Práctica 7

Programación del uC del periférico de comunicación serie utilizando interrupciones.

Objetivo: Mediante esta práctica el alumno aprenderá el uso básico para inicializar y operar, bajo un esquema de interrupciones, el puerto serie del microcontrolador.

Equipo:
- Computadora Personal
- Módulo T-Juino

Teoría:
- Manejo de las interrupciones del Periférico de Comunicación Serie (UART) del microcontrolador ATmega1280/2560.
- Secuencias de escape ANSI.

Actividades a realizar:

1. **Bajo los esquemas de la Figura 1 y Figura 2, implemente el manejador para recibir y enviar datos vía puerto serie 0 (UART0) utilizando interrupciones.**

Transmisión y Recepción de datos bajo el uso de interrupciones. Para hacer uso eficiente de la transmisión y recepción de datos bajo el esquema de interrupciones es necesario una cola circular para transmitir, y otra para recibir.

En la cola de transmisión se introducen los datos a transmitir para que después la rutina de servicio de interrupción (ISR) correspondiente los tome al momento adecuado para su transmisión. En la Figura 1 se observa que la función ***UART_putchar( )*** es la que introduce cada dato a la cola circular **siempre y cuando exista lugar para hacerlo**; de lo contrario tendrá que esperar a que se libere un lugar.

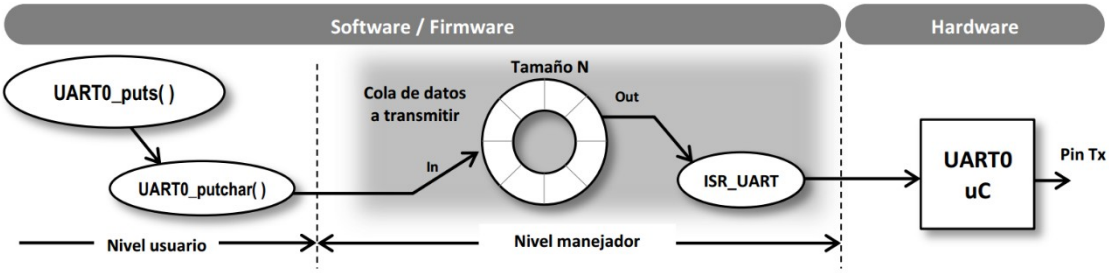

Figura 1. Esquema para el manejo de paquetes de transmisión mediante cola circular.

Listado 1. Tipo de dato **ring_buffer_t**

```
#define BUFFER_SIZE   64

typedef struct {
  char buffer[BUFFER_SIZE];          /* espacio reservado          */
  volatile unsigned char in_idx;     /* indice entrada (Head)      */
  volatile unsigned char out_idx;    /* indice entrada (tail)      */
} ring_buffer_t;
```

La ISR correspondiente a la transmisión del UART0 es la que tiene la responsabilidad de tomar dato por dato de la cola circular e introducirlos al UART0, esto a cada momento que el registro UDR0 (de transmisión) esté libre para poder escribir en dicho registro. Si por algún motivo no

existen datos en la cola circular la ISR debería de deshabilitar su interrupción correspondiente para evitar que continuamente se invoque la interrupción.

Considerando lo anterior, implementar la siguiente rutina:

a) **ISR(SIG_USARTx_DATA)** : Rutina de servicio de interrupción para el evento de **registro de transmisión vació**. Esta rutina **extrae de la cola circular** el dato que será transmitido por el UARTx.

Y modificar:

b) `void` **`UART_putchar`**`(uint8_t com, char data)`: Función que coloca el dato en la cola circular.

En la cola de recepción, la ISR es quien introduce los datos recibidos. En la Figura 2 se observa que UART_getchar( ) es quien extrae los datos de la cola. Implemente el manejador para recibir datos vía puerto serie 0 (UART0). Considera que se requiere la implementación de un mecanismo de recepción almacenar los datos en la cola circular por lo que se requieren las siguientes funciones.

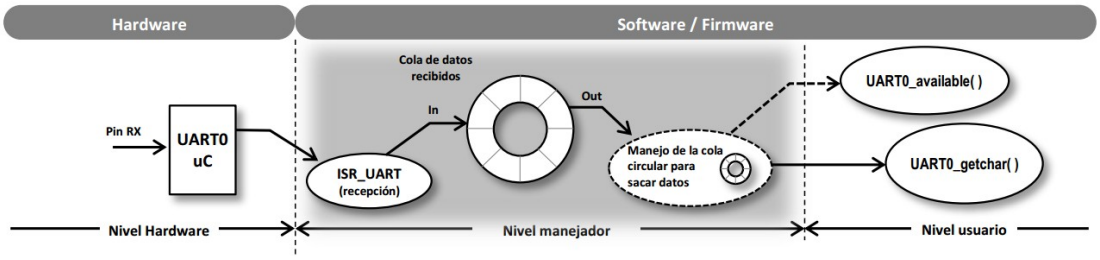

Figura 2. Esquema para el manejo de paquetes de recepción mediante cola circular.

Bajo el esquema de la Figura 2 implemente el manejador para recibir datos vía puerto serie 0 (UART0). Considera que se requiere la implementación de un mecanismo de recepción almacenar los datos en la cola circular por lo que se requieren las siguientes funciones:

c) ISR(SIG_USART*x*_RECV): Rutina de servicio de interrupción para el evento de **recepción completa**. Esta rutina **inserta a la cola circular** el dato que fue recibido por el UARTx.

d) uint8_t **UART_available**(`uint8_t com`): Función que retorna 1 si existe(n) dato(s) en la cola circular.

e) `char` **`UART_getchar`**`( uint8_t com )`: Función toma el **dato correspondiente a salir de la cola circular**. Esto si existe alguno dato en la cola circular, de lo contrario espera a que exista uno para tomarlo y retornarlo.

2. Reutilizar práctica anterior, migrando la implementacion del UART bajo las interrupciones de transmisión y recepción.

Comentarios y Conclusiones.

Bibliografía.