#230 - 2250 Boundary Road,
Burnaby, BC, V5M 3Z3, Canada.
Web site: http://rmv.com
Email: customer@rmv.com

ITC232A
Manual de Instrucciones

LEA ESTO CON ATENCIÓN:

RMV ELECTRONICS INC. no asume ninguna responsabilidad legal ó de ninguna otra índole por el uso de el ó los producto(s) aquí descriptos ni otorga ningún tipo de licencia sobre la(s) patentes ó derechos propietarios referentes a estos productos. Los productos de RMV ELECTRONICS INC. no deberán ser utilizados para aplicaciones militares ó biomédicas en las cuales la salud ó la vida humana están en riesgo sin la autorización, por escrito, de RMV ELECTRONICS INC.

INTRODUCCIÓN:

El circuito integrado ITC232-A contiene una interfase serie-paralelo "inteligente". Además de esa interfase propiamente dicha, este circuito integrado contiene una serie de funciones pre-programadas para adquisición de datos y control por computadora.

El ITC232-A permite el fácil acceso, desde una terminal ó computadora equipada con un puerto serie (EIA RS-232) a 32 líneas de entrada (input) ó salida (output) agrupadas en 5 puertos que pueden ser leídos ó escritos utilizando simple comandos en formato ASCII. Esto permite controlar este integrado desde un programa escrito por el usuario ó desde programas que emulan terminales como Procomm©, MAC240©, ó el incluido en Windows©, lo que resulta muy práctico en el desarrollo de sistemas.

El ITC232-A es independiente del "hardware" utilizado; cualquier terminal ó computadora con un puerto serie RS232 puede ser utilizado. La conección requiere 3 cables y puede ser operada a cualquier velocidad standard entre 300 y 115200 Baudios. Los únicos componentes externos necesarios son los "drivers" para crear los voltajes requeridos por el standard RS232 y un cristal de 3,6864 MHz.

Aparte de la interfase serie/paralelo, el ITC232-A brinda las siguientes funciones:

1. Ejecutar las transacciones en notación Decimal, Hexadecimal ó Binaria.

2. 2 patillas de interrupción, una en la transición de 1 a 0 (Low) y la otra de 0 a 1 (High).
3. Modulación de ancho de pulso de 10 a 10000 Hz, 0-100% en intervalos del 1%.
4. Velocidad de transmisión configurable a cualquier valor standard entre 300 y 115200 Baudios.
5. Cuatro canales de lectura directa de resistencia ó capacidad (sólo se requiere una resistencia y un condensador, ideal para uso con termistor).
6. Tres puertos lógicos para motores paso a paso que pueden funcionar en modo bifásico, monofásico ó de medio paso.
7. Un comando de repetición del comando previo que requiere solamente el envío del carácter "@".
8. Ayuda en pantalla ("?" envía un sumario de todos los comandos a la terminal).
9. Sumario, a demanda de la configuración presente en los puertos, la modulación de ancho de pulso ó los motores paso a paso.
10. Operación remota por via telefónica utilizando un modem en un integrado tal como el AD7911 en lugar del driver para el puerto RS232.

La flexibilidad y el fácil empleo del ITC232-A hacen de este integrado un componente ideal en toda aplicación que requiera la utilización de un computadora para control ó adquisición de datos.

CONSIDERACIONES GENERALES:

Los comandos del ITC232-A est n formados por las iniciales de las correspondientes instrucciones en inglés. Por lo tanto, a continuación incluimos un pequeño diccionario con algunas de las palabras inglesas utilizadas en este manual:

Port = Puerto
Serial = Serie
Width = Ancho
Pulse Width Modulation = Modulación de ancho de pulso, abreviamos como PWM.
Step = paso
Stepping motor ó stepper = motor de paso a paso
Enable = Habilitar
Carriage Return (CR) = carácter 13 del código ASCII
Line Feed (LF) = carácter 10 del código ASCII
Resistance ó Resistor = Resistencia
Capacitance = Capacidad ó Capacitancia
Configuration = Configuración
Results = Resultados
High = Alto ó 1 binario
Low = Bajo ó 0 binario
Driver = Circuito utilizado para llevar señales a los niveles de voltaje y corriente requeridos por otros circuitos (un típico caso de driver es el MAX232 que crea +9 y -9 Voltios para el puerto RS232 a partir de 0 y 1 binario).
Help = Ayuda
$ ó H precediendo un valor indica que el mismo est expresado en formato hexadecimal.

% ó B precediendo un valor significa que el mismo est expresado en formato binario.

Otros términos menos usados se traducen en el texto.

La palabra "Terminal" incluye computadoras y "periférico" se refiere a todo circuito que contenga un ITC232-A.

Como convención, los valores se ENVÍAN desde la terminal y RETORNAN del ITC232-A.

PARA COMENZAR:

La Figura 1 muestra el diagrama de las patillas del ITC232-A. La Figura 2 muestra un ejemplo de circuito básico para conectar un ITC232-A a una terminal. Un MAX232 ó un ITC232 se utiliza como driver en el ejemplo, pero puede ser substituido por drivers de menor costo tales como el par 1488/1489 si se cuenta con una fuente de voltaje positivo y negativo.

Parámetros para las comunicaciones:

Si la patilla BAUD = 1 seleccione 9600 Baudios en su programa. Si BAUD = 0, elija 300 Baudios. Los otros par metros son siempre N,8,1 (sin paridad, 8 bits, 1 bit de stop). Cuando utilice un programa comercial de comunicaciones, trabaje sin traducción de CR a CR/LF en la transmisión ya que aunque el carácter LF es ignorado por el ITC232-A igual toma cierto tiempo en ser enviado.
IMPORTANTE: Asegúrese de que la tecla "BACKSPACE" envíe un ASCII 8 para borrar un error. Algunos programas tales como Terminal en Windows© requieren otras combinaciones de teclas (Ctrl H en este caso) para enviar un ASCII 8.
Aplique +5V al circuito. El siguiente mensaje es enviado a la terminal:

Welcome to the ITC232-A
? or h for help
>
("Bienvenido al ITC232-A, ? ó h para obtener ayuda >")

El último carácter de este mensaje es un ASCII 7 (a continuación del ">") el que al ser impreso genera una señal de campanilla (carácter 7 ASCII) de corta duración.

COMANDOS Y RESULTADOS DEVUELTOS:

Generalidades:

Los comandos desde la computadora son siempre enviados en formato ASCII y deben ser siempre terminados con un CR (ASCII 13). El ITC232-A comienza a interpretar un comando al recibir el CR. Se pueden utilizar mayúsculas ó minúsculas indistintamente. Los espacios y los signos de puntuación son ignorados con la importante excepción del ";" que se utiliza para separar parámetros.

El envío de un > ó un ESC (ASCII 27) en cualquier punto de un comando lo interrumpe y genera un nuevo > quedando el ITC232-A a la espera de un próximo comando.
Un comando correcto es siempre respondido con un "OK". Un comando erróneo devuelve "?n" donde n es el código del error en formato hexadecimal (de $1 a $B). El ITC232-A no se "cuelga" como respuesta a un comando erróneo, siempre genera un mensaje de error.
El último carácter retornado a la terminal luego de un comando es siempre un ">". Esto permite la rápida localización del fin del mensaje para comenzar su interpretación desde dentro de un programa. Los casos en que ">" no está al final de un mensaje son: (1) Luego del mensaje de bienvenida al resetear el ITC232-A, (2) Luego del mensaje enviado cuando el ITC232-A contesta el teléfono (En estas 2 situaciones el último carácter es un ASCII 7 (Beep) y (3) en el caso de las interrupciones que se traducen en una "L" ó una "H" solamente.

Hay 2 tipos de comandos; los de tipo "procedimiento", que solamente ejecutan una acción y los de tipo "función" los que aparte de ejecutar ó no una acción retornan un valor a la terminal.

El ITC232-A se puede configurar para que el formato por defecto de los resultados retornados sea binario, decimal ó hexadecimal. En cualquiera de estos formatos el integrado añade un CR y un LF luego del OK y después del resultado para permitir una fácil visualización a la terminal. Existe un 4to formato llamado de "programa" en el que los resultados son enviados en formato decimal y sin ningún CR ó LF para su fácil y rápida interpretación por parte de un programa escrito por el usuario. Los formatos decimal y binario retornan los dígitos como una cadena (string) en ASCII sin ningún prefijo. El formato hexadecimal retorna el valor precedido de un "$" lo que permite su incorporación directa dentro de una variable en algunos lenguajes de programación.
La posibilidad de cambiar el formato de los valores retornados a la terminal se limita a los valores que no pueden nunca superar 255 (1 byte de largo). Aquellos valores que pueden en algunos casos ser superiores a 255 (ver bajo los comandos RESISTENCIA y MOTORES PASO A PASO) son siempre retornados en formato decimal.

Al resetear el ITC232-A el formato por defecto de los resultados retornados es decimal. Esto puede cambiarse utilizando el comando de CONFIGURACIÓN de resultados. Aparte de eso, se puede forzar el retorno de un resultado dado en cualquier otro formato agregando B ó % para binario, D para decimal y H ó $ para hexadecimal al final del comando (inmediatamente antes de enviar el CR).

El formato de los valores enviados al ITC232-A también puede ser especificado, agregando para ello B ó % para binario, D ó nada para decimal y H ó $ para hexadecimal antes del valor. Esto sólo se aplica para aquellos valores que nunca pueden ser mayores de 1 byte (255). Valores que puedan ser mayores (aunque sean en un caso particular menores) a 255 deben ser enviados en formato decimal. En decimal se permite el uso de números de largo variable
(005, 05 y 5 pueden ser utilizados indistintamente). No as¡ en el caso de valores en binario ó hexadecimal los que requieren un largo fijo; 8 dígitos para los binarios y 2 para los hexadecimales aunque el uso de espacios entre los dígitos está permitido. Las iniciales que forman los comandos están agrupadas por item. Por ejemplo, todos los comandos relativos a los puertos comienzan con la letra P.

LISTA DE COMANDOS:

Los caracteres ó valores entre <> son los únicos que deben ser enviados al ITC232-A. El resto de la palabra está escrito a título de explicación.

<@>gain (Otra vez): Este carácter ejecuta el último comando nuevamente. No requiere el envío de un Enter. Ideal para la repetición r pida de comandos ya que requiere sólo 1 byte para ser enviado.

Baudios:
<B>audios <300> ó <600> ó <1200> ó <2400> ó <4800> ó <9600> ó <19200> ó <38400> ó <57600> ó <115200>. Selecciona la velocidad de transmisión de datos por el puerto RS232 independientemente del voltaje en la patilla BAUD del integrado. Éste comando se ejecuta luego de enviarse el OK a la computadora. Asegúrese de que la computadora maneje adecuadamente los caracteres sin sentido resultantes del cambio de Baudios en el ITC232-A hasta que la computadora cambie también. A 115200 Baudios y utilizando el comando <@>gain se puede leer un puerto unas 1500 veces por segundo. Tenga en cuenta sin embargo de que esto normalmente requiere el uso de un programa compilado y que para utilizar el puerto serie a esta velocidad se requiere de buena programación. La posibilidad de generar errores también aumenta con la velocidad, en particular si el cable es largo.

Configuración del formato de intercambio de datos:

El ITC232-A puede enviar y aceptar datos en formato decimal, binario y hexadecimal. Por defecto, luego de un Reset, el ITC232-A funciona en decimal. Para cambiar el formato por defecto use: C>onfigurar <R>esultados <A>SCII <B>inario ó <D>ecimal ó <H>exadecimal ó <P>rograma. La configuración CRAP es ideal para trabajar desde dentro de un programa escrito por el usuario en lugar de utilizar un programa de emulación de terminal ya que al eliminar los caracteres CR y LF necesarios sino para separar las líneas en la terminal, la velocidad de ejecución aumenta. En CRAP lo siguiente es cierto:

- No se insertan CR ni LF.
- El formato por defecto de los resultados retornados es decimal (otro formato puede forzarse agregando B ó % para binario y H ó $ para hexadecimal al final del comando (ver bajo PUERTOS)).
- Las siguientes funciones quedan deshabilitadas:
  Ayuda en pantalla (pedida con <?> ó <H>)
  La leyenda en los mensajes de error (sólo ?n es retornado como código de error)
  PCp? en donde p = puerto (ver bajo configuración de puertos)
  S? (Ver más adelante configuración de motores paso a paso (steppers))
- La frecuencia real de la modulación de pulso (PWM) no es retornada a la terminal.

El intento de obtener información no disponible en CRAP resulta en el error ?3

HELP ó AYUDA:

<?> ó <H>elp Retorna un sumario de todas las funciones a la terminal. Para ver la segunda y tercera pantallas apriete una tecla. Para salir de la ayuda, apriete > ó Esc. Tenga en cuenta que algunos programas de emulación de terminal tienen un buffer muy pequeño para recibir caracteres del puerto serie y utilizan el protocolo XON-XOFF para requerirle al periférico que interrumpa el envío de datos. El ITC232-A ignora dicho protocolo y por lo tanto si el buffer no es suficientemente grande se pueden perder caracteres si bien esto no es usual. La ayuda no esta disponible en la configuración CRAP.

INTERRUPCIONES:

Las interrupciones (interrupts) no son comandos propiamente dichos ya que se originan en el ITC232-A sin intervención de la terminal. Las interrupciones se traducen en el envío de los caracteres H (High ó 1) y L (Low ó 0) a la computadora, según sean originados por las patillas IRQH ó IRQL respectivamente. Las interrupciones se originan como resultado de la transición de nivel (edge detected interrupt) lo que permite llevar, a través de condensadores de valor adecuado, varias líneas de interrupción a la misma patilla. Una vez detectada una interrupción el usuario puede detectar su origen utilizando para ello los puertos paralelos.

El envío de una interrupción es independiente de la operación del ITC232-A, el que deja lo que está haciendo para enviar el carácter correspondiente a la terminal.

Use Ud. ó no las patillas de interrupción, manténgalas siempre al nivel opuesto al que detectan. Si las patillas son utilizadas, polarízelas usando una resistencia de 10 K . La detección de la transición de un nivel a otro posibilita el uso de interrupciones múltiples. Para ello, conecte varias líneas de interrupción a la correspondiente patilla utilizando un condensador en paralelo con una resistencia de 100 K para cada interrupción. Una vez detectada la interrupción, se puede verificar el origen de la misma leyendo las puertas a las cuales cada línea de interrupción debe además estar conectada. Las interrupciones son particularmente útiles para detectar el fin de excursión de un brazo de robot a fines de registrar la posición del brazo con una variable en el programa en la computadora que lo dirige. Las interrupciones no envían ni OK ni CR ni LF ni >.

Prioridad de las interrupciones:

El ITC232-A utiliza el mismo reloj interno (timer) para todas las interrupciones y para medir el tiempo en actividades tales como el movimiento de los motores paso a paso ó la modulación de pulso. La prioridad de cada nivel de interrupción es la siguiente:
(1) IRQH
(2) IRQL
(3) motores paso a paso y
(4) modulación de pulso.
Esto quiere decir que en caso de ocurrir 2 interrupciones simultáneas sólo la mas prioritaria tendrá lugar. Si la modulación de pulso está trabajando muy cerca de los límites autorizados

(definidos por la combinación de frecuencia y ancho del pulso), una interrupción puede resultar en una pequeña alteración momentánea de la señal generada (glitch).

PUERTOS:

El ITC232-A tiene 6 puertos. Uno es el RS232 que le conecta con la computadora a través de las patillas 232TX y 232RX (Figura 1). Los otros 5 puertos pueden ser utilizados en el circuito asociado y se llaman PA, PB, PC, PD y PS. PA, PB y PC son puertos de 8 bits, de uso general. Cada patilla de estos 3 puertos puede ser configurada como un entrada (Input) de alta impedancia ó una salida (Output) con niveles TTL compatibles. El valor de cada patilla puede cambiarse individualmente ya que al escribir un valor dado a un puerto, los bits (patillas) no modificados no sufren ninguna alteración durante la escritura (no hay "saltos" de nivel). El cambio inadvertido del nivel de un bit otro que el objeto del cambio puede obviarse fácilmente leyendo primero el valor del puerto en cuestión (esta operación retorna el valor previamente escrito en el puerto) y ejecutando un AND ó un OR con el valor deseado antes de escribir el resultado al mismo puerto. Por ejemplo, supongamos que el valor en PA escrito previamente es B11000000. Queremos cambiar PA.0 a 1. Leemos el valor en PA utilizando el comando PRA (ver mas abajo) y ponemos el valor en una variable V en el programa. Luego hacemos V = V OR 1 y escribimos V en PA. Para cambiar PA.7 de 1 a 0, hacemos V = V AND 127 (B0111111) y escribimos V en PA.
PD es siempre 4 bits de entrada (Input) y comparte patillas con PS que es un puerto serie sincrónico (SPI ó Serial Peripheral Interface). PS puede ser utilizado para comunicarse con otros integrados es el circuito tales como un convertidor analógico digital, un "shift register", etc. En la Figura 1, PD y PS están referidos como PDx/PSx.
PD siempre está disponible. PS debe ser configurado antes de ser usado a los efectos de evitar el mensaje de error ?2 Port must be configured or enabled first (Este puerto debe ser configurado ó habilitado antes de usarse). Cuando se escribe ó se lee PS, PD "entrega" las patillas a PS, la transacción toma lugar utilizando la configuración entrada previamente y PD "recupera" las patillas. Esto es de tenerse en cuenta ya que luego de utilizar PS, las patillas vuelven al estado de alta impedancia. Esto exige el uso de un resistor de drenaje para la patilla del reloj (PS_CK). Si bien PD y PS pueden ambas utilizarse en el mismo circuito, esto no es recomendable.

Al encender el ITC232-A ó luego de un Reset, PA, PB y PC son automáticamente configuradas como entradas de alta impedancia, PD es siempre entradas y PS no tiene ninguna configuración.

Comandos referentes a los puertos:

Todos los comandos referentes a los puertos comienzan con la letra P. Hay 3 tipos de comandos para los puertos: configuración, lectura y escritura.

CONFIGURACIÓN DE UN PUERTO:

PA, PB y PC: <P>uerto <C>onfigurar <A> ó <B> ó <C> <valor>.

En <valor>, aquellos bits con valor 1 resultan en la correspondiente patilla configurada como salida, los 0 resultan en entradas. El preceder <valor> de una H ó un $ permite el uso de nomenclatura hexadecimal, una B ó un % se utiliza para nomenclatura binaria y una D ó nada es interpretado como decimal. Ejemplo: PCB240 , PCBD240, PCB $F0 y PCB B11110000 son comandos idénticos que configuran PA.0-PA.3 como entradas y PA.4-PA.7 como salidas. Los espacios en los comandos no son necesarios pero pueden ser agregados para clarificar el comando (tenga en cuenta que los espacios son ignorados pero aún así lleva tiempo enviarlos).

El intento de configurar PD resulta en el error ?A Port D is always a 4 bit input port (Puerto D es siempre 4 bits de entrada).

Puerto serie sincrónico (PS ó SPI):

La comunicación con otros integrados utilizando este puerto tiene la enorme ventaja de ahorrar patillas ya que la comunicación se hace a través de sólo 3 líneas. Cuando se utiliza el PS, la patilla PD3/PS_VDD DEBE estar conectada a VDD (sino, se genera el error ?B SPI requires pin PD3/PS_VDD always high, change and try again (El PS requiere que la patilla PD3/PS?VDD sea siempre igual a 1 (High). Modifíquelo e intente nuevamente).

El PS funciona a la velocidad fija de 57.6 KHz y puede ser considerado como un "shift register" circular del cual 8 bits est n dentro del ITC232-A y el resto dentro del integrado periférico. Esto significa que al entrar 8 bits al ITC232-A, otros 8 salen del mismo. Como resultado, cuando se lee PS, se escribe automáticamente el último valor escrito ó 0 si ninguno lo fuera previamente.
Importante: (1) SIEMPRE lleve la patilla del reloj (CLK) del periférico al nivel opuesto al requerido por el mismo a través de una resistencia adecuada (470 -10K). De otra forma, la primera transición del reloj no ser detectada. Si fuera necesario utilizar la patilla del reloj como entrada a PD, ponga un condensador de 0.1 uF entre la patilla del ITC232-A y el reloj (CLK) del periférico donde la resistencia va conectada. (2) Para utilizar PS, la patilla PD3/PS_VDD DEBE estar conectada High (VDD).

Para configurar el PS:

<P>uerto <C>onfigurar <S> <R>ead ó <W>rite ó <A>ll <valor>. Algunos integrados periféricos requieren ser sólo escritos. Otros, sólo leídos y otros aún requieren ser escritos y leídos. La posibilidad de configurar las 3 modalidades ofrece la ventaja de poder conectar mas de un integrado al PS.
Supongamos que se necesite leer del integrado X y escribir al integrado Y. La configuración de la interfase sincrónica es diferente para ambos. Utilizando PCSR V1 y PCSW V2 se logra la configuración en V1 para el X y la V2 para el Y. Cuando se escribe a uno ó se lee el otro, el ITC232-A automáticamente utiliza la configuración correcta para el periférico correspondiente. Si bien puede no ser necesario, se recomienda seleccionar el integrado periférico desde una de las patillas de PA, PB ó PC antes de leer ó escribir al mismo a través del PS. Haciendo V1 = V2 se obtiene el mismo resultado que utilizando <A>ll. Esto es necesario para la interfase con integrados que requieren ser leídos y escritos.

<valor> contiene información sobre la polaridad del reloj, la relación de fase entre los datos y el reloj, el sentido del byte retornado (derecho ó invertido) y la activación (enable) del PS según la Tabla y la descripción siguientes:

Bit	7	6	5	4	3	2	1	0	Hex	Dec
	A	{ Irrelevantes				} P	F	O		
	1	0	0	0	0	0	0	0	$80	128
	1	0	0	0	0	0	0	1	$81	129
	1	0	0	0	0	0	1	0	$82	130
	1	0	0	0	0	0	1	1	$83	131
	1	0	0	0	0	1	0	0	$84	132
	1	0	0	0	0	1	0	1	$85	133
	1	0	0	0	0	1	1	0	$86	134
	1	0	0	0	0	1	1	1	$87	135

Bit 7: 1 Activa el PS, 0 desactiva el PS.

Bits 6..3: Son irrelevantes. Use 1 ó 0 indistitantámente.

Bit 2: Polarización del reloj. Si se hace 1, el PD2/PS_CK reposa High, si se hace 0, dicha patilla reposa Low.

Bit 1: Fase entre datos y reloj. Si se hace 1, PD2/PS_CK cambia de nivel en el medio del cambio de PD1/PS_TX ó PD0/PS_RX, es decir el reloj y los datos est n fuera de fase. Si esta patilla se hace 0 los cambios de nivel del reloj se hacen en fase con niveles de las líneas de datos.

Bit 0: 0 preserva el orden de los bits recibidos desde el integrado periférico. 1 invierte el orden antes de enviarlo a la terminal.

La patilla PD3/PS_VDD no necesita estar High para la configuración pero sí para escribir ó leer el PS.

<P>uerto <C>onfigurar <A> ó <B> ó <C> ó <S> <?> {B, %, D, H ó $} retorna el valor de la configuración especificada entre {} ó en el modo por defecto si no se especifica. Este comando no est disponible bajo el modo CRAP. S se refiere al puerto serie sincrónico (SPI).

PARA LEER ó ESCRIBIR UN PUERTO:

Para leer PA, PB, PC ó PD: <P>uerto <R>ead <A> ó <B> ó <C> ó <D> {B,%,D,H,$}.
Cuando se lee una patilla previamente configurada como salida, el valor retornado es el presente en la patilla. {B,%,D,H,$} es opcional y se refiere al formato numérico del valor a retornar.

Para escribir a PA, PB ó PC: <P>uerto <W>rite <A> ó <B> ó <C> {B,%,D,H,$} <Valor>. Si se escribe a una patilla configurada como una entrada (input), el valor escrito queda retenido en

un bit "fantasma" detrás de cada patilla del puerto correspondiente. Si luego de escribir a una entrada se configura la misma como una salida, el valor escrito previamente es inmediatamente transferido a la patilla correspondiente. Esto permite "arrancar" una salida con un High en lugar del Low resultante luego de un Reset. Hay que tener cuidado cuando se escribe a un puerto con patillas configuradas como entradas que luego se cambian a salidas ya que si no, se pueden poner inadvertidamente niveles indeseados en las salida.

Para leer y escribir al PS: <P>uerto <R>ead <S> {B,%,D,H,$} y <P>uerto <W>rite <S> {B,%,D,H,$} <Valor>. Estos comandos son similares a los utilizados en PA, PB, PC y PD con las siguientes excepciones:

- La patilla PD3/PS_VDD DEBE estar High.
- La relación entre el valor de una patilla y su configuración es diferente.
- No hay bit "fantasma" detrás del valor escrito al PS como lo hay en PA, PB y PC.

Conección de un convertidor analógico/digital MC145041 al ITC232-A:

Surge frecuentemente la necesidad de medir y archivar el valor de un voltaje analógico. Mostraremos aquí, como ejemplo de utilización del puerto serie sincrónico, la implementación de un convertidor analógico digital (ADC) al PS. El ADC en cuestión es un MC145041 y la Figura 3 muestra como conectarlo al ITC232-A. Dicho ADC funciona de la siguiente forma: Primero se escribe, a través de P1D/PS_TX el valor correspondiente al canal que se quiere leer. El MC145041 es capaz de leer 11 canales separados, AN0-AN10. Dada la organización interna de este integrado, el AN0 se elige con PWS0, AN1 con PWS16, AN2 con PWS32, etc. Para un canal dado x el valor es = ANx * 16. La selección del canal genera simultáneamente la conversión analógica/digital de dicho canal que queda guardada en el "shift register" del MC145041 conectado al PS. Para extraer el valor, se debe leer el PS con el comando PRS. Ténganse en cuenta que al ejecutar una lectura, el reloj (PS_CK) extrae el byte del ADC y lo pone en el ITC232-A por PC_RX pero al mismo tiempo, el reloj empuja un nuevo byte del ITC232-A hacia el ADC a través de PC_TX. Este byte es el último valor escrito al PS ó 0 luego de un Reset. Por lo tanto, al leer el valor previo alojado en el MC145041, se realiza una nueva conversión cuyo valor queda retenido en el registro hasta un nuevo PRS. Si se cambia de canal se pierde el valor de la última lectura ya que si bien el MC145041 lo envía hacia el ITC232-A, ,éste último lo ignora (sólo envía a la computadora los valores resultantes de un PRS). Este mecanismo tiene varias consecuencias:

(1) Lecturas consecutivas y sobre todo rápidas de un mismo canal se pueden obtener enviando un "@" ya que el valor que se escribe al MC145041 es siempre el correspondiente al mismo canal.

(2) No puede pasar mucho tiempo entre (a) 2 lecturas consecutivas ó (b) una escritura y una lectura al PS. Si no, el valor leído es antiguo ya que corresponde a la conversión ejecutada inmediatamente después de la escritura. Si se hacen lecturas espaciadas, se debe enviar PRS dos veces en rápida sucesión y descartar el primer resultado.

Operación remota del ITC232-A utilizando una conección telefónica:

Se puede implementar, a muy bajo costo, un sistema operado remotamente desde una computadora con un modem. Para ello es necesario conectar el ITC232-A a un modem en un integrado como el AD7910 ó AD7911 según se muestra en la Figura 4.
Este circuito se muestra sólo a título de ejemplo ya que la implementación real del mismo requiere otros componentes como ser un transformador para aislar la línea del sistema, etc, de acuerdo a los requerimientos de la compañía de teléfonos local. Para que el ITC232-A entre en modo "remoto" es necesario: (a) Que la patilla BAUD está a masa, eligiendo así 300 Bauds y (b) que ocurra una interrupción a través de la patilla IRQL antes de que se reciba ningún comando luego de un Reset. En la Figura 4, la señal de llamada (Ring) genera el pulso que dispara el IRQL. Inmediatamente después, la patilla PA.0 se vuelve una salida y asume un nivel High. Esto se usa para contestar la llamada a través de un transistor ó relais que intercala una resistencia adecuada en la línea telefónica. PA.0 queda, de ahora en mas, excluída funcionalmente del ITC232-A. Los comandos de configuración y escritura del PA referentes a esa patilla son ignorados. A los 7 segundos de contestar la llamada (el tiempo necesario para que ambos modems se comuniquen), el ITC232-A envía el siguiente mensaje a la computadora remota: "Send a command within 30 sec or the ITC232 will hang up. This message will thereafter repeat itself if no commands within 5 minutes. Send OFF to hang up." traducido como "Envíe un comando dentro de los 30 segundos siguientes ó el ITC232-A interrumpir la llamada (colgar el teléfono). Este mensaje se repetir si no se recibe ningún comando en 5 minutos. Envíe OFF para colgar.". Al mensaje sigue un carácter ASCII 7 y el símbolo >. El ASCII 7 no sólo produce un pitido sino que además es útil para reconocer este mensaje desde dentro de un programa. Si no se envía ningún mensaje ó un CR en los primeros 30 segundos, el ITC232-A ejecuta la siguiente secuencia: (1) Se envía a la computadora el mensaje "DISCONNECTING" (desconectando) seguido de un ASCII(#7) y el símbolo >, (2) Se interrumpe la llamada (cuelga el teléfono) al hacer PA.0 = Low, (3) se reconfigura PA.0 como una entrada (input) y (4) el ITC232-A queda pronto para recibir una nueva llamada. Todos los otros valores y configuraciones previas quedan retenidos.
Dado que puede suceder que una llamada válida se interrumpa accidentalmente y a los efectos de no requerir una patilla extra para detectar las señal de "carrier" del modem, si el ITC232-A no recibe ningún comando ó CR en 5 minutos, el mensaje de alerta "Send a command within 30 sec or ..." se repite. 30 segundos después de no recibir respuesta el integrado corta la llamada siguiendo la secuencia descrita anteriormente. Lo mismo ocurre si se envía el command "OFF" desde la computadora remota. Puede también utilizarse el comando "RESET" pero esto tiene el inconveniente que se pierden todas las configuraciones y modos previos.
Importante: Luego de establecido el contacto, y mientras PA.0 = High, la patilla IRQL re-adquiere su carácter de interrupción y enviar una "L" cada vez que sea llevada a Low. Cuando suena el teléfono y se crea una interrupción = Low resultando en que el ITC232-A conteste la llamada, IRQL NO genera una "L". Se recomienda, en modo remoto, sólo utilizar el IRQL para detectar la señal de llamada (Ring) y el IRQH para todos las otras interrupcciones que hagan falta.

A los efectos de simular el modo remoto, Ud. puede crear un IRQL operando a 300 Baudios con el ITC232-A conectado directamente a la computadora según el circuito en la Figura 2.

Modulación del ancho de pulso (Pulse Width Modulation ó PWM):

La patilla PWM permite generar señales de frecuencia y ancho de pulso (duty cycle) variables. Su amplitud es constante. Hay 2 comandos para utilizar el PWM:

(1) <W>idth (ancho) <frecuencia> mediante el cual el ITC232-A genera una onda cuadrada (50% duty cycle).
(2) <W>idth (ancho) <frecuencia> <;> <ancho> mediante en el cual se genera una onda cuyo ancho de pulso (la parte del ciclo en que PWM vale 1 (High)) puede ser especificado en intervalos del 1%. La frecuencia, en Hertz, puede ser cualquier número entero entre 10 y 10000. El único formato permitido es el decimal para ambos par metros.

Los comandos WL ó WH se pueden utilizar para hacer la patilla PWM igual a 0 (Low) ó 1 (High) respectivamente. Esto equivale a especificar cualquier frecuencia en el rango permitido y un ancho del 0% ó del 100% (lo cual también está permitido). W? retorna el último comando enviado con referencia al PWM (Atención, la frecuencia devuelta es la requerida y NO la real, ver Nota 1).

Notas:

(1) Cuando se entra un comando de modulación de pulso, el ITC232-A retorna f=XXXXX (excepto si la configuración CRAP est siendo utilizada) donde XXXXX es SIEMPRE un decimal de 5 dígitos (con ceros a la izquierda si fuese necesario) que expresa la porción entera de la frecuencia REAL que el ITC232-A est generando. Puede haber una pequeña diferencia entre la frecuencia requerida y la real debido a las limitaciones impuestas por la resolución del cristal y el redondeo en los cálculos internos del integrado. La frecuencia real se puede calcular aplicando la siguiente ecuación: frecuencia real = 460800 ó Redondeo a enteros de (460800 ó frecuencia requerida).
(2) Durante el uso de un motor paso a paso: (a) Si WL ó WH están activos, la patilla permanece en ese estado. (b) Si el PWM est pulsando, dicha patilla se vuelve 0 (Low) durante el funcionamiento del motor. Una vez el motor que el motor se detiene, PWM reasume el pulsado.
(3) El ancho del pulso puede ser variado en intervalos del 1%. Sin embargo, una frecuencia muy elevada con un ancho de pulso muy alto, ó muy pequeño, requiere la medición de un período demasiado corto para la resolución del cristal. El resultado de esta limitación es que cuanto mas alta es la frecuencia mas se restringe el ancho de pulso por arriba y por debajo de 50%. Anchos del 1% ó 99% pueden obtenerse con frecuencias hasta de 220 Hz. Si se pide una frecuencia demasiado alta para el ancho requerido, el ITC232-A devuelve el siguiente mensaje de error: ?8 Frequency too high for requested duty cycle (frecuencia demasiado alta para el ancho de pulso requerido).

RESETEADO DEL ITC232-A: Comando: <RESET>. Este comando es idéntico a resetear el ITC232-A llevando la patilla 1 a masa (0 ó Low).

MEDIDA DE UNA CAPACIDAD ó DE UNA RESISTENCIA:

<R>esistencia <0> ó <1> ó <2> ó <3>. <0-3> representa el bit del puerto C que ejecuta la medición. La impedancia de entrada de PC es extremadamente alta. Por lo tanto, si se conecta una red RC según se muestra en la Figura 5, se puede medir el tiempo necesario para que el condensador se cargue al voltaje requerido para que la correspondiente patilla del PC detecte una transición de 0 a 1 (Low a High). Este punto de transición es constante para una misma patilla, pero varía de una patilla a otra, por lo cual los valores NO pueden compararse entre ellas (en otras palabras, cada patilla debe calibrarse independientemente). Cuando se envía un comando Rx, la siguiente secuencia toma lugar: (1) La patilla seleccionada se vuelve una salida y se pone a masa por un corto período a fines de descargar el condensador. (2) La patilla se vuelve una entrada y un reloj interno al ITC232-A mide el tiempo necesario para cargar el condensador al punto de transición de 0 a 1 para esa patilla. (3) El resultado es enviado a la computadora. El tiempo est medido en unidades arbitrarias y por lo tanto todos los resultados son relativos. Es por lo tanto necesario calibrar el sistema con una resistencia ó un condensador de valor conocido. Luego, se pueden extrapolar valores en Ohmios ó microfaradios haciendo una simple interpolación ya que la ecuación T = RC es lineal. El error de linealidad depende de la calidad del condensador y puede ser tan pequeño como el 0.5% en lecturas entre 10 y 32767. Los condensadores "Poly" dan los mejores resultados. Evítese el uso de electrolíticos. El resultado es siempre retornado como un decimal de 5 dígitos en el rango 00000-32767. Se agregan ceros a la izquierda si es necesario. En caso de que el resultado exceda 32767 el error ?7 Time out error es retornado. El rango de resistencia medible linealmente es de 200 a 10 M.

ADVERTENCIA: Resistencia menores de 200 ó capacidades mayores de 5 uF pueden resultar en la destrucción de la patilla correspondiente por exceso de corriente. La medida de resistencia menores de 500 introduce un mayor error de lectura y linealidad.
Si se utiliza simultáneamente el PWM por encima de los 5 KHz ó en una combinación de frecuencia y ancho de pulso muy cercana a los límites tolerables, el error de lectura aumenta ya que el reloj del PWM tiene prioridad sobre el reloj utilizado para la lectura de resistencia/capacidad.
No es necesario configurar las patillas 0-3 del PC como inputs antes de una lectura, el ITC232-A hace esto automáticamente. Ténganse en cuenta sin embargo que:

1) El cambio de la configuración de PC.0-PC.3 y su escritura entre lecturas puede introducir errores hasta de un 10% dependiendo de la calidad del condensador utilizado (Este error, utilizando un condensador Siemens© Poly de 0.47 uF es menor del 1%). Por lo tanto, se recomienda NO cambiar la configuración de las patillas utilizadas en este comando entre lecturas.
2) Se recomienda NO conectar ningún otro circuito a las patillas en uso para este comando.
3) Cambios muy grandes de la frecuencia del PWM entre lecturas de resistencia introducen errores que aumentan a medida que el valor retornado se hace mas pequeño.

Aparte de la aplicación obvia de la medida de una resistencia ó un condensador dados, este comando encuentra aplicación en muchos otros casos. Por ejemplo, a menudo pueden ser utilizados en lugar de una conversión analógica/digital, cuando por ejemplo se quiere medir la caída de voltaje en un puente de resistencia entre VCC y masa. En robótica, es necesario a menudo utilizar un codificador para determinar la posición de un elemento mecánico. Esto puede llevarse a cabo, uniendo mecánicamente un potenciómetro al eje de movimiento y leyendo la resistencia que varía a medida que dicho eje se mueve. También puede medirse la

intensidad de una fuente luminosa utilizando una célula de Cadmio (fotoresistencia ó una temperatura utilizando un termistor). Una aplicación interesante es la medida de la conductividad de una solución. En este caso, es necesario evitar la polarización de los electrodos resultante de la corriente continua que circula en la red RC. Para ello se pueden invertir los electrodos unas 1000 veces por segundo con una llave analógica como ser el 4066 que conecta los electrodos en forma directa ó cruzada con el resto del circuito. La inversión de los electrodos se obtiene conectando las patillas de control del 4066 al PWM y generando a través del mismo un pulso cuadrado (50% de duty cycle) de frecuencia adecuada (1000 Hz). Nuestros experimentos en conductividad han mostrado una sorprendente linealidad y un amplio rango de lectura utilizando este procedimiento.

CONTROL DE MOTORES PASO A PASO (Stepping motors ó steppers):

Los motores paso a paso reciben su nombre del hecho que se mueven en pasos discretos, con un ángulo constante para cada paso. Esto permite un movimiento muy preciso y el saber exactamente la posición del rotor en todo momento, contando los pasos desde una posición de "registro" inicial. La Figura 6A muestra una versión simplificada de un motor paso a paso. El motor consiste de un rotor que es un imán permanente y de un estator que contiene un número dado de bobinados. Estos bobinados, que pueden ser por ejemplo 52, están conectados en forma alternada según se muestra en la Figura 6A para 2 de ellos. La polaridad magnética del estator puede ser invertida según la dirección de la corriente a través de los bobinados (también llamados fases ó phases en Inglés). El motor en el ejemplo se puede hacer girar aplicando una corriente en los bobinados en 3 secuencias distintas: La 1¦ es AB/CD/BA/DC según se muestra en la Figura 6B (AB y BA son el mismo bobinado pero con inversa polaridad). Esta secuencia se denomina "monofásica" ya que sólo una de las 2 fases (bobinados) está activa a la vez. La 2da posibilidad es aplicar energía a las 2 fases simultáneamente. Esta modalidad, llamada "bifásica" y está ilustrada en la Figura 6C. El rotor no queda alineado con los polos del estator sino entre los mismos. La secuencia es ABCD/BACD/BADC/ABDC. Finalmente, la 3ra secuencia es una combinación alternada de las otras 2 y est representada en la Figura 6D. La secuencia es AB/ABCD/CD/BACD/BA/BADC/DC/ABDC. Nótese que en lugar de 4 posiciones en una vuelta ahora el rotor asume 8 posiciones. Esto resulta en la reducción del ángulo de cada paso a la mitad; de allí el nombre de esta modalidad, medio paso ó "half step" en Inglés. Estas 3 diferentes modalidades tienen su ventajas y sus contras. La 1¦ usa menos corriente pero es poco eficaz ya que sólo la mitad del alambre bobinado funciona a la vez, la otra mitad es peso muerto. La 2¦ modalidad da el mayor torque y es la mas comúnmente usada. La 3¦ en fin, ofrece la ventaja de una discriminación del ángulo de cada paso aumentada al doble, pero a costa de una menor eficiencia que la 2¦ y la necesidad de doblar la velocidad de la secuencia de los pasos si se quiere mantener la velocidad de giro.
Las 3 modalidades est n implementadas en el ITC232-A como veremos posteriormente. Nótese que el primer paso de una sesión tiene un efecto indeterminado ya que depende de donde est, el rotor en relación al estator para que el primero se mueva en un sentido ó el inverso ó aún no se mueva en absoluto si los polos est n por casualidad alineados. La secuencia comienza a efectuarse correctamente a partir del 2§ paso. Esto en general no tiene mayor trascendencia pero ha de tenerse en cuenta al diseñar un sistema.

En nuestro ejemplo, el ángulo por paso es de 90° y de 45° para la modalidad de medio paso. En la realidad, estos motores tienen ángulos de paso mucho mas pequeños. Valores comunes son 15° (24 pasos/vuelta), 7.5° (48 pasos/vuelta), 3.75° (96 pasos/vuelta), 3.6° (100 pasos/vuelta) y 1.8° (200 pasos/vuelta) aunque los llega a haber de mucha mayor resolución. Los motores paso a paso tienen un polo mas en el estator que en el rotor. Las bobinas del primero están conectadas en forma alternada de tal manera que el número de fases efectivo sigue siendo el mismo.
Se pueden utilizar 4 patillas de un puerto del ITC232-A para ejecutar las secuencias descriptas, conectándolas a un "driver" de corriente adecuado. Supongamos que se conectan los bobinados de la Figura 6 a un puerto dado así:

Bobinado	C	D	B	A
Bit	2	3	1	0

En modo monofásico (Figura 6B), la secuencia sería 0001, 1000, 0010, 0100 para girar en un sentido y la inversa para girar en sentido contrario. En modo bifásico, (Figura 6C) la secuencia sería 1001, 1010, 0110, 0101. Intercalando ambos modos obtenemos el modo de medio paso (Figura 6D) con la siguiente secuencia: 0001, 1001, 1000, 1010, 0010, 0110, 0100, 0101. Una vez terminada la secuencia, la misma se repite hasta llegar a la posición deseada. Estas secuencias, dicho sea de paso, pueden ser comenzadas en cualquier punto, basta con que el orden se mantenga. La inversión de corriente requerida se obtiene usando un puente en H como el mostrado en la Figura 7. Si bien este puede hacerse son componentes discretos, es mas conveniente utilizar un driver en un circuito integrado como ser el L298. Otra forma de resolver el problema de la inversión de la corriente es utilizar un motor de bobinados dobles. En lugar de un bobinado por fase, hay 2, bobinados en sentido opuesto. Si se hace pasar la corriente en uno de los bobinados, la polaridad de esa fase es la opuesta si se hace pasar la corriente por el otro bobinado. Estos motores sólo requieren 4 transistores para ser movidos (Figura 8) y se caracterizan por tener 5 ó 6 cables en lugar de 4. La desventaja de estos motores es que la mitad del alambre bobinado está inactivo la mitad del tiempo. La secuencia de corrientes aplicadas a ambos tipos de motor es idéntica.

Trabajar con motores paso a paso puede ser frustrante especialmente si se carece de las especificaciones del motor. Utilizando el ITC232-A el problema se reduce drásticamente ya que el tiempo entre los pasos así como toda la lógica necesaria está contenida en el integrado. Si salen 4 cables del motor, determine con un multímetro cuáles son las fases y conéctelos de acuerdo con la Figura 9. Si salen 5 ó 6 cables del motor, encuentre las fases con un multimetro teniendo en cuenta que el cable común es aquel que tiene la mitad de la resistencia con respecto a los otros 2. En algunos casos, sales sólo 5 cables del motor ya que todos los comunes est n conectados dentro del mismo. Conecte el motor de acuerdo con la Figura 8. En ambos casos, invirtiendo una fase se invierte el sentido de giro del motor.

Motores paso a paso y el ITC232-A:

El integrado ITC232-A se presta, de forma excelente, para el control de motores paso a paso. Hay, como para casi cualquier aplicación en donde este integrado se utilice, varias formas de llevar a cabo el control de estos motores. Una es escribiendo, directamente a los puertos, los valores descritos anteriormente. Esto requiere cierta complicación en el programa de control, ya

que aparte de escribir los valores en los puertos hay que medir cuidadosamente el tiempo entre los pasos, pero tiene la ventaja de permitir el movimiento sincrónico de 2 motores simultáneamente. Un motor es controlado por los 4 bits inferiores del puerto y el otro motor lo es por los 4 bits superiores del mismo puerto. Esta forma de controlar los motores tiene también la ventaja de permitir controlar la aceleración y desaceleración del motor, variando el tiempo entre escrituras consecutivas al puerto de control. Otra manera de controlar motores paso a paso es utilizar un driver lógico como ser el L297 y generar, a partir de un puerto, ó de la patilla PWM una sucesión de pulsos. A cada pulso corresponde un paso y el número de pulsos se sabe, conectando por ejemplo la patilla IRQL a PWM y contando las "L" recibidas.

La mejor forma de controlar motores paso a paso sin embargo, es utilizando las patillas y comandos previstos específicamente para este propósito, los que son muy fáciles de usar. Las siguientes reglas generales describen el sistema:

1) Los motores paso a paso se controlan a través de los 4 bits superiores de PA, PB y PC. Puede por lo tanto usarse 3 motores independientes.
2) El nombre del motor es el nombre del puerto al que est conectado.
3) Los motores deben ser habilitados (enabled) y configurados para poder ser utilizados.
4) La habilitación de un motor dado convierte los 4 bits superiores del correspondiente puerto en entradas (inputs) de alta impedancia. Al deshabilitar un motor se dejan las patillas correspondientes configuradas como entradas con el último valor escrito en el correspondiente bit "fantasma" (ver bajo PUERTOS).
5) La última configuración comandada es la v lida para todos los motores.
6) No se puede tener 2 motores configurados en forma diferente a la vez. Sin embargo, dado que el último valor es retenido en el ITC232-A, se puede reconfigurar otro puerto, hacer girar ese motor, reconfigurarlo como estaba previamente y mover el primer motor sin perder pasos. El cambio del modo (Monofásico, Bifásico y Half-Step (medio paso)) de un motor configurado previamente en otro modo puede llevar a la pérdida del sincronismo (pérdida de pasos) dependiendo del último que se haya ejecutado. Esto no sucede si se cambia la velocidad de frenado.
7) Mientras un motor paso a paso está girando la patilla PWM está inactiva. Si estaba siempre en 0 (por un WL previo) ó en 1 (por un WH previo) la patilla queda en su valor. Si estaba pulsando, el PWM se vuelve 0 (Low) hasta que el motor se detiene y se recibe el OK en la computadora.

Para habilitar y configurar un motor paso a paso:

<S>tepper (motor paso a paso) <E>nable (habilitar) <A> ó <B> ó <C> <M>onofásico ó <B>ifásico ó <H>alf step (medio paso) <velocidad en pasos/segundo> <;> <frenado>.
<A>, <B> y <C> se refieren a los 4 bits superiores (Px.4-Px.7) del PA, PB ó PC..
<M>onofásico, <B>ifásico y <H>alf step (medio paso) es el modo de operación según lo descripto anteriormente. La velocidad en pasos por segundo debe ser entrada siempre como un entero decimal entre 10 y 4000. <frenado> es el número ADICIONAL de pasos que dura el último paso y tiene un rango de 0 a 255 expresado siempre como un entero decimal. El propósito de <frenado> es prevenir el movimiento por inercia del motor una vez terminada la

secuencia de pasos. El valor de frenado debe ser mas alto cuanto mayor sea el motor y la velocidad de giro. 10% del valor de la velocidad es un buen valor para comenzar a probar.

ATENCIÓN: Se puede reconfigurar un puerto que está habilitado para un motor paso a paso con el comando PCxn pero esto puede ser peligroso. (1) Si se configuran las patillas del motor como salidas, el último valor utilizado en la secuencia de giro aparecer inmediatamente en las mismas. (2) La escritura de una combinación inapropiada de salidas puede llevar a la catastrófica destrucción de los transistores del driver del motor.

Todo lo anterior parece complicado pero en realidad es tan simple como esto:

- Para habilitar (enable) y configurar un motor paso a paso conectado al PA para que gire a 500 pasos/segundo (2 ms/paso), en modo bifásico y con un frenado de 10 pasos de duración (2 * 10 = 20 ms) , el comando es: SEAB500;10.
- Una vez que la configuración ha sido entrada, habilitar otro motor requiere sólo <S>tepper <E>nable <A> ó <B> ó <C>, los otros parámetros son los elegidos para el primer motor. El cambio de par metros es general para todos los motores.
- El error máximo del tiempo entre 2 pasos es <2.2%.

Para deshabilitar un motor paso a paso:

<S>tepper <D>isable <A> ó <B> ó <C>. Nótese que la deshabilitación de todos los motores no destruye la configuración vigente y que por lo tanto un <S>tepper <E>nable <A> ó <B> ó <C> funcionar con los parámetros previos.

Para requerir la configuración de todos los motores:

<S>tepper <?> {B ó %, D, H ó $} ó <S>tepper <E>nable <?> {B ó %, D, H ó $} donde los caracteres entre {} son opcionales y determinan la notación del valor en los puertos (si se omiten estos caracteres, el valor es devuelto en la notación por defecto). Estos comandos NO son válidos cuando la configuración de resultados CRAP est vigente.

Para hace girar un motor paso a paso:

<S>tep <A> ó <B> ó <C> <L>eft (izquierda) ó <R>ight (derecha) <Número de pasos>.

<L>eft ó <R>ight es el sentido arbitrario de giro. <Número de pasos> puede ser cualquier entero decimal entre 0 y 65535 incluídos.

Para detener un motor paso a paso mientras est girando:

Algunas veces es necesario forzar la parada del motor. Esto puede hacerse enviando cualquiera de los siguientes caracteres desde la computadora al ITC232-A: Espacio (ASCII 32)), "S", "s", ">", Esc (ASCII 27)), ó Enter (ASCII(#13)). Esto resulta no sólo en la detención del motor, incluyéndose el frenado configurado, sino además en el envío, a la terminal, del número de

pasos remanentes para la finalización del comando previo como un decimal de 5 dígitos (con ceros a la izquierda si fuera necesario), seguidos del mensaje "steps to go" (pasos remanentes). Éste último mensaje no aparece si la configuración CRAP está vigente.

Un buen ejemplo de detención forzada del motor es en la secuencia inicial de un sistema mecánico que utiliza un motor paso a paso:

1) Configúrese y habilítese todos los motores que serán utilizados. Si la velocidad de giro es muy alta (mas de 1000 pasos por segundo) configúrese el motor a una velocidad mas baja. Luego se cambiar a otra mas alta (ver explicación mas abajo).

2) La computadora no sabe en donde está el elemento mecánico movido por el motor. Una llave, situada en el extremo de la excursión del elemento mecánico es apretada cuando el mismo llega a dicho punto llamado de "registro". La llave está conectada a una de las patillas de interrupción (IRQL ó IRQH). Recuérdese que se pueden conectar muchas entradas a la misma interrupción utilizando condensadores en serie.
3) Hágase girar el motor un número de pasos superior a la excursión máxima posible. Esto asegura la generación de una interrupción antes de que el motor se detenga. La computadora, al recibir la "H" ó "L" de la interrupción envía uno de los caracteres descritos mas arriba para parar el motor. El número de paso remanentes se descarta. En este momento, la computadora sabe exactamente la posición del motor y puede comenzar a contar los pasos para saber donde se encuentra el elemento mecánico en todo momento. La velocidad inicial del motor no puede ser demasiado alta porque sino el tiempo transcurrido entre el envío de la interrupción y la respuesta de la computadora se hace demasiado largo y el motor sigue girando varios pasos. Si bien esto no es generalmente un problema (ya sea porque el número de pasos posteriores a la interrupción es siempre el mismo, ó porque la precisión del sistema lo permite), se recomienda en estos casos registrar el motor a menor velocidad que la utilizada posteriormente.

En algunos casos es necesario utilizar mas de 3 motores paso a paso ó es preferible mover todos los motores desde un sólo puerto y habilitar un driver u otro desde otro puerto a los efectos de ahorrar patillas. Por ejemplo, en la Figura 9, las líneas ENA y ENB del L298 pueden conectarse a las patillas de otro puerto. De esta forma se puede elegir que motor se est controlando.
Supongamos que tenemos 2 motores, M1 y M2 conectados al PA. Se utiliza PB para habilitar el driver de uno u otro. El problema es que es improbable que donde se dejó en la secuencia de giro de M1 sea donde se requiere comenzar para M2 y viceversa. Esto se arregla haciendo "girar" el motor el número de pasos requeridos para completar una vuelta sin habilitar ningún driver. Ejemplo: M1 y M2 tienen ambos una resolución de 3.6§ (100 pasos/vuelta) (se puede utilizar motores de distinta resolución, sólo hay que hacer los cálculos correspondientes para cada motor).
1) Actívese mediante el PC el driver del M1 y hágaselo dar 1230 pasos.

2) Calcúlese el resto de la división 1230/100 = 30. Desactívense todos los drivers y hágase dar al mismo puerto 100 - 30 pasos. No se modifique el contador de pasos en el programa ya que el motor no se ha movido.
2) Actívese el motor M2 con otra patilla del PC y hágaselo dar 13 pasos.

3) Desactívense M2 y hágaselo dar al puerto 100 - 13 = 87 pasos en la misma dirección.
4) Actívese el motor M1 y repítase el proceso todas las veces que sea necesario.

Nota: Evítese el cambio de modo (Bifásico, Monofásico, Half step) de un motor a otro.

SUMARIO DE COMANDOS:

Los ítems entre <> son obligados. Aquellos entre {} son opcionales.

* Comandos que no est n disponibles cuando la configuración <C>onfigure
<R>esults <A>SCII <P>rogram (CRAP) est activa.

<@>gain.

<B>aud rate <300>, <600>, <1200>, <2400>, <4800>, <9600>, <19200>, <38400>,
<57600> and <115200>.

<C>onfigure <R>esults <A>SCII <B>inary or <D>ecimal or <H>exadecimal or
<P>rogram.

* <H>elp or <?>.

Interrupts: L or H.

<OFF>. Returns DISCONNECTING ASCII(#7) > and makes PA.0 an input (to hang up the phone). Only available if in phone mode (BAUD pin = Low and IRQL asserted before a command is received after reset or power-up).

<P>ort <C>onfigure <A> or <B> or <C> {B,%,D,H,$} <Value>.
<P>ort <C>onfigure <S>erial <R>ead or <W>rite or <A>ll {B,%,D,H,$} <V>alue.

La Tabla siguiente muestra todas las combinaciones de configuración para PS:

Bit 7	6	5	4	3	2	1	0	Hex	Dec
MUST BE	{	Irrelevant	}			POL	PHASE	ORD	
1	0	0	0	0	0	0	0	$80	128
1	0	0	0	0	0	0	1	$81	129
1	0	0	0	0	0	1	0	$82	130
1	0	0	0	0	0	1	1	$83	131
1	0	0	0	0	1	0	0	$84	132
1	0	0	0	0	1	0	1	$85	133
1	0	0	0	0	1	1	0	$86	134
1	0	0	0	0	1	1	1	$87	135

PCS0 Disables the serial port.

* <P>ort <C>onfiguration <A>,<B> or <C> or <S> <?> {B,%,D,H or $}. Returns the port configuration.

<P>ort <R>ead <A> or <B> or <C> or <D> or <S> {B,%,D,H,$}. Reading PS sends previously written value out the PD1/SP_TX pin using the Read configuration.

<P>ort <W>rite <A> or <B> or <C> or <S> {B,%,D,H,$} <Value>.

<RESET> Equivalent to a hardware reset.

<R>esistance <0> or <1> or <2> or <3>. Function type.
0-3 are the bit or pins on port C.

<W>idth <L>ow. Forces PWM pin Low.

<W>idth <H>igh. Forces PWM pin High.

<W>idth <frequency>.
<frequency> can be any value between 10 and 10,000 Hz and it MUST be specified in decimal format. A 50% duty cycle is assumed.

<W>idth <frequency> <;> <Duty cycle>.
<frequency> can be any value between 10 and 10,000 Hz and it MUST be specified in decimal format. <Duty cycle> can be any integer value from 0 to 100 %.

* <W>idth <?>. Returns the last <W> command.

<S>tepper <E>nable <A> or <B> or <C> <M>onophasic or <B>iphasic or <H>alf step <Speed> <;> <Stop delay>.
<Speed> is in steps/s (10-4000). <Stop delay> is in steps (0-255).

<S>tepper <D>isable <A> or <B> or <C>.

* <S>tepper <?> {B or % or D or H or $} or <S>tepper <E>nable <?> {B or % or D or H or $}. Returns the configuration, the active steppers and the last value written to each active stepper in the requested format.

<S>tep <A> or <B> or <C> <L>eft or <R>ight <Number of steps>. Makes the motor step.

To stop a motor while stepping:
Send ASCII(#32) (space bar), or an "S" or "s" or an ">", or Esc (ASCII(#27)),

or Enter (ASCII(#13)). * The number of remaining steps is sent to the terminal as a 5 digit (with leading zeros if necessary) decimal number.

LISTA DE ERRORES

?1 Syntax error. (Error de sintaxis).

?2 Port must be configured or enabled first. (El puerto debe ser configurado ó habilitado antes de usarse).

?3 Command not allowed in current configuration. (Comando no disponible en la configuración vigente).

?4 No such port. (Dicho puerto no existe).

?5 Value out of range or syntax error. (Valor fuera del rango permitido ó error de sintaxis).

?6 Pin configured as an output. (Patilla configurada como salida).

?7 Time out error. (Se requiere un tiempo mayor del permitido para la ejecución del comando).

?8 Frequency too high for required duty cycle. (Frecuencia demasiado elevada para el ancho de pulso requerido).

?9 Baud rate not supported. (Valor de Baudios no válido).

?A Port D is always a 4 bit input port. (El puerto D consiste siempre de 4 bits de entrada).

?B SPI requires pin PD3/PS_VDD always high, change and try again. (El uso del PS requiere que la patilla PD3/PS_VDD est, siempre en 1 (High), cámbielo e intente otra vez).

DIAGRAMA DE CONECCIONES:

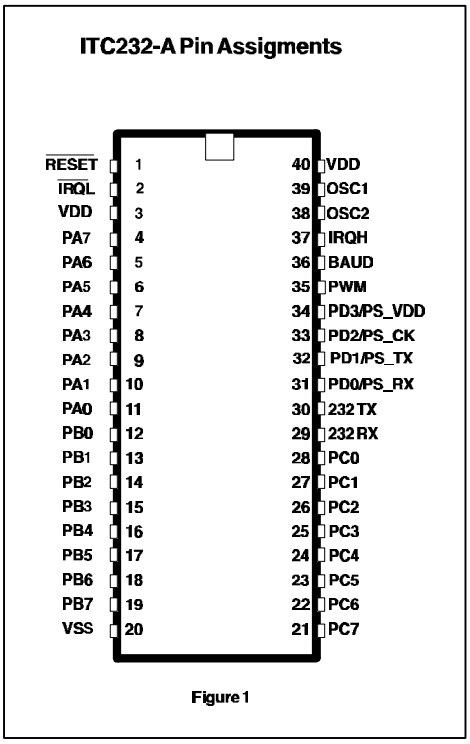


Figure 1

(1) RESET: Traer esta patilla a 0 (Low) hace: (a) Toda configuración previa se pierde. (b) PA, PB & PC son configuradas como entradas. (c) Los Baudios se establecen de acuerdo al nivel de la patilla BAUD. (d) La configuración CRAD se asume por defecto. (e) La patilla PWM se lleva a 0 (Low). (f) El mensaje:
Welcome to the ITC232-A
? or h for help
seguido de ASCII(#7), CR, LF y ">" son enviados por la patilla 232 TX.

(2) IRQL: Sólo sensible a la transición de 1 a 0. Si la patilla BAUD = 0 (Low, 300 Baudios) e IRQL cambia de 1 a 0 antes de que se reciba un comando, entonces el ITC232-A entra en modo remoto. A partir de entonces un IRQL envía, como de costumbre una L a la computadora. (V,ase interrupciones).

(3 & 40) VDD: +4.5 to +5.5 Voltios con referencia a VSS.

(4-11) PA0..PA7: Puerto paralelo A. PA0 es excluido del PA cuando el ITC232-A est en modo remoto. PA4..PA7 son usadas para el motor paso a paso en A.

(12-19) PB0..PB7: Puerto paralelo B. PB4..PB7 son usadas para el motor paso a paso en B.

(20) VSS: El voltaje digital mas bajo conectado al ITC232-A (usualmente=masa).

(21-28) PC0..PC7: Puerto paralelo C. PC0..PC3: Usadas para medir la constante RC de una red RC. PC4..PC7: Usadas para el motor paso a paso en C.

(29) 232 RX: Recibe datos del puerto RS232-C de la terminal.

(30) 232 TX: Transmite datos al puerto RS232-C de la terminal.

(31) PD0/PS_RX: Patilla com£n a PD (siempre entradas) y PS. Cuando PS est activo, esta patilla recibe datos del integrado periférico en forma sincronizada con la patilla PD2/PS_CK.

(32) PD1/PS_TX: Patilla com£n a PD (siempre entradas) y PS. Cuando PS est activo, esta patilla envía datos al integrado periférico en forma sincronizada con la patilla PD2/PS_CK.

(33) PD2/PS_CK: Patilla com£n a PD (siempre entradas) y PS. Cuando PS est activo, esta patilla sirve de reloj (clock) para sincronizar los datos entrados y salidos a través de PS_TX y PS_RX. El reloj puede estar en fase ó fuera de fase con los datos y puede reposar en 1 ó en 0 de acuerdo a la configuración del PS.

(34) PD3/PS_VDD Patilla com£n a PD (siempre entradas) y PS. Debe estar a nivel 1 (High) para que el PS funcione.

(35) PWM: Salida de la modulación de ancho de pulso (Pulse Width Modulation).

(36) BAUD: Selecciona la velocidad de transmisión hacia y desde la terminal. 0 ó Low = 300 Baudios, 1 ó High = 9600 Baudios. El cambio posterior de esta patilla es ignorado hasta un nuevo reset.

(37) IRQH: Sólo sensible a la transición de 0 (Low) a 1 (High). Envía un H a la terminal.

(38) OSC1: Al cristal de 3.6864 MHz ó a un reloj externo.

(39) OSC2: Al cristal. Si se aplica un reloj externo al OSC1 OSC2 debe quedar sin conectar.

ESPECIFICACIONES ELÉCTRICAS:

Valores máximos (Los voltajes son con referencia a VSS)

Parámetro	Valor	Unidades
Voltaje de la fuente	-0.3 a + 7.0	V
Corriente de salida por patilla	25	mA
Voltaje en las entradas	VSS-0.3 a VDD+0.3	V
Temperatura de almacenamiento	-65 a +150	°C
Rango de Temperatura de operación	0 a +70	°C

Características eléctricas (VDD-VSS = 5.0 V DC)

Característica	Min	Típico	Max	Unidades
Voltaje de salida (I<10uA)	VDD-0.1	-	0.1	V
Voltaje de salida (I=0.8mA)	VDD-0.8	-	0.4	V
Entrada = 1 (High) PA, PB, PC, PD, IRQ's, BAUD, 232 RX, RESET	0.7 VDD	-	VDD	V
Entrada = 0 (Low) PA, PB, PC, PD, IRQ's, BAUD, 232 RX, RESET	VSS	-	0.2 VDD	V
Corriente total	4.7	7.0	-	mA
Corriente de pérdida en entradas (PA,PB,PC,PD)	-	-	10	µA
Capacidad PA,PB,PC,PD	-	-	12	pF
Capacidad RESET, IRQ's, 232TX, 232 RX, BAUD	-	-	8	pF

Notas:

- Todos los valores muestran medidas promediables.
- Las medidas fueron realizadas a 25 °C.

Programa de ejemplo utilizando lenguaje BASIC

```
CLS
TRUE = 1
FALSE = 0
REM Open COM port
OPEN "Com1: 9600,N,8,1,CD0,CS0,DS0,OP0,RS,TB2048,RB2048" FOR RANDOM AS #1
'Esto abre un buffer de 2 Kb para recibir y transmitir datos a 9600 Baudios. Se
‘puede utilizar COM2 y cualquier otra velocidad hasta 19200 Baudios en Basic.
‘También se puede exceder este límite si la computadora es lo suficientemente rápida
‘introduciendo (poking) valores directamente en los registros del 8250 en la
‘puerta serie.

PRINT "SIRVASE RESETEAR EL ITC232-A"
PRINT
GOSUB READSERIAL
'Espere hasta que el mensaje "Welcome...etc" sea recibido

PRINT S$
'e imprímalo en la pantalla (S$ contiene el string con el mensaje recibido).
```

```
W$ = "crap": GOSUB WRITESERIAL
'Todo comando va a la subrutina de escritura WRITESERIAL en W$. El comando "CRAP"
‘pone al ITC232-A en el modo de programa que evita el envío de CR y LF para
‘optimizar la velocidad de las transacciones. Para que los resultados devueltos
‘en V y V$ sean correctos, "CRAP" DEBE estar activa.
W$ = "prF": GOSUB WRITESERIAL
'Esto es un ejemplo de error ya que no hay puerto F
END
SUBROUTINES

REM  Para escribir al puerto serie
WRITESERIAL:
PRINT #1, W$
'Esto envía el comando en W$ al ITC232-A. Dado que como respuesta a un comando
‘el integrado siempre devuelve un OK ó un mensaje de error, ahora leemos el
‘puerto serie.

GOSUB READSERIAL
RETURN
'y volvemos al punto de llamada de la subrutina. El texto enviado por el
‘ITC232-A as¡ como los resultados extraídos del mismo quedan en S$, V$ y V
(véase mas abajo).

REM  Para leer el puerto serie
READSERIAL:
S$ = ""
'Vaciar la variable antes de leer
IF LOC(1) = 0 THEN GOTO READSERIAL
'Esta línea es para cuando se hace un "polling" del puerto serie como es el
‘caso cuando se resetea el ITC232-A y se espera el mensaje de "Welcome...".

REM Poner el string recibido en S$
Lp1:
C$ = INPUT$(1, #1)
S$ = S$ + C$
IF C$ <> ">" THEN GOTO Lp1
'El programa hace un bucle (loop) hasta que encuentra un ">" ya que el
‘ITC232-A SIEMPRE termina el envío con un ">".

REM Decodificar el string (V$) y extraer su valor en (V)
'Lo que sigue no es siempre necesario pero sirve obtener el resultado de un
‘comando y ponerlo en las variables V y V$. Para ello, el ITC232-A debe estar
‘configurado con el comando CRAP. Si no se requieren los valores devueltos por
‘el ITC232-A entonces ubique un RETURN aquí y prescinda de lo que sigue.

VALIDERROR = TRUE
```

```
ERRORCODE$ = ""
V$ = ""

FOR H = 1 TO LEN(S$)
IF MID$(S$, H, 1) = CHR$(7) THEN VALIDERROR = FALSE
IF MID$(S$, H, 1) = "?" THEN ERRORCODE$ = MID$(S$, H + 1, 1)
NEXT H
'Detectar si vino un  ?" en S$ lo que indica un error. Eliminar el falso error
‘resultante del "?" en el mensaje "Welcome...etc" utilizando para ello el hecho
‘que este mensaje también contiene un CHR$(7).

IF (VALIDERROR = TRUE AND ERRORCODE$ <> "") THEN GOSUB ERRORSUB:
RETURN

IF LEN(S$) > 3 THEN V$ = RIGHT$(S$, LEN(S$) - 2): V$ = LEFT$(V$, LEN(V$) - 1)
'Si el comando exige un resultado entonces el ITC232-A devuelve un string de
‘mas de 3 caracteres. En ese caso, extraer el resultado en V$ y

V = VAL(V$)
'ponerlo en una variable numérica.

RETURN

ERRORSUB:
'Esta subrutina se explica por sí misma.
PRINT
PRINT "Error #"; ERRORCODE$
RETURN
```

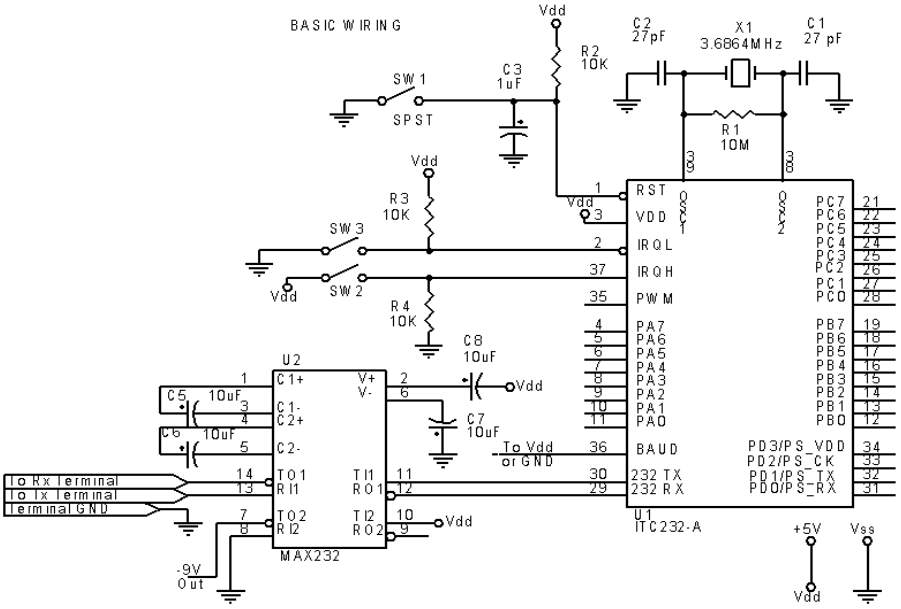


Figura 2

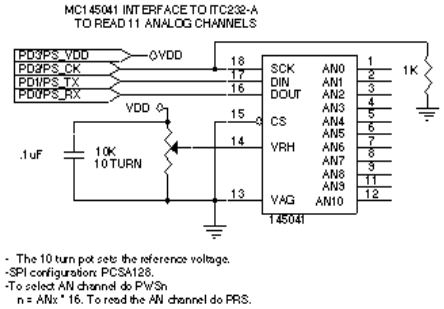


Figura 3

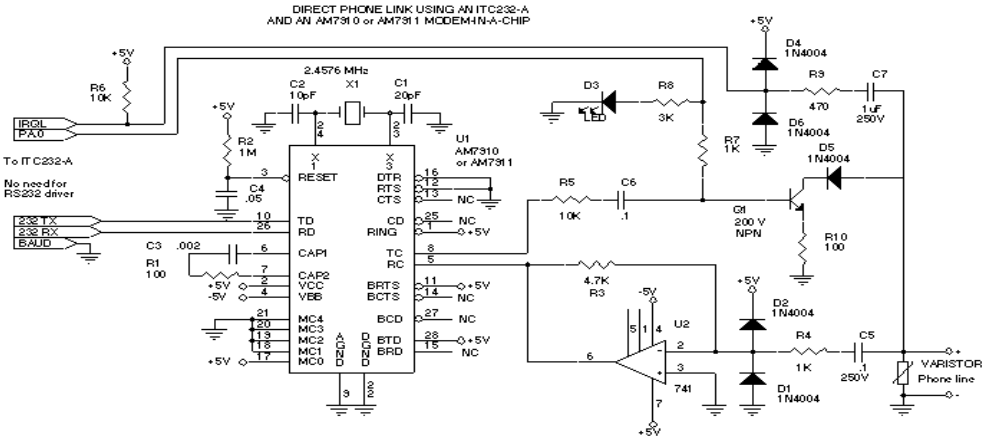


Figura 4

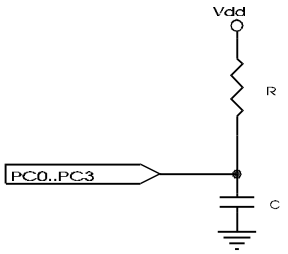


Figure 5

Resistor network for measuring resistance or capacitance

Figure 5

Resistor network for measuring resistance or capacitance

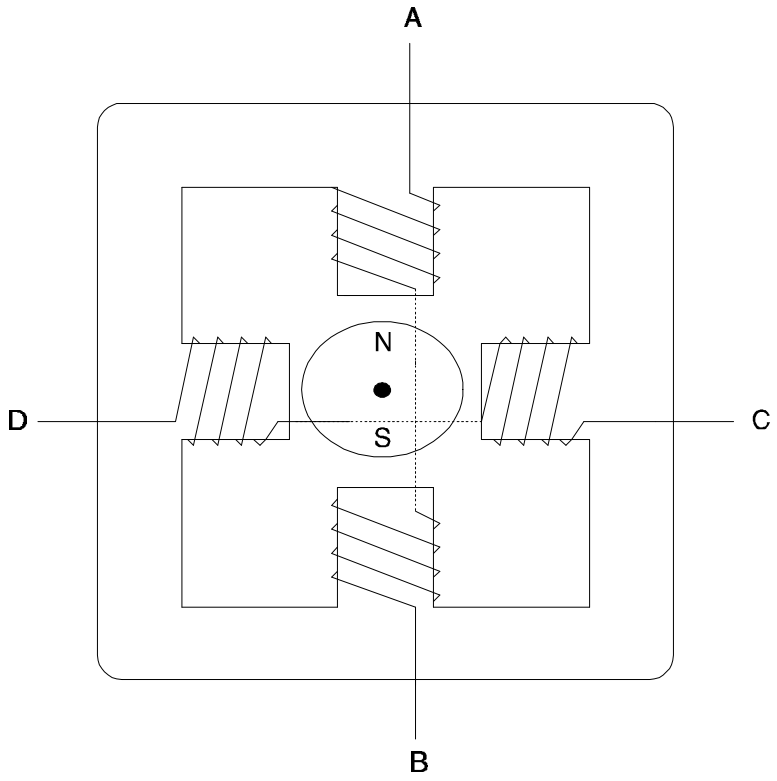


Figure 6a

Stepping motors

Figure 6a

Stepping motors

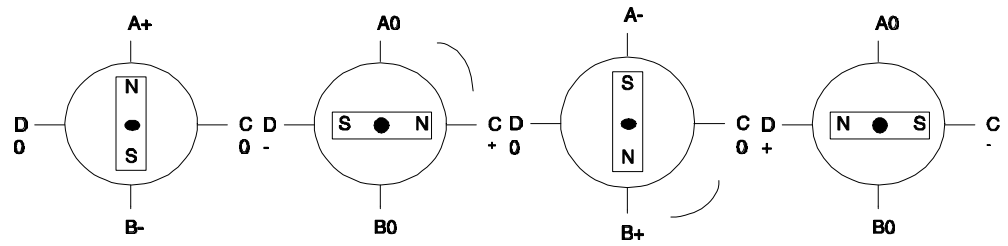


Figure 6B

Figure 6B Stepping motor (Monophasic)

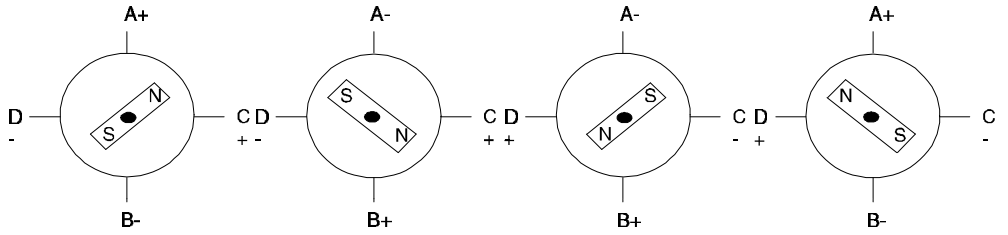


Figure 6C

Figure 6C Stepping motor (Biphasic)

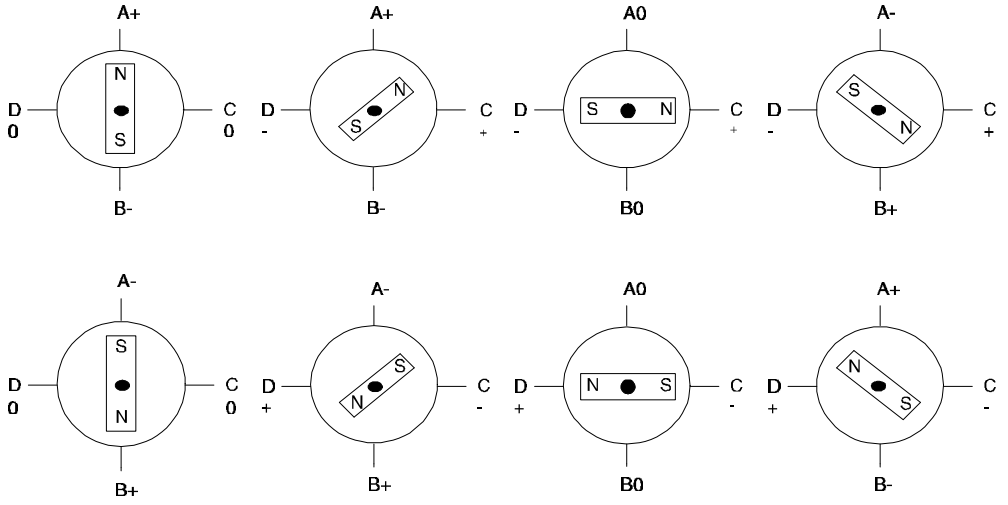


Figure 6D

Figure 6D Stepping motor (Half step)

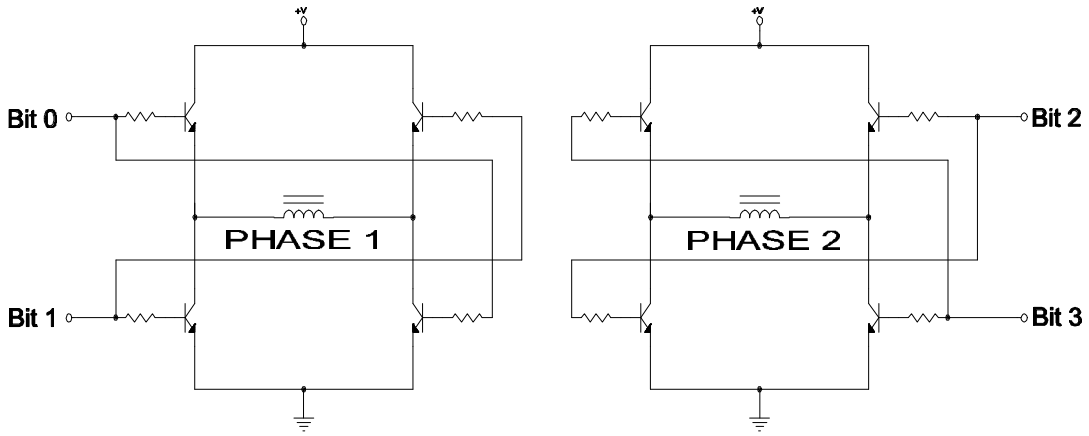


Figure 7 Stepping motor H driver

Figure 7

Stepping motor H driver

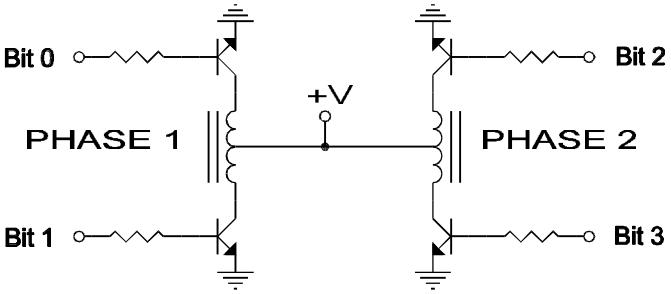
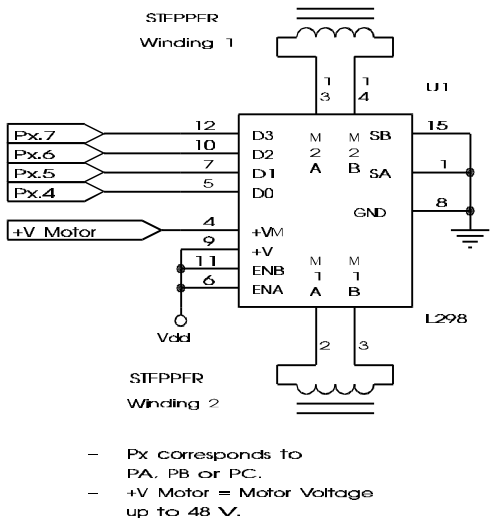


Figure 8 Stepping motor 4 transitor driver

Figure 8

Stepping motor
4 transitor driver



- Px corresponds to PA, PB or PC.
- +V Motor = Motor Voltage up to 48 V.

Figure 9

Four wire stepping motor IC driver

Figure 9

Four wire stepping motor IC driver