
SISTEMA SOPORTE DE COMUNICACIONES PARA UN ROBOT

JOSÉ ANTONIO ABADÍA N.
Profesor titular, Departamento de Electricidad
Universidad del Valle

RESUMEN

Una parte de los sistemas robotizados depende del computador como un componente operacional más de control que de cómputo. El software en estos casos es de tiempo real, en el sentido que su operación debe ser sincronizada con tiempo y con eventos que ocurren en el robot. Para proveer comunicación en estas condiciones entre el computador y el robot, el intercambio de información en forma controlada permite que los procesos puedan continuar, aunque ellos sean asincrónicos, aleatorios o concurrentes.

Otra parte de los sistemas robotizados utiliza el computador para tareas más tradicionales como lo son la traducción de lenguaje en instrucciones utilizables por el robot y la planificación de trayectorias.

Este artículo presenta uno de los resultados obtenidos en el segundo objetivo de la segunda etapa del proyecto de investigación en Robótica Industrial financiado por la Universidad del Valle y Codinter Ltda. [1]. El sistema diseñado permite crear programas en un microcomputador, almacenarlos, enviarlos al robot, y ejecutarlos. Igualmente se pueden transmitir programas desde el robot al microcomputador.

El sistema utiliza el algoritmo de comunicación diseñado para controlar el flujo de información desde el computador hacia el robot y viceversa, una interfaz serial RS232, un robot de laboratorio y un microcomputador.

El artículo presenta una descripción del problema, el diseño general del sistema, el algoritmo de comunicación diseñado, su implementación y la evaluación realizada.

1. INTRODUCCION

UN ROBOT es un manipulador programable, multifuncional, diseñado para mover o manipular material, partes, herramientas o aparatos especiales a través de movimientos programados para la ejecución de varias labores.

Un robot industrial es de propósito general. Es una máquina programable que posee ciertas características antropomórficas. La característica antropomórfica, o humana, más típica, es su brazo. Este brazo, junto con la capacidad del robot de ser programado, lo hace ideal para una variedad de trabajos de producción, incluyendo la carga de maquinaria, soldadura de punto, pintura y ensamblaje. El robot puede ser programado para realizar una secuencia de movimientos mecánicos, y puede repetir ese movimiento infinidad de veces hasta que sea programado para ejecutar otra labor.

Los robots industriales varían en tamaño, configuración y capacidades, aunque comparten una estructura común. Todos los robots tienen cuatro componentes básicos: un manipulador, sensores, un controlador y una fuente de potencia. El manipulador es el ensamble de ejes capaces de moverse en varias direcciones. Los sensores monitorean posición,

velocidad, aceleración o torque. El tercer componente, el controlador, inicia, termina y coordina los movimientos y secuencias del robot. Proveer y regular la energía que es convertida en movimiento por los actuadores del robot es la función de la fuente de potencia. Esta puede ser neumática, eléctrica o hidráulica [7].

En el controlador, los componentes del computador pueden ser usados para comunicarse con el mundo exterior y/o entre componentes en el controlador del robot. Adicionalmente, un computador puede ser comunicado a una terminal que sea usada para programar el robot o monitorear sus actividades.

Recientemente han aparecido robots en el comercio que utilizan microcomputadores como su controlador maestro. Estos sistemas facilitan la implementación de robots en la industria y en la educación porque el sistema operativo y el hardware son bien conocidos. Un ejemplo es el robot RTX que se puede comunicar directamente a un IBM PC/XT o compatible a través de una RS232-C. Para este robot existe una biblioteca de interfaces para el lenguaje PASCAL.

En el Laboratorio de Robótica de la Universidad del Valle hay un robot HERO1, ET-18 [2], móvil, con un brazo de 6 grados de libertad que puede ser programado para ejecutar

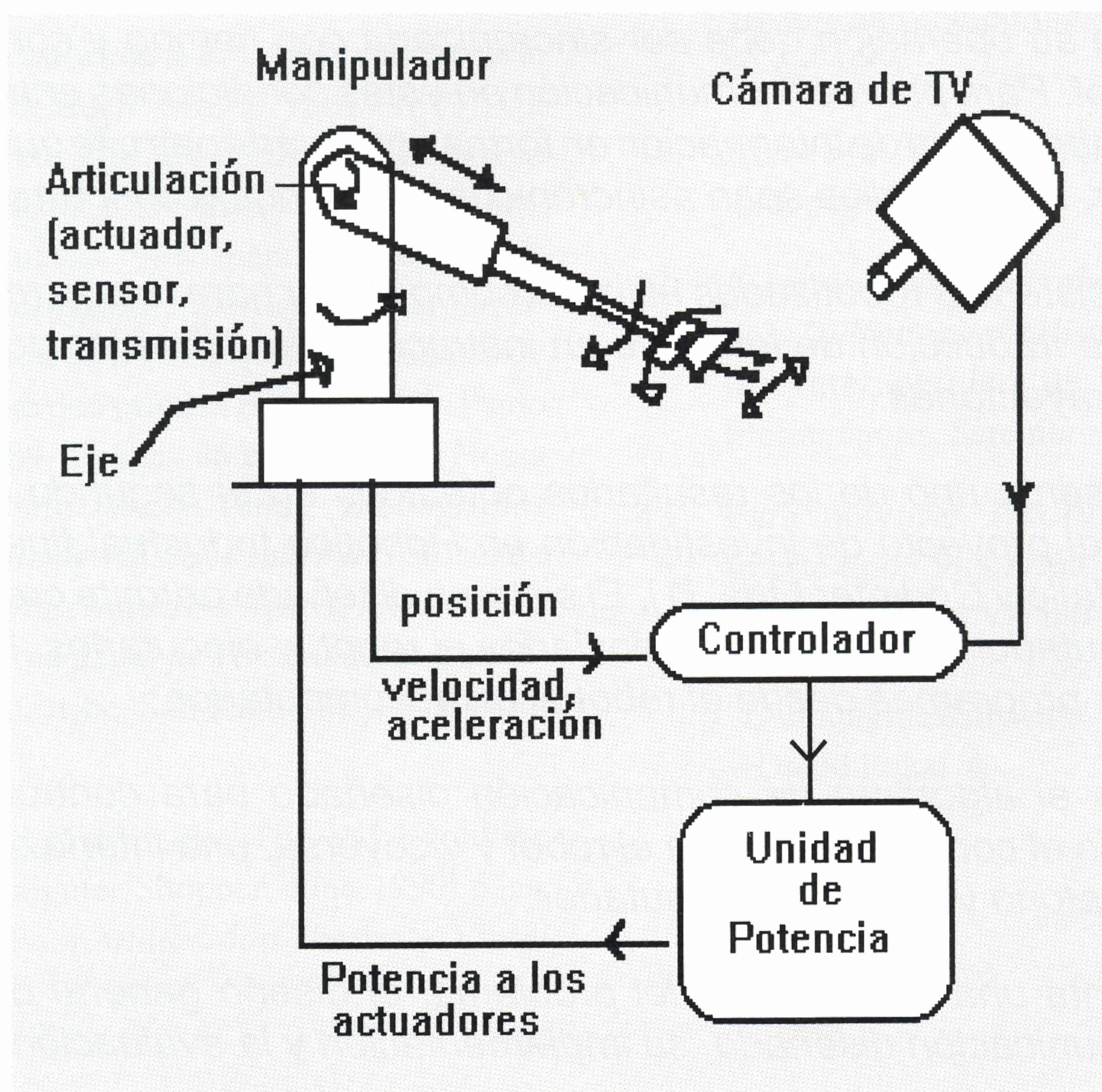


Figura 1
Componentes
de un robot

una gran variedad de tareas en el lenguaje del procesador Motorola 6808. Los programas elaborados se pueden almacenar en una cinta normal de casete utilizando una grabadora en un ambiente donde no haya mucha interferencia.

Debido a estas limitaciones y por la necesidad de desarrollar software para varios niveles en robótica, se diseñó el sistema soporte de comunicaciones para el robot HERO1, utilizando un microcomputador compatible IBMPC/XT, y una interfaz RS232.

2. DESCRIPCION DEL PROBLEMA

El robot ET-18 HERO 1 consta de una base móvil, un tronco, una cabeza, un brazo y un sintetizador de voz. El computador interno tiene un microprocesador de 8 bits Motorola 6808, una memoria ROM de 8Kb y una RAM básica de 4Kb. Cuenta con sensores para detectar luz, sonido, movimiento y un transmisor/receptor ultrasónico.

Para el movimiento de las articulaciones, HERO 1 utiliza motores paso-paso.

El HERO 1 puede ser programado por un teclado hexadecimal, por el sistema de enseñanza-repetición por medio de un control remoto o por una palanca de control [3]. Los programas se pueden almacenar para uso futuro en una cinta de casete convencional con los problemas conocidos de pérdida de información y de lentitud de operación.

Con el fin de brindar mayor versatilidad, facilidad y rapidez en la programación del robot se hace necesario desarrollar software que permita realizar esta labor desde un microcomputador compatible IBM-PC. El software requerido puede ser de varias clases:

editores, para escribir programas en el formato Motorola del robot o en lenguajes de alto

nivel como Pascal, Basic, C, etc.; **cross-ensambladores**, para convertir los programas en lenguajes de alto nivel al código Motorola del robot; **comunicadores**, para transferir y recibir información directamente entre computador y robot.

El sistema desarrollado hace parte de la última clase. En este caso se utilizó el editor GABOEDIT desarrollado en el proyecto de investigación GABO-Formateador de Texto [4], para editar los archivos de prueba del sistema.

Para hacer posible este desarrollo se hizo necesario expandir la memoria RAM del robot a 12 Kb y conectar la tarjeta de interfaz serial RS232.

La tarjeta de interfaz serial, Modelo ETW-18-10 provee un medio para realizar comunicaciones seriales bidireccionales según la norma RS-232 con un computador o un terminal remoto. Esta permite comunicaciones en cualquiera de las siguientes velocidades en baudios (bits por segundo): 300, 600, 1200, 2400, 4800 o 9600.

La gran ventaja que representa la utilización de esta tarjeta es que una vez instalada y usando el software adecuado se podrán crear los programas en microcomputador, guardarlos en disco flexible y luego cargarlos en el robot a través de la interface; evitándose así tener que usar el engorroso, lento y poco confiable sistema de almacenamiento mediante grabadora y casetes convencionales que se venía utilizando hasta el momento. Otra aplicación importante de la tarjeta es cuando se utiliza al robot como un instrumento de medición de alguna variable física en tiempo real y se necesita almacenar los resultados de tal forma que puedan ser procesados posteriormente.

Se instaló también una memoria RAM 2764 de 8 K en el socket U103 de la tarjeta de expansión. La importancia de

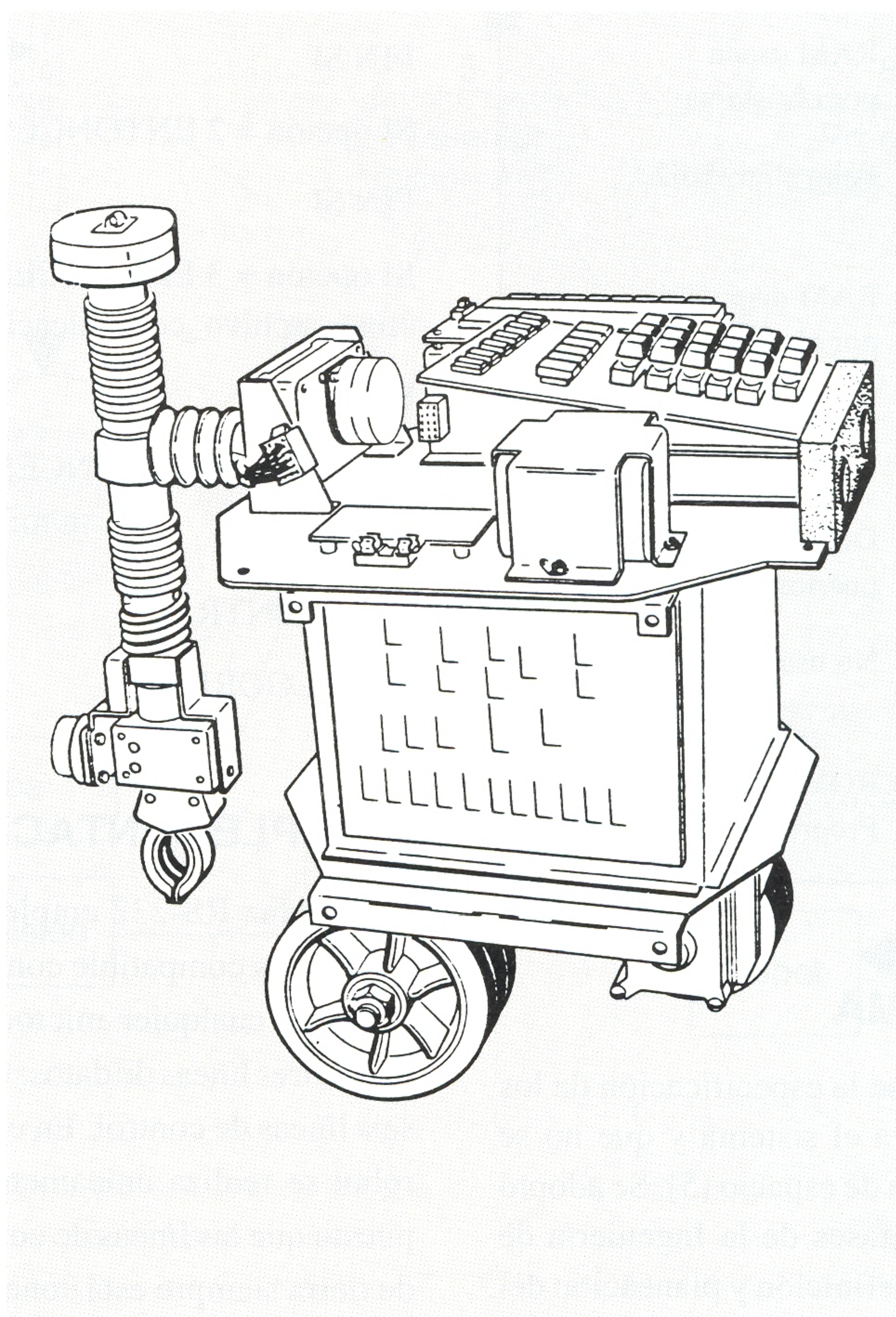


Figura 2. El Robot HERO 1

esta tarjeta radica en que con más memoria disponible es posible transmitir o recibir un volumen de información mucho mayor y de una manera más rápida a través de la interfaz RS-232. Además, al disponer de una capacidad de memoria mayor es posible cargar el robot con programas mucho más complejos que permiten que se puedan explotar todas las posibilidades del robot.

La organización de las direcciones de memoria o mapa de memoria que se obtiene con la instalación de la tarjeta de expansión se puede observar en la siguiente tabla:

Dirección Inic./Final	Cantidad memoria	Descripción
0000 003C	61 Bytes	RAM usada por el sistema
003D 0EE7	3,7 Kbytes	RAM disponible por el usuario
0EE8 0FFF	280 Kbytes	RAM usada por el sistema
1000 1FFF	4 Kbytes	Falta chip U102
2000 3FFF	8 Kbytes	RAM disponible por el usuario
4000 BFFF	32 Kbytes	Faltan chips U104, U105, U106 Y U107
C000 CFFF	4 Kbytes	Decodificación de puertos del sistema
D000 DFFF	4 Kbytes	No usado
E000 FFFF	8 Kbytes	ROM del sistema. Programa MONITOR

3. DISEÑO DEL SISTEMA

Para el diseño se tomó como base la especificación de los requerimientos establecidos para el sistema y que no se incluyen en este artículo por falta de espacio [5]. Se adoptó la estrategia del desarrollo por fases de la Ingeniería de Software [6] que comprende la definición y planeación del problema, su especificación, el diseño y la implementación.

En la **figura 3** se muestra el Diagrama de Flujo de Datos del sistema.

El proceso **configurar**, por ejemplo, permite establecer los parámetros de comunicación y el formato de transferencia de archivo.

4. ALGORITMO DE COMUNICACION

A continuación se presenta el pseudocódigo del primer nivel del algoritmo principal del sistema de comunicaciones robot-computador.

ALGORITMO “Comunicación Hero1-PC”

Seleccionar velocidad de transmisión e iniciar puerto;

MIENTRAS desee_seguir HACER

Seleccionar opción;

SI opción = 1 ENTONCES transmitir_datos por_teclado;
por_archivo;

FIN SI

SI opción = 2 ENTONCES recibir_datos;

FIN SI

SI opción = 3 ENTONCES
crear_archivo_comunicación;

FIN SI

SI opción = 4 ENTONCES salir_programa;

FIN SI

FIN MIENTRAS

FIN ALGORITMO

5. IMPLEMENTACION DEL SISTEMA

La interfaz RS-232 emplea un conector D hembra de 25 terminales compatible con el conector D macho del puerto serial de cualquier microcomputador. La norma RS-232 define tres líneas de datos: transmisión, recepción y tierra; y seis líneas de control. En este caso, la comunicación con el robot se realiza únicamente con las tres líneas de datos, puesto que las líneas de control no son necesarias. La línea de tierra siempre está conectada al pin 7, pero las líneas de transmisión y recepción están conectadas según el tipo de

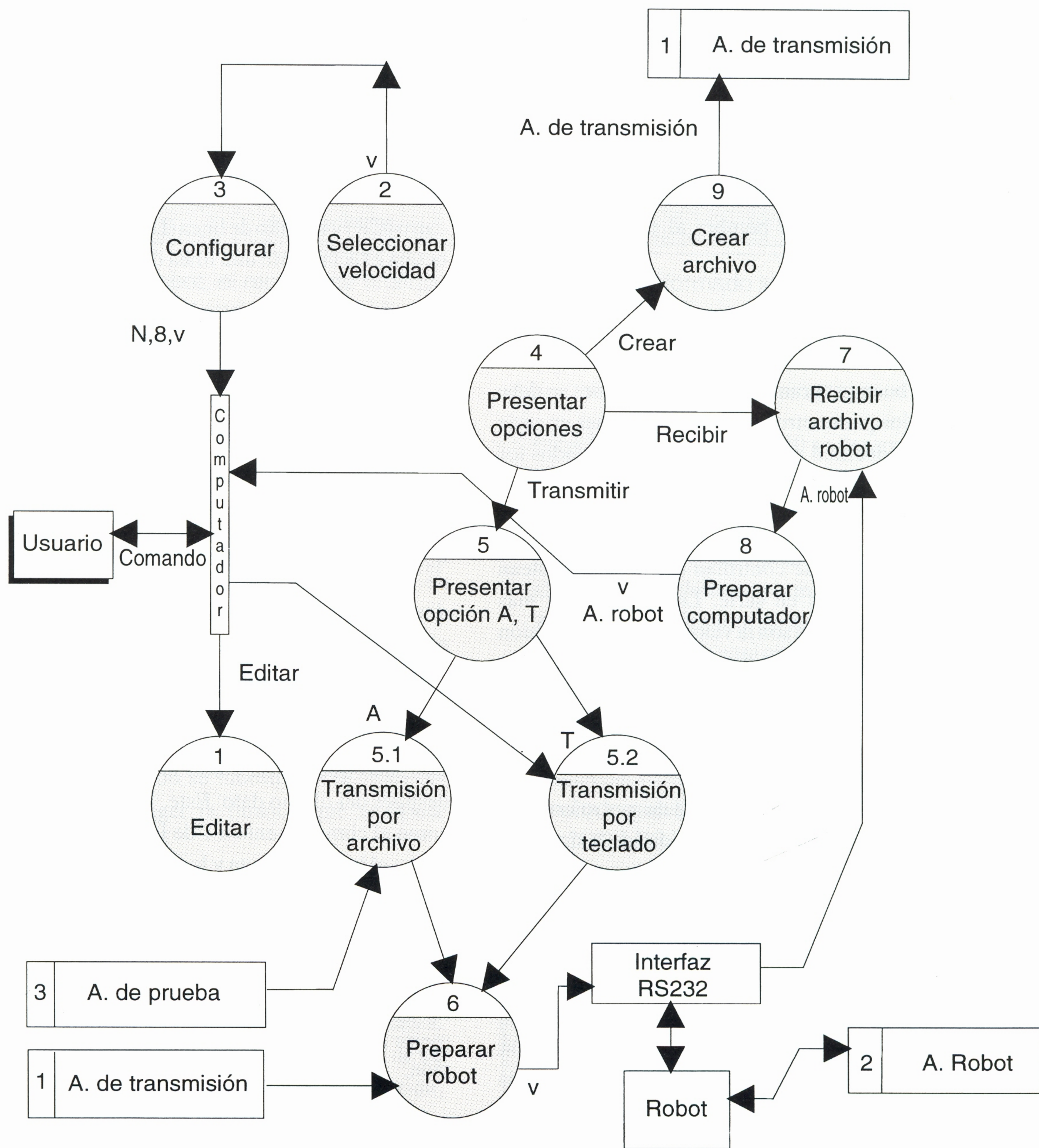


Figura 3. Diagrama de Flujo de Datos.

equipo. Existen dos tipos de equipos: los DTE (Data Terminal Equipment) y los DCE (Data Communication Equipment); en los DTE la línea de transmisión está conectada al pin 2 y la línea de recepción al pin 3, mientras que en los DCE se conectan al contrario.

En general todos los microcomputadores trabajan como DTEs, mientras el robot está configurado para trabajar como DCE; por lo tanto, el cable de conexión debe constar de un conector macho DB25 del cual salen tres líneas de los pines 2, 3 y 7 que se conectan a los respectivos pines 2, 3 y 7 de un conector hembra DB25. El conector macho se conecta a la interfaz serial y el conector hembra al puerto serial del microcomputador.

5.1 Desarrollo del Programa

Para diseñar un buen programa de comunicaciones se debe tener en cuenta los parámetros de comunicaciones en serie y el formato hexadecimal en que deben ser manejados los archivos. Generalmente estos dos aspectos los impone el DCE que en este caso es el robot.

Hay varios parámetros que controlan cómo se formatean datos en comunicaciones en serie. Los parámetros más comunes que se encontrarán son la velocidad de transmisión en baudios, el número de bits de datos por palabra, el tipo de paridad y el número de bits de parada. Según los manuales de la Heathkit, el robot debe trabajar con los siguientes parámetros.

- La tarjeta interface posee un sistema de tres microinterruptores que permite ajustar la velocidad de la siguiente manera:

Velocidad (Baudios)	Posición de los microinterruptores		
	1	2	3
300	OFF	OFF	OFF
600	ON	ON	OFF
1200	OFF	OFF	ON
2400	ON	OFF	ON
4800	OFF	ON	ON
9600	ON	ON	ON

- 8 bits de datos por palabra.
- Paridad nula o no se necesita bit de paridad.
- 1 bit de parada.

En cuanto al formato hexadecimal para la transferencia de archivos, se tiene que el robot opera con el formato “MIKBUC” de Motorola. Este formato, también conocido como formato de Registros “S”, representa información binaria como un texto en código ASCII, organizado como una secuencia de registros de datos, seguido de un registro de finalización. Cada registro finaliza con un retorno de carro (CR) y un salto de línea (LF) y por lo tanto cada registro ocupa exactamente una línea del archivo.

En cada registro, los datos binarios son representados como dígitos hexadecimales en código ASCII; dos dígitos por cada byte binario. Cada dígito hexadecimal es un carácter separado. Por ejemplo, el valor binario de 255 se convierte en los caracteres “FF” en el archivo.

Cada registro tiene un campo de direcciones (4 dígitos, la parte alta primero). En un registro de datos, este campo proporciona la dirección donde se debe cargar el primer byte de datos. En un registro de finalización, proporciona la dirección de inicio del programa; esto es, el punto de partida para la ejecución del programa. Antes del campo de direcciones, cada registro posee un byte de cuenta (2 dígitos), que indica el número de datos que posee dicho registro, incluyendo los dos bytes del campo de direcciones y un byte adicional que se usa como *checksum* (chequeo de suma) después del último dato. Este, de 8 bits (1 byte) se obtiene como el complemento a 1 de la suma de todos los bytes de datos, el byte de cuenta y los bytes alto y bajo del campo de direcciones. Es importante anotar que los valores de los bytes binarios y no los caracteres ASCII son los que se utilizan en la obtención del *checksum*.

En la tabla de la página siguiente se puede comprender mejor la conformación de los registros de datos y finalización:

5.2 Codificación

Una vez determinados los parámetros de comunicación y el formato de transferencia de archivos, se procedió al diseño del programa. Se escogió el lenguaje BASIC para la codificación del programa, debido a que presenta un manejo sencillo y eficiente del puerto de comunicación serial, a

Registro de datos:					
S1	Cuenta 2 caracteres	Dirección inicial de carga 4 caracteres	Bytes de datos 2xcuenta-3 caracteres	Checksum CR 2 caracteres	LF
Registro de finalización:					
S9	Cuenta (03)	Dirección inicial 4 caracteres		Checksum 2 caracteres	CR LF
NOTA: En el robot, el registro de finalización puede estar conformado simplemente con los caracteres "S9".					

través de los comandos COM1 y COM2. Además dispone de otra serie de comandos que facilita el manejo de cadenas de caracteres ASCII.

En síntesis, el funcionamiento del programa elaborado es el siguiente: (Ver programa en el Anexo N°1)

- En las líneas 80, 90 y 100 se programa el puerto COM1 con la velocidad (SPEED\$) deseada, paridad nula (N) y 8 bits de datos por palabra (8).

- Entre las líneas 120 y 160 se muestra en pantalla un menú que permite escoger la función que se desea realizar.
- En la línea 180 se efectúa el salto correspondiente dependiendo de la opción seleccionada.
- Entre las líneas 210 y 280 se desarrolla un procedimiento que permite la transmisión directa por teclado desde el microcomputador (DTE) hacia el robot (DCE), de tal manera que una vez se entra en la bucla de transmisión, toda tecla que sea oprimida se transmite inmediatamente y la única forma de salirse es con "CONTROL E".
- Entre las líneas 300 y 1060, el programa permite la transmisión completa de un archivo creado y editado con anterioridad. Simplemente se da el nombre del archivo en formato Motorola que se desea transmitir, se alista el receptor (robot) y se oprime cualquier tecla.

- Entre las líneas 1500 y 1650 se desarrolla la opción de recibir datos desde el robot y cargarlos en un archivo cuyo nombre se solicita con anterioridad.
- Finalmente entre las líneas 2000 y 2530 aparece el algoritmo completo que permite la creación de un archivo en formato Motorola. Se desarrollan los procedimientos necesarios para que los datos que se van entrando por teclado queden organizados en registros según el formato Motorola. El programa se hizo de tal manera que todos los registros, excepto posiblemente el último, contengan 10 bytes (0A en Hex) de datos. Esto da como resultado que el campo CUENTA contenga los caracteres "OD" (13 en decimal) y el campo de direcciones vaya aumentando de 10 en 10 (0A en Hex) a partir de la dirección que da inicialmente.

El cálculo de la dirección del siguiente registro (líneas 2210 a 2330) y del CHECKSUM (líneas 2340 a 2440) fueron los procedimientos que demandaron un mayor número de líneas de programa. Debido a esto hay que analizar cada carácter según su código ASCII para poder determinar su valor hexadecimal.

5.3 **Procedimiento para enviar/recibir datos del robot**

Para enviar datos o un archivo del computador al robot, seleccione la velocidad de transmisión con los microinterruptores de la tarjeta interface (ya explicado) y

oprime las teclas [3] [A] en el teclado del robot. Luego inserte el disquete del programa en el microcomputador, entre al BASIC y cargue el programa ROBOT.BAS. Finalmente corra el programa siguiendo las instrucciones que se dan en el mismo.

Para recibir datos del robot se realiza básicamente el proceso contrario. Primero se corre el programa ROBOT.BAS y luego se oprimen las teclas [3][9] en el teclado del robot. Cuando aparece "____Fr" en el robot, entre la dirección de inicio del bloque de memoria que se desea transmitir del robot al microcomputador.

Luego, cuando aparece "____La", entre la dirección final.

NOTA: Cuando se transmiten datos desde el robot a un microcomputador, es recomendable usar 300 baudios como velocidad de transmisión para disminuir la posibilidad de pérdida de información.

6. EVALUACION

Se crearon varios archivos mediante el programa ROBOT.BAS usando los ejemplos de programas que se encuentran en los manuales. (Ver ejemplo en Anexo 2).

Posteriormente estos archivos fueron transmitidos a la máxima velocidad (9600 baudios), comprobándose que la tarjeta de interface serial y el programa de comunicaciones están operando correctamente.

El gran inconveniente de este algoritmo es que no se permite ningún tipo de edición, es decir, que una vez se ha oprimido una tecla no es posible borrarla. En otras palabras, no se admiten errores al entrar los caracteres. Este problema se puede corregir desarrollando una rutina que permita preguntar si el último carácter es correcto o no.

7. CONCLUSIONES

Se desarrolló un sistema de comunicación entre un PC-XT y el robot HERO 1 que permite enviar programas de aplicación para ejecutarlos en el robot y recuperarlos posteriormente para almacenarlos en disco del computador.

El diseño e implementación del sistema se hizo siguiendo las estrategias de desarrollo de la Ingeniería de Software que nos permiten obtener un sistema confiable y ajustado a los requerimientos de los elementos disponibles en el proyecto de investigación en Robótica Industrial.

El sistema se puede mejorar adicionándole un módulo de ayudas, un módulo de manejo de errores, e involucrando el editor GABOEDIT en el sistema como otro de sus módulos. Actualmente se están haciendo pruebas para convertir el código a Pascal y C. ♦

8. BIBLIOGRAFIA

1. ABADIA, J. A. *Especificación, diseño, y construcción de un robot tipo industrial - Etapa III*. Univalle. Cali. 1991.
2. HERO1, ET-18 ROBOT, Heathkit, Zenith Educacional Systems. U.S.A. 1982
3. HERO 1, User's Manual, Heathkit, U.S.A. 1982.
4. VALENCIA, M. E., ABADIA, J. A. *GABO-Formateador de Texto*. Univalle, Cali, 1990.
5. HENNINGER, K. *Specifying Software Requirements for Complex Systems*, IEEE Transactions on Software Engineering 6(1):2-23, January 1980.
6. FAIRLEY, R. *Ingeniería de Software*, McGraw-Hill, U.S.A. 1987.
7. GROOVER, M. P. y otros. *Robótica Industrial*, McGraw-Hill, U.S.A. 1989.

ANEXO 1

“Programa ROBOT.BAS - José Antonio Abadía - Univalle”

```

10  SCREEN 0,0:WIDTH 80
20  KEY OFF:CLS:CLOSE
30  DEFINT A-Z
40  FALSE=0:TRUE=NOT FALSE
50  MENU=5
60  XOFF$=CHR$(19):XON$=CHR$(17)
70  LOCATE 1,1:PRINT "PROGRAMA PARA TRANSMISION
    ASINCRONICA DE DATOS - FORMATO MOTOROLA";
80  LOCATE 3,1:LINE INPUT "¿Velocidad de transmision en
    baudios?"; SPEED$
90  COMFIL$="COM1:"+SPEED$+"N,8"
100 OPEN COMFIL$ AS #1
110 OPEN "SCRN:"FOR OUTPUT AS #3
115 CLS
120 LOCATE 1,1:PRINT "Entre la opción deseada ?";
130 LOCATE 3,5:PRINT "1. Transmitir datos"
140 LOCATE 4,5:PRINT "2. Recibir datos"
150 LOCATE 5,5:PRINT "3. Crear archivo comunicacion"
155 LOCATE 6,5:PRINT "4. Salir del programa"
160 LOCATE 8,1:INPUT "Entre su opcion (1,2,3 o 4)";OPCION
170 CLS:PAUSE=FALSE
180 ON OPCION GOTO 200,1500,2000,9997
200 LOCATE 1,1:LINE INPUT "Transmision por teclado(T) o por
    archivo(A)?";T$
210 IF T$="A" THEN 300
212 LOCATE 3,1:PRINT "Entre los datos oprimiendo las teclas co-
    rrespondientes";
214 LOCATE 5,1:PRINT "Finalice con ^E":LOCATE 7,1
215 A$=INKEY$:IF A$="" THEN 230
220 IF ASC(A$)=MENU THEN 115 ELSE PRINT #1,A$;
225 PRINT #3,A$;
230 IF EOF(1) THEN 215
240 IF LOC(1)>128 THEN PAUSE=TRUE:PRINT #1,XOFF$;
250 A$=INPUT$(LOC(1),#1)
260 IF LOC(1)>0 THEN 240
270 IF PAUSE THEN PAUSE=FALSE:PRINT #1,XON$;
280 GOTO 215
300 LOCATE 1,1:PRINT STRING$(80,32):LOCATE 1,1
310 LINE INPUT "¿Nombre del archivo en disco?";DSKFIL$
400 LOCATE 1,1:PRINT STRING$(80,32):LOCATE 1,1
410 PRINT "Aliste el RECEPTOR y oprima una tecla para iniciar la
    transmision";
420 B$=INKEY$:IF B$="" THEN 420
430 OPEN DSKFIL$ FOR INPUT AS #2
1000 WHILE NOT EOF(2)
1010 A$=INPUT$(1,#2)
1020 PRINT #1,A$;
1030 END
1040 PRINT #1,CHR$(28);
1050 CLOSE #2:CLS:LOCATE 23,10:PRINT "*** Fin de Transmision
    ***";
1060 GOTO 9990
1500 CLS:LOCATE 1,1

```

```

1510 LINE INPUT "¿Nombre del archivo a cargar datos?"; DSKFIL$
1520 OPEN DSKFIL$ FOR OUTPUT AS #2
1530 PRINT #1,CHR$(13);
1540 IF EOF(1) THEN GOSUB 1600
1550 IF LOC(1)>128 THEN PAUSE=TRUE:PRINT #1,XOFF$;
1560 A$=INPUT$(LOC(1),#1)
1570 PRINT #2,A$;:IF LOC(1)>0 THEN 1550
1580 IF PAUSE THEN PAUSE=FALSE:PRINT #1,XON$;
1590 GOTO 1540
1600 FOR I=1 TO 5000
1610 IF NOT EOF(1) THEN I=9999
1620 NEXT I
1630 IF I>9999 THEN RETURN
1640 CLOSE#2:CLS:LOCATE 23,10:PRINT "***Fin de Recepción***";
1650 GOTO 9990
2000 CLS:LOCATE 1,1
2010 LINE INPUT "Nombre del archivo a crear?";DSKFIL$
2020 OPEN DSKFIL$ FOR OUTPUT AS #2
2030 LOCATE 3,1:INPUT "Número de datos(Bytes)";NUMDATOS
2040 LOCATE 5,1:LINE INPUT "Direccion inicial en Hexadecimal
    (4 digitos)?";DIRHEX$
2045 LOCATE 7,1
2050 DATLIN=16:DT$="13"
2060 IF NUMDATOS>=16 THEN 2080
2070 DATLIN=NUMDATOS:DT$=HEX$(DATLIN+3)
2073 IF LEN(DT$)<2 THEN DT$="0"+DT$
2080 A$= "S1"+DT$+DIRHEX$
2090 SUMDATOS=0
2100 FOR I=1 TO DATLIN
2110 FOR J=2 TO 1 STEP -1
2120 B$=INKEY$:IF B$="" THEN 2120
2125 PRINT #3,B$;
2130 BASCII=ASC(B$)
2140 IF BASCII>57 THEN 2160
2150 BDEC=(BASCII-48)*J^4:GOTO 2170
2160 BDEC=(BASCII-55)*J^4
2170 SUMDATOS=SUMDATOS+BDEC
2180 A$=A$+B$
2190 NEXT J
2200 NEXT I
2210 POSICION=0:SUMDIR=0:NUEVADIR=0
2220 FOR K=1 TO 2
2230 FOR L=2 TO 1 STEP -1
2240 POSICION=POSICION+1
2250 D$=MID$(DIRHEX$,POSICION,1)
2260 DIRASCI=ASC(D$)
2270 IF DIRASCI>57 THEN 2290
2280 Z=(DIRASCI-48):DIRDEC=Z*L^4:ND=Z*16^
    (4-POSICION):GOTO 2300
2290 Z=(DIRASCI-55):DIRDEC=Z*L^4:ND=Z*16^(4-POSICION)
2300 SUMDIR=SUMDIR+DIRDEC
2310 NUEVADIR=NUEVADIR+ND
2320 NEXT L
2330 NEXT K
2340 CHECK$=HEX$(DATLIN+3+SUMDIR+SUMDATOS)
2350 CHSUM1$=RIGHT$(CHECK$,2)

```



```

2355 IF LEN(CHSUM1$)=2 THEN 2360
2357 CHSUM1$="0"+CHSUM1$:GOTO 2355
2360 CHSUM2$=""
2370 FOR I=1 TO 2
2380 X$=MID$(CHSUM1$,I,1)
2390 XASC=ASC(X$)
2400 IF XASC>57 THEN 2420
2410 COMDEC=63-XASC:GOTO 2430
2420 COMDEC=70-XASC
2430 CHSUM2$=CHSUM2$+HEX$(COMDEC)
2440 NEXT I
2450 A$=A$+CHSUM2$
2460 PRINT #2,A$
2470 DIRHEX$=HEX$(NUEVADIR+16)
2473 IF LEN(DIRHEX$)=4 THEN 2480
2477 DIRHEX$="0"+DIRHEX$:GOTO 2473
2480 NUMDATOS=NUMDATOS-16
2490 IF NUMDATOS>0 THEN 2060
2500 PRINT #2,"S9"

```

```

2510 CLOSE #2
2520 LOCATE 23,10:PRINT "***Archivo Terminado***";
2530 GOTO 9990
9990 LOCATE 25,5:PRINT "Oprima tecla para continuar";
9995 C$=INKEY$:IF C$="" THEN 9995
9996 GOTO 115
9997 CLOSE
9999 KEY ON:END

```

ANEXO 2

```

"Archivo de prueba SENSORES.RBT"
S11305F8B60EE1810026013F4583BDF64E9611BD36
S1130608F7ADCE10000926FD20F0010101B60EE178
S1130618810026013F4183BDF64EB6C240BDF7AD09
S1130628CE10000926FD20EFB60EE1810026013F19
S11306384283BDF64EB6C240BDF7ADCE30000926A2
S1130648FD20EF0101010101B60EE181FF260183BE
S11306580E7F0040867E9727CE066ADF283F4B83AD
S111066820FE96404C9740BDF64EBDF7AD39CE
S9

```