TP N°9: SERIALIZACIÓN y COMUNICACIÓN SERIE

Ejercicio 9.1

Realizar una función que sea capaz de sacar el contenido de un byte por el puerto P0.1 bit a bit, cuya duración de bit sea la mínima que proporcione el algoritmo a realizar (no recurrir a demoras adicionales).

Ejercicio 9.2

Realizar una driver que sea capaz de sacar el contenido de un byte por el puerto P0.1 bit a bit, cuya duración de bit sea de 2,5 ms.

Ejercicio 9.3

Realizar una función que reciba un byte y sea capaz de serializar su contenido por el puerto P0.1, que por cada bit que saque ingrese el dato presente en el puerto P0.2, y por ultimo retorne el dato paralelizado. No recurrir a demoras adicionales entre la escritura y la lectura del puerto.

Ejercicio 9.4

Configure a la UART1 en 9600,8,N,1, y mediante la utilización de estrategias de atención de interrupciones, realizar un driver que despache en forma serializada y sincrónica el contenido de un buffer circular (vector de unsigned char de N elementos)

Ejercicio 9.5

Configure a la UART1 en 9600,8,N,1, y realice una función que reciba la dirección de comienzo de una string y transmita por polling uno a uno el contenido de sus bytes.

void Transmitir (char *mensaje);

Ejercicio 9.6

Configure a la UART1 en 9600,8,N,1. Luego, realice una función que reciba como argumento la dirección de comienzo de un buffer y almacene en este el contenido de los bytes recibidos vía serie sabiendo que el último dato de llegada será el **nul** y nunca serán más de diez bytes. Utilice técnicas de polling para su resolución.

void Recibir (char *mensaje);

Ejercicio 9.7

Configure a la UART1 en 9600,8,N,1, Y luego, realice las siguientes cuatro funciones, que formarán parte de un programa mayor que estará desarrollado utilizando técnicas de atención de interrupción para la transmisión y la recepción de datos.

```
01 #define     MAX_RX      20  //!< Tamaño del Buffer Circular de Recepción
02 #define     MAX_TX      20  //!< Tamaño del Buffer Circular de Transmisión
03
04 unsigned char BufferRecepcion[MAX_RX];      //!< Buffer Circular de Reception
05 unsigned char BufferTransmision[MAX_TX];    //!< Buffer Circular de Transmisión
06 unsigned char iTin;      //!< Indice de ingreso de datos en el Buffer Circular de Transmisión
07 unsigned char iTout;     //!< Indice de salida de datos en el Buffer Circular de Transmisión
08 unsigned char iRin;      //!< Indice de ingreso de datos en el Buffer Circular de Recepción
09 unsigned char iRout;     //!< Indice de salida de datos en el Buffer Circular de Recepción
10
11 /**
12  *  \fn void PushTransmitir (unsigned char )
13  *  \brief Coloca un byte en la cola circular BufferTransmision
14  *  \param [ in ] dato a guardar
15  *  \return void
16  */
17 void PushTransmitir ( unsigned char );
18
19 /**
20  *  \fn int PopTransmitir (void )
21  *  \brief Toma un byte de la cola circular BufferTransmision
22  *  \param void
23  *  \return dato si la cola tiene elementos y -1 si esta vacía
24  */
25 int PopTransmitir ( void );
26
27 /**
28  *  \fn void PushRecibir (unsigned char )
29  *  \brief Coloca un byte en la cola circular BufferRecepcion
30  *  \param [ in ] dato a guardar
31  *  \return void
32  */
33 void PushRecibir ( unsigned char );
34
35 /**
36  *  \fn int PopRecibir (void )
37  *  \brief Toma un byte de la cola circular BufferRecepcion
38  *  \param void
39  *  \return dato si la cola tiene elementos y -1 si esta vacía
40  */
41 int PopRecibir ( void );
```

Ejercicio 9.8

Realice una función primitiva que reciba como argumento la dirección de comienzo de una string y se encargue de cargarla en el buffer circular de transmisión mencionado en el Ejercicio 9.7. La función deberá considerar la situación de que el buffer este vacío y, en ese caso, **iniciar** la transmisión.

```
43 /**
44  *  \fn void Transmitir ( char * )
45  *  \brief Carga una string en la cola circular BufferTransmision
46  *  \param [ in ] direccion de la string
47  *  \return void
48  */
49 void Transmitir ( char * );
```

Ejercicio 9.9

Teniendo en cuenta las funciones realizadas en el ejercicio 9.7 realice una función que sea capaz de colocar en un buffer (cuya dirección le haya sido enviada como argumento), los bytes considerados como datos recibidos (válidos) de la trama que ilustra la siguiente figura.

'#'	D2	D1	D2	'$'

'#': Comienzo de trama
D2,D1 y D0: Datos. Dígitos de un número de tres cifras representado en formato **ASCII**
'$': Fin de trama

Para la confección de la función tenga en cuenta las siguientes consideraciones:

- No debe ser bloqueante. Es decir no se debe recurrir a lazos cerrados que se apropien de la ejecución del programa.

- Si el buffer esta completo retorna 0 y si aun no se completo retorna -1

```
51 /**
52  *  \fn void Recibir ( char * )
53  *  \brief Representación del dato recibido
54  *  \param [ in ] dirección de la string donde cargar los datos
55  *  \return 0 Buffer lleno o -1 No se completo el buffer
56  */
57 int Recibir ( char * );
```

Ejercicio 9.10

Valiéndose de las funciones realizadas en el ejercicio 9.7 realice un programa que genere un "eco" de los caracteres ingresados por el puerto serie (UART1 en 9600,8,N,1)

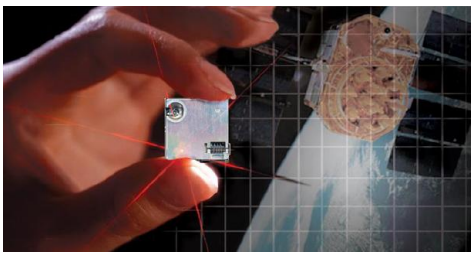
Ejercicio 9.11

Valiéndose de las funciones realizadas en el Ejercicio 9.7 realice un programa que genere el "eco inteligente" de las letras ingresadas por el puerto serie, (UART1 en 9600,8,N,1).
El programa debe seguir la siguiente lógica: A partir de que recibe el carácter '0', el "eco" devuelve la letra ingresada sin modificaciones; si se ingresa el carácter '1' el "eco" devuelve la letra en minúscula independientemente de si haya sido ingresada en minúscula o mayúscula; por último si se ingresa el carácter '2' el "eco devuelve la letra en mayúscula.

NOTA: Se deberá chequear la paridad para cada dato recibido y en caso de detectar error en vez de transmitir el dato recibido se deberá enviar el código **0xFF**

Módulos GPS

- *El módulo receptor GPS, por sus siglas en Inglés de Sistema de Posicionamiento Global, es un dispositivo capaz de recibir información de distintos satélites y procesarla con algoritmos matemáticos, con el fin de brindarnos información de latitud, longitud, velocidad de desplazamiento y altura entre otras.*
- *Su principal aplicación es en sistemas de navegación, pero su utilización se ha popularizado, y hoy podemos encontrar un GPS en una amplia gama de celulares y cámaras de fotos.*
- *La precisión del GPS está dada por distintos factores y se debe tener en cuenta según la aplicación. Por ejemplo, para ciertas aplicaciones 10 metros de error en la posición y 0,1metros por segundo en la velocidad es totalmente aceptable.*
- *Muchos módulos disponen de un puerto serie para comunicar los datos al mundo exterior a través de tramas series (por ejemplo a 4800 baudios 8,N,1) que ya están estandarizadas.*
- *Una estrategia puede ser recibir una trama completa y luego procesarla para obtener el dato que necesito.*
- *Dispone de diferentes tramas no todas dan los mismos datos. Las tramas tienen un encabezado para que al procesar la misma se pueda evaluar que tipo de trama es, por ejemplo, GGA, VTG, RMC, etc…*
- *Según la aplicación, se los pueden pre-configurar para que ofrezcan una o mas tramas. Ejemplo, no es lo mismo navegación terrestre que marítima.*



Ejercicio 9.12

El GPS, según su configuración, puede enviar distinto tipos de trama. Para obtener la velocidad será necesario reconocer únicamente la trama **"Track Made Good and Ground Speed"** (VTG) y de ella extraer la velocidad en km/h.

Trama VTG: *$GPVTG,x.x,T,x.x,M,x.x,N,x.x,K*hh*

Campo	Descripción de Campos
$	Comienzo de trama
GPVTG	Indicador de trama
,	Carácter coma, separador de campos
x.x	Ángulo Azimuth del receptor, campo variable
,	Carácter coma, separador de campos
T	Carácter T, campo fijo
,	Carácter coma, separador de campos
x.x	Variación Magnética, campo variable
,	Carácter coma, separador de campos
M	Carácter M, campo fijo
,	Carácter coma, separador de campos
x.x	Velocidad expresada en Nudos, campo variable
,	Carácter coma, separador de campos
N	Carácter N, campo fijo
,	Carácter coma, separador de campos
x.x	Velocidad expresada en Km/h , campo variable
,	Carácter coma, separador de campos
K	Carácter K, campo fijo
*	Carácter asterisco, indicador de comienzo de checksum
hh	Checksum, se calcula realizando la XOR de todos los caracteres excepto del $ y del *
‘\n’	Fin de trama

Se pide implementar un dispositivo que obtenga la velocidad de un sensor GPS

Ejemplo de trama recibida
$GPVTG,,T,,M,37.9,N,**70.2**,K,A*2B

Analizando la trama anterior, vemos que la velocidad de desplazamiento es de **70,2** km/hora, que es el dato que aparece después de la séptima coma.

Ejercicio 9.13

Valiéndose de las funciones realizadas en los ejercicios anteriores y de la posibilidad que brinda el UART1 de trabajar en modo 485, se pretende realizar un sistema que realice la gestión del cobro de combustible en una estación de servicio.
Para llevar a cavo la tarea, se deberá tener en cuenta que la comunicación que se establece entre la caja y los surtidores es del tipo amo esclavo. La caja (el amo) enviará una trama con la identificación del equipo con el cual desea establecer la comunicación, el surtidor seleccionado (el esclavo) contestará mandando la información solicitada por la caja (el amo), y así el amo ira encuestando a todos los surtidores para que les envíe la información que necesita para llevar adelante el sistema de gestión.
Se pide:

a) SURTIDOR
   a. Drivers, primitivas y aplicación de la parte del software involucrada en la comunicación del equipo instalado en los surtidores.
   b. Cada vez que la caja solicite información se deberá transmitir el contenido de las variables ***litros*** y ***Tipo***.

b) CAJA
   a. Drivers, primitivas y aplicación de la parte del software involucrada en la comunicación del equipo instalado en la caja.
   b. El programa deberá realizar la encuesta de todos los surtidores cada 2 minutos.
   c. Calcular y totalizar los litros consumidos por tipo de combustible

Tipo	Descripción
1	Nafta común
2	Nafta Super
3	Nafta Premium
4	Gasoil común
5	Gasoil Premium

   d. Presentación cíclica cada cinco segundos de los totales del punto anterior en un display de 7 segmentos dividido en dos zonas: la primera de un digito para el ID y la otra de 5 dígitos para los totales.
   e. Si en la trama recibida el valor del ***Tipo*** es cero, significa que hay una carga en curso, por lo tanto descartar el esa oportunidad el valor del campo ***litros.***

Trama de encuesta enviada por la caja

Numero de equipo	1

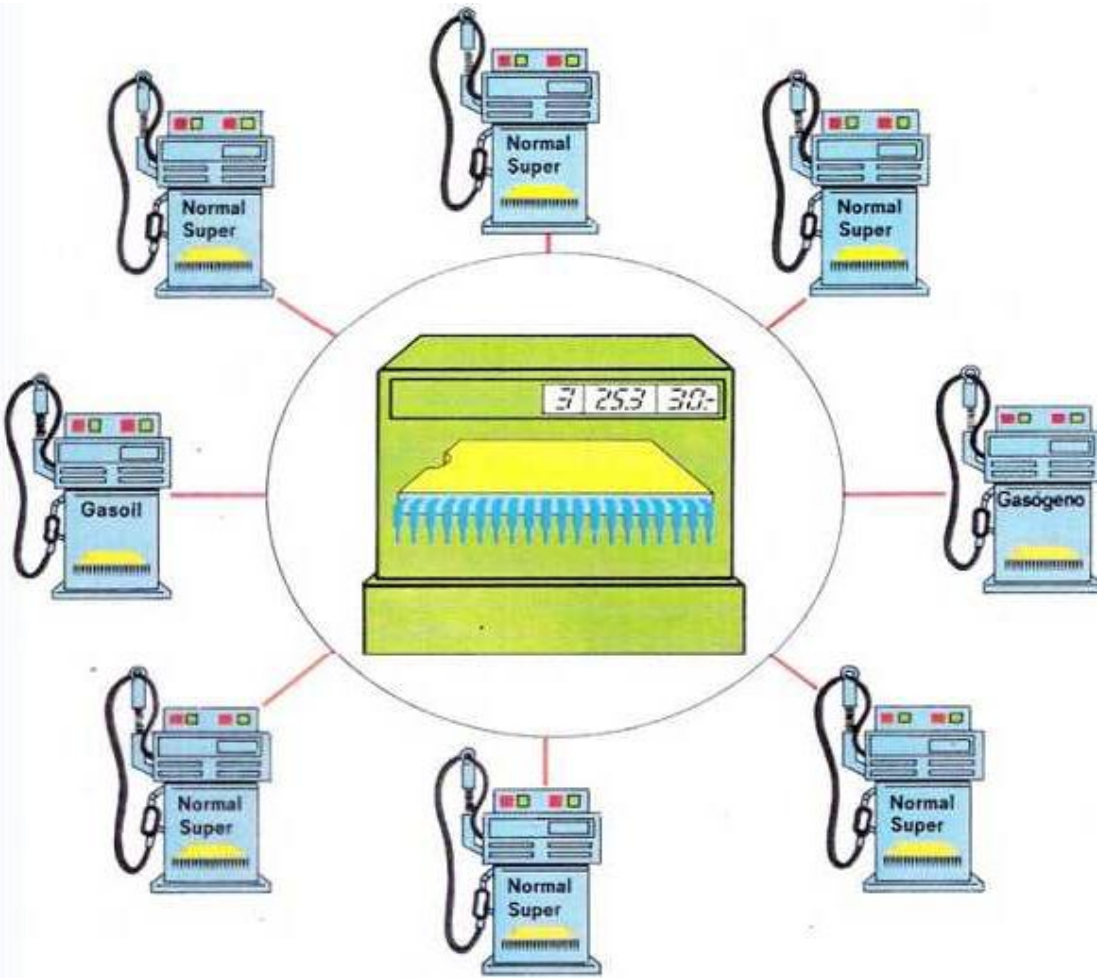
Noveno bit

Numero de equipo: valor binario de rango 0 a 7

Trama de respuesta del surtidor

Tipo	**0**	**litros$_3$**	**0**	**litros$_2$**	**0**	**litros$_1$**	**0**	**litros$_0$**	**0**

Tipo : valor binario de rango 0 a 5
litros$_n$: Valor binario componente un dato entero de 32 bits


Normal Super
Normal Super
Normal Super
Gasoil
Gasógeno
Normal Super
Normal Super
Normal Super