INPUT 1

FOR n = 1 TO 4

  DO
    'Espera el estado alto
    LOOP UNTIL IN1=1

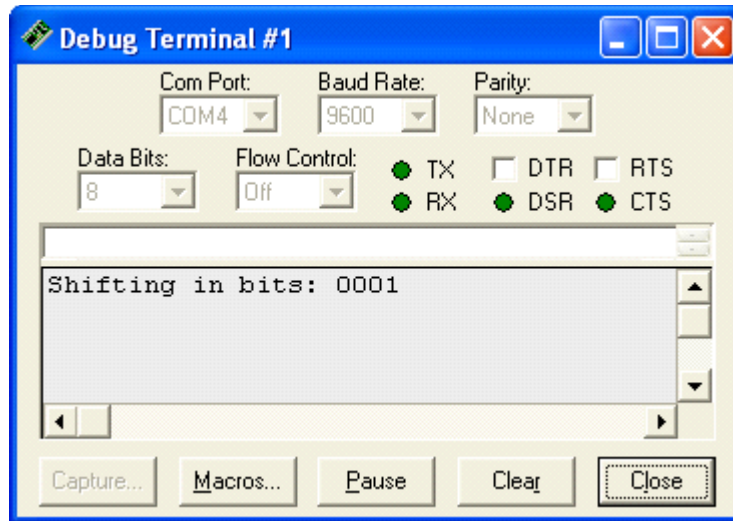
  DO
    'Espera el estado bajo
    LOOP UNTIL IN1=0

  d = d << 1
  d = d + IN0
  DEBUG HOME, "Ingresando bit: ", BIN4 d
NEXT

DEBUG CR, CR, "Listo.", CR, CR
DEBUG "Valor decimal: ", DEC2 d, CR, CR
```

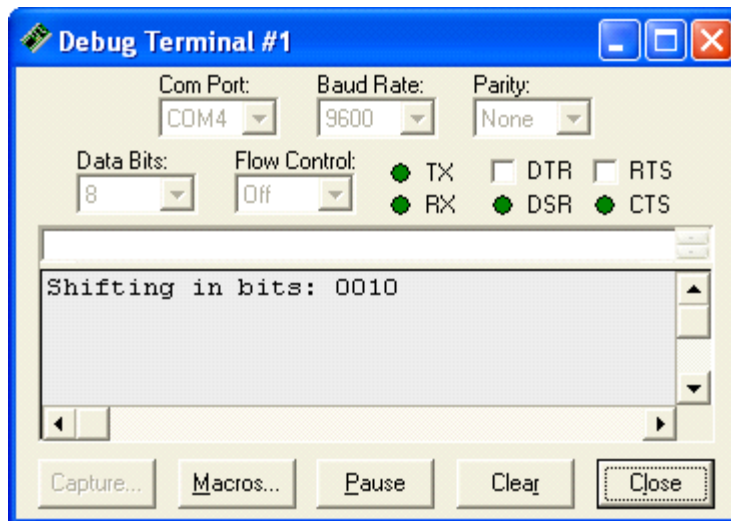
### La Salida

La ventana **DEBUG** se inicializará mostrando la pantalla vacía cuando usted arranque el programa. Siga cuidadosamente estas instrucciones para enviar datos en forma serial sincrónica. Primero, presione y retenga el botón derecho. Luego presione y suelte el botón izquierdo. La salida debería verse como la Figura 2-5 de abajo.



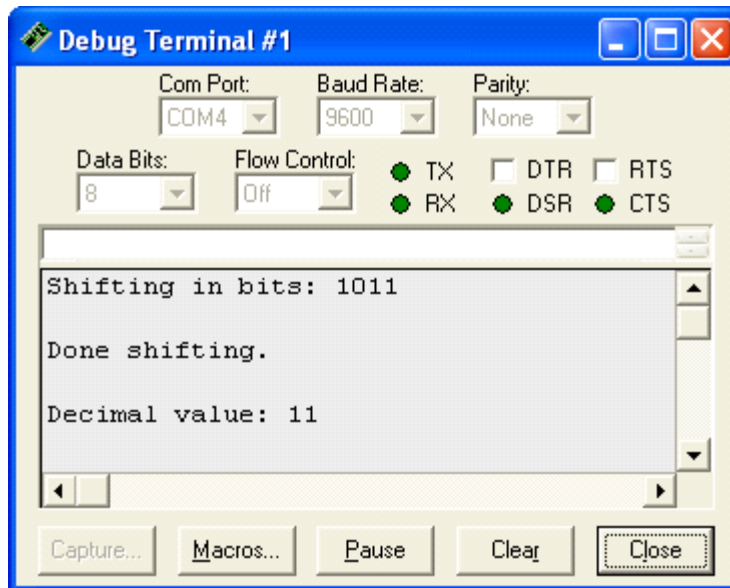
**Figura 2-5**  
*Salida de Debug para el Programa 2.2.*

Luego suelte el botón derecho y presione y libere el botón izquierdo nuevamente. La salida debería cambiar y verse como en la Figura 2-6.



**Figura 2-6**  
*Salida de Debug para el Programa 2.2.*

Presione y retenga el botón derecho, luego presione y suelte el botón izquierdo dos veces. La salida debería verse como en la Figura 2-7

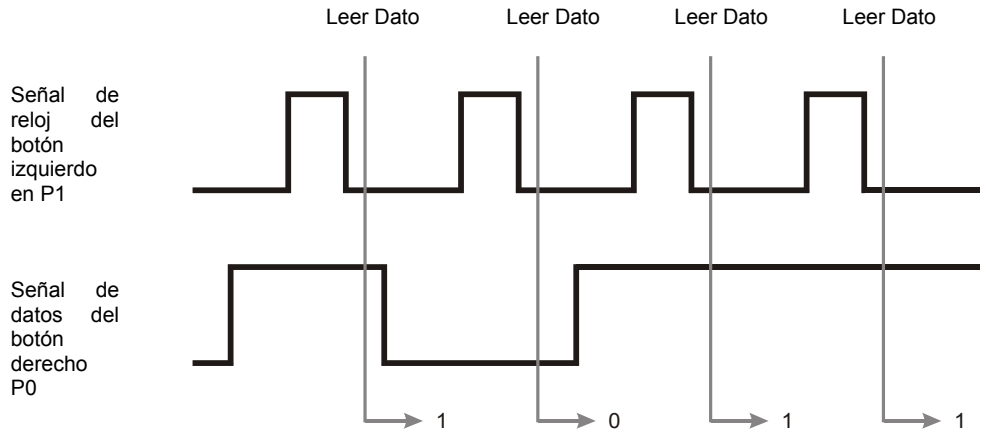


**Figura 2-7**  
Salida de Debug  
para el Programa  
2.2.

Si su programa funcionó correctamente, usted desplazó 4 bits en forma serial sincrónica en la RAM del BASIC Stamp. Además, se verificó que el valor decimal de 1011 es realmente 11 en el sistema numérico decimal.

La Figura 2-8 muestra como ocurren estos eventos (de izquierda a derecha) en un diagrama de tiempo. El botón de la izquierda se comporta como señal de reloj. Ésta consiste de una serie de pulsos de reloj. Cada pulso se genera al presionar y soltar el botón izquierdo. Esto genera una señal: bajo-alto-bajo en P1. Cada vez que se libera el pulsador izquierdo, el BASIC Stamp controla el estado de la línea de datos, que depende del estado del pulsador derecho (P0).


El BASIC Stamp lee los datos de entrada cada vez que la señal de reloj pasa de estado alto a bajo. Esta transición recibe el nombre de flanco descendente del pulso de reloj.



**Figura 2-8** Diagrama de Tiempo..

**Explicación del Programa**

Al igual que en el Programa 2.1, usamos un comentario para incluir información sobre el programa al comienzo. Recuerde, tan pronto como aparece un apóstrofe en una línea de código en PBASIC, todo lo que se encuentra a su derecha es ignorado por el BASIC Stamp. Luego se definen dos variables tipo nibble y los pines P0 y P1 se configuran como entradas.

	<b>Bucle FOR-NEXT</b>	
	FOR n = 1 TO 7	En el ejemplo que se muestra a la izquierda, el bucle <b>FOR-NEXT</b> ejecuta los comandos PBASIC entre el comando <b>FOR</b> y el comando <b>NEXT</b> 7 veces. En la primera pasada por el bucle el valor de n es 1, en la segunda n es 2 y así hasta que n llega a 7.
	PBASIC commands	
	.	
	NEXT	Después de la 7ma pasada por el bucle <b>FOR-NEXT</b> , el programa sale del bucle y comienza a ejecutar los comandos PBASIC que se encuentran a continuación del comando <b>NEXT</b> .
	PBASIC Commands	

El código que estaba dentro del bucle **FOR-NEXT** es diferente de lo que habíamos visto antes. Comienza con un bucle condicional **DO..LOOP**. Este bucle es usado para controlar el valor en el pin P0. Si el valor de P0 es bajo, el **LOOP UNTIL IN1=1** envía el programa hacia atrás a la etiqueta **DO**. Si el valor en el pin P1 es alto, el programa ejecuta la línea siguiente, después de la instrucción **LOOP**.

```
DO
    'Espera el estado alto
    LOOP UNTIL IN1=1
```

La misma técnica se aplica en las dos siguientes líneas de código (y un comentario), que se repiten hasta que se reciba una señal de estado bajo.

```
DO
    'Espera el estado bajo
    LOOP UNTIL IN1=0
```

Como se muestra abajo,  $d = d \ll 1$  desplaza los bits en  $d$  un lugar a la izquierda. Cuando se desplazan los valores, el bit de la derecha (el LSB) se llena automáticamente con un cero. Luego se carga el valor del bit del pin P0 en la posición de menor peso (LSB) del nibble. Esto se lleva a cabo con el comando  $d = d + IN0$ , que suma el bit medido en P0 al valor de la variable tipo nibble  $d$ . La segunda vez que se repite el bucle, el valor que había sido colocado en el LSB se desplaza un lugar a la izquierda, a la posición de bit-1. Mientras tanto el valor obtenido en P0 es colocado en el LSB. El proceso de desplazamiento y suma se repite cuatro veces a medida que los cuatro bits se desplazan dentro de la variable.

```
d = d << 1
d = d + IN0
```

Cada vez que se ingresa un bit, usamos el comando **DEBUG** para mostrar el nuevo valor. Cada vez que el programa llega a la línea **NEXT** salta al comando **FOR n = 1 TO 4** e incrementa el valor de  $n$ , hasta que llega a 4. Finalmente el programa sale del bucle.

```
    DEBUG HOME, "Ingresando bit: ", BIN4 d
NEXT
```



Una vez que el bucle **FOR-NEXT** finaliza y todos los bits se desplazaron a la variable **d**, se muestran dos mensajes. El segundo de los dos comandos **DEBUG** tiene un modificador nuevo: **DEC2**. Este modificador se usa para lograr que la ventana debug muestre el valor de **d** como un decimal de dos dígitos.

```
DEBUG CR, CR, "Listo.", CR, CR
DEBUG "Valor decimal: ", DEC2 d, CR, CR
```

El Capítulo 1 introdujo la tensión analógica y éste introdujo las bases que permiten enviar y recibir números binarios. En el próximo capítulo, combinaremos estos temas construyendo un voltímetro de CC digital, que es un dispositivo que mide tensión analógica y la muestra como valor digital.

### ¿Qué aprendí?

En las oraciones de abajo, inserte las palabras apropiadas de la lista de la izquierda.

más grande	El _____ puede ser usado para procesar información de mediciones analógicas usando _____. Antes de trabajar con conversiones de analógico a digital, es importante comprender cómo el BASIC Stamp envía, recibe y almacena datos binarios. También es importante poder usar el _____ para programar el BASIC Stamp para realizar estas tareas.
word	
decimal	
BASIC Stamp	Cuando se convierte un número binario en su equivalente _____, cada bit debería ser multiplicado por su respectiva potencia de dos. El _____ es el bit del extremo derecho de un número binario y aporta la mínima cantidad al valor del número. También se conoce como bit-0. El MSB es el bit del extremo izquierdo. Hace la contribución _____ al valor de un número binario. El bit que está justo a la izquierda del LSB es el bit-1, el bit que está dos bits a la izquierda del LSB es el bit-2 y así sucesivamente.
sincrónica	
LSB	
PBASIC	
byte	El BASIC Stamp puede almacenar un bit, un nibble, que consta de 4-bits, un _____, que son 8-bits, o una _____, de 16-bits. Para ahorrar pines de E/S, el BASIC Stamp puede enviar y recibir datos en forma serial en lugar de recibirlos en paralelo que necesita muchas líneas de datos. Los datos binarios se pueden enviar _____ o asincrónicamente.
números binarios	

**Preguntas**

1. Determine el equivalente decimal de estos números binarios: 1010, 1111, 0010 y 0100.
2. El comando  $d = d \ll 1$  se usó para desplazar los bits en la variable  $d$ , un lugar a la izquierda. ¿Qué comando cree que se usará para desplazar los bits a la derecha? ¿Qué comando usaría para desplazar los bits 3 lugares a la izquierda?
3. Explique la diferencia entre datos en serie y datos en paralelo.
4. Explique la diferencia entre transmisión de datos sincrónica y asincrónica.

**Desafío**

1. El Programa 2.1 es usado para contar hasta tres. Escriba un programa que use tres bits en paralelo (use el pin P2 para el tercer bit) y cuente hasta 7.
2. Modifique el Programa 2.2 para que muestre los pulsos de reloj que se aplican en el botón de la izquierda, con el LED izquierdo.
3. Modifique el Programa 2.2 para que desplace los bits hacia afuera, mostrándolos con el LED derecho.

### **¿Por qué aprendí esto?**

La meta de este capítulo es demostrar cómo un dispositivo como el BASIC Stamp, que procesa datos binarios, puede conectarse con el mundo analógico. Los estados lógicos y los números binarios son las bases para el proceso de datos de los microcontroladores, microprocesadores y cualquier circuito binario.

Los datos analógicos pueden ser efectivamente procesados mediante números binarios usando las técnicas de este experimento. Entender las bases del procesamiento de datos binarios hará más fácil de entender el funcionamiento de miles de aplicaciones electrónicas. Estas también son las bases del funcionamiento de las computadoras personales. Comprender los datos en el nivel binario hace más fácil de comprender varios lenguajes de programación

### **¿Cómo puedo aplicarlo?**

En algunos de los próximos capítulos, procesaremos datos analógicos usando comunicación serie y paralelo, en forma sincrónica o asincrónica. El BASIC Stamp puede ser conectado a otros circuitos integrados para intercambiar datos binarios. El BASIC Stamp también puede ser programado para convertir los datos binarios a un formato decimal más significativo.

Usaremos estas técnicas para medir tensión, sonido, luz, etc.

El BASIC Stamp tiene comandos que automatizan los procesos de transmisión y recepción serial de datos, tanto en modo sincrónico como asincrónico. Encontraremos el método serial sincrónico en el Capítulo 3. El BASIC Stamp también tiene características que simplifican la emisión y recepción de datos en paralelo como descubriremos en el Capítulo 4 donde transmitiremos datos en paralelo asincrónicamente.

## Capítulo 3: Conversión Analógica a Digital Básica

---

**3**

### CONSTRUYA SU PROPIO VOLTÍMETRO DIGITAL DE CC

Un voltímetro digital de CC es una herramienta útil para medir tensión entre dos puntos. En este Capítulo, construiremos un voltímetro para mediciones de CC (corriente continua) en el rango de 0 a 5 Volts. Un uso común de un voltímetro es controlar la tensión entre los dos terminales de una batería o una pila.

El voltímetro digital se llama así debido a que muestra las mediciones con dígitos. Se usan los dígitos de 0 a 9 y un punto decimal para mostrar las mediciones de tensión como valores decimales. Se podrían usar solamente los dígitos 0 y 1 y se seguiría llamando voltímetro digital, pero nos mostraría el valor en el sistema binario en lugar de decimal, lo que nos haría perder mucho tiempo interpretando los datos. Dado que nuestro voltímetro procesa las mediciones en binario, comenzaremos mostrando los valores binarios y luego los modificaremos a una forma más convencional y fácil de leer en decimal.

En el Capítulo 1, usamos un circuito con LED para mostrar los cambios en el nivel de tensión analógica aplicada a un circuito. Como “valor variable continuo”, la tensión analógica varía dentro de un rango continuo. Usaremos el potenciómetro como en el Capítulo 1 para generar un rango de tensiones continuo entre 0 y 5 Volts en la Plaqueta de Educación.

Si bien la información sobre la tensión analógica puede ser eficientemente procesada por dispositivos binarios, la tensión primero debe ser muestreada para poder describirla usando números binarios. El ADC0831 es un circuito integrado común que realiza esta tarea. Describe la información analógica con números binarios para dispositivos que procesan información binaria, como el BASIC Stamp.

En este capítulo, construiremos un voltímetro usando un BASIC Stamp junto con el circuito integrado ADC0831. Se conectará un potenciómetro a la Plaqueta de Educación y se ajustará para producir la tensión de salida analógica. El voltímetro luego será usado para medir muestras del rango continuo de salida de la tensión.



**Rango continuo:** Conjunto de todos los valores posibles entre un valor máximo y uno mínimo. Cuando una fuente de tensión varía sobre un rango continuo, se considera tensión analógica.

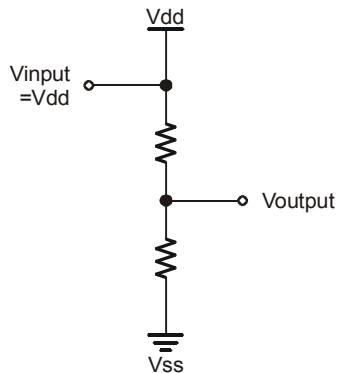
Usaremos nuestro voltímetro para hacer un muestreo de la tensión sobre un rango continuo, de 0 a 5 Volts. La tensión que medimos podría ser 1,234 Volts, 3,857564 Volts, 4,9999 Volts, etc.

### Componentes Requeridos

- (1) ADC0831
- (1) Potenciómetro de 100K $\Omega$
- (10) Cables de interconexión

### El Potenciómetro, una Fuente de Tensión Variable

Hay una razón para que la tensión en el terminal del cursor del potenciómetro cambie cuando gira la perilla. El cursor hace que el elemento resistivo del potenciómetro se comporte como dos **resistores en serie**. La Figura 3-1 muestra dos resistores en serie. Cuando se aplica una tensión de entrada y se mide la tensión de salida como se muestra en la Figura 3-1, el circuito se conoce como divisor de tensión. R1 y R2 son las resistencias entre el cursor y los otros dos terminales del potenciómetro y su valor cambia cuando se desplaza el cursor. Dado que el potenciómetro hace que R1 y R2 varíen, podemos llamar al terminal del cursor salida del divisor de tensión variable.



**Figura 3-1**

El Circuito Divisor de Tensión

*Muestra cómo el cursor de un potenciómetro hace que el elemento resistivo se vea como dos resistores en serie.*

*Vinput es la tensión medida en el terminal del cursor.*



**Resistores en serie:** En la figura de abajo se muestran tres resistores en serie. Los resistores en serie conforman una resistencia equivalente igual a la suma de todas las resistencias:

**3**

**Valores de Resistencia** Cuando sabe el valor de las dos resistencias de la Figura 3.1, puede predecir la tensión de salida usando esta ecuación:

$$V_{\text{output}} = V_{\text{input}} \times \frac{R_2}{R_1 + R_2}$$

Como era de esperarse, se llama ecuación de divisor de tensión. Este método para modificar el valor de una tensión de entrada por una fracción conocida, se conoce como divisor de tensión.

### **El Circuito Integrado ADC0831. Un Conversor Analógico Digital de 8-bits**

El ADC0831 es un circuito integrado conocido como conversor analógico a digital de 8-bits (conversor A/D) con salida serial sincrónica. Veamos que es lo que significa cada uno de estos términos:

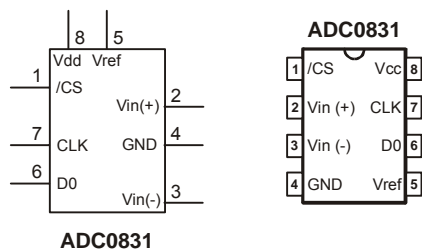
- **Circuito Integrado (IC)** es un circuito con componentes microscópicos implantados sobre la superficie de una pastilla de silicio. Usaremos tres chips en este libro. Cada chip está encapsulado en plástico negro y tiene 8 pines. El encapsulado es para proteger el circuito integrado.
- **Conversor A/D** mide una muestra de tensión analógica y entrega un número binario que la describe.
- **8-bits** es la cantidad de dígitos binarios que el ADC0831 usa para describir la tensión analógica que muestrea. 8-bits también es la resolución del conversor A/D. Puede contar de 0 a 255 (decimal) usando un número binario de 8-bits. Esto significa que el ADC0831 puede aproximar la tensión que mide a uno de 256 niveles. Un conversor con una resolución mayor, como 12-bits, puede descomponer el mismo rango de tensión en 4096 niveles, debido a que puede contar de 0 a 4095 con 12 bits binarios.

- **Serial y sincrónico** son términos que se aprendieron en el Capítulo 2. Enviamos dígitos binarios seriales (bits) al BASIC Stamp usando un botón y los bits fueron sincronizados con un segundo botón que se usó para enviar la señal de reloj. El ADC0831 trabaja en forma similar. La diferencia está en que el ADC0831 depende de una señal de reloj emitida por el BASIC Stamp para coordinar el envío de cada bit de salida serial.

El BASIC Stamp será programado para leer y almacenar los 8-bits transmitidos serialmente por el ADC0831. También programaremos al BASIC Stamp para que muestre el equivalente decimal de la salida binaria. Luego este equivalente decimal se usa para calcular y mostrar la tensión medida en formato digital. El BASIC Stamp también debe ser programado para enviar las señales de control binarias que hacen funcionar al ADC0831.

La Figura 3-2 muestra la distribución de pines del ADC0831. Cada pin tiene un número y un rótulo. El número es importante para realizar el conexionado correctamente cuando se construye el circuito. Los rótulos indican la función de cada pin.

**Señales de control binarias:** Son señales de tensión con dos estados posibles, bajo o alto, que se envían a un dispositivo para decirle cómo o cuándo hacer algo. El ADC0831 requiere señales de control para activarse y una señal de reloj para enviar cada uno de los bits de salida.



**Figura 3-2**

Símbolo del ADC0831 y Distribución de Pines.

*La distribución de pines de la derecha muestra los pines con sus respectivos rótulos, de acuerdo a como se ubican en el chip. El símbolo de la izquierda muestra la misma información pero con el típico formato para ser usado en diagramas.*

La notación de las entradas y salidas del ADC0831 funciona así:  $V_{in}(+)$  es la entrada analógica y D0 es la salida serial. VREF y  $V_{in}(-)$  se usan para polarizar el chip. Vcc y GND se usan para alimentar el IC. Vcc es esencialmente lo mismo que Vdd en la Plaqueta de Educación y GND es lo mismo que Vss. /CS indica selección de chip activo bajo y CLK representa el reloj (clock). Ambas son entradas de señales de control binarias.





**Polarizar:** Método por el cual se aplican niveles de tensión específicos en ciertos puntos de un circuito para calibrarlo o ajustarlo.

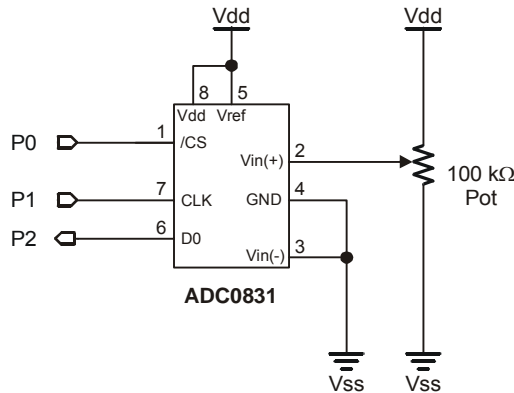
**3**

Para activar el ADC0831 para tomar una medición, el pin /CS debe recibir una señal del BASIC Stamp que comience en estado alto y luego pase a bajo. Esta señal debe permanecer en estado bajo durante la conversión. Luego la entrada CLK debe recibir un pulso de reloj (término introducido en el Experimento 2, Figura 2-8) para indicar que la conversión debería comenzar en el siguiente pulso de reloj. Para este IC, un pulso de reloj comienza en bajo, pasa a alto y luego baja nuevamente. Se necesitan 8 pulsos más de reloj para completar la conversión. Cada vez que se recibe un pulso de reloj en la entrada CLK, se envía otro de los bits seriales por la salida D0.

Los diseñadores en electrónica usan hojas de datos para encontrar la información que acabamos de mencionar. Cada fabricante de chips publica hojas de datos para cada modelo de circuito integrado que produce. La información que mostramos en la Figura 3-2 fue extraída de una hoja de datos publicada por National Semiconductor, fabricante del ADC0831. Por supuesto que todas las hojas de datos están disponibles en los sitios web de los fabricantes.

### **Constrúyalo**

Figura 3-3 muestra el diagrama para este experimento. Es un circuito bastante fácil de construir, así que intentemos armarlo sin el ejemplo de la protoboard. Afortunadamente puede obtener la lista de conexiones del diagrama de abajo. Recuerde que cuando trabaja con las conexiones de un chip, debe usar la marca índice junto con el diagrama de pines para identificar correctamente los pines.



**Figura 3-3**  
Diagrama de Circuito

Lista de conexiones de este esquema:

- Pin 1 del ADC0831 se conecta al pin P0 del BASIC Stamp.
- El cursor del potenciómetro se conecta al pin 2 del ADC0831.
- De los otros dos terminales del potenciómetro, uno se conecta a Vdd en la Plaqueta de Educación y el otro se conecta a Vss.
- Pines 3 y 4 del ADC0831 se conectan a Vss.
- Pines 5 y 8 del ADC0831 se conectan a Vdd.
- Pines 7 y 6 del ADC0831 se conectan a los pines P1 y P2 del BASIC Stamp, respectivamente.

### Prográmelo

El Programa 3.1 es el primer paso para obtener un Voltímetro de CC. Este programa muestra la salida serial de 8-bits del ADC0831. Ingrese el código y guárdelo con el nombre P3\_1R0.bs2. Haremos tres versiones de este programa, así que será importante poder distinguir entre las tres revisiones.

Modificaremos el programa para que también muestre la conversión decimal para el número binario de 8-bits. Luego agregaremos un poco más de código para ajustar el número a una escala de 5 Volts. Asegúrese que el circuito esté correctamente montado, el cable de programación y la alimentación estén conectados y luego ejecute el programa.

```
' Analógico y Digital Básicos - PL3 1R0.bs2
' Programa 3.1 Revision 0.
' {$STAMP BS2}
' {$PBASIC 2.5}

' -----[ Declaraciones ]-----
adcBits    VAR    Byte
v          VAR    Byte
r          VAR    Byte
v2         VAR    Byte
v3         VAR    Byte
```

```

' -----[ Inicialización ]-----
CS          PIN      0
CLK         PIN      1
DataOutput  PIN      2

DEBUG CLS      'Start display.

' -----[ Rutina Principal ]-----
-
DO
  GOSUB ADC Data
  GOSUB Calc_Volts
  GOSUB Display
LOOP

' -----[ Subrutinas ]-----
ADC Data:
  HIGH CS
  LOW CS
  LOW CLK
  PULSOUT CLK, 210
  SHIFTIN DataOutput,CLK,MSBPOST,[adcBits\8]
RETURN

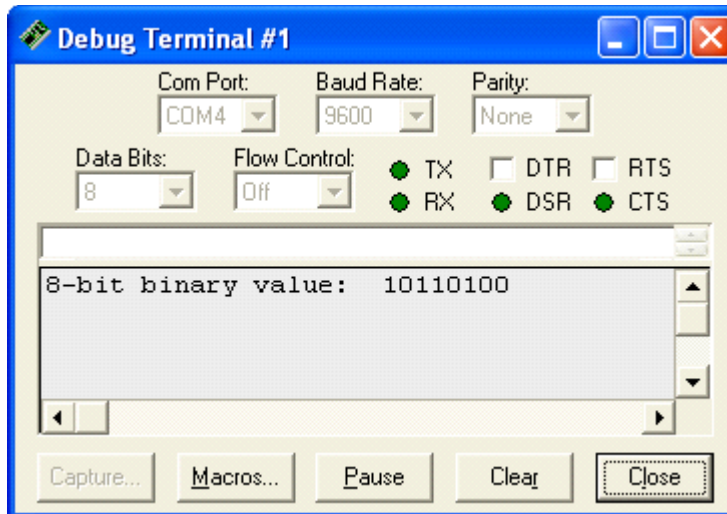
Calc_Volts:
RETURN

Display:
  DEBUG HOME
  DEBUG "Valor binario de 8-bit: ", BIN8 adcBits
RETURN

```

### La Salida

Si el potenciómetro se encuentra aproximadamente en el punto medio, la salida de la pantalla debug podría verse similar a la Figura 3-4. A medida que gira el potenciómetro, los ceros y unos deberían cambiar rápidamente. Cada vez que deja de moverlo, la salida debería permanecer estática mostrando un patrón de ocho ceros y unos.



**Figura 3-4**  
Salida de Debug para el Programa 3.1..

Si su ventana debug responde de esta forma, seguramente su circuito y su programa estarán funcionando correctamente. Si no es así, controle el conexionado de su circuito. También asegúrese de haber escrito correctamente el código. La ventana Debug Terminal podría estar fuera de la vista, actívela con la secuencia Run/Debug/New Terminal en la versión para Windows del software. Algunas veces una letra equivocada hace que el programa no funcione correctamente.

### Explicación del Programa

Las primeras líneas de texto de este programa son comentarios que comienzan con apóstrofes y no tienen ninguna función más allá de brindar información a la persona que lee el código, identificando el modelo del BASIC Stamp y la versión PBASIC.

```
' Analógico y Digital Básicos - PL3_1R0.bs2
' Programa 3.1 Revision 0.
' {$STAMP BS2}
' {$PBASIC 2.5}
```

A continuación está la sección de declaración de variables, que comienza con un comentario. Este programa solamente usa la variable `adcBits`. En las próximas revisiones agregaremos código que usará las otras cuatro variables, `v`, `R`, `v2` y `v3`.

```
' -----[ Declaraciones ]-----
```

```

adcBits      VAR    Byte
v            VAR    Byte
r            VAR    Byte
v2           VAR    Byte
v3           VAR    Byte

```

A continuación hay un nuevo tipo de declaraciones que no hemos usado anteriormente. Se definen tres constantes usando la directiva **PIN**. Después de definir las constantes, podemos usar **CS** en vez del número 0, **CLK** en vez de 1 y **DataOutput** en vez de 2. Los nombres de las constantes se escogieron de acuerdo a los rótulos de los pines del ADC0831. Los números se basan en los pines de E/S del BASIC Stamp.

```

' -----[ Inicialización ]-----
CS          PIN    0
CLK         PIN    1
DataOutput  PIN    2

```

Luego está la rutina principal que contiene tres comandos **GOSUB**. El bucle **DO...LOOP** ejecuta 3 diferentes subrutinas en forma indefinida. Las subrutinas son llamadas **ADC\_Data**, **Calc\_Volts** y **Display**.



**Subrutina:** Una subrutina es un pequeño programa que realiza una tarea específica dentro de un programa más grande.

```

' -----[ Rutina principal ]-----
DO
  GOSUB ADC_Data
  GOSUB Calc_Volts
  GOSUB Display
LOOP

```

Bueno, ¿cómo funciona el comando **GOSUB**? Como se muestra en el diagrama de flujo de la Figura 3-5, **GOSUB ADC\_DATA** da la orden de ir a la subrutina rotulada **ADC\_DATA**: y volver al finalizar su ejecución. El programa salta a la etiqueta **ADC\_DATA**: y comienza a ejecutar comandos. Tan pronto como se encuentra con el comando **RETURN**, el programa regresa a la línea inmediatamente posterior a **GOSUB ADC\_DATA**. En este caso, el siguiente comando es otra instrucción **GOSUB**, más específicamente: **GOSUB CALC\_VOLTS**.

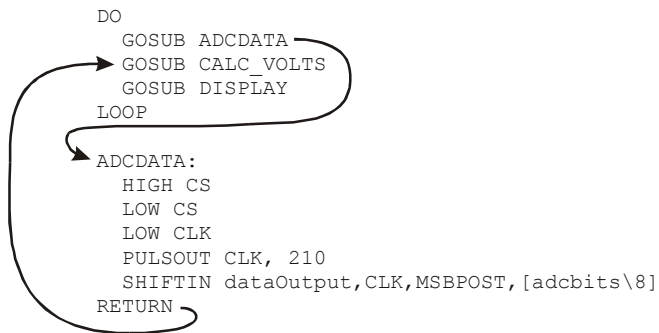

**Figura 3-5**

Diagrama de Flujo

Una subrutina envía el programa a una etiqueta específica. En este caso **ADCDATA:** Luego el programa continúa ejecutando comandos hasta encontrar la instrucción **RETURN**. Esta instrucción regresa la ejecución del programa al comando inmediatamente posterior al **gosub** que originó el salto. En este caso es otro comando **GOSUB**

La subrutina **ADC\_DATA:** envía señales de control y recibe datos del ADC0831. Esta subrutina es donde se demuestra la utilidad de la directiva **con. P0** en el BASIC Stamp está conectado al pin /CS del ADC0831. De la misma forma, los pines P1 y P2 están conectados a CLK y D0. Cuando se envían señales al pin /CS, podemos ingresar un comando como **HIGH CS** en lugar de **HIGH 0**. Tiene más sentido al escribir el código y también lo hace más fácil de leer. También es más fácil modificar el valor de una constante al inicio del programa si se decide conectar el ADC0831 a un pin de E/S distinto del BASIC Stamp.

El comando **HIGH CS** envía una señal de estado alto al pin /CS del ADC0831. Para iniciar la conversión, necesitamos enviar un estado alto (5 Volts), seguido por un estado bajo (0 Volts) en la entrada /CS del ADC0831 usando **LOW CS**. La señal enviada a la entrada /CS del ADC0831 debe permanecer en estado bajo durante la conversión.

```

ADC_Data:
  HIGH CS
  LOW CS
  
```

El comando **LOW CLK** es necesario para que los pulsos de reloj tengan la forma correcta. Al usar este comando se garantiza que al usar la siguiente instrucción (**PULSOUT**) se enviará un pulso de reloj con la forma correcta, bajo-alto-bajo. Usamos los comandos **HIGH** y **LOW** para generar estados altos y bajos como alternativa a **OUT0=1** y **OUT0=0** que empleamos en el primer capítulo.

LOW CLK

El comando **PULSOUT CLK,210** envía un pulso de reloj a la entrada CLK del ADC0831. Este es el primer pulso de reloj y todo lo que hace es avisar al ADC0831 que inicie la conversión en el siguiente pulso de reloj. Por este motivo, no necesitamos controlar el estado de la entrada D0 con anterioridad a este pulso.

PULSOUT CLK, 210

Dado que pusimos el reloj en estado bajo justo antes de este comando, **PULSOUT** envía la señal bajo-alto-bajo deseada. La duración en estado alto es dos veces el número especificado en el comando **PULSOUT**, en microsegundos ( $\mu s$ ).  $1 \mu s = 1/1.000.000$  segundos. Por lo tanto, la duración de este pulso es de  $2 \mu s \times 210 = 420 \mu s$ .

El comando **SHIFTIN D0,CLK,msbpost,[adcbits\8]** es una instrucción poderosa que se encarga de realizar toda la comunicación serial sincrónica, así que no tendremos que programarla manualmente como en el Capítulo 2. En efecto, este comando envía las señales de control a la entrada CLK del ADC0831 y lee los bits de la salida D0 del ADC0831. Este comando también carga cada uno de los bits de salida del ADC 0831 en el byte **adcbits**.

SHIFTIN DataOutput,CLK,MSBPOST,[adcBits\8]

El comando **SHIFTIN** está desarrollado con más detalle en el BASIC Stamp Manual (en Inglés), pero el formato general es así:

**SHIFTIN data\_pin, clock\_pin, mode, [variable\bits]**

En nuestro caso, el pin de datos es **DATA OUTPUT**, una constante igual a 2. Esta constante es usada como referencia al pin de E/S P2 del BASIC Stamp en este programa. Del mismo modo, el pin de reloj es **CLK**, que es una constante igual a 1 y hace referencia al pin de E/S P1 del BASIC Stamp. El modo en este caso es **MSBPOST** que es uno de los cuatro modos de transmisión que pueden ser usados con este comando. Indica que se transmite primero el MSB (bit más significativo) y que los bits de salida del ADC0831 están listos después del flanco descendente del reloj (post). Los parámetros entre corchetes [adcbits\8], indican que los bits se desplazan dentro de la variable **adcbits** y se esperan 8-bits.

La subrutina **Calc\_Volts** está vacía por el momento, pero desarrollaremos su código en breve. Esta subrutina calculará la tensión medida al centésimo lugar decimal.

```
Calc_Volts:
RETURN
```

Hasta el momento, la subrutina **Display** solamente muestra la salida binaria de cada tensión analógica tomada por el ADC0831. Será modificada para mostrar el equivalente decimal del valor binario de 8-bits. También se modificará para mostrar la tensión medida.

El comando **DEBUG-HOME**, envía el cursor a la posición superior izquierda (inicio o "home") de la ventana debug. Luego imprime el mensaje que aparece entre comillas. El modificador **BIN8** hace que el valor de **adcBits** se muestre como 8 dígitos binarios.

```
DEBUG HOME
DEBUG "Valor binario de 8-bit: ", BIN8 adcBits
```

Si la cantidad de dígitos mostrados puede variar, cuando use el comando **DEBUG HOME** siempre especifique la cantidad máxima de dígitos que podría tener en número, usando los modificadores como **bin8**, **dec3**, etc. Cuando se usa **DEBUG CLS**, no es necesario especificar la cantidad de dígitos, así que podrían usarse los modificadores **BIN** y **DEC**.

El comando **DEBUG HOME** es mejor para programas que se repiten constante y rápidamente. Si se usa **DEBUG CLS** bajo estas circunstancias, al limpiar repetidamente la ventana Debug se produce un parpadeo que dificulta la lectura.

El comando **RETURN** envía el programa de regreso a la línea posterior al comando **GOSUB display**.

Modificaremos la subrutina **DISPLAY**: para que muestre el equivalente decimal del contenido binario de **adcBits** en la ventana debug. También se agregará código para mostrar la lectura del voltímetro.



**Interpretando la Salida**

El ADC0831 mide la tensión analógica que se presenta en su entrada. Luego envía al BASIC Stamp un número binario que describe el valor medido. Por ahora, nos concentraremos en una escala de tensión que comienza en 0 Volts y termina en 5 Volts.

Con un número binario de 8-bits, puede comenzar a contar con 00000000 y terminar en 11111111. Traducido a números decimales, es lo mismo que contar de 0 a 255. Cuando se aplica a una escala de 5 Volts que comienza en 0 Volts, es lo mismo que contar de 0 a 5 Volts usando 255 pasos de tensión.

Para la escala de 5 Volts, cuando el ADC0831 mide 0 Volts, usted obtiene 00000000. Cuando mide 5 Volts, la salida es 11111111. Podemos calcular que la salida de la ventana debug de 10110100 de la Figura 3-4 es igual al número decimal 180. Esto corresponde a una tensión medida de 3.53 Volts.

**Repaso de Conversión Binaria a Decimal**

¿Cómo sabemos que se pueden obtener 256 combinaciones de un número binario de 8-bits? Recuerde, siempre puede decir la cantidad de números (combinaciones de 0s y 1s) que se pueden obtener de una cantidad determinada de bits usando la fórmula del Capítulo 2

$$\text{combinations} = 2^{\text{bits}}$$

Esto significa que la cantidad de combinaciones es igual a dos elevado a la cantidad de bits. Para 8-bits, la cantidad de combinaciones es  $2^8 = 256$ . Para 12-bits, la cantidad de combinaciones es  $2^{12} = 4096$  y así.

Usemos el método de dos pasos del Capítulo 2 para convertir el número binario de 8-bits 10100101 a su equivalente decimal. Repetiremos la tabla de multiplicadores para trabajar con ella:

<b>Table 3-1: Multiplicadores de Bits para un Número Binario de 8-bits</b>								
<b>Bit</b>	7	6	5	4	3	2	1	0

Multiplicador	128	64	32	16	8	4	2	1
---------------	-----	----	----	----	---	---	---	---

Primero, multiplique cada bit por su potencia de dos de la Tabla 3.1

$128 \times 1 = 128$   
 $64 \times 0 = 0$   
 $32 \times 1 = 32$   
 $16 \times 0 = 0$   
 $8 \times 0 = 0$   
 $4 \times 1 = 4$   
 $2 \times 0 = 0$   
 $1 \times 1 = 1$

Segundo, sume los 8 valores decimales:

$128 + 0 + 32 + 0 + 0 + 4 + 0 + 1 = 165$

Ahora sabemos que el número binario 10100101 es igual al número decimal 165. Para mostrar esta conversión, agregaremos un comando **DEBUG** a la subrutina **DISPLAY**: Las líneas agregadas están señaladas con "□".

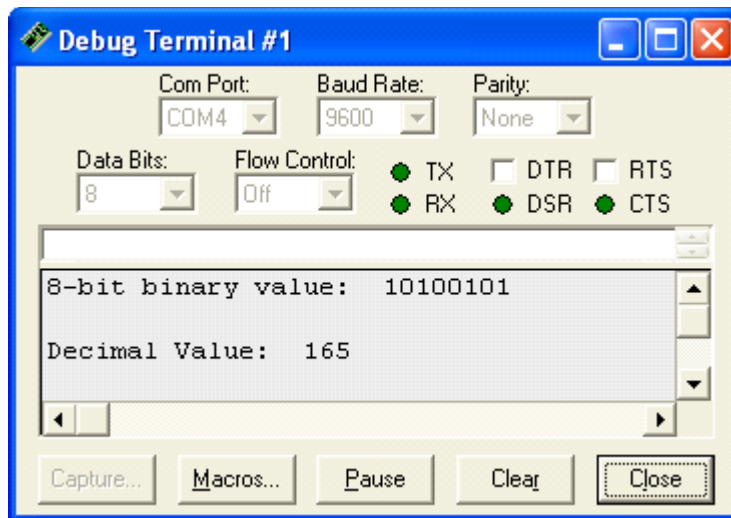
```

Display:
  DEBUG HOME, "Valor binario de 8 bits:  ", BIN8 adcBits
  DEBUG CR, CR, "Valor decimal:  ", DEC3 adcBits  '□ nueva
RETURN

```

El comando **DEBUG CR, CR, "Valor decimal:", dec3 adcbits** le dice a la ventana debug que ejecute dos saltos de línea y muestre el mensaje entre comillas, seguido por el valor decimal de 3 dígitos de **adcBits**. Si el valor real solamente tiene uno o dos dígitos, la ventana debug completará los espacios faltantes con ceros debido al uso de **DEC3**. Por ejemplo, el número 7 se mostrará como 007 y el número 85 como 085, etc

Con un ajuste cuidadoso del potenciómetro, podremos lograr una salida como la de la Figura 3-6.



**Figura 3-6**  
Salida de Debug  
para el Programa  
3.1, Revisión 1.

### Cálculo de la Tensión

Ahora que sabemos el equivalente decimal de la salida binaria del ADC0831, podemos realizar unos cálculos para obtener la tensión medida. Para averiguar a que tensión corresponde el número decimal, debemos calcular a que punto del rango de tensión corresponde este número. Éste es un método para razonar el problema.

- Sabemos que la tensión está en el rango de 0 a 5 Volts y también sabemos que la salida del ADC0831 está en el rango de 0 a 255.
- En otras palabras, la tensión es a 5 como la salida del conversor es a 255.

Esto se traduce en la ecuación:

$$\frac{\text{Voltage}}{5} = \frac{\text{Decimal A/D Output}}{255}$$

Podemos despejar esta ecuación para calcular la tensión:

$$\text{Voltage} = \frac{5 \times (\text{Decimal A/D Output})}{255}$$

Ahora sabemos que multiplicando por 5 y dividiendo por 255 obtenemos una escala de 5 Volts con 256 niveles. Podemos calcular la tensión de la Figura 3-6 donde la salida del ADC0831 es 10100101 = 165. La tensión medida es:

$$\text{Voltage} = \frac{5 \times 165}{255} = 3.24 \text{ Volts rounded to two decimal places.}$$

Para calcular esta tensión usando el BASIC Stamp, deberemos agregar código a las subrutinas **CALC\_VOLTS:** y **DISPLAY** Primero, la ecuación de tensión debe ser expresada en código PBASIC. Este es un ejemplo de cómo pretendería usted resolver esta ecuación.

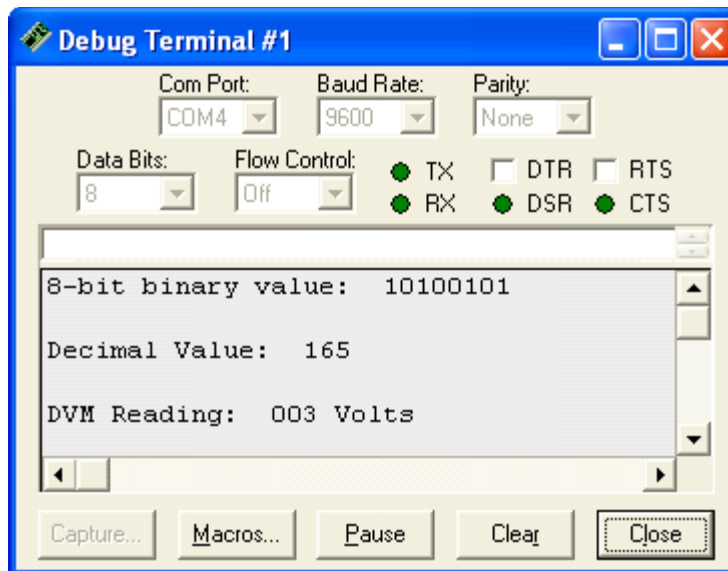
```
v = 5 * adcBits / 255
```

Este cálculo en PBASIC al parecer nos daría la salida que buscamos, pero no lo hará. Igualmente intentaremos usar esta ecuación para ver que ocurre. Modifique las subrutinas **CALC\_VOLTS:** y **DISPLAY:** en el Programa 3.1 como a continuación:

```
Calc_Volts:
  v = 5 * adcBits/255           '□ nueva línea
RETURN

Display:
  DEBUG HOME
  DEBUG "Valor binario de 8-bits: ", BIN8 adcBits
  DEBUG CR, CR, "Valor decimal: ", DEC3 adcBits
  DEBUG CR, CR, "Lectura: ", DEC3 v, " Volts" '□ nueva línea
RETURN
```

Calculamos que 165 correspondería a una tensión de entrada de 3,24 Volts. Los 003 Volts que aparecen en la Figura 3-7 tienen redondeo a valores enteros. ¿Qué sucedió?



**Figura 3-7**  
Salida de Debug para  
el Programa 3.1,  
Revisión 2.

Los comandos PBASIC del BASIC Stamp trabajan solamente con aritmética entera. Los enteros son los números que usamos para contar: ...-2, -1, 0, 1, 2, 3, etc. El entero más grande que puede procesar un BASIC Stamp es 65535. Cuando se usa aritmética entera, la parte fraccionaria de la respuesta es descartada. Afortunadamente, aún podemos usar aritmética entera para encontrar los valores fraccionarios que estamos intentando mostrar.

Antes de la división, la salida del conversor A/D output es multiplicada por 5. Esto no causa problemas.

$5 \times (\text{Decimal A/D Output})$  es esencialmente lo mismo que  $5 \times \text{adcBits}$ .

En nuestro ejemplo de cálculo de tensión es dado que 825 es un entero que es menor que 65535, esta parte del cálculo se realiza sin problemas. El inconveniente se presenta al dividir 825 por 255. La respuesta tiene un componente fraccionario que nunca se calcula con aritmética entera.

Si hace la división con papel y lápiz, necesitará varios pasos y solamente usará matemática entera. Veamos cómo calcular la respuesta para este problema de división.

$$\text{Voltage} = \frac{5 \times (\text{Decimal A/D Output})}{255} = 3 + \text{a remainder of } 60$$

El proceso para calcular la parte fraccionaria de la división es repetitivo. Se multiplica el resto por 10, luego se divide por 255, obteniendo otro resto que se multiplica por 10 y divide por 255 otra vez, etc. Para calcular directamente los dos primeros decimales podemos tomar el resto y multiplicarlo por 100, luego dividimos por 255. Intentémoslo.

$$(60 \times 100) \div 255 = 6000 \div 255 = 23.5924... \xrightarrow{\text{Integer Math}} 23$$

Recuerde: el BASIC Stamp descarta todo lo que se encuentre a la derecha de la coma sin redondear el número. Esto se llama truncar. El resultado que obtendremos es 23. Este resultado debería haber sido redondeado a 24 debido a que 23,5294 está más cerca de 24 que de 23. Por ahora, usemos a 23 a la derecha de la coma.

Nuestra respuesta usando este algoritmo es el entero 3 a la izquierda de la coma y el entero 23 a la derecha. Dado que solamente usamos aritmética entera en este cálculo, debería funcionar con PBASIC en el BASIC Stamp.



**Algoritmo:** Procedimiento para resolver un problema. El procedimiento se descompone en pasos repetitivos

Dado que el BASIC Stamp trabaja con enteros, no nos sorprende que exista un comando PBASIC para calcular el resto entero de una división. El operador para la división es / y el del resto es //. Intentemos convertir nuestro **algoritmo** en código PBASIC para que realice la cuenta por nosotros. Los pasos de la división que mostramos a continuación, corresponden a los comandos PBASIC que usaremos.

$$\begin{array}{l} \overline{) 5 \times \text{adcBits}} \quad \text{v+r} \quad \leftarrow R = (5 \times \text{adcBits} // 255) \\ 255 \overline{) 5 \times \text{adcBits}} \end{array}$$
  

$$\begin{array}{l} \overline{) 100 \times r} \quad \text{v2} \\ 255 \overline{) 100 \times r} \quad \text{v2} = (100 \times R) / 255 \end{array}$$

De aquí obtenemos los 3 comandos PBASIC para calcular los valores a izquierda y derecha de la coma decimal. Para reconstruir el valor fraccionario en la pantalla, imprimiremos una coma "," entre ambos valores.

El primero de los tres comandos ya se encuentra en nuestra subrutina **CALC\_VOLTS:**. Agregue las otras dos instrucciones para completar el algoritmo.

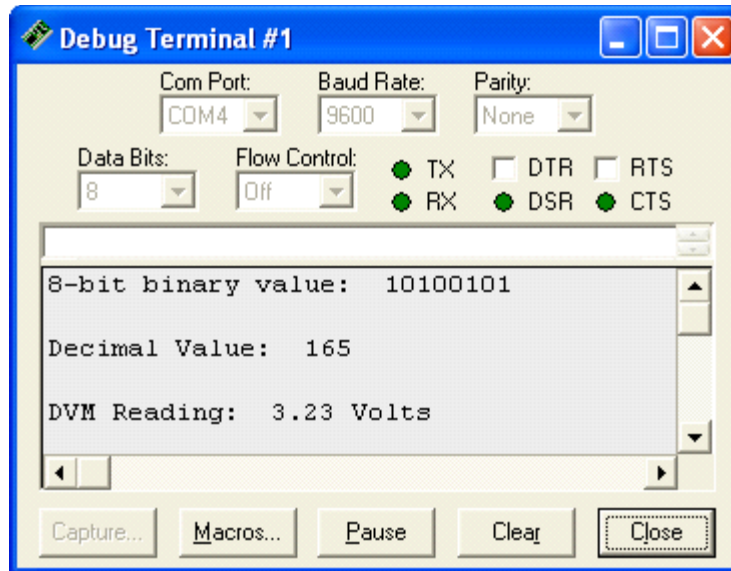
```
Calc_Volts:
  v = 5 * adcBits / 255
  r = 5 * adcBits // 255          '□ nueva línea
  v2 = 100 * R / 255             '□ nueva línea
RETURN
```

La subrutina **DISPLAY** también debe ser actualizada para imprimir los dos valores con una coma en el medio. Asegúrese de actualizar la línea de la subrutina **DISPLAY** exactamente como se muestra abajo.

```
Display:
  DEBUG HOME
  DEBUG "Valor binario de 8-bit: ", BIN8 adcBits
  DEBUG CR, CR, "Valor decimal: ", DEC3 adcBits
  DEBUG CR, CR, "Lectura DVM: "          '□ nueva línea
  DEBUG DEC1 v, ".", DEC2 v2, " Volts"   '□ nueva línea
RETURN
```

Ahora ejecute nuevamente el programa y vea lo que sucede. La Figura 3-8 muestra un ejemplo de la salida. El programa está casi listo para mostrar el valor más cercano a la centésima de Volt. Solamente debemos corregir el error de redondeo en las centésimas.





**Figura 3-8**  
Salida de  
Debug Para el  
Programa 3.1,  
Revisión 3.

3

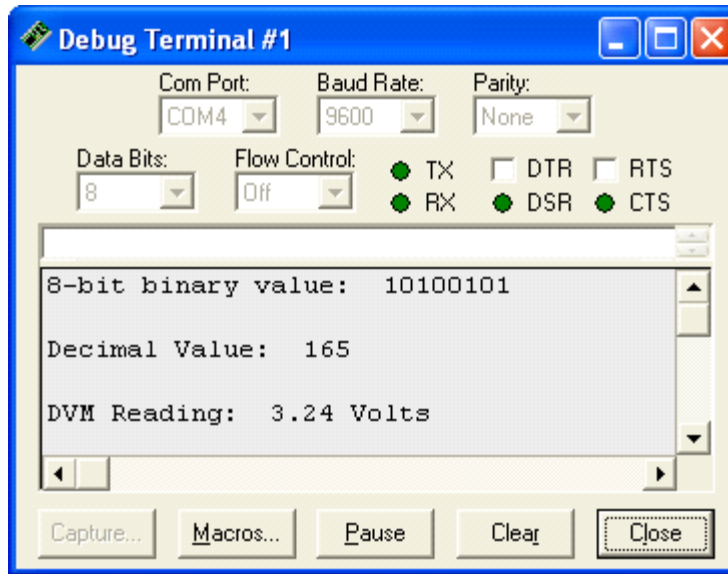
Este problema de redondeo puede ser corregido agregando el segmento de código que se muestra abajo a la subrutina **CALC\_VOLTS**:

```
Calc_Volts:
  v = 5 * adcBits / 255
  r = 5 * adcBits // 255
  v2 = 100 * R / 255
  v3 = 100 * R // 255
  v3 = 10 * v3 / 255
  IF (v3 >= 5) THEN v2 = v2 + 1
  IF (v2 >= 100) THEN
    v = v + 1
    v2 = 0
  ENDIF
RETURN
```

'□ nueva línea  
'□ nueva línea  
'□ nueva línea  
'□ nueva línea  
'□ nueva línea  
'□ nueva línea  
'□ nueva línea

## La Salida

El ejemplo de salida de la Figura 3-9 indica que el voltímetro ahora calcula correctamente las centésimas de Volt.



**Figura 3-9**  
Salida de Debug para  
el Programa 3.1,  
Revisión 4.



Tan pronto como esté seguro de que su programa funciona correctamente, guárdelo como P3\_1R4.bs2. En el próximo capítulo usaremos este código y el circuito.

## Explicación del Programa

Para redondear correctamente las centésimas, necesitamos conocer el dígito de las milésimas. Usando las reglas de división entera, podemos simplemente crear una variable **v3** igual al resto del cálculo de **v2**, y dividir nuevamente por 255.

```
v3= 100 * R // 255
v3 = 10 * v3 / 255
```

Para evitar el uso de otra variable, **v3** es redefinida en la segunda línea. El valor de **v3** a la derecha del signo igual es el que se calculó en la primera línea. El valor de **v3** a la izquierda del igual es el valor redefinido, que es diez veces el anterior **v3**, dividido por 255.

Este proceso podría repetirse una y otra vez para obtener el dígito en la posición la diezmilésima de Volt, cienmilésima y así sucesivamente.

Una vez que se conoce el valor del dígito de las milésimas, se aplican las siguientes reglas de redondeo:

- Si el dígito de las milésimas es menor a 5, saltar agregar 1 a las centésimas.
- Caso contrario, agregar 1 a las centésimas.
- En todos los casos, truncar los valores más allá de las centésimas.

Dado que el valor **v2** ya está truncado, solamente necesitamos el código para decidir si le sumamos 1 o no a las centésimas. Mostramos una versión posible a continuación.

```
IF (v3 >= 5) THEN v2 = v2 + 1
```

Dado que el valor de las unidades está almacenado en otra variable, necesitamos verificar si al agregar uno a las centésimas es necesario o no, incrementar ese valor. Sin este código, 3,996 se redondearía a 3,00 en lugar de a 4,00.

```
IF (v2 >= 100) THEN
  v = v + 1
  v2 = 0
ENDIF
```

Guarde este programa y, si es posible, deje el circuito como está, debido a que podríamos usar el voltímetro para tomar mediciones en el circuito que construiremos en el Capítulo 4.

### **Resolución**

El BASIC Stamp está ahora programado para calcular con precisión la tensión asociada a la salida binaria del ADC0831 con una aproximación de centésimas de Volt. Aunque se eliminaron las fuentes de error provenientes del cálculo, hay otra fuente de error que es la limitación en la resolución del conversor A/D.

El chip conversor A/D que estamos usando es capaz de discriminar 256 valores binarios. Esto significa que cada medición de tensión se redondea a uno de 256 valores discretos. El tamaño del salto es el rango de tensión entre estos valores discretos. Dado que el primer valor es cero, hay 255 pasos de tensión. El tamaño del paso está dado por:

$$\text{Step Size} = \frac{5 \text{ Volts}}{255 \text{ steps}} = 0.0196 \text{ Volts/step} \approx 0.02 \text{ Volts/step}$$

Con esto en mente, cada vez que mueve el potenciómetro, el conversor aproxima su salida al valor analógico, pero no es exacto debido a las limitaciones por la resolución. Así, todavía hay cierta incertidumbre sobre el valor real de las centésimas. En algunas aplicaciones, la incerteza se muestra junto con el valor. Asumiendo que el ADC0831 redondea a mitad de salto de tensión, podríamos usar esta convención para mostrar el valor de tensión: "3,24 Volts más o menos 0,01 Volts."

Existen conversores de mayor resolución, como de 12 y 16 bits (y mayores también), pero debido a su mayor precisión, su costo también se incrementa. La mejora en la resolución es significativa. Como se mencionó antes, un conversor de 12-bits le dará una resolución de 4095 pasos. Esto resulta en 5 volts / 4095 pasos, o un paso cada 0,0012 Volts. Un conversor de 12-bits normalmente cuesta más que uno de 8-bits. También hay un costo en términos de la cantidad de memoria necesaria para tomar la medición, (12 en oposición a 8-bits) y en la cantidad de procesamiento también (13 pulsos de reloj en lugar de 9).

### **Calibración**

¿Qué sucede si la fuente de alimentación de la Plaqueta de Educación suministra 4,963 Volts en lugar de 5,000 Volts? El voltímetro del BASIC Stamp puede ser calibrado usando un voltímetro de alta precisión. La diferencia entre Vdd y Vss puede ser medida usando el voltímetro preciso. El desarrollo del código para corregir este error requiere la representación de más valores fraccionarios usando aritmética entera, lo que dejamos como problema para el alumno.

Otro tema a tener en cuenta cuando se busca alto grado de precisión es que diferentes consumos de corriente sobre la fuente de alimentación pueden causar variaciones en la tensión de referencia. En este caso se requeriría equipamiento adicional. Como se imagina, el desarrollo de instrumental con un alto grado de precisión involucra muchos desafíos de diseño. Para los experimentos restantes, la precisión actual de nuestro voltímetro es suficiente.

**¿Qué aprendí?**

En las oraciones de abajo, inserte las palabras apropiadas de la lista de la izquierda.

3

binario	Al usar el conversor A/D se hace posible procesar información _____ con el BASIC Stamp, un dispositivo digital ( _____ ).
resolución	El conversor usado en este experimento es el circuito integrado ADC0831, un conversor A/D _____ de 8-bit.
analógica	Para una _____ analógica dada, el conversor A/D emite un número binario de _____. El BASIC Stamp puede ser usado para controlar y recolectar datos de un conversor A/D. Varias técnicas de programación pueden ser usadas para leer, almacenar y mostrar estos datos.
serial	
entrada	La representación digital de la señal analógica convertida es muy buena, pero no perfecta debido a las limitaciones en la _____ inherentes al conversor A/D. Una segunda fuente de error para el voltímetro del Stamp, puede resultar del hecho que la fuente de alimentación de la Plaqueta de Educación no necesariamente suministre exactamente 5 Volts.
8-bit	

### **Preguntas**

1. Con sus propias palabras, explique la función de un conversor A/D.
2. ¿Cuál sería la resolución si se usara un conversor A/D de 16-bit en este experimento?
3. ¿Cómo se relaciona la ecuación de divisor de tensión con el cursor del potenciómetro? ¿Cuál sería el valor esperado en la salida si las resistencias son iguales? ¿Puede probar esto?
4. ¿En qué se diferencian las lecturas de tensión sobre el cursor del potenciómetro de este experimento con las del Capítulo 1? ¿Qué se ganó al usar el ADC0831 para las mediciones de tensión, en lugar de conectar directamente un pin de E/S del BASIC Stamp?
5. Dada la resolución de nuestro conversor A/D de 8 bit, cuando la tensión sobre el potenciómetro es de 3,6 volts, ¿qué valor decimal se mostrará? ¿Cuál es el valor binario?

### **Desafío**

1. Use un cable para conectar el cursor del potenciómetro a un pin del BASIC Stamp que esté sin usar. Agregue una subrutina al programa del voltímetro que controle el estado de este pin, configurado como entrada. Determine si el umbral de tensión que usó en el Capítulo 1 es en realidad de 1,4 Volts.
2. Escriba un programa que controle el valor analógico de un potenciómetro de 100 k $\Omega$  y avise cuando se supere un cierto valor preestablecido.
3. Escriba un programa y arme un circuito que cree una “zona de seguridad” entre 1,0 Volt y 2,0 Volts. Si la tensión analógica sale de este intervalo, un LED parpadea.
4. Dibuje el diagrama completo para el Desafío 3 y modifique el programa para que el LED solamente se encienda cuando la tensión (en el cursor del potenciómetro) sea exactamente igual a 2,0 Volts.

5. Asuma que la fuente de alimentación de la Plaqueta de Educación suministra 4,960 Volts. Desarrolle una subrutina para ajustar las mediciones de tensión a esta escala.

### **¿Por qué aprendí esto?**

Hay una gran variedad de aplicaciones electrónicas donde se miden señales analógicas y se usan dispositivos digitales para procesar los datos de las señales analógicas. En este experimento, usamos un BASIC Stamp y un conversor A/D para construir un voltímetro de CC digital. Usaremos esta herramienta en varios de los experimentos restantes. Hay una sorprendente variedad de usos para tal dispositivo como descubrirá en cada experimento.

### **¿Cómo puedo aplicarlo?**

Para desarrollar el voltímetro digital se introdujo: el proceso de muestreo de tensión, el proceso de conversión y el procesamiento digital de datos. Un ejemplo de otro uso para una interfaz A/D es el muestreo digital de la señal analógica de un micrófono, para aplicaciones de grabación digital. Otro ejemplo que podría usar el circuito que construimos es un sensor de apertura de puerta. Nuestro potenciómetro podría estar conectado a la bisagra de la puerta y la información analógica podría usarse para controlar qué tan abierta está la puerta. Este circuito podría incorporarse a un sistema más grande que se encargue de controlar la apertura de la puerta.

El campo de la conversión analógica a digital es una industria en sí mismo. Hay compañías de fabricación de semiconductores que se especializan únicamente en la creación de chips y sistemas de conversión A/D. Ya sea que se encuentre diseñando a nivel de componentes o a nivel de circuitos integrados, siempre necesitará interfases analógicas creativas, simplemente debido a que el mundo no es blanco y negro (binario), sino que es de todos los colores imaginables (analógico).





## Capítulo : Conversión Digital a Analógica Básica

### CONSTRUCCIÓN DE UNA RED RESISTIVA EN ESCALERA

La conversión digital a analógica (conversión D/A) es, en líneas generales, la inversa de la conversión A/D. En la conversión A/D, comenzábamos con un rango continuo de tensión en la entrada del conversor. El conversor A/D redondeaba al paso de tensión más cercano al real y enviaba un número binario indicando el paso medido.

4

La conversión D/A comienza con un número binario como entrada y la salida es un paso de tensión. Mientras que el proceso de un conversor A/D comienza con una entrada analógica y finaliza con una salida binaria, el proceso de un conversor D/A comienza con una entrada binaria y finaliza con un paso de tensión como salida. No es un valor analógico verdadero que varía continuamente, sino que es una tensión discreta que varía a pasos.

El término resolución se introdujo al final del Capítulo 3. Dado que la salida de un conversor D/A siempre será redondeada a un valor de paso de tensión (valor de tensión discreto) es importante seleccionar correctamente la resolución del conversor D/A. Recuerde que con resoluciones más altas obtendrá mayor precisión, pero siempre deberá pagar un mayor costo en recursos, memoria y pasos de procesamiento.

La cantidad de niveles de tensión que un conversor D/A puede producir, está determinada por la cantidad de bits binarios que puede manejar, lo que está expresado en la resolución. Podemos usar nuevamente la ecuación de combinaciones para obtener la cantidad.

$$\text{combinations} = 2^{\text{bits}}$$

El conversor D/A que usaremos en este experimento tiene una resolución de 4-bits, así que la cantidad de niveles de tensión de salida será:

$$\text{combinations} = 2^{\text{bits}} = 2^4 = 16$$

En el Capítulo 3, usamos un circuito integrado para realizar la conversión A/D. En este capítulo, construiremos un conversor D/A usando resistores. Este circuito se llama red resistiva en escalera y se le pueden quitar o agregar resistores para modificar la resolución. Con una red resistiva en escalera, si comienza con un conversor de 4-bits y quiere aumentar la resolución en 1-bit, solamente debe agregar dos resistores a la red.

En este capítulo construiremos una red resistiva en escalera y programaremos al BASIC Stamp para lograr que la red efectúe la conversión D/A. El PBASIC se usará para programar al BASIC Stamp para que envíe a la red un conjunto de niveles de tensión binarios. Estos se convertirán en tensiones discretas de salida en la red resistiva en escalera.

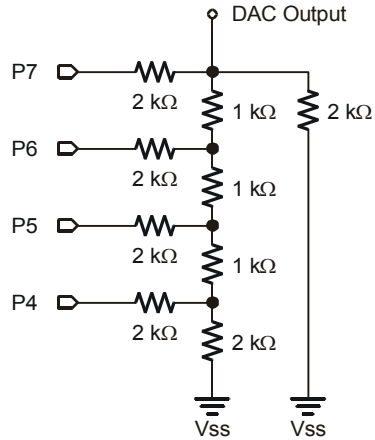
El voltímetro del Capítulo 3 se usará para medir las tensiones de salida del conversor. La medición de todos los niveles de salida del conversor D/A se llama barrido de tensión. Usaremos PBASIC para automatizar esta tarea. De esta forma, las tensiones de salida del conversor D/A se pueden medir sin tener que repetir manualmente este proceso.

### **Componentes Requeridos**

Separe estos componentes antes de comenzar:

- (6) Resistores de 2 k Ohm
- (3) Resistores de 1 k Ohm
- (1) Conversor ADC0831 A/D
- (1) LED rojo
- (1) Resistor de 270 Ohm

La red resistiva en escalera para este capítulo se muestra en la Figura 4-1. El nombre surge del hecho de que el diagrama de la red resistiva, se ve como una escalera. Es una alternativa muy barata en comparación con un circuito integrado de un conversor digital a analógico (conversor D/A o DAC). Los resistores solamente cuestan una fracción de lo que cuesta un circuito integrado.



**Figura 4-1**  
Red Resistiva en Escalera.

*Esta red puede ser usada como conversor D/A. La entrada del número binario se produce en paralelo como 4-bits por las líneas P4 a P7. Mientras que el valor de los cuatro bits esté presente, la salida del conversor D/A (DAC output) tendrá el mismo valor discreto de tensión.*

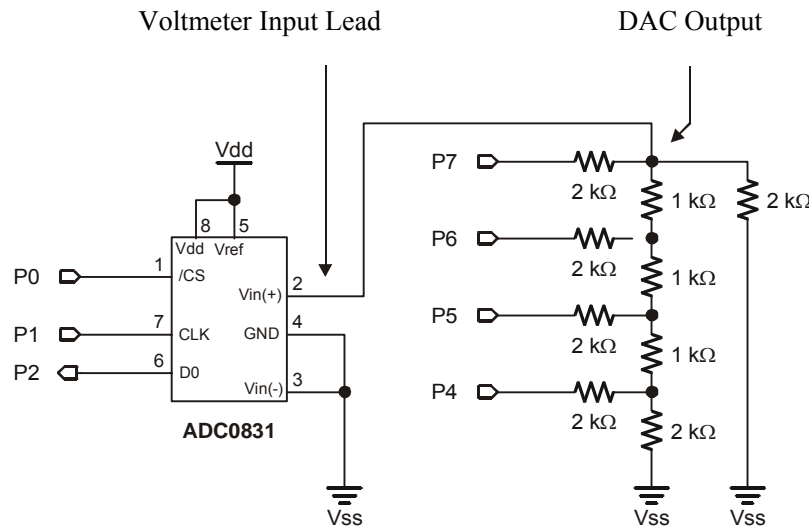
4

Siendo este el caso, ¿por qué no usan todos las redes resistivas en escalera para las conversiones A/D y D/A? La red resistiva en escalera es también usada en muchos circuitos integrados de conversores A/D y D/A, tales como el ADC0831. Los resistores usados en los circuitos integrados son implantes microscópicos sobre la superficie de una pastilla de silicio. Una ventaja de los conversores en circuito integrado es que tienen un alto grado de precisión. Otra ventaja de los circuitos integrados es que incluyen más componentes internamente, como en el caso del seguidor de tensión del Capítulo 1.

### **Constrúyalo**

Arme el circuito que se muestra en la Figura 4-2. Preste mucha atención a los valores de los resistores así como también al modo en que se conectan. Si su circuito del Capítulo 3 aún está intacto, simplemente quite el potenciómetro y construya la red resistiva en escalera cerca de los conectores de los pines P4 a P7. El pin de entrada del voltímetro que estaba conectado al cursor del potenciómetro, debería conectarse a la salida del conversor D/A. Asegúrese de prestar mucha atención para evitar que los resistores se toquen entre sí fuera de los nodos de la protoboard.

Es mejor que intente montar el circuito directamente del diagrama de la Figura 4-2. Este será un circuito difícil de hacer entrar en la protoboard. La ubicación de los componentes y la forma de interconectarlos normalmente queda a criterio de la persona que lo arma, ¡así que está por su cuenta!



**Figura 4-2**  
Diagrama de circuito.

*El voltímetro de CC del Capítulo 3 es conectado a la salida del conversor D/A de red resistiva en escalera.*

### Prográmelo

No solo podemos usar el voltímetro para medir la tensión de salida del conversor D/A, sino que podemos automatizar el proceso de medición de los 16 niveles de tensión de salida del conversor D/A. Puede que no nos ahorremos mucho trabajo debido a que solamente se trata de 16 mediciones pero, ¡imagine intentar controlar los 4096 pasos de tensión de un conversor de 12-bits!

Con algunos agregados relativamente simples al código del Capítulo 3, que fue guardado con el nombre P3\_1R3.bs2, podremos controlar ambos dispositivos. El PBASIC puede usarse para ordenarle al BASIC Stamp que envíe una señal de salida al conversor D/A. El código para esto se agregará a la versión final del Programa 3.1. De esta forma podemos usar nuestro voltímetro para medir la salida del conversor D/A.

El Programa 4.1 se muestra abajo. Es la última modificación del Programa 3.1 con el agregado de una subrutina rotulada DAC: para enviar tensiones binarias al conversor D/A. Hay algunos pocos cambios adicionales que fueron señalados usando el comentario '□ que significa agregar esta línea y 'Δ que indica las líneas que han sido modificadas.

Si guardó el programa del Capítulo 3, agregue y modifique el código para este experimento y guárdelo con el nombre PL4\_1R0.bs2. Si no tiene el código del Capítulo

3, ingrese todo el programa de abajo usando el editor del BASIC Stamp y asegúrese de guardarlo para usarlo en el futuro. Una vez montado el circuito y tecleado el programa, ejecute el Programa 4.1 y vea cómo trabaja.

```
' Analógico y Digital Básicos - PL4_1R0.bs2      'Δ
' Voltimetro Digital (DVM). Conversor D/A agregado  'Δ
' {$STAMP BS2}
' {$PBASIC 2.5}

' -----[ Declaraciones ]-----
adcBits      VAR    Byte
v            VAR    Byte
r            VAR    Byte
v2           VAR    Byte
v3           VAR    Byte
n            VAR    Nib      '□

' -----[ Inicialización ]-----
CS           PIN    0
CLK          PIN    1
DataOutput   PIN    2

DEBUG CLS                                'Inicia ventana debug

' -----[ Rutina Principal ]-----
DO
  GOSUB DAC                                '□
  GOSUB ADC_Data
  GOSUB Calc_Volts
  GOSUB Display
LOOP

' -----[ Subrutinas ]-----
DAC:
  n = 11                                '□

  OUTPUT 7                                '□
  OUTPUT 6                                '□
  OUTPUT 5                                '□
  OUTPUT 4                                '□

  OUT7 = n.BIT3                          '□
  OUT6 = n.BIT2                          '□
  OUT5 = n.BIT1                          '□
  OUT4 = n.BIT0                          '□
RETURN                                  '□

ADC_Data:
  HIGH CS
  LOW CS
```

```

LOW CLK
PULSOUT CLK, 210
SHIFTIN DataOutput,CLK,MSBPOST,[adcBits\8]
RETURN

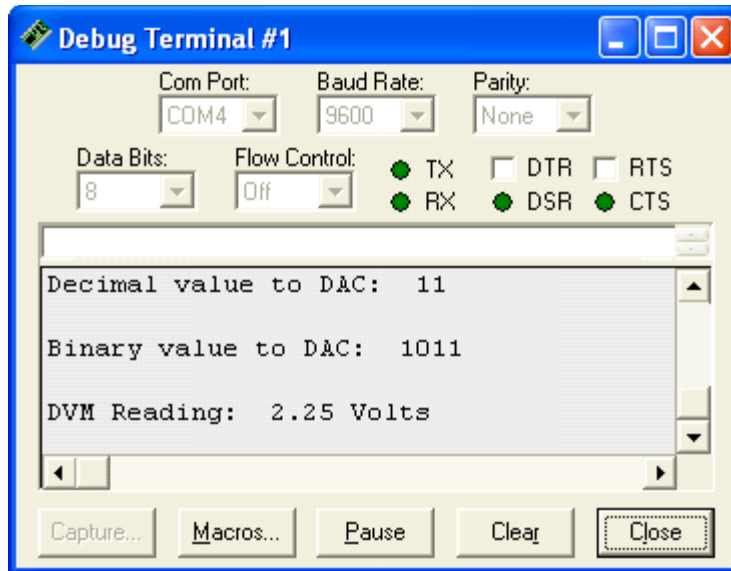
Calc_Volts:
v = 5 * adcBits / 255
r = 5 * adcBits // 255
v2 = 100 * R / 255
v3 = 100 * R // 255
v3 = 10 * v3 / 255
IF (v3 >= 5) THEN v2 = v2 + 1
IF (v2 >= 100) THEN
    v = v + 1
    v2 = 0
ENDIF
RETURN

Display:
DEBUG HOME, CR, CR, "Valor decimal del DAC: ", DEC2 n          'Δ
DEBUG CR, CR, "Valor binario del DAC: ", BIN4 n                'Δ
DEBUG CR, CR, "Lectura DVM g: ", DEC1 v, ".", DEC2 v2, " Volts" 'Δ
RETURN

```

## La Salida

Con valores perfectos de resistores, la salida máxima debería ser de 3,00 Volts. Los resistores usados en este ejemplo tienen una tolerancia del 10%. Esto significa que la resistencia medida en cada resistor debería tener un valor dentro de  $\pm 10\%$  del valor supuesto. Debido a esto, son de esperar ligeras variaciones en la salida, tales como la medida en la Figura 4-3. Con resistores de valores exactos, la medición debería haber sido 2,20 Volts.



**Figura 4-3**  
Salida de Debug para  
el Programa 4.1.

4

### Explicación del Programa

El primero de los dos comentarios fue actualizado para indicar que se trata del Programa 4.1. El segundo indica que se agregó una función al voltímetro que realiza la conversión D/A.

```
' Analógico y Digital Básicos - PL4_1R0.bs2      'Δ
' Voltímetro Digital (DVM). Conversor D/A agregado  'Δ
' {$STAMP BS2}
' {$PBASIC 2.5}
```

Una variable tipo nibble **n** se agrega en la sección de declaración de variables y será usada para almacenar el valor binario para el conversor D/A.

```
n      VAR    Nib      '□
```

Se agrega un comando **GOSUB** a la rutina principal que envía el programa a la subrutina **DAC**:

```
GOSUB DAC      '□
```

Este es el inicio de la subrutina de conversión digital a analógica (DAC), que recibió la etiqueta DAC: El valor de **n** se fija en 11. Esto significa que la salida debería estar **n**

pasos por encima de 0 para una escala de 0 a 16. El valor de **n** puede ser modificado para especificar la tensión.

```
DAC:                                '□
n = 11                              '□
```

Los pines de E/S del BASIC Stamp conectados al conversor D/A se configuran como salidas. Estos comandos normalmente se encuentran en la sección de declaraciones. Si se hubiesen incluido en ese sector, el programa se ejecutaría más rápido debido a que estos comandos se ejecutarían una sola vez al inicio del programa. En lugar de eso, se ejecutan cada vez que se llama la subrutina. La razón por la que incluimos estas instrucciones en la subrutina es para expandir el conocimiento sobre las propiedades de PBASIC.

```
OUTPUT 7                            '□
OUTPUT 6                            '□
OUTPUT 5                            '□
OUTPUT 4                            '□
```

Luego, la salida paralela binaria del BASIC Stamp se envía al conversor D/A. Usamos los mismos comandos que en los Capítulos 1 y 2, pero agregamos una nueva característica. La variable **n** tiene una extensión que indica que **BIT** del **nibble** completo está siendo usado. Por ejemplo, el comando **out7=n.bit3** iguala el valor del pin de salida P7 al valor del bit3 de la variable **n**. Dado que asignamos el valor de **n** igual a 11, el valor binario de **n** es 1011. El bit 3 es el de la izquierda del número binario, en este caso 1, lo que significa que P7 es puesto en nivel alto. También para **n = 11**, P6 está en bajo, P5 en alto y P4 en alto.

```
OUT7 = n.BIT3                       '□
OUT6 = n.BIT2                       '□
OUT5 = n.BIT1                       '□
OUT4 = n.BIT0                       '□
```

Eso es todo lo que se necesita para programar una conversión digital a analógica, usando una red resistiva en escalera. El comando **return** envía la ejecución del programa a la instrucción inmediatamente posterior al comando **GOSUB DAC** de la rutina principal:

```
RETURN                              '□
```

Las primeras dos líneas de la subrutina **DISPLAY:** se modificaron para mostrar los valores decimal y binario de **n**.

```
DEBUG HOME, CR, CR, "Valor decimal del DAC: ", DEC2 n    'Δ
DEBUG CR, CR, "Valor binario del DAC: ", BIN4 n           'Δ
```



**Modifique el Código**

Si el conversor D/A funciona como se esperaba, cada vez que se incrementa el valor de  $n$ , la salida del conversor D/A debería incrementarse 0,2 Volts. Intente comenzar con  $n=0$  modificando el valor de  $n$  en la subrutina **DAC**:

```
n = 0                                     ' Δ
```

Luego ejecute el programa nuevamente con  $n = 1$ .

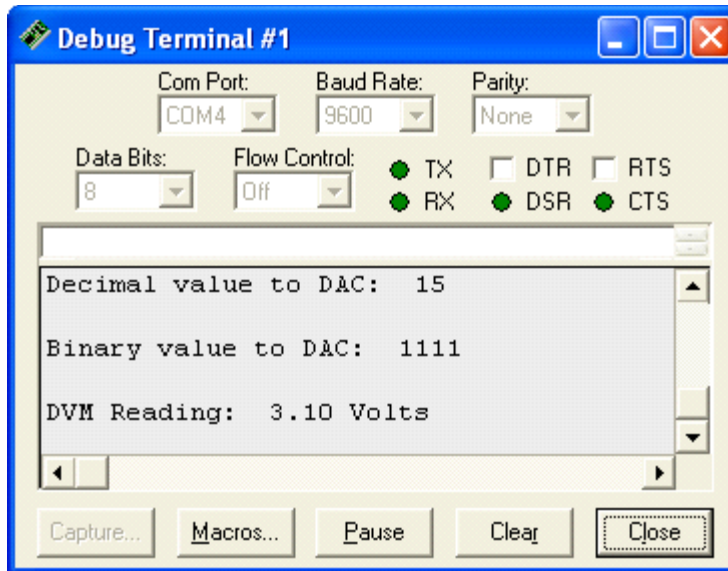
```
n = 1                                     ' Δ
```

Luego cambie  $n$  a 2 y ejecute el programa por tercera vez.

```
n = 2                                     ' Δ
```

Continúe para cada valor hasta  **$n=15$**

La Figura 4-4 muestra la medición de una conversión D/A para un valor de  $n$  igual a 15. Recuerde que el 10% de tolerancia de los resistores generan algún error en la salida. En el Capítulo 3, estábamos interesados en programar nuestro voltímetro con una precisión cercana a una centésima de Volt. En este experimento, cualquier dato dentro del 10% del valor esperado se considera como correcto. Así que la salida para  **$n=15$**  podría ser tan baja como 2,7 Volts o tan alta como 3,3 Volts. Si los errores son mayores, controle su circuito para asegurarse de no haber intercambiado las posiciones de los resistores de 1k $\Omega$  con los de 2k $\Omega$  en algún lugar de la red en escalera.



**Figura 4-4**  
Salida de Debug para  
el Programa 4.1,  
Revisión 1.

### **Direccionamiento**

Hasta ahora, hemos direccionado cada una de las líneas de E/S a la vez. Esto funciona bien cuando se quiere tener control sobre el estado de una línea en particular. Por ejemplo, un LED es fácilmente direccionado por un pin individual de E/S al que está conectado usando **OUTp=valor**, donde **p** es el número del pin entre 0 y 15, y **valor** es 0 ó 1.



**Direccionamiento:** Cuando un pin de E/S es direccionado, significa que se escribió un valor en un lugar específico de la RAM del BASIC Stamp. Por ejemplo, una dirección específica de memoria debe ser puesta en estado alto, para configurar a un pin de E/S como una salida. Otra dirección podría ser usada para modificar el estado del pin a alto o bajo.

Realizar estas operaciones un bit a la vez no es siempre eficiente. Las direcciones de memoria son adyacentes, así que se pueden realizar operaciones de 1 nibble (4-bits) a la vez, un byte (8-bits) a la vez, o incluso una word (16-bits) a la vez.

Dado que los pines de E/S P4 a P7 se usan como salidas en este experimento, sería más fácil y más eficiente tener un método para direccionar este grupo de pines. Observe que se necesitan cuatro líneas de código para configurar los pines como salidas y cuatro

instrucciones más para cargar el estado de cada bit. No es un problema por ahora, pero a medida que avance en la creación de programas más complejos, se encontrará buscando formas más eficientes para obtener el máximo rendimiento con el mínimo de código.

Hay dos registros que debemos configurar para controlar las salidas de un grupo específico de líneas de E/S. El primer registro se llama “dirección”. El comando “**OUTPUT**” configura a un pin como “output” (salida). Por el contrario, “**INPUT**” configura a una línea de E/S como entrada. El segundo es el registro “data”. Si el pin de E/S ha sido configurado como salida, entonces el registro “data” se puede poner en 0 ó 1. Al asignar un valor a este registro se pone el pin en estado bajo o alto y la tensión en la salida será de 0 ó 5 Volts.

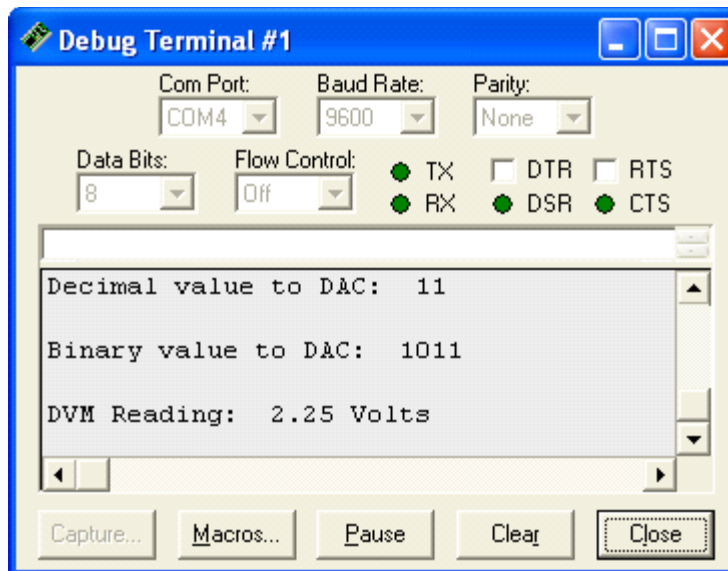
Ciertos comandos de PBASIC nos permiten direccionar directamente los pines de E/S como una word (16 bits individuales), dos bytes (dos juegos de 8 bits individuales) o como 4 nibbles (cuatro juegos de cuatro bits individuales). Para modificar nuestro código, queremos direccionar un nibble a la vez, para controlar los cuatro bits que están usando P4-P7. De acuerdo al BASIC Stamp Manual (del cual ya debería tener una copia ya que se puede bajar gratuitamente de [www.stampsinclass.com](http://www.stampsinclass.com)) el grupo “P4-P7” se llama nibble “b”. El siguiente grupo de cuatro (P8-P11) es nibble “c”, etc.

Intentemos usar esto en un programa y veamos como trabaja. Rescriba la subrutina DAC: en el Programa 4.1 como sigue:

```
DAC:
  n = 11

  DIRB = 15
  OUTB = n
  RETURN
```

La Figura 4-5 muestra que la salida es idéntica a la de la versión anterior de la subrutina **DAC**: Realizamos el mismo trabajo con dos líneas de código en lugar de ocho.



**Figura 4-5**  
Salida de Debug para  
el Programa 4.1,  
Revisión 2

Así es como se cuenta de 0 a 15 usando un nibble:

0 = 0000	4 = 0100	8 = 1000	12 = 1100
1 = 0001	5 = 0101	9 = 1001	13 = 1101
2 = 0010	6 = 0110	10 = 1010	14 = 1110
3 = 0011	7 = 0111	11 = 1011	15 = 1111

Cuando se selecciona el nibble b usando **DIRB**, cada bit del número **DIRB** configura, de acuerdo al valor correspondiente, la dirección de un pin de salida:

Bit in nibble B	3	2	1	0
I/O pin	P7	P6	P5	P4

Si usáramos el comando "**DIRB =4**" los bits del registro de dirección quedarían así:

Bit value	0	1	0	0
I/O pin	P7	P6	P5	P4

Esto haría que el pin de E/S P6 sea configurado como salida y los otros pines (P0,P1,P3) como entradas. Por lo tanto el comando **DIRB=15** (los cuatro bits son "1") configura a las cuatro líneas de E/S como salidas.

Esto haría que el pin de E/S P6 sea configurado como salida y los otros pines (P0,P1,P3) como entradas. Por lo tanto el comando **DIRB=15** (los cuatro bits son “1”) configura a las cuatro líneas de E/S como salidas.

Barriendo el valor de **n** de 0 a 15 debería resultar el mismo valor de antes.

Un aspecto realmente poderoso del uso de este método de direccionamiento, es que podemos usar PBASIC para contar hacia arriba o hacia abajo para direccionar automáticamente los pines de E/S. El resultado es que podemos programar al BASIC Stamp para que controle más eficientemente la salida del conversor D/A.

**4**

Modifique el código que aparece después de la etiqueta **start Display** en el Programa 4.1. Primero, modifique el comando **DEBUG CLS**, luego agregue la segunda línea como se muestra.

```
'Inicia ventana debug                                '□
DEBUG CLS, "Valores Nibble del DAC ", CR              '△
DEBUG "Decimal      Binario", CR                      '□
```

Modifique la rutina principal: como se muestra:

```
' ----[ Rutina Principal ]-----
FOR n = 0 TO 15
  GOSUB DAC                                '□
  GOSUB ADC_Data
  GOSUB Calc_Volts
  GOSUB Display
NEXT
STOP
```

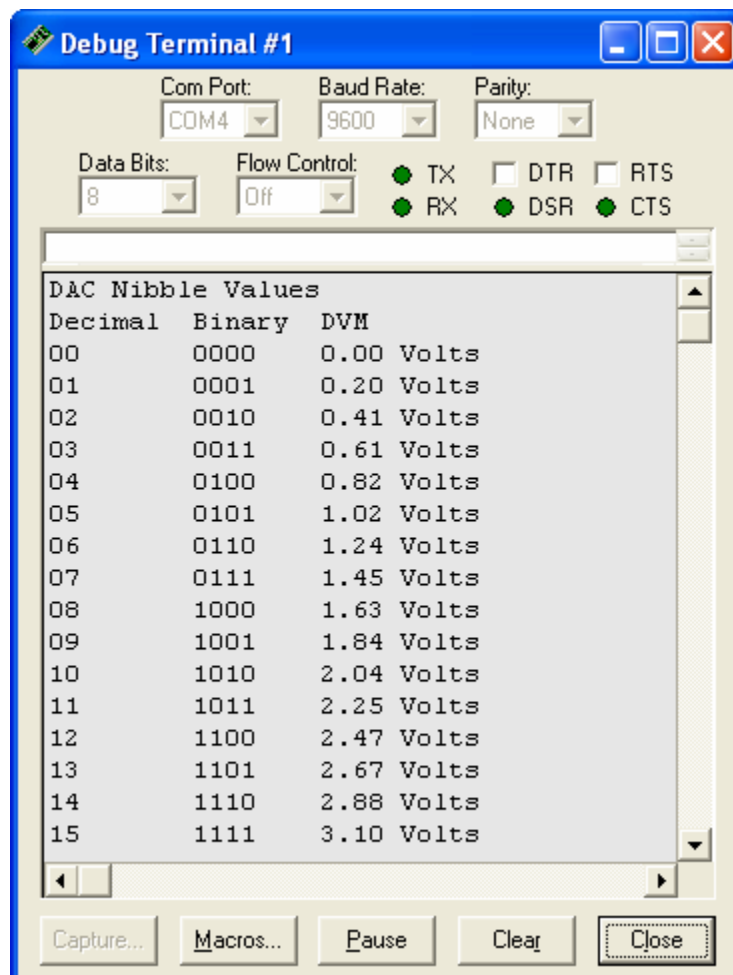
Borre la línea que asigna el valor de **n** en la subrutina **DAC**: Una vez modificada debería verse así:

```
DAC:
  DIRB = 15
  OUTB = n
RETURN
```

También modifique la subrutina **DISPLAY** como a continuación.

```
Display:
  DEBUG DEC2 n, " ", BIN4 n, " "
  DEBUG DEC1 v, ".", DEC2 v2, " Volts", CR
RETURN
```

La Figura 4-6 muestra la salida. Verifique los valores con un voltímetro de mano y notará la utilidad de combinar el BASIC Stamp con interfaces analógicas. Imagine intentar verificar los 4096 niveles de un convertor DAC de 12-bis, uno a la vez.



**Figura 4-6**  
Salida de Debug para  
el Programa 4.1,  
Revisión 3.

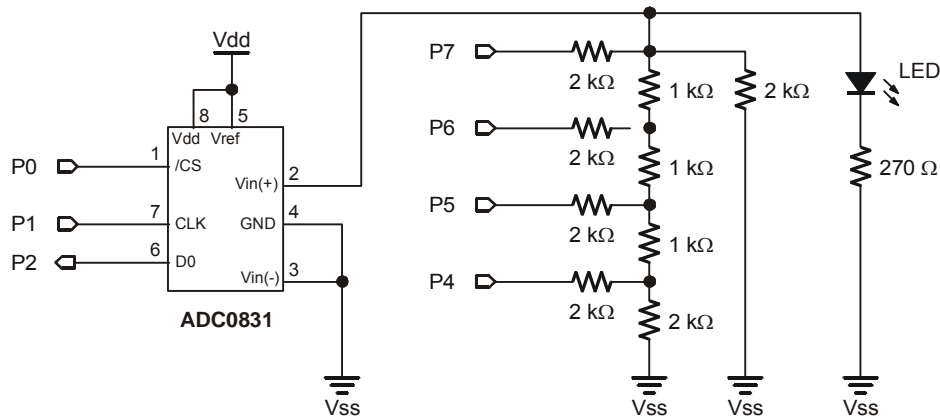
Esta es una forma muy eficiente de obtener datos de un barrido de tensión. Observando los datos obtenidos en el barrido de tensión, notamos varias cosas interesantes. Primero, la tensión de salida del convertor D/A siempre es ligeramente alta. Segundo, el error

aumenta a mayor tensión de salida. Tercero, el error máximo es 0,1 Volts. Este tipo de datos puede ser extremadamente útil en el diseño electrónico y en procesos de prueba automatizados nos permite ahorrar mucho tiempo.

### **El Seguidor de Tensión**

Usemos el barrido de tensión para analizar lo que sucede cuando la salida del conversor D/A es conectado a otro circuito. Usaremos la salida del conversor D/A para alimentar un LED. Primero conectaremos directamente la salida del conversor D/A a la entrada del circuito del LED. Luego usaremos el seguidor de tensión como paso intermedio entre la salida del conversor D/A y la entrada del circuito del LED.

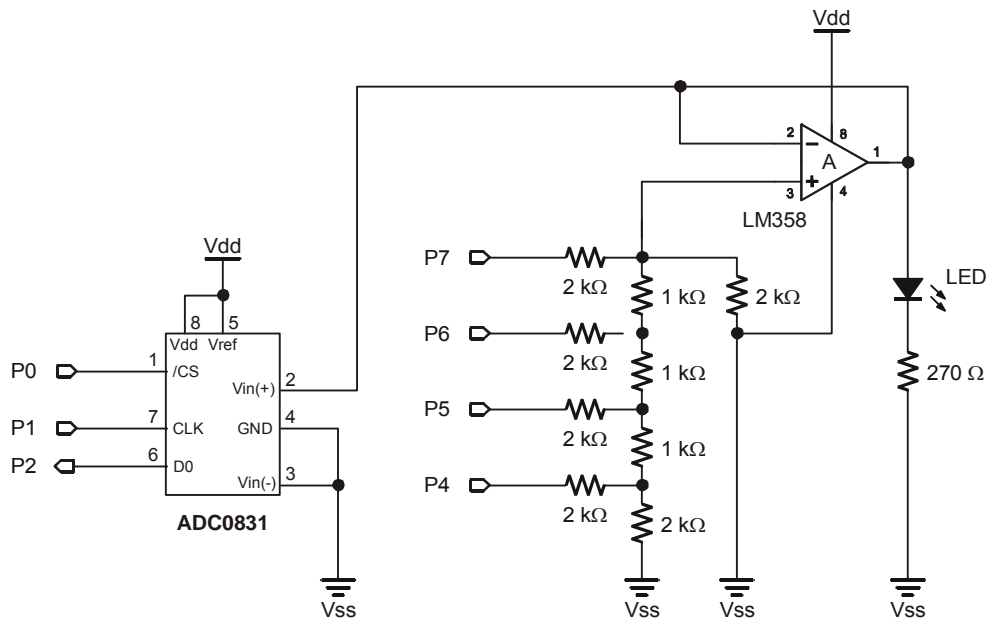
La Figura 4-6 muestra el conversor D/A con el circuito del LED agregado. El LED es la "carga" que el conversor D/A debe "alimentar". Ejecute un barrido de tensión sobre este circuito y complete la tabla que aparece a continuación. Luego realice el barrido de tensión usando la salida del seguidor de tensión como se muestra en la Figura 4-7. Llene la misma tabla con la información del barrido de tensión del segundo circuito y compare los datos. El seguidor de tensión de la Figura 4-7 se suele llamar "buffer".



**Figura 4-6** Circuito D/A sin Buffer

Table 4-1: Salida del Conversor D/A sin Buffer		
Decimal	Binario	Tensión (Volts)
00	0000	_____
01	0001	_____
02	0010	_____
03	0011	_____
04	0100	_____
05	0101	_____
06	0110	_____
07	0111	_____
08	1000	_____
09	1001	_____
10	1010	_____
11	1011	_____
12	1100	_____
13	1101	_____
14	1110	_____
15	1111	_____



**Figura 4-7** Conversor D/A con Buffer

Comparando las dos tablas, debería quedar bastante claro que el buffer (seguidor de tensión) elimina el problema causado al conectar el circuito del LED directamente a la salida del conversor D/A. La tensión de salida del conversor D/A sin un buffer alcanzó un valor máximo bien por debajo de los 3 Volts que es capaz de entregar el conversor D/A. Por el contrario, la salida con buffer llegó hasta los 3 Volts sin problemas.

Table 4-2: Salida del Conversor D/A con Buffer		
Decimal	Binario	Tensión (Volts)
00	0000	_____
01	0001	_____
02	0010	_____
03	0011	_____
04	0100	_____
05	0101	_____
06	0110	_____
07	0111	_____
08	1000	_____
09	1001	_____
10	1010	_____
11	1011	_____
12	1100	_____
13	1101	_____
14	1110	_____
15	1111	_____

El origen de este problema nos lleva a recordar la Ley de Ohm:  $V = I \times R$  (tensión es igual a corriente por resistencia). Cada pin de E/S del BASIC Stamp puede suministrar hasta 20 mA de corriente. En el caso de la red resistiva en escalera sin el buffer, los pines de E/S del BASIC Stamp alcanzaron su máxima corriente de salida. Mientras tanto, la resistencia vista desde los pines de E/S permaneció en el mismo valor. En otras palabras,  $I \times R$  alcanzó un valor límite debido a que  $I$  (la corriente) no pudo aumentar más y  $R$  es un valor fijo. Así que la tensión de salida es igual a un valor de corriente que no puede aumentar lo suficiente por un valor fijo de resistencia. Esta es la razón por la que la tensión de salida deja de incrementarse.

Además de aislar un circuito del otro, un op-amp en la configuración de seguidor de tensión normalmente puede suministrar mucha más corriente que el circuito que está conectado a su entrada. El nombre buffer es comúnmente usado cuando un seguidor de tensión se usa para suministrar corriente extra.

El circuito de la Figura 4-7 se usará nuevamente al comienzo del Capítulo 5, así que no lo desarme al finalizar el Capítulo 4.

### ¿Qué aprendí?

En las oraciones de abajo, inserte las palabras apropiadas de la lista de la izquierda.

binarios	Un juego de valores de tensión _____ enviados a las entradas de una red resistiva en escalera dan como resultado niveles de tensión _____ en la salida. Este método es una alternativa económica a los circuitos integrados. Sus ventajas son flexibilidad en la resolución y bajo costo. Su principal desventaja es la precisión.
discretos	
bit	
modificar	La extensión .bit puede agregarse a una variable para seleccionar un _____ en particular dentro de un nibble, byte o word. Esto puede ser usado para seleccionar bits de un número binario de la memoria o _____ el valor de estos bits.
bit	
corriente	El PBASIC puede ser usado para direccionar los pines de E/S del BASIC Stamp un _____ a la vez. Existen alternativas para _____ grupos de pines o variables tipo nibble, byte o word. Esto permite modificar los valores de salida y la _____ de los pines de E/S, como grupos de bits en lugar de hacerlo de a uno.
direccionar	
dirección	
barrido	Un _____ de tensión puede ejecutarse en un conversor D/A. Esto permite ver la salida del conversor D/A para todas las entradas binarias posibles. El BASIC Stamp puede ser programado para _____ este proceso y mostrar los datos en formato de tabla.
automatizar	
	El seguidor de tensión puede ser usado como buffer, que suministra _____ extra al circuito conectado a la salida del conversor D/A.

**Preguntas**

1. ¿Cuál es el valor de “1101” en el sistema numérico decimal? ¿Qué tensión debería esperarse en la salida del conversor D/A si se envía este número binario?
2. ¿Qué función realiza un conversor D/A?
3. ¿Cuáles son algunas de las ventajas y desventajas de la red resistiva en escalera?
4. ¿Por qué la tensión del conversor D/A “salta” de un valor a otro, en lugar de realizar un barrido continuo como el potenciómetro?
5. ¿Cómo el seguidor de tensión soluciona el problema del rango de voltaje de la salida cuando la salida del convertidor de D/A está conectado a la entrada del circuito del LED?

**Desafío**

1. Cree un conversor D/A de 8 bits en configuración de “escalera resistiva”. Dibuje el esquema eléctrico completo.
2. Escriba un programa que genere 256 pasos por diferentes tensiones. Cada tensión debería estar presente durante 100 milisegundos en el voltímetro.

### **¿Por qué aprendí esto?**

Hay muchos circuitos diferentes del “mundo real” que necesitan algún tipo de tensión analógica. Por ejemplo, cuando reproduce un disco compacto, está escuchando una señal analógica (del micrófono) que ha sido digitalizada (A/D) y almacenada en un CD. Estos datos digitales luego son enviados a un amplificador y a los parlantes (después del proceso de conversión D/A).

Cuando diseña productos comerciales, es su responsabilidad determinar el método más apropiado (y económico) para realizar una tarea dada. Un conversor D/A resistivo es un método muy económico para obtener una tensión analógica de un dispositivo digital.

### **¿Cómo puedo aplicarlo?**

Acaba de construir una fuente de tensión variable y un voltímetro digital en la misma protoboard. También usó algunas técnicas de programación de PBASIC para obtener información del conversor mediante un barrido de tensión. Una aplicación de esto es la prueba de circuitos, aunque hay muchas más. En el Capítulo 5, veremos cómo puede usarse el PBASIC para controlar el volumen de los tonos emitidos por un parlante. En el Capítulo 7, usaremos la conversión D/A para controlar el brillo de un LED que transmita una señal a un foto resistor.

Las aplicaciones para las interfaces D/A y A/D combinadas con un microcontrolador solamente están limitadas por su imaginación. Estas técnicas pueden ser aplicadas a automatización de casas, sistemas de riego, sistemas de guía de misiles y robótica, para mencionar algunas. La ingeniería de sistemas de control es un campo de estudios dentro de la ingeniería eléctrica que podría resultarle útil si planea diseñar sistemas.

## Capítulo 5: Señales que Varían en el Tiempo

---

En este Capítulo veremos señales con el Stamp-O-Scope. Para hacer esto usaremos el BASIC Stamp junto con los circuitos A/D y D/A contruidos en los Capítulos 3 y 4 para emular la función de un **osciloscopio**.

La tensión de salida de la red resistiva en escalera variará en el tiempo y los cambios se registrarán con nuestro emulador de osciloscopio, el Stamp-O-Scope. El Stamp-O-Scope obtendrá los datos de la tensión de entrada desde el ADC0831 y los mostrará gráficamente en la ventana debug.

**5**

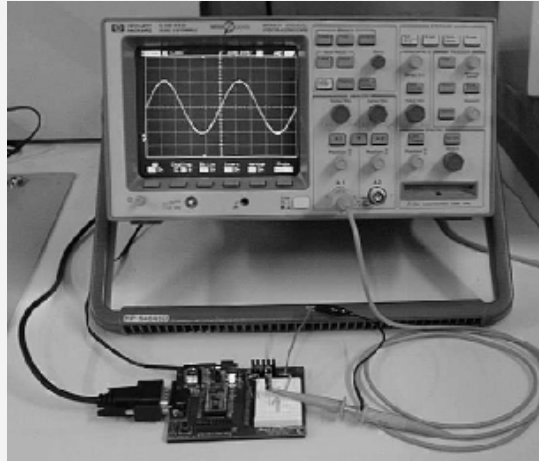
Veremos dos señales diferentes con el Stamp-O-Scope y usaremos el conversor D/A para ajustar los atributos de ambas señales. La primera señal con la que trabajaremos se llama onda triangular y la segunda onda cuadrada. El LED se usará para monitorear la actividad de ambas señales.

A diferencia de un osciloscopio normal, el Stamp-O-Scope no puede mostrar señales de tensión a frecuencias del rango audible. Sin embargo, el BASIC Stamp puede fácilmente generar una señal de tensión dentro del rango audible, mediante el conversor D/A. Esta señal puede ser usada para que un parlante emita sonidos de tono y volumen variable.

El tono y el volumen se ajustarán modificando dos de las propiedades de la onda cuadrada. Pero primero, comencemos a apreciar estas propiedades con el Stamp-O-Scope.

También en este Capítulo, el BASIC Stamp es usado para generar sinusoides (ver la Figura 5-1) de frecuencia variable. Estas sinusoides se usan para emitir notas musicales.

**Osciloscopio:** Es un dispositivo que mide y muestra señales que varían en el tiempo. Es una herramienta común, usada por muchos técnicos e ingenieros en Electrónica para observar estas señales. Las señales de interés a menudo se repiten a sí mismas varias veces por segundo. El osciloscopio se puede usar para determinar la forma general de la señal, qué tan rápido se repite y los valores de tensión máximo y mínimo.



**Figura 5-1**  
Osciloscopio  
mostrando una  
sinusoide  
generada por el  
Programa 5.3.

**Frecuencia:** Velocidad a la que la señal se repite a sí misma. La frecuencia se mide en repeticiones por segundo. El nombre Hertz (Hz) se usa como unidad de medición de frecuencia. Un Hertz es una repetición por segundo:

$$1 \text{ Hz} = 1 \text{ repetition/second} = 1/\text{s}$$

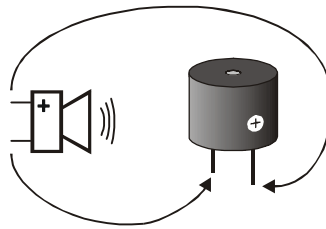
### **Componentes Requeridos**

Para comenzar con este experimento se usa el mismo circuito del Capítulo 4, Figura 4-7. Luego se reemplazará el circuito del LED con un parlante piezoeléctrico. Para este experimento necesitará los siguientes componentes:

- (6) Resistores de 2 k Ohm
- (3) Resistores de 1 k Ohm
- (1) Conversor ADC0831 A/D
- (10) Cables de interconexión
- (1) Parlante piezoeléctrico



El parlante piezoeléctrico que se muestra en la Figura 5-2 tiene terminales positivo y negativo que se aprecian en el componente y en su símbolo esquemático. El parlante del kit tiene un signo positivo (+) en la cara superior sobre el pin apropiado.



**Figura 5-2**  
Parlante  
piezoeléctrico.

*Componente y  
símbolo  
esquemático.*

**5**

Si conservó el circuito desarrollado en los Capítulos 3 y 4, está listo para el siguiente paso. Caso contrario, arme nuevamente el circuito mostrado en el Capítulo 4, Figura 4-7.

### Prográmelo

Comience con la versión sin modificar del Programa 4.1, realizando los cambios que se muestran a continuación. Las líneas modificadas se marcaron con un  $\Delta$  y las agregadas con un  $\square$ .

Modifique la primer línea así:

```
' Analógico y Digital Básicos - PL5_1R0.bs2          '\Delta
```

Inserte una rutina llamada **loop:** entre el comando **DEBUG cls** y la etiqueta **main:** como se muestra a continuación:

```
'Inicia ventana debug
DEBUG CLS

DO                                     '\square
  FOR n = 7 TO 15                      '\square
    GOSUB Main                         '\square
  NEXT                                '\square
  FOR n = 14 TO 8                      '\square
    GOSUB Main                         '\square
  NEXT                                '\square
LOOP                                  '\square
```

Ahora modifique la rutina principal:. Hay dos cambios a realizar. Primero, el comando **GOSUB CALC\_VOLTS** debería convertirse en comentario agregándole un apóstrofe y segundo, la instrucción **goto principal** debería cambiarse por un comando **RETURN**.

```
Main:                                     'Δ
    GOSUB DAC
    GOSUB ADC_Data
    'GOSUB Calc_Volts                     'Δ Comente esta línea
    GOSUB Display
    RETURN                               'Δ
```

Modifique la subrutina **DAC**: para que se vea así:

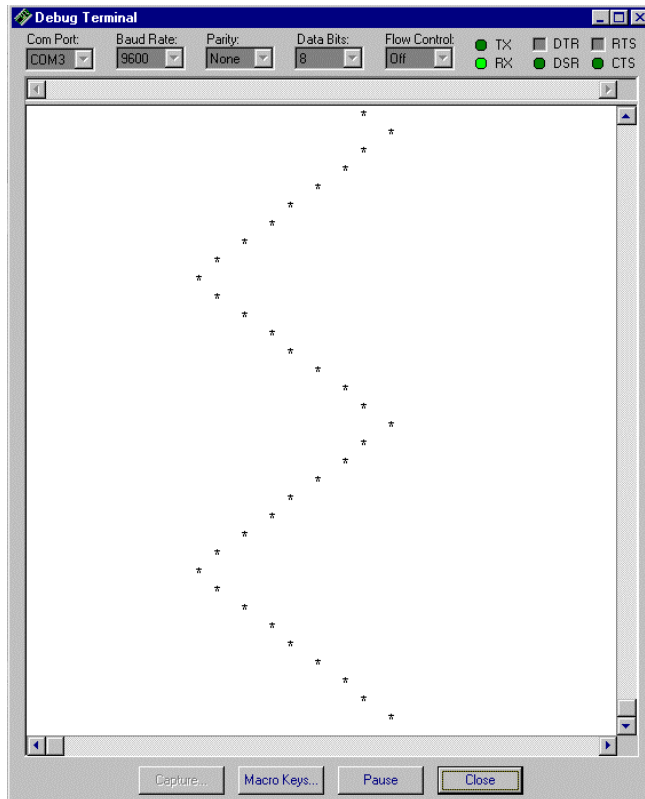
```
DAC:
    DIRB = 15
    OUTB = n
    PAUSE 50
    RETURN
```

También modifique la subrutina **DISPLAY**: del Programa 4.1 de la siguiente forma:

```
Display:
    v3 = adcBits / 8
    DEBUG REP " "\v3, "*", CR
    RETURN
```

## La Salida – Una Onda Triangular

Guarde el programa como P5\_1R0.bs2 y descárguelo en el BASIC Stamp. La salida del Stamp-O-Scope debería verse como la Figure 5-3. El LED rojo sobre la protoboard debería aumentar y disminuir el brillo gradualmente, de la misma forma en que varía la onda en la ventana debug. El asterisco de la última línea de la ventana debug indica la tensión medida actual. Cuanto más a la derecha va el asterisco, mayor es la tensión medida.



**Figure 5-3**  
Muestra de Salida

*del Stamp-O-Scope del  
Programa 5.1.  
La forma de onda de la salida  
normalmente se denomina  
“onda triangular”*

5

Si girara la pantalla del Stamp-O-Scope noventa grados en sentido antihorario, la imagen sería más parecida a la generada por un osciloscopio. La Figure 5-4 muestra el Stamp-O-Scope rotado de esta forma. También muestra tres de las mediciones más comunes que se efectúan en un osciloscopio: amplitud, frecuencia y desplazamiento de CC (DC offset). Un osciloscopio normalmente muestra el tiempo en el eje horizontal y la tensión en el vertical. El tiempo transcurre de izquierda a derecha y a medida que la señal se dirige hacia arriba, mayor es la tensión. La mayoría de los gráficos de formas de onda que varían en el tiempo también se presentan en este formato.

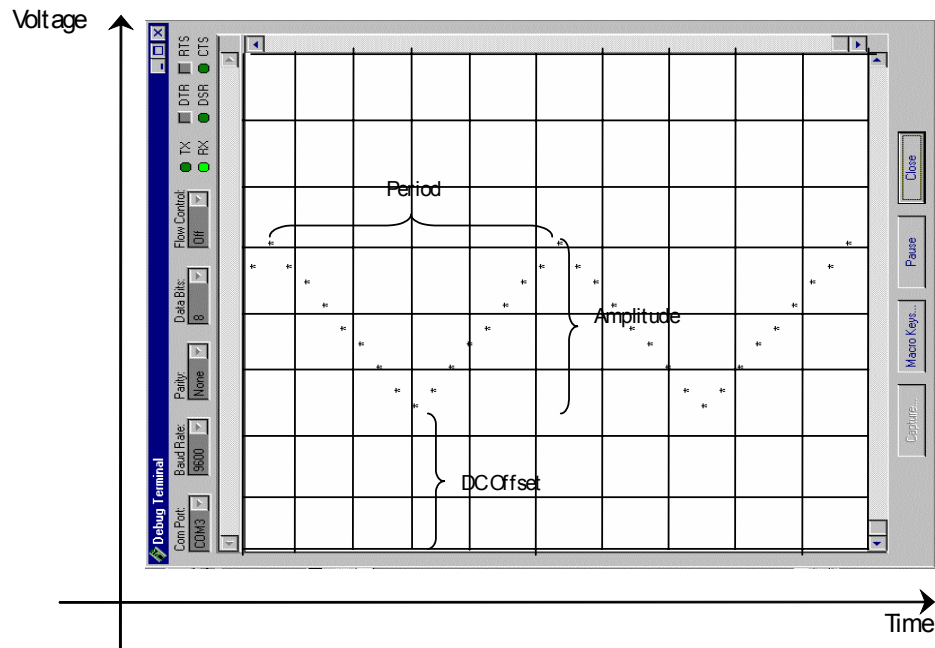


Figure 5-4: Salida Rotada 90°

Esta es una forma más convencional de ver señales. Normalmente se ven de esta forma en un osciloscopio. La amplitud, el período y el desplazamiento de CC (DC offset) son tres mediciones que normalmente se efectúan con un osciloscopio.

Note que el tiempo es el eje horizontal en esta imagen y la tensión es el vertical.

El período de la forma de onda es la cantidad de tiempo entre repeticiones. En este caso, el período es de aproximadamente 1 segundo. La frecuencia es la inversa del período, así que podemos usar una ecuación simple para determinar la frecuencia cuando conocemos el período:

$$f = 1/T$$

El término  $f$  es la frecuencia, medida en Hertz (Hz) y  $T$  es el período, medido en segundos (s). Dado que el período es un segundo y  $1/1 = 1$ , la frecuencia es de

aproximadamente 1 Hz. Veamos otro ejemplo, si el período es de una centésima de segundo ( $1/100$ ), la frecuencia es de  $1/(1/100) = 100$  Hz.

Amplitud y desplazamiento de CC (DC offset) son dos cantidades relacionadas a la tensión de la señal. La amplitud mostrada en la Figure 5-4 es llamada amplitud pico a pico. La salida del conversor D/A es programada para un máximo de 3 Volts y un mínimo de 1,6 Volts. Por lo tanto la amplitud pico a pico es de  $3 - 1,6$  Volts = 1,4 Volts.

El desplazamiento de CC es la diferencia entre 0 Volts y el valor mínimo de la forma de onda. Ya sabemos que el punto más bajo de la forma de onda es 1,6 Volts, que es el valor de DC offset.

**5**

Un punto alternativo para medir la amplitud y el DC offset es el punto medio entre máximo y mínimo de la señal. En este caso el punto estaría a  $(3 + 1,6) / 2$  Volts = 2,3 Volts. La amplitud sería de  $3 - 2,3$  Volts = 0,7 Volts. El DC offset sería de  $1,6 + 0,7$  Volts = 2,3 Volts.

### Explicación del Programa

La rutina **Main** ha sido relegada a subrutina a favor de la rutina **DO-LOOP**. Esta rutina aumenta y disminuye el valor de **n**, que se envía al conversor D/A. Es por este motivo que la salida del conversor D/A aumenta y disminuye en la pantalla del Stamp-O-Scope.

En cada pasada por el **FOR-NEXT** de la rutina **DO-LOOP** se incrementa (o decrementa) el valor de **n**, llamando luego a la subrutina **Main**.

```
DO
  FOR n = 7 TO 15
    GOSUB Main
  NEXT
  FOR n = 14 TO 8
    GOSUB Main
  NEXT
LOOP
```

La subrutina **Main** aún realiza la misma tarea que en el experimento anterior. La única diferencia es que ahora se desempeña como subrutina. La subrutina **CALC\_VOLTS**: no es necesaria para nuestro Stamp-O-Scope, así que el comando **GOSUB CALC\_VOLTS**: se marcó como comentario. Para ahorrar memoria EEPROM, podría borrar completamente esta subrutina. Tenga en cuenta que si usa la técnica anterior en aplicaciones más importantes, puede quedarse sin memoria EEPROM para almacenar el programa.

```
Main:
  GOSUB DAC
  GOSUB ADC_Data
  'GOSUB Calc_Volts
  GOSUB Display
RETURN
```

Esta es la subrutina **DAC** más simple que se usó en el Capítulo 4.

```
DAC:
  DIRB = 15
  OUTB = n
  PAUSE 50
RETURN
```

El comando **DEBUG** de la subrutina **Display** es el que coloca los asteriscos en la ventana debug en su lugar, de acuerdo a la tensión medida en ese instante. A mayor tensión medida, más a la derecha se imprime el asterisco.

El modificador de debug **REP** es el que hace que el caracter " " (espacio en blanco) se imprima una y otra vez. En este caso se imprime **adcBits/4** (el valor de **adcBits** dividido por cuatro) veces. Después de todos esos espacios, se imprimen un asterisco y un salto de línea, dejando la ventana preparada para el asterisco de la próxima medición.

```
Display:
  DEBUG REP " "\adcBits/4,"*", CR
RETURN
```

¿Cuántos espacios se imprimen delante del asterisco? Supongamos que estamos midiendo 3 Volts, el nivel de tensión más alto que puede generar el conversor DAC. Recuerde que **adcBits** es el número enviado por el ADC0831, dentro del rango de 0 a 255 en una escala de 5 Volts. Por lo tanto, cuando se miden 3 Volts:

$$\text{adcBits} = (3/5) \times 255 = 153$$

La cantidad de espacios es:

$\text{adcBits}/4$ , que es  $153/4 = 38.25 = 38$  redondeado a números enteros.

La tensión máxima pondrá un asterisco a 39 espacios del margen izquierdo de la ventana debug.

Pruebe modificaciones en los valores de **n** de los bucles **FOR-NEXT** en la rutina **DO-LOOP**. En este caso, cambiarán el período y la amplitud de la onda triangular. Abajo hay un

ejemplo donde los límites del bucle **FOR-NEXT** se fijaron para la excursión máxima de tensión del conversor D/A de red resistiva en escalera.

```
DO
  FOR n = 1 TO 15
    GOSUB Main
  NEXT
  FOR n = 14 TO 0
    GOSUB Main
  NEXT
LOOP
```

Note que aumentan la amplitud y el período de la forma de onda. Esto es debido a que el BASIC Stamp está programado para realizar una pausa por una cantidad fija de tiempo a cada nivel de tensión de salida. La instrucción pause es parte de la rutina DAC.

## La Onda Cuadrada

La siguiente forma de onda que examinaremos se denomina onda cuadrada. En este caso, la amplitud es la tensión entre los niveles de tensión alto y bajo y el DC offset es la tensión entre 0 Volts y la parte inferior de la señal. El período y la frecuencia son los mismos que en el caso de la onda triangular. De esta forma el período sigue siendo la cantidad de tiempo que tarda en repetirse la forma de onda y la frecuencia la cantidad de veces que se repite en un segundo. La frecuencia continúa siendo la inversa del período.

La rutina de bucles puede ser modificada para generar una onda cuadrada que tenga la misma frecuencia de la onda triangular original. La variable v3 no es usada hasta el momento debido a que saltamos la subrutina `Calc_Volts`. Ésta puede ser usada como contador. El valor de `n`, que controla la salida del conversor D/A puede fijarse a un valor alto y uno bajo. Comencemos con un valor alto de 3 Volts y uno bajo de 0 Volts. Modifique la rutina de bucles y guarde el programa modificado como P5\_1R1.bs2

```
DO
  FOR v2 = 0 TO 15
    n=15
    GOSUB Main
  NEXT
  FOR v2 = 15 TO 0
    n=0
    GOSUB Main
  NEXT
LOOP
```

Ejecute el programa y observe la onda cuadrada. ¿Es realmente cuadrada? Puede ajustar el valor de **n** en el primer bucle **FOR-NEXT** para cambiar la amplitud. Puede ajustar los valores de **n** del segundo bucle **FOR-NEXT** para obtener distintos DC offsets. Solamente recuerde que **n** debe estar entre 0 y 15.

Los límites del primer bucle **FOR-NEXT** modifican la cantidad de tiempo que la señal permanece en estado alto. En otras palabras, puede cambiar el ancho de pulso modificando **v3** en el primer bucle **FOR-NEXT**.

Los límites del segundo bucle **FOR-NEXT** ajustan la duración de la parte baja de la señal.

Modificando las duraciones de las partes baja y alta de la señal en forma proporcionada, se modifica la frecuencia sin afectar el **ciclo de trabajo (duty cycle)**. Si modifica una sola de las duraciones, se modifica el ciclo de trabajo así como también la frecuencia de la señal.

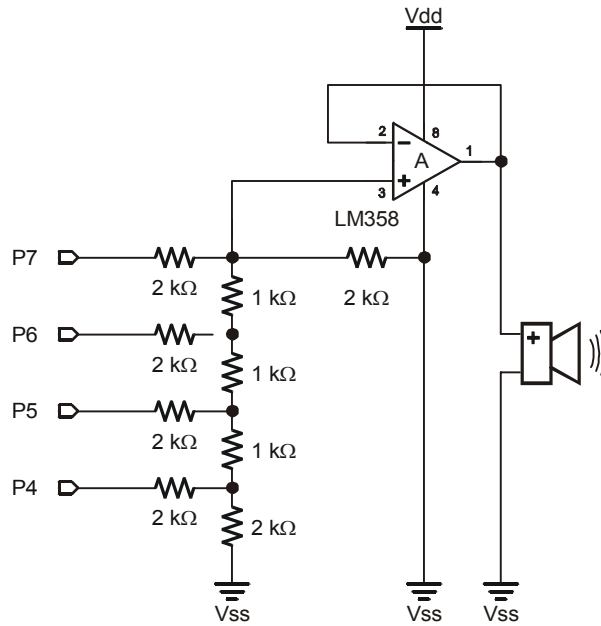


**Ancho de pulso:** Es la cantidad de tiempo que una señal binaria permanece en estado alto (desde que sube hasta que baja).

**Ciclo de Trabajo (Duty cycle):** Es la relación entre el tiempo que la señal permanece en estado alto y el tiempo que le toma repetirse. En otras palabras, es el ancho de pulso dividido por el período de la señal.

Elevemos ahora la frecuencia de la onda cuadrada hasta el rango de frecuencias audibles y escuchemos el resultado de modificar estas características de la señal. Quite el ADC0831 de la Plaqueta de Educación. Además, reemplace el circuito del LED por el del piezoeléctrico que se muestra en la Figure 5-5.





**Figure 5-5**  
Circuito del Parlante  
Piezoeléctrico

*conectado a la salida con búfer  
del conversor D/A.*

5

Ingresa este programa en el editor del BASIC Stamp. Esta no es una modificación a los programas previos. Igualmente, la salida es el parlante piezoeléctrico y esa no es la ventana de salida de Debug. Guarde el programa como PL5\_2RO.bs2.

```
' Analógico y Digital Básicos - PL5 2R0.bs2
' Conversor D/A generando sonido, n cambia el volumen, m cambia la frecuencia
' {$STAMP BS2}
' {$PBASIC 2.5}

n      VAR   Nib
m      VAR   Word

DIRB=15
n = 15
m = 500

DAC:
  OUTB = n
  PAUSE m
  OUTB = 0
  PAUSE m
GOTO DAC
```

Primero ejecute el programa como se muestra. El parlante piezoeléctrico debería emitir un tic cada aproximadamente medio segundo, indicando la transición entre el estado bajo y alto de la señal.

Luego, cambie el valor de  $m$  de 500 a 100 y observe que el tic se produce mucho más rápido. También pruebe con los valores 50, 20, 10, 5 y 1. Experimente un poco.

El valor  $m = 1$  debería hacer que el parlante piezoeléctrico emita un tono bastante claro.

Luego, cambie la amplitud modificando el valor de  $n$ . Primero pruebe con  $n = 1$ . El tono debería ser el mismo, solo que de menor volumen. Luego pruebe con  $n = 5$ . El sonido sigue siendo del mismo tono, pero con mayor volumen. Para valores de  $n = 10$  y  $15$ , el volumen debería incrementarse bastante.

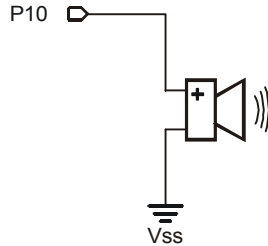
En esta aplicación, hicimos que el parlante piezoeléctrico emitiera sonidos, aplicándole pulsos con el BASIC Stamp. Incrementamos gradualmente la frecuencia al disminuir la cantidad de tiempo de las pausas en estado alto y bajo, disminuyendo efectivamente el período de la señal. Dado que la frecuencia es la inversa del período, la disminución en el período aumentó la frecuencia. Al aumentar la frecuencia, el tono del sonido también aumentó. Luego cambiamos la amplitud de la onda cuadrada modificando el valor de salida para la señal en estado alto. Esto modificó el volumen del sonido emitido por el parlante piezoeléctrico.

Las variaciones en la presión del aire son las que causan los sonidos. Cualquier sonido puede ser representado por algún tipo de forma de onda que varía en el tiempo. Para lograr que un parlante piezoeléctrico emita sonido, enviamos una señal de tensión que varía en el tiempo a su entrada. El parlante convirtió la señal en movimiento de una membrana en el interior del encapsulado plástico. A medida que la membrana vibra, causa variaciones en la presión del aire. Nuestro tímpano (otra membrana) sensa estas variaciones de presión, lo que nos permite oír el sonido.

La onda cuadrada que enviamos al parlante piezoeléctrico generó las variaciones en la presión del aire. Variamos la amplitud de la señal para aumentar la amplitud de las variaciones en la presión del aire, lo que incrementó el volumen del sonido. La frecuencia de la onda cuadrada se modificó para alterar la frecuencia de las variaciones de presión, lo que afectó el tono del sonido.

### La Sinusoide y la Modulación de Ancho de Pulso (PWM)

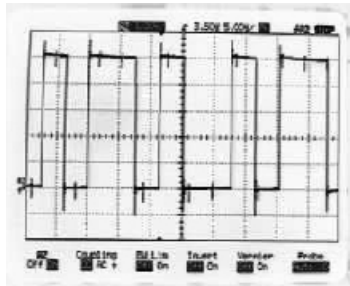
El BASIC Stamp tiene una función interna que genera sonidos en forma similar a la que utilizamos en la sección anterior. Desconecte el terminal del resistor de 100  $\Omega$  conectado a la salida del búfer (pin 1 del LM358) y conéctelo a P10 en la Plaqueta de Educación como se muestra en la Figura 5-6. Estos tres componentes es todo lo que se necesita conectar BASIC Stamp para que pueda emitir sonidos.



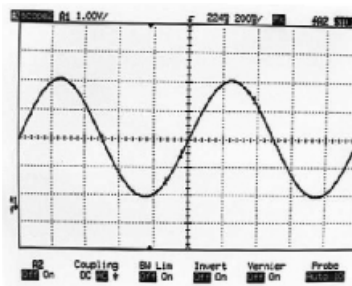
**Figura 5-6**  
Circuito de Sonido  
para el BASIC Stamp

Sin el parlante piezoeléctrico, el capacitor y el resistor mostrados en la Figura 5-6 funcionan como una batería recargable, que se carga y descarga en períodos de tiempo muy cortos. Aplicando muchos pulsos al circuito, la salida de tensión puede tomar la forma de una señal.

Estos pulsos pueden tener distintos anchos y ser aplicados más o menos frecuentemente. Dependiendo de estos parámetros, el capacitor gana o pierde tensión. Esta es la base de funcionamiento de la modulación por ancho de pulso (PWM), que puede ser usada para generar señales de tensión. La Figura 5-7 muestra un ejemplo de señal PWM aplicada a la entrada de un circuito RC, así como también la salida sinusoidal.



Señal PWM I



Sinusoide

**Figura 5-7**  
Imágenes de  
Osciloscopio

*De un PWM y  
la sinusoide  
resultante.*

Las sinusoides pueden usarse para representar notas musicales. Cuando se toca una senoide en un parlante piezoeléctrico, el sonido es de mejor calidad. El Programa 5.3 define una octava completa de notas musicales con sus frecuencias correspondientes (en el sistema americano). Una lista de estas notas se puede tocar empleando comandos **FREQOUT**. Ahora programémoslo.

### Programa de Notas Musicales

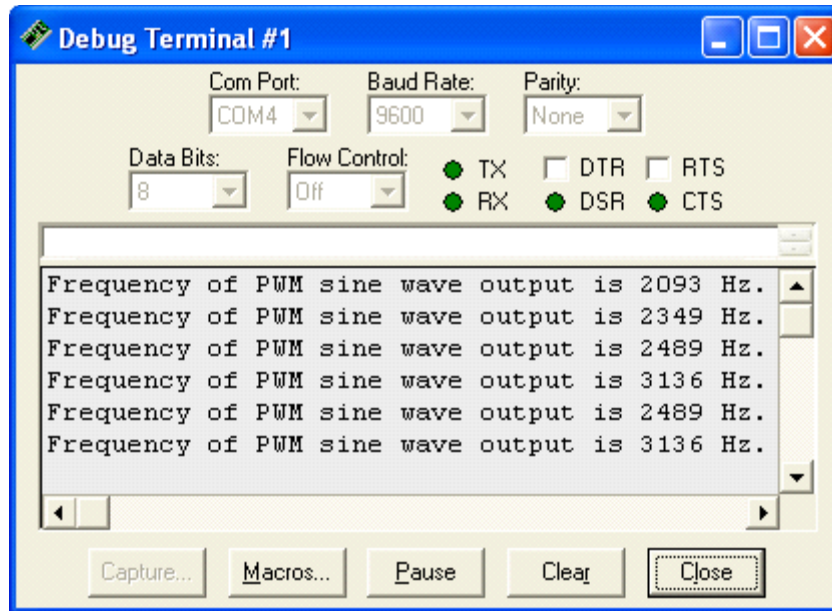
```
' Analógico y Digital Básicos - PL5_3R0.bs2
' Generando sonido
' {$STAMP BS2}
' {$PBASIC 2.5}

C          CON      2093
CSharp     CON      2218
D          CON      2349
DSharp     CON      2489
E          CON      2637
F          CON      2794
FSharp     CON      2960
G          CON      3136
GSharp     CON      3322
A          CON      3520
ASharp     CON      3729
B          CON      3951

DEBUG CLS
FREQOUT 10, 100, C
DEBUG HOME, "Frecuencia de senoide de salida:", DEC4 C, " Hz.", CR
FREQOUT 10, 100, D
DEBUG " Frecuencia de senoide de salida ", DEC4 D, " Hz.", CR
FREQOUT 10, 100, DSharp
DEBUG " Frecuencia de senoide de salida ", DEC4 DSharp, " Hz.", CR
FREQOUT 10, 200, G
DEBUG " Frecuencia de senoide de salida ", DEC4 G, " Hz.", CR
FREQOUT 10, 100, DSharp
DEBUG " Frecuencia de senoide de salida ", DEC4 DSharp, " Hz.", CR
FREQOUT 10, 200, G
DEBUG " Frecuencia de senoide de salida ", DEC4 G, " Hz.", CR
```

### La Salida

La Figura 5-8 muestra la salida en la ventana debug. Los sonidos son de mejor calidad que los generados por la onda cuadrada.



**Figura 5-8**  
Salida de  
Debug  
mostrando  
las  
frecuencias  
emitidas por  
el Programa  
5.3.

5

### Explicación del Programa

El único comando introducido en esta sección es **FREQOUT**. Éste se usa así:

**FREQOUT *pin, duration, frequency\_1, frequency\_2***

Ya sabemos que **pin** es un número entre 1 y 15 que indica el pin de E/S del BASIC Stamp que usaremos. La **duración** es un número entre 1 y 65535 que especifica la cantidad de tiempo que se emitirá el sonido, en milisegundos (ms). El término **frecuencia 1** especifica la frecuencia del sonido que se generará. Una segunda frecuencia (**frecuencia 2**) se puede tocar simultáneamente para lograr efectos muy interesantes. Por ejemplo, los sonidos que escucha cuando marca un número telefónico, en realidad están compuestos por dos tonos superpuestos simultáneamente. El BASIC Stamp también tiene un comando para tonos telefónicos llamado **DTMF**, que usa el mismo principio.

### ¿Qué aprendí?

En las oraciones de abajo, inserte las palabras apropiadas de la lista de la izquierda.

tensión	Un osciloscopio es una herramienta usada por técnicos e ingenieros que mide y muestra _____. Amplitud, DC offset, desplazamiento de fase, período y _____ son características útiles de una señal, que pueden medirse con un osciloscopio.
período	
amplitud	
tiempo	Un osciloscopio muestra el _____ en el eje horizontal y la _____ en el vertical. _____ y DC offset son dos magnitudes relacionadas a la tensión, que se consideran de acuerdo a la forma de la señal.
inversa	
Amplitud	Para una onda cuadrada, la amplitud es la tensión entre las partes alta y la baja de la señal. Para una onda triangular simétrica _____, así como también la senoide, la amplitud es la máxima desviación de la línea central de la señal. Para estas señales, el _____ es la tensión entre la línea central y 0 Volts.
circuito RC	
frecuencia	
simétrica	La frecuencia es la cantidad de veces que una forma de onda se repite a sí misma y el _____ es la cantidad de tiempo que toma cada repetición. La frecuencia es la _____ del período.
sinusoidales	
señales	Las señales de tensión que varían en el tiempo que se envían a un parlante pueden dar como resultado sonidos audibles. justando la _____ de la señal se controla el volumen del sonido. Ajustando la frecuencia se controla el _____.
tono	
DC offset	
	Las notas musicales se representan por ondas _____ que oscilan a ciertas frecuencias. El BASIC Stamp envía una modulación por ancho de pulso (PWM) a la entrada de un _____ para generar una senoide a la salida.

**Preguntas**

1. ¿Cuál es la diferencia entre la pantalla del Stamp-O-Scope y la de un osciloscopio común?
2. Si la frecuencia de una señal es de 1000 Hz, ¿cuál es el período?
3. ¿Cómo puede ajustar la escala para mostrar variaciones más pequeñas de tensión con el Stamp-O-Scope? Pista: La respuesta involucra un ajuste al código PBASIC del Programa 5.1.
4. ¿Cómo puede aumentar la velocidad de toma de datos del Stamp-O-Scope? Pista: Examine el código nuevamente; ¿qué factor limita la velocidad a la que se mide la señal?

**5****Desafío**

1. Modifique el Programa 3.1 para que toque dos notas simultáneamente.
2. Modifique el código y agregue un segundo pin que mida solamente niveles lógicos (0 y 1). Modifique el código para que muestre la actividad de esta segunda punta y la entrada del Stamp-O-Scope simultáneamente en la misma pantalla. ¡Buena suerte!
3. Si tuvo éxito con el Desafío 2, agregue un potenciómetro a la Plaqueta de Educación y conéctelo como divisor de tensión. Conecte el cursor del potenciómetro a la punta nueva. Mueva el potenciómetro por encima y por debajo del umbral de tensión y vea si puede medir la frecuencia de la onda triangular.
4. Sería interesante demostrar un fenómeno llamado desplazamiento de fase. Conecte la entrada que desarrolló en el Desafío 2 a la salida del conversor D/A. Además, conecte la punta del canal 1 (ADC0831 Vin(+)) a la salida del conversor D/A. Ajuste el conversor D/A para que cuente hacia arriba y abajo entre 4 y 15. Note que hay un retardo entre el punto medio de la onda triangular y la transición de la onda cuadrada. Este retardo es denominado desplazamiento de fase.

**¿Por qué aprendí esto?**

El osciloscopio es una herramienta esencial en la industria electrónica y empleado también por muchos aficionados. Esta es una buena introducción al uso del osciloscopio y a la forma de medir señales que varían en el tiempo. Si intenta aprender a usar un

osciloscopio, algunos de los conceptos introducidos aquí harán que le resulte más fácil de entender el funcionamiento de algunos

### **¿Cómo puedo aplicarlo?**

Al familiarizarse con el osciloscopio y formas de onda temporales, le resultará más fácil comprender explicaciones de fenómenos que varían con el tiempo. Éstos aparecen en libros de química, física y electrónica.



## Capítulo 6: Capturando Datos sobre Frecuencia

---

En el Capítulo anterior usamos el BASIC Stamp para que un conversor D/A produjera sonidos audibles. En este Capítulo, usaremos un temporizador 555 para generar frecuencias en el rango de sonido audible, luego usaremos el BASIC Stamp para hacer un muestreo y procesar los datos sobre la frecuencia.

Comenzaremos observando la onda cuadrada generada por un temporizador 555 en una versión simplificada del Stamp-O-Scope. Se usarán dos potenciómetros para controlar la frecuencia y el ciclo de trabajo de la onda cuadrada. Después de ver las características de la onda cuadrada, la salida del 555 se conectará a un parlante piezoeléctrico. La salida del 555 también se conectará a un pin de E/S del BASIC Stamp configurado como entrada. El BASIC Stamp se programará para monitorear y registrar las frecuencias generadas por el temporizador 555.

**6**

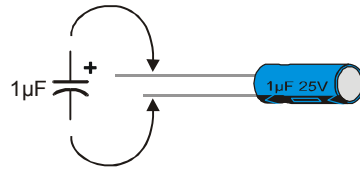
### **Componentes Requeridos**

Este experimento requiere los siguientes componentes:

- (2) Potenciómetros de 100 K Ohms
- (1) Parlante piezoeléctrico
- (1) Temporizador 555
- (1) Capacitor electrolítico de 10 uF
- (1) Capacitor de 0.1 uF
- (2) Resistor de 2 K Ohm

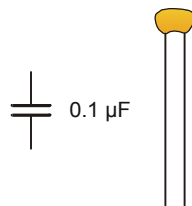
No todos los componentes serán usados a la vez, en el mismo circuito. Ciertos resistores y capacitores se reemplazarán en el segundo circuito, para incrementar la frecuencia de salida del 555.

El capacitor electrolítico mostrado en la Figura 6-1 tiene un terminal positivo y otro negativo. El alambre que sale de la lata de metal y que es más cercano a la raya negra es el terminal negativo. Las flechas (>>) en el punto negro de la raya al terminal negativo. La lata también tiene el valor impreso en ella. 1uF o 1MF indica 1 microfaradio, mientras que 10uF O 10MF indica un valor de capacitancia de 10 microfaradios.



**Figura 6-1**  
Capacitor Electrolítico.  
Símbolo del Circuito y  
Componentes.

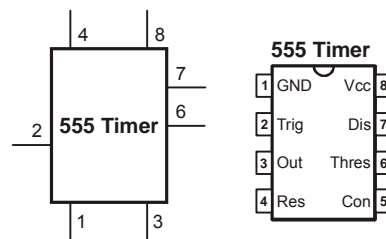
La Figura 6-2 muestra el símbolo esquemático y un dibujo de un capacitor de 0.1  $\mu\text{F}$  del Kit de componentes. A diferencia de los capacitores electrolíticos, este capacitor no posee polaridad. En otras palabras, no tiene importancia qué terminal recibe un nivel mayor de tensión. Tampoco hay terminales + y - y la forma del símbolo esquemático es un poco diferente.



**Figura 6-2**  
Capacitor de 0.1  
Micro Faradio.

*No hay polaridad en  
este capacitor.*

La Figura 6-3 muestra el símbolo y la distribución de pines para el temporizador 555. Como con el ADC0831, el símbolo esquemático para el temporizador 555 se dibuja según la conveniencia del diagrama. Esto significa que la ubicación de los pines y el tamaño del símbolo pueden cambiar de un diagrama a otro. La distribución de pines por otro lado, no cambia. Asegúrese de ubicar la marca índice. También recuerde usar la distribución de pines conjuntamente con el diagrama cuando arme un circuito.



**Figura 6-3**  
Símbolo y  
distribución de pines  
del Temporizador  
555.

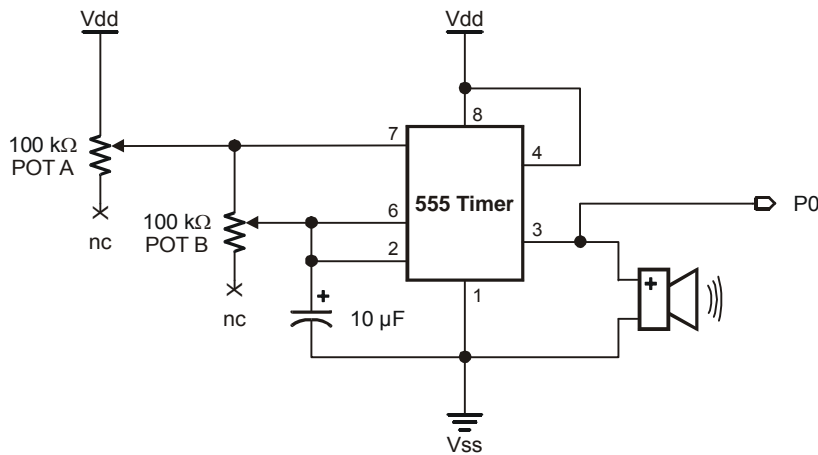
*Como siempre,  
asegúrese de ubicar  
correctamente la  
marca índice cuando  
coloque el  
componente en la  
protoboard.*

### Constrúyalo

El circuito de la Figura 6-4 es esencialmente el mismo "multivibrador astable" usado en el libro "¿Qué es un Microcontrolador?". El circuito es usado para generar una secuencia estable de pulsos. Hay dos diferencias entre este circuito y el de "¿Qué es un Microcontrolador?". La primera es que la salida del temporizador 555 alimenta un parlante piezoeléctrico en lugar de un LED. La segunda es que este circuito emplea dos potenciómetros como resistores variables en lugar de uno solo. Esto nos permite controlar el ancho de pulso y el ciclo de trabajo de la salida.

Construya el circuito que se muestra en la Figura 6-4. Note que aparece "nc" en minúsculas debajo de dos terminales en los potenciómetros. Esto significa "no conectado". Dado que el potenciómetro es usado como un resistor variable en lugar de divisor de tensión, el tercer terminal queda sin conectar. Es correcto insertar el terminal nc en la protoboard; simplemente deje el resto de la fila sin conectar.


6



**Figura 6-4**  
Circuito de  
Multivibrador  
Astable

*Con tiempos  
en estado alto  
y bajo  
ajustables.*

Después de construir el circuito, ajuste el potenciómetro hasta que el parlante piezoeléctrico emita un ligero pero estable sonido de "tic-toc". Una frecuencia de 1 Hz es la óptima. Puede hacer que el tic-toc se detenga en cualquier momento conectando el pin 5 del temporizador 555 a masa (Vss).



**¿Cómo predecir la frecuencia del Temporizador 555?**

$f = 1.45/[C \times (R_A + 2R_B)]$

Más información puede ser extraída de la hoja de datos. TH, o ancho de pulso, es dado por:

$0.69 \times C \times (R_A + R_B)$ ,

donde C,  $R_A$  y  $R_B$  son los valores del capacitor y dos valores de resistencia de los potenciómetros A y B, como se muestra en la Figura 6-4. TL, o duración en estado bajo (tiempo entre pulsos) es dado por:

$0.69 \times C \times R_B$

En el experimento previo, el Stamp-O-Scope realizó un muestreo digital y mostró los datos de la señal. El Programa 6.1 es una versión muy simplificada del Stamp-O-Scope (el Stamp-O-Scope 2) que solamente muestra variaciones en niveles lógicos (0 ó 1). Dado que la salida del temporizador 555 es una onda cuadrada que varía entre 0 y 5 Volts, no se requiere conversión A/D. Solamente necesita el Stamp-O-Scope 2.

```
' Basic Analog and Digital - PL6_1R0.bs2

' Stamp-O-Scope 2
' {$STAMP BS2}
' {$PBASIC 2.5}

DEBUG CLS      'Inicia ventana debug

DO
  DEBUG REP "  \IN0*20, "*", CR
  PAUSE 100

LOOP
```

### Explicación del Programa

El comando **DEBUG REP " "\IN0\*20, "\*", CR** usa el valor de entrada medido en el pin P0 para indicar cuantos espacios deben imprimirse. No se usa ninguna variable. Cuando la entrada es cero, el asterisco se imprime sin ningún espacio delante, indicando un 0 a entrada. Cuando el valor en P0 es 1, se imprimen veinte espacios delante del asterisco, indicando un valor de entrada medido de 1.

### La Salida

La Figura 6-5 muestra la pantalla del Stamp-O-Scope 2. Puede ajustar los potenciómetros para cambiar las características y la frecuencia de la forma de onda. Por ejemplo, pot B afecta directamente la cantidad de tiempo que la señal está en estado bajo. También afecta indirectamente la cantidad de tiempo que está en estado alto. Ajuste pot B hasta que la parte baja de la señal tenga 5 asteriscos de ancho. Luego puede ajustar pot A para variar el ancho del pulso. Ajuste pot A para que el ancho del pulso también sea de 5 asteriscos de ancho. Las versiones más nuevas de la ventana Debug incluyen un botón “pause” y “resume” que permite detener momentáneamente la impresión de datos.

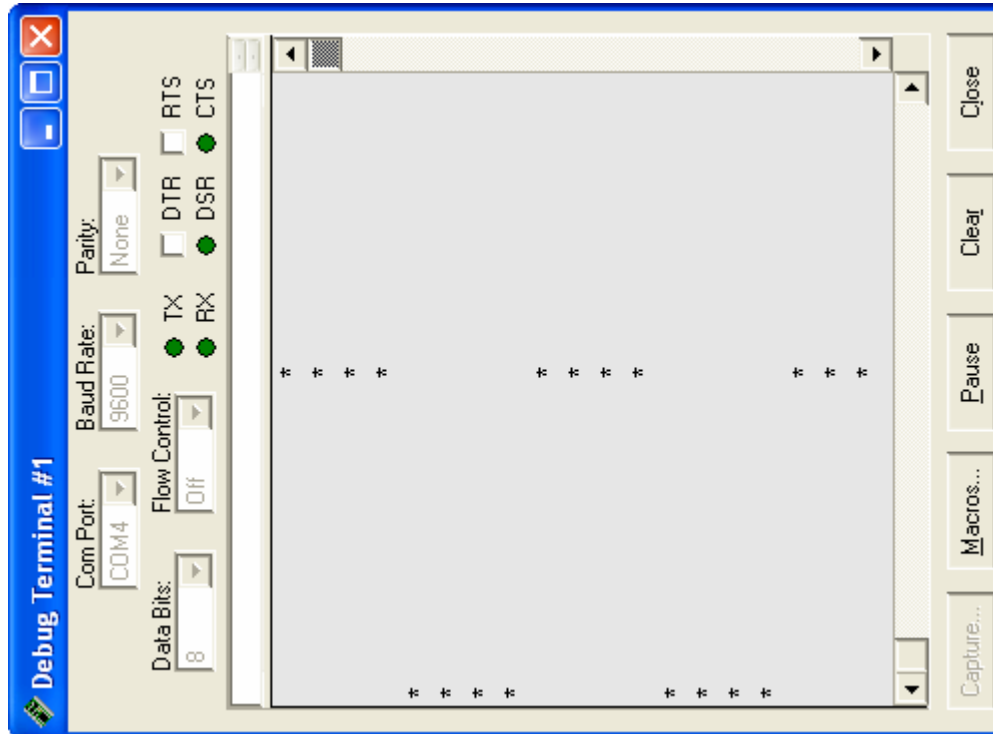


Figura 6-5: Ejemplo de Salida

Así se ve la onda cuadrada generada por el temporizador 555 medida por el pin P0. Nuevamente la ventana se ha rotado 90° para que se vea como en los osciloscopios normales.

Ahora ajuste pot B para que la parte baja de la señal tenga dos asteriscos de ancho. ¿Qué sucedió con el ancho de pulso? También debería haberse angostado.

Luego ajuste pot B hasta su tope máximo, para incrementar la frecuencia (haciendo el ancho de pulso más angosto). El parlante piezoeléctrico debería estar emitiendo los clics bastante rápido y el Stamp-O-Scope 2 ya no estaría funcionando muy bien, como se muestra en la Figura 6-6: Lo reemplazaremos con otro programa que nos permita medir señales audibles.

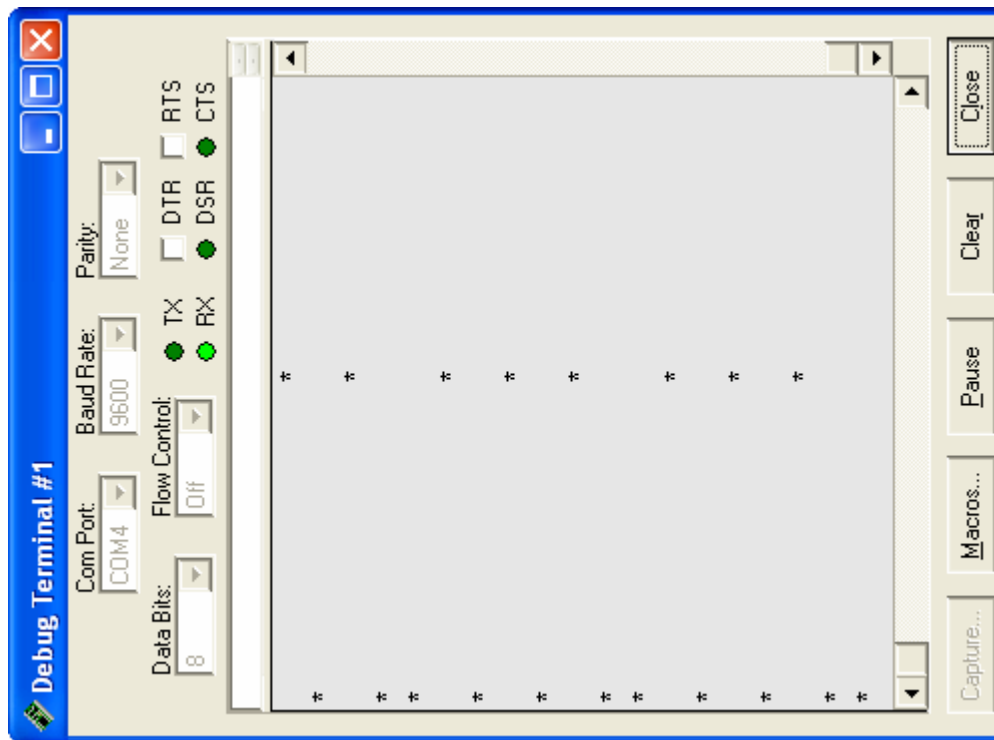


Figura 6-6: Ejemplo de Salida

La onda cuadrada se está repitiendo a sí misma más rápidamente que la velocidad de muestreo. Por este motivo, la imagen generada por el Stamp-O-Scope 2 ya no es válida

Ahora regrese a pot B a la posición original, donde escuche un clic periódico por el parlante piezoeléctrico. Volveremos a mover a pot B para escuchar el sonido más adelante, pero primero consideremos lo que sucedió con la pantalla del Stamp-O-Scope 2 cuando la frecuencia del temporizador 555 fue incrementada.

La imagen mostrada en la Figura 6-6: muestra actividad de señal, aunque es errónea. Esto se denomina aliasing. El aliasing se produce cuando no se muestrea una señal suficientemente rápido como para obtener una representación válida. El aliasing puede causar problemas debido a que algunas veces la señal que se muestra en la pantalla parece la real.

6

En este caso, la ventana debug es el factor limitante para la velocidad de muestreo. Hagamos un programa que le permita al BASIC Stamp trabajar a máxima velocidad en la adquisición de datos y luego envíe la información a la ventana debug.



**¿Cuándo es un problema el aliasing?** La velocidad de muestreo óptima, depende de las características de la señal a medir. Algunas señales se muestrean unas pocas de veces por ciclo mientras que otras se muestrean miles de veces.

La mínima velocidad de muestreo absoluta, en teoría, que puede ser empleada para obtener datos válidos, es el doble de la frecuencia de la señal bajo medición. Cuando la velocidad de muestreo es menor que el doble de la frecuencia de la señal, es inevitable el aliasing. Esta frecuencia mínima se denomina límite de Nyquist.

Un pin de E/S del BASIC Stamp puede ser usado para controlar la cantidad de cruces por el umbral de tensión, en un intervalo de tiempo. El BASIC Stamp puede ser programado para realizar este control una vez cada dos microsegundos. El período de muestreo es de 2 microsegundos.

Del Capítulo 5, sabemos que la frecuencia es la inversa del período:

$$f = 1 / T$$

Así, la velocidad de muestreo es:

$$1 \div 2 \times 10^{-6} \text{ seconds} = 250 \times 10^3 = 500 \text{ kHz}$$

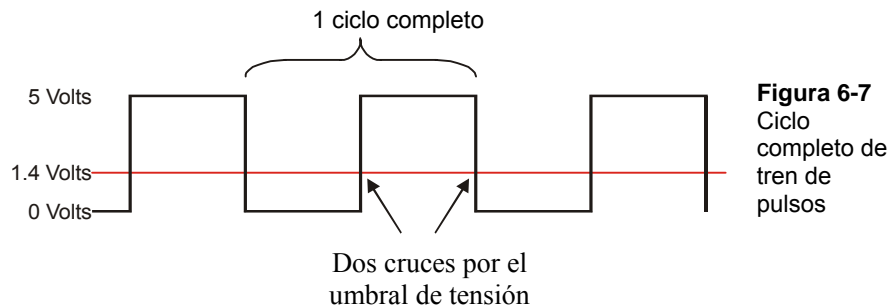
Dado que la velocidad de muestreo es de 500 kHz, la máxima frecuencia teórica a la que podríamos realizar el muestreo es de 250 kHz. El rango de audición es de 20 Hz a 20 kHz

y los sonidos con los que trabajaremos en este experimento variarán entre 50 Hz y 3,5 kHz. Es claro que el aliasing no será un problema.

### Prográmelo

Hagamos un programa que cuente cuántas veces se repite un tren de pulsos. En efecto, haremos un programa que determine la frecuencia de la onda cuadrada. El BASIC Stamp tiene una función interna que le permite registrar una frecuencia y se trata del comando PBASIC llamado count.

El BASIC Stamp incrementa el contador cuando la tensión de entrada atraviesa dos veces el umbral de 1,4 Volts en el pin de E/S. Esto hace fácil de interpretar y programar datos de frecuencia de señales periódicas tales como la onda cuadrada, la onda triangular y la sinusoide. La **Error! Reference source not found.** muestra por qué debe haber dos cruces por el umbral por cada repetición de la forma de onda. La parte que se repite, se denomina ciclo.



Ingresa el Programa 6.2 en el Stamp Editor y guárdalo como PL6\_2R0.bs2.

```
' Analógico y Digital Básicos - PL6_2R0.bs2
' Frecuencímetro
' {$STAMP BS2}
' {$PBASIC 2.5}

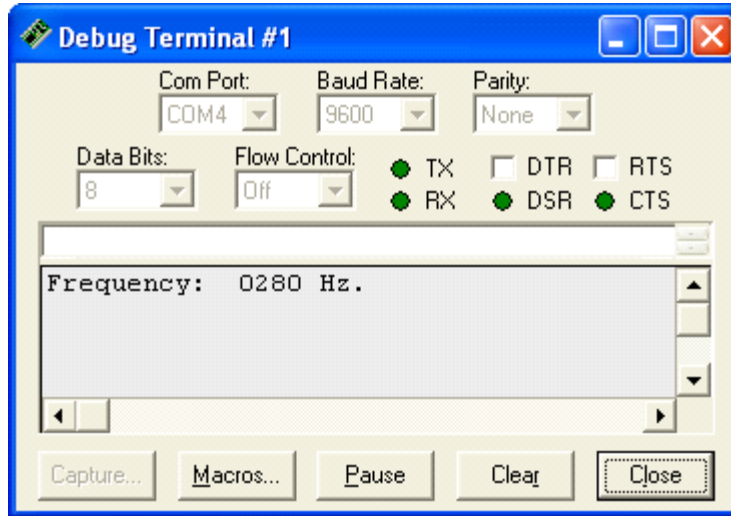
f      VAR      Word

DO
  COUNT 0, 1000, f
  DEBUG HOME, "Frecuencia: ", DEC4 f, " Hz.", CR, CR
LOOP
```



Cuando ejecute el programa y ajuste pot B al máximo, la salida debería ser similar a la frecuencia mostrada en la **Error! Reference source not found.**

When you run the program, and adjust pot B to its maximum, the output should be somewhere in the neighborhood of the frequency shown in Figura 6-8.

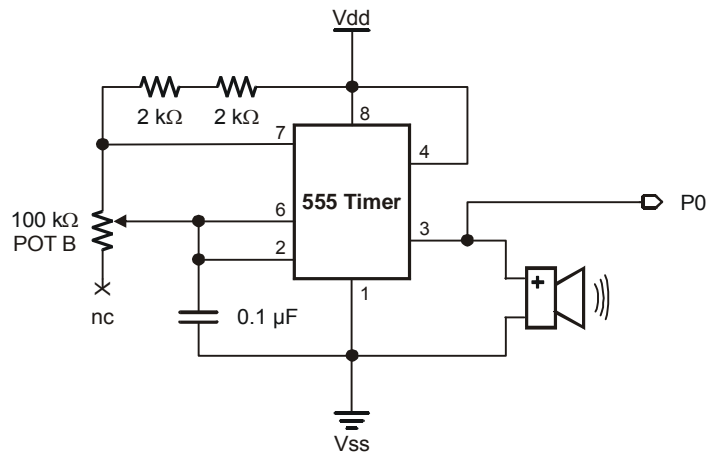


**Figura 6-8**  
Salida de Debug  
para el Programa  
6.2.

6

Recuerde que puede apagar el sonido (ruido, zumbido, o cualquier nombre que le haya dado) conectando el pin 5 del temporizador 555 a Vss. Cuando quiera activar nuevamente el sonido, desconecte el cable del terminal de Vss.

Modifique el circuito de prueba reemplazando pot A por dos resistores de 2 k $\Omega$  en serie y el capacitor del 10  $\mu$ F por uno de 0.1 micro Faradios. La Figura 6-9 muestra como debería verse el circuito una vez realizados los cambios.



**Figura 6-9**  
Circuito del  
Multivibrador Astable

con  $R_A=4\text{ k}\Omega$  fija.  
 $R_B$  aún se puede  
ajustar con pot B.

Ajustando pot B, debería poder medir sonidos con un rango de frecuencia de 60 Hz a 3,5 kHz. Puede obtener lecturas de frecuencia a medida que ajusta el potenciómetro con la versión revisada del Programa 6.2 de abajo.

```
' Analógico y Digital Básicos - PL6 2R1.bs2
' Frecuencímetro
' {$STAMP BS2}
' {$PBASIC 2.5}

f      VAR      Word(10)
n      VAR      Nib

DO
  DEBUG CLS

  FOR n = 0 TO 9
    COUNT 0, 1000, f(n)
    DEBUG HOME, "Frecuencia: ", DEC4 f, " Hz.", CR, CR
  NEXT

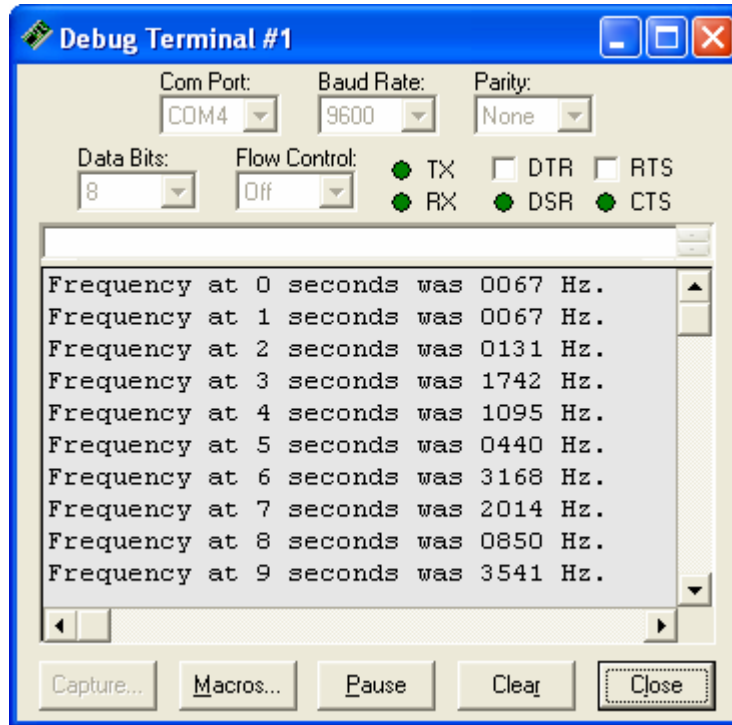
  PAUSE 1000

  FOR n = 0 TO 9
    DEBUG "La frecuencia a los ", DEC1 n
    DEBUG " segundos fué ", DEC4 f(n), " Hz.", CR
  NEXT

  PAUSE 5000
LOOP
```

Durante el tiempo que el programa guarda la frecuencia, la ventana Debug es similar a la revisión previa del programa. Cuando el programa está guardando, el display tiene 10 ejemplos de frecuencia.

Los datos de la ventana debug mostrados en la Figura 6-10 se generaron mientras se movía el potenciómetro hacia uno y otro lado, generando algunas frecuencias al azar.



6

**Figura 6-10**  
Salida de Debug  
para el Programa  
6.2, Revisión 1.

### Explicación del Programa

El Programa 6.2 Revisión 2 introduce el concepto de almacenamiento de datos usando un arreglo (o vector). Usamos el comando:

```
f      VAR      Word(10)
```

para reservar un espacio de 10 words en la RAM del BASIC Stamp. Una word se llama  $f(0)$ , otra es  $f(1)$  y así hasta  $f(9)$ . Cada una de estas words se almacenan una al lado de la otra, en la memoria RAM del BASIC Stamp.

A medida que el valor de  $n$  se incrementa en el bucle **for-next**, también lo hace el índice del arreglo. La primera vez se carga un valor en la word  $f(0)$ , la segunda vez se carga en la word  $f(1)$ , y así hasta la word  $f(9)$

```
FOR n = 0 TO 9
  COUNT 0, 1000, f(n)
  DEBUG HOME, "Frecuencia: ", DEC4 f, " Hz.", CR, CR
NEXT
```

El mismo concepto se empleó para imprimir los valores en la ventana debug.

```
FOR n = 0 TO 9
  DEBUG "Frecuencia a los ", DEC1 n
  DEBUG " segundos fué ", DEC4 f(n), " Hz.", CR
NEXT
```

Las formas de onda que varían en el tiempo que hemos visto hasta ahora han sido periódicas. Esto quiere decir que tenían una porción que se repite. En los próximos capítulos, estudiaremos formas de onda que varían en el tiempo, pero que no son periódicas.

**¿Qué aprendí?**

En las oraciones de abajo, inserte las palabras apropiadas de la lista de la izquierda.

muestreo	Los sonidos pueden ser caracterizados por _____ que varían en el tiempo. El rango de frecuencias audibles es de 20 a 20.000 Hz. Cuando una forma de onda tal como una onda cuadrada de cierta frecuencia y con suficiente amplitud se envía a un parlante, éste emite un sonido audible.
onda cuadrada	
formas de onda	El temporizador 555 en configuración de multivibrador astable puede ser ajustado para generar una _____. La frecuencia, el ancho de pulso y el _____ pueden ser ajustados modificando los valores de los elementos pasivos del circuito, tales como el capacitor y los dos resistores.
cantidad	
arreglo o vector	El ancho de pulso es la _____ de tiempo que una señal permanece en estado alto y el ciclo de trabajo es la relación entre el ancho de pulso y el _____ a forma de onda.
incrementado	
período	Aliasing es un fenómeno que se produce cuando la velocidad de _____ es menor al doble de la frecuencia de la señal. Esto puede causar importantes errores en la interpretación de los datos de la señal.
ciclo de trabajo	Los datos pueden ser eficientemente almacenados y recuperados de la RAM usando un _____. El índice de un arreglo puede ser _____ usando un bucle for-next. Cuando especifica un arreglo de 10 bytes, se numeran del 0 al 9..

### Preguntas

1. Dadas estas cuatro frecuencias de sonido: 3,5 Hz, 350 Hz, 3.500 Hz, 35.000 Hz, ¿cuáles puede oír y cuáles no? Fundamente sus respuestas. Además, compare el tono de los sonidos que podría oír.
2. Explique que es aliasing. Si una señal es muestreada a 1 kHz, ¿cuál es la máxima frecuencia de la señal a medir que comenzaría a producir aliasing?
3. En el circuito de la **Error! Reference source not found.**, ¿qué valor de resistencia debería tener el potenciómetro para que el circuito del multivibrador astable genere una señal de 2 kHz? La solución requiere algo de álgebra.
4. Desarrolle una instrucción en PBASIC que declare un arreglo de 5 nibbles. Escriba la instrucción para asignarle al primer nibble el valor del quinto nibble del arreglo.

### Desafío

1. En el Programa 6.2 Revisión 2, las muestras de frecuencia se realizaban cada segundo debido a que el comando **count** realizaba la cuenta durante un segundo. Modifique el programa para que obtenga mediciones de frecuencia cada  $\frac{1}{2}$  segundo. Asegúrese de que los datos de salida sean correctos, comparándolos con valores de frecuencias específicas de los experimentos anteriores. Repare todos los errores.
2. Diseñe un programa que grabe y reproduzca tonos. Hágalo de forma que el tono que se detecta y almacena se emita por un parlante y el tono que se reproduce se emita por otro parlante. Agregue una función al programa que apague el temporizador 555 enviando un estado bajo al pin-4 del temporizador, cuando desee escuchar el sonido de la reproducción. Pista: Puede combinar partes de dos programas, una del Experimento 5 y otra de este experimento, quedando casi todo el trabajo resuelto. Lo único que deberá agregar es la señal de control que deshabilita el temporizador 555.
3. Diseñe y arme el circuito para el Desafío 2. Necesitará ser creativo para encontrar espacio suficiente en la protoboard.

4. Agregue dos pulsadores al sistema desarrollado en los Desafíos 2 y 3 de forma que pueda controlar cuando grabar y cuando reproducir sonidos simples. Ahora necesitará ser realmente creativo para encontrar espacio suficiente en la protoboard. Use resistores de 10 k $\Omega$  con los pulsadores y diseñelos para que generen estados altos o bajos. Pista: Repase las técnicas de programación y montaje de circuitos del Capítulo 1 para colocar los pulsadores.

#### **¿Por qué aprendí esto?**

El campo del procesamiento digital de señales de audio aún se está expandiendo en la industria del audio. Formas nuevas y más eficientes de transmitir y almacenar sonidos se están desarrollando para poder transmitir música más rápidamente por internet y para que su voz suene más parecida a su verdadera voz, al otro extremo de una línea telefónica.

**6**

El procesamiento digital de señales es también el centro de la industria de telecomunicaciones con teléfonos celulares y teléfonos comunes. Cuando habla por teléfono, la señal de su voz se convierte a unos y ceros que se envían por la red telefónica en formato binario. Su teléfono aún envía y recibe señales de voz analógicas, pero la información de su voz es convertida de A/D y D/A en varios puntos de la red telefónica.

#### **¿Cómo puedo aplicarlo?**

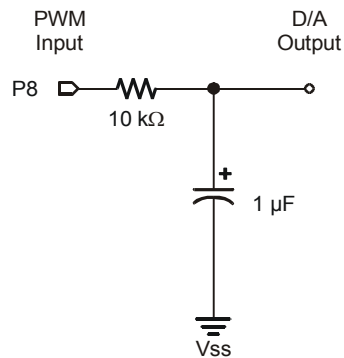
La frecuencia está muy relacionada al sonido, pero hay una miríada de usos distintos. Por ejemplo, la velocidad de un auto y la del motor se determinan por la velocidad de giro de ciertas partes del vehículo. Cuando algo rota, lo que sucede en cada giro se repite. Si una polea está rotando a 100 giros por segundo, su frecuencia de rotación es 100 Hz. La vibración de las máquinas también puede medirse en frecuencia, así como también muchas señales metabólicas como la respiración, el ritmo cardíaco, etc.





## Capítulo 7: Digital a Analógico Fácil con PWM

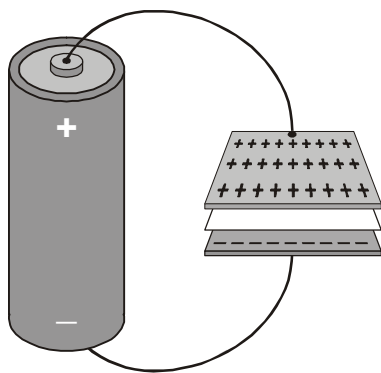
Un resistor, un capacitor, un BASIC Stamp y una sola línea de código PBASIC. Eso es todo lo que se necesita para construir un conversor D/A con una resolución de 8-bits. ¿Suena demasiado bueno para ser verdad? En algunos casos se necesitará un búfer para que el circuito pueda mantener la tensión en la salida. Aún así, es fácil de construir y ofrece un grado de precisión superior a la de la red resistiva en escalera. La Figura 7-1 muestra el circuito usado, que se trata de un circuito RC simple. La entrada recibe una señal de modulación de ancho de pulso (PWM) y la salida sube o baja al nivel de tensión deseado.



**Figura 7-1**  
Circuito RC

*que puede ser  
conectado al  
BASIC Stamp  
para realizar la  
conversión D/A.*

¿Cómo pueden una serie de pulsos de ancho variable, generar un nivel de tensión a la salida del conversor D/A? La respuesta es el circuito RC mostrado en la Figura 7-1. Este circuito se comporta como una batería recargable. La Figura 7-2 muestra un ejemplo del tipo más simple de capacitor, el capacitor de placas paralelas. Las cargas se desplazan desde los terminales de la batería, acumulándose en las placas metálicas hasta que la tensión sobre el capacitor es esencialmente la misma que en la batería.



**Figura 7-2**  
Capacitor de Placas Paralelas  
Cargado por una Batería.

*Las cargas opuestas se agrupan en sus respectivas placas. Quieren cruzar el salto entre las placas, pero no pueden hacerlo debido a un material aislante que las separa, denominado dieléctrico.*

Si desconecta la batería del capacitor, igualmente mantendrá su tensión. Normalmente, hay una pequeña corriente que se pierde a través del dieléctrico que separa las placas. Recibe el nombre de corriente de fuga. La corriente de fuga en un capacitor cargado, hará que la tensión disminuya lentamente.

El circuito RC mostrado se puede cargar hasta un 1% por debajo de 5 Volts muy rápidamente. Suponga que se envía un único pulso al circuito de la Figura 7-1 en lugar de los pulsos que enviaría una señal de PWM. Si los valores de los componentes son exactos, se necesitarían 6,93 milisegundos para que el capacitor se cargue a 2,5 Volts y llevaría 46,1 milisegundos para que la salida llegue hasta los 4,95 Volts, que es el 99% del total de 5 Volts.

Para entusiastas avanzados de las matemáticas, echemos un vistazo a las pruebas. La ecuación para el circuito RC de la Figura 7-1 respondiendo a un pulso que va de 0 a 5 Volts es:

$$V_{\text{Output}} = V_{\text{Input}} \times \left( 1 - e^{\frac{-t}{R \times C}} \right)$$

Despejando para separar el término exponencial:

$$1 - \frac{V_{\text{Output}}}{V_{\text{Input}}} = e^{\frac{-t}{R \times C}}$$

El logaritmo natural se aplica a ambos lados de la ecuación para quitar el término exponencial y ordenando los términos obtenemos:

$$\ln\left(1 - \frac{V_{\text{Output}}}{V_{\text{Input}}}\right) = \ln\left(e^{\frac{-t}{R \times C}}\right) \Rightarrow \frac{-t}{R \times C} = \ln\left(1 - \frac{V_{\text{Output}}}{V_{\text{Input}}}\right) \Rightarrow t = -R \times C \times \ln\left(1 - \frac{V_{\text{Output}}}{V_{\text{Input}}}\right)$$

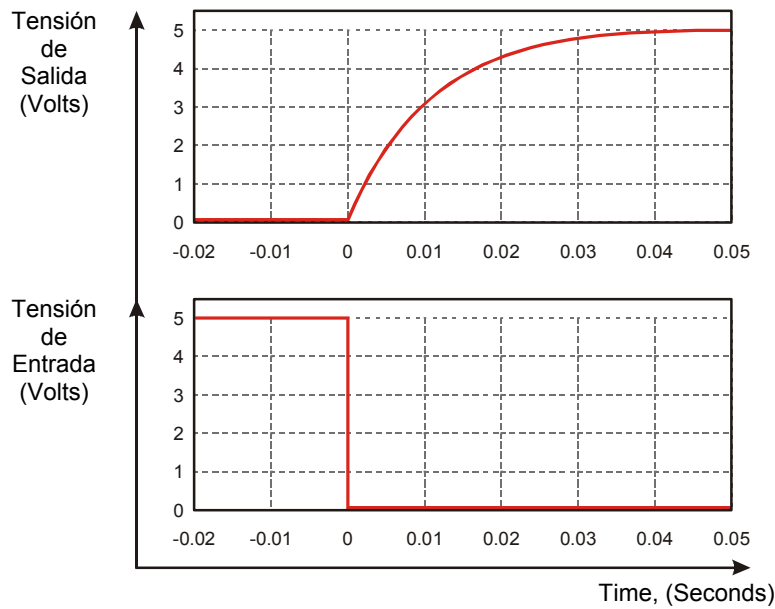
Lo único que falta es reemplazar los valores para resolver en  $t_{2,5 \text{ Volts}}$  y en  $t_{4,95 \text{ Volts}}$ .  $R$  y  $C$  son los valores de resistencia y capacitancia mostrados en la Figura 7-1 y  $V_{\text{Entrada}} = 5 \text{ Volts}$ . La primer  $V_{\text{Salida}}$  es 2,5 Volts.

$$t_{2,5 \text{ Volts}} = -R \times C \times \ln\left(1 - \frac{V_{\text{Output}}}{V_{\text{Input}}}\right) = -(10 \times 10^3) \times (1 \times 10^{-6}) \times \ln\left(1 - \frac{2,5}{5,0}\right) = 6,93 \times 10^{-3} \text{ seconds}$$

Luego, se realizan los mismos cálculos para  $t_{4,95 \text{ Volts}}$ .

$$t_{4,95 \text{ Volts}} = -R \times C \times \ln\left(1 - \frac{V_{\text{Output}}}{V_{\text{Input}}}\right) = -(10 \times 10^3) \times (1 \times 10^{-6}) \times \ln\left(1 - \frac{4,95}{5,0}\right) = 46,1 \times 10^{-3} \text{ seconds}$$

Se podría usar un osciloscopio para controlar la tensión de salida del circuito RC de la Figura 7-1. La respuesta cuando se aplica un único pulso muy ancho (un escalón de tensión) sería similar al de la Figura 7-3. Como puede observar, solamente demora una fracción de segundo hasta que el capacitor se carga a 5 Volts. Igualmente, se trata de un cambio de estado gradual cuando se compara con los pulsos enviados por el BASIC Stamp, que pueden ser de un ancho mínimo de 2 microsegundos.



**Figura 7-3**  
La respuesta al  
escalón del  
circuito RC

*Muestra la  
respuesta de  
tensión a un  
único escalón de  
tensión aplicado  
a la entrada.*

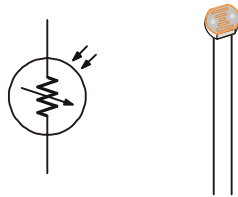
Al aplicar muchos pulsos angostos en lugar de un único pulso muy ancho como el de la Figura 7-3, se mejora la precisión. Como verá muy pronto, esta técnica de conversión D/A es sorprendentemente precisa. Más aún teniendo en cuenta que el resistor tiene una tolerancia del 10% y el capacitor solamente del 20%.

### **Componentes Requeridos**

Este experimento requiere los siguientes componentes:

- (1) Conversor A/D ADC0831
- (1) Op-amp LM358
- (2) Resistor de 10 k-Ohm
- (1) Capacitor electrolítico de 1.0  $\mu\text{F}$
- (1) Fotorresistor
- (1) Resistor de 470 Ohm
- (1) LED amarillo

El fotorresistor sensa niveles de luz. Bajo luz intensa, puede caer a unos pocos Ohms. En la oscuridad, la resistencia puede aumentar hasta 50 K Ohms. La Figura 7-4 muestra el símbolo esquemático y un dibujo de un fotorresistor.

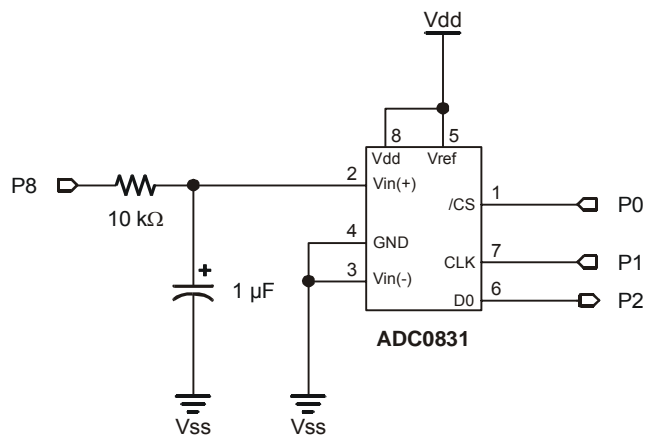


**Figura 7-4**  
Fotorresistor: Símbolo  
esquemático y  
componente

### **Constrúyalo**

Arme el circuito mostrado en la Figura 7-5. Recuerde del Capítulo 4 que las entradas del ADC0831 tienen resistencia muy alta y se tratan como un circuito abierto. Debido a esto, es correcto conectar la salida del conversor D/A a la entrada  $V_{in}(+)$  del ADC0831 sin un búfer.

7



**Figura 7-5**  
Circuito RC para la  
Conversión D/A con PWM

*Conectado al terminal  $V_{in}(+)$  del ADC0831. Éste es conectado a la salida del conversor D/A para medir las tensiones de salida.*

Vamos a usar el comando **PWM** para enviar pulsos al circuito RC para realizar la conversión D/A. Luego mediremos la tensión usando el ADC0831 como voltímetro.

El formato del comando **PWM** es:

**PWM *pin, trabajo, ciclos***

El parámetro **pin** se refiere al pin de E/S del BASIC Stamp y puede especificar un número entre 0 y 15. Dado que la entrada del conversor está conectada al pin P8 en el diagrama, usaremos un valor de pin igual a 8.

Ahora, ¿qué hay sobre **trabajo**? Recuerde del Capítulo 5 que el ciclo de trabajo es la relación entre ancho de pulso y período. El parámetro trabajo es diferente en el sentido de que no se trata del trabajo de un ciclo. Es el trabajo de todos los pulsos sobre cierto período de tiempo. En otras palabras, el término **trabajo** especifica la relación entre tiempo en estado alto de la señal durante todos los pulsos en un intervalo de tiempo específico.

Cuando trabajo es 255 significa que todos los pulsos están en estado alto y no hay estados bajos entre ellos. Cuando todos los pulsos son altos, el capacitor se cargará hasta los 5 Volts. En teoría, en realidad el capacitor nunca podría cargarse hasta exactamente 5 Volts, pero en la práctica, un tiempo de  $5 \times R \times C$  hace que la tensión sobre el capacitor llegue a 99.3% de 5 Volts.

Por lo tanto, **trabajo** es el parámetro que controla el nivel de tensión pero, ¿cómo sabremos cuánto deberá valer trabajo? Es bastante simple ya que se trabaja con una escala de 0 a 255, donde 0 es 0 Volts y 255 son 5 Volts. Esto es igual a lo que tratamos en el Capítulo 3.

Supongamos que deseamos generar 3.25 Volts. En el Capítulo 3, aprendimos a convertir una escala de 0 a 5 Volts a un rango de 0 a 255. Recuerde que la tensión medida es a 5 como la salida del conversor A/D es a 255. Esto traducido a fracciones es:

$$\frac{\text{Voltage}}{5} = \frac{\text{Decimal A/D input}}{255}$$

En este caso, la tensión de salida del conversor D/A es a 5 como trabajo es a 255, lo que se traduce así a fracciones:

$$\frac{\text{D/A Output}}{5} = \frac{\text{duty}}{255}$$

Normalmente sabemos la tensión que queremos, así que despejemos la ecuación para obtener el parámetro trabajo que necesitamos:

$$\text{duty} = 255 \times (\text{D/A Output} \div 5)$$

Dado que queremos una salida de 3,25 Volts:

$$\text{duty} = 255 \times (3.25 \div 5) = 165.75$$

El número 165,75 debería ser redondeado a 166, de forma que podamos asignar a trabajo un valor entero.

La última cantidad que debemos determinar para el comando **PWM** es el parámetro **ciclos**, que especifica la cantidad de milisegundos que durará la salida del PWM y puede valer hasta 65.535. La regla para determinar la cantidad de ciclos es:

$$\text{cycles} = 4 \times R \times C, \text{ in milliseconds}$$

Dado que usamos un capacitor de 1.0 uF y un resistor de 10 kΩ,

$$\text{cycles} = 4 \times 0.000001 \times 10,000 = 0.04 = 40 \times 10^{-3} = 40 \text{ ms}$$

Para los componentes que estamos usando, debemos usar un valor de **ciclos** en el comando **PWM** igual a 40.

Ahora sabemos que para generar una salida de 3,25 Volts, nuestro comando **PWM** debe ser:

```
PWM 8, 166, 40
```

Probémoslo.

### **Prográmelo**

Modifique una sección de las declaraciones del Programa 4.1, así la variable **n** se define como **Word** en lugar de **nibble**

```
' ----[ Declaraciones ]-----
adcBits      VAR    Byte
v            VAR    Byte
```

```

r          VAR    Byte
v2         VAR    Byte
v3         VAR    Byte
n          VAR    Word          'Δ línea cambiada

' -----[ Inicialización ]-----
CS          PIN    0
CLK         PIN    1
DataOutput  PIN    2

```

Modifique la subrutina **DAC**: del Programa 4.1 como sigue:

```

DAC:
  n = 166
  PWM 8, n, 40
RETURN

```

También modifique la subrutina **MOSTRAR**: para que solamente muestre la tensión, como se muestra abajo.

```

Display:
  DEBUG HOME, CR, "lectura DVM: ", DEC1 v
  DEBUG ".", DEC2 v2, " Volts"
RETURN

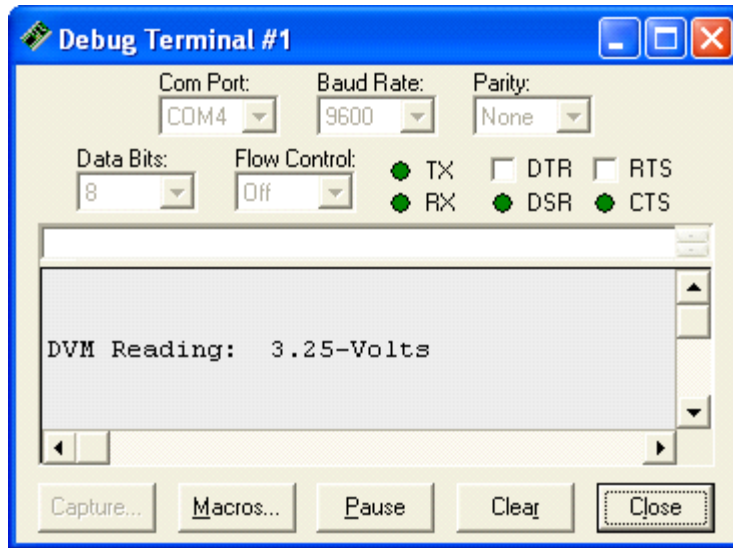
```

También actualice la primera línea del programa para referencia futura y guárdelo como P7\_1R0.bs2. Luego ejecute el programa y controle la salida.

## La Salida

Mire la Figura 7-6; ¡No está mal!





**Figura 7-6**  
Salida de Debug  
para el Programa  
7.1

7

Pruebe usar distintos valores de n.



**Alta impedancia de entrada:** Los pines de E/S del BASIC Stamp configurados como entradas, los terminales de entrada de un op-amp y los terminales Vin del ADC0831 tienen un aspecto muy importante en común, que los hace invisibles desde el punto de vista de otros circuitos. Su impedancia de entrada.

El término impedancia incluye el efecto de la resistencia y la capacitancia. Una impedancia de entrada alta implica que la resistencia es muy grande y la capacitancia muy pequeña.

La impedancia en los terminales de entrada de estos dispositivos es tan alta que la mayoría de los circuitos conectados a ellos las ven como si se tratara de circuitos abiertos. Por este motivo, puede conectar la salida de casi cualquier circuito a estas entradas.

Por otro lado, hemos visto como circuitos de relativamente bajas impedancias, tales como un resistor de 10kΩ o un LED pueden afectar drásticamente la salida de los circuitos conversores D/A.

Una ventaja de los conversores D/A que vienen en circuitos integrados es que están provistos de una salida con búfer, que se ve menos afectada por las cargas que conectemos.

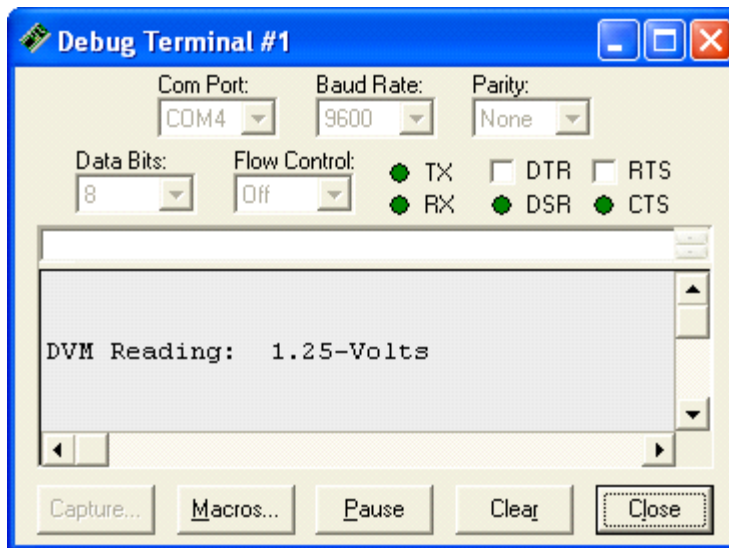
Importante: Este circuito debe ser aislado de otros. La impedancia de entrada del pin V(+) del ADC0831 es muy alta, lo suficiente como para que éste no sea detectado por el

circuito RC. Pero si intenta alimentar un LED con este circuito, sin usar un búfer, toda la carga del capacitor se escapará a través del LED y el resistor.

Intente conectar un resistor de 10k Ohm desde la salida del conversor D/A a Vss. La Figura 7-7 muestra lo que sucede cuando intenta enviar 3,25 Volts con esta carga conectada, al conversor D/A.

Dado que el pwm del D/A se repite sin pausa en el programa, el capacitor no se descarga completamente hasta 0 Volts.

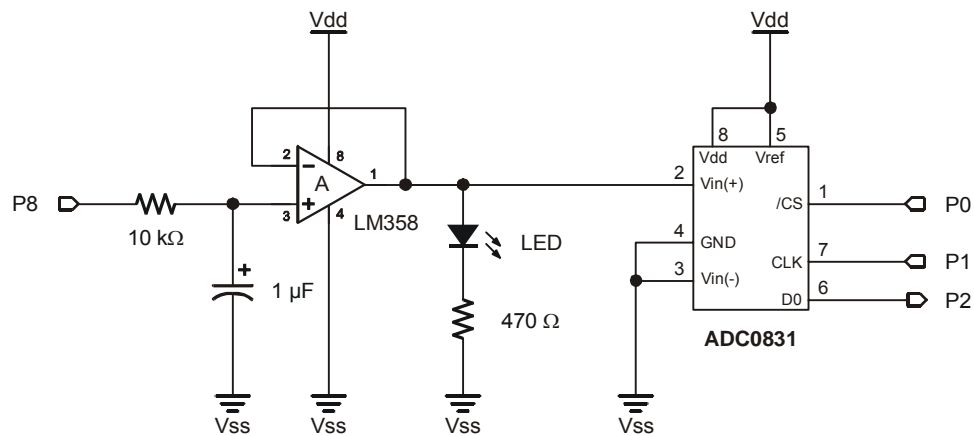
Si el comando **pwm** se ejecuta una sola vez, el capacitor se descargará rápidamente.



**Figura 7-7**  
Salida de Debug  
para el Programa  
7.1 con una carga  
directa de 10 kΩ..

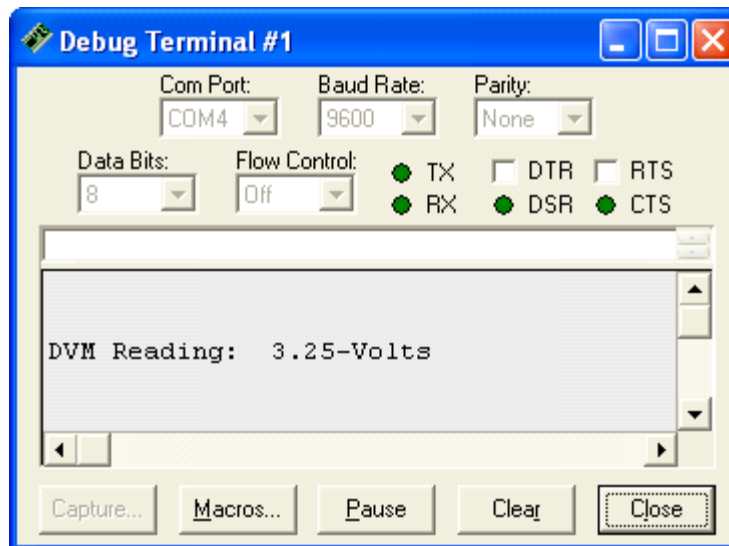
Como puede ver, un búfer es esencial para mantener la tensión estable. Puede usar el mismo seguidor de tensión que usamos en los Capítulos 2, 4 y 5 para alimentar la salida.

Coloque un seguidor de tensión entre la salida del conversor D/A y la entrada del LED como se muestra en la Figura 7-8, luego ejecute el Programa 7.1 nuevamente.



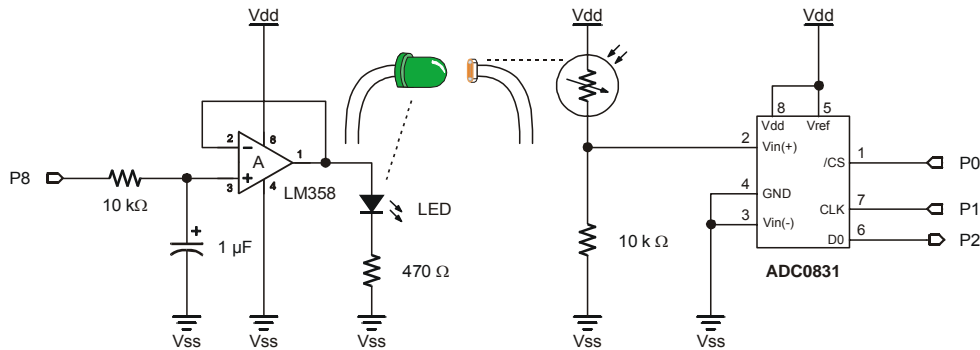
**Figura 7-8:** DAC PWM con búfer Op-amp y un LED en serie con un resistor como carga.

La Figura 7-9 indica que el búfer está cumpliendo su trabajo.



**Figura 7-9**  
Salida de Debug  
para el Programa  
7.1 con un LED  
conectado a través  
de un búfer.

Luego, agregue un fotorresistor con un resistor de 10 k Ohm en serie como se muestra en la Figura 7-10. La punta positiva de su “Stamp voltímetro” debería ser conectada al nodo donde el fotorresistor se conecta al resistor de 10 kW. Esto conforma una salida de divisor de tensión y a medida que la resistencia del fotorresistor varía, lo mismo hará la tensión a la salida del divisor de tensión. Además, apunte el encapsulado del LED amarillo directamente a la cara del fotorresistor como se muestra.



**Figura 7-10:** Transmisión de una Señal Óptica.

Un PWM fija la tensión de salida del circuito RC, que alimenta un LED a través de un seguidor de tensión.

La luz del LED es transmitida al fotorresistor. Apunte un dispositivo hacia el otro. El voltímetro del Capítulo 3 sensa los cambios en la salida del divisor de tensión.

Intentemos modificar el brillo del LED y veamos lo que sucede a la salida del divisor de tensión. Modifique las subrutinas principal:, **MOSTRAR:** y **DAC:** como se muestra abajo. Luego guarde el programa como PL7\_1R1.bs2 y ejecútelo.

```
' Analógico y Digital Básicos - PL7 1R1.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

' -----[ Declaraciones ]-----
adcBits    VAR    Byte
v          VAR    Byte
r          VAR    Byte
```

```

v2          VAR      Byte
v3          VAR      Byte
n           VAR      Word

' -----[ Inicialización ]-----
CS          PIN      0
CLK         PIN      1
DataOutput  PIN      2

DEBUG CLS    'Inicia ventana debug

' -----[ Rutina Principal ]-----
DO
  FOR n = 82 TO 199 STEP 9
    GOSUB DAC
    GOSUB ADC Data
    GOSUB Calc Volts
    GOSUB Display
    PAUSE 4000
  NEXT
LOOP

' -----[ Subrutinas ]-----
DAC:
  PWM 8, n, 40
RETURN

ADC Data:
  HIGH CS
  LOW CS
  LOW CLK
  PULSOUT CLK, 210
  SHIFTIN DataOutput,CLK,MSBPOST,[adcBits\8]
RETURN

Calc_Volts:
  v = 5 * adcBits / 255
  r = 5 * adcBits // 255
  v2 = 100 * R / 255
  v3 = 100 * R // 255
  v3 = 10 * v3 / 255
  IF (v3 >= 5) THEN v2 = v2 + 1
  IF (v2 >= 100) THEN
    v = v + 1
    v2 = 0
  ENDIF
RETURN

Display:
  DEBUG HOME, "Valor decimal del DAC: ", DEC3 n
  DEBUG CR, CR, "Lectura DVM: ", DEC1 v

```

```
DEBUG ".", DEC2 v2, "-Volts", CR, CR
RETURN
```

Como puede ver, la salida del divisor de tensión varía a medida que el LED modifica su brillo. Para obtener el barrido más amplio en la medición de tensión, apague las luces de la habitación. El circuito debería funcionar bien bajo condiciones normales de iluminación. Si el circuito está expuesto al sol directo, las diferencias en la tensión de salida podrían ser muy pequeñas para poder medirlas.

La ecuación del divisor de tensión puede usarse para determinar la resistencia del fotorresistor. Del Capítulo 3, la ecuación del divisor de tensión es:

$$V_{\text{Output}} = V_{\text{Input}} \times \frac{R_2}{R_1 + R_2}$$

$R_2$  es un valor de resistor conocido y  $R_1$  es el valor de resistencia desconocido del fotorresistor.  $V_{\text{entrada}}$  es 5 Volts y  $V_{\text{salida}}$  debería tomarse de la ventana Debug. Se necesita algo de álgebra para despejar la ecuación y llegar al resultado final:

$$R_1 = \left( R_2 \times \frac{V_{\text{Input}}}{V_{\text{Output}}} \right) - R_2$$

Este método usa un resistor variable como sensor. Claramente puede detectar diferentes niveles de luz y el ADC0831 se encarga de convertir la información analógica en digital. En el próximo Capítulo, introduciremos otro método para medir resistencias variables, que usa muchísimos menos recursos de hardware y código.

### ¿Qué aprendí?

En las oraciones de abajo, inserte las palabras apropiadas de la lista de la izquierda.

total		El BASIC Stamp tiene una función interna para realizar conversiones D/A que usa modulación por ancho de pulso. La técnica consiste de una serie de pulsos que cargan el capacitor de un circuito _____ a la tensión deseada. Un parámetro del PWM es el trabajo, que es diferente al ciclo de trabajo, cuyo objetivo es controlar la tensión a la que se cargará el capacitor. Trabajo se refiere a una serie de pulsos, en lugar de a un único ciclo. Se calcula como la cantidad de tiempo que la señal está en estado _____ dividida por el tiempo _____ del tren de pulsos.
analógica		
RC		
búfer		
divisor de tensión	de	Un _____ es esencial para este circuito cuando se alimenta una carga. Sin éste, el _____ perdería rápidamente su carga. Un capacitor también pierde carga debido a la corriente de fuga, así que el PWM debería repetirse periódicamente para actualizar la tensión de salida.
alto		
capacitor		Un fotorresistor es un dispositivo cuya resistencia varía con la intensidad de la luz que incide sobre su superficie. El fotorresistor o cualquier resistor cuyo valor cambia de acuerdo a un parámetro analógico puede ser usado en un _____ originando una salida de tensión variable. Esta información puede ser interpretada usando un ADC0831 y el BASIC Stamp para medir la tensión de salida _____.

### **Preguntas**

1. ¿Se puede mejorar la resolución del conversor D/A con PWM modificando los valores del resistor y el capacitor? Explique su respuesta.
2. Dado el comando: pwm 14,51,50, ¿aproximadamente cuántos milisegundos durará su ejecución? ¿Durante cuántos milisegundos la señal estará en estado alto?
3. ¿Por qué el capacitor pierde su carga si se le conecta un elemento resistivo a la salida del conversor D/A con PWM?
4. Si desea obtener una salida de 2,5 Volt, ¿cuánto deberá valer el parámetro trabajo? Asuma que está usando el comando pwm del Programa 7.1. Explique su razonamiento.
5. Si arma el DAC con un capacitor de 10uF y un resistor de 47 KOhms, ¿cuánto valdría el parámetro ciclos?
6. Si la tensión medida en un divisor de tensión con un fotorresistor como el de la Figura 7-10 es 2,3 Volts, ¿cuál es el valor del fotorresistor?

### **Desafío**

1. Escriba un programa que controle la tensión en 2 conversores D/A separados que mantengan diferentes niveles de tensión.
2. Diseñe y construya el circuito para el programa del Desafío 1. Necesitará usar el otro op-amp del LM358. Vea el diagrama de la Figura 1-5.
3. Programe la salida de los dos conversores D/A para que el LED de carga del canal 1 aumente gradualmente su brillo a medida que el LED del canal 2 lo reduce.
4. Escriba un programa que muestre el valor de resistencia medido en el fotorresistor de la Figura 7-8. Pista: Aplique la ecuación de divisor de tensión y use el programa del voltímetro que realizó anteriormente. Necesitará modificar las subrutinas `Calc_Volts` y `Mostrar`.



### **¿Por qué aprendí esto?**

Usar un circuito RC simple con un búfer, es una forma económica de obtener un conversor D/A de 8-bits. Además es muy fácil de programar y la precisión es mucho más cercana a la de un conversor D/A en circuito integrado que la de la red resistiva en escalera con resistores de 10% de tolerancia.

Hay muchos casos en los que un fenómeno analógico modifica la resistencia de un dispositivo. El divisor de tensión puede ser un medio eficaz para registrar estos cambios de resistencia. La tensión medida a la salida del divisor de tensión puede darnos información no sólo del valor de la resistencia, sino además del fenómeno analógico que provoca la variación de ésta.

### **¿Cómo puedo aplicarlo?**

La óptica tiene muchas aplicaciones en la industria donde se controlan y miden niveles de luz. En la industria de las comunicaciones, se emplea en los cables de fibras ópticas usados para enviar las conversaciones telefónicas a través de grandes distancias. En la robótica, sensor la intensidad de la luz es un aspecto importante para controlar el movimiento de las partes mecánicas del robot.

La aplicación que podría resultarle más familiar es el control remoto de su televisor. Cuando apunta el control remoto hacia su televisor y presiona un botón para cambiar un canal, la información se envía empleando luz, siendo recibida por un sensor en el frente del TV.

Muchos procesos industriales necesitan de cierta realimentación y miden el nivel de la iluminación para mantenerla dentro de un rango específico. En fotografía, muchas cámaras dependen de mediciones analógicas tomadas por un fotómetro (medidor de luz). Estas mediciones pueden usarse para controlar la apertura del obturador cuando se toma una fotografía. La apertura es lo que determina cuánto se abre el obturador para dejar entrar la luz dentro de la cámara. En el próximo capítulo construiremos un medidor de intensidad de la luz.



## Capítulo 8: Fotómetro con constantes de Tiempo RC

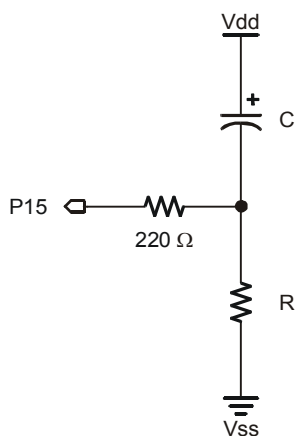
En este Capítulo usaremos los conceptos aprendidos en los anteriores para desarrollar un fotómetro. Esta es la lista de especificaciones del fotómetro:

- El fotómetro emitirá un beep cada medio segundo.
- Cuando aumente el nivel de luz de la habitación, el beep subirá de tono.
- Cuando disminuya el nivel de luz, el beep bajará de tono.

Este Capítulo introduce un modo de usar un pin de E/S del BASIC Stamp para medir una resistencia o capacitancia variable. Muchos sensores, además del fotorresistor, varían su resistencia o su capacitancia. Así, una forma simple y directa de medir estas cantidades, puede hacer que conectar estos sensores sea muchísimo más fácil.

8

En este Capítulo comenzaremos construyendo nuestro fotómetro, luego estudiaremos de cerca cómo funciona esta técnica de medición, para ver como puede ser usada para controlar una resistencia o capacitancia variable. La Figura 8-1 muestra un circuito CR genérico que puede ser usado para tomar mediciones de la constante de tiempo RC.



**Figura 8-1**  
Circuito CR

*Para medir  
valores de C o R.*

En el Capítulo 7, se demostró la cantidad de tiempo que tarda en cargarse un capacitor en un circuito RC con un escalón. El mismo principio se aplica aquí. Se aplicará un escalón al circuito y se medirá la cantidad de tiempo que tarda el capacitor en cargarse.

El BASIC Stamp debe ser programado para poner a P15 (el pin de E/S que usaremos) en estado alto durante unos milisegundos para que la caída de tensión sobre el capacitor se aproxime a 0 Volts. La razón por la que la caída de tensión sobre el capacitor se aproxima a 0 Volts es debido a que Vdd (5 Volts) están conectados al terminal positivo del capacitor y el pin P15 envía 5 Volts al terminal negativo. Así, la diferencia de tensión entre los terminales del capacitor es 5-5 Volts = 0 Volts.

Cuando la caída de tensión sobre el capacitor es suficientemente cercana a 0 Volts, P15 pasa a funcionar como entrada. Inmediatamente, el BASIC Stamp comienza a contar en incrementos de 2 microsegundos. Tan pronto como P15 se vuelve una entrada, deja de tener una tensión de salida de 5 Volts. P15 se comporta como un circuito abierto desde el punto de vista del circuito CR.

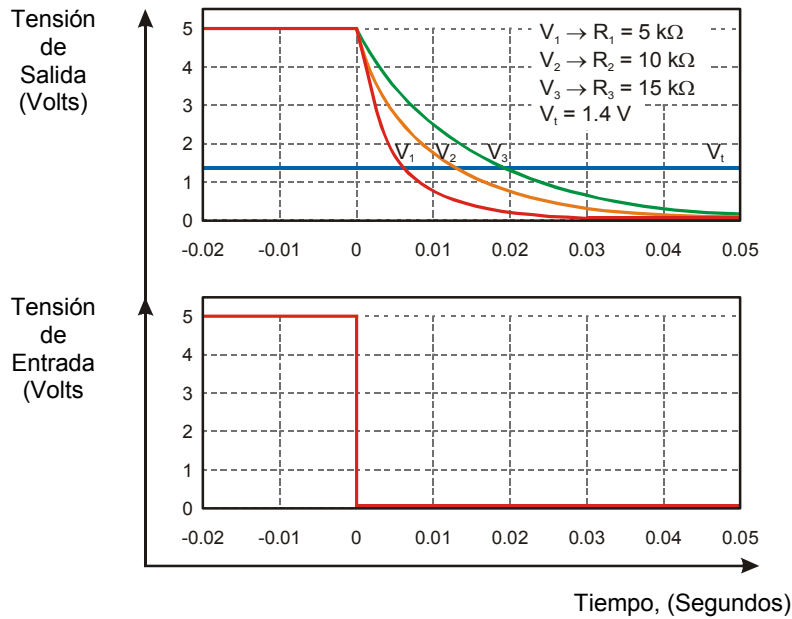
A partir de ese cambio, el circuito es diferente. Ahora se trata de un resistor y un capacitor en serie. El capacitor inicia descargado, pero el circuito completo tiene una caída de 5 Volts sobre él ( $V_{dd} - V_{ss}$ ), así que el capacitor comienza a cargarse. A medida que el capacitor se carga, la tensión en P15 comienza a caer. La tensión en P15 cae hasta que cruza el umbral de tensión del pin de E/S del BASIC Stamp. Se necesita una cierta cantidad de tiempo para que la tensión en P15 caiga desde 5 Volts, a los 1,4 Volts del umbral de tensión.

El BASIC Stamp puede ser usado para medir esta cantidad de tiempo y la medición puede usarse para determinar un valor desconocido de resistencia o capacitancia.

La ecuación para la tensión en P15 es dada por:

$$V_{P15} = V_{dd} \times e^{\frac{-t}{R \times C}}$$

Esta ecuación es usada en muchos campos y se suele llamar ecuación de decrecimiento exponencial. En nuestro caso, es una ecuación de caída exponencial de tensión. La Figura 8-2 muestra la caída exponencial de tres circuitos RC, que difieren solamente en su resistencia. Cada una de las tensiones de salida, demora una cantidad de tiempo distinta para decaer de 5 Volts a 1,4 Volts.



**Figura 8-2**  
Respuesta del  
Circuito RC

a un escalón de  
entrada  
negativo para  
tres valores de  
resistencia  
distintos. Note  
como cada  
salida demora  
distinta cantidad  
de tiempo, para  
decaer de 5 a  
1,4 Volts.

8

La diferencia en el tiempo que demora cada circuito en decaer, puede ser usada para determinar el valor de la resistencia si el valor de la capacidad permanece fijo. Del mismo modo, si se fija el valor de la resistencia pero varía la capacitancia, ésta puede determinarse midiendo el tiempo entre el escalón de tensión y el cruce por el umbral.

Un comando PBASIC llamado **RCTIME** puede ser usado para que el BASIC Stamp determine la cantidad de tiempo que tarda la salida del circuito RC en caer de 5 Volts a 1,4 Volts, que es el valor de  $t$  en la ecuación de decrecimiento exponencial. El valor de  $t$  que mediremos es  $t_{1,4\text{ Volts}}$ . El valor  $t_{1,4\text{ Volts}}$  es la cantidad de tiempo necesario para que la tensión en el pin de E/S P15 caiga de 5 Volts a 1,4 Volts.

Esta cantidad de tiempo es directamente proporcional a  $R$  multiplicado por  $C$ .  $R \times C$  recibe el nombre de constante de tiempo RC y a menudo se representa por la letra griega tau.

En el instante que la tensión de P15 pasa por 1,4 Volts, el valor de entrada de pin P15 pasa de 1 a 0 y el BASIC Stamp detiene la cuenta de intervalos de 2 us.

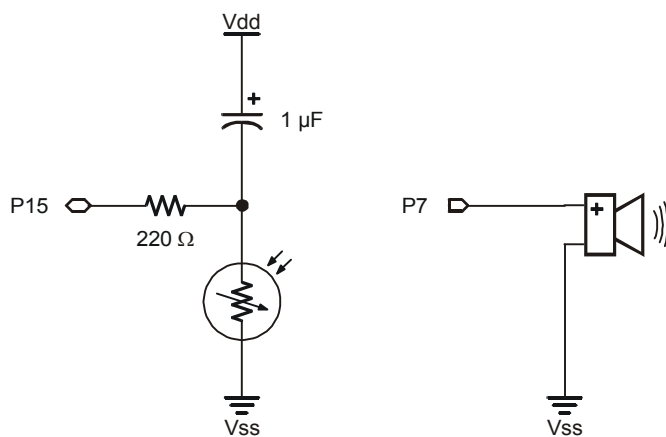
### **Componentes Requeridos**

Este experimento requiere los siguientes componentes:

- (1) Capacitor de poliéster de 0.1  $\mu\text{F}$
- (1) Capacitor electrolítico de 1  $\mu\text{F}$
- (1) Capacitor electrolítico de 10  $\mu\text{F}$
- (1) Parlante piezoeléctrico
- (1) Resistor de 220 Ohms
- (1) Fotorresistor

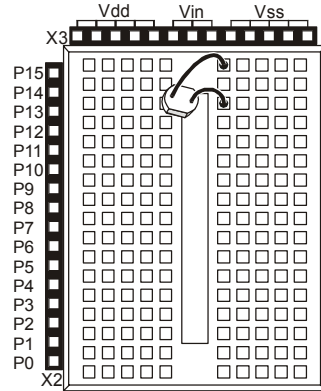
### **Constrúyalo**

Arme el circuito mostrado en la Figura 8-3. Note que el circuito de la izquierda usa un fotorresistor variable. El circuito de la derecha es el mismo que utilizamos en el Capítulo 5 para reproducir melodías.



**Figura 8-3**  
Circuito del Medidor de Luz Ambiente.

Deberá calibrar su circuito y su programa debido a variaciones en las condiciones de la iluminación. No apunte el fotorresistor directamente al sol. Usaremos la ranura central de la protoboard como colector de luz. Apunte el fotorresistor a la ranura como se muestra en la Figura 8-4



**Figura 8-4**  
Ubicación del Fotorresistor.

*Apunte el fotorresistor a la ranura central de la protoboard. De esta forma el fotorresistor detectará la luz reflejada en el plástico blanco. Esto hará que el fotorresistor sea más estable a cambios bruscos que pudieran producirse en la fuente de luz.*

El juego de instrucciones PBASIC tiene un comando llamado **RCTIME**, que mide el tiempo de descarga de un capacitor bajo las condiciones que explicamos anteriormente. El formato del comando **RCTIME** es:

**RCTIME pin, estado, resultado**

Como siempre, **pin** se refiere al pin de E/S del BASIC Stamp que usaremos. De acuerdo a la Figura 8-1, usaremos el pin P15. Ahora, ¿qué hay sobre **estado**? El parámetro estado es 1 cuando usamos un circuito RC como el mostrado en la Figura 8-1. Si usamos un circuito RC con el resistor arriba y el capacitor abajo, el estado inicial debería ser 0. El parámetro **resultado** es la cantidad de incrementos de 2 microsegundos que el BASIC Stamp contó durante la descarga. ¿Cómo funciona esto?

Primero el pin 15 debe ponerse en estado alto, para llevar el pin del capacitor a 5 Volts. Cuando el BASIC Stamp ejecuta el comando **RCTIME**, convierte al pin 15 en entrada. La tensión comienza a caer como se muestra en la Figura 8-2. El BASIC Stamp registra la cantidad de incrementos de 2 microsegundos desde el momento que el pin pasa a ser una entrada y la tensión del mismo cuando cruza el umbral de 1,4 Volts. La cantidad de incrementos de 2 microsegundos se almacena en la variable **resultado**.

Usaremos este segmento de código del Programa 8.1 para medir el tiempo RC. Primero esperaremos durante 100 milisegundos con el pin P15 en estado alto. Esto le permite al capacitor cargarse.

```
HIGH 15
PAUSE 100
```

Luego **RCTIME** se encarga de medir el tiempo y almacenarlo en la variable.

```
RCTIME 15, 1, light
```

### **Prográmelo**

Ingrese el Programa 8.1 en su editor de BASIC Stamp y descarguelo en el BASIC Stamp.

```
' Analógico y Digital Básicos - PL8 1R0.bs2
' Programa 8.1 Revision 0.
' {$STAMP BS2}
' {$PBASIC 2.5}

light          VAR      Word
dischargeTime  VAR      Word
sound          VAR      Word

a              CON      40

DO
  HIGH 15
  PAUSE 100
  RCTIME 15, 1, light

  dischargeTime = light * 2
  sound = (65535 - light) / a

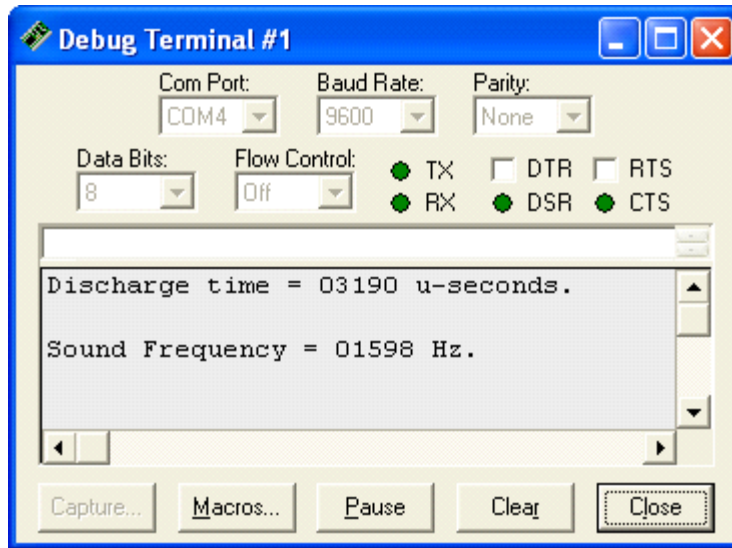
  FREQOUT 7, 500, sound

  DEBUG HOME
  DEBUG "Tiempo de descarga = ", DEC5 dischargeTime, " u-segundos."
  DEBUG CR, CR, "Frecuencia de sonido = ", DEC5 sound, " Hz."
  PAUSE 500
LOOP
```

### **La Salida**

El parlante piezoeléctrico debería emitir un beep cada aproximadamente medio segundo. Si pone su mano sobre el circuito para hacerle sombra, el tono del beep será más bajo. Si enfrenta la protoboard directamente a la luz, el tono será más agudo. La Figura 8-5 muestra un ejemplo de salida para un ambiente muy iluminado.

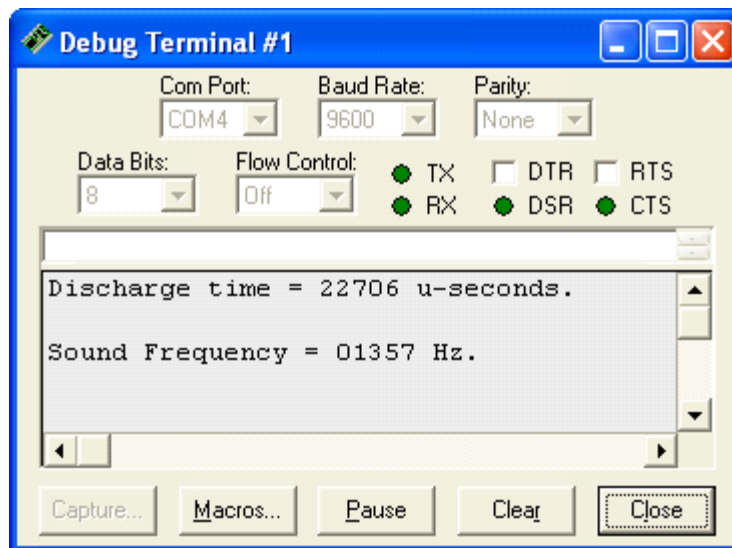




**Figura 8-5**  
Salida de Debug  
para un ambiente  
muy iluminado sin  
luz directa del sol

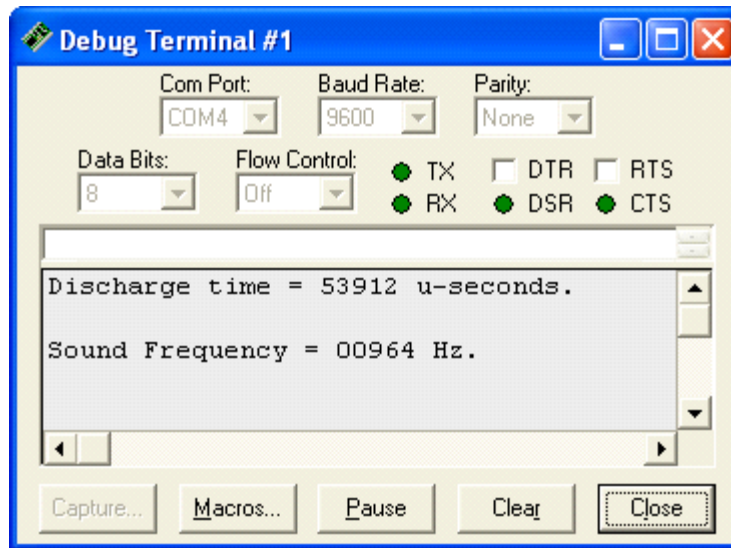
8

La Figura 8-6 muestra la salida cuando se realizó sombra con un libro sobre la protoboard. El tono emitido por el parlante piezoeléctrico fue notoriamente más grave.



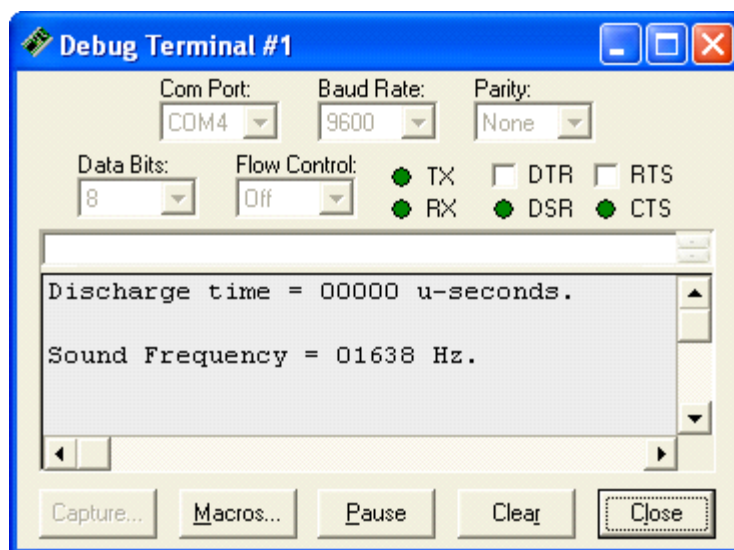
**Figura 8-6**  
Salida de Debug  
en la sombra

Figura 8-7 muestra la salida en un ambiente con iluminación escasa.



**Figura 8-7**  
Salida de  
Debug en  
ambiente con  
poca luz

La Figura 8-8 muestra la salida para oscuridad total. Se produjo un desborde de la variable (overflow). El tiempo de descarga superó los 65.536 microsegundos. En consecuencia, la variable luz se reinicia en cero. Dado que `tiempo_desc` y `luz` valen cero, la variable `sonido` alcanza su valor máximo. Esto es debido a que `sonido` se programó así:  $\text{sonido} = (65535 - \text{luz}) / 40$ . Luz pasó a valer cero por el desborde de la variable, así que sonido se calculó como  $(65535 - 0) / 40 = 1638$ .



**Figura 8-8**  
Salida de Debug  
luego de que  
alguien apagara  
las luces.

8

Bajo condiciones de mucha iluminación, la resolución es muy pobre. Puede reemplazar el capacitor de 1.0 uF por uno de 10.0 uF para aumentar la resolución a intensidades mayores de luz. Para exposición directa al sol, ayudaría poner algún tipo de filtro que haga un poco de sombra sobre el fotorresistor. Si trabajará en condiciones de luz escasa, le conviene usar un capacitor de 0.1 uF.

Se usan diferentes capacitores debido a que el rango de resistencias del fotorresistor varía para distintos niveles de iluminación. Con escasa iluminación, el fotorresistor tiene un rango de valores de resistencia mucho más elevado que para una exposición directa a la luz del sol.

Como se trató anteriormente, el tiempo de descarga depende de la constante de tiempo RC time,  $R \times C$ . Si el valor de R crece mucho, usando un menor valor de C hace que el producto  $R \times C$  quede dentro de un rango aceptable. Del mismo modo, si R baja mucho, al usar un valor de C mayor se mantiene el rango de  $R \times C$ .

### Matemática involucrada

Las mediciones de tiempo RC junto con la ecuación de decaimiento exponencial pueden ser una herramienta poderosa para determinar una resistencia o capacitancia desconocida.

$$V_{P15} = V_{dd} \times e^{\frac{-t}{R \times C}}$$

Dado que usaremos PBASIC para hacer que el BASIC Stamp nos diga el tiempo medido t, usaremos la información que nos muestra la Tabla 8.1.

Table 8-1: Valores conocidos en el instante en que $V_{P15}$ cruza el umbral de tensión	
Valores	Comentarios
Vdd = 5.0 volts	Condición inicial de la tensión en P15.
VP15 = 1.4 volts	Condición final de la tensión en P15.
$e \approx 2.718$	El valor de e es una constante que se encuentra en muchos libros.
C = a calcular	Si se conoce el valor de la resistencia (R) se puede calcular la capacitancia (C).
t = conocido	Tiempo obtenido por el BASIC Stamp en incrementos de 2 microsegundos.
R = a calcular si	Si se conoce el valor de la capacitancia (C) se puede calcular la resistencia (R).

Se puede aplicar álgebra a la ecuación de decaimiento exponencial para despejar R o C. Primero, aplique el logaritmo natural a ambos lados de la ecuación. De esta forma se elimina e.

$$\ln(V_{P15}) = \ln\left(V_{dd} \times e^{\frac{-t}{R \times C}}\right) \Rightarrow \ln(V_{P15}) = \ln(V_{dd}) - \frac{t}{R \times C}$$

Redistribuyendo los términos queda:

$$\frac{t}{R \times C} = \ln(V_{dd}) - \ln(V_{P15})$$

Usando propiedades de logaritmos, se simplifica a:

$$\frac{t}{R \times C} = \ln\left(\frac{V_{dd}}{V_{P15}}\right)$$

Redistribuyendo los términos nos queda una ecuación con R y C en función de t:

$$R \times C = \frac{t}{\ln\left(\frac{V_{dd}}{V_{P15}}\right)}$$

Si R es el valor a determinar, divida ambos miembros por C. Si desea calcular C divida ambos miembros por R. Las dos ecuaciones resultantes son útiles para determinar R si C es conocido, o viceversa.

$$R = \frac{t}{C \times \ln\left(\frac{V_{dd}}{V_{P15}}\right)} \quad \text{or} \quad C = \frac{t}{R \times \ln\left(\frac{V_{dd}}{V_{P15}}\right)}$$

8

Pruebe con diferentes combinaciones de resistores y capacitores del Kit de componentes. Recuerde que los valores de los capacitores normalmente pueden variar un 20% de su valor nominal (valor impreso). Esto introducirá error en los valores medidos. Sin embargo, si se dispone de un resistor con una tolerancia muy baja, un 1% por ejemplo, la medición de una capacidad puede ser muy precisa. Una vez que se calibró el valor del capacitor, luego es posible tomar mediciones de resistencias más precisas con el circuito RC.

### ¿Qué aprendí?

En las oraciones de abajo, inserte las palabras apropiadas de la lista de la izquierda.

umbral	Los sensores analógicos que varían su resistencia o su capacidad son empleados para medir muchas magnitudes físicas. Un circuito _____ puede ser usado en conjunto con un pin del BASIC Stamp para medir resistencia o capacidad.
capacitancia	
capacitor	El pin del _____ se pone a 5 Volts, luego se deja que circule corriente y se mide el tiempo que tarda en caer la tensión desde 5 Volts hasta la tensión de _____ del pin de entrada del BASIC Stamp.
RC	La ecuación de la caída _____ de la tensión puede usarse para determinar el valor de resistencia o capacitancia, si la otra magnitud es conocida.
exponencial	
descarga	Un circuito RC donde la resistencia varía puede ser calibrado a un cierto rango de valores de tiempo de _____, cambiando el valor del capacitor. Este método puede usarse debido a que la resistencia multiplicada por la _____ está relacionada logarítmicamente con la velocidad de decaimiento de la tensión.

**Preguntas**

1. Dado el circuito de la Figura 8-3, ¿Cuánto tiempo durará la descarga si el valor del fotorresistor es  $47\text{ k}\Omega$ ?
2. Para el tiempo de descarga de la Pregunta1, ¿esperaría que se produjera un desbordamiento en alguna de las variables del Programa 8.1? Si es así, ¿cuál o cuáles? Explique su respuesta.
3. Si las condiciones de iluminación son tan bajas que ninguno de los capacitores del Kit funciona, tal vez se necesitaría comprar uno. ¿Qué valores de capacidad buscaría? Explique su respuesta.
4. Para obtener una mejor resolución en el extremo superior de la escala del nivel de iluminación, también podría reducir el valor de  $a$  en el Programa 8.1. ¿Cómo afectaría esto a la lectura en la ventana debug y al sonido del parlante piezoeléctrico?

**8****Desafío**

1. Diseñe un circuito que controle el estado de dos fotorresistores. Asuma que mostrará la información sobre la intensidad de la luz en la ventana debug y que no usará un parlante piezoeléctrico.
2. Programe el BASIC Stamp para que controle el nivel de luz de ambos fotorresistores y muestre la información obtenida en la ventana debug.
3. Sostenga papeles de distintos colores delante de los fotorresistores; blanco y negro serían los mejores. Escriba un programa que distinga qué fotorresistor detecta un papel blanco y cuál uno negro.
4. Asegúrese de que el programa del Desafío 3 no sea afectado por el nivel de luz ambiente de la habitación. Pista: Necesitará comparar las intensidades RELATIVAS de la luz medidas por ambos fotorresistores, en lugar de realizar comparaciones sobre valores fijos determinados experimentalmente sobre valores de iluminación anteriores.

### **¿Por qué aprendí esto?**

Muy interesante, ¿no? Usamos la luz como entrada y el sonido como salida. Medimos la intensidad de la luz y generamos un sonido cuyo tono dependía de la intensidad de luz medida.

Este Capítulo muestra como pueden integrarse los conceptos introducidos en todos los capítulos para diseñar un dispositivo electrónico real. También demuestra que los capacitores no tienen exactamente el valor que indican sus encapsulados, debido a las tolerancias. Ahora sabe cómo calcular el valor de un capacitor.

### **¿Cómo puedo aplicarlo?**

Mientras aprendió todo esto, cubrimos varios temas de diseño de circuitos y técnicas de programación. También tuvimos experiencias con fenómenos analógicos. También desarrollamos cómo conectar electrónicamente, procesar y en general, realizar tareas con un dispositivo digital como el microcontrolador BASIC Stamp. Abajo hay una lista de los conceptos con los que trabajó en los experimentos.

Capítulo 1: Datos en serie y paralelo y comunicación sincrónica y asincrónica.

Capítulo 2: Comparador, búfer y umbral de tensión.

Capítulo 3: Conversión analógica a digital, divisor de tensión y mediciones de tensión normalizadas.

Capítulo 4: Conversión digital a analógica, red resistiva en escalera y volumen y frecuencia de un sonido.

Capítulo 5: El osciloscopio y las señales que varían en el tiempo.

Capítulo 6: El temporizador 555, frecuencia y medición de la frecuencia de distintas señales.

Capítulo 7: Acoplamiento óptico y control del ancho de pulso para conversiones D/A.

Capítulo 8: Construcción de un fotómetro y conversión A/D usando constantes de tiempo RC.

Estos 8 experimentos le dan una primera exposición al análisis de fenómenos analógicos, las bases de los circuitos electrónicos y los principios del proceso digital de información obtenida de mediciones analógicas, usando un dispositivo electrónico digital.

Si usted es un aficionado a la electrónica, habrá aprendido algunos trucos para aplicar en sus diseños. Si es un estudiante de alguna carrera de electrónica, sin duda volverá a



encontrar estos temas en el futuro, cubiertos con mucho más detalle. El BASIC Stamp puede ser extremadamente útil para gran variedad de proyectos en las ciencias naturales, así como también en la ingeniería.

La mayoría de los componentes usados en estos experimentos, son baratos y muy comunes. Hay gran disponibilidad de componentes que pueden conectarse al BASIC Stamp para concretar infinidad de tareas. Algunos ejemplo son: teclados digitales, pantallas de cristal líquido (LCD) y servos.

Si completó estos experimentos, así como también los de “¿Qué es un microcontrolador?”, unir los materiales y los conocimientos de ambos libros debería resultarle fácil. Además, imagine lo que podría diseñar y construir usando un teclado digital, un LCD y los conceptos de interfaz analógica aprendidos. Podría montar un control automatizado de procesos de una fábrica, un sensor de altitud y temperatura para su cohete, o una estación de monitoreo del clima.



## Apéndice A: Listado de Componentes

Todos los componentes (página siguiente) usados en los experimentos de Analógico y Digital Básicos se pueden adquirir en cualquier negocio de componentes electrónicos. Los que deseen comprar el kit completo en Parallax también pueden hacerlo. Para completar este libro necesita tres cosas: (1) un módulo BASIC Stamp II (se vende suelto o con la Plaqueta de Educación - Full Kit); (2) una Plaqueta de Educación (se vende suelta o con la Plaqueta de Educación - Full Kit); y 3) el Kit de componentes de Analógico y Digital Básicos.

### Kits de la Plaqueta de Educación

Los manuales de Stamps en Clase usan el módulo BASIC Stamp y la Plaqueta de Educación como núcleo. Estos componentes se pueden comprar por separado.

Board of Education Full Kit (#28102)		
Código Parallax #	Descripción	Cantidad
28150	Plaqueta de Educación	1
800-00016	Cables de interconexión	10
BS2-IC	Modulo BASIC Stamp II	1
750-00008	Fuente de alimentación 300 mA 9 VCC	1
800-00003	Cable Serial	1



### Los experimentos usan el Kit de Componentes de Analógico y Digital Básicos (#28128)

Al igual que con el resto de los libros de Stamps en Clase, necesita una Plaqueta de Educación con un BASIC Stamp y el Kit de componentes. El contenido del Kit se lista abajo. Estos componentes se pueden comprar en Parallax pero también se consiguen en comercios de electrónica.

Kit de Componentes de Analógico y Digital Básicos: (#28128)		
Parallax Code#	Description	Quantity
150-01011	resistores 100 Ohm 5%	3
150-01020	resistores 1K $\Omega$ 5%	8
150-01030	resistores 10K $\Omega$ 5%	3
150-02020	resistores 2K $\Omega$ 5%	10
150-02210	resistores 220 $\Omega$ 5%	2
150-02710	resistores 270 $\Omega$ 5%	2
150-04710	resistores 470 $\Omega$ 5%	4
152-01040	potenciómetros 100K $\Omega$	2
200-01040	Capacitor radial 0.1 $\mu$ F	1
201-01050	Capacitor electrolítico 1.0 $\mu$ F	1
201-01062	Capacitor electrolítico 10.0 $\mu$ F	1
350-00006	LEDs, rojos	3
350-00007	LED, amarillo	1
350-00009	fotoresistores	2
400-00002	pulsadores	2
602-00015	IC Op - amp LM 358	1
604-00009	IC temporizador 555	1
800-00016	Cables de interconexión (paquetes de 10)	2
900-00001	Parlante piezoeléctrico	2
ADCO831	convertor A/D ADC 0831 8-bit	1

## Apéndice B: Código de Color de Resistores

La mayoría de los tipos comunes de resistores tienen bandas de colores que indican su valor. Los resistores que usaremos en esta serie de experimentos son normalmente “1/4 Watt, de carbón, con una tolerancia del 5%”. Si se fija en la secuencia de bandas, observará que una de las bandas (en un extremo) es dorada. Ésta es la cuarta banda y el color dorado significa que tiene una tolerancia del 5%.

El código de colores del resistor es un estándar industrial para la identificación de valores de resistores. Cada banda de color representa un número y el orden en que se encuentran tiene un significado. Las dos primeras bandas indican un número. La tercera banda de color indica el multiplicador, o en otras palabras, la cantidad de ceros. La cuarta banda indica la tolerancia del resistor +/- 5, 10 o 20 %.

Color	1er Dígito	2do Dígito	Multiplicador	Tolerancia
negro	0	0	1	
marrón	1	1	10	
rojo	2	2	100	
naranja	3	3	1,000	
amarillo	4	4	10,000	
verde	5	5	100,000	
azul	6	6	1,000,000	
violeta	7	7	10,000,000	
gris	8	8	100,000,000	
blanco	9	9	1,000,000,000	
dorado				5%
plateado				10%
sin color				20%

Un resistor tiene las siguientes bandas de color:

Banda 1.= Rojo

Banda 2.= Violeta

Banda 3.= Amarillo

Banda 4.= Dorado

Mirando la lista de arriba vemos que el rojo vale 2.

Así que escribimos: “2”.

Violeta tiene un valor de 7.

Así que escribimos: “27”

Amarillo tiene un valor de 4.

Así que escribimos: “27 y cuatro ceros” o “270,000”.

Este resistor tiene un valor de 270,000 Ohms (o 270k) y una tolerancia del 5%.

## Índice Alfabético

---

### - A -

algoritmo, 61  
aliasing, 119  
alta impedancia de entrada, 137  
amplificador operacional LM358, 5  
analógico, 1  
ancho de pulso, 102

### - B -

BASIC Stamp, 1  
bucle FOR NEXT, 36  
buffer, 85

### - C -

ciclo de trabajo, 102  
circuito integrado, 45  
comparador, 15  
comunicación, 31  
    asincrónica, 31  
    paralela, 32  
    serie, 32  
    sincrónica, 32

conversión A/D, 71  
conversión D/A, 71  
conversor A/D, 45  
corriente de fuga, 130  
corriente/Amperes, 4

### - D -

diagrama de circuito, 3  
digital, 1  
direccionado, 80  
divisor de tensión, 44

### - E -

EEPROM, 26

### - F -

frecuencia, 94

### - L -

LED, 3  
ley de Ohm, 4, 88

### - M -

multivibrador astable, 115

### - N -

números binarios, 19

### - O -

Ohms, 3  
onda cuadrada, 93, 101  
onda triangular, 93  
op-amp, 5  
osciloscopio, 93

### - P -

paralelo, 31  
polarizar, 47  
potenciómetro, 4  
pulsador, 20

### - R -

RAM, 26  
rango continuo, 44  
resistencia/Ohm, 4  
resistor, 3  
resistores en serie, 44

- S -

señales de control binarias, 46  
sincrónica, 31

- T -

tamaño de memoria, 25

tensión, 2

- V -

voltímetro digital, 43



