CURSO DE STM32

25 ENERO 2021

004 Comunicacion Serial Polling IT y DMA

Creado por: Ing. Christian Salazar

SECCION 4

004 Comunicacion Serial Polling IT y DMA

Que aprenderemos?

En este video vamos a aprender a manipular el periférico USART (huart1) para transmitir y recibir datos en modo polling, por interrupción y por DMA desde una terminal serial (Software libre RealTerm) con la ayuda de un conversor de USB a serial TTL (ch340 converter), además configuraremos el USART para poder usar la función printf() de librería stdio.h.

> ***"Usaremos HAL Drivers, lo cual nos ayudará en gran medida a portar y reciclar rutinas de código de un procesador de una Familia a otro de otra Familia".***

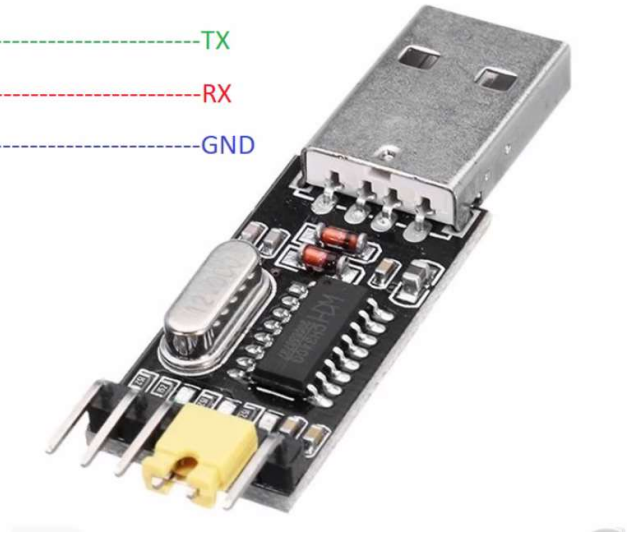
Puntos Clave

CONEXIÓN

PA10(RX)----------TX

PA9 (TX)----------RX

GND----------GND



LIBRERÍA STDIO Y FUNCION PRINTF

```
/* Includes ------------------------------------------------------------------*/
#include "main.h"

/* Private includes ----------------------------------------------------------*/
/* USER CODE BEGIN Includes */
// For to use pcintf function
#include "stdio.h"
/* USER CODE END Includes */

/* Private typedef -----------------------------------------------------------*/
/* USER CODE BEGIN PTD */

/* USER CODE END PTD */

/* Private define ------------------------------------------------------------*/
/* USER CODE BEGIN PD */

/* USER CODE END PD */

/* Private macro -------------------------------------------------------------*/
/* USER CODE BEGIN PM */

/* USER CODE END PM */

/* Private variables ---------------------------------------------------------*/
UART_HandleTypeDef huart1;

/* USER CODE BEGIN PV */
int __io_putchar(int ch) {
HAL_UART_Transmit(&huart1, (uint8_t *)&ch, 1, 100);
return ch;
}
/* USER CODE END PV */
```

Como debemos saber la función printf de la librería stdio.h es muy útil para transmitir datos ASCII desde cualquier parte de nuestro

código, esta función por lo general es muy usada para depurar código donde se maneja tramas en ASCII, para ello incluiremos la linea 27 que nos compilará las funciones disponibles de ANCI C para esa biblioteca, asimismo pondremos la función de imprimir un dato através de USART 1 (linea 50) la cual será usada al llamar a printf().

MODO POLLING

```
HAL_StatusTypeDef HAL_UART_Receive(UART_HandleTypeDef *huart, uint8_t *pData, uint16_t Size, uint32_t Timeout)
HAL_StatusTypeDef HAL_UART_Transmit(UART_HandleTypeDef *huart, uint8_t *pData, uint16_t Size, uint32_t Timeout)
```

Para transmitir o recibir datos en Modo Polling tenemos dos funciones, las cuales necesitan del manejador del USART que se usaá para comunicar en el caso del video es el huart1, asimismo el puntero a la información a enviar o a su vez el puntero de la variable donde se va a recibir la trama de datos, el tamaño de la trama a transmitir o recibir y finalmente un timeout en milisegundos, que es el tiempo máximo que se debe esperar a que el periferico termine su trabajo cuando el IDLE nos indique que ha iniciado un proceso el USART, normalmente esta manera de transmitir se usa en el while infinito, para la recepción se lo usa para tramas fijas.

MODO POR INTERRUPCION

```
HAL_StatusTypeDef HAL_UART_Receive_IT(UART_HandleTypeDef *huart, uint8_t *pData, uint16_t Size)
HAL_StatusTypeDef HAL_UART_Transmit_IT(UART_HandleTypeDef *huart, uint8_t *pData, uint16_t Size)
```

El modo por interrupción tenemos igual dos funciones básicas, las que se parece a las dos anteriores solo que sin el timeout ya que ahora tenemos la opción de no consultar el estado del periférico (SI hay que recibir o transmitir tramas) en el while infinito sino que con estas funciones cuando haya algo que recibir se levantará la interrupción del USART1 en nuestro caso y es entonces que como en

las interrupciones EXTI antes estudiadas tenemos una función de Callback que se ejecuta inmediatamente después de atender al interrupción del periférico, esta función es llamada por todos los periféricos USART que tengan habilitada la interrupcón por lo cual recibe como argumento la Instancia del USART que la activó o llamó y podemos diferenciarlos basicamente de igual manera con un bucle if(), entonces estas funciones configura y ordena al periférico USART para que una vez que sea el caso de recibir un dato y este dato se encuentre en el registro buffer del USART1 nos informara que hay información disponible, es entonces en este instante que podemos descargar la información en nuestro propio buffer para despues usar esa información recibida, si no se descarga la información al recibir, esta se sobreescribirá por la siguiente trama de información que pueda llegar, asimismo para el caso de la transmisión podemos llamar a la función en cualquier instante en nuestro programa y se levantará una interrupción en el momento que se haya terminado de transmitir el último byte de la trama que se le haya ordenado enviar.

MODO DMA

```
HAL_StatusTypeDef HAL_UART_Receive_DMA(UART_HandleTypeDef *huart, uint8_t *pData, uint16_t Size)
HAL_StatusTypeDef HAL_UART_Transmit_DMA(UART_HandleTypeDef *huart, uint8_t *pData, uint16_t Size)
```

El periférico DMA (Acceso Directo a Memoria) al igual que el CPU es un periférico de tipo Master, es decir que puede leer y escribir información desde un periférico (TIMER, USART, I2C, etc) a memoria, de memoria a un periférico, de memoria a memoria y de periférico a periférico, en el caso del USART usamos la transmisión por DMA para enviar información desde memoria al periférico y para recibir se lo hace desde periférico a memoria, la sintaxis es parecida a las dos

funciones anteriores, solo cambia el nombre de la función, estas funciones también usan la misma función Callback que las dos anteriores además de que pueden levantar la interrupción del periférico DMA, es decir que al termianr de enviar o recibir datos podemos tomar decisiones al finalizar dichos procesos, el uso del DMA libera de carga o procesamiento al CPU principal ya que este proceso de transportar la información cualquiera que sea el modo lo hara el periférico DMA.

RECOMENDACION

Se recomienda usar para la recepcion, la interrupción de un Byte a la vez y la transmisión por DMA de la trama completa de transmisión, es muy práctico para el protocolo ModBus por ejemplo, se puede usar todo por DMA y la recepción hacerlo solo de Byte en Byte.