



# **matrix** | EBLOCKS 2

## **GSM Communications**



CP2832

**matrix**  
[www.matrixtsl.com](http://www.matrixtsl.com)

Copyright © 2018 Matrix Technology Solutions Limited

# Contenido

1. Introducción a la telefonía móvil .....	4
2. Resumen de los ejercicios .....	5
3. Configuración del hardware .....	6
3.1 Configuración del módem GSM .....	6
3.2 Configuración del componente RS232 .....	6
3.2.1 Velocidad BAUD .....	7
3.2 Configuración del sistema microcontrolador .....	7
4. Ejercicio 1: Un teléfono básico .....	8
4.1 Introducción .....	9
4.2 Objetivos .....	10
4.3 Requisitos .....	11
4.4 El programa Flowcode en detalle .....	11
4.4.1 Flowcode RS232 componente - SendChar .....	11
4.4.2 Macro - Tx_Command .....	11
4.4.3 Uso de comandos AT .....	12
4.5 Qué hacer .....	12
4.6 Nuevos trabajos .....	13
5. Ejercicio 2: Una "máquina de estados" sencilla .....	14
5.1 Introducción .....	14
5.1.1 Máquina de estados .....	14
5.2 Objetivos .....	15
5.3 Requisitos .....	15
5.4 El programa Flowcode en detalle .....	16
5.5 Qué hacer .....	16
5.6 Nuevos trabajos .....	16
6. Ejercicio 3: Respuestas del módem .....	17
6.1 Introducción .....	17
6.1.1 Visualización de mensajes .....	17
6.1.2 Componente LCD Flowcode .....	17
6.2 Objetivos .....	18
6.3 Requisitos .....	18
6.4 El programa Flowcode en detalle .....	18
6.4.1 Caracteres del mensaje .....	18
6.4.2 Caracteres con eco - macro Tx_Command modificada .....	18
6.5 Qué hacer .....	19
6.6 Nuevos trabajos .....	19
7. Ejercicio 4: Escuchar los mensajes .....	20
7.1 Introducción .....	20
7.1.1 Reconocer una llamada entrante .....	20
7.1.2 Recepción de mensajes .....	20
7.2 Objetivos .....	21
7.3 Requisitos .....	21
7.4 El programa Flowcode en detalle .....	21
7.5 Qué hacer .....	22
7.5.1 Detección de mensajes .....	21
7.5.2 Temporizador de intervalo de mensajes .....	22
7.5.3 Modificaciones del estado IDLE .....	23
7.5.4 Modificaciones del estado de RINGING .....	24
7.6 Nuevos trabajos .....	25
8. Ejercicio 5: Gestión automática de llamadas .....	26
8.1 Introducción .....	26

8.1.1	Automatización del proceso .....	26
8.2	Objetivos .....	27
8.3	Requisitos .....	27
8.4	El programa Flowcode en detalle .....	27
8.5	Qué hacer .....	28
8.6	Nuevos trabajos .....	28
9.	Ejercicio 6: Enviar un mensaje de texto .....	29
9.1	Introducción .....	29
9.1.1.	Introducción del SMS .....	29
9.1.2	Formato de los mensajes .....	29
9.1.3	Enviar un mensaje .....	29
9.1.4	Modificaciones de la macro Rx_Message .....	30
9.1.5	Modificaciones de la macro Tx_Command .....	30
9.2	Objetivos .....	31
9.3	Requisitos .....	31
9.4	El programa Flowcode en detalle .....	31
9.5	Qué hacer .....	32
9.6	Otros trabajos .....	33
10.	Ejercicio 7: Recibir un SMS .....	34
10.1	Introducción .....	34
10.1.1	Filtrar el mensaje entrante .....	34
10.2	Objetivos .....	35
10.3	Requisitos .....	35
10.4	El programa Flowcode en detalle .....	36
10.5	Qué hacer .....	37
10.6	Otros trabajos .....	37
11.	Ejercicio 8: Responder automáticamente a un mensaje de texto .....	38
11.1	Introducción .....	38
11.1.1	Tratamiento de mensajes .....	38
11.1.2	Descodificación de mensajes .....	38
11.1.3	Transmisión de la respuesta .....	38
11.3	Requisitos .....	39
11.4	El programa Flowcode en detalle .....	39
11.5	Qué hacer .....	40
11.6	Continuación de los trabajos .....	40

## 1. Introducción a la telefonía móvil

Los teléfonos móviles GSM (Global System for Mobile communication) son un medio de conectar a los usuarios a la red telefónica.

Al encenderse, el móvil busca una red adecuada. La red localiza el móvil para poder enviarle las llamadas entrantes.

Las estaciones base proporcionan conexión a través de una serie de células, normalmente representadas como hexágonos entrelazados. Durante una llamada, la voz, en forma de datos digitales, pasa de una estación base a otra. Al mismo tiempo, tanto el móvil como la estación base monitorizan la situación, juzgando si hay o no una célula mejor disponible. Cuando no está en una llamada, el móvil comprueba qué estaciones base puede alcanzar, y la red hace un seguimiento de la ubicación del móvil.

Cuando el móvil se desplaza, pasa de una célula a otra. La red conmuta las llamadas para que el cambio se produzca sin interrupciones.

### 1.1 Características -

#### **Tarjeta SIM (Subscriber Identity Module):**

La tarjeta SIM es una pequeña tarjeta inteligente que almacena el número de teléfono móvil y la agenda del usuario. También contiene varios números de serie únicos para el teléfono y el usuario, información de autenticación y detalles de la red y contraseñas del usuario.

#### **TDMA;**

Un móvil se conecta a una sola célula a la vez, pero esa célula puede estar conectada simultáneamente a varios otros móviles. Todas estas conexiones utilizan el acceso múltiple por división de tiempo (TDMA). El flujo de datos de cada móvil se divide en pequeños segmentos de unos 20 ms de duración. Estos segmentos se intercalan con los de los demás móviles. De hecho, se turnan para transmitir sus segmentos, todos en el mismo canal de frecuencia.

#### **Salto de frecuencia:**

Cada banda de frecuencia GSM (900 MHz, 1800 MHz y 1900 MHz) utiliza varios pares de frecuencias de subida (móvil a base) y de bajada (base a móvil) para transmitir mensajes. Para minimizar los efectos de las interferencias, el móvil y la base saltan de frecuencia (cambian entre pares de frecuencias) durante una llamada.

#### **Cifrado y autenticación:**

El estándar GSM utiliza dos niveles de seguridad:

- los datos se cifran antes de su transmisión;

- la red autentifica (comprueba la identidad) del móvil al establecer una llamada.

El planteamiento es similar para ambos niveles. La tarjeta SIM almacena una "clave" privada (número binario). La red tiene una copia de esta clave. La clave nunca se transmite.

Para autenticar un móvil, la red envía un número aleatorio al terminal. Éste se combina con la clave privada mediante un algoritmo de cifrado, y el resultado se transmite de vuelta a la red. Al mismo tiempo, la red realiza el mismo cálculo utilizando su copia de la clave privada. Si las dos respuestas coinciden, el móvil queda autenticado. Del mismo modo, el móvil y la red generan una clave cifrada utilizando un algoritmo diferente. El resultado se utiliza para cifrar cada paquete de datos.

**Itinerancia:**

La itinerancia permite al usuario de un teléfono GSM hacer y recibir llamadas utilizando cualquier red GSM, por ejemplo en el extranjero. El teléfono siempre intenta encontrar primero su red local, pero si falla, busca otras redes.

**SMS:**

Los mensajes del Servicio de Mensajes Cortos (SMS) pueden ser lentos debido al limitado ancho de banda disponible para transmitirlos. El envío de SMS es un sistema de almacenamiento y reenvío, en el que el mensaje se almacena en la red, que lo reenvía al móvil de destino cuando está accesible.

## 2. Resumen de los ejercicios

Los ejercicios pretenden introducir temas importantes, proporcionar información relevante y reforzar el proceso de aprendizaje mediante el desarrollo de soluciones de hardware y software que funcionen.

En el CD-ROM adjunto encontrará ejemplos de soluciones para cada ejercicio, que también podrá solicitar a su profesor. En todos los casos se pueden mejorar y desarrollar. Éstas, o la propia solución del alumno, pueden utilizarse como punto de partida para el debate y la demostración. Al final de cada ejercicio se proponen temas de desarrollo.

La documentación del módem GSM contiene información sobre una serie de funciones que no se utilizan en los ejercicios suministrados. Con la experiencia adquirida en este curso, los estudiantes más avanzados deberían ser capaces de incorporar estas características en desarrollos posteriores.

**Nota:**

Algunos comandos del módem forman parte de la especificación GSM y deben ser implementados total o parcialmente por todos los dispositivos GSM. Otros pueden ser específicos de un fabricante o modelo y deben utilizarse con cuidado si se requiere compatibilidad con otros dispositivos. La documentación suministrada proporciona detalles de cada comando soportado.

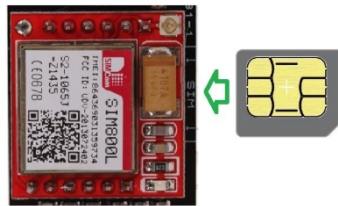
### 3. Configuración del hardware

#### 3.1 Configuración del módem GSM

El módem GSM es el corazón del sistema de telefonía móvil. Puede enviar y recibir señales a través de la antena y comunicarse con otros sistemas mediante protocolos de comunicación RS232.

**IMPORTANTE**

Necesitas una tarjeta SIM activa. El módem GSM necesita una tarjeta SIM de teléfono móvil con saldo. Como cualquier otro teléfono móvil, el módem GSM necesita una tarjeta SIM para funcionar. Tendrás que comprar una tarjeta SIM en el mismo lugar donde comprarías una para un teléfono móvil normal.

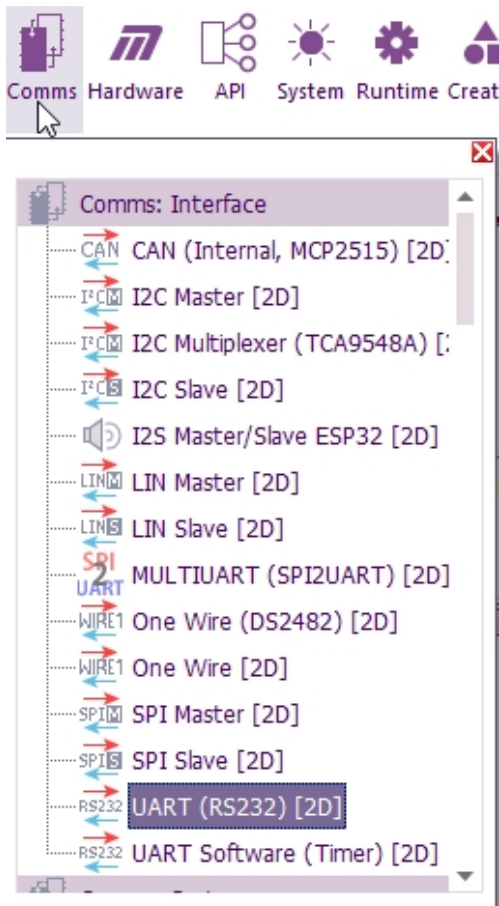


**Nota:**

Las tarjetas SIM de contrato no funcionarán debido al cifrado de datos, por lo que necesitarás una tarjeta SIM de "pago por uso". También tendrá que asegurarse de que la tarjeta SIM tiene saldo suficiente para los mensajes que vaya a enviar. El saldo de la tarjeta SIM puede recargarse del mismo modo que el de un teléfono móvil normal.

Para utilizar Flowcode, versión 8 o posterior, debe instalarlo en su ordenador local.

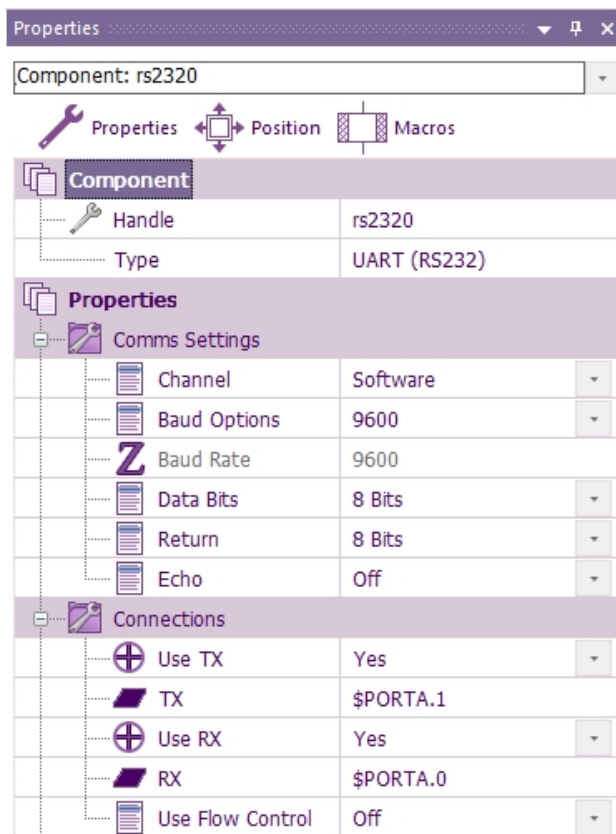
Los ejemplos de ejercicios Flowcode de este tutorial están disponibles en el CD-ROM (CD2130) suministrado con el kit de comunicaciones GSM.



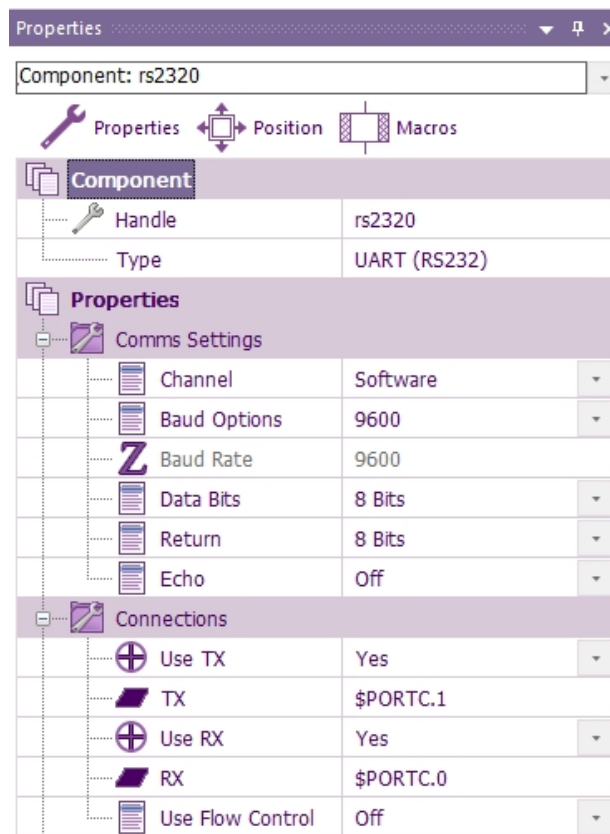
#### 3.2 Configuración del componente RS232

El componente RS232 de Flowcode se encuentra en la sección 'Comms'. de la Barra de Componentes, como se muestra aquí.

El componente Flowcode RS232 incluye una sección de Propiedades del Componente que permite configurar la velocidad de comunicación en baudios. Los ajustes ilustrados aquí deben utilizarse para todos los ejercicios.



BL0011



BL0055

### 3.2.1 BAUD tasa

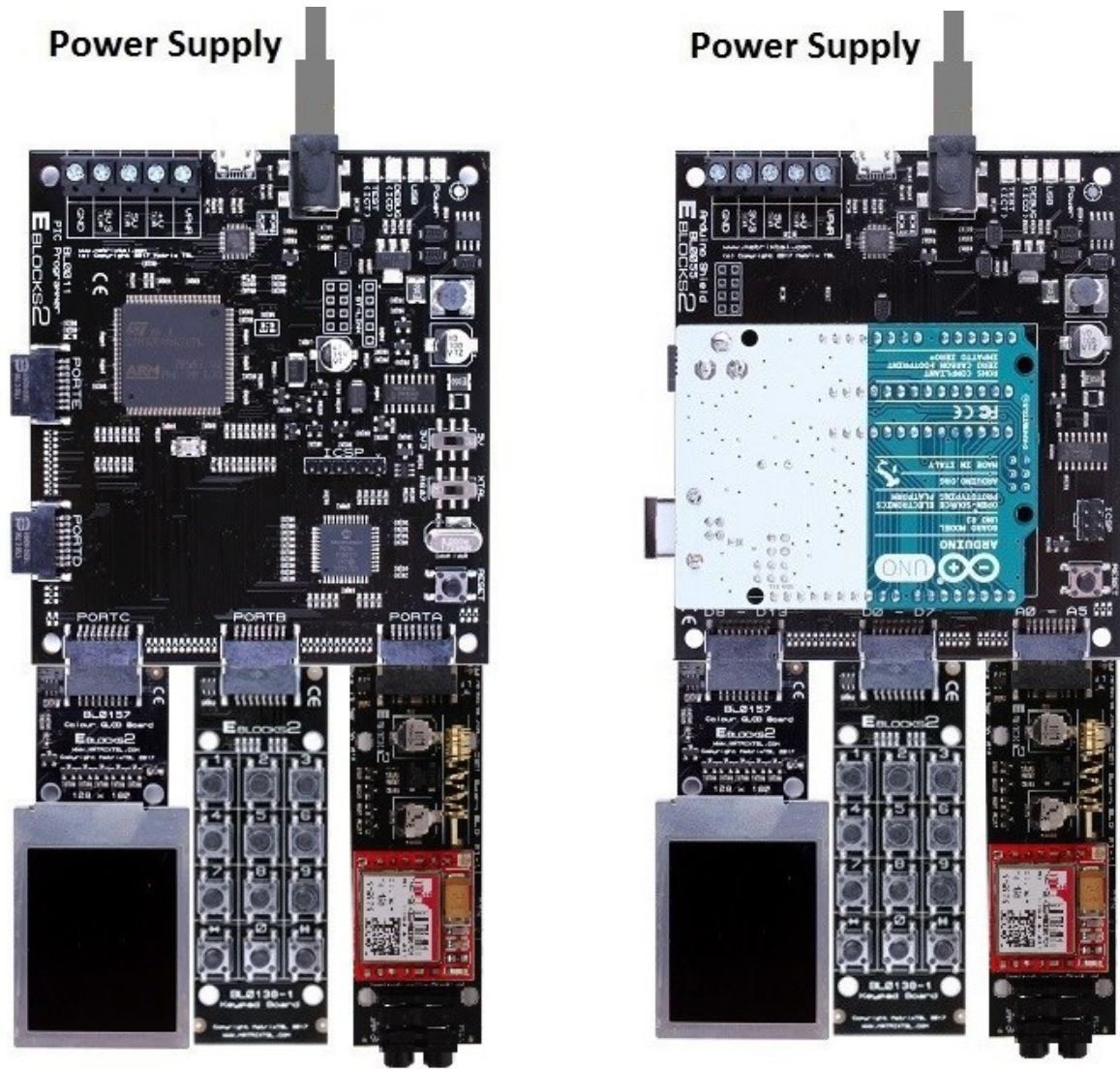
Número de bits de datos y de formato transmitidos o recibidos por segundo.

### 3.2 Configuración del sistema microcontrolador

Los ejemplos de ejercicios Flowcode que se pueden utilizar con este plan de estudios se centran en placas pro- cesoras y periféricas E-blocks2 específicas.

Las placas de procesador tienen el número de pieza BL0011 para la placa de procesador PIC y BL0055 para la placa Arduino Uno. Al utilizarlas como dispositivos de destino para Flowcode, se preconfigurarán todos los ajustes del procesador.

Las placas E-blocks2 deben fijarse a la placa del procesador como se muestra en la tabla y las imágenes siguientes.



PIC BL0011			Arduino BL0055		
Puerto A	Puerto B	Puerto C	A0-5 (C)	D0-7 (D)	D8-13 (B)
BL0131	BL0138	BL0157	BL0131	BL0138	BL0157

BL0131	Placa GSM E-blocks2
BL0138	Teclado E-blocks2
BL0157	Placa gLCD E-blocks2

## 4. Ejercicio 1: Un teléfono básico

### 4.1 Introducción

En este ejercicio se desarrollará un teléfono sencillo capaz de marcar un número precargado, responder a una llamada entrante y finalizar una conexión (colgar). Las funciones se controlarán manualmente mediante botones individuales del módulo de teclado.

Un teléfono debe ser capaz de realizar tres funciones básicas de control:

Realizar una conexión con otro teléfono seleccionado.

Aceptar una conexión desde otro teléfono. Desconectarse de un teléfono remoto.

#### 4.1.1 Comandos AT

El módem GSM permite realizar estas funciones mediante tres sencillos comandos AT:

ATD<número>;	(Dial)
ATA	(Respuesta)
ATH	(Hang-up)

Los comandos se transmiten al módem a través de la tarjeta RS232. La información se transmite carácter a carácter utilizando códigos del juego de caracteres ASCII.

#### 4.1.2 Caracteres ASCII

El conjunto de caracteres ASCII utiliza valores numéricos individuales para representar cada uno de los caracteres alfabéticos (mayúsculas y minúsculas), dígitos numéricos, signos de puntuación y códigos de control.

Los códigos ASCII útiles son:

<LF>	10	Avance de línea
<CR>	13	Retorno del carro
<CTRL-Z>	26	Fin del archivo
0 - 9	48 - 57	
A - Z	65 - 90	
a - z	97 - 122	

Una variable byte puede ajustarse a la mayoría de los valores de caracteres ASCII rodeando el carácter con comillas simples

Char= 'A'                      es equivalente a:                      Char= 65

La mayoría de los caracteres no imprimibles (<CR>, <LF>, etc.) sólo pueden leerse y escribirse utilizando sus valores numéricos.

#### 4.1.3 Cuerdas

Una cadena consiste en una serie de valores de bytes individuales en direcciones adyacentes, referenciados con un único nombre.

Los valores de los bytes individuales de una cadena suelen representar caracteres ASCII.

Una cadena termina con un byte cuyo valor numérico es 0

Las funciones de manipulación de cadenas de Flowcode operan sobre todo el contenido del array entre la dirección de inicio y el primer valor 0.

Una cadena también puede tratarse como una matriz de bytes, lo que permite acceder a cada una de sus posiciones mediante un puntero de índice.

Un ejemplo de suma de cadenas ilustra la estructura de una cadena y la forma en que pueden manipularse:

**Nombre="Matriz"**

Name	M	a	t	r	i	x					
index	0	1	2	3	4	5	6	7	8	9	10
Name[]	77	97	116	114	105	120	0				

Numérico puede incluir el contenido de una matriz. El array se referencia utilizando su nombre con el elemento individual referenciado por un valor del índice entre corchetes. cálculos individual

Nombre[2] contiene el carácter "t" que tiene el valor numérico ASCII 116

**Nombre= Nombre+ " Ltd"**

(nótese el espacio delante de la "L")

Name	M	a	t	r	i	x		L	t	d	
index	0	1	2	3	4	5	6	7	8	9	10
Name[]	77	97	116	114	105	120	32	76	116	100	0

**El nombre** contiene "Matrix Ltd"

El 0 que terminaba la cadena "Matrix" original (índice 6) se utiliza como punto de partida para la adición, sobrescribiéndose en el proceso.

El 0 al final de la cadena " Ltd" (índice 10) se convierte en el terminador de la nueva cadena compuesta cuando se suman las dos.

## 4.2 Objetivos

Los objetivos son:

- establecer un enlace de comunicación RS232 entre el microcontrolador y el módem;
- desarrollar una macro que permita enviar cadenas de comandos al módem;
- utilizar las entradas del microcontrolador para controlar la transmisión de comandos AT al módem;
- desarrollar un programa Flowcode que haga que el módem se comporte como un teléfono.

Para ello, los resultados del aprendizaje son:

- identificar el papel de cada componente en el sistema;
- configurar y controlar el componente RS232;
- configurar y controlar el componente del teclado;
- desarrollar, depurar y descargar un programa Flowcode;
- crear una macro Flowcode;
- comprender el almacenamiento y la manipulación de cadenas;
- comprender la transmisión de los comandos AT;
- comprender las funciones básicas de un teléfono.

### 4.3 Requisitos

- Una tarjeta multiprogramador conectada a un PC que ejecuta Flowcode
- Una tarjeta E-blocks2 con teclado
- Un módem GSM Placa E-blocks2 con una tarjeta SIM activa y auriculares de audio conectados
- Un teléfono en funcionamiento

### 4.4 El programa Flowcode en detalle

El programa principal contiene una variable de cadena llamada NÚMERO. Se coloca un bloque lógico en la parte superior del programa principal, para que sea fácil de localizar y editar. Esto inicializa NÚMERO, para que contenga el número de teléfono móvil utilizado en este ejercicio,

#### 4.4.1 Componente Flowcode RS232 - SendChar

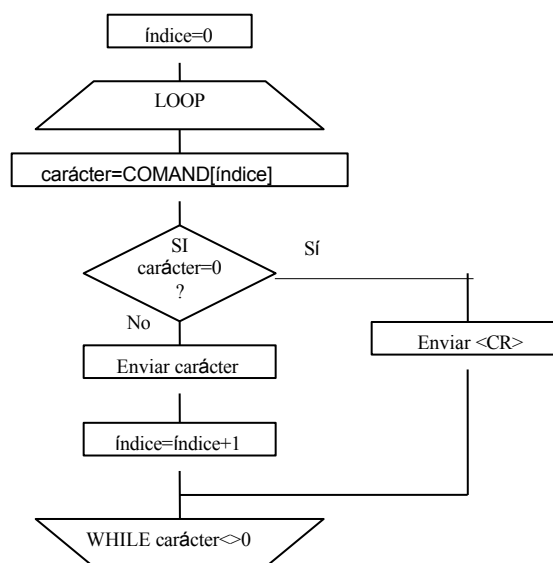
El componente RS232 de Flowcode debe cargarse en el programa para poder acceder a las funciones RS232. Una de estas funciones es SendChar, que permite la transmisión de un solo carácter desde el puerto RS232.

Todos los comandos del módem constan de varios caracteres. Cada carácter de un comando se puede transmitir con un uso individual de la función SendChar, pero algunos comandos constan de un gran número de caracteres, y el resultado sería un programa excesivamente complejo. Un punto de partida útil para éste, y muchos otros programas, es el desarrollo de una macro (subrutina) que permita transmitir grupos de caracteres (cadenas) utilizando un único comando. Además, todos los comandos de módem deben completarse con el carácter de retorno de carro <CR>. Esto puede ser difícil de incluir en una cadena utilizando caracteres imprimibles normales, pero la macro puede escribirse para añadir el carácter al final de cada secuencia.

#### 4.4.2 Macro - Tx\_Command

Esta macro se llama Tx\_Command. El programa principal contiene una variable de cadena llamada COMANDO. Cuando el macro se ejecuta, transfiere cada carácter en COMANDO a la función EnviarCarácter hasta que encuentra un carácter con el valor 0. Cuando se alcanza el 0, el macro transfiere el valor 13 (<CR>) a la función EnviarCarácter y regresa al programa principal. Cuando se alcanza el 0, el macro transfiere el valor 13 (<CR>) a la función SendChar y regresa al programa principal.

El siguiente diagrama de flujo resume esta macro:



**4.4.3 Uso de comandos AT**

La cadena de comandos a transmitir debe cargarse en COMMAND antes de ejecutar la macro. Los comandos ATA y ATH son sencillos, pero el comando ATD debe tener la cadena NUMBER y un punto y coma añadido. Ej.

**COMANDO="ATD "+NÚMERO+";'**

A continuación se muestra un ejemplo de cadena de comandos y el funcionamiento general de la macro Tx\_Command.

Número de teléfono



COMANDO	A	T	D	0	1	2	3	4	5	6	7	8	9	0	;					
índice	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
COMANDO [ ]	6 5	8 4	6 8	4 8	4 9	5 0	5 1	5 2	5 3	5 4	5 5	5 6	5 7	4 8	5 9	0				

**4.5 Qué hacer**

Este ejercicio requiere la transmisión de tres cadenas de mensajes, controladas por tres botones del teclado.

Asignaciones clave sugeridas:

<u>Clave</u>	<u>Función</u>	<u>Acción</u>
0	Marca el número	Envía el comando ATD<número>;.
#	Responder a la llamada	Enviar el comando ATA
*	Colgar	Enviar el comando ATH

El programa consta de tres copias de la sección de código que se indica a continuación, ejecutándose en un bucle continuo. Cada copia debe configurarse para probar una de las teclas asignadas y transmitir el comando AT asociado.

Utilizando el diagrama de flujo como guía, construya el programa Flowcode arrastrando los iconos apropiados al área de trabajo desde la barra de herramientas de iconos. A continuación, haga doble clic en el icono para abrir el cuadro de diálogo Propiedades que permite configurar la función.

En concreto, haga doble clic en el primer icono de manipulación de cadenas e introduzca el número de teléfono al que llamará para probar el programa. Esto se hace en el formulario:

NÚMERO= "XXXXXXXXXXXX"

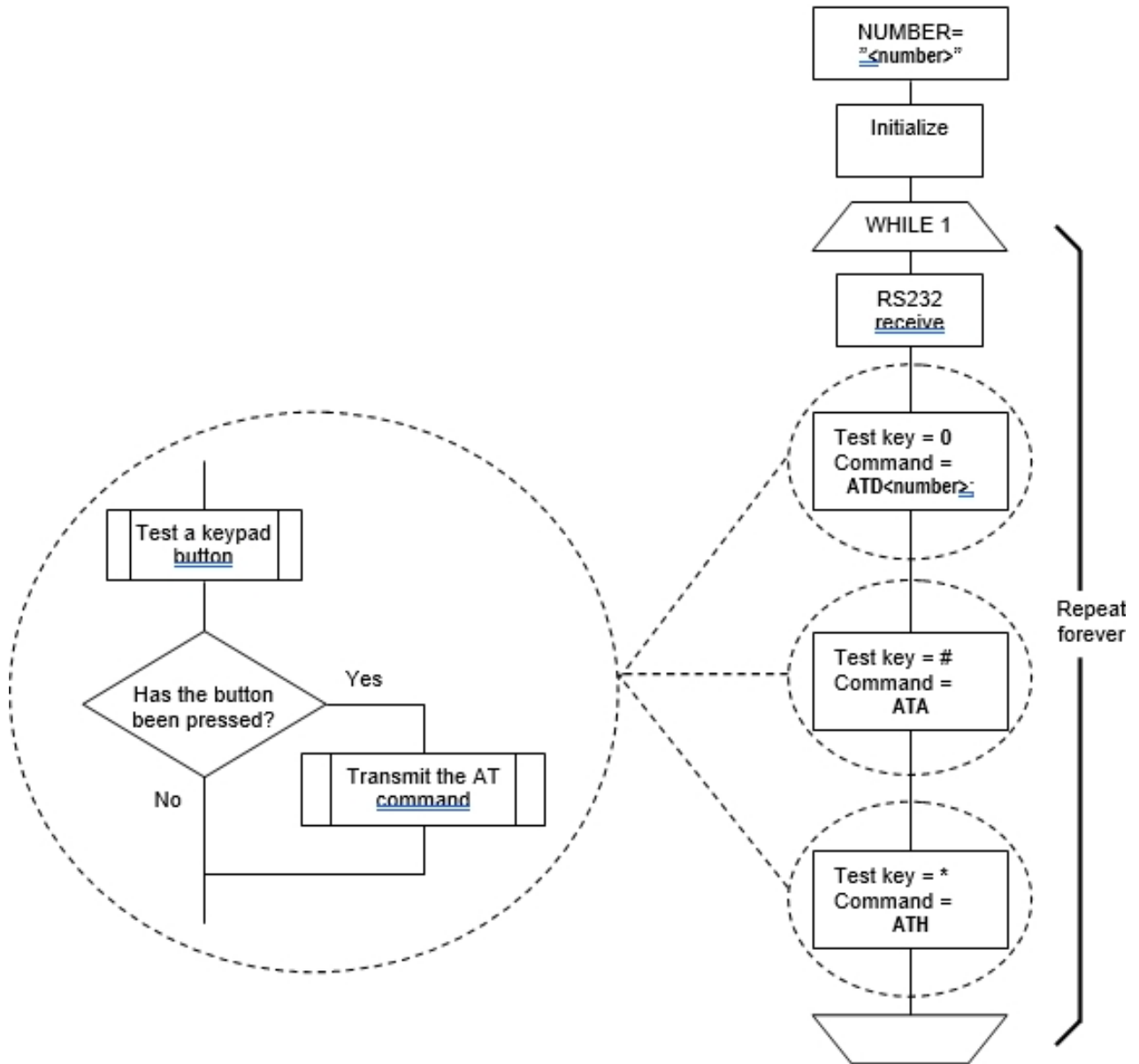
donde XXXXXXXXXXXXX representa el número de teléfono al que va a llamar.

**Nota:** La función "Recepción RS232" es necesaria porque el módem transmite mensajes de respuesta durante el funcionamiento y, si éstos no son aceptados, el búfer de transmisión del módem se llenará, impidiendo la continuación de la comunicación.

Los caracteres transmitidos por el módem se descartan en este ejercicio. Descarga el

programa en el microcontrolador.

Pruébalo ejecutando el programa y comprobando que cada una de las teclas asignadas funciona como se espera.



Telephone control structure

## 4.6 Otros trabajos

Actualmente, el programa sólo puede marcar un único número precargado. Amplíe el programa para asignar diferentes números de teléfono a más teclas numéricas del módulo del teclado; creando un útil teléfono de "marcación rápida".

## 5. Ejercicio 2: Una "Máquina de estados " sencilla

### 5.1 Introducción

En el ejercicio anterior se desarrolló una sencilla aplicación telefónica. Aunque el sistema resultante era funcional, adolecía de una serie de inconvenientes, uno de los cuales era la posibilidad de pulsar el botón equivocado en el momento equivocado, haciendo que el módem se comportara de forma impredecible.

Por ejemplo, si se pulsa la tecla de marcación cuando ya hay una llamada en curso, el módem pierde la conexión existente y no consigue obtener una nueva conexión.

En este ejercicio, se mejorará la funcionalidad del teléfono introduciendo una máquina de estados para controlar el estado actual del sistema y evitar acciones inadecuadas.

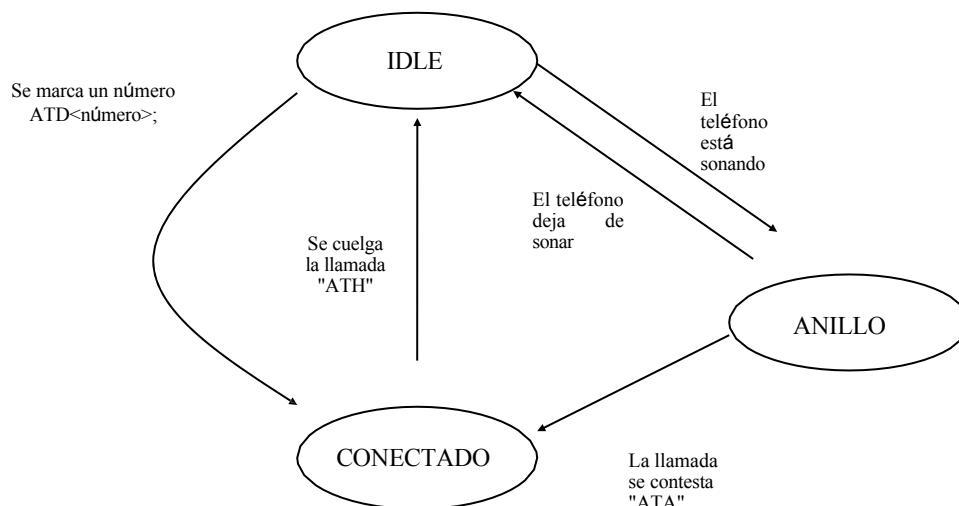
El funcionamiento de un teléfono, como el de la mayoría de los aparatos, puede dividirse en una serie de estados. Cada estado describe una de las condiciones en las que puede existir un dispositivo. El estado actual puede modificarse, de forma predefinida, por efecto de influencias externas o internas.

#### 5.1.1 Estado máquina

El funcionamiento del teléfono puede definirse mediante tres sencillos estados:

- **IDLE** - No hay llamadas conectadas. El teléfono puede detectar llamadas entrantes y permitirá llamadas salientes a marcar
- **TIMBRE** - Se detecta una llamada entrante. El teléfono puede permitir que se conteste la llamada.
- **CONECTADO** - Se ha marcado una llamada saliente o se ha contestado una llamada entrante. El teléfono puede permitir la finalización de la llamada.

No es necesario implementar todos los caminos posibles entre estados. En este caso no hay una ruta directa de CONNECTED a RINGING



**Diagrama de estado del teléfono**

Cada estado permite iniciar una única acción:

Estado	Acción	Controlar	Comando
IDLE	Marcar un número	1 tecla	ATD<número>;
ANILLO	Responder a una llamada	# clave	ATA
CONECTADO	Colgar	* clave	ATH

## 5.2 Objetivos

El objetivo es desarrollar una máquina de estados que mejore la funcionalidad de la aplicación telefónica desarrollada en el ejercicio 1.

Para ello, los resultados del aprendizaje son:

- comprender las ventajas del código estructurado;
- definir y aplicar máquinas de estado sencillas.

### Nota:

En este ejercicio, se pierde temporalmente la capacidad de responder a una llamada. En el siguiente ejercicio se desarrolla una versión mejorada.

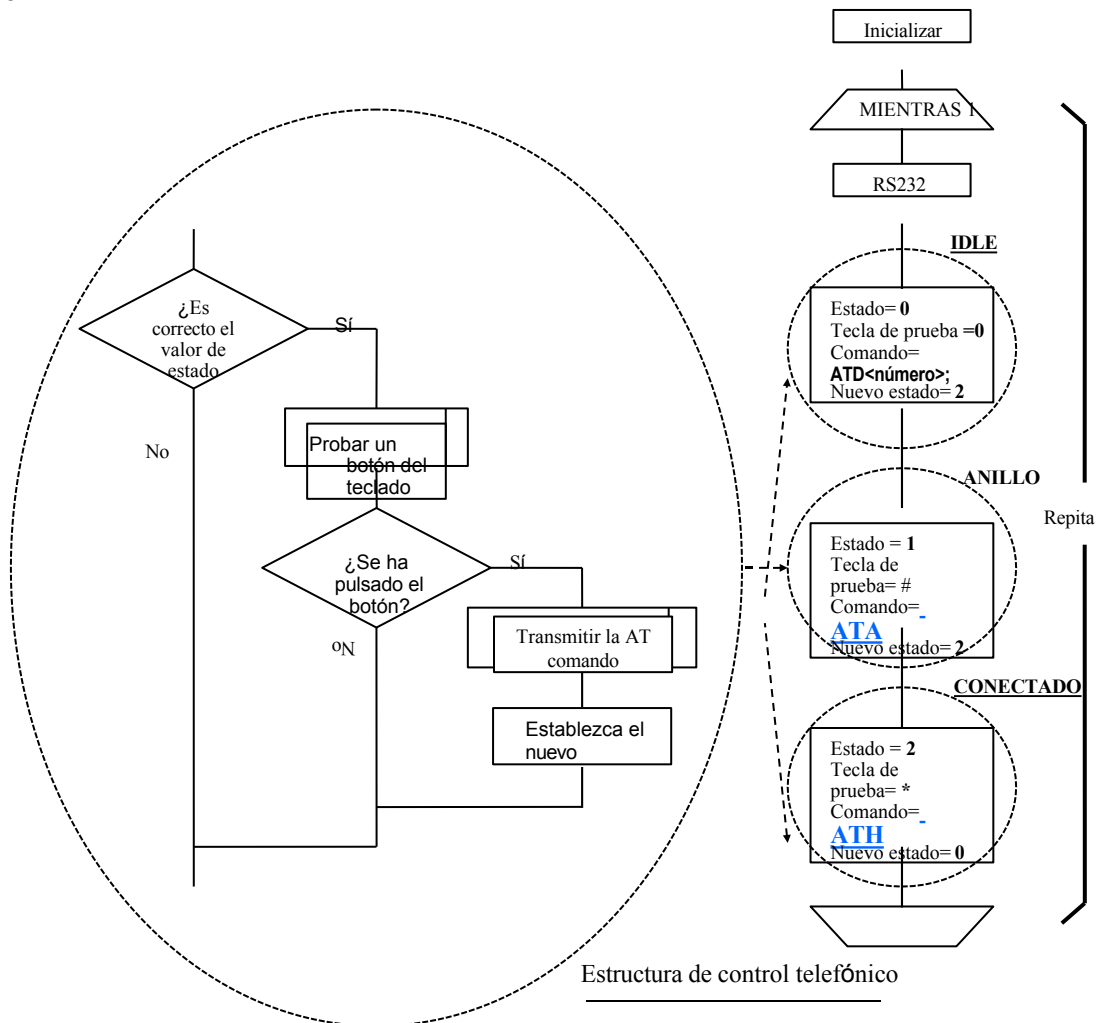
Este ejercicio se centra en la estructura del software y en la solución de algunos problemas prácticos. La pérdida de la función de respuesta a llamadas puede parecer un paso atrás, pero el trabajo realizado constituye la base del siguiente ejercicio y demuestra las ventajas del código estructurado.

## 5.3 Requisitos

- Una placa multiprogramador conectada a un PC que ejecuta Flowcode
- Una placa E-blocks2 con teclado
- Un módem GSM placa E-blocks2 con una tarjeta SIM activa y un auricular de audio conectado
- Un teléfono que funcione
- Una solución del ejercicio 1, o una copia de **Phone\_01.fcfx**.

### 5.4 El programa Flowcode en detalle

El siguiente diagrama presenta la estructura propuesta para el programa:



### 5.5 Qué hacer

A partir de la solución del ejercicio 1

- añade una variable de estado e inicialízala a 0;
- utilizar la variable de estado para decidir si se comprueba cada tecla;
- actualizar la variable de estado si se transmite un comando.

Valores de estado sugeridos

- 0 = INACTIVO
- 1 ANILLO=
- 2 = CONECTADO

Descárgalo y pruébalo del mismo modo que en el ejercicio 1.

### 5.6 Más información en

Determina los requisitos necesarios para que el sistema detecte una llamada entrante. Consulta la ficha técnica del módem para encontrar posibles soluciones al problema.

## 6. Ejercicio 3: Respuestas del módem

### 6.1 Introducción

En los dos primeros ejercicios, la comunicación con el módem ha sido esencialmente en una dirección. La recepción de caracteres del módem sólo se ha incluido en los programas para evitar que el búfer de transferencia del módem se llene y bloquee la comunicación.

El programa desarrollado en este ejercicio proporciona una visualización de los mensajes transmitidos por el módem, y genera software que puede hacer uso de ellos posteriormente.

#### 6.1.1 Visualización de los mensajes

El módem proporciona mucha información útil sobre su estado y el de cualquier conexión que esté realizando. La capacidad de leer y comprender estos mensajes mejora la funcionalidad de cualquier dispositivo desarrollado para utilizarlos y permite automatizar por completo las aplicaciones de conexión y monitorización de llamadas.

Los mensajes se transmiten al controlador como cadenas de caracteres ASCII, similares a los comandos que se le envían. El programa del ejercicio anterior puede modificarse para mostrar la información transmitida por el módem durante el funcionamiento normal.

Los caracteres pueden ser transmitidos por el módem por una de tres razones principales:

1. **Caracteres con eco:** Cada carácter transmitido al módem se transmite inmediatamente de vuelta (eco) para permitir al controlador comprobar la integridad de las comunicaciones.
2. **Mensaje de respuesta:** La ejecución de cada comando producirá una respuesta de confirmación. Puede ser un simple mensaje "OK", los datos solicitados o un mensaje de error.
3. **Mensajes no solicitados:** El módem puede transmitir mensajes que no son respuestas a comandos del controlador, sino que indican cambios en el estado del módem o de una llamada conectada.

La respuesta típica a un comando que se ha ejecutado correctamente, y no es necesario suministrar ningún dato es:

```
<CR><LF>OK<CR><LF>
```

La pantalla LCD se utilizará para mostrar las respuestas del módem. Para evitar confusiones, los mensajes se dividirán en dos grupos. Los caracteres con eco se mostrarán en la línea superior, y el resto de mensajes en la línea inferior.

#### 6.1.2 Componente Flowcode LCD

El componente LCD debe cargarse en el programa antes de poder utilizarse.

La función LCD Start debe ejecutarse antes de cualquier otro comando LCD.

La función Borrar LCD garantiza que el área de visualización esté en blanco y que el cursor se encuentre al principio de la línea superior.

Cada carácter enviado a la pantalla LCD se muestra en la posición actual del cursor. A continuación, el cursor se desplaza una posición hacia la derecha.

El cursor puede enviarse a cualquier posición, en cualquier línea, utilizando la función Cursor.

El cursor está configurado para ser invisible.

La función PrintAscii puede utilizarse para enviar un carácter cada vez.

## 6.2 Objetivos

El objetivo es desarrollar un programa Flowcode que reciba y muestre los caracteres transmitidos por el módem, utilizando el componente LCD.

Para ello, los resultados del aprendizaje son:

- reconocer el contenido y el formato de los mensajes de módem;
- identificar el significado y la finalidad de los mensajes transmitidos por el módem;
- desarrollar una estrategia para extraer mensajes del flujo de caracteres y utilizarlos para sus intenciones.

## 6.3 Requisitos

- Una tarjeta multiprogramador conectada a un PC que ejecute Flowcode
- Una tarjeta E-blocks2 con teclado
- Una placa LCD E-blocks2
- Un módem GSM Placa E-blocks2 con una tarjeta SIM activa y auriculares de audio conectados
- Un teléfono en funcionamiento
- Una solución del ejercicio 2, o una copia de Phone\_02.fcf.

## 6.4 El programa Flowcode en detalle

### 6.4.1 Mensaje caracteres

La función ReceiveChar que ya está en uso en el bucle principal del programa puede utilizarse como fuente de los caracteres visualizados recibidos del módem.

Para hacer esto, agregue una variable llamada 'rx\_char' al programa y úsela para recibir el valor retornado por ReceiveChar. Un valor de 255 indica que no se recibió ningún carácter antes de que la función expirara, así que este valor no debe ser enviado a la LCD. Cualquier otro valor representa un carácter recibido y debe ser enviado.

Las funciones LCD deben ser usadas para 'Iniciar' y 'Borrar' la LCD, y fijar la posición del cursor al inicio de la línea inferior, antes de entrar en el bucle del programa principal. Esto permitirá que todos los caracteres del mensaje sean reproducidos en la línea inferior del LCD. Los caracteres con eco serán manejados por el macro Tx\_command y mostrados en la línea superior de la LCD.

### 6.4.2 Caracteres con eco - macro Tx\_Command modificada

La macro Tx\_Command es un lugar útil para detectar caracteres con eco. La función LCD Clear borra la pantalla y coloca el cursor al principio de la línea superior cada vez que se envía un comando al módem.

La función ReceiveChar se utiliza dentro del bucle de la macro para recibir cada carácter, a su vez, desde el puerto RS232, y la función LCD PrintAscii luego lo muestra.

La función Cursor LCD sitúa el cursor al principio de la línea inferior cuando se han enviado todos los caracteres, lo que permite visualizar las respuestas en la línea inferior.

## 6.5 Qué hacer

Modifique el programa del ejercicio 2 siguiendo los pasos descritos anteriormente.

La macro Tx\_Command necesita ser modificada, nuevamente como se describe arriba, agregando íconos para limpiar el módulo LCD, cerca del comienzo de la macro, agregando una función ReceiveChar y una función LCD PrintAscii al final del bucle, y agregando una función LCD Cursor justo después del bucle.

A continuación, prueba el programa como se indica a continuación:

### 1. Reiniciar la placa controladora

- Utilice la tecla 0 del teclado para marcar el número precargado del teléfono donante.
- Tenga en cuenta los caracteres de eco en la línea superior de la pantalla y los caracteres de mensaje en la línea inferior - los caracteres en blanco adicionales son <LF> y <CR>, que no se pueden mostrar correctamente.
- Finalice la llamada a distancia colgando el teléfono del donante.
- Observe el nuevo mensaje añadido en la línea inferior de la pantalla. No se puede visualizar completamente pero debería ser "NO CARRIER" - el espacio entre NO y CARRIER es un espacio genuino, no un <LF>.
- Colgar la llamada localmente pulsando la tecla \*.
- Observe el nuevo conjunto de caracteres

de la pantalla. Reiniciar la placa controladora

- Marque el módem desde el teléfono donante.
- Observe los mensajes que se producen en la línea inferior de la pantalla.

Reiniciar la placa controladora

- Apague el módem y vuelva a encenderlo.
- Observe el mensaje que aparece en la línea inferior de la pantalla. Se trata de un mensaje no solicitado que se transmite cuando se enciende el módem y que podría causar problemas con otras funciones de comprobación de mensajes.

## 6.6 Más información en

El funcionamiento del módem se ve favorecido por la correcta recepción, interpretación y reacción a la información que envía al controlador.

La interpretación requiere la separación de los caracteres de eco de los caracteres de mensaje, y la separación de los mensajes de los caracteres de control circundantes.

- Desarrolle una rutina para eliminar todos los caracteres inesperados del módem antes de iniciar el programa principal.
- Desarrolle la macro Tx\_Command para eliminar todos los caracteres con eco recibidos por el puerto RS232, dejando sólo los caracteres de mensaje para ser tratados por el programa principal.

## 7. Ejercicio 4: Escuchar los mensajes de

### 7.1 Introducción

Los ejercicios anteriores dejaron un vacío importante en la funcionalidad del teléfono: la incapacidad de responder a una llamada entrante. Este ejercicio aborda este problema desarrollando una macro para detectar la presencia y persistencia del mensaje 'RING' como indicación de una llamada entrante, moviendo el sistema entre el estado 0 (IDLE) y el estado 1 (RINGING) automáticamente.

#### 7.1.1 Reconocer una llamada entrante

Cuando el módem detecta una llamada entrante, transmite el mensaje "RING" a intervalos regulares (aproximadamente 2 segundos). Si el controlador es capaz de detectar esta secuencia de caracteres, recibida por el puerto RS232, puede controlar automáticamente las transiciones entre los estados IDLE y RINGING. Tanto el estado IDLE como RINGING deben 'escuchar' el mensaje "RING" y realizar las siguientes acciones:

##### IDLE

Mensaje "RING" detectado:

- inicializar un temporizador con un periodo superior al tiempo previsto entre mensajes "RING";
- pasar el sistema al estado RING

##### ANILLO

Mensaje "RING" detectado:

- reinicializar el temporizador, con un periodo superior al intervalo previsto entre mensajes "RING", para evitar que se agote el tiempo de espera mientras se reciben los mensajes.

El temporizador se ha agotado:

- devolver el sistema al estado de reposo.

Llamada contestada (tecla \* pulsada):

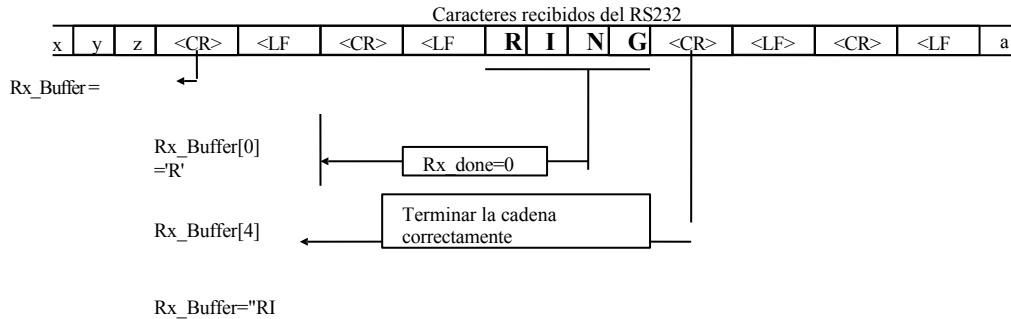
- enviar el mensaje ATA al módem;
- mover el sistema al estado CONECTADO.

#### 7.1.2 Mensaje recepción

El mensaje RING, y cualquier otro mensaje, puede ser detectado desarrollando una macro que guarde los caracteres entrantes como una variable de cadena (Rx\_Buffer). Este enfoque se basa en el hecho de que la macro Tx\_Command está ahora eliminando todos los caracteres con eco del flujo de datos entrantes.

Como se ha señalado anteriormente, todos los mensajes y respuestas comienzan y terminan con los caracteres <CR><LF>. Una forma de aislar el mensaje sería guardar todos los caracteres recibidos hasta que se detecte un <CR> y, a continuación, poner la cadena guardada a disposición del programa principal.

Este proceso se ilustra en el siguiente diagrama:



## 7.2 Objetivos

Este ejercicio tiene como resultado el desarrollo de un teléfono funcional estructurado para funcionar correctamente en todas las circunstancias. Esto implica:

- Desarrollar una macro para recibir mensajes y respuestas individuales del módem (pero no caracteres con eco).
- reconocer el mensaje "RING" como indicación de una llamada entrante.
- ajuste automático del estado del programa al detectar/perder el mensaje "RING"
- creando una función de temporización que:
  - mantiene el estado RINGING entre mensajes 'RING' consecutivos;
  - se agota si se dejan de recibir los mensajes "RING";
  - no suspende el funcionamiento del programa.

## 7.3 Requisitos

- Una placa multiprogramador conectada a un PC que ejecuta Flowcode
- Una placa E-blocks2 con teclado
- Una placa LCD E-blocks2
- Un módem GSM placa E-blocks2 con una tarjeta SIM activa y un auricular de audio conectado.
- Una solución del ejercicio 2, o una copia de **Phone\_03.fcfx**

## 7.4 El programa Flowcode en detalle

El programa lo hará:

- utilizar una variable byte (Rx\_index) para indexar Rx\_Buffer;
- utilizar otra variable de byte (Rx\_done) para informar al programa principal de la finalización del mensaje copiando Rx\_index en ella cuando se detecta el <CR>, (y ponerla a cero en todos los demás casos;) descartar el carácter <LF>;
- descarta el carácter <CR>, pero lo utiliza para indicar el final del mensaje.

Las secuencias <CR><LF><CR><LF> producirán una cadena de mensajes adicional cuando el segundo <CR> es recibido. Como la macro no guarda caracteres <CR> o <LF> el valor de Rx\_index copiado a Rx\_done será cero, por lo que el programa principal no detectará ninguna recepción.

## 7.5 Qué hacer

### 7.5.1 Mensaje detección

Crear una macro, Rx\_Message, para realizar la función de detección de mensajes.

Utilice un valor distinto de cero de la variable Rx\_done para forzar al programa principal a probar el mensaje en Rx\_Buffer.

Utilice la función de manipulación de cadenas, Compare\$, para detectar la presencia de 'RING' en Rx\_Buffer, y utilice una variable byte, match, para almacenar el resultado de la comparación, es decir match = Com-

pare\$("RING", Rx\_Buffer, 1)

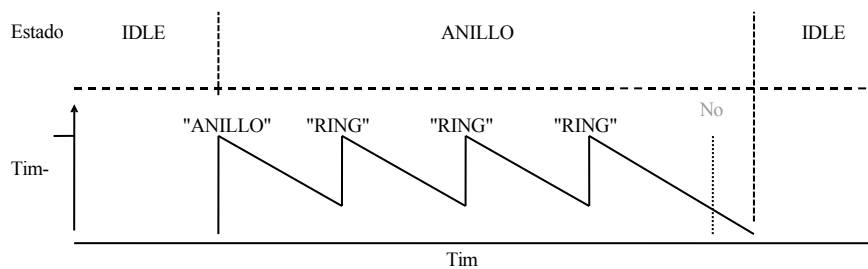
**Nota:** La función Compare\$ devuelve un cero si las dos cadenas coinciden

Utilice Rx\_Message para detectar las respuestas "OK" a los comandos del módem y confirmar la ejecución antes de cambiar la variable de estado.

### 7.5.2 Intervalo de mensajes timer

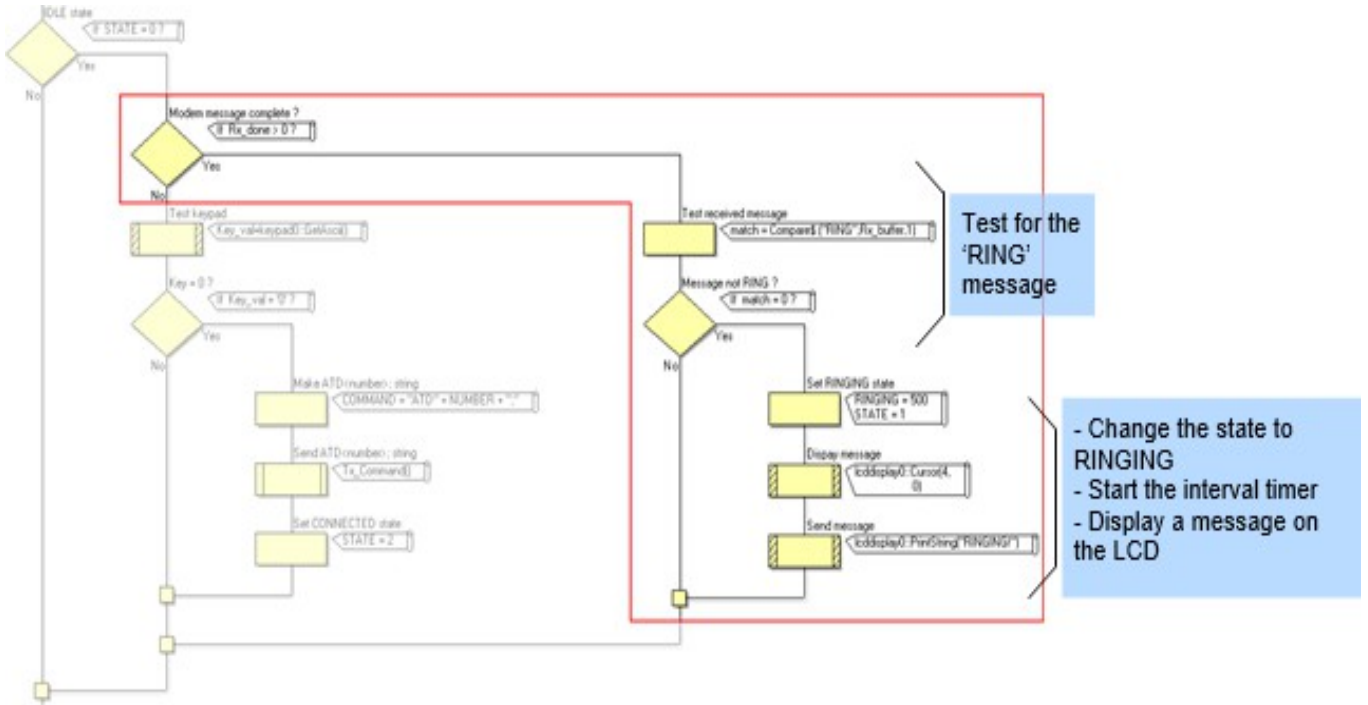
La función 'Retardo' de Flowcode permite introducir retardos de tiempo precisos en los programas, pero suspende el funcionamiento del programa durante estos periodos. En este ejercicio es necesario mantener el funcionamiento del programa en el período entre mensajes consecutivos 'RING', permitiendo que el puerto RS232 sea leído y que la tecla \* (respuesta) sea probada.

Para ello, el temporizador debe basarse en una variable entera que se pone a un valor (500) cuando se detecta un mensaje 'RING' en los estados IDLE o RINGING. El temporizador se reduce en 1 cada vez que se ejecuta el código de estado RINGING, hasta que llega a cero - time-out. Se evitará que el temporizador llegue a cero si se reciben mensajes 'RING' regulares, manteniendo el estado RINGING y permitiendo que se conteste la llamada. El periodo del temporizador no es exacto, debido a los efectos de otras partes del programa, pero es adecuado para esta aplicación.



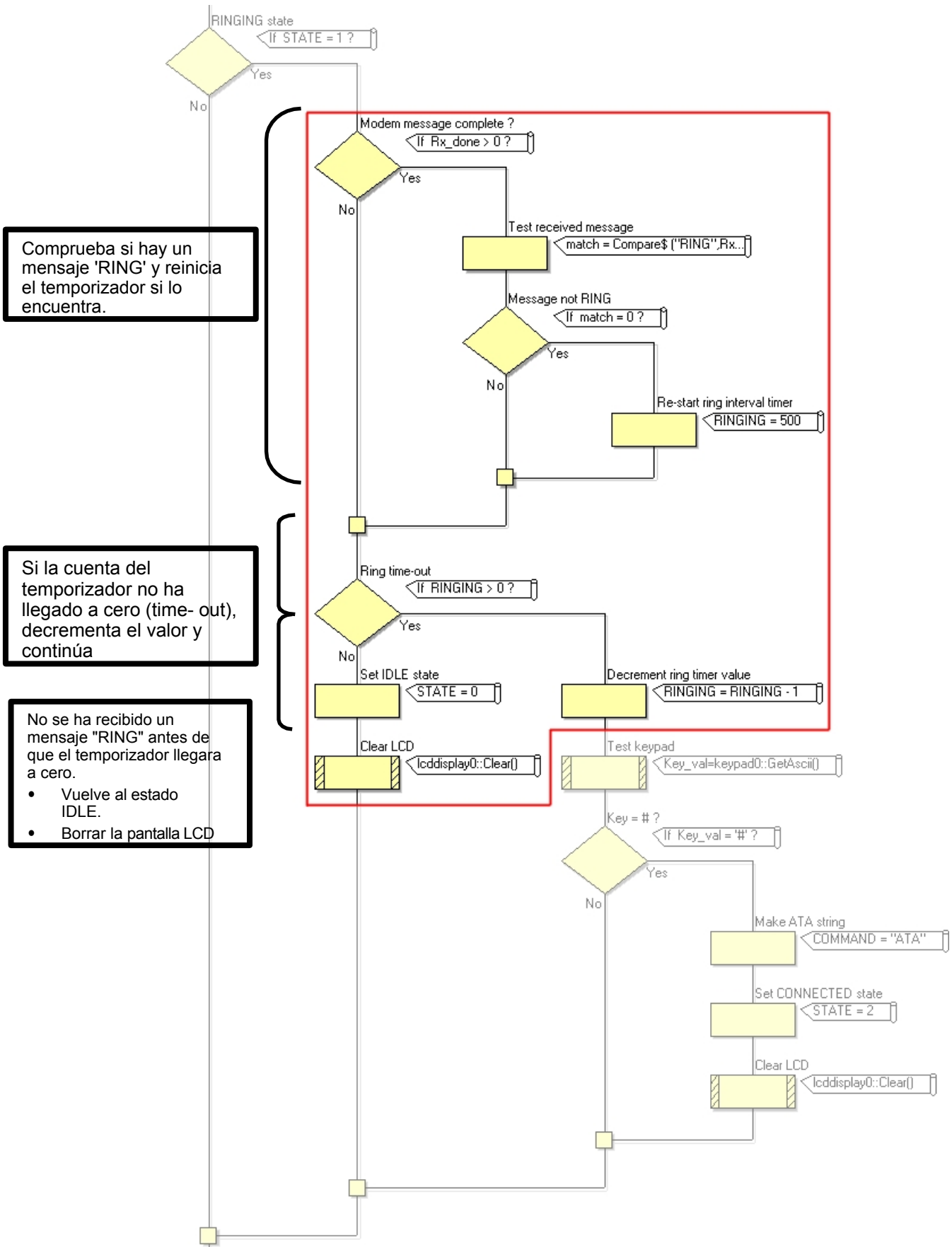
### 7.5.3 Estado IDLE modificaciones

El código de estado IDLE del ejercicio 2 puede modificarse para reconocer el mensaje 'RING' y realizar las funciones necesarias para cambiar al estado RINGING:



**7.5.4 Estado RINGING modificaciones**

El código de estado RINGING del ejercicio 2 puede modificarse para detectar el mensaje 'RING', gestionar la operación del temporizador y volver al estado IDLE si se produce un time-out. Los cambios necesarios se muestran a continuación:



## 7.6 Más información en

Las principales características que faltan en el teléfono desarrollado en este ejercicio son:

- indicación práctica de una llamada entrante: el auricular y la pantalla LCD son insuficientes;
- la posibilidad de marcar números alternativos sin editar el programa;
- visualización de la información

de la llamada. Mejoras sugeridas:

- Utilice la detección del mensaje "RING" para crear una indicación más práctica de una llamada entrante.
- Construye una cadena de números introducidos desde el teclado, y utiliza el resultado con el comando ATD para iniciar una llamada.
- Muestra en la pantalla LCD el número que se está marcando y permite editarlo antes de realizar la llamada.
- El módem puede transmitir varios mensajes diferentes que indican el estado de una conexión de llamada (consulte la documentación). Pruebe algunos de estos mensajes, además de 'RING', y haga que el programa responda correctamente.
- (Avanzado) El módem puede proporcionar información sobre las llamadas entrantes (consulte la documentación). Envíe los comandos AT apropiados al módem, compruebe las respuestas esperadas y muestre la información en la pantalla LCD.

## 8. Ejercicio 5: Gestión automática de llamadas

### 8.1 Introducción

Los ejercicios anteriores han dado como resultado el desarrollo de una aplicación análoga a un teléfono original accionado mecánicamente. En este ejercicio las ventajas de tener un microcontrolador en el sistema serán explotadas desarrollando un teléfono que conteste automáticamente las llamadas entrantes y cuelgue correctamente cuando las llamadas terminen. La detección de mensajes específicos del módem hará que el controlador transmita las cadenas de comandos AT necesarias.

#### 8.1.1 Automatización del proceso

El controlador puede programarse para responder a la detección de mensajes de módem específicos y enviar automáticamente los comandos de control adecuados al módem.

El controlador ya está programado para detectar el mensaje 'RING', pero sólo responde mostrando un mensaje en la pantalla LCD y gestionando un temporizador. Esto requiere una intervención manual para transmitir el comando "ATA" y responder a una llamada. El controlador puede ser reprogramado para enviar el comando "ATA" en la primera detección del mensaje "RING", mientras que en el estado IDLE, y pasar directamente al estado CONNECTED, lo que resulta en una instantánea, automática, función de respuesta.

El módem envía el mensaje 'NO CARRIER' cuando se termina una llamada de forma remota. La detección de este mensaje, mientras se está en el estado CONECTADO, permitiría al controlador enviar el mensaje ATH y pasar directamente al estado IDLE.

Partiendo de la solución del ejercicio anterior, es posible modificar el código de la siguiente manera: El código de estado IDLE responde sólo a la detección del mensaje 'RING', no a la tecla de marcación.

El código de estado RINGING responde a la llamada inmediatamente, sin esperar a la tecla de respuesta, ni a ningún mensaje 'RING' posterior.

El código de estado CONNECTED cuelga la llamada al recibir un mensaje de fallo de línea (NO CARRIER), no la tecla HANG-UP.

Si el dispositivo no da ninguna indicación de que está respondiendo a una llamada entrante, y el auricular suministrado se vuelve a colocar con un circuito de micrófono sensible, el resultado podría utilizarse como un dispositivo de escucha de largo alcance.

Sustituir los auriculares por un amplificador de audio y un altavoz más potentes permitiría que el dispositivo fuera utilizado como sistema de megafonía de acceso remoto.

#### **Nota:**

El sensible micrófono y el potente altavoz no deben utilizarse al mismo tiempo debido a los problemas de retroalimentación de audio.

## 8.2 Objetivos

Este ejercicio tiene como resultado cambiar la función del dispositivo de un teléfono normal a un enlace de audio con respuesta automática.

Esto se consigue mediante:

- reconocer la función de los distintos bloques de código;
- editar y reordenar las secuencias de bloques de código para modificar el funcionamiento del dispositivo.

## 8.3 Requisitos

- Una placa multiprogramador conectada a un PC que ejecuta Flowcode
- Una placa E-blocks2 de módem GSM con una tarjeta SIM activa y unos auriculares de audio conectados.
- Una solución del ejercicio 4, o una copia de **Phone\_04.fcx**

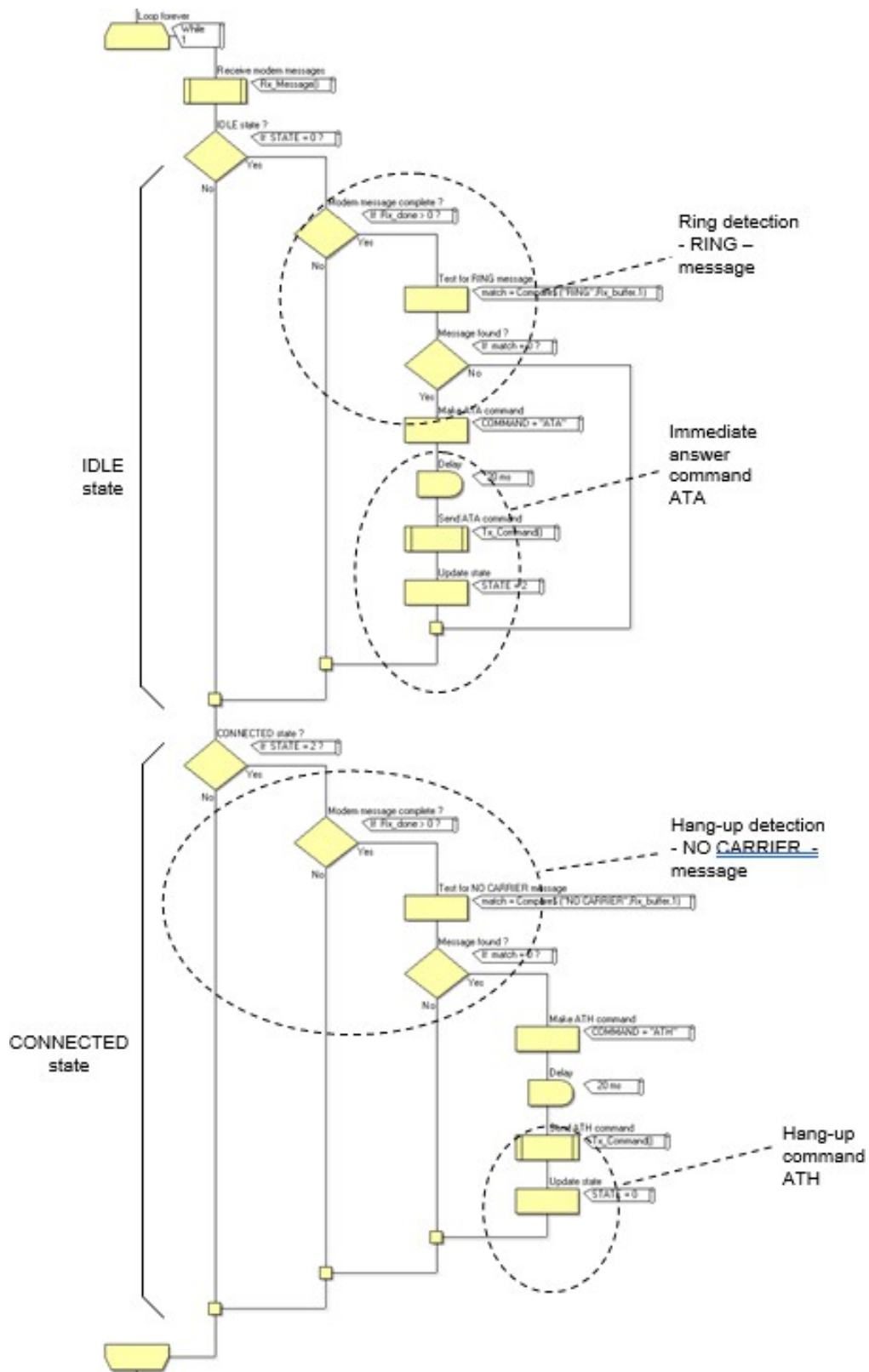
## 8.4 El programa Flowcode en detalle

El programa del ejercicio 4 se modifica como sigue:

- la detección de la tecla de marcación sale del estado IDLE, ya que este teléfono sólo responde a las llamadas entrantes;
- se suprime todo el estado RINGING, ya que el teléfono se contesta inmediatamente desde el estado de reposo. estado;
- el comando ATA se emite directamente desde el estado IDLE cuando se detecta el primer mensaje 'RING';
- la detección de la tecla de colgar en el estado CONECTADO se sustituye por la detección del mensaje "NO CARRIER".

### 8.5 Qué hacer

Modifica el programa del ejercicio 4 como se ha descrito anteriormente, y como se muestra en el siguiente diagrama:



### 8.6 Más información en

Añade seguridad al sistema detectando el identificador de llamada y contestando sólo si coincide con uno de los números preconfigurados (consulta la documentación del módem).

## 9. Ejercicio 6: Enviar un mensaje de texto

### 9.1 Introducción

La mensajería de texto SMS (Short Message Service) forma parte integrante de la norma GSM desde su creación. En los estándares GSM se ha incluido un conjunto de comandos AT que permiten enviar, recibir, almacenar, recuperar y formatear mensajes. Los ejercicios restantes representan una breve introducción a las funciones de mensajería de texto, pero dan lugar al rápido desarrollo de potentes aplicaciones.

Debido a la naturaleza de la mensajería de texto, es necesario introducir algunas de las complejidades del conjunto. Todas las soluciones se basan en los conceptos desarrollados en la sección de audio y no deberían plantear problemas significativos a quienes hayan completado con éxito los ejercicios de audio.

#### 9.1.1. SMS introducción

Las funciones de mensajes de texto del módem GSM se gestionan de forma similar a las llamadas de voz, mediante comandos AT, pero con un conjunto diferente de funciones.

Existen varias opciones para controlar la forma en que el módem gestiona y presenta los mensajes. Éstas se configurarán en el código de inicialización para conseguir los resultados deseados

#### 9.1.2 Formato del mensaje

Es mejor configurar el módem antes de intentar transmitir o recibir cualquier mensaje. El comando AT+CMGF establece el formato de los mensajes de texto y la mayoría de las respuestas del módem. Los mensajes de texto pueden enviarse en formato PDU (Unidad de datos de protocolo) comprimido o en formato de texto.

- El formato PDU es eficiente (7 bits por carácter), pero puede ser difícil de decodificar en texto legible.
- El formato texto utiliza un código ASCII estándar de 8 bits para cada carácter del mensaje, que puede transferirse directamente a variables de cadena.

El comando AT necesario para configurar esta opción es el comando Formato de mensaje: AT+CMGF=1

#### 9.1.3 Envíe un mensaje a

En el modo de voz, los comandos AT se utilizan para establecer y controlar una conexión de llamada, pero el módem gestiona los datos de voz por separado utilizando circuitos de audio adicionales.

En el modo de texto SMS, toda la información de control y datos pasa a través del puerto serie en forma de comandos, respuestas y mensajes.

El comando AT para enviar un mensaje de texto es: AT+CMGS=<"número">

El módem responde transmitiendo un mensaje <CR><LF>> cuando está preparado. (Observe el espacio después del carácter > y la ausencia de un carácter <CR> o <LF> como terminación). Esto indicaría la posición de inicio del texto si el sistema estuviera siendo controlado desde la pantalla y el teclado de un terminal.

El texto del mensaje puede transmitirse al módem tras la recepción de los caracteres '> ', y toda la secuencia finaliza con el carácter <CTRL-Z> - valor 26 - un histórico marcador de "Fin de archivo".

Aquí está la secuencia completa:

Transmitir	AT+CMGS=<"número"><CR>
Espere	<CR><LF>><espacio>
Transmitir	<mensaje>
Transmitir	<CTRL-Z>

A continuación, el módem compila una cabecera de mensaje, añade el texto <mensaje> y transmite los datos resultantes a la red SMS.

### 9.1.4 Rx\_Message macro modificaciones

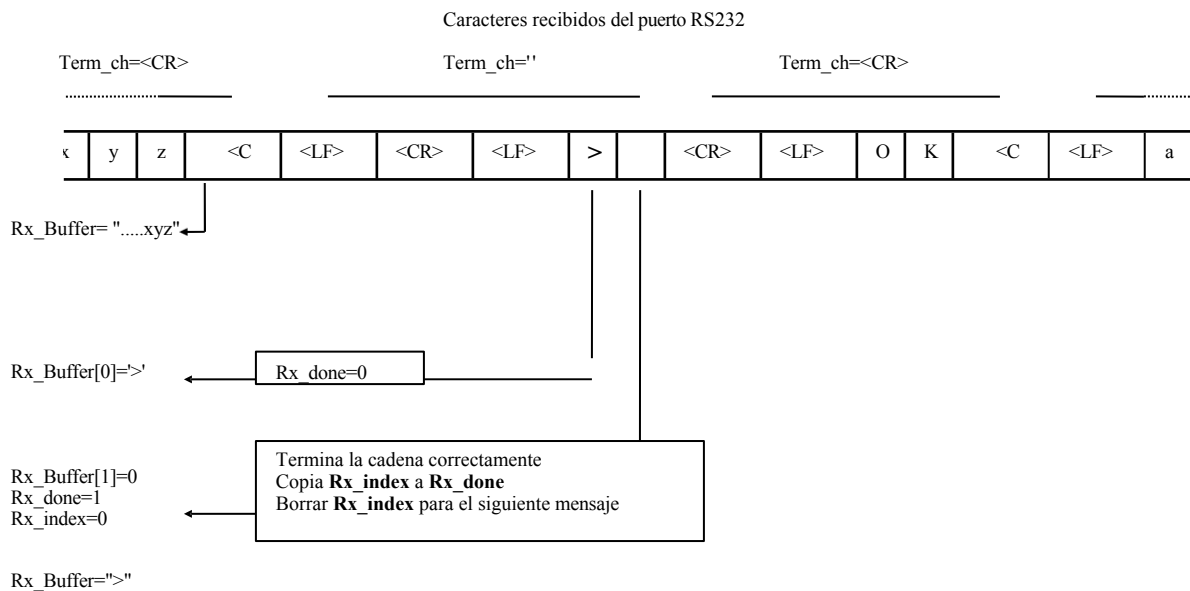
La gama de formatos de mensaje a detectar en modo SMS es mayor que en modo audio. La detección de los caracteres ">" no puede lograrse con la macro Rx\_Message existente debido a la ausencia de una función carácter <CR>. La macro Rx\_Message debe modificarse ahora para utilizar un carácter suministrado por el programa principal para detectar la finalización de un mensaje, o segmento de mensaje.

Utilice una variable byte para contener el carácter de terminación del mensaje (Term\_ch) y establezca el valor requerido desde el programa principal.

La macro debe comparar cada carácter recibido con Term\_ch, en lugar de con <CR>, pero realiza la operación mismas funciones de finalización de mensajes que las desarrolladas anteriormente.

Como antes, <CR> y <LF> no deben añadirse a la cadena Rx\_Buffer.

Es posible que el valor de Term\_ch deba modificarse periódicamente, en función de los mensajes esperados.



### 9.1.5 Tx\_Command macro modificaciones

La macro Tx\_Command original transmite un carácter <CR> al final de cada cadena de comandos. Ahora hay una serie de requisitos para que las cadenas se transmitan sin la adición automática de <CR>. Esto permitirá que los comandos se envíen al módem en secciones y que el <CR> sólo se agregue a la última sección.

Añada una variable byte (Send\_CR) y utilízela para controlar la transmisión del carácter <CR> por la macro Tx\_Command: 0=no enviar.

Cada comando, o segmento de comando, debe copiarse en la cadena COMMAND, y el valor de Send\_CR establecido antes de ejecutar la macro Tx\_Command.

## 9.2 Objetivos

El objetivo es desarrollar un programa Flowcode que transmita un mensaje de texto precargado a un número precargado, cuando se pulse una tecla del teclado asignada.

## 9.3 Requisitos

- Una placa multiprogramador conectada a un PC que ejecuta Flowcode
- Una placa E-blocks2 con teclado
- Un módem GSM placa E-blocks2 con una tarjeta SIM activa (auriculares de audio no es necesario)
- Un teléfono móvil capaz de recibir mensajes de texto (el número de marcación de este teléfono debe introducirse en la cadena de comandos AT+CMGS del software Flowcode).

## 9.4 El programa Flowcode en detalle

Los pasos de este programa son:

- inicializar el módem para que funcione correctamente en modo texto;
- iniciar una secuencia de transmisión de mensajes de texto al pulsar una tecla;
- espere la respuesta requerida del módem;
- enviar y terminar el texto del mensaje.

El programa Flowcode transmite el comando AT+CMGS="**<número>**" cuando se pulsa la tecla '#' del teclado.

El número a marcar, **<número>**, se introduce en una cadena en la parte superior del programa para facilitar su localización y modificación.

No es posible incluir el carácter " en una cadena, por lo que este carácter se transmite de forma independiente, antes y después de la transmisión de la cadena **<número>**.

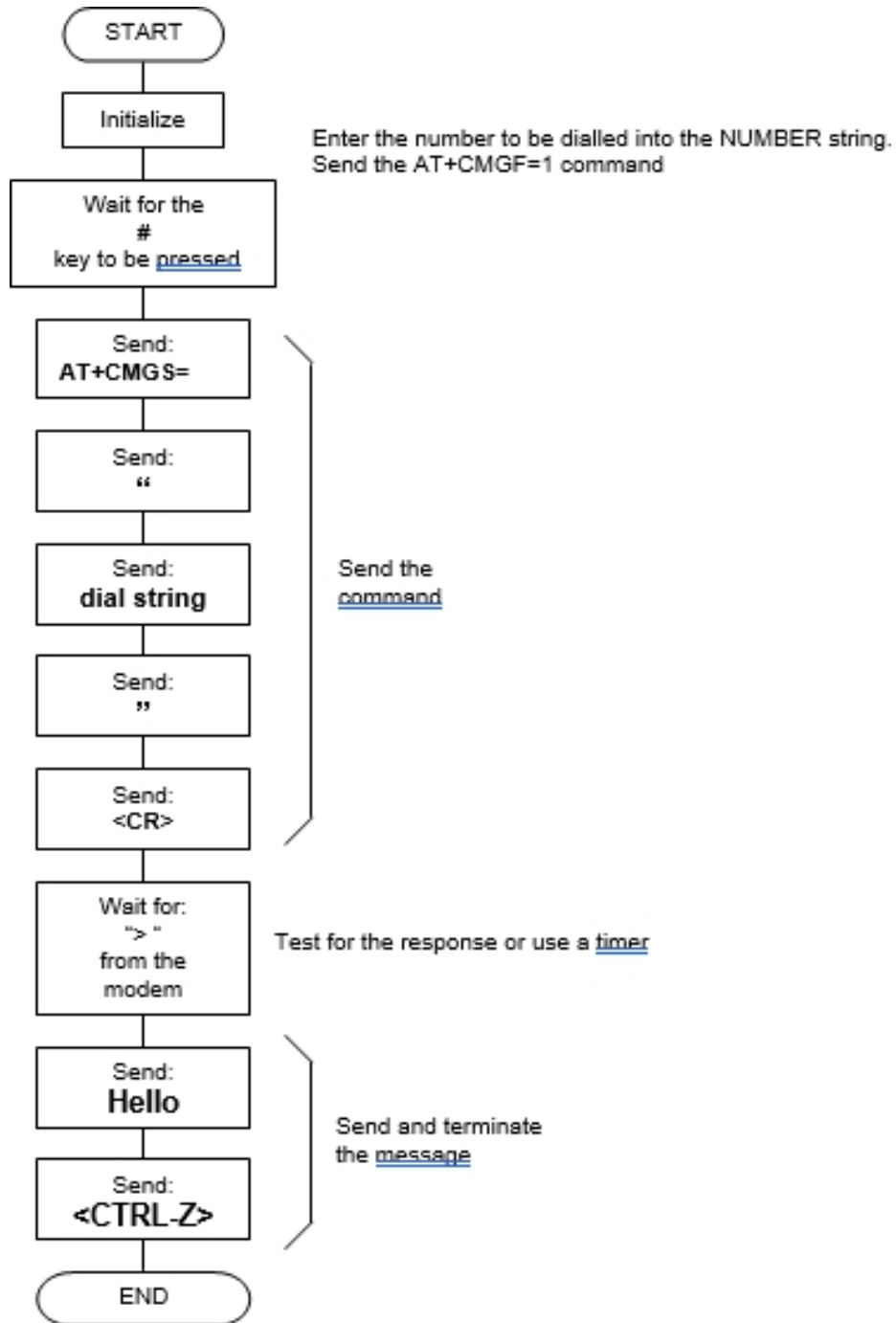
Es importante esperar a que el módem genere la respuesta "> " antes de enviar el mensaje de texto, por lo que el código lo permite. La nueva macro Rx\_Message detecta el espacio después del carácter > y comprueba el Rx\_Buffer para el carácter >.

### Nota:

La función Compare\$ no funciona de forma fiable con cadenas de caracteres simples, por lo que se utiliza una función de cálculo para comparar Rx\_Buffer[0] con '>' directamente.

Cuando el módem está preparado para recibir el mensaje de texto, éste debe enviarse y terminar con un carácter <CTRL-Z> (26). Es necesario poner a 0 la variable Send\_CR antes de la transmisión para evitar que se añada el carácter <CR> después de <CTRL-Z>.

Una vez finalizado, el programa termina para evitar la transmisión accidental de más mensajes. La estructura general del programa se muestra en el siguiente diagrama.



### 9.5 Qué hacer

- Como ya se ha señalado:
  - modificar la macro Tx\_Command;
  - modificar la macro Rx\_Message.
- Inserte un icono de función String y configure el contenido con el número del teléfono móvil a llamar, y el mensaje de texto a enviar.
- Establezca el modo Texto, utilizando el comando AT+CMGF=1.
- Crea un bucle para esperar hasta que se pulse la tecla '#'.
- Transmite el número de teléfono móvil al que desea llamar.
- Comprueba si ese teléfono móvil responde, esperando el mensaje ">".
- A continuación, transmite el mensaje de texto y termina la llamada.

## 9.6 Más información en

El programa desarrollado en este ejercicio permite la transmisión de un único mensaje de texto SMS cuando se pulsa una tecla del teclado.

- Desarrolle el programa para que realice un bucle continuo y supervise una serie de eventos de entrada.
- Transmitir mensajes diferentes para cada evento.
- Incluir datos variables en los mensajes transmitidos

### **Nota:**

Una solución para este ejercicio permite desarrollar numerosas aplicaciones en el campo de la telemática a distancia. El sistema puede programarse para transmitir mensajes significativos cuando se activan por una serie de eventos, entre ellos: condiciones de entrada digital; niveles de entrada analógica; o valores de temporizador. Los mensajes pueden contener hasta 140 caracteres e incluir contenido de texto variable y valores de datos reales; se trata principalmente de un ejercicio de manipulación de cadenas.

El sensor de temperatura suministrado con esta solución puede utilizarse para simular una serie de aplicaciones industriales y domésticas, y generar alarmas por mensaje de texto cuando la temperatura detectada alcanza niveles preestablecidos. Alternativamente, los valores numéricos de las lecturas del sensor pueden transmitirse a intervalos regulares para proporcionar un registro remoto de la temperatura a lo largo del tiempo.

## 10. Ejercicio 7: Recibir un mensaje de texto

### 10.1 Introducción

Un mensaje de texto se recibe como un flujo de caracteres que incluyen grandes cantidades de datos además del texto original. El módem puede configurarse para responder de varias maneras a la recepción de un mensaje de texto.

Algunas de las posibles respuestas son:

- Guardar el mensaje de texto en la memoria de la tarjeta SIM (el mensaje se puede encontrar leyendo algunos de los parámetros del módem más adelante).
- Guarda el mensaje de texto en la memoria de la tarjeta SIM y transmite un mensaje no solicitado al controlador indicando su ubicación.
- Transmite el mensaje de texto al controlador inmediatamente y no lo guarde en la memoria de la tarjeta SIM.

Los siguientes ejemplos configuran el módem para producir esta última respuesta, ya que proporciona resultados inmediatos y no llena la memoria de la tarjeta SIM. El comando AT necesario para conseguirlo es el comando New Message Indication:

```
AT+CNMI=2,2
```

Un mensaje recibido del módem tendrá la siguiente forma:

```
+CMT: "+441234567890",,"07/02/27,09:14:15+00"<CR><LF>Hello<CR><LF>
```

<b>+CMT:</b>	Cabecera del mensaje
<b>"+441234567890"</b>	Identificador de llamadas
<b>07/02/27</b>	Fecha
<b>09:14:15+00</b>	Hora+ desfase horario local
<b>&lt;CR&gt;&lt;LF&gt;</b>	Terminación de la cabecera del mensaje
<b>Hola</b>	Mensaje de texto
<b>&lt;CR&gt;&lt;LF&gt;</b>	Terminación del mensaje de texto

Toda esta información es potencialmente útil y puede extraerse del flujo de datos utilizando la macro RX\_Message mejorada del ejercicio anterior.

Para este ejercicio, sólo se utilizará la cabecera del mensaje, para indicar la detección, y el propio mensaje de texto.

#### 10.1.1 Filtrado del mensaje entrante

El mensaje entrante puede dividirse en segmentos utilizando la macro RX\_Message. La selección correcta del carácter de terminación permitirá aislar y extraer la información necesaria. Se puede utilizar una sencilla máquina de estados para contar los segmentos del mensaje, controlar el funcionamiento del programa y seleccionar el carácter de terminación adecuado.

Una variable de byte (SEGMENTO) controla la función de detección de segmentos de mensajes.

La detección de la cabecera del mensaje puede lograrse estableciendo el carácter terminador de RX\_Message en ' ' (el carácter de espacio). Cuando la macro indica la recepción de un mensaje, la cadena del búfer puede compararse con "+CMT:".

Si se detecta la cabecera +CMT:, el programa incrementa el valor de SEGMENTO y procede a la detección del identificador de llamada, la fecha y la hora. Esta información no es necesaria para este ejercicio, por lo que si se establece el carácter delimitador en <CR> se recuperarán todos los caracteres en una sola operación.

La detección del primer carácter <CR> permite incrementar la variable SEGMENTO, indicando que los siguientes datos serán el mensaje de texto original.

La detección del segundo carácter <CR> indicará la finalización de la recepción del mensaje, permitiendo que el valor de SEGMENTO se restablezca al valor de detección de cabecera y que el carácter delimitador se restablezca a ' '.

El mensaje de texto original se almacena como una cadena en 'Rx\_Buffer', lo que permite visualizarlo, probarlo o manipularlo, según requiera la aplicación. En este caso se visualiza.

**Por ejemplo:**

El mensaje generado por el módem para entregar un mensaje de texto es:

```
<CR><LF>+CMT:
"+441234567890", "07/02/27,09:14:15+00"<CR><LF>Hello<CR><LF>
```

①

②

③

Llave:

1. La llamada a **Rx\_Message** con el delimitador ' ' (espacio) hace que la función establezca un valor en **Rx\_done** cuando se recibe el segmento +CMT: del mensaje.  
 +CMT: también debe estar disponible en la cadena **Rx\_Buffer** para confirmar la recepción de la mensaje correcta.  
 sabio.
2. Al llamar a **Rx\_Message** con el delimitador establecido en <CR>, se establece un valor en **Rx\_done** cuando se reciben los caracteres de identificación de llamada, fecha y hora. La información no es necesaria en este ejemplo y puede ignorarse.
3. Al llamar de nuevo a **Rx\_Message** con el delimitador establecido en <CR> se establece un valor en **Rx\_done** cuando se recibe la sección de texto del mensaje.  
 Está disponible en la cadena Rx\_Buffer para su comparación y/o visualización.

**Nota:**

Los caracteres delimitadores <CR> y <LF> nunca se incluyen en la memoria intermedia de recepción.

**10.2 Objetivos**

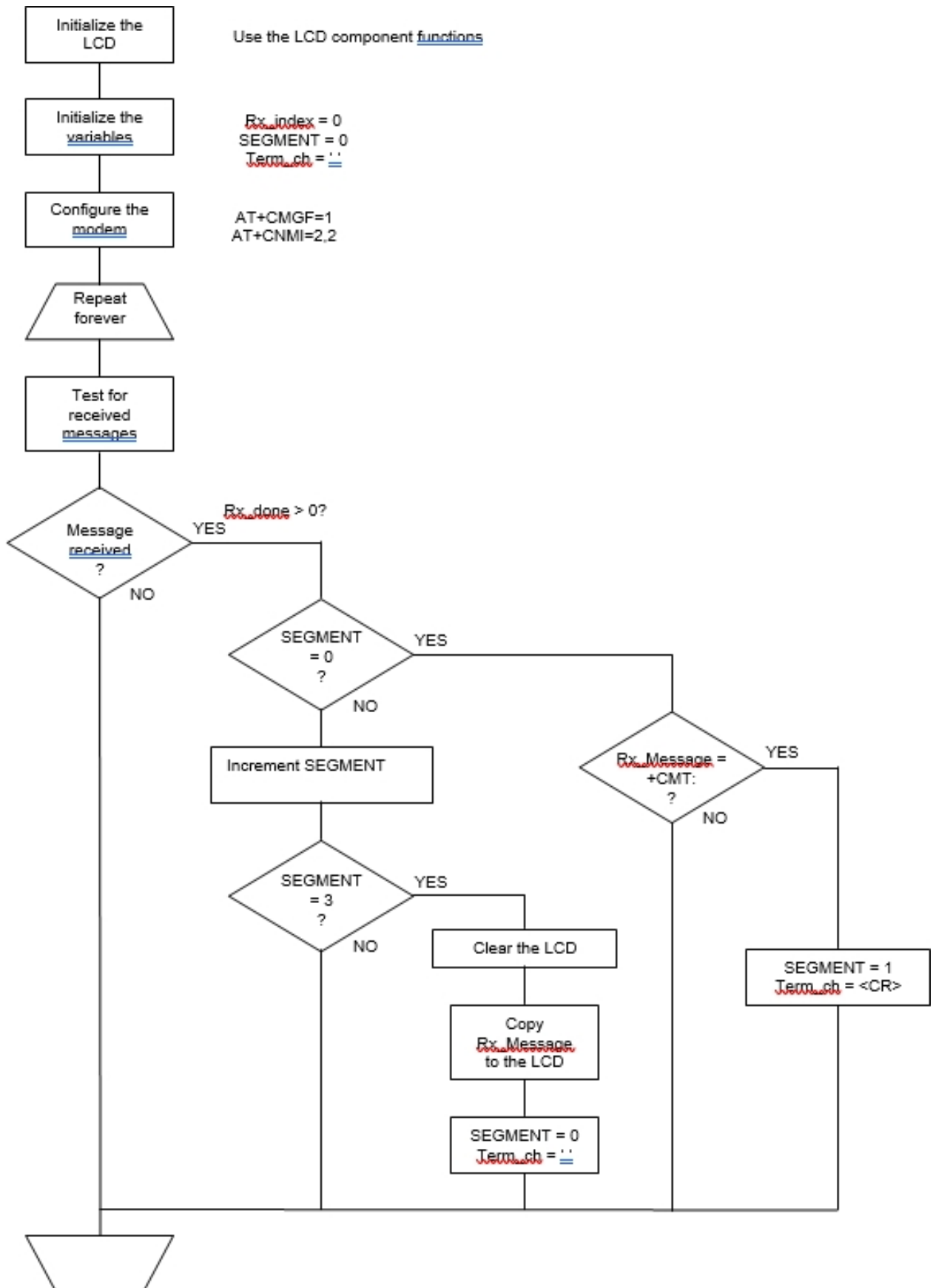
Este ejercicio crea un programa más complejo para recibir un mensaje de texto simple, extraerlo de la información de acción y mostrarlo en una pantalla.

**10.3 Requisitos**

- Una placa multiprogramador conectada a un PC que ejecuta Flowcode.
- Una placa E-blocks2 con teclado
- Una placa LCD E-blocks2
- Un módem GSM placa E-blocks2 con una tarjeta SIM activa (auriculares de audio no es necesario).
- Un teléfono móvil capaz de recibir mensajes de texto (el número de marcación de este teléfono debe estar en la cadena de comandos AT+CMGS del software Flowcode).

### 10.4 El programa Flowcode en detalle

El siguiente diagrama muestra las etapas del programa desarrollado en este ejercicio:



## 10.5 Qué hacer

Construya el programa para este ejercicio, utilizando como guía el diagrama de flujo anterior. Los pasos a seguir son:

- Inicializar y borrar la unidad LCD.
- Inicialice el módem para que funcione correctamente en modo texto.
- Comprueba si se ha recibido un SMS.
- Si es así, compruebe qué segmento de la cabecera se ha aislado.
- Cuando reciba el mensaje, visualícelo en el módulo LCD. Descárgalo en el microcontrolador y pruébalo enviando un mensaje de texto.

## 10.6 Más información en

El programa desarrollado en el ejercicio es capaz de identificar un flujo de informes de mensajes de texto entrantes, leerlo y mostrar el contenido del mensaje en la pantalla.

Desarrollar el software para que lea el contenido del mensaje y realice una serie de funciones en función del mensaje recibido:-.

- Activar y desactivar salidas
- Mostrar diferentes mensajes
- Lectura y visualización de los valores de entrada

### **Nota:**

Una solución a estas extensiones representa una aplicación práctica de control remoto. El desarrollo de un código de detección e interpretación de cadenas permitirá al sistema responder a los comandos y datos contenidos en el mensaje de texto recibido, incluida la capacidad de controlar los dispositivos conectados.

## 11. Ejercicio 8: Responder automáticamente a un mensaje de texto

Este ejercicio, el más complejo de todos, pone a prueba los requisitos de recepción y transmisión de mensajes de texto e ilustra su potencial en aplicaciones prácticas.

El ejercicio completado representa una aplicación de telemetría y control remoto que podría utilizarse en conexión con una amplia gama de sensores y actuadores adicionales, proporcionando un acceso global a las funciones proporcionadas.

### 11.1 Introducción

Los programas desarrollados en los dos ejercicios anteriores han demostrado individualmente la transmisión y recepción de mensajes de texto SMS. En este ejercicio se combinan los dos programas anteriores para crear una aplicación útil, capaz de leer un mensaje de texto entrante, y crear y enviar un mensaje de respuesta.

#### 11.1.1 Gestión de mensajes

Se puede desarrollar un programa de recepción de mensajes de texto que detecte la cabecera del mensaje (+CMT:) y extraiga tanto el identificador del llamante como el contenido del mensaje. El contenido del mensaje puede utilizarse para controlar una operación, y la información sobre el identificador de la persona que llama puede utilizarse para dar formato a un mensaje de respuesta.

#### 11.1.2 Mensaje descodificación

La macro RX\_Message puede utilizarse para identificar la cabecera del mensaje, extraer y guardar el identificador de llamada (incluidas las comillas) y extraer el contenido del mensaje.

Un ejemplo típico es:

```
+CMT: "+441234567890", "07/02/27,09:14:15+00"<CR><LF><message><CR><LF>
```

Si se reconoce el contenido del mensaje, se puede ejecutar una función de envío de mensajes, utilizando el número de identificación de llamada almacenado, para generar un mensaje de respuesta.

Utilizando la macro RX\_Message:

- Establezca el delimitador en ' ' para aislar la cabecera del mensaje - compruebe los resultados con "+CMT:".
- Establezca el delimitador en ',' para aislar el identificador de llamada - guarde los resultados en una cadena NÚMERO - incluyendo las marcas de voz.
- Ajuste el delimitador a <CR> para aislar los caracteres de hora y fecha - descarte los resultados.
- Deje el delimitador como <CR> para aislar el contenido del mensaje

Una vez finalizada la recepción del mensaje de texto, utilice la función Compare\$ para comparar el mensaje recibido en la cadena Rx\_Buffer con una cadena de muestra. Utilice "Estado" como cadena de muestra y realice la comparación con la coincidencia de mayúsculas desactivada.

Utiliza una coincidencia entre las dos cadenas para iniciar una secuencia de transmisión de mensajes.

#### 11.1.3 Respuesta transmisión

Se puede utilizar un código similar al del Ejercicio 6 para transmitir un mensaje de respuesta:

- No es necesario detectar una clave de transmisión, ya que el código sólo se ejecutará cuando se requiera la transmisión de un mensaje.
- El número que se va a marcar ya está almacenado, junto con las marcas de voz que lo rodean, en la cadena NUM- BER.
- El código de transmisión de mensajes debe devolver la operación al código de recepción de mensajes cuando el la transmisión se ha completado

Envía el mensaje "Listo" en respuesta al mensaje "Estado".

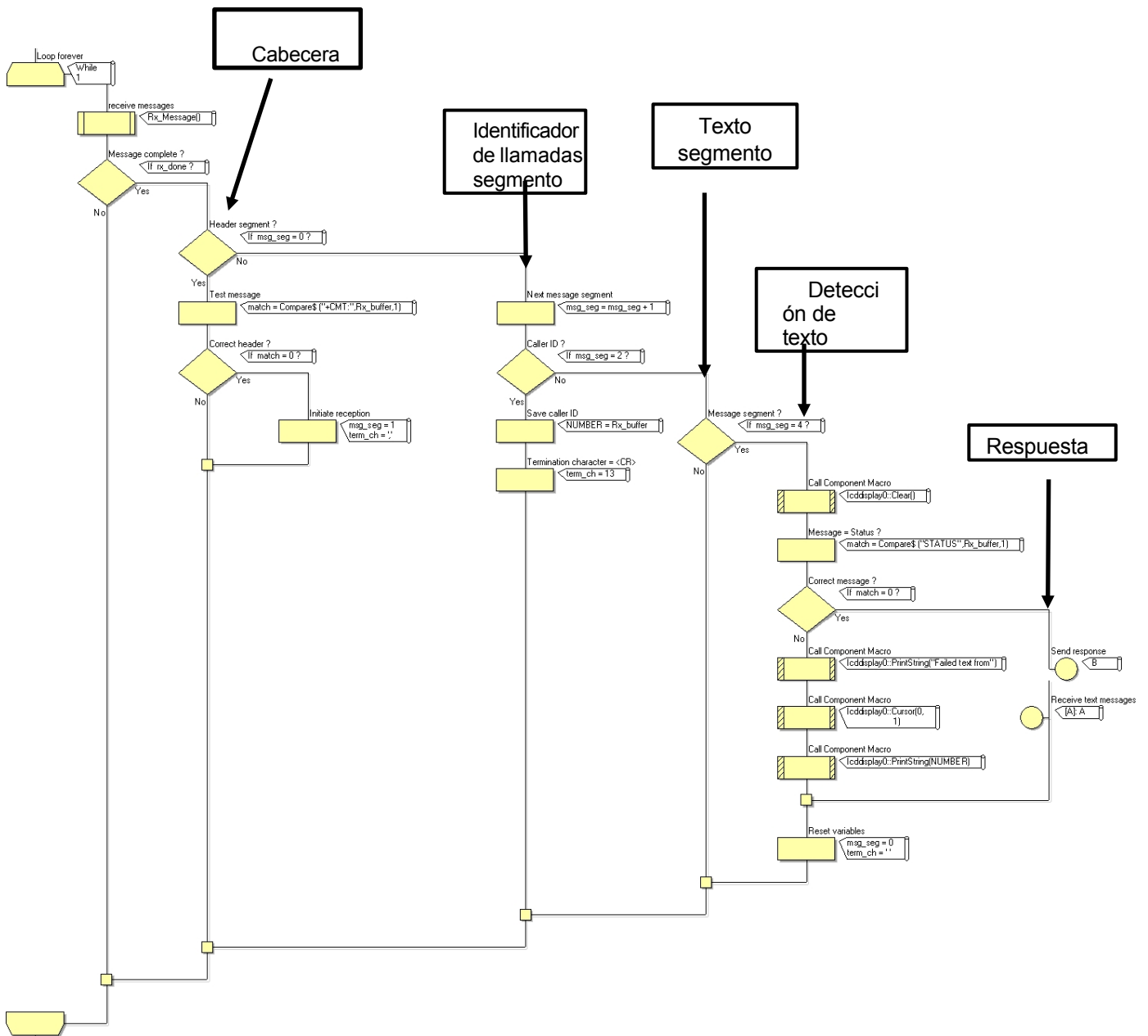
## 11.2 Objetivos

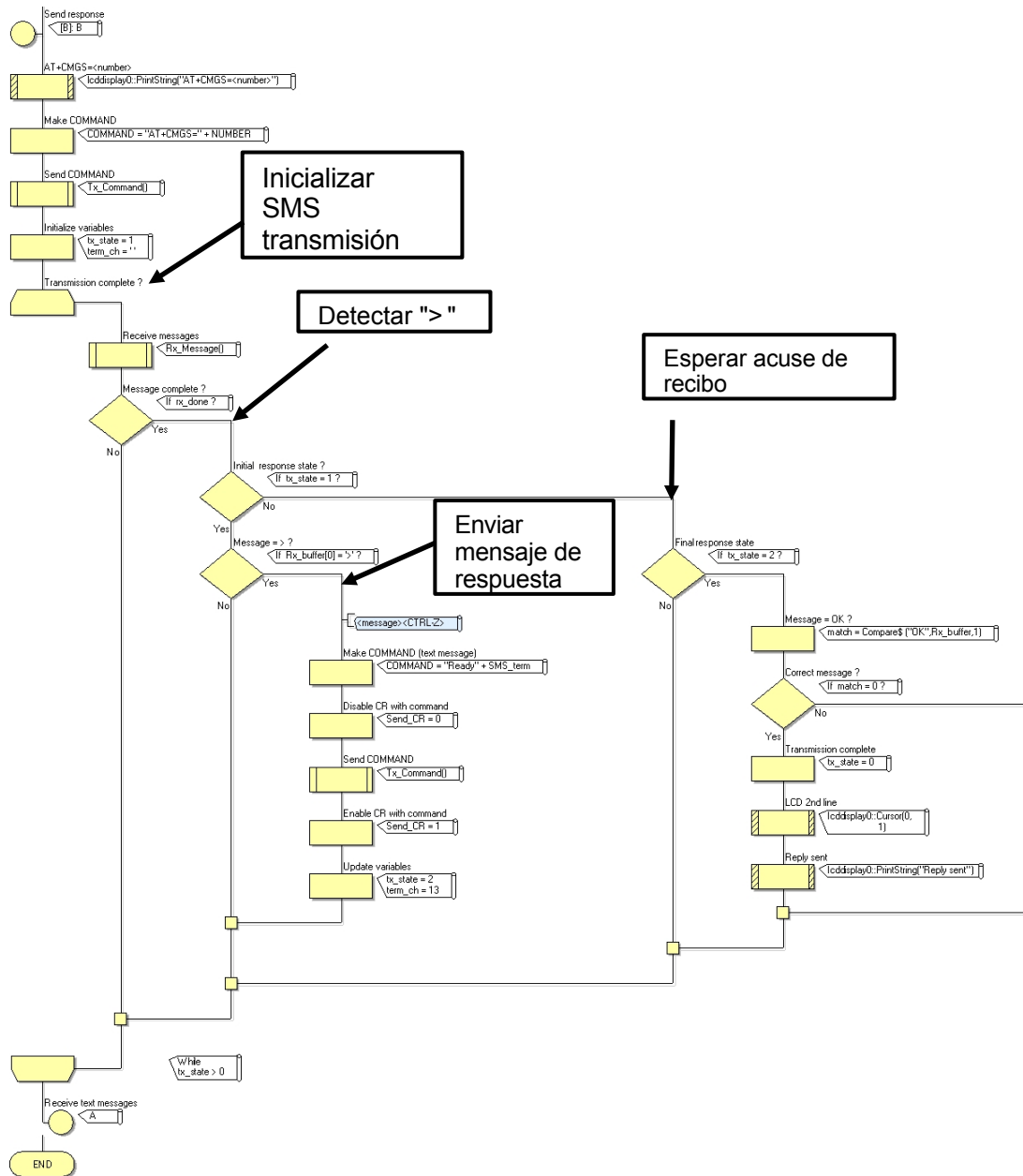
- Recibir un mensaje de texto entrante.
- Extraer e interpretar secciones del flujo de datos.
- Utilizar los datos del mensaje y del sistema para generar y transmitir un mensaje de respuesta.

## 11.3 Requisitos

- Una placa multiprogramador conectada a un PC que ejecuta Flowcode.
- Una placa E-blocks2 con teclado
- Una placa LCD E-blocks2
- Un módem GSM placa E-blocks2 con una tarjeta SIM activa (auriculares de audio no es necesario).
- Un teléfono móvil capaz de recibir mensajes de texto (el número de marcación de este teléfono debe introducirse en la cadena de comandos AT+CMGS en el software Flowcode).

## 11.4 El programa Flowcode en detalle





### 11.5 Qué hacer

Construye el programa que se muestra en los dos diagramas anteriores. Descárgalo en el microcontrolador de la forma habitual.

Pruébalo enviando un mensaje de texto al kit de teléfono móvil y comprueba si hay respuesta.

### 11.6 Más información en

- Añada seguridad al sistema cotejando el identificador de llamadas con una lista precargada de números aprobados. Reaccione sólo ante números reconocidos (¡y ahorre dinero!)
- Aumentar el número de mensajes que pueden identificarse y utilizarse para realizar funciones específicas.

- Desarrollar un sistema de alarma que pueda enviar un mensaje, con información específica, cuando se active. Permitir que la alarma se restablezca a distancia.
- Utilice la pantalla LCD para visualizar el progreso de la recepción y transmisión de mensajes.
- Asegúrese de que el sistema puede recuperarse de cualquier condición de error (respuestas inesperadas del módem, etc.).

**Nota:**

La solución a este ejercicio representa una aplicación completa de telemetría remota. Se pueden enviar comandos y datos al dispositivo remoto además de recuperar los datos. En este ejemplo trivial, el comando "Estado" simplemente activó la respuesta "Listo". Con el código de manipulación de cadenas apropiado, se puede utilizar una serie de comandos y datos para controlar la aplicación remota, así como para recuperar datos.

Una posible aplicación sería un sistema de alarma antirrobo que pudiera activarse, desactivarse y configurarse. La alarma enviaría un mensaje con los detalles de cualquier evento desencadenante (zonas activadas, etc.) y permitiría el restablecimiento a distancia.