UTN - FRSN

Técnicas digitales III

Proyecto

BRAZO ROBÓTICO

Integrantes:

- Bianco, Pablo Sebastián
- Gutiérrez Segovia, Juan Carlos
- Rodríguez, José María
- Ventura, Federico

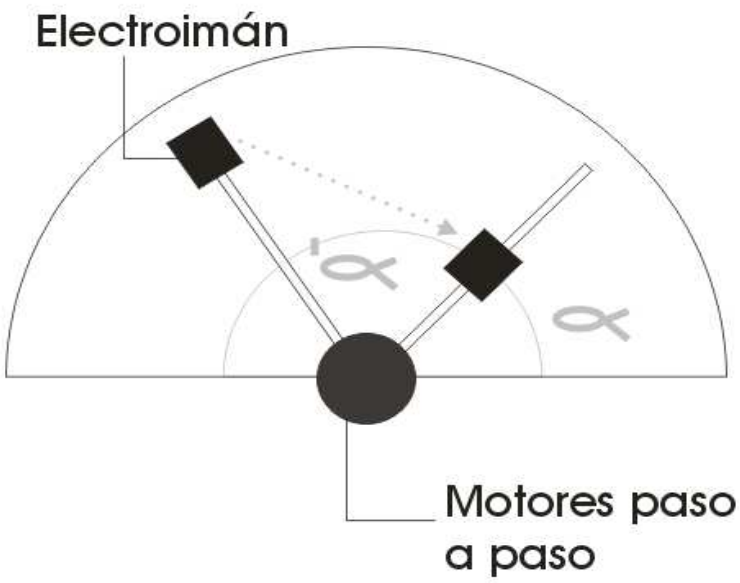
Docentes:

- Profesor: ing. Felipe Poblete
- Auxiliar: Mariano González

INDICE

OBJETIVOS DEL PROYECTO:

El proyecto consiste en mover una pieza metálica de un lugar a otro dentro de un semicírculo. Las coordenadas polares (ángulo y radio) se trasmiten en forma serial desde una PC a un PIC por medio de una interfaz RS-232.



MATERIAS INTEGRADAS:

- Elec. aplicada (Control, Potencia).
- Informática 2 (turbo C++).
- Maquinas e Instalaciones Eléctricas (motores paso a paso).
- Medio de enlace (electromagnetismo).
- Téc. digitales 2 (microcontroladores).

BIBLIOGRAFÍA:

- Enciclopedia Cekit microcontroladores.
- Internet (Micro 16f84, Transmisión RS232, motores paso a paso).
- Turbo C++ (Manual).

PROFESORES ENTREVISTADOS:

- Fernando Albertario (Pic).
- Germán Godziewski (motor paso a paso).
- Luciano Cullen (transmisión rs232).
- Ramiro Vota (Turbo C++)

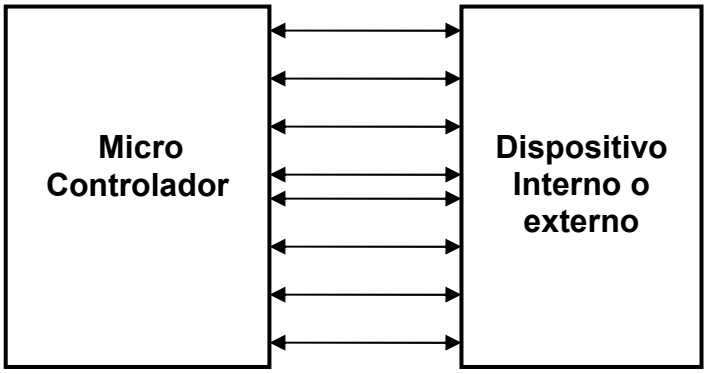
POSIBLES APLICACIONES:

- Mover objetos en forma secuencial.
- Objetivo didáctico.

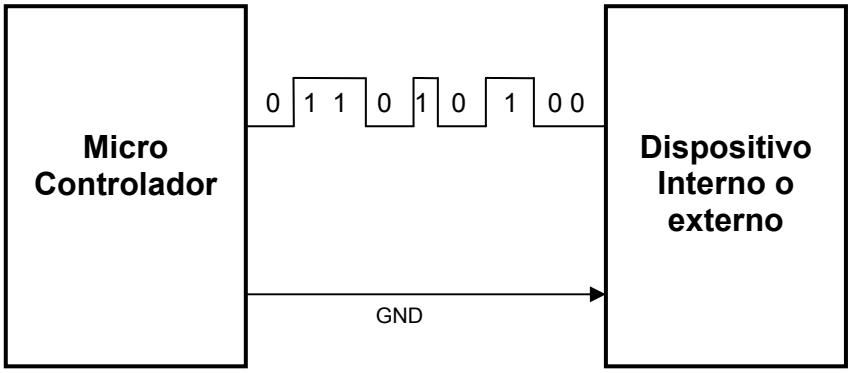
INTRODUCCIÓN

TRANSMISIÓN DE DATOS

Comunicación en paralelo: Se envían los datos simultáneamente. Es rápida pero ocupa muchos pines del microcontrolador.



Comunicación serie: Se envían bits uno detrás de otro. Es lenta pero ocupa pocos pines, por lo que permite utilizar otras funciones con los pines libres.



Los niveles de voltaje que se usan en el estándar RS-232, el equivalente en niveles lógicos (TTL) y la terminología correspondiente, se muestran en la siguiente Tabla.

Voltaje	Lógico	Control	Terminología
+3[v] a +25[v]	0	Activo	Espacio
-3[v] a - 25[v]	1	Inactivo	Marca

De la Tabla se observa que un "1" lógico, equivale a un voltaje negativo (-3v a -25 v), y un "0" lógico, equivale a un voltaje positivo (+3v a +25v). Un voltaje que está entre +3v y -3v se considera como indeterminado.

Cuando la línea se mantiene en "1" (Marca), está en estado de reposo. Cuando la línea está en "0" (Espacio) se toma como activa.

La velocidad a la que se envían datos en forma serial a través de una línea de comunicación, se denomina velocidad en baudios. La velocidad de baudios es expresada en unidades de bits por segundo. Una conexión RS-232 con velocidad de 1200 baudios tiene la capacidad de enviar 1200 bits de datos en 1 segundo.

Si se pueden enviar 1200 bits en un segundo, como máximo, el inverso de 1200 dará como resultado el tiempo de bit (período de un bit).

$$\frac{1}{(Velocidad\ de\ baudios)} = tiempo\ de\ bit = \frac{1}{1200} = 833\,\mu s$$

Si un receptor y un transmisor se conectan a 1200 baudios, el transmisor enviará bits de datos cada 833us, y el receptor tomará lectura de los bits de datos cada 833us.

En este proyecto se emplea la comunicación serial asíncrona, en la cual la transmisión está inactiva en el estado de Marca (1 lógico). La transmisión de cada carácter en una línea de comunicación asíncrona va precedida de un bit de inicio. El bit de inicio es un Espacio (0 lógico) con duración igual al tiempo de bit. En el receptor, cuando la línea cambia de Marca a Espacio se interpreta como el bit de inicio, después de este bit se reciben los bits de datos con un tiempo de bit igual a 833us, si la transmisión es a 1200 baudios.

Después de que el último bit de datos ha sido enviado, el transmisor pasa al nivel de Marca durante un tiempo de bit. Este bit es llamado bit de paro. El bit de paro indica que todos los bits de datos han sido enviados y la transmisión del carácter se ha completado. Si el receptor detecta un bit de inicio y después los bits de datos pero no detecta el nivel de Marca al final, esto indica un error en la transmisión.

En la Figura 1 se muestran los niveles de voltaje con el estándar RS-232, cuando se transmite un "0" ASCII, observe que después del bit de inicio se envía el bit menos significativo (LSB) del dato.

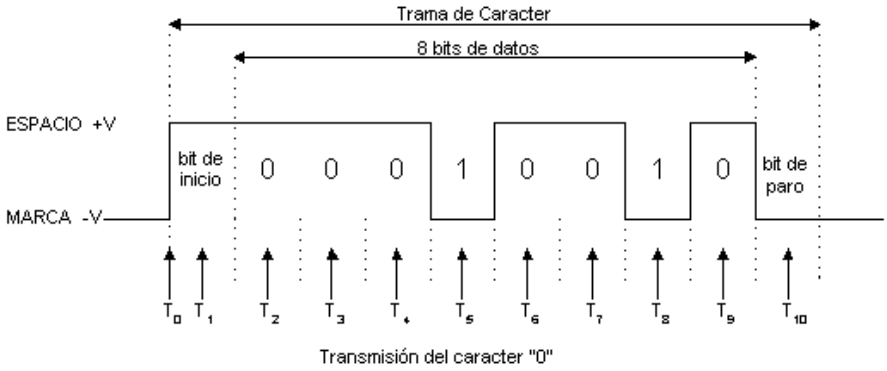


Figura 1.

T_0: La transición del nivel marca a espacio indica al receptor que la transmisión de un nuevo carácter ha comenzado.

T_1: El receptor espera la mitad del tiempo de bit (a 1200 baudios este tiempo es 416us) y toma otra muestra de la línea. Si la línea sigue en el nivel espacio, el bit de inicio es valido. En otro caso, si la línea de recepción regresa al nivel de marca, se trata de un bit de inicio falso que se atribuye a una línea ruidosa.

T_2: El receptor espera un tiempo de bit y muestrea la línea de entrada, el nivel será el correspondiente al bit menos significativo.

T_3-T_9: Se realizan 7 muestreos más, cada 833us (para 1200 baudios), y se obtienen los niveles correspondientes a los bits de datos restantes. Después de T9 los 8 bits de datos han sido capturados.

T_{10}: Se muestrea el bit de paro, observe que la línea regresa al nivel de marca.

Microcontroladores

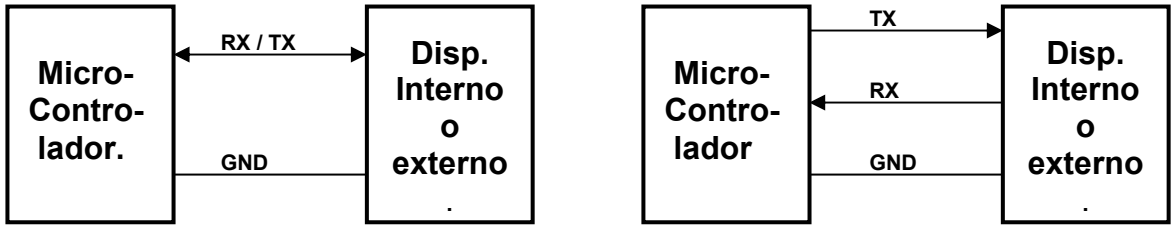
- **Alto nivel:** Tienen módulos internos de comunicaciones con sus pines especializados para esta función y sus instrucciones de programación (más fácil).

- **Bajo nivel:** La comunicación se hace por los pines de entrada/ salida (I/O) ó puertos y los protocolos se establecen en el programa.

Las comunicaciones seriales

- **Unidireccionales**: (half duplex): Transfiere datos de a una vez en una sola dirección y se utiliza un cable para señal y otro para masa. (se envía y luego se recibe)

- **Bidireccionales**: (full duplex): Envía y recibe a la vez y tiene un mínimo de tres cables; enviar, recibir y masa.



Las comunicaciones pueden ser sincrónicas o no sincrónicas. Un factor importante que se debe conocer en una comunicación serial es la velocidad de transmisión de los datos (bits/seg.) ó (baudios) que es la velocidad con que cambian los estados de la señal (de 1 a 0 ó de 0 a 1).

Para que los equipos se comuniquen satisfactoriamente ambos deben manejar un mismo conjunto de normas conocido como protocolo.

En la comunicación serial, los grupos de datos se descomponen en bits que se transmiten uno a uno de emisor a receptor, donde se establece de nuevo el grupo original.

La unidad mínima de información compuesta de varios bits se llama carácter. Si el emisor genera unidades de información de más de 8 bits, como el microcontrolador PIC tiene una CPU de 8bits, se deben dividir las unidades de información en varios segmentos de 8bits. El receptor ensambla nuevamente los segmentos y recupera el mensaje original, se trata a cada segmento de 8bits como un carácter. Ejemplo es el conversor A/D de 10, 12 ó mas bits.

Al transmitir caracteres en forma serial, los bits se envían de manera secuencial, distribuidos en el tiempo, cuando el transmisor envía estos bits, el receptor debe ser capaz de:

- Determinar el momento exacto en que deben llegar.
- Reconocer cuando empieza y cuando termina cada uno de los bits.
- Reconocer cuando empieza y termina la serie de bits que conforman el carácter.

Para facilitar el reconocimiento de todos los caracteres se estableció un sistema de sincronización por medio de algunos bits de delimitación y separación. Transmisión sincrónica o no sincrónica. (No sincrónica en nuestro caso).

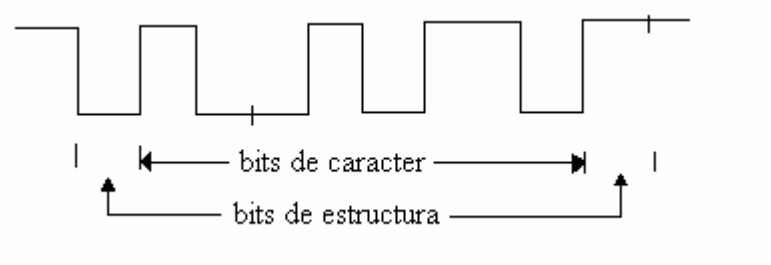
TRANSMISIÓN NO SINCRÓNICA:

Asocia unos bits especiales a cada carácter conformando una estructura ó paquete, incorpora un bit antes y uno después del carácter.

Todos los bits incluyendo los de la estructura se envían a la misma velocidad (todos los bits con el mismo periodo) y cuando llegan al receptor este ya sabe que el primer bit recibido corresponde al bit de estructura y que después de el en intervalos fijos de tiempo encontrará los

bits de datos.

Tanto el emisor como el receptor deben trabajar a la misma tasa de bits (velocidad), como el primer bit es quien informa la aparición de un paquete y después de el se localizan los bits de datos a una tasa constante, el carácter puede ser enviado en cualquier momento. Por ello, este se conoce como carácter no sincrónico y bit sincrónico.



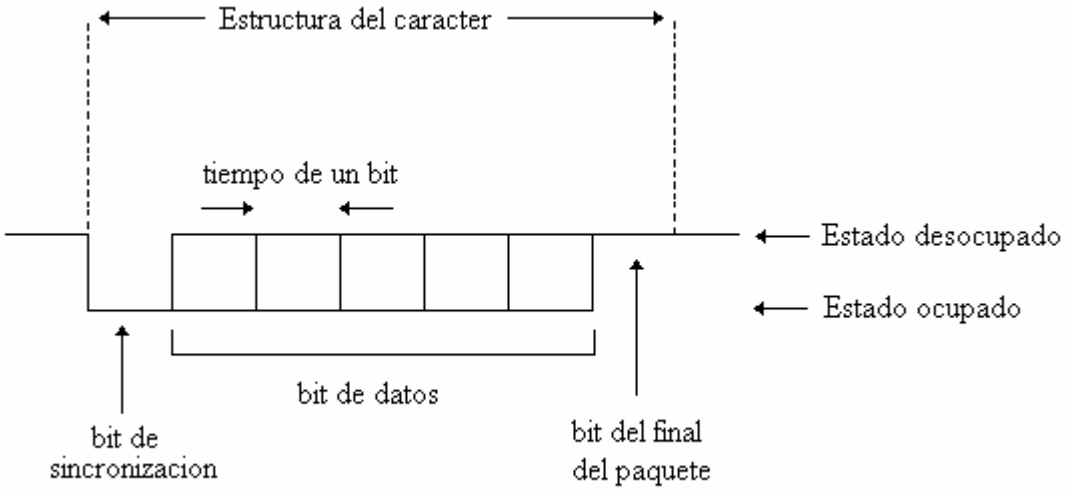
PROTOCOLO SERIAL NO SINCRÓNICO:

Es el protocolo utilizado en comunicaciones seriales no sincrónicos.

En la transmisión, los caracteres son transmitidos a intervalos indeterminados pero a una tasa fija de bits (carácter no sincrónico, bit sincrónico).

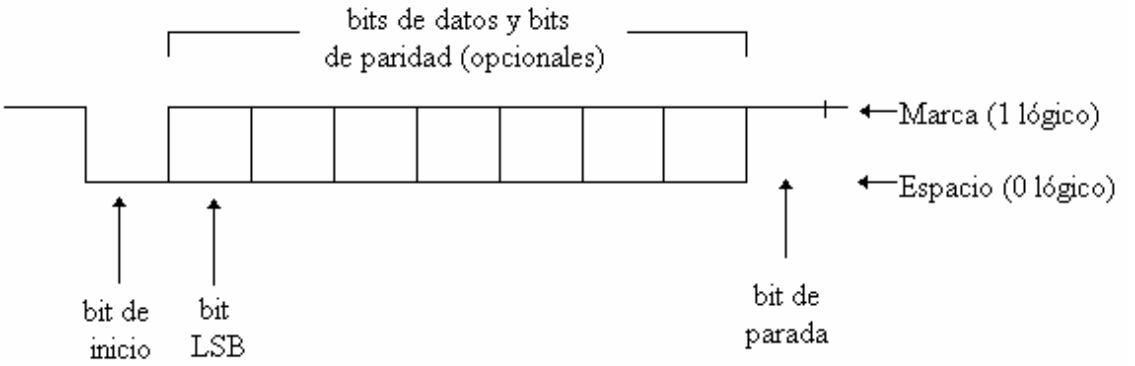
Según este protocolo, el estado vacío de la transmisión es un 1 lógico conocido como estado de marca. El estado de ocupado es un 0 lógico, estado de espacio. Al bit de sincronización bit de inicio y al bit de finalización de estructura bit de parada. El bit de inicio es una transición desde el estado de marca hacia el estado de espacio y el bit de parada es el estado de marca.

Solo puede existir un bit de inicio pero pueden existir varios de parada. El protocolo especifica que deben ser 1, ½ ó 2 bits de parada como mínimo y 5, 6, 7 u 8 bits de datos en el carácter. El carácter de datos se transmite empezando por el bit menos significativo y a continuación del ultimo bit de datos del carácter puede haber (aunque no es necesario) un bit especial denominado bit de paridad.



El bit de paridad se usa para detectar errores en la transmisión. Se determina la paridad por la cantidad de 1 lógicos en los datos, primero se decide si trabajar con paridad par o impar. Ejemplo, para paridad impar el numero de 1's en el carácter de datos debe ser impar, o sea, se cuenta el numero de 1's que hay en el carácter de datos, si la cuenta es impar el bit de paridad se pone en 0. Si la cuenta es par, el bit de paridad se pone en 1 para hacer la cantidad de 1's nuevamente impar.

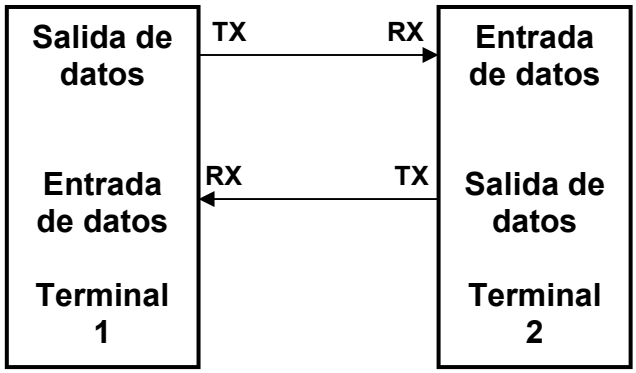
Formato del carácter:



El canal de comunicación:

En las comunicaciones en 2 vías se requiere un transmisor y un receptor al final de las 2 vías del canal de comunicaciones. Requieren de 2 líneas una para cada dirección de viaje de los datos.

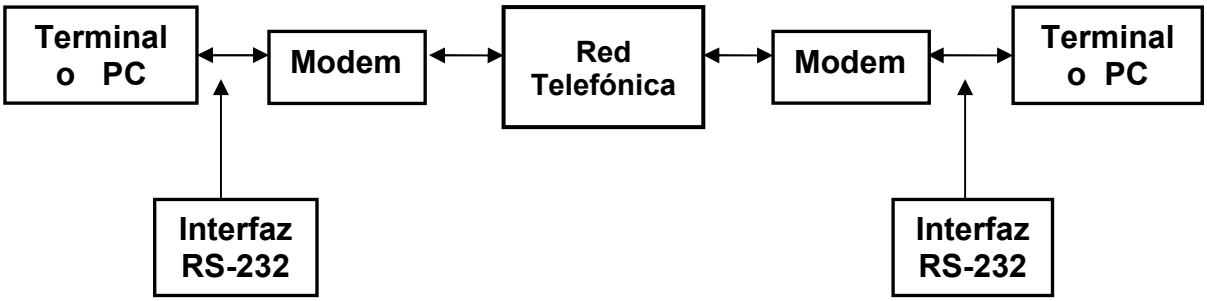
En la comunicación por medio de 2 canales, los datos pueden viajar en ambas direcciones simultáneamente (full duplex) ó en ambas direcciones no simultáneamente (half duplex).



El estándar RS-232:

Una aplicación real de la comunicación serial no sincrónica es la que se ha definido en el estándar internacional EIA RS-232, es una norma que define las características físicas que debe tener el canal y las funciones de las líneas de control y de datos. Dentro de esta norma los equipos terminales toman nombres específicos: Equipo terminal de datos (DTE) ó Equipo de comunicación de datos (DCE). Ejemplo: DTE (computadora) y DCE (módem).

Canal de comunicación serial:



El estándar define 25 líneas con sus correspondientes números de pines dentro de un conector. Según el estándar, el equipo DTE debe tener conector macho y el DCE conector tipo hembra.

Las líneas se clasifican en, las de funciones de control y las de funciones de datos que son solo 2 (TD y RD).

PIN	LINEA	NOMBRE	DIRECCION	FUNCION
DB9			DTE DCE	
3	TD	Datos TX	Salida Entrada	Datos de salida
2	RD	Datos RX	Entrada Salida	Datos de entrada
7	RTS	Petición para enviar	Salida Entrada	DTE quiere enviar
8	CTS	Listo para enviar	Entrada - Salida	OK para que DTE envíe
6	DSR	Datos listos	Entrada - Salida	DCE listo para comunicarse
5	Común	Común	----- -----	Tierra
1	DCD	Detección de portadora	Entrada - Salida	Enlace de Comunicación en Proceso
4	DTR	Equipo terminal listo	Salida - Entrada	DTE listo para comunicarse
9	RI	Indicador de llamadas	Entrada - Salida	Anuncia en llamado entrante

CARACTERÍSTICAS ELÉCTRICAS:

El estándar RS-232 se aplica a tasa de datos de hasta 20000 bits por segundo y hasta 20 metros de longitud de cable, aunque se puede extender mediante el uso de repetidores de señal que corrigen los niveles de voltaje.

En comunicaciones seriales externas con un microcontrolador las salidas digitales de 0 y 5 voltios deben acondicionarse para llegar a los niveles RS-232.

Para esto se utiliza una configuración de transistores ó un circuito integrado especial, al cual se le entregan los niveles TTL y arroja los niveles RS-232.

Comunicación no sincrónica en microcontroladores:

Como los microcontroladores mas pequeños no cuentan con módulos especializados de comunicación (UART), esas funciones se pueden desarrollar con algunas líneas de códigos. El objetivo será la comunicación con la PC mediante el puerto serie, que será el equipo DTE y quien deberá estar ejecutando un programa terminal que permita monitorear todos los datos que transmita ó reciba por este puerto.

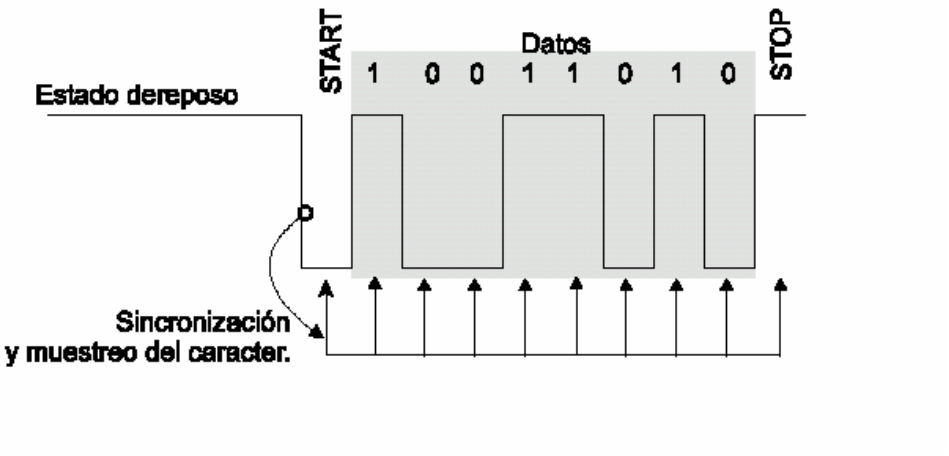
El sistema Windows ofrece el programa Hyperterminal para establecer la comunicación.

COMO SINCRONIZAR EN UNA TRANSMISIÓN ASINCRÓNICA:

Cuando el emisor no transmite, en el periodo entre caracteres, la línea se mantiene a "1" lógico. Cuando decide transmitir un carácter, primero transmite un "0" que se denomina bit de START y sirve para que el receptor sincronice (empieza a contar tiempos desde ese momento). El instante de sincronismo es el flanco de bajada de la señal (ver figura). Tras el bit de START se transmiten los bits de datos y después es obligatorio al menos un bit de STOP a "1" lógico. La secuencia se repite tantas veces como caracteres se transmitan. Obsérvese que este mecanismo de sincronización con el bit de START impide que la deriva de muestreo por diferencias entre los relojes continúe en el siguiente carácter. Se asume que la deriva de muestreo no debe ser tan grande que provoque un error de muestreo en los bits de cada carácter.

La transmisión asíncrona se lleva a cabo tal y como se describe en el punto anterior. En concreto además del bit de START utiliza:

- 5, 6, 7 o 8 bits de datos,
- 0 o 1 bit de paridad (la paridad puede ser "par"(Even), "impar"(Odd), "siempre a cero"(Reset) y "siempre a uno"(Set).
- 1, 1.5 o 2 bits de STOP.



Para agilizar el lenguaje se suele emplear una nomenclatura abreviada como, por ejemplo, "8N1" que indica que la transmisión serie RS232 se ha configurado para transmitir 8 bits de datos, No paridad y 1 bit de STOP. Otro ejemplo sería "6E2" que indica 6 bits de datos, paridad par y 2 bits de STOP.

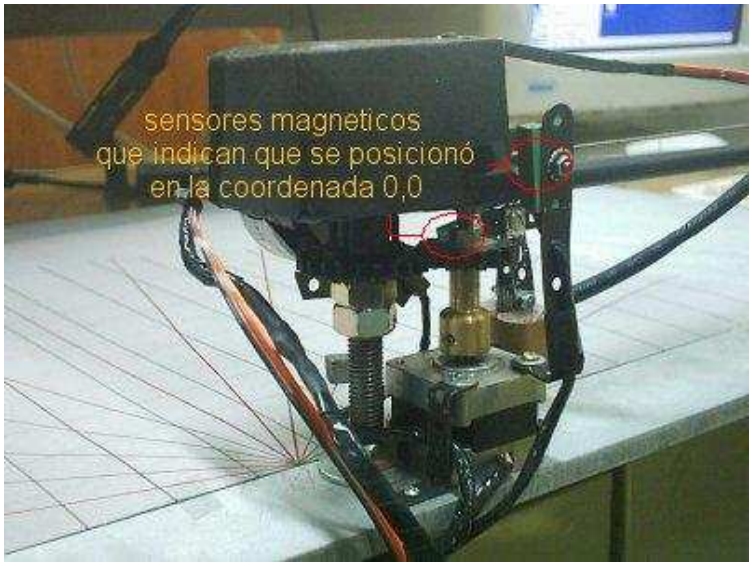
SINTESIS DEL FUNCIONAMIENTO

1. Se establece la comunicación entre PC y microcontrolador, de manera tal que se sincronicen ambas interfaces (PC, microcontrolador).

La PC espera un dato del microcontrolador.

Aparece un mensaje de error en caso de no establecerse la comunicación.

2. El microcontrolador posiciona la grúa en la coordenada 0 (alfa), 0 (ro).

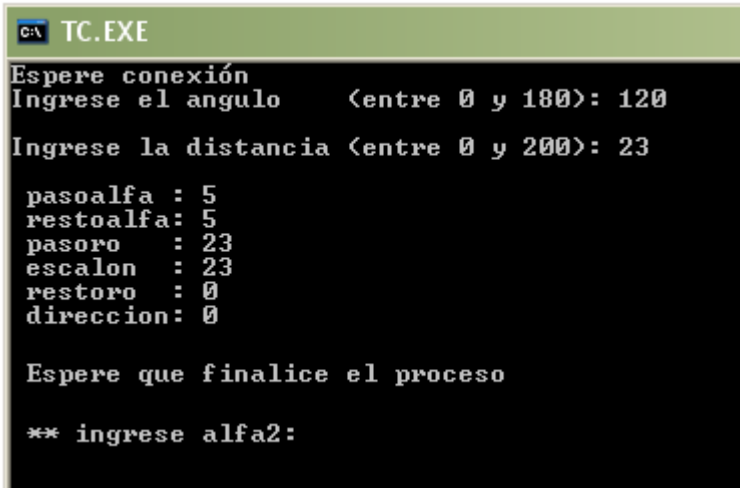


3. La PC pide que el usuario ingrese las coordenadas (ángulo y distancia), procesa los mismos y los transmite.

Los datos transmitidos luego de realizar el proceso de cálculo la PC transmite hacia el controlador los siguientes datos:

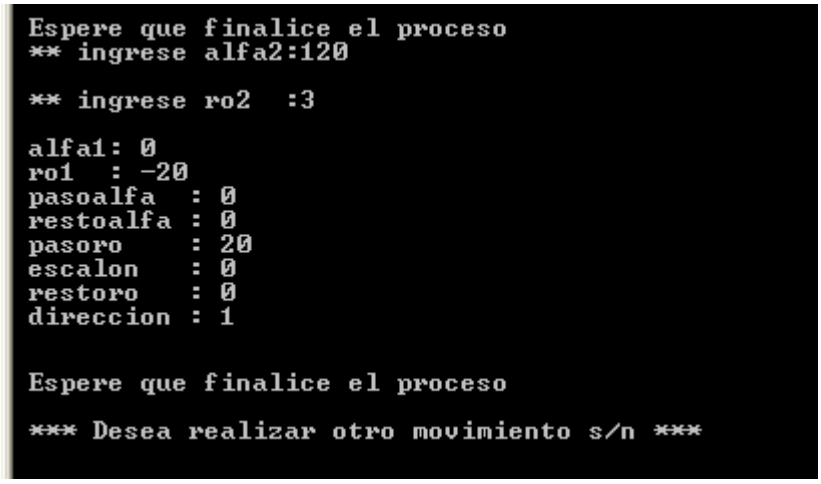
- Paso alfa
- Resto alfa
- Paso ro
- Escalón
- Resto ro
- Dirección

Que se pueden apreciar en pantalla, el PIC devuelve los valores que se transmitieron para verificar el correcto funcionamiento.



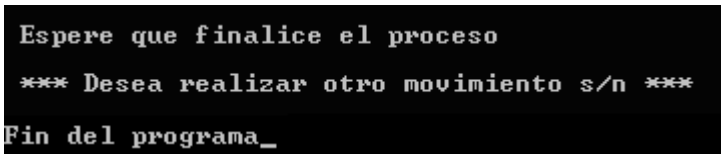
4. Una vez que el microcontrolador recibe los datos, efectúa el movimiento hacia la posición indicada por el usuario, levanta la pieza y envía un dato de fin de posición, lo que le permite al usuario poder ingresar las dos siguientes coordenadas; no es posible ingresar las coordenadas si el proceso se esta ejecutando debido que las entradas de teclado están bloqueadas.

5. La PC recibe el dato y pide al usuario las coordenadas del movimiento final. Nuevamente se procesan los datos y efectúa la transmisión.

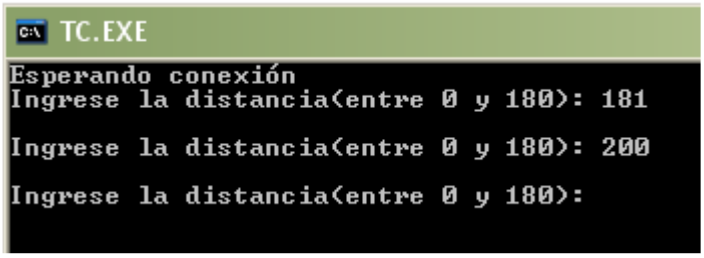


6. Al recibir los datos el microcontrolador efectúa el nuevo movimiento, luego deja la pieza, se vuelve a acomodar a las coordenadas 0,0 de tal manera que se reduzca los errores que se podrían cometer si el proceso seguiría desde ese punto, y envía un dato a la PC que finalizó de ejecutar las instrucciones.

7. La PC pregunta si el operario desea realizar un nuevo movimiento, en caso afirmativo retorna al punto 3, de lo contrario finaliza el programa.



RESEÑA: en el programa se tomaron algunas medidas de precaución con respecto al ingreso erróneo de parte del usuario, como bloquear las teclas cuando se esta ejecutando el proceso y limitar el numero ingresado por el usuario, de tal manera que sea el permitido en el rango preestablecido en la maquina; ejemplo: el rango preestablecido de la distancia es de 0 a 180, si el número introducido esta fuera del mismo, el programa reitera que se vuelva a introducir la coordenada; lo mismo sucede con el ángulo.



ESPECIFICACIONES TÉCNICAS

Mediciones angulares

Para alfa deseado = 120° alfa medido 124° error relativo = 3.33%

El error cometido en el desplazamiento es de 1º

DESARROLLO DE LA GRÚA

El sistema contiene dos programas, un programa desarrollado en C que corre en PC, encargado de la interfaz hombre-máquina y los cálculos, y el otro en asembler que corre en el microcontrolador PIC 16F84A, encargado de efectuar los desplazamientos del brazo, ambos se encuentran comunicados en forma serial mediante el estándar RS-232. Cada programa será explicado en forma detallada más adelante.

Para comenzar con el armado, obtuvimos en primera instancia los dos motores paso a paso, donde pudimos mediante pruebas obtener los datos necesarios para comenzar a desarrollar la etapa de potencia que los controlaría.

Uno de los motores tiene un desplazamiento angular de 1.8º por paso lo que da 200 pasos para efectuar un giro completo del eje. Y tiene un consumo de 2 Amp. El otro tiene un desplazamiento angular de 15º por paso, lo que da 24 pasos para efectuar un giro completo del eje. Y tiene un consumo de 1.5 Amp.

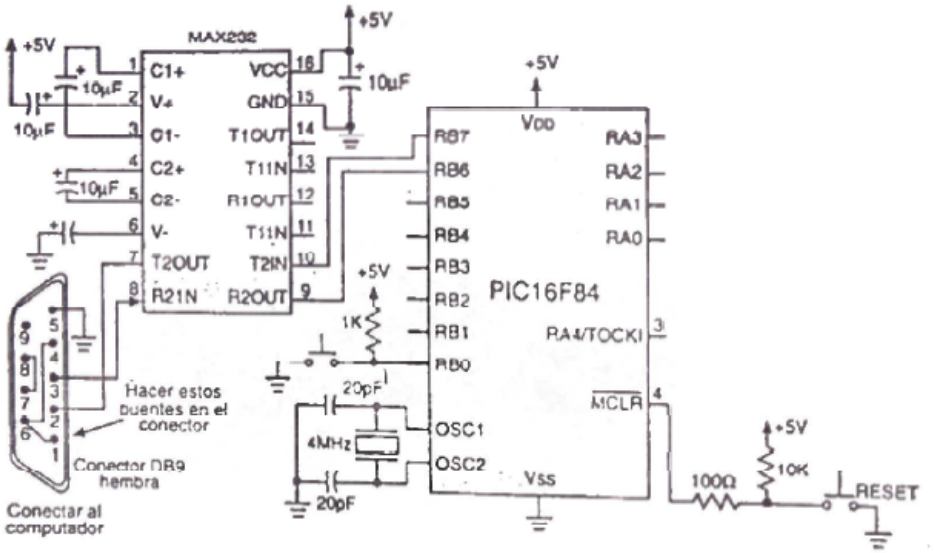
Una vez obtenido los datos anteriores comenzamos a pensar la forma constructiva del brazo. En este paso debíamos decidir que motor efectuaría cada desplazamiento. Y por la mayor cantidad de pasos decidimos que el motor de 200 pasos por giro, efectúe el movimiento angular y el otro el radial.

En forma conjunta se comenzó a desarrollar la plaqueta electrónica, la cual debía contener la etapa de potencia que maneja los motores PP, la etapa de comunicación y la de control.

Los Circuitos utilizados son los siguientes:

COMUNICACIÓN SERIAL.

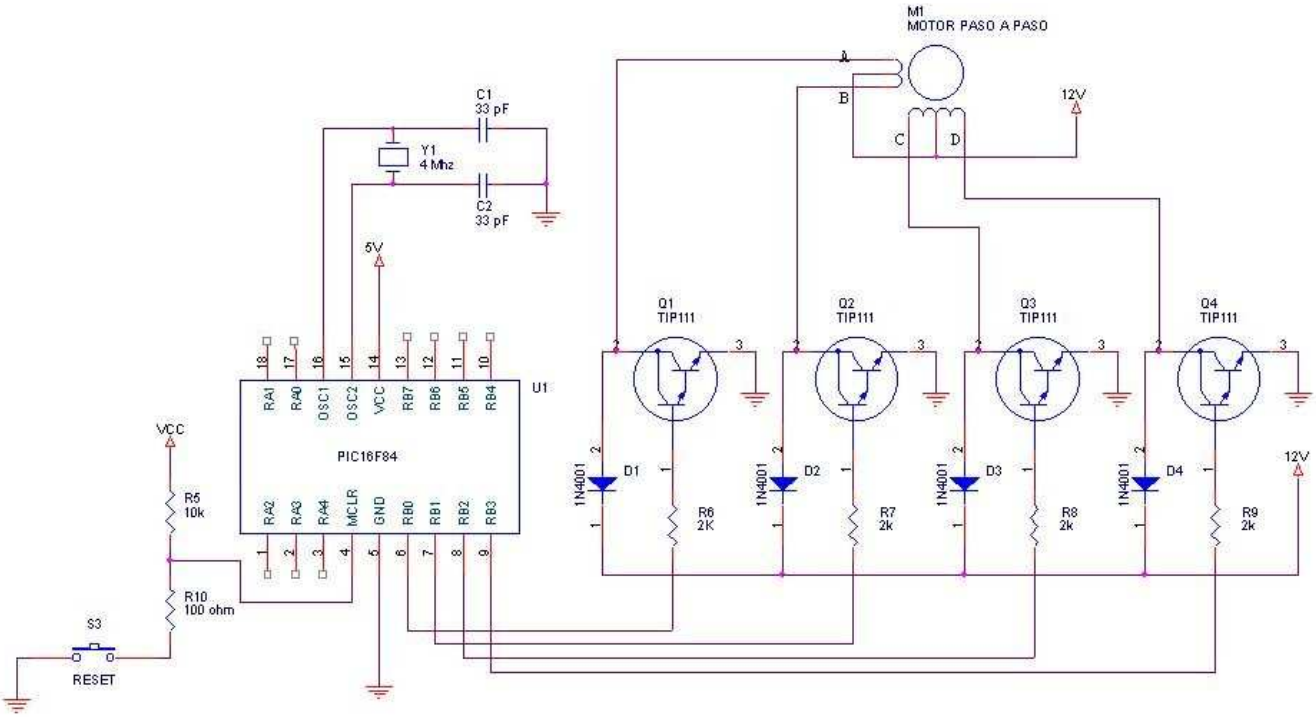
El circuito integrado MAX232 cambia los niveles TTL a los del estándar RS-232 cuando se hace una transmisión, y cambia los niveles RS-232 a TTL cuando se tiene una recepción. El circuito típico se muestra en la siguiente figura:



Circuito rs232

MOTORES PASO A PASO CONTROLADOS POR MICROCONTROLADOR PIC16F84:

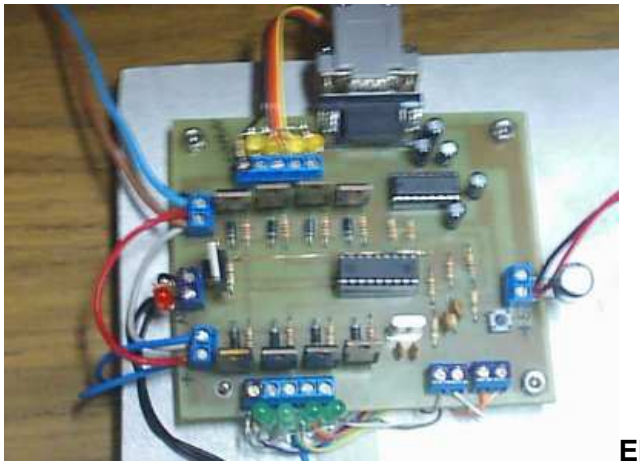
En los pines RB0 a RB3 se conecta por medio de transistores TIP 111 unos de los motores que simula el movimiento angular (alfa), y de igual manera se conecta otro que simula el desplazamiento del modulo del vector (ro) en los pines RB4 a RB7, quedando así como se ve en la figura:



EL brazo tiene una posición de inicio denominada posición cero. El cero esta determinado por dos switch's magneticos, uno ubicado en el 0º y el otro en 0 distancia (ro).

Se colocaron sensores magnéticos, debido a la robustez de los mismos ya que anteriormente se habían implementado otro tipo de switch y en las pruebas, al salir el brazo en una dirección incorrecta y hacer tope sobre los switch, los mismos se rompieron y fueron descartados por ser muy frágiles. Se podían haber implementado otro tipo de sensor, como una barrera óptica, finales de carrera tipo industrial, etc.

Una de las falencias de este sistema es que no tiene en cuenta si los motores siguen girando por alguna razón ajena al sistema de control y se pasan de los límites máximos. Una solución podría ser la colocación de sensores en dichos extremos o bien una traba mecánica para asegurar que la grúa no exceda estos limites, para garantizar un área de trabajo limitada y fuera de la misma un área segura donde podría haber por ejemplo operarios.



El circuito terminado

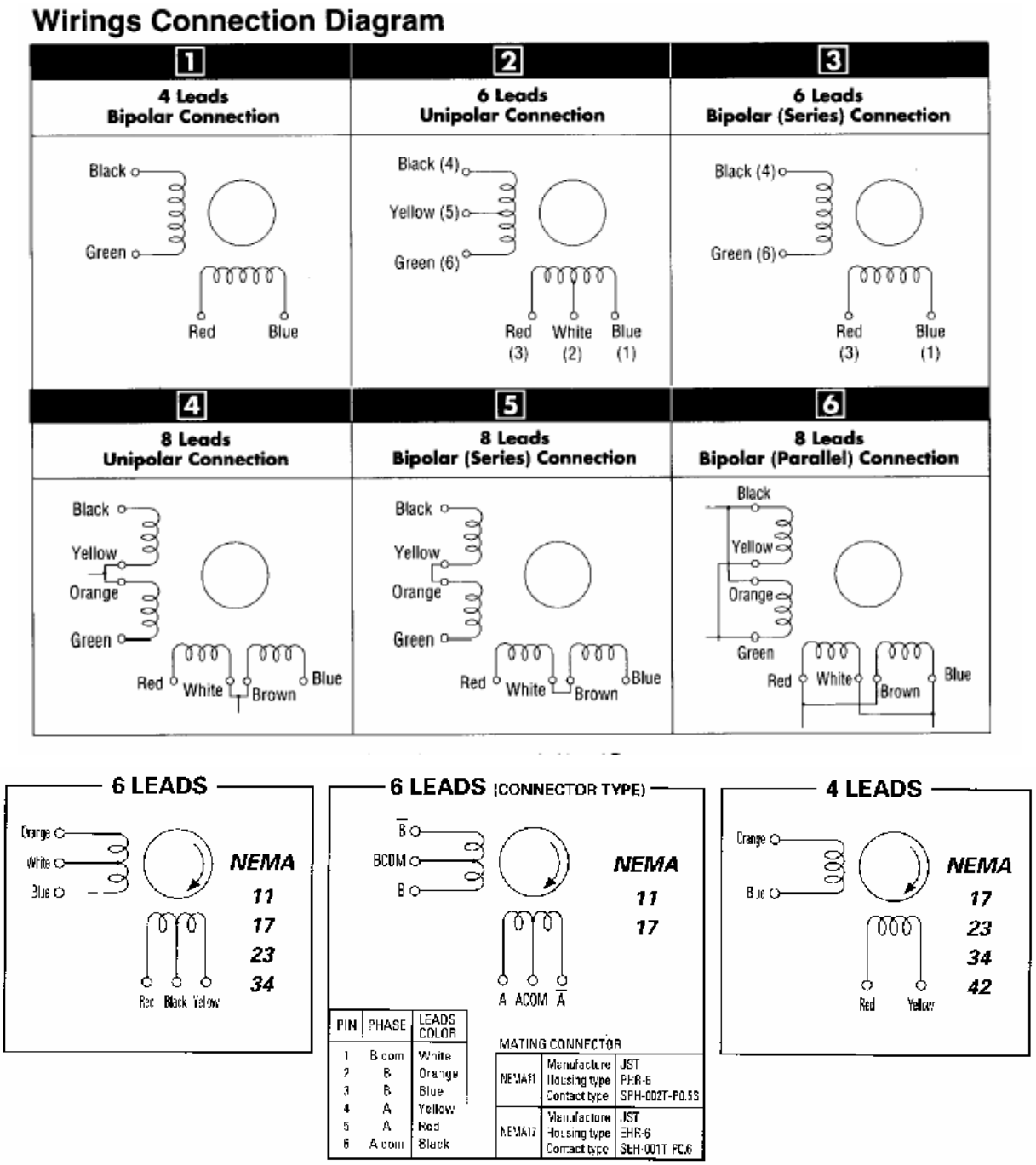
COMUNICACIÓN ENTRE LA PC Y EL MICROCONTROLADOR

La primera comunicación que hay entre ambos es para ver si está en condiciones el medio o si el microcontrolador esta operando con normalidad. La modalidad es la siguiente, la PC envia el número al microcontrolador, el mismo incrementa el dato recibido y se lo envía nuevamente a la PC. La próxima comunicación que se realiza es el envío de datos desde la PC hacia el PIC. Los datos que se envían son seis en el orden que se mencionan anteriormente. Cada vez que el microcontrolador recibe un dato lo reenvía nuevamente a la PC.

Lo que le faltaría al programa en C es chequear si los datos llegaron correctamente al microcontrolador y de no ser así, la grúa debería quedarse en el lugar e indicarle al operador por medio de un mensaje en pantalla que hubo un problema en la transmisión de datos.

La siguiente comunicación que se realiza, es la que efectúa el microcontrolador hacia la PC indicándole que termino de posicionarse.

Datos extraídos de Internet sobre motores Paso a Paso



PROGRAMA EN EL MICROCONTROLADOR

Reseña del programa en ASM

El programa esta diseñado para recibir los datos enviados por la PC y en base a ello efectuar el desplazamiento.

Los datos son los siguientes y los recibe en el mismo orden:

- Paso alfa: indica la cantidad de pasos que debe dar el motor de desplazamiento angular máximo 8 bits
- Resto alfa: registro de 8 bits
- Paso ro: Indica la cantidad de pasos que de be dar el motor de distancia máximo 8 bits
- Escalón: indica la cantidad de combinaciones de ambos movimientos, de alfa y ro, son necesarios para llegar a la posición deseada máximo 8 bits
- Resto ro: registro de 8 bits
- Dirección: es un registro de 8 bits de los cuales utilizamos los cuatro primeros bits:

El primer BIT indica la dirección del motor de desplazamiento angular, 1 dirección hacia delante, 0 dirección hacia atrás.

El segundo BIT indica la dirección del motor de desplazamiento del motor de distancia, 1 dirección hacia delante, 0 dirección hacia atrás.

El tercer BIT indica si se debe encender el electroimán o no, 1 encender electroimán, 0 no encender electroimán.

El cuarto BIT indica si el brazo debe posicionarse en la posición cero o no, 1

Listado del programa en ASM

```
;==============definición de registros============
;==============puerto A============================
FINANG          EQU     00H;    LIMITE DE CARRERA DE MOTOR DE ANGULO
FINDIST         EQU     01H;    LIMITE DE CARRERA DE MOTOR DE DISTANCIA
RX              EQU     03H
TX              EQU     02H
SOLENOIDE       EQU     04H
;==============puerto B============================
PASO1MA                 EQU     00H
PASO2MA                 EQU     01H
PASO3MA                 EQU     02H
PASO4MA                 EQU     03H
PASO1MD                 EQU     04H
PASO2MD                 EQU     05H
PASO3MD                 EQU     06H
PASO4MD                 EQU     07H
;========definicion de variables==================
LOOP            EQU     10H
LOOP2           EQU     11H
```

```
R0E             EQU     12H
RECEP           EQU     13H
TRANS           EQU     14H
CONTA           EQU 15H
R0D             EQU     16H
BANDERA         EQU     18H
;--------bits de la bandera ------------------------------
DIRGIRO                 EQU 00H;        Dirección de giro
DIRDIST                 EQU 01H;        Dirección de distancia
ENCELECTRO      EQU     02H;            Apagado o encendido de electroinman
POSI            EQU     03H
;-----------------------------------------------------------------
ESCALONES       EQU     19H
AUX             EQU     20H
CONTVUELTA      EQU     21H
ROTAANG         EQU     22H;    REGISTRO PARA SABER POSISION DEL MOTOR DE ANGULO
ANGULO          EQU     23H;    REGISTRO PARA LUEGO AGREGARSELO AL PUERTOB
ROTADIST        EQU     24H
DISTANCIA       EQU     25H
PGIRO           EQU     26H;    Pasos de Giro
PDISTANCIA      EQU     27H;    Pasos de distancia
RGIRO           EQU     28H;    Resto de giro
RDISTANCIA      EQU     29H;    Resto distancia
PGIROAUX        EQU     2AH;    Registro auxiliar de pasos de giro
PDISTAUX        EQU     2BH;    Registro auxiliar de pasos de distancia
AUXPORTB        EQU     2CH;    Registro auxiliar del puerto B para apagar los motores y poder encenderlos en la
posision que estaban
;=========registros de la eeprom==================
RCARGA          EQU     30H;    registro de carga
REPUERTA        EQU     31H;    registro de espera de apertura de puerta
RAPUERTA        EQU     32H;    registro de puerta abierta
RCPUERTA        EQU     33H;    registro entre que cierra la puerta y abre la carga
UNI             EQU     34H
DEC             EQU     35H
AUXL            EQU     36H
AUXILIAR        EQU     37H
;================================================================================================
        LIST P=PIC16F84a
        INCLUDE "p16F84a.INC"
                ORG     00H
                GOTO    INICIO
;================================================================================================
;================================================================================================
;gira hacia adelante de a un paso el motor de angulo.
CAMBIOADANG
```

```
                MOVLW B'00000000'
                MOVWF ROTAANG
                GOTO  SIGOADANG
PASOADANG
                MOVF  ROTAANG,0
                XORLW B'00000101'
                BTFSC STATUS,Z
                GOTO  CAMBIOADANG
SIGOADANG
                INCF  ROTAANG,1
                MOVF  ROTAANG,W
                CALL  TABLAGIRO
                BCF   STATUS,Z
                MOVWF ANGULO
                XORLW B'00001000'
                BTFSS STATUS,Z
                GOTO  SALIR
                MOVLW B'00000000'
                MOVWF ROTAANG
SALIR
                MOVLW D'50'
                CALL  RETARDO
                RETURN
TABLAGIRO
                ADDWF PCL,1
                NOP
                RETLW B'00000001'
                RETLW B'00000010'
                RETLW B'00000100'
                RETLW B'00001000'
;-------------------------------------------------------------------------------------------------------------------------------------------
;gira hacia atras de a un paso el motor de angulo
CAMBIOATANG
                MOVLW B'00000101'
                MOVWF ROTAANG
                GOTO  SIGOATANG
PASOATANG
                MOVF  ROTAANG,0
                XORLW B'00000000'
                BTFSC STATUS,Z
                GOTO  CAMBIOATANG
SIGOATANG
                DECF  ROTAANG,1
                MOVF  ROTAANG,W
```

```
                CALL    TABLAGIRO
                BCF     STATUS,Z
                MOVWF ANGULO
                XORLW B'00000001'
                BTFSS   STATUS,Z
                GOTO    SALIRT
                MOVLW B'00000101'
                MOVWF ROTAANG
SALIRT
                MOVLW D'50'
                CALL    RETARDO
                RETURN
;================================================================================================
;gira hacia adelante de a un paso el motor de angulo.
CAMBIOADDIST
                MOVLW B'00000000'
                MOVWF ROTADIST
                GOTO    SIGOADDIST
PASOADDIST
                MOVF    ROTADIST,0
                XORLW B'00000101'
                BTFSC   STATUS,Z
                GOTO    CAMBIOADDIST
SIGOADDIST
                INCF    ROTADIST,1
                MOVF    ROTADIST,W
                CALL    TABLADIST
                BCF     STATUS,Z
                MOVWF DISTANCIA
                XORLW B'10000000'
                BTFSS   STATUS,Z
                GOTO    SALIRD
                MOVLW B'00000000'
                MOVWF ROTADIST
SALIRD
                MOVLW D'50'
                CALL    RETARDO
                RETURN
TABLADIST
                ADDWF PCL,1
                NOP
                RETLW  B'00010000'
                RETLW  B'00100000'
                RETLW  B'01000000'
```

```
                RETLW  B'10000000'
;================================================================================================
CAMBIOATDIST
                MOVLW B'000000101'
                MOVWF ROTADIST
                GOTO   SIGOADDIST
PASOATDIST
                MOVF   ROTADIST,0
                XORLW B'00000000'
                BTFSC  STATUS,Z
                GOTO   CAMBIOADDIST
SIGOATDIST
                DECF   ROTADIST,1
                MOVF   ROTADIST,W
                CALL    TABLADIST
                BCF     STATUS,Z
                MOVWF DISTANCIA
                XORLW B'00010000'
                BTFSS  STATUS,Z
                GOTO   SALIRDIST
                MOVLW B'00000101'
                MOVWF ROTADIST
SALIRDIST
                MOVLW D'50'
                CALL    RETARDO
                RETURN

;================================================================================================
;El retardo generado por esta rutina es de w ms,antes de llamar
;debo cargar W con el valor en ms del retardoque quiero hacer
RETARDO
                MOVWF LOOP
TOPP2
                MOVLW D'122'
                MOVWF LOOP2
TOPP
                NOP
                NOP
                NOP
                NOP
                NOP
                DECFSZ          LOOP2,1
                GOTO   TOPP
                DECFSZ          LOOP,1
```

```
                GOTO   TOPP2
        RETURN
;=============================================================================================
UNOYMEDIO                       ; RUTINA PARA RETORNAR AL BIT Y MEDIO PARA UN CRISTAL DE 4MHz
                MOVLW .249      ; CARGA PARA 1250 us APROXIMADAMENTE
                GOTO   STARTUP  ; SALTA A EJECUTAR TIEMPO
DELAY1
                MOVLW .166      ; CARGA PARA 833 us APROXIMADAMENTE
STARTUP
                MOVWF R0E       ; LLEVA EL VALER DE DE CARGA AL RETARDO
REDO
                NOP
                NOP
                DECFSZ    R0E   ; DECREMENTAR RETARDO, SALTA SI ES CERO
                GOTO   REDO     ; REPETIR HASTA TERMINAR
                RETLW  0        ; RETORNAR
;=============================================================================
RECIBIR
                NOP
                CLRF   RECEP    ; LIMPIA REGISTRO DE RECEPCION
                BTFSC  PORTA,RX ; LINEA DE RECEPCION ESTA EN BAJO?
                GOTO   RECIBIR  ; SI NO LO ESTA, VOLVER A LEER
                CALL   UNOYMEDIO ; LLAMAR A RUTINA UNOYMEDIO BITS
RCVR
                MOVLW D'8'      ; CARGAR EL CONTADOR CON EL
                MOVWF CONTA     ; NUMERO DE BITS
RNEXT
                BCF    STATUS,C ; LIMPIAR EL CARRY
                BTFSC  PORTA,RX ; PREGUNTAR POR EL ESTADO DE LA LINEA
                BSF    STATUS,C ; ACTIVAR EL CARRY SI ESTA EN ALTO
                RRF    RECEP    ; ROTAR REGISTRO DE RECEPCION
                CALL   DELAY1   ; LLAMAR RUTINA DE UN BIT
                DECFSZ    CONTA      ; DECREMENTAR CONTADOR, SALTAR SI ES 0
                GOTO   RNEXT         ; REPETIR HASTA COMPLETAR DATO
                RETLW  0


;=============================================================================
ENVIAR
                MOVWF TRANS
XMRT
                MOVLW D'8'
                MOVWF R0D
                BCF    PORTA,TX
                CALL   DELAY1
```

```
XNEXT
                BCF     PORTA,TX
                BCF     STATUS,C
                RRF     TRANS
                BTFSC   STATUS,C
                BSF     PORTA,TX
                CALL    DELAY1
                DECFSZ          R0D,1
                GOTO    XNEXT
                BSF     PORTA,TX
                CALL    DELAY1
        RETLW 0
;========retardo de un segundo==================
SEGUNDO
                MOVLW 4h
                MOVWF AUX
ACA             MOVLW   d'250'
                CALL    RETARDO
                DECFSZ          AUX
                GOTO    ACA
        RETURN
;-------------------------------------------------------------------------------
;================Movimiento de los motores=======================================
MOVIMIENTO
                CLRF    W
                MOVF    ANGULO,0                ; Muevo el registro angulo al acumulador para luego
                IORWF   DISTANCIA,0             ; agregarle el registro distancia y sacarlo al puertoB
                MOVWF PORTB                     ; para desplazar los motores.


                BCF             STATUS,Z        ; Chequeo si escalones es igual a cero
                MOVF    ESCALONES,0             ; si es así retorno porque no hay que moverse
                XORLW D'00'
                BTFSC   STATUS,Z
                RETURN
;*******************************************************************************
MOVIMIENTO1
                BCF     STATUS,Z                ; Muevo el registro Pasos de Giro a Pasos
                MOVF    PGIRO,0                 ; de Giro auxiliar para poder contar los pasos
                MOVWF PGIROAUX                  ; Y chequeo si los pasos que hay que mover son cero
                XORLW D'00'                     ; si es así salto al desplazamiento de distancia sobre
                BTFSC   STATUS,Z                ; el eje si no efectuo desplazamiento angular con
                GOTO    ESCDIST                 ; la cantidad de pasos de cada escalon
;*******************************************************************************
```

```
ESCANG
                CALL    MOVIMIENTOANG       ; llamo a la funcion de desplazamiento de un paso angular
                BCF     STATUS,Z            ; Chequeo si el registro de resto de giro es cero o no
                MOVF    RGIRO,0             ; si no es cero es que debo desplazar un paso mas por escalon
                XORLW D'00'                 ; si el registro es cero no debo agregar ningun paso
                BTFSC   STATUS,Z            ;
                GOTO    NORESTOGIRO         ;
                CALL    MOVIMIENTOANG       ;
                DECF    RGIRO               ;
NORESTOGIRO
                DECFSZ          PGIROAUX,1  ; Decremento el registro Aux de pasos de giro se es cero termino
con el escalon
                GOTO    ESCANG              ; si no es cero continuo formando el escalon.
;-----------------------------------------------------------------
                BCF     STATUS,Z
                MOVF    PDISTANCIA,0
                MOVWF PDISTAUX
                XORLW D'00'
                BTFSC   STATUS,Z
                GOTO    NODIST
ESCDIST
                CALL    MOVIMIENTODIST
                BCF     STATUS,Z
                MOVF    RDISTANCIA,0
                XORLW D'00'
                BTFSC   STATUS,Z
                GOTO    NORESTODIST
                CALL    MOVIMIENTODIST
                DECF    RDISTANCIA,1
NORESTODIST
                DECFSZ          PDISTAUX,1
                GOTO    ESCDIST
NODIST
                DECFSZ          ESCALONES,1
                GOTO    MOVIMIENTO1
RETURN
;-------------------------------------------------------------------------------
MOVIMIENTOANG
                BTFSS   BANDERA,DIRGIRO
                CALL    PASOADANG
                BTFSC   BANDERA,DIRGIRO
                CALL    PASOATANG
                CLRF    W
                MOVF    ANGULO,0            ; Muevo el registro angulo al acumulador para luego
                IORWF   DISTANCIA,0         ; agregarle el registro distancia y sacarlo al puertoB
```

```
                MOVWF PORTB             ; para desplazar los motores.
RETURN
;-------------------------------------------------------------------------
MOVIMIENTODIST
                BTFSS  BANDERA,DIRDIST
                CALL   PASOADDIST
                BTFSC  BANDERA,DIRDIST
                CALL   PASOATDIST
                CLRF   W
                MOVF   ANGULO,0         ; Muevo el registro angulo al acumulador para luego
                IORWF  DISTANCIA,0      ; agregarle el registro distancia y sacarlo al puertoB
                MOVWF PORTB             ; para desplazar los motores.
RETURN
;-------------------------------------------------------------------------
;=========================================================================================
POSISION
POSISIONGIRO
                CLRF   W
                MOVF   ANGULO,0         ; Muevo el registro angulo al acumulador para luego
                IORWF  DISTANCIA,0      ; agregarle el registro distancia y sacarlo al puertoB
                MOVWF PORTB             ; para desplazar los motores.
                CALL   PASOADANG
                CLRF   W
                MOVF   ANGULO,0         ; Muevo el registro angulo al acumulador para luego
                IORWF  DISTANCIA,0      ; agregarle el registro distancia y sacarlo al puertoB
                MOVWF PORTB             ; para desplazar los motores.
                BTFSC  PORTA,FINANG
                GOTO   POSISIONGIRO
                MOVLW D'50'
                BTFSC  PORTA,FINANG
                GOTO   POSISIONGIRO
POSISIONDIST
                CALL   PASOATDIST
                CLRF   W
                MOVF   ANGULO,0         ; Muevo el registro angulo al acumulador para luego
                IORWF  DISTANCIA,0      ; agregarle el registro distancia y sacarlo al puertoB
                MOVWF PORTB             ; para desplazar los motores.
                BTFSC  PORTA,FINDIST
                GOTO   POSISIONDIST
                MOVLW D'50'
                BTFSC  PORTA,FINDIST
                GOTO   POSISIONDIST

RETURN
```

```
;*********************************************************************************
;*********************************************************************************
INICIO
;----------Configuración del puerto A-----------------------------------------------
                BSF     STATUS,RP0
                CLRF    PORTA
                MOVLW   B'00001011'                     ;(00000000) , puerto configurado como salida
                MOVWF   TRISA
                BCF     STATUS,RP0
;---------Configuración del puerto B-----------------------------------------------
                BSF     STATUS,RP0
                CLRF    PORTB
                MOVLW           B'00000000'
                MOVWF   TRISB
                BCF     STATUS,RP0

;-------------------------------------------------------------------------------
;*************************************************************************************
;-----------------------------------------------------------------
                MOVLW   B'00000010'
                MOVWF   ROTADIST
                MOVLW   B'00000010'
                MOVWF   ROTAANG
                CALL    RECIBIR
                MOVF    RECEP,0
                MOVWF   AUXILIAR
                INCF    AUXILIAR,1
                MOVF    AUXILIAR,0
                CALL    ENVIAR
                CALL    SEGUNDO
                CLRF    PORTB
HOME
                BCF     PORTA,SOLENOIDE
                CLRF    PORTB
                CLRF    ESCALONES
                CLRF    AUX
                CLRF    CONTVUELTA
                CLRF    ANGULO
                CLRF    DISTANCIA
                CLRF    PGIRO
                CLRF    PDISTANCIA
                CLRF    RGIRO
                CLRF    RDISTANCIA
                CLRF    PGIROAUX
```

```
                CLRF    PDISTAUX
                CALL    POSISION
                CLRF    PORTB
NOPOS
                MOVLW D'70'
                CALL    ENVIAR


;-------------------------------------------------------------------------------------
                CALL    RECIBIR         ; RECIBO CANTIDAD DE PASOS DE GIRO Y LO ALMACENO EN PGIRO
                MOVF    RECEP,0
                MOVWF PGIRO
;*****************************************
                MOVF    PGIRO,0
                INCF    0,1
                CALL    ENVIAR
;*****************************************
                CALL    RECIBIR         ; RECIBO CANTIDAD DE RESTOS Y LO GUARDO EN RGIRO
                MOVF    RECEP,0
                MOVWF RGIRO
;*****************************************
                MOVF    RGIRO,0
                INCF    0,1
                CALL    ENVIAR
;*****************************************
                CALL    RECIBIR         ; RECIBO CANTIDAD DE PASOS DE DISTANCIA Y LO GUARDO EN PDISTANCIA
                MOVF    RECEP,0
                MOVWF PDISTANCIA
;*****************************************
                MOVF    PDISTANCIA,0
                INCF    0,1
                CALL    ENVIAR
;*****************************************
                CALL    RECIBIR         ; RECIBO LA CANTIDAD DE ESCALONES Y LO GUARDO EN ESCALONES
                MOVF    RECEP,0
                MOVWF ESCALONES
;*****************************************
                MOVF    ESCALONES,0
                INCF    0,1
                CALL    ENVIAR
;*****************************************
                CALL    RECIBIR         ; RECIBO LA CANTIDAD E RESTOS DE DISTANCIA Y LO GUARDO EN RDISTANCIA
                MOVF    RECEP,0
                MOVWF RDISTANCIA
;*****************************************
```

```
                MOVF    RDISTANCIA,0
                INCF    0,1
                CALL    ENVIAR
;*****************************************
                CALL    RECIBIR         ; RECIBO LAS DIRECCIONES DE CADA MOTOR Y LAS GUARDO EN BANDERA
                MOVF    RECEP,0
                MOVWF   BANDERA
;*****************************************
                MOVF    BANDERA,0
                INCF    0,1
                CALL    ENVIAR
;*****************************************
;-------------------------------------------------------------------------------------------------------------
                MOVLW   D'10'
                CALL    RETARDO
                CALL    MOVIMIENTO
                BTFSC   BANDERA,ENCELECTRO
                BSF     PORTA,SOLENOIDE
                BTFSS   BANDERA,ENCELECTRO
                BCF     PORTA,SOLENOIDE
                CALL    SEGUNDO
                CLRF    PORTB
                MOVLW   D'128'
                CALL    ENVIAR
                BTFSC   BANDERA,POSI
                GOTO    NOPOS
                GOTO    HOME
END
```

PROGRAMA DE CONTROL EN LA PC

Reseña del programa en C++

Comienza transmitiendo un dato hacia el PIC luego lo recibe para confirmar que están en sincronismo.

El programa pide al usuario que introduzca las dos coordenadas (alfa -ángulo- y ro –distancia-). El programa comienza hacer los cálculos de los datos que serán transmitidos, se hace una comparación para ver cual es el mayor para que el menor sea el divisor, se divide entre ambos, obteniéndose los siguientes datos que serán transmitidos en este orden:

- Paso alfa: indica la cantidad de pasos que debe dar el motor de desplazamiento angular.
- Resto alfa: indica el resto de la división cuando alfa es mayor que ro; en caso contrario es 0.
- Paso ro: Indica la cantidad de pasos que debe dar el motor de distancia.
- Escalón: indica la cantidad de combinaciones de ambos movimientos (alfa y ro), necesaria para llegar a la posición deseada; esta dada por la menor coordenada.
- Resto ro: indica el resto de la división cuando ro es mayor que alfa; en caso contrario es 0.
- Dirección: nos indica hacia donde va a desplazarse los motores.

El programa espera un dato del PIC que indica que se terminaron de ejecutar las instrucciones del mismo.

Se vuelve a permitir la entrada de las siguientes coordenadas, volviendo a realizar los cálculos y la transmisión de datos de la misma forma. A diferencia del calculo anterior que se realiza con los dos números introducidos, los nuevos cálculos se realizan a partir de la diferencia de primeo introducido con el segundo para complementar el movimiento.

El programa vuelve a esperar un dato del PIC que indica que se terminaron de ejecutar las instrucciones y pregunta al usuario si desea volver a ejecutar nuevamente el proceso o finalizar.

Listado del programa en C++

ASPECTO GENERAL DEL USO DE LA FUNCIÓN BIOS.H

Uso del Bioscom para tener acceso RS232 por DOS ----------

Declaración:

```
int bioscom(int cmd, char abyte, int port);
unsigned _bios_serialcom(int cmd, int port, char abyte);
```

Interrupción utilizada: 0x14
Especificación del Puerto COM: 0 = COM1, 1 = COM2, 2 = COM3, 3 = COM4.

Descripción de comandos ----------

Cmd esta definido en BIOS.H

bioscom	_bios_serialcom	función
0	_COM_INIT	Setea los parámetros de comunicación
1	_COM_SEND	envía el dato
2	_COM_RECEIVE	recibe el dato
3	_COM_STATUS	devuelve el estado del Puerto

Definiendo la librería Bioscom ----------

bioscom	_bios_serialcom	significado	
0x02	_COM_CHR7	7 data bits	**Cantidad de datos a enviar o recibir**
0x03	_COM_CHR8	8 data bits	
0x00	_COM_STOP1	1 stop bit	**Cantidad de bits de stop para detener la trasmisión**
0x04	_COM_STOP2	2 stop bits	
0x00	_COM_NOPARITY	Sin paridad	**Selección de bit de paridad**
0x08	_COM_ODDPARITY	Paridad impar	
0x18	_COM_EVENPARITY	Paridad par	
0x00	_COM_110	110 baudios	**Selección de velocidad de trasmisión**
0x20	_COM_150	150 baudios	
0x40	_COM_300	300 baudios	
0x60	_COM_600	600 baudios	
0x80	_COM_1200	1200 baudios	
0xA0	_COM_2400	2400 baudios	
0xC0	_COM_4800	4800 baudios	
0xE0	_COM_9600	9600 baudios	

```
//-----------------------------------------------------------------------------------------------------------------------
// INICIO DEL PROGRAMA
//-----------------------------------------------------------------------------------------------------------------------

#include <stdio.h>
#include <conio.h>
#include <bios.h>
#include <dos.h>
#include <math.h>
#define DATA_READY 0x100
#define SETTINGS (0x80 | 0x03 | 0x00 | 0x00)

int datos();
int transmitir();
int recibir();
int outpt=0,inpt=0,status;
int
escalon,comp,comp1,restoalfa,restoro,pasoalfa,pasoro,direccion=0,alfa,ro,alfa2,alfa1,ro2,ro1,direc
cion1=0,direccion2=0;
char sigue;

main() {
clrscr();
printf("Espere conexi¢n");
outpt=128;
transmitir();                          // envía un dato para comenzar

recibir();
inpt=0;

do{

do { printf ("\nIngrese el angulo (entre 0 y 180): ");
    scanf(" %d",&alfa);                // ingreso el ángulo
} while(alfa>180);
do { printf ("\nIngrese la distancia (entre 0 y 170): ");
```

```
    scanf(" %d",&ro);                    // ingreso la distancia
} while (ro>170);

alfa=alfa/1.26;                          // divido el alfa por la relación de engranajes
ro=ro*1.4655; if (ro>255){ ro=255; }  // multiplico el ro por la relación de los dientes del motor
printf("el alfa real: %d\n", alfa);
printf("el ro ro es: %d\n", ro);

if ((alfa==0) && (ro==0))
{ pasoro=0; pasoalfa=0; restoro=0; restoalfa=0; escalon=0;   }          // datos a transmitir
else
{ datos(); }

direccion=direccion+13;

outpt= pasoalfa;   transmitir(); recibir();printf(" pasoalfa: %d\n", pasoalfa); // transmitiendo  datos
outpt= restoalfa;  transmitir(); recibir();printf(" restoalfa: %d\n",restoalfa);
outpt= pasoro;     transmitir(); recibir();printf(" pasoro: %d\n",pasoro);
outpt= escalon;    transmitir(); recibir();printf(" escalon: %d\n",escalon);
outpt= restoro;    transmitir(); recibir();printf(" restoro: %d\n",restoro);
outpt= direccion;  transmitir(); recibir();printf(" direccion: %d\n",direccion);
comp=0;
comp1=0;
printf("\n Espere que finalice el proceso");

inpt=0;                  //  espero hasta que el proceso me habilite para seguir
do{                      //  espero hasta recibir un dato
outpt=128;               //
transmitir();            //
recibir();   }           //
while (inpt==0);         //

do { printf ("\nIngrese el angulo (entre 0 y 180): ");
    scanf(" %d",&alfa2);                 // ingreso el angulo
} while(alfa2>180);
do { printf ("\nIngrese la distancia (entre 0 y 170): ");
    scanf(" %d",&ro2);                   // ingreso la distancia
} while (ro2>170);


alfa2=alfa2/1.26;                // divido el alfa por la relacion de engranajes
ro2=ro2*1.4655;                  // multiplico el ro por la relacion de los dientes del motor
if (ro2>255){ ro2=255; }
printf("el alfa real es: %d\n", alfa2);
printf("el ro real es: %d\n", ro2);

ro1=ro2-ro;                      // diferencia para conocer para donde se mueve el motor
alfa1=alfa2-alfa;
printf("  alfa1: %d\n ",alfa1);
printf("  ro1: %d \n",ro1);
direccion=1;

if(ro1<0)   { direccion= direccion+2; }
if(alfa1<0) { direccion= direccion-1; }

ro  = abs  (ro1);
alfa= abs(alfa1);
```

```
datos();

outpt= pasoalfa;    transmitir(); recibir();printf(" pasoalfa: %d\n",pasoalfa);   // trasmicion de  datos
outpt= restoalfa;   transmitir(); recibir(); printf(" restoalfa: %d\n",restoalfa);
outpt= pasoro;      transmitir(); recibir(); printf(" pasoro: %d\n",pasoro);
outpt= escalon;     transmitir(); recibir(); printf(" escalon: %d\n",escalon);
outpt= restoro;     transmitir(); recibir(); printf(" restoro: %d\n",restoro);
outpt= direccion;   transmitir(); recibir(); printf(" direccion: %d\n",direccion);

printf("\n\n Espere que finalice el proceso");

inpt=0;                    //  espera un dato para seguir
do{                        //
outpt=128;                 //
transmitir();              //
recibir();   }             //
while (inpt==0);           //

recibir();          // aca se mete la parte de recepción

printf("\n \n *** Desea realizar otro movimiento s/n ***\n");
scanf(" %c",&sigue);

//inpt=0;                  // espera un dato para seguir
//do{                      //
//outpt=128;               //
//transmitir();            //
//recibir();   }           //
//while (inpt==0);         //

comp=0;
comp1=0;
direccion=0;

} while(sigue=='s');
}

//////////////////////////////////////////////////////////////////

int datos()                          // calculos de datos a trasmitir con los posibles casos
{
if(alfa >= ro)
{ if(ro==0)
            { pasoalfa=alfa;
              restoalfa=0;
              comp=alfa;
              comp1=ro;
              escalon=1;}
  else    { pasoalfa=alfa/ro;
              restoalfa=alfa-(pasoalfa*ro);
              comp=(pasoalfa*ro)+restoalfa;
              comp1=ro;
              escalon=ro;}

  pasoro=1;////////////////////
  restoro=0; }
```

```
else
{ if(alfa==0)
              { pasoro=ro;
                restoro=0;
                comp1=ro;
                comp=alfa;
                escalon=1;}
  else      { pasoro=ro/alfa;
                restoro=ro-(pasoro*alfa);
                comp1=(pasoro*alfa)+restoro;
                comp=alfa;
                escalon=alfa;}
  pasoalfa=1;////////////////////////
  restoalfa=0; }
}

//////////////////////////////////////////////////////////////////////

int transmitir()
{ bioscom(0, SETTINGS,0);       // inicializa la UART
   // 0= setea el puerto
   // SETTINGS= direcci¢n base del puerto
   // O CONFIGURA EL COM2
   bioscom(1, outpt,0);
   // 1= enviar dato
   // outpt= variable
   // 0 CONFIGURA EL COM2
   delay(100);
   return (0);
}

//////////////////////////////////////////////////////////////////////

int recibir()
{  bioscom(0,SETTINGS,0);            // inicializa la UART
   status=bioscom(3,0,0);            // Verifica el estado del compuerta
   if(status & DATA_READY)           // verifica si hay un dato en la compuerta
    {      inpt=bioscom(2,0,0);      // asigno a la variable el dato de la compuerta
           printf("el numero es %d",inpt);
    }
   return(0);
}
```

INCONVENIENTES Y SUGERENCIAS DE MEJORA

- Problemas para conseguir engranajes que tengan una relación mayor de manera de lograr una mejor precisión.
- La maqueta debe trabajar en posición horizontal debido a que una vez ubicados los motores los desenergizamos debido al incremento de la temperatura de los mismos.
- El electroimán esta diseñado para 5 V y esta trabajando con 12 V, con unas resistencias en serie para atenuar la corriente.
- Problema de ruido en la placa, el microcontrolador energizaba todos los pines, esto producía un incremento de corriente elevado, se soluciono con un filtro (capacitor de 0.01 µF en los pines de alimentación del PIC).
- Problemas con la comunicación serial, los datos no llegaban correctamente.
- Problemas de fuente de alimentación, características de la misma 12 V 2 A. El circuito es alimentado con 12 VCC y tiene un consumo de 3 A. Solución: fuente de mayor corriente.
- El puerto del PIC16F84 se ponía en cortocircuito debido al elevado consumo, que generaba el MAX232. Solución: colocación de resistencia de 330 Ω en serie con estos pines, para asegurar una corriente máxima de 15 mA.
- Se hicieron dos maquetas, una de ellas tenía mucho movimiento lateral (Figura1). En la Figura 2 se puede ver la maqueta definitiva.

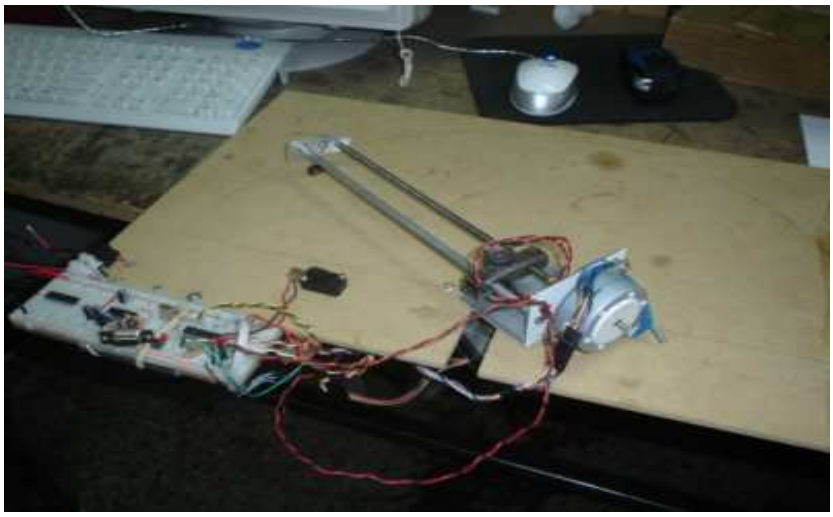


Figura1

Figura 2