



UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

MANUAL DE SENSORES, MOTORES Y CONTROLADORES

Un trabajo conjunto de:

**SEBASTIÁN SALINAS V.
MAURICIO CERDA E.
OSCAR SANHUEZA
PEDRO ROMERO G.**

INDICE

INDICE	3
INTRODUCCION	4
MOTORES	5
MOTORES DE CONTINUA	6
SERVO MOTORES	7
MOTORES PASO A PASO O STEPPER	8
SENSORES	11
SENSOR INFRARROJO	12
SENSOR DE ULTRASONIDO SRF04	15
SENSOR DE TACTO	17
CONTROLADORES	18
CONTROLADOR SERIAL DE SERVO MOTORES	19
PUENTE H	22
CONTROLADOR DE MOTORES DE PASO O STEPPER	33
MANEJO DE PUERTOS DE UN PC	35
DESCRIPCIÓN DEL PUERTO SERIAL	36
MANEJO DEL PUERTO SERIAL DESDE C	37
MANEJO DEL PUERTO SERIAL DESDE VISUAL BASIC	39
DESCRIPCIÓN DEL PUERTO PARALELO	41
MANEJO DEL PUERTO PARALELO DESDE C	42
MANEJO DEL PUERTO PARALELO DESDE VISUAL BASIC	43

INTRODUCCION

MOTORES

MOTORES DE CONTINUA

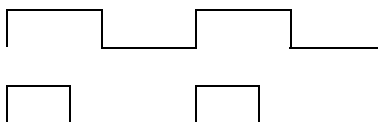
Son los motores por excelencia de juguetes. Son los más sencillos de manejar pero no por eso los mejores. Constan como todos los motores de un rotor y de un estator. Este último está compuesto generalmente de un imán permanente mientras que el rotor es bobinado.

El motor tiene dos contactos, los cuales al ser polarizados provocan el giro de este. La dirección del giro depende de la polaridad de la tensión y la velocidad depende de la magnitud de la tensión.



Si bien es posible controlar la dirección de estos con una configuración de transistores (Punto H) el problema se presenta cuando se desea controlar su velocidad. Como generalmente no es posible entregarle un voltaje de tensión variable, se recurre a una técnica digital basada en modulación por ancho de pulsos o PWM por su sigla en inglés.

Esta técnica consiste en aplicar un tren de pulsos de período fijo e ir variando el ancho del pulso.



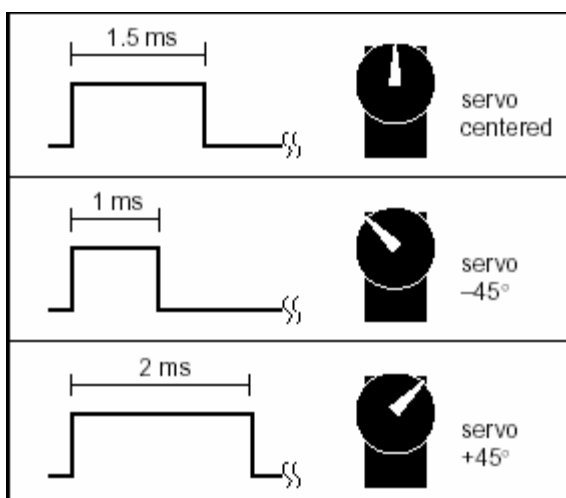
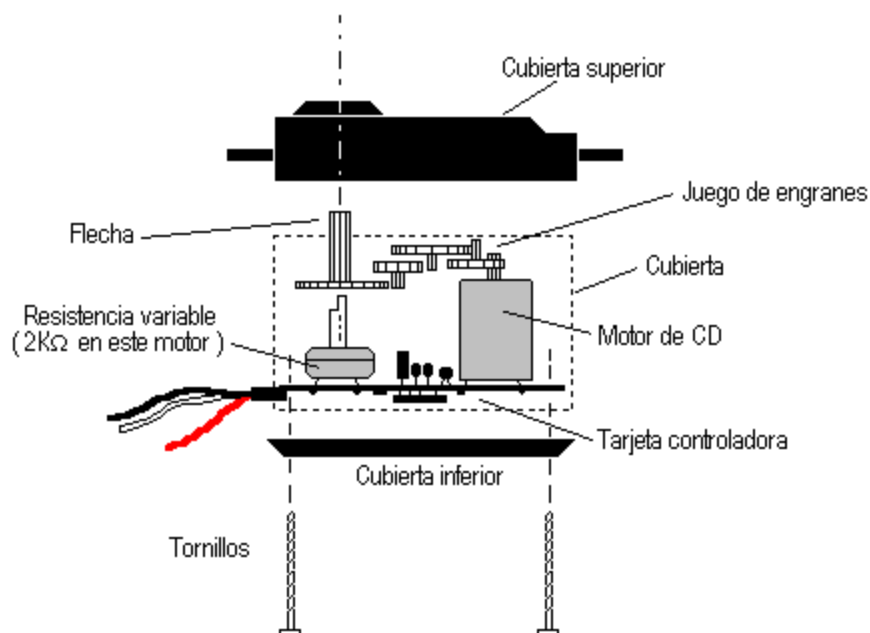
Con esto se logra que el motor “vea” un voltaje efectivo menor provocando así que gire a una velocidad menor.

Los motores de continua se presentan en un amplio rango de voltajes, potencias y precios. Si bien se pueden conseguir en juguetes viejos, estos solo sirven a modo experimental y no para aplicaciones de mayor exigencia.

Un buen motor de continua se puede comprar en Casaroyal <http://www.casaroyal.cl> por unos \$ 8.500. Es de 12 [V], tiene 6 [kg/cm] de torque, tiene una caja reductora incorporada y una velocidad de 60 [rpm] a voltaje nominal.

SERVO MOTORES

Los servo motores son motores que traen incorporado retroalimentación de posición. Su rango típico de movimiento es de 90° o 180° . Son capaces de entregar gran precisión en su control. Este se realiza en base a pulsos que van de 1 a 2 milisegundos para motores cuyo rango de movimiento es de 90° y de 0.5 a 2.5 milisegundos para los que tienen un rango de 180° , repetidos 60 veces por segundo. La posición del servo motor es proporcional al ancho del pulso. Por convención para pulsos de 1.5 milisegundos estos se sitúan en su posición central.

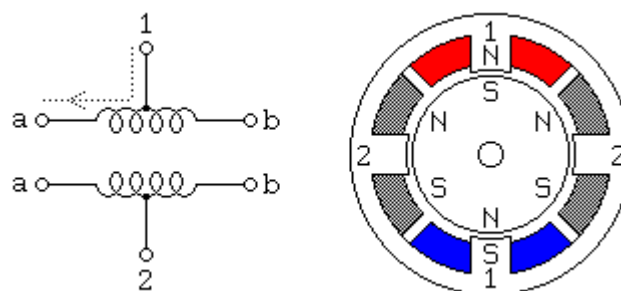


Los servo motores se pueden adquirir en tiendas de modelismo como Mirax (<http://www.mirax.cl>) y Hobbylandia.

Se pueden adquirir desde \$ 6.300 y su precio aumenta junto con el torque y tipo de rodamientos y eje (plásticos, metálicos, etc).

El rango de voltajes de estos motores va desde los 4.8 [V] hasta los 7.2 [V].

MOTORES PASO A PASO O STEPPER



Los motores, tanto de corriente continua como de corriente alterna, son muy efectivos en muchas labores cotidianas. Pero debido a problemas tales como la inercia mecánica o su dificultad para controlar su velocidad, se desarrollaron otro tipo de motores cuya característica principal es la precisión de giro. En efecto, en un motor paso a paso no sólo se puede controlar la cantidad de vueltas del mismo, sino que hasta centésimas de las mismas.

Internamente un motor de este tipo está compuesto por dos bobinas con punto medio. Estas bobinas se ubican en lo que se denomina **estator**, es decir la carcasa exterior del motor. La parte móvil de este motor al igual que en los de corriente continua es estriada y se denominada **rotor**.

Exteriormente posee 6 o 5 cables. Cuatro de estos cables corresponden a cada uno de los extremos de las dos bobinas existentes, mientras que los otros dos corresponden al punto medio de cada una. En el caso de que el cable restante sea uno corresponde a estos dos últimos unidos internamente.

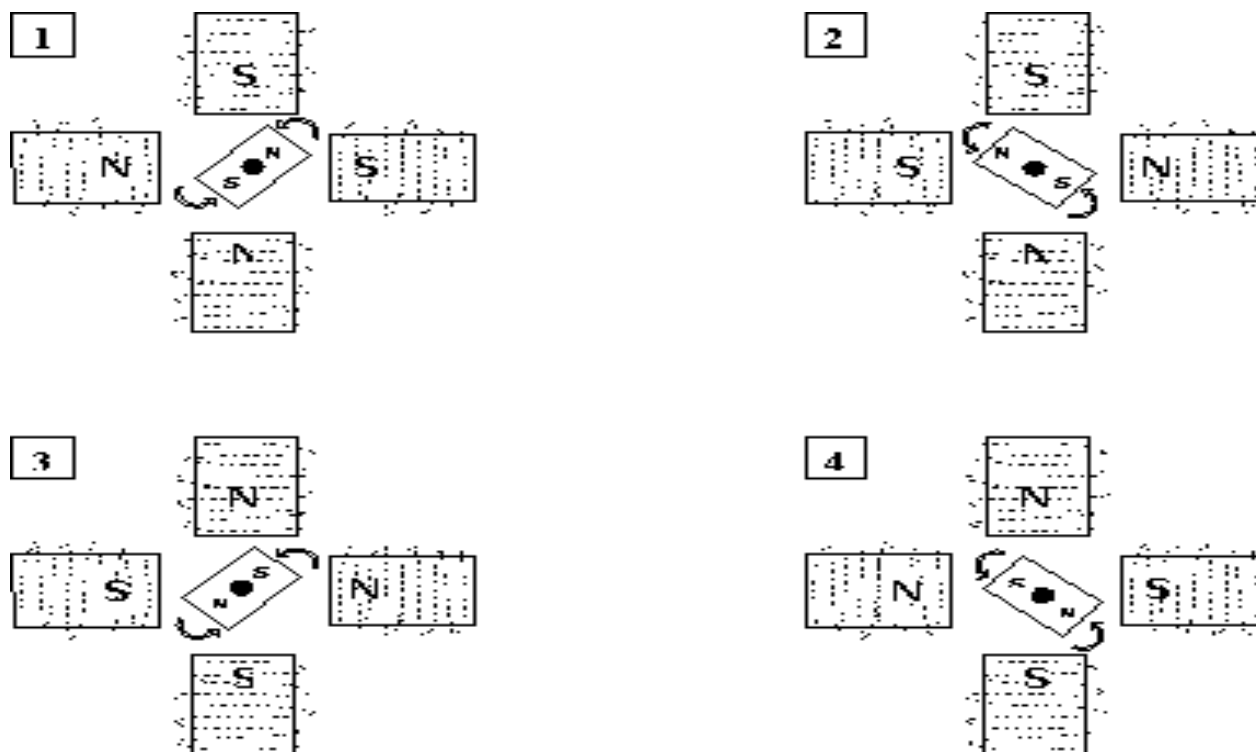
Cuando se aplica tensión a cualquiera de las cuatro bobinas existentes ésta genera un campo magnético. Ante esta situación una estría del rotor se alinea con este campo, desplazándose así un determinado número de grados. A este desplazamiento se lo denomina **paso**.



El motor Paso a Paso es un elemento capaz de transformar pulsos eléctricos en movimientos mecánicos. El eje del motor gira un determinado ángulo por cada impulso de entrada, con lo que el movimiento es muy preciso y fiable.

Estos motores están constituidos por un rotor sobre el que van aplicados distintos imanes permanentes y por un cierto número de bobinas excitadoras en su estator. Toda la conmutación (o excitación de las bobinas) deber ser externamente manejada por un controlador.

El motor Paso a Paso puede girar en los dos sentidos, y el ángulo de giro puede variar entre $0,72^\circ$ (500 pasos / 1 vuelta) y 90° (4 pasos / 1 vuelta).



El motor Paso a Paso perfecto sería el que tuviera polos infinitos, así se obtendrían giros de 0 grados.

Para permitir una mejor resolución por paso, se añaden más polos al estator, además en dichos polos se mecanizan.

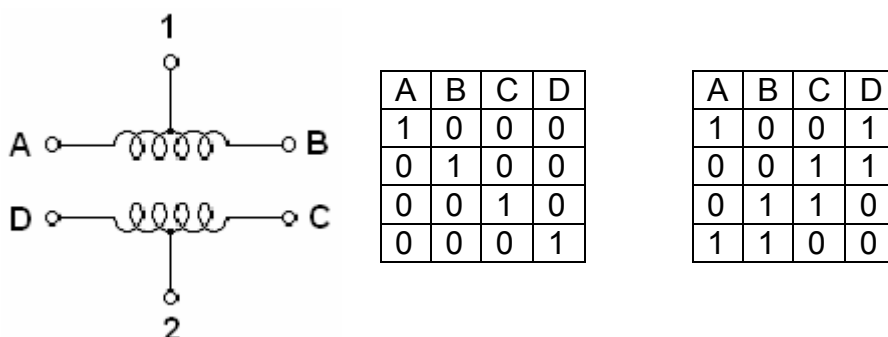
La característica principal de estos motores es el hecho de poder moverlos un paso a la vez por cada pulso que se le aplique. Este paso puede variar desde 90° hasta pequeños movimientos de tan solo 1.8° , es decir, que se necesitarán 4 pasos en el primer caso (90°) y 200 para el segundo caso (1.8°), para completar un giro completo de 360° .

Estos motores poseen la habilidad de poder quedar enclavados en una posición o bien totalmente libres. Si una o más de sus bobinas está energizada, el motor estará enclavado en la posición correspondiente y por el contrario quedará completamente libre si no circula corriente por ninguna de sus bobinas.

Existen varios tipos de motores paso a paso pero nosotros aquí nos centraremos en los motores paso a paso de imán permanente y en sus tipos, unipolares y bipolares.

- **Bipolar:** Estos tienen generalmente cuatro cables de salida. Necesitan ciertos trucos para ser controlados, debido a que requieren del cambio de dirección del flujo de corriente a través de las bobinas en la secuencia apropiada para realizar un movimiento. Como se aprecia, será necesario un Puente-H por cada bobina del motor, es decir que para controlar un motor Paso a Paso de 4 cables (dos bobinas), necesitaremos usar dos Puente-H.
- **Unipolar:** Estos motores suelen tener 6 o 5 cables de salida, dependiendo de su conexión interna. Este tipo se caracteriza por ser más simple de controlar.

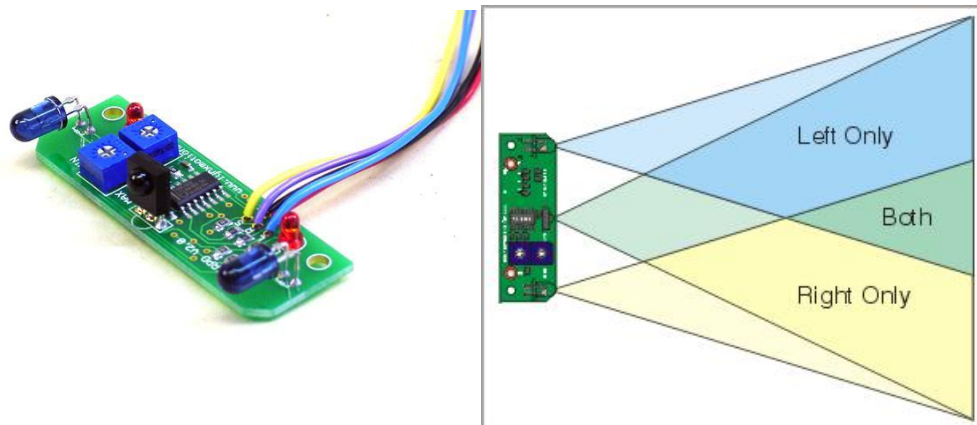
Secuencia simple y doble



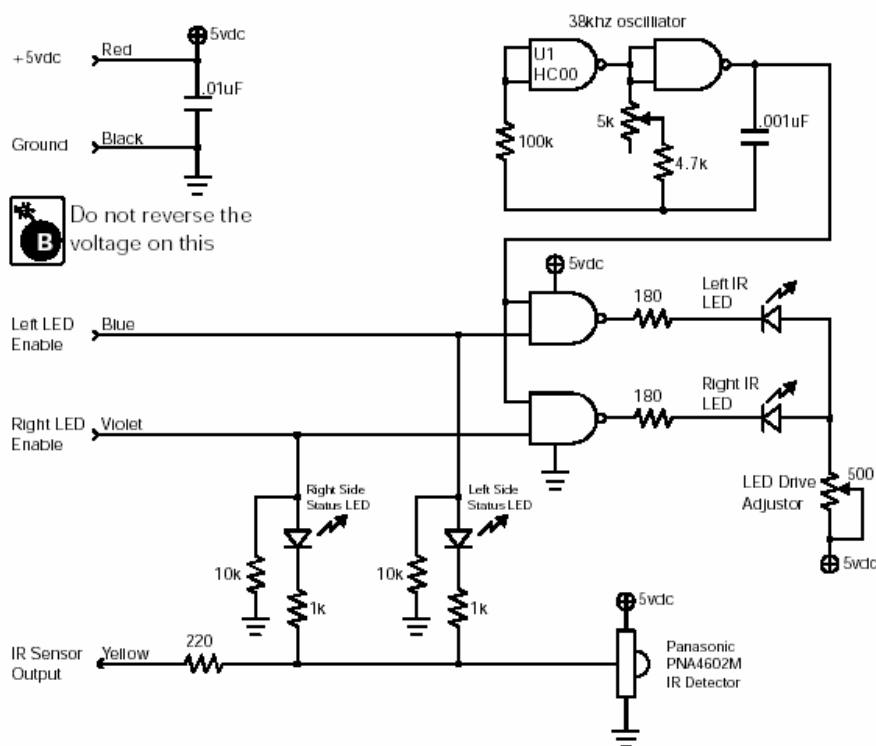
Los motores de paso se los puede encontrar en una gran variedad de voltajes, potencias y pasos, por lo mismo su rango de precio es muy amplio. Si bien en los lugares establecidos su precio es relativamente alto, se los puede comprar a bajo costo en desarmaduras o lugares de chatarra como los de la calle Carrascal. Otra fuente importante de estos motores son impresoras viejas.

SENSORES

SENSOR INFRARROJO

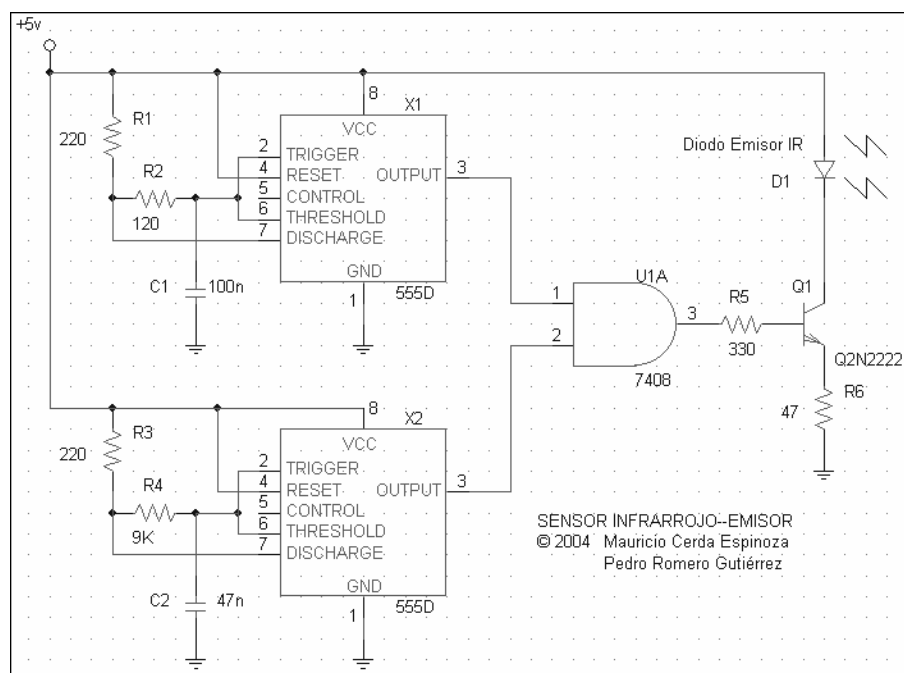


Este dispositivo, consiste en un par de leds infrarrojos orientados de manera que para un receptor infrarrojo situado entre ambos exista un rango para el cual la señal reflejada pueda solamente provenir de uno, otro rango para el cual solo pueda provenir del otro y uno donde provenga de ambos. Su modo de operación consiste en ir activando cada led por separado, nunca juntos, e ir registrando la salida. El haz emitido tiene una frecuencia de entre 38 y 42 [KHz], debido a que los módulos receptores infrarrojos operan en torno a ese rango. Por último la distancia de detección para este tipo de sensor es muy limitada, con una distancia mínima de aproximadamente 10 [cm] y una máxima de 70 [cm].

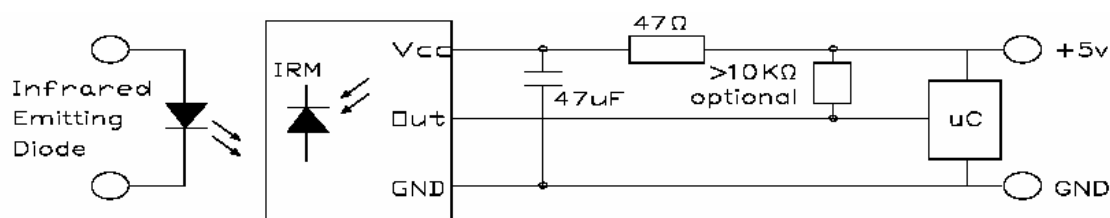


Otra alternativa de circuito es la propuesta por Mauricio Cerda y Pedro Romero.

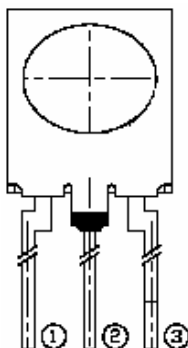
Circuito esquemático del emisor:



Circuito esquemático del receptor:



* uC es un circuito o controlador "cualquiera" que recibe la salida del receptor.
Ejemplo: Basic Stamp.



- ① OUTPUT
- ② GND
- ③ Vcc

Modo de operación del sensor (Emisor):

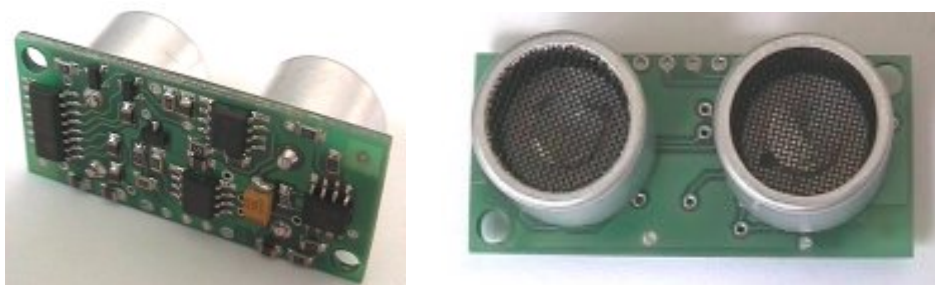
- El primer subcircuito con 555 es un reloj generador de frecuencia de 32 a 36 kHz aprox. Este se necesita para que el receptor perciba la señal que se quiere enviar, pues dicha frecuencia es la óptima de trabajo.
- El segundo subcircuito es un modulador de la onda anterior. Esto es, los pulsos de 3x kHz “viajan” en un tren de pulsos de mayor amplitud y menor frecuencia, 2 a 20 Hz aprox.
- La salida de ambos circuitos se conectan al chip 7408 que contiene 4 AND's. Éstos son operadores lógicos, que envían la salida sólo si sus dos entradas son “verdaderas” o “están en alta”.
- Todo lo anterior se realiza pues si constantemente enviamos a 38 kHz el receptor se saturará y no “sabrá” cuando está detectando o cuando no hay nada. Por lo que enviando la señal de 3x kHz periódicamente, modulada dentro de un tren de pulsos, se obtendrá una mejor respuesta y se sabrá exactamente cuando estamos en presencia de algo.
- El consumo es ≤ 5 mA aprox.

Algunas notas:

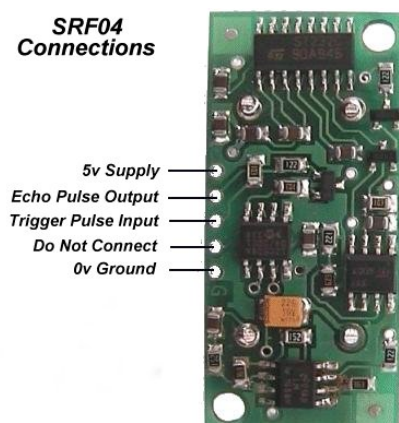
- Es muy conveniente trabajar en los subcircuitos antes de montar todo. Por ejemplo, depende de la calidad de los Led's emisores el valor de la resistencia, dando el rango de detección del sensor. Por esto, el valor de R3 y R4 puede variar, siendo recomendable el uso de un potenciómetro de 50 o 100 Kohms (altamente recomendable para R4). R4 comanda la frecuencia con que se recibe la señal (2 a 20 Hz).
- Junto con lo anterior, para verificar que los relojes funcionan, el uso de led's comunes para ver la frecuencia no está demás. Para regular el tiempo de clock, se juega con las resistencias y condensadores conectados al 555, yendo desde 100 nF, pasando por 220 o 440 nF, y llegando hasta 1, 4 o 47 uF.
- El circuito receptor se puede probar con un control remoto cualquiera. Se conecta el osciloscopio a la salida del receptor IRM. La gran mayoría de los controles remoto envía un tren de pulsos que se debería visualizar claramente en el osciloscopio.
- La calidad del sensor dependerá de la calidad de los elementos utilizados. Que sus alambres estén lo más derecho posible. Los IC deben ser manipulados con sumo cuidado (electricidad estática) para que no sufran daños. Los Led's también influyen en la calidad del sensor. Hay diferentes modelos que tienen distintos requerimientos de corriente, y por ende, de potencia. Esto último es un factor que influye en el rango de alcance.
- El color del objeto a detectar también influye en la distancia de detección. El blanco se detecta a mayor distancia que el negro.

Se los puede comprar en Lynxmotion (<http://www.lynxmotion.com>) y su precio en Estados Unidos es de USD\$ 30.

SENSOR DE ULTRASONIDO SRF04

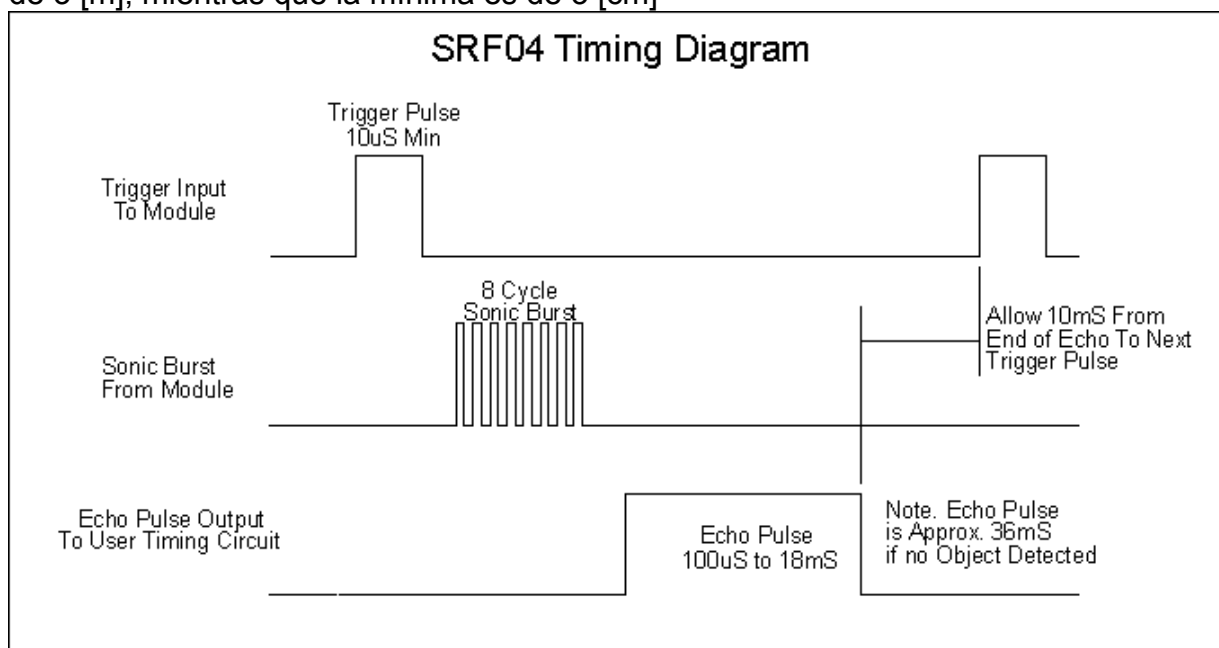


SRF04 Connections



Este sensor envía un haz de ultrasonido a una frecuencia de 40 [kHz] y luego espera por su eco. Este tipo de sensor es muy preciso y con él uno es capaz de determinar distancias con un error muy bajo. Su modo de operación es muy sencillo, uno tiene que enviarle un pulso de mínimo 10 [us] a través del pin *Trigger Pulse Input*, y luego registrar el ancho del pulso (PWM) que el dispositivo genera a través del pin *Echo Pulse Output*. Este pulso es proporcional a la distancia que demoró el sonido en ir y volver del objeto detectado, entonces conociendo la velocidad a la que viaja el

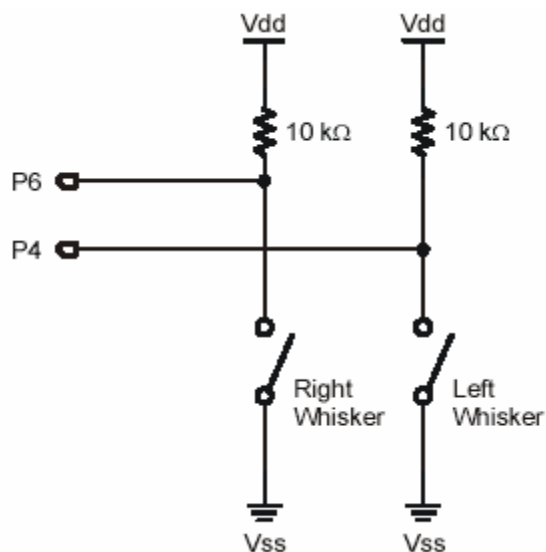
sonido, se tiene la distancia del objeto. Si el pulso generado es de 36 [ms], entonces no encontró un obstáculo hasta la distancia máxima para la cual el sensor es capaz de detectar. El SRF04 tiene una capacidad de detección máxima de 3 [m], mientras que la mínima es de 3 [cm]



Se pueden comprar en Robot Electronics (<http://www.robot-electronics.co.uk>) por £14.30 y en Acroname (<http://www.acroname.com>) por USD\$ 34.50.

SENSOR DE TACTO

El sensor de tacto es en si un interruptor on/off, la configuración más común es un interruptor pull-up. Este se basa en una configuración normalmente abierta que se cierra al momento de hacer contacto.

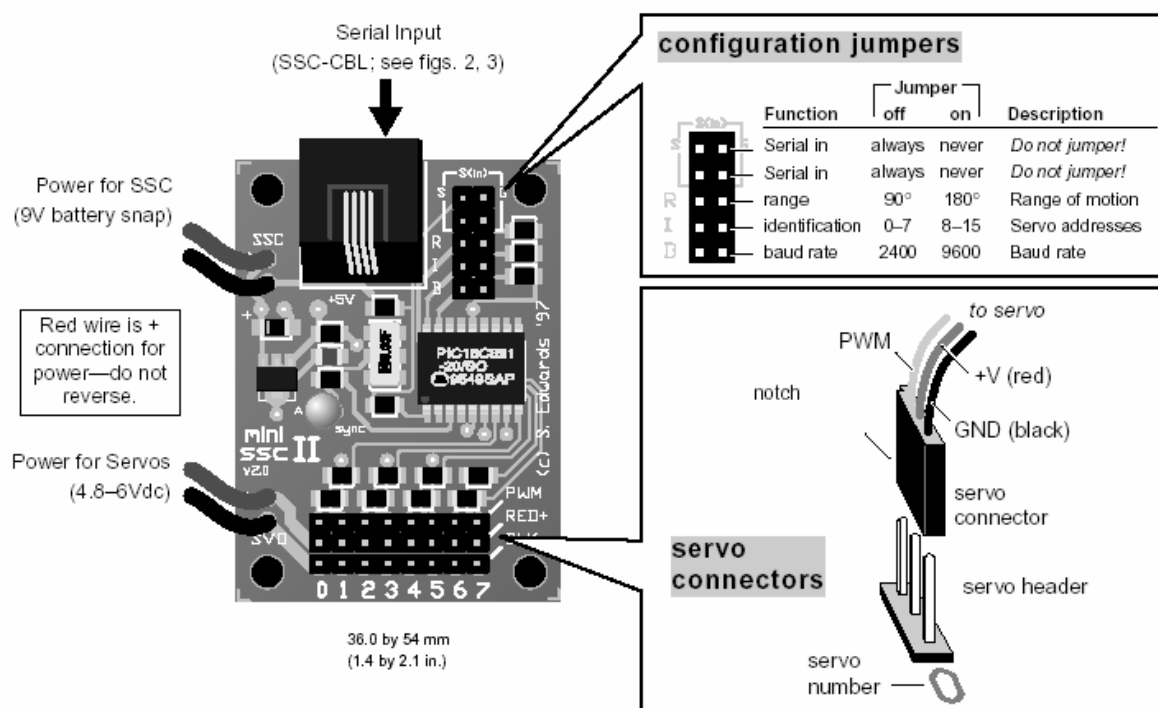


En el caso de un sensor de tacto como el anterior, este se encuentra normalmente en un nivel lógico alto y al momento de hacer contacto el nivel lógico pasa a ser bajo. Para esta configuración es importante que la resistencia sea grande para que la corriente que va a circular por la rama sea pequeña.

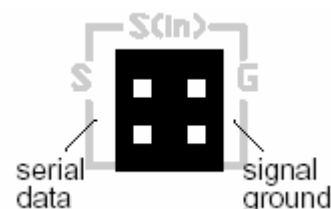
CONTROLADORES

CONTROLADOR SERIAL DE SERVO MOTORES

Un controlador serial de servo motores o SSC es una tarjeta electrónica capaz de controlar 8 servo motores mediante una conexión serial. Esta tarjeta se encarga de proveer a cada uno de los motores los pulsos necesarios para mantenerlos en una posición determinada.



La tarjeta posee un conector RJ-11 (como el de un teléfono) por donde recibe la información relativa a la posición de los motores. Además esta información puede ser ingresada a través de los pines “Serial in” ubicados junto a los jumpers (ver figuras). Para cualquiera de las dos configuraciones es necesario tener además del cable con la información, un cable con tierra que provenga de la fuente emisora de la información.



Es importante destacar que la comunicación utiliza el protocolo RS232 en modo asíncrono, por lo cual puede ser controlada desde cualquier computador o microcontrolador. La señal debe tener 8 bits de datos, no debe tener paridad y debe tener un bit de parada (8N1).

La velocidad puede ser de 9600 o 2400 bits por segundo (bps) y esta se configura en el jumper **B** (por baudio). Si dicho jumper está sin colocar la tarjeta funciona a 2400 bps y si está puesto lo hará a 9600 bps.

Existe otro jumper, el **R** (por rango), que sirve para cambiar el rango de los pulsos. Si el jumper no está puesto los pulsos tendrán un ancho mínimo de 1 [ms] y un ancho máximo de 2 [ms]. Por otro lado si está puesto estos tendrán un ancho mínimo de 0.5 [ms] y un ancho máximo de 2.53 [ms].

Esto es así porque existen algunos servo motores que tienen un rango de movimiento de $\pm 45^\circ$ para los que el jumper no debe estar puesto y otros con un rango de $\pm 90^\circ$ para los que el jumper debe estar puesto. Sin embargo para ambos tipos de servos la posición central se obtiene aplicando pulsos de 1.5 [ms].

Existe un último jumper, el **I**, con el que se puede configurar la identidad de los servos. Si este no está puesto, la identificación de los motores corresponderá al número que se encuentra bajo el conector respectivo, que para este caso tienen valores desde el 0 (cero) hasta el 7 (siete). Por otro lado, en caso de encontrarse puesto habrá que sumarle 8 al número que aparece bajo el servo respectivo. De esta manera los servos tienen identidades que van desde el 8 (ocho) hasta el 15 (quince).

Esto es útil porque si se tienen dos SSC una con el jumper **I** y otra sin, se pueden conectar a una misma línea serial y controlar los 16 motores en forma independiente.

La configuración de los jumpers se adopta al encenderse y si se cambia una vez encendida no tendrá efecto hasta que la placa haya sido apagada y vuelta a encender.

El control de la SSC se realiza enviando 3 bytes. El primero corresponde a una señal de sincronismo y es siempre el mismo, el segundo corresponde al número del motor que se quiere mover y el tercero es la posición a la que se desea mover dicho motor.

Como los bytes tienen 8 bits se pueden enviar números desde el 0 (cero) hasta el 255 (doscientos cincuenta y cinco).

El 255 es un número reservado, ya que corresponde al número de sincronismo. De esta forma tanto el segundo como el tercer byte pueden tener valores entre 0 y 254.

Byte 1	Byte 2	Byte 3
Sincronismo (255)	Motor (0-254)	Posición (0-254)

Quizás se estén preguntando ¿por qué el número del motor puede ser hasta 254? La respuesta es que existen otros modelos de SSC con otros índices, por ejemplo, sin jumper **I** tienen valores desde el 16 al 23 y con él tiene valores van desde el 24 al 31. Con ello si se conectan en paralelo, se pueden manejar un mayor número de motores en forma independiente por una misma línea de control.

En lo que respecta la posición si se envía un 0, la placa entenderá que deberá moverá dicho servo hasta el extremo izquierdo. Si se envía un 254 lo moverá hasta el extremo derecho y si se le envía un 127 moverá el servo hasta la posición central. Esto sin importar cual de las dos configuraciones de rango se esté utilizando.

Si se utiliza la que va entre 1 y 2 [ms] cada posiciones tendrán un cambio de 0.36° aproximadamente mientras que si se utiliza la que va entre 0.5 y 2.53 [ms] cada posición tendrá un cambio de 0.72° .

Al encender la SSC todos los servos son posicionados inicialmente en la posición 127, es decir, en la posición central con un ancho de pulso de 1.5 [ms].

Entonces suponiendo que se tienen dos SSC conectadas a una misma línea una con identidades 0-7 y otra con valores 8-15. Si se les envían los siguientes bytes:

255 – 0 – 0 – 255 – 8 – 254.

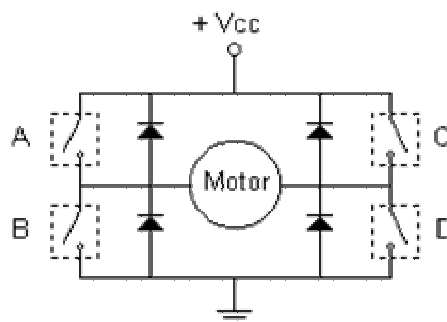
Lo que ocurrirá será que el servo conectado en la posición 0 del que no tiene jumper / se moverá hacia el extremo izquierdo y el servo conectado en la posición 0 de la placa con jumper / se moverá hacia el extremo derecho.

Por último hay que mencionar que la SSC tiene dos fuentes de alimentación. Una de entre 4.8 y 7.2 [V] que es exclusivamente para los motores y otra de 9 [V] que es exclusivamente para parte electrónica. Es importante utilizar dos fuentes porque los motores generan mucho ruido en la línea del voltaje y si se llegara a conectar la parte electrónica a esa línea tendría un comportamiento irregular.

El fabricante es Scott Edwards Electronics <http://www.seetron.com> USD\$44.00. También se puede adquirir en Lynxmotion <http://www.lynxmotion.com> por USD\$43.95.

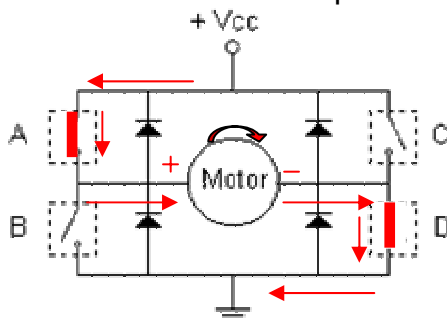
PUENTE H

Un puente H es básicamente un arreglo de cuatro interruptores acomodados de la siguiente manera:

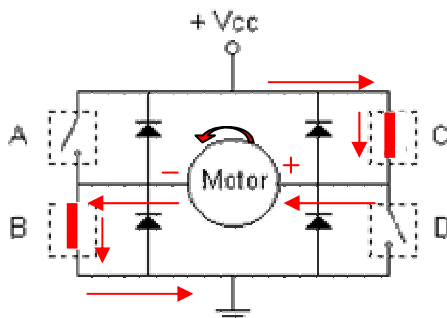


Estos interruptores (A, B, C y D) pueden ser de transistores bipolares, mosfets, jfets, relés o de cualquier combinación de elementos. El objetivo central es el de poder controlar el sentido de un motor de corriente continua sin la necesidad de aplicar voltaje negativo.

Si se cierran solamente los contactos A y D la corriente circulará en un sentido a través del motor o del elemento conectado en la parte central.



Y si se cierran solamente los contactos B y C la corriente circulará en sentido contrario.



Hay que observar también que un puente H necesita de cuatro diodos de protección para el motor.

Un puente H tiene por lo general cuatro estado de operación:

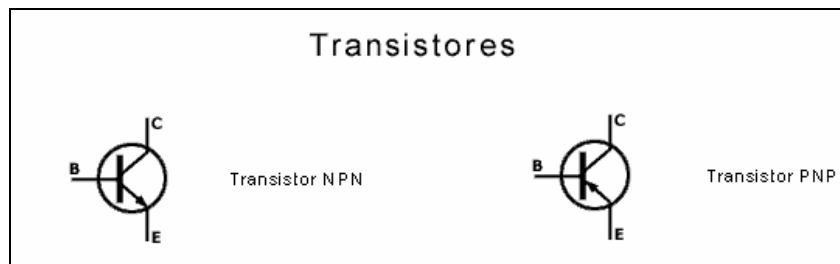
Interruptores		Salida		Función
AD	CD	AB	CD	
0	0	0	0	Motor en libertad de acción
1	0	1	0	Motor gira en un sentido
0	1	0	1	Motor gira en el otro sentido
1	1	1	1	Motor se bloqueará y frenará

Donde un 0 corresponde a un interruptor abierto o una salida sin alimentación y un 1 corresponde a un interruptor cerrado o una salida con alimentación.

Por otro lado muchas veces es necesario controlar la velocidad de un motor de continua, una manera de hacerlo es variando es el voltaje aplicado mediante modulación por ancho de pulso o PWM. Este método consiste en aplicar un tren de pulsos cuadrados con un período fijo, en el cual se va variando el ancho del pulso. Este método también se puede entender como apagar y encender en motor a una tasa muy rápida de manera de lograr una velocidad menor. El período de la señal no debe ser muy largo ya que se quiere que este tenga un movimiento continuo y no avance a “tirones”.

El transistor como interruptor

Bueno ustedes se preguntaran como es posible que un transistor pueda servir como interruptor, bueno el transistor no es fácil de explicar pero haré lo posible porque entiendan como podemos usarlo como interruptor. El transistor consta de una base un colector y un emisor, y se distribuye de la siguiente manera dependiendo si es PNP o un NPN,



Bueno lo relevante de estos es que cuando ingresamos una corriente por la base, esta puede alterar de cierta forma la corriente que pasa entre el colector y el emisor o entre el emisor y el colector (por ejemplo amplificarla), lo que mas nos incumbe a nosotros es que cuando la corriente o en su defecto el voltaje supera

cierto nivel el transistor se satura, y esto produce que la corriente pase del colector al emisor o viceversa sin alteración alguna.

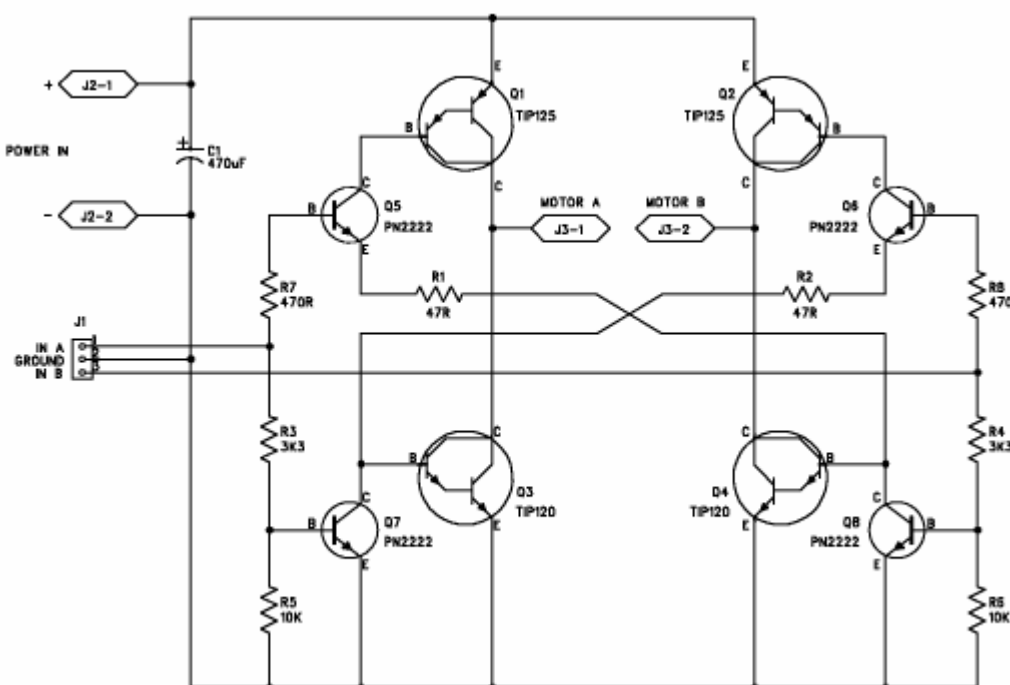
PRIMER PUENTE H:

A continuación señalaremos un tipo de puente H que es muy efectivo, ya que se puede ocupar directamente un microcontrolador y tiene componentes que soportan altas corrientes.

Los materiales que necesitamos para armar nuestro puente H son:

- 2 transistores NPN, TIP120 o en su defecto TIP31C (en este caso necesitaremos los diodos de protección)
- 2 transistores PNP, TIP125 o en su defecto TIP32C (en este caso necesitaremos los diodos de protección)
- 4 transistores pn2222
- 1 condensador 470 [μ F]
- 2 resistencias de 47 [Ω]
- 2 resistencias de 470 [Ω]
- 2 resistencias de 3 [$k\Omega$]
- 2 resistencias de 10 [$k\Omega$]
- 4 diodos zener (van a depender del voltaje que ocupen los motores)

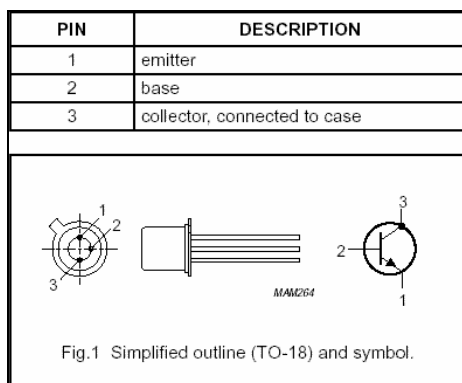
ESQUEMATICO:



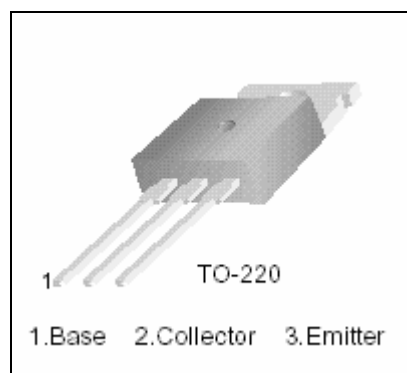
Este esquemático tiene como ventajas de que las señales de control requieren muy, pero muy poca corriente por lo que puede ser controlado desde cualquier

circuito integrado o microcontrolador directamente. Además los transistores TIP125 y TIP120 (o TIP127 y TIP122) traen incorporados los diodos de protección. Si se van a cambiar por ejemplo por los transistores TIP31C y TIP32C se deben agregar los diodos. Aunque esto es posible no es recomendable ya que estos últimos poseen una ganancia muy baja en comparación con los utilizados.

Aunque este esquemáticos puede parecerles un tanto confuso es muy fácil de implementar en el protoboard, donde ven motor A y motor B es donde va ubicado el motor, tienen que fijarse muy bien en los transistores y en como están ordenados la base, el emisor y el colector, aquí les muestro los dibujos del 2n2222 y de el tip125(los otros son iguales al TIP125).



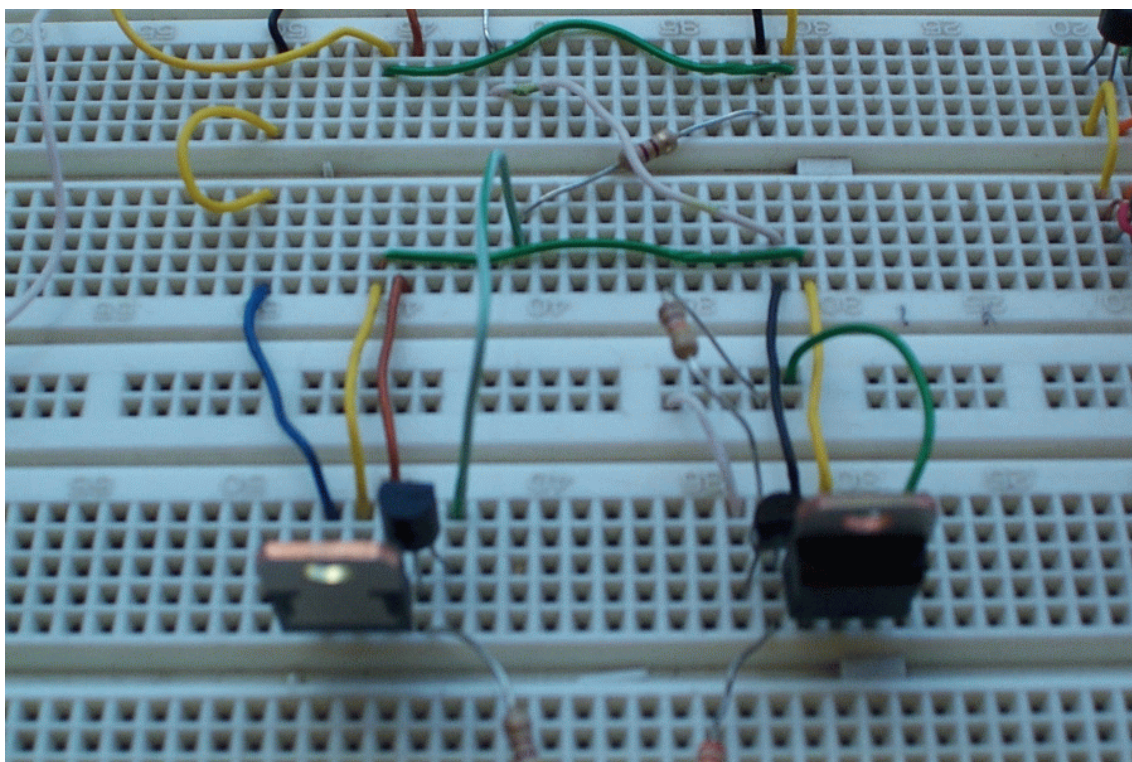
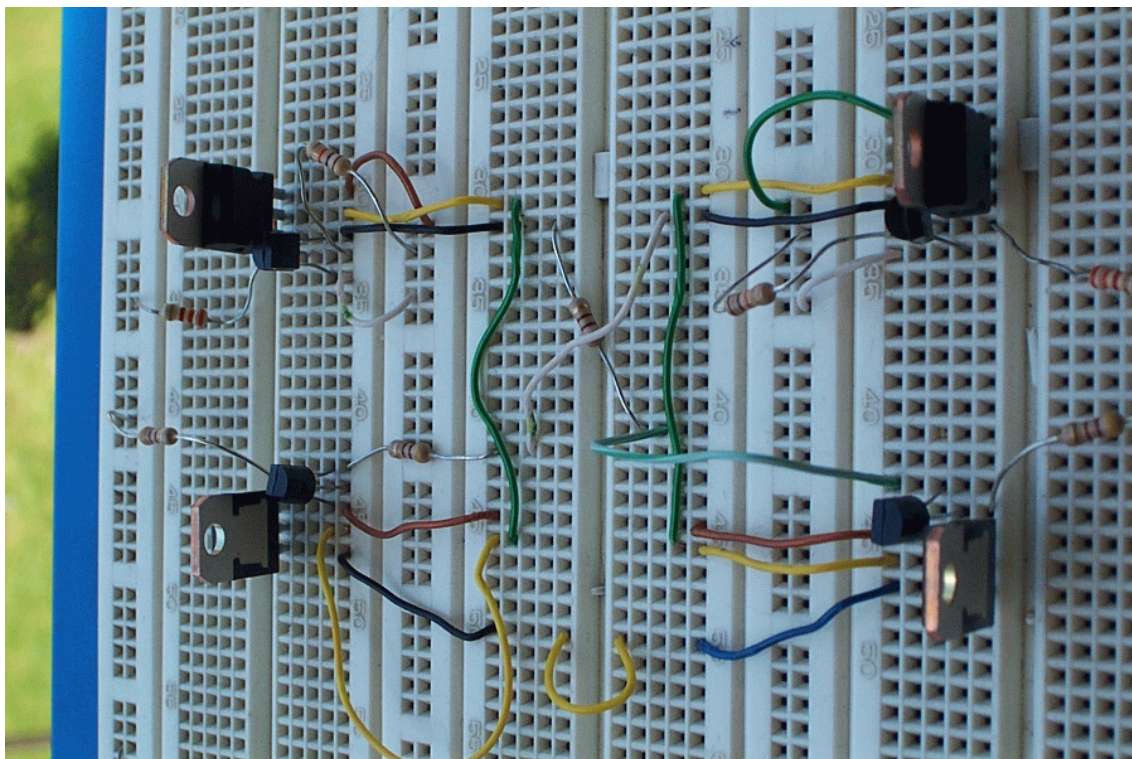
2n2222



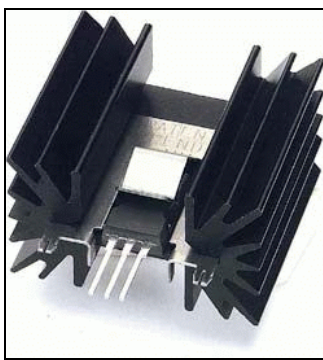
TIP125

Fíjense también que la fuente de los motores, donde dice POWER IN, el + y el - están conectados con un condensador :-S, porque??, Bueno este condensador se usa para proteger el motor del encendido, ya que este se produce una subida de voltaje y corriente que los puede dañar, no es tan necesario que lo utilicen pero si su motor es muy grande o esta muy forzado es mucho mejor, recuerden no equivocarse en la magnitud del condensador, ni tampoco en la polaridad ya que si la invierten este puede explotar.

Enseguida se adjuntan las fotos del protoboard con el puente H construido y funcionando, esto quizás les pueda servir como guía para armar un en un protoboard para luego pasarlo a una placa y soldar.



Cabe destacar que, aunque estemos ocupando estos transistores que resisten mucha corriente, puede ser que después que nuestro robot funcione un cierto tiempo, los transistores empiecen a calentarse, por esto se suele ocupar DISIPADORES cuando los transistores son sometidos a un funcionamiento continuo, que es un disipador???, Bueno, un disipador como su nombre lo dice disipa el calor que se produce en el transistor, este puede ser construido de cualquier metal puesto que estos son muy buenos conductores tanto como de la corriente como del calor, pero el que se suele usar es aluminio que es un buen conductor del calor bañado en mica que es un mal conductor de la corriente, como sabemos el calor se disipa primero por la superficie por lo cual mientras mas grande sea la superficie de nuestro disipador, mejor, el mas usado para este tipo de transistores es el disipador tipo mariposa que señalamos en la siguiente imagen.



A continuación realizaremos algunas pruebas con el puente H y distintos motores usando un controlador básico (Basic Stamp). Si en su proyecto no pueden recurrir a los puente H del laboratorio pueden armar uno en el protoboard y después pasarlo a placa puesto que es de muy poca dificultad solo necesitan ser ordenados y precavidos.

Primero realizaremos pruebas con un motor JC-35I/h 12 Volts ocupando el basic stamp como controlador. Mediremos la corriente que ocupa el motor cuando esta libre cuando es sometido a un torque (por ejemplo producido por un par de orugas) y cuando cambia de sentido con lo cual mediremos los pics de corriente que tienen que soportar los transistores.

Al hacer las pruebas llegamos a los siguientes resultados:

Corriente requerida por los transistores:

a) Motores libres:

1- 2N2222: 1.314 [mA]

2- TIP125: 80 [mA]

b) Motores forzados a un torque constante:

1- 2N2222: 1.314 [mA]

2- TIP125: 200 – 300 [mA]

c) Motores cambiando de dirección repentinamente:

1- 2N2222: 1.317 [mA]

2- TIP125: 300 [mA] (peak de corriente, cuando el motor se enciende hacia algún lado alcanza un peak de corriente)

OBS: Notamos que los transistores 2N2222 siempre requieren de la misma corriente, lo cual es el objetivo de esta configuración para el puente H, ya que los controladores como el basic stamp no dan mucho mas que eso. En cambio los transistores TIP tienen que soportar más corriente y si estas cambiando los motores de dirección continuamente esta corriente alcanza pics más altos aun.

Aquí podemos observar lo bueno de esta configuración, o sea que no forzamos los controladores para saturar los transistores (transistores en saturación dejan pasar la corriente) y hacerlos funcionar como interruptores.

También es en este caso donde ocupamos transistores que se saturan (los TIP) y transistores que amplifican (los 2N2222).

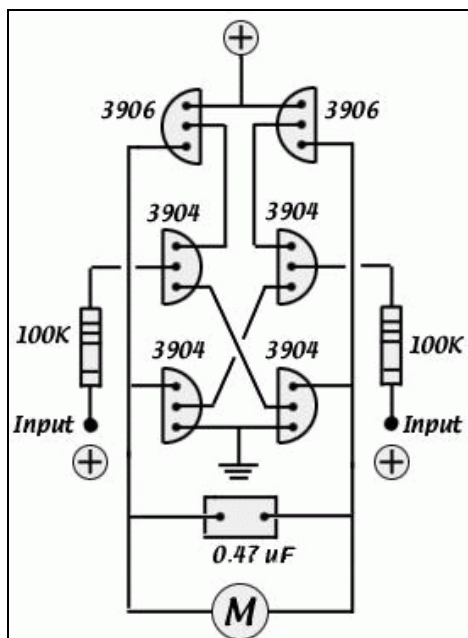
SEGUNDO PUENTE H:

Este puente H es para motores chicos que no sobrepasen los 200 [mA] o sea que no se estén forzando mucho, cualquier motor de bajo voltaje como los que vienen en juguetes, es muy importante si analizar el motor y ver cuanta corriente usan tanto cuando están libres como cuando están forzados.

Los materiales que necesitamos para armar nuestro puente H son:

- 1- 3 transistores 3904
- 2- 3 transistores 3906
- 3- 2 resistencias de 100 [k Ω]
- 4- 1 condensador de 0.47 [μ F]

ESQUEMATICO:



Cabe destacar que aquí ocupamos un servomotor de 6 [V].

Corriente requerida por el circuito:

a) Motores libres:

- 1- Basic Stamp: 60 [mA]
- 2- Transistores: 70 [mA]

b) Motores forzados a un torque constante:

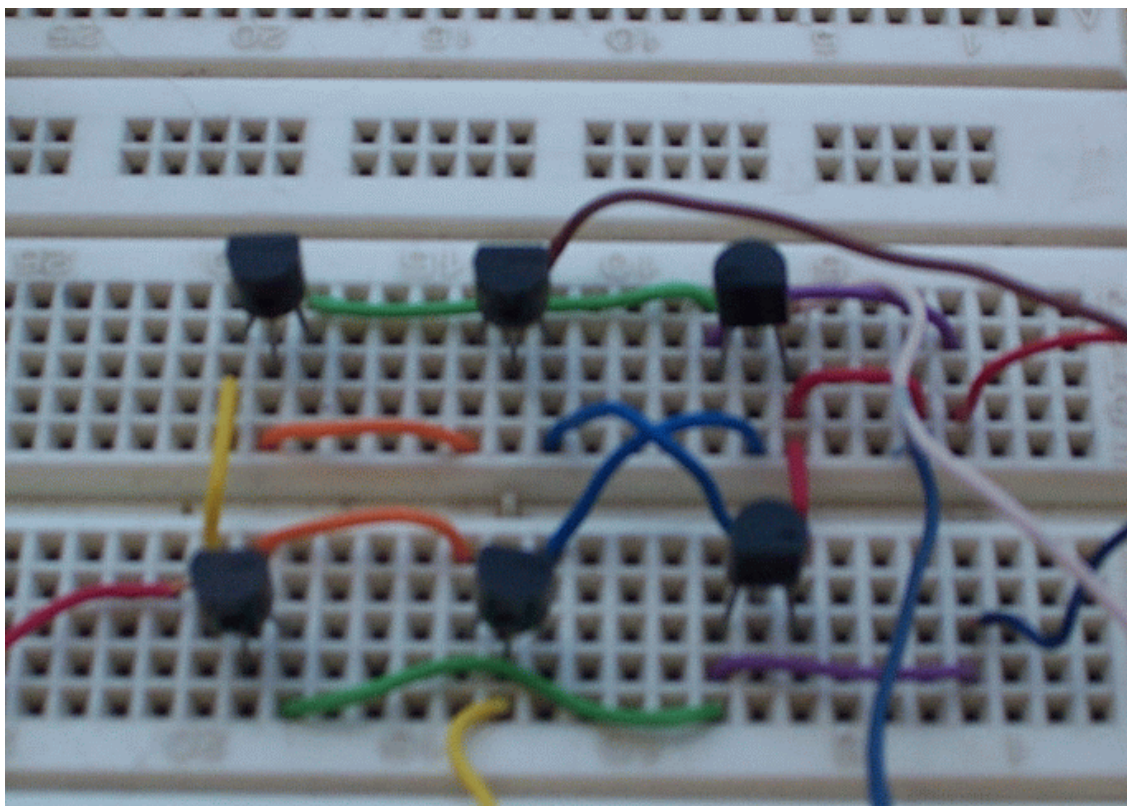
- 1- Basic Stamp: 60 [mA]
- 2- Transistores: 100-120 [mA]

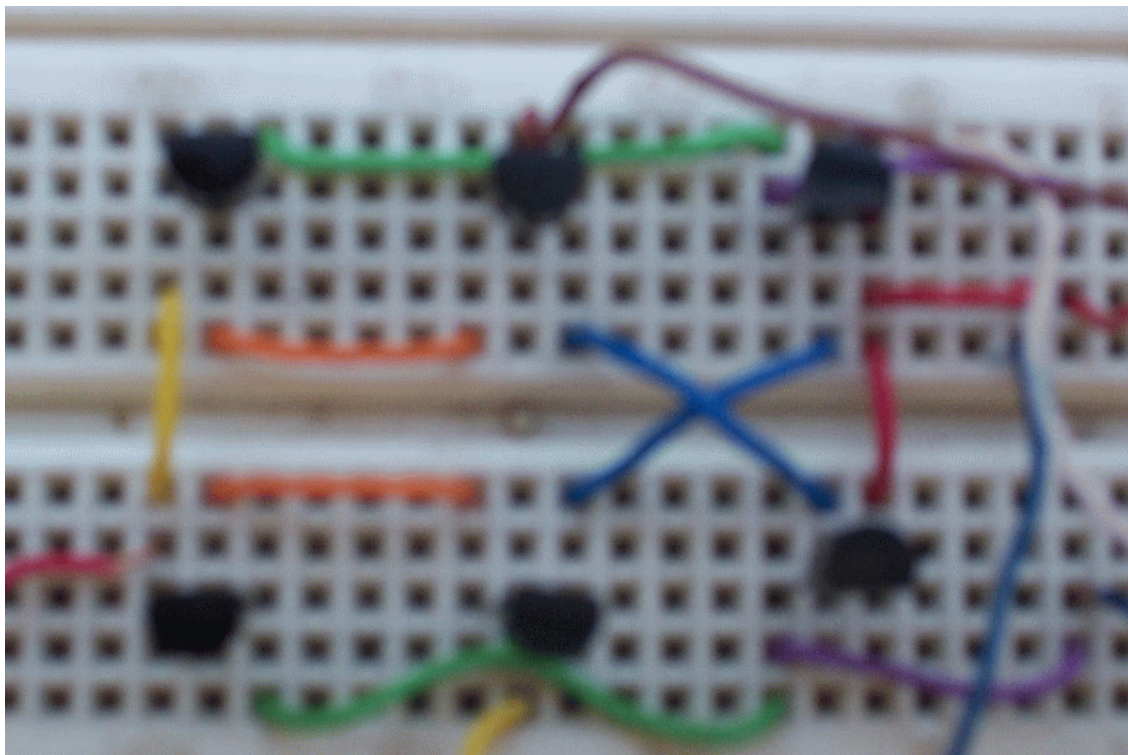
c) Motores cambiando de dirección repentinamente:

- 1- Basic Stamp: 60 [mA]
- 2- Transistores: 80 [mA] (peak de corriente, cuando el motor se enciende hacia algún lado alcanza un peak de corriente)

OBS: vemos que este motor ocupa un rango de amperaje que se adapta muy bien a este tipo de puente H, pero recuerden para motores mas grande hay que ocupar la otra configuración.

Aquí adjunto las fotos del circuito para que se guíen si es que lo tienen que hacer, como observación cabe destacar que para la basic solo usamos resistencias de 4 [Ω] ya que con mas la potencia de los motores bajaba, es aquí donde podemos notar una influencia clara de los transistores sobre la corriente que necesita el motor no como el primer puente H que hacia un trabajo perfecto.





El controlador usado para hacer esta base para Uds. fue un basic stamp, si en laboratorio no acceden a una, no importan tienen infinitas maneras mas de controlar su puente H, por el puerto paralelo, con un PIC y otros.

El código usado rutinariamente para que el motor se moviera para un lado y luego para el otro fue:

```
Loop:
HIGH 11
LOW 6

PAUSE 1000

LOW 11
HIGH 6

PAUSE 1000

GOTO Loop
```

Como vemos con este mismo código podríamos dar un tren de pulsos al la stamp para controlar la velocidad de alguno de los motores y por ejemplo no hacer doblar a nuestro tanque con un motor entero hacia adelante y el otro entero hacia atrás sino que disminuyendo la velocidad de alguno de ellos.

```
Square VAR Byte
StepSize VAR Byte
Square1 VAR Byte
```

```
StepSize = 1
Square = 1
Square1 = 1
```

```
FOR Square = 1 TO 500 STEP StepSize
  HIGH 11
  PAUSE 120
  LOW 11
  PAUSE 120
  StepSize = StepSize + 1
NEXT
```

```
FOR Square1 = 1 TO 500 STEP StepSize
  HIGH 6
  PAUSE 120
  LOW 6
  PAUSE 120
  StepSize = StepSize + 1
NEXT
```


CONTROLADOR DE MOTORES DE PASO O STEPPER

Existen diversas formas de controlar los motores stepper: mediante microcontroladores (cpu's), por computador mediante puerto serial o paralelo, o a través de controladores diseñados para este propósito. Se pondrá como ejemplo uno de estos últimos.

Los materiales necesarios para controlar un motor son:

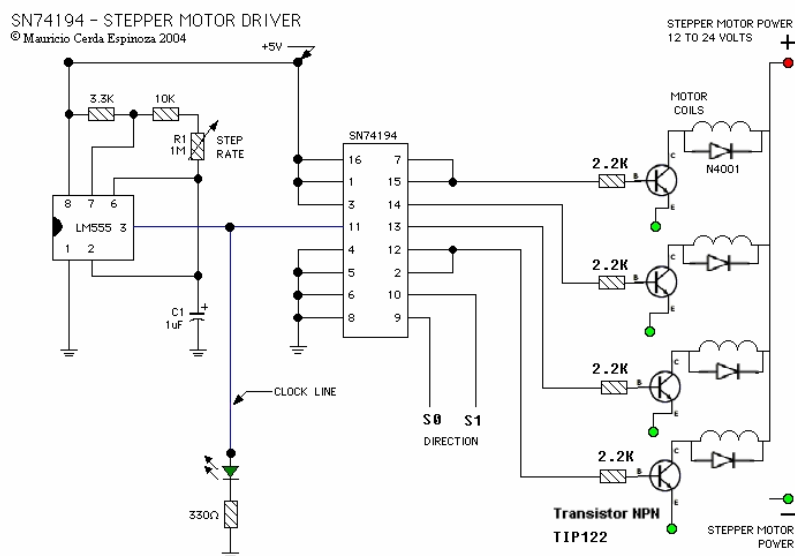
- Protoboard
- Fuente de poder
- Multímetro
- 1 chip 74194 (4-Bit Bidirectional Shift Register)
- 4 transistores TIP122
- 4 resistencias de 2.2 KOhms
- 1 resistencia de 3.3 KOhms
- 1 resistencia de 10 KOhms
- 1 potenciómetro de 100 K a 1 MOhm
- 4 diodos 4001 o 4007
- 1 chip 555

Introducción:

Un controlador de motores de paso sería muy útil en el caso de necesitar alta precisión en el giro del motor. La gracia de estos motores es esa: su precisión de giro junto con un torque nada despreciable.

El IC 74194 (cambio bidireccional de registros) permite, según el diseño del circuito, controlar en especial motores de paso unipolares (5 a 6 cables), aunque se ha probado con éxito en los otros tipos. El chip proporciona los controles básicos de avance, retroceso, parada y velocidad variable.

El control de velocidad está dado por un potenciómetro que ajusta la frecuencia de pulsos (ciclos de reloj) entregado por el IC oscilador 555.

Circuito esquemático:Modo de operación del controlador:

- El oscilador 555 produce una serie de pulsos que se conectan al pin clock del 74194.
- Cada vez que el tren de pulsos está en Alta, en las salidas de 74194, el registro introducido inicialmente comienza a ser intercambiado hacia la derecha o izquierda, dependiendo de la configuración de los pines S0 y S1. Mientras uno de ellos tenga voltaje > 0 y el otro tenga voltaje $= 0$ habrá intercambio. Cuando ambos están en $V = 0$ se produce detención.

Algunas notas:

- Con $C1 = 1\mu F$ y $R1$ (potenciómetro) = "0" la frecuencia de Clock es 100Hz aprox. Esta frecuencia será lenta para la mayoría de los motores, dependiendo mucho del modelo del motor utilizado.
- Para disminuir la velocidad de clock por motivos prácticos de prueba, $C1 = 5\mu F$ es un buen valor.
-

Para aumentar la velocidad, una vez que el prototipo funcione aceptablemente, unos 220 o 440 nF para $C1$ es aconsejable, pero no siempre fiable! Ya que depende si el motor "acepta" tan alta frecuencia de control. (Se debe saber que el 74194 soporta hasta 35 MHz de clock aprox.) Se recomienda buscar el data-sheet del motor para conocerlo mejor.

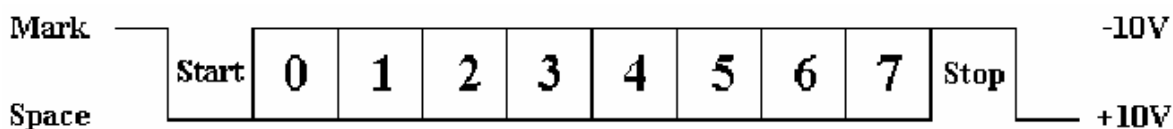
MANEJO DE PUERTOS DE UN PC

DESCRIPCIÓN DEL PUERTO SERIAL

El puerto serial, como su nombre lo indica envía su información de manera serial, es decir, como un tren de pulsos, utilizando el protocolo RS-232. Para la transmisión de información solo son necesarios 3 pines, uno a través del cual se envía la información, otro a través del cual se recibe y otro como referencia de voltaje o tierra. Pero el puerto serial posee 9 pines, los 5 restantes son para el control de datos, petición de información, libre para enviar, etc.

Una característica de este puerto que NO se debe olvidar son los valores de voltaje que utiliza para sus niveles lógicos. Un 0 (cero) lógico corresponde a un voltaje de entre +3 y +25 volts, mientras que un 1 (uno) lógico va de -3 a -25 volts. Esto es importante ya que si se desea interfacear el puerto con algún circuito TTL o CMOS se debe adaptar el valor del voltaje. El circuito integrado más utilizado para esta labor es el MAX232 aunque recomiendo utilizar el MAX233 que es igual al anterior pero no requiere de los 4 condensadores externos.

Por último para el envío de información es necesario que tanto el emisor como el receptor estén configurados para trabajar a la misma tasa de transferencia, ya que la comunicación es asíncrona y la señal de reloj no es enviada con la información. Para lograr la sincronización el puerto utiliza un protocolo el cual envía un bit de partida, el cual no es configurable. Lo que sí es configurable es el bit de parada, la paridad y el número de bits. La configuración más utilizada es la 8N1 (8 bits de información, sin paridad y 1 bit de parada).



D-Type-25 Pin No.	D-Type-9 Pin No.	Abbreviation	Full Name
Pin 2	Pin 3	TD	Transmit Data
Pin 3	Pin 2	RD	Receive Data
Pin 4	Pin 7	RTS	Request To Send
Pin 5	Pin 8	CTS	Clear To Send
Pin 6	Pin 6	DSR	Data Set Ready
Pin 7	Pin 5	SG	Signal Ground
Pin 8	Pin 1	CD	Carrier Detect
Pin 20	Pin 4	DTR	Data Terminal Ready
Pin 22	Pin 9	RI	Ring Indicator

MANEJO DEL PUERTO SERIAL DESDE C

El siguiente programa verifica si existe información recibida y si existe la despliega en pantalla, además envía lo que se tipéa.

```
#include <dos.h>
#include <stdio.h>
#include <conio.h>
#define PORT1 0x3F8

/* Define la dirección del puerto Serial */
/* COM1 0x3F8 */
/* COM2 0x2F8 */
/* COM3 0x3E8 */
/* COM4 0x2E8 */

void main(void)
{
    int c;
    int ch;

    _outp(PORT1 + 1 , 0); /* Apaga las interrupciones - Port1 */

    /* PORT 1 - Configuración de la Comunicación */

    _outp(PORT1 + 3 , 0x80); /* SET DLAB ON */
    _outp(PORT1 + 0 , 0x18); /* Set Baud rate - Divisor Latch Low Byte */

    /* Default 0x03 = 38,400 BPS */
    /* 0x01 = 115,200 BPS */
    /* 0x02 = 56,700 BPS */
    /* 0x06 = 19,200 BPS */
    /* 0x0C = 9,600 BPS */
    /* 0x18 = 4,800 BPS */
    /* 0x30 = 2,400 BPS */

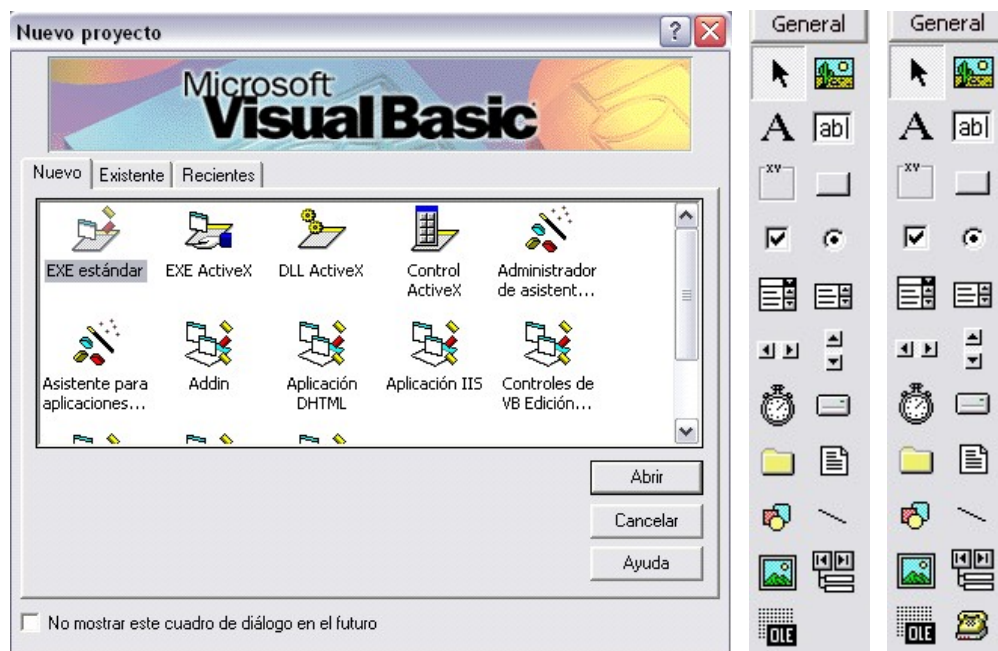
    _outp(PORT1 + 1 , 0x00); /* Set Baud rate - Divisor Latch High Byte */
    _outp(PORT1 + 3 , 0x03); /* 8 Bits, No Parity, 1 Stop Bit */
    _outp(PORT1 + 2 , 0xC7); /* FIFO Control Register */
    _outp(PORT1 + 4 , 0x0B); /* Turn on DTR, RTS, and OUT2 */

    //Comienza el programa

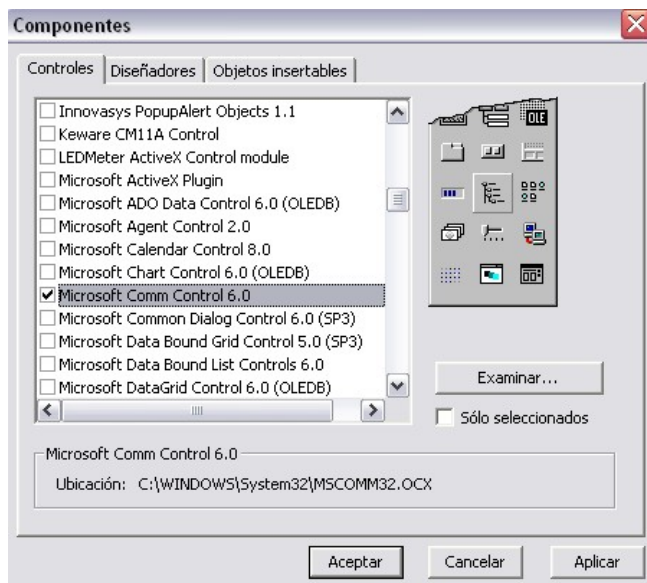
    printf("\nComunicación del Puerto Serial. Presione ESC para salir \n");
    do {
```

```
c = _inp(PORT1 + 5); /* Verifica si se ha recibido algo */
if (c & 1)
{
    ch = _inp(PORT1); /* Es caso de recibir, guarda el Char */
    printf("%c ",ch+48); /* Imprime el Char en pantalla */
}
if (kbhit())
{
    ch = getch(); /* Si se presiona una tecla, guarda el Char */
    _outp(PORT1, ch-48);/* Envía el Char al puerto Serial */
}
} while (ch !=27); /* Sale cuando ESC (ASCII 27) es presionado */
}
```

MANEJO DEL PUERTO SERIAL DESDE VISUAL BASIC



Una vez creado un nuevo proyecto, lo primero que se debe hacer es agregar el componente “Microsoft Comm Control” (icono de teléfono) a la barra de herramientas.



Para ello es necesario pulsar el botón derecho sobre el menú, irse a la opción componentes.

En la ventana que se abre se pueden agregar componentes adicionales a los que ya existen y entre ellos se encuentra el que controla los puertos Comm.

Es importante que antes de utilizar el componente se le configure la velocidad de transferencia o baudios, la paridad, la cantidad de bits de datos y el puerto a utilizar. Esto puede ser hecho en las propiedades del componente o bien al momento de cargar la aplicación o ventana mediante el siguiente código.

```
Private Sub Form_Load( )  
    MSComm1.Settings = "9600,N,8,1" ' Baud rate=9600, No parity, 8-bits data, 1  
    stop-bit.  
    MSComm1.CommPort = 1           ' We will be using Com1 as the default com  
    port.  
    MSComm1.PortOpen = True        ' Open the port for use.  
End Sub
```

En este código además se habilita (abre) el puerto. Es importante abrir el puerto antes de ocuparlo y cerrarlo después de ocuparlo. El no realizar estas operaciones ocasiona que se caiga el programa e incluso el bloqueo del computador.

Para cerrar el puerto basta agregar el siguiente código bajo el botón de cerrar o bien en Form_Unload.

```
If MSComm1.PortOpen = True Then  
    MSComm1.PortOpen = False  
End If
```

Finalmente solo resta enviar o recibir información a través del puerto. Esto se realiza con las opciones output e input del componente MSComm.

Por ejemplo para enviar los números 255, 0, 127 que sirven para fijar la posición del motor cero de una SSC a su posición central se debe utilizar el código:

```
MSComm1.Output = Chr$(255)  
MSComm1.Output = Chr$(0)  
MSComm1.Output = Chr$(127)
```

A continuación se presenta un ejemplo para recibir a través del componente.

```
Dim InString as String
```

```
MSComm1.InputLen = 0  
If MSComm1.InBufferCount Then  
    InString = MSComm1.Input  
End If
```


DESCRIPCIÓN DEL PUERTO PARALELO

El puerto paralelo como su nombre lo indica envía información de manera paralela, utilizando el protocolo Centronics. Consta de 25 pines de los cuales 8 son de salida, 5 son de entrada, 4 son de control, y 8 son tierra. Por lo general utiliza niveles lógicos TTL, donde un 1 (uno) lógico son +5 volts y un 0 (cero) lógico son 0 volts. Esta característica hace que sea muy fácil de interfacear con circuitos, sin embargo la corriente que uno le puede sacar o inyectar no es mucha y varía considerablemente de puerto en puerto, pero como idea en algunos puede ser de 4 [mA] y en otros de 20 [mA], por lo que es siempre recomendable utilizar un buffer o un optoacoplador.

Es importante mencionar que el puerto paralelo retiene la última información enviada, por lo que es uno quien tiene que borrar su salida en caso de ser necesario. Esta propiedad es sumamente útil si lo que se desea es controlar motores de paso desde un computador.

Pin No (D-Type 25)	Pin No (Centronics)	SPP Signal	Direction In/out	Register	Hardware Inverted
1	1	nStrobe	In/Out	Control	Yes
2	2	Data 0	Out	Data	
3	3	Data 1	Out	Data	
4	4	Data 2	Out	Data	
5	5	Data 3	Out	Data	
6	6	Data 4	Out	Data	
7	7	Data 5	Out	Data	
8	8	Data 6	Out	Data	
9	9	Data 7	Out	Data	
10	10	nAck	In	Status	
11	11	Busy	In	Status	Yes
12	12	Paper-Out PaperEnd	In	Status	
13	13	Select	In	Status	
14	14	nAuto-Linefeed	In/Out	Control	Yes
15	32	nError / nFault	In	Status	
16	31	nInitialize	In/Out	Control	
17	36	nSelect-Printer nSelect-In	In/Out	Control	Yes
18 - 25	19-30	Ground	Gnd		

MANEJO DEL PUERTO PARALELO DESDE C

Manejar el puerto paralelo desde C es muy sencillo, para ello lo único que se debe hacer es definir la dirección del puerto.

Programa que envía y retiene el número 14 (binario 0000 1110).

```
#include <dos.h>
#include <stdio.h>
#include <stdlib.h>
#include <conio.h>
#define PORT1 0x378

void main()
{
    printf("Mandando %d\n",14);
    _outp(PORT1,14);
}
```

MANEJO DEL PUERTO PARALELO DESDE VISUAL BASIC

Para poder manejar el puerto paralelo desde Visual Basic, se debe contar con una librería de manejo de puertos. Existen varias de ellas en internet, en este manual nos basaremos en la librería inpout32.dll.

Para poder utilizarla es necesario contar con la librería y su módulo asociado. Estos se pueden conseguir en:

<http://mecatronica.li2.uchile.cl/material.html>

<http://www.logix4u.net>

<http://www.lvr.com/parport.htm>

Luego se deben copiar la librería (inpout32.dll) y módulo (inpout32.bas) a la carpeta de trabajo y desde el proyecto importar el módulo "inpout32.bas" que declara la librería a utilizar. Ello se hace seleccionando la opción "Agregar Módulo" bajo el menú "Proyecto".

Con esto y sabiendo la dirección del puerto estamos listos para trabajar con ellos.

Para enviar se utiliza el siguiente código: Out PortAddress, ValueToWrite

Ej:

Out &h378, &h55 'Envia el byte 01010101 por el puerto paralelo

Para recibir se utiliza en siguiente código: ValueRead = Inp(PortAddress)

Ej:

Dim ValueRead As Integer
ValueRead = Inp(&h378)