Configuración de GPIO en STM32CubeIDE

En este manual se mostrará como configurar el USART de la tarjeta de desarrollo STM32F401RE para hacer comunicación serial, con un ejemplo donde se enviarán datos, y dependiendo del dato recibido se encenderá o apagará un LED.

Para ello se usará STM32CubeIDE para generar el código y depurar creando un nuevo proyecto.

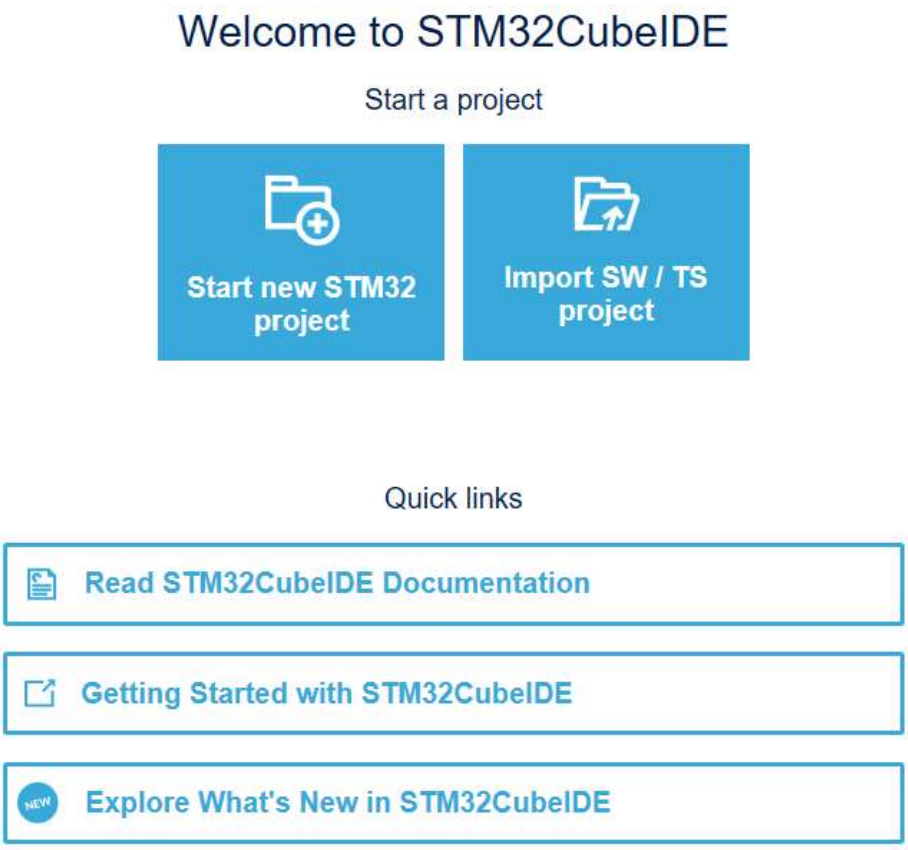

Figura 1: Página de bienvenida de STM32CubeIDE

Después se deberá buscar la tarjeta a programar, se da clic a siguiente, y se asigna un nombre al proyecto, después de lo cual se le preguntará si se desea iniciar los periféricos en su modo predeterminado, a lo que es recomendable decir que sí para que automáticamente configure los periféricos embebidos en la tarjeta, preguntará también si se quiere pasar a la vista de CubeMX, a lo cual también se dirá que si.

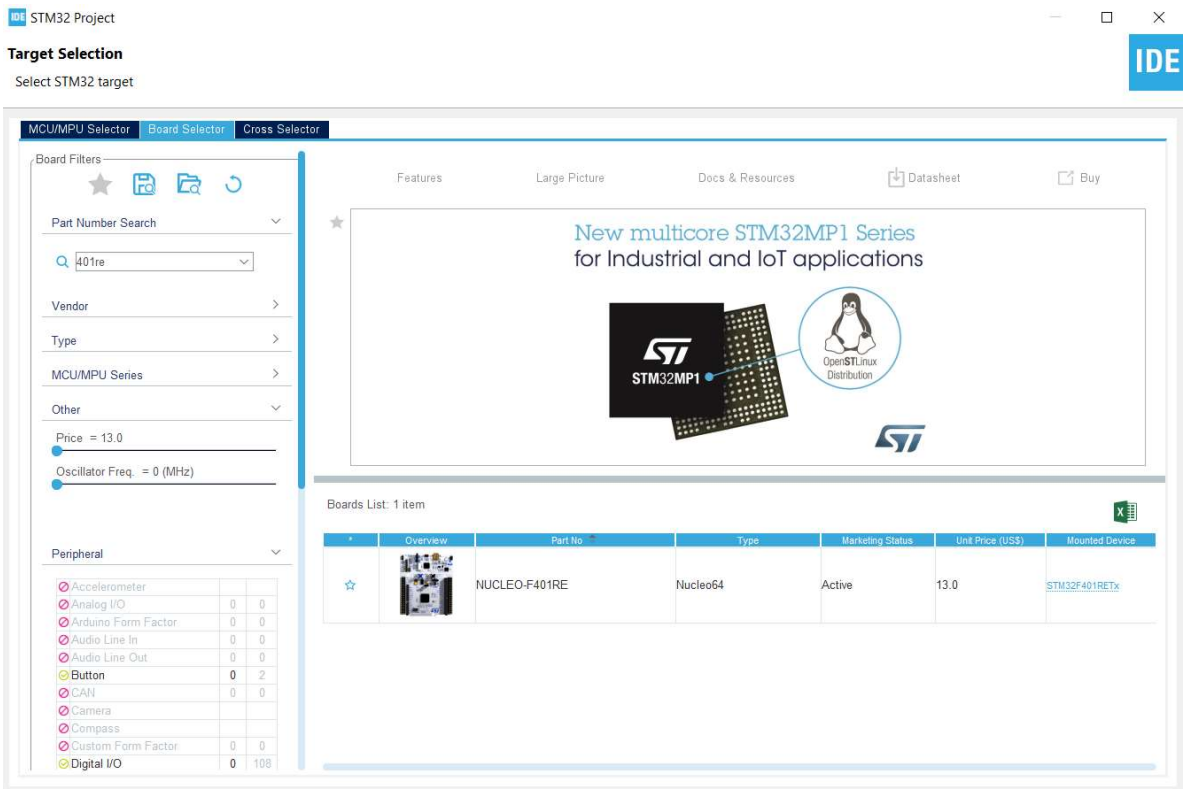

Figura 2: Selector de tarjeta

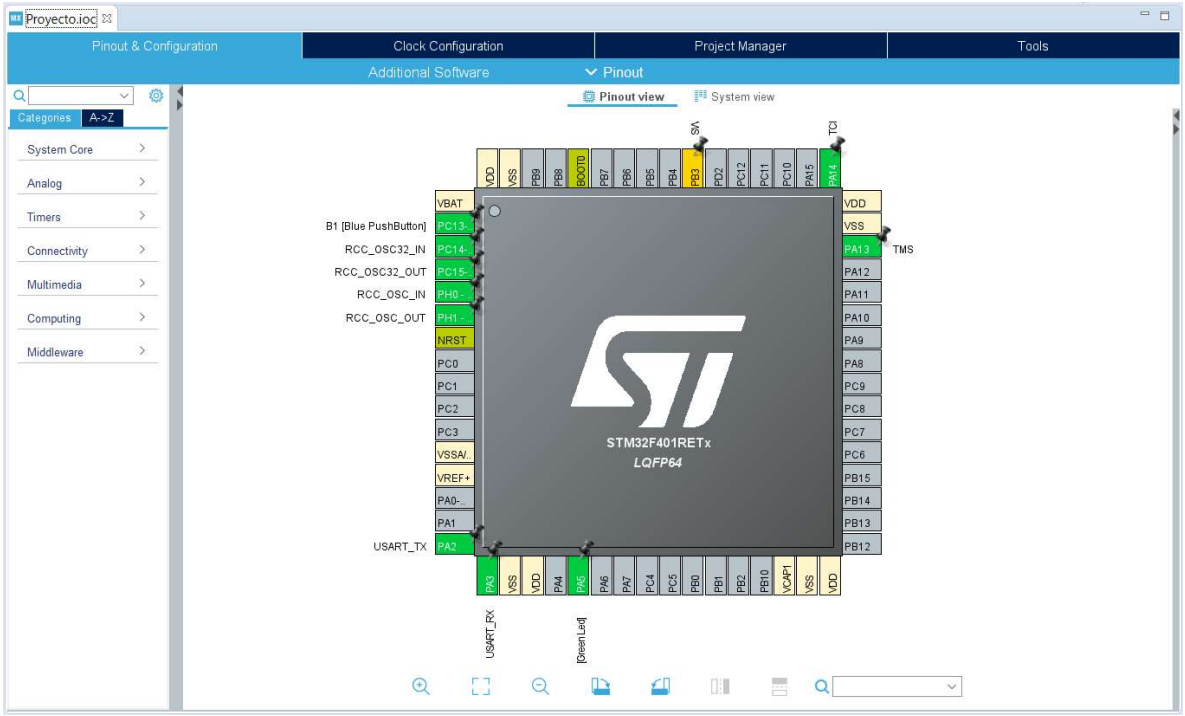

Figura 3: vista de CubeMX, herramienta de configuración del dispositivo

Ahora hay que acceder a la configuración del USART2, en la pestaña de “connectivity”, seleccionar USART2, seleccionar modo asíncrono y configurarlo como se muestra a continuación.

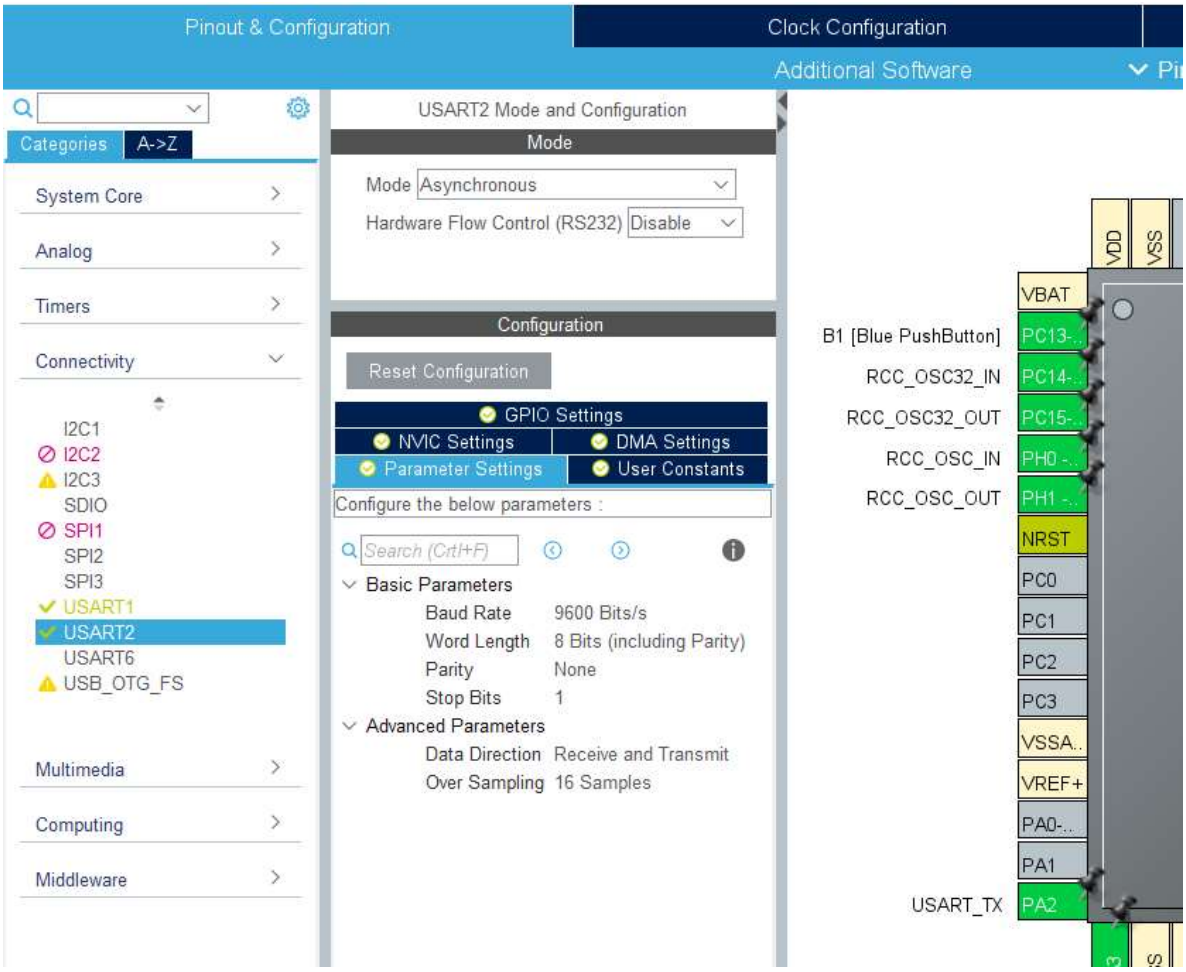

Figura 4: configuración del USART2

Después de cerciorarse que los parámetros estén iguales a la figura 4, hay que ir a Project manager -> Code Generator, y activar la casilla “*Generate peripherial initialization as a pairo f ‘.c/.h’ files per peripherial”*, y posteriormente se genera el código.

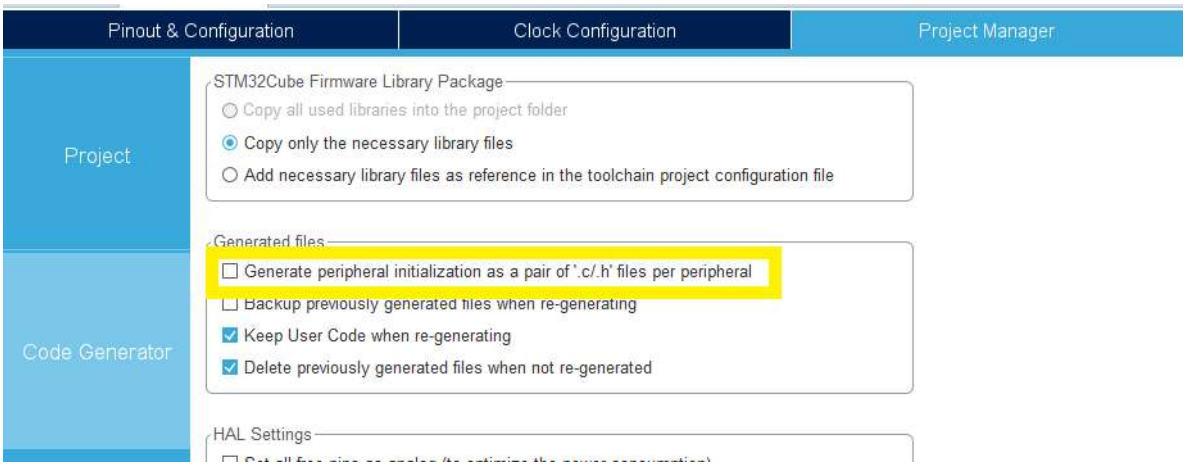

Figura 5: casilla para inicializaciones.

Ahora, ya que están configurados los puertos, se hace clic en Project y luego en Generate Code.

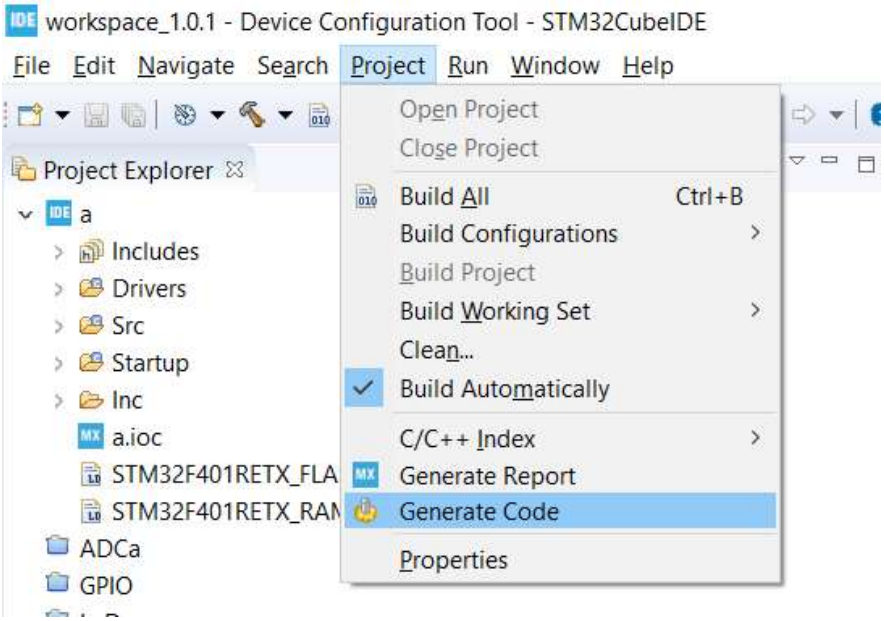

Figura 6: Generate Code.

Para modificar el código, hay que cambiar de perspectiva dando clic en la parte superior derecha de la pantalla en el botón de C/C++, y abriendo el main.c haciendo clic en src y luego doble clic en el archivo.

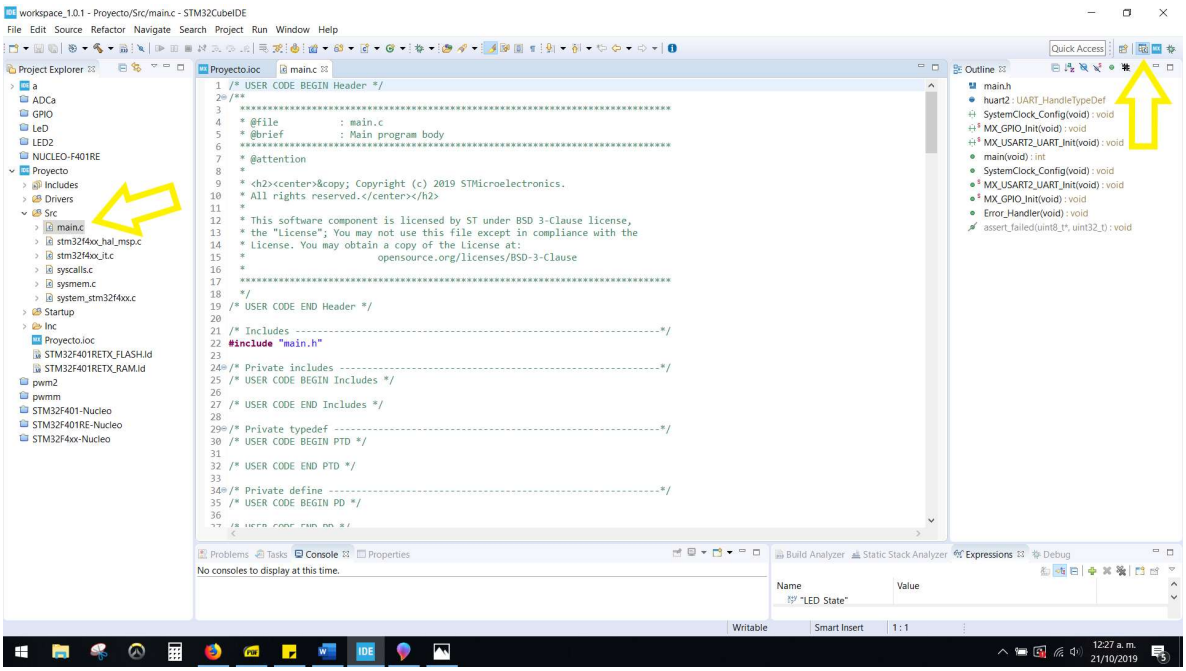

Figura 7: Vista en C/C++

Para comprobar la funcionalidad del USART, se enviará la leyenda ‘Hola Mundo’, a la vez que se recibe un carácter como una condición, la cual recibe si es una ‘A’, para encender el led, o una ‘E’, para apagarlo.

Ahora primero se debe ir alrededor de la línea 27 e incluir las librerías necesarias, así como crear las variables necesarias.

```
27 /* USER CODE BEGIN Includes */
28 #include <stdio.h>
29 #include <string.h>
30 /* USER CODE END Includes */
```

Figura 7: librerías necesarias

```
68 int main(void)
69 {
70   /* USER CODE BEGIN 1 */
71     uint8_t DatoRecibido[1];
72     char bufer[30];
73   /* USER CODE END 1 */
74
```

Figura 8: Variable y bufer

Ahora se procede a desarrollar el código del while infinito, para que siempre esté ejecutándose.

```
102   while (1)
103   {
104     /* USER CODE END WHILE */
105
106     /* USER CODE BEGIN 3 */
107       HAL_UART_Receive(&huart2, DatoRecibido,(uint16_t)1,(uint32_t)100);
108       if(DatoRecibido[0]=='E'){
109           HAL_GPIO_WritePin(LD2_GPIO_Port, LD2_Pin, GPIO_PIN_SET);
110       }else if(DatoRecibido[0]!='A'){
111           HAL_GPIO_WritePin(LD2_GPIO_Port, LD2_Pin, GPIO_PIN_RESET);
112       }
113       sprintf(bufer, "Hola mundo \n");
114      HAL_UART_Transmit(&huart2, (uint8_t*)bufer, (uint16_t)strlen(bufer), (uint32_t)100);
115   }
116   /* USER CODE END 3 */
```

Figura 9: Código para enviar y recibir.

La función *HAL_UART_Receive(&huart3, DatoRecibido,(uint16_t)1,(uint32_t)100)* se encarga de recuperar los datos que provienen del puerto serial y los guarda en una variable o arreglo, en este caso DatoRecibido. Los parámetros usados se explicarán a continuación:

- **huart2**: puntero a la estructura UART_HandleTypeDef que contiene la información de configuración para el módulo UART (puerto utilizado) especificado.
- **DatoRecibido:** variable o apuntador que contendrá los datos provenientes del puerto, estos tienen que ser de tipo uint8_t.
- **1**: Cantidad de datos a recibir de tipo uint16_t.
- **100**: tiempo de espera (ms), este dato tiene que se de tipo uint32_t.

La función *HAL_UART_Transmit(&huart3, (uint8_t*)bufer, (unit16_t)strlen(bufer), (uint32_t)100)* se encarga de enviar los datos provenientes de una variable o arreglo, hacia el puerto serial, en este caso en particular los datos enviados se encuentran en un arreglo llamado **bufer**. Sus parámetros se presentan a continuación:

- **&huart2**: puntero a la estructura UART_HandleTypeDef que contiene la información de configuración para el módulo UART (puerto utilizado) especificado.
- **(uint8_t*)bufer**: variable o apuntador que contiene los datos a ser enviados, estos tienen que ser de tipo uint8_t.
- **(unit16_t)strlen(bufer):** El tercer parámetro debe ser cantidad de datos a enviar de tipo uint16_t. la función strlen calcula la cantidad de caracteres contenidos en el arreglo bufer y devuelve ese valor.
- **100**: tiempo de espera del puerto (ms), este dato tiene que se de tipo uint32_t.

La función sprintf otorga la capacidad de enviar una cadena de datos o una variables hacia un buffer o arreglo de tipo char. Es una función estándar de C y soporta todos los parámetros de la misma.

Finalmente, se compila el código y se descarga a la tarjeta para comenzar con la implementación, para lo cual se usará un convertidor FTDI USB, TeraTerm como terminal, y dos jumpers para conectar los Tx y Rx.

Para saber donde se encuentran los pines de Tx y Rx, hay que dirigirse a CubeMX y en la configuración del USART2 hacer clic en 'GPIO Settings' y entonces se puede apreciar el pin en el que se esta conectado, así como se puede ubicar en la imagen del microprocesador.

Figura 10: posición de los pines a usar para USART2

Se debe conectar el pin de Tx de la tarjeta al Rx del FTDI, así como el Rx de la tarjeta al Tx del FTDI.

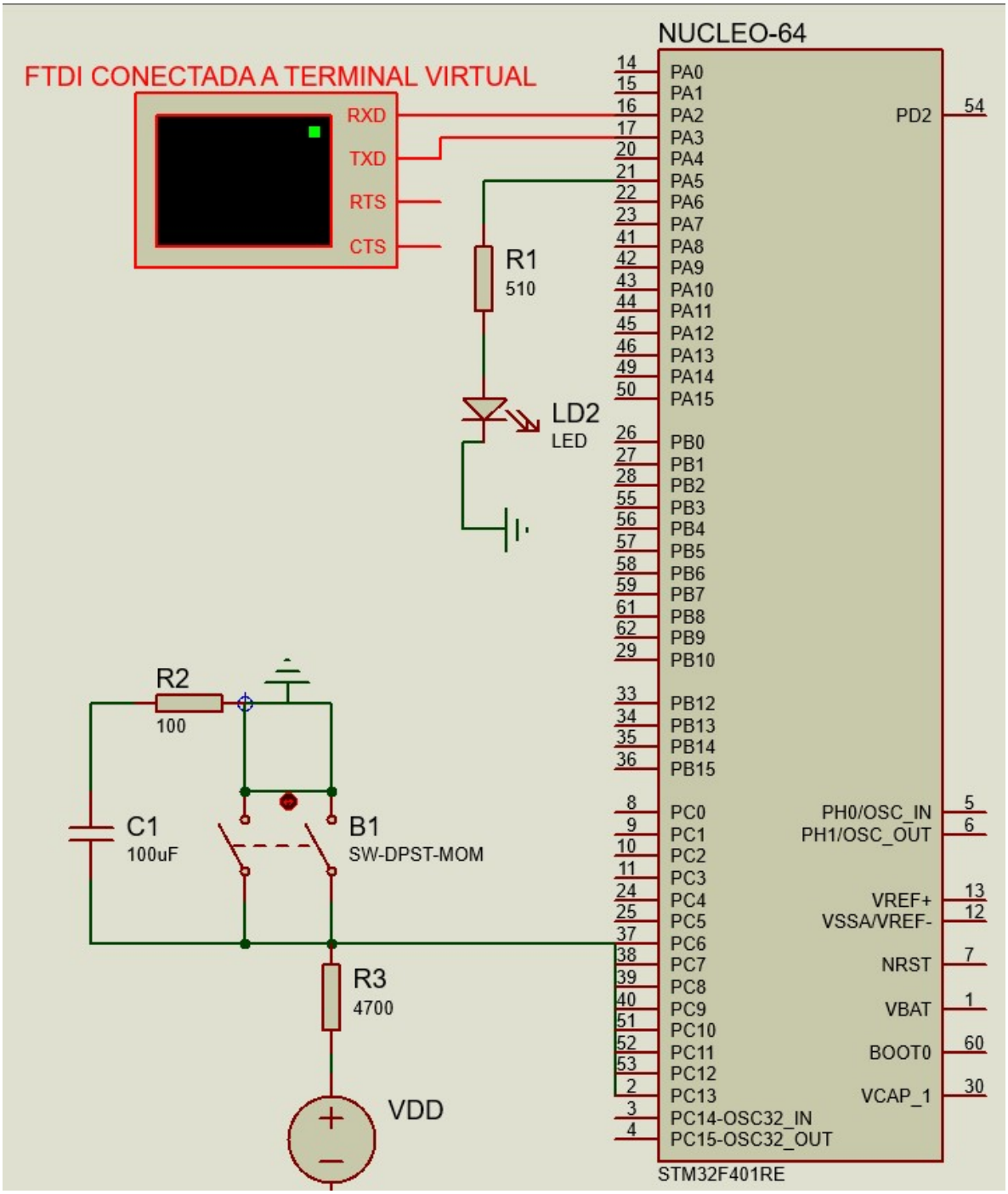

Figura 11: Esquemático de conexiones del procesador.

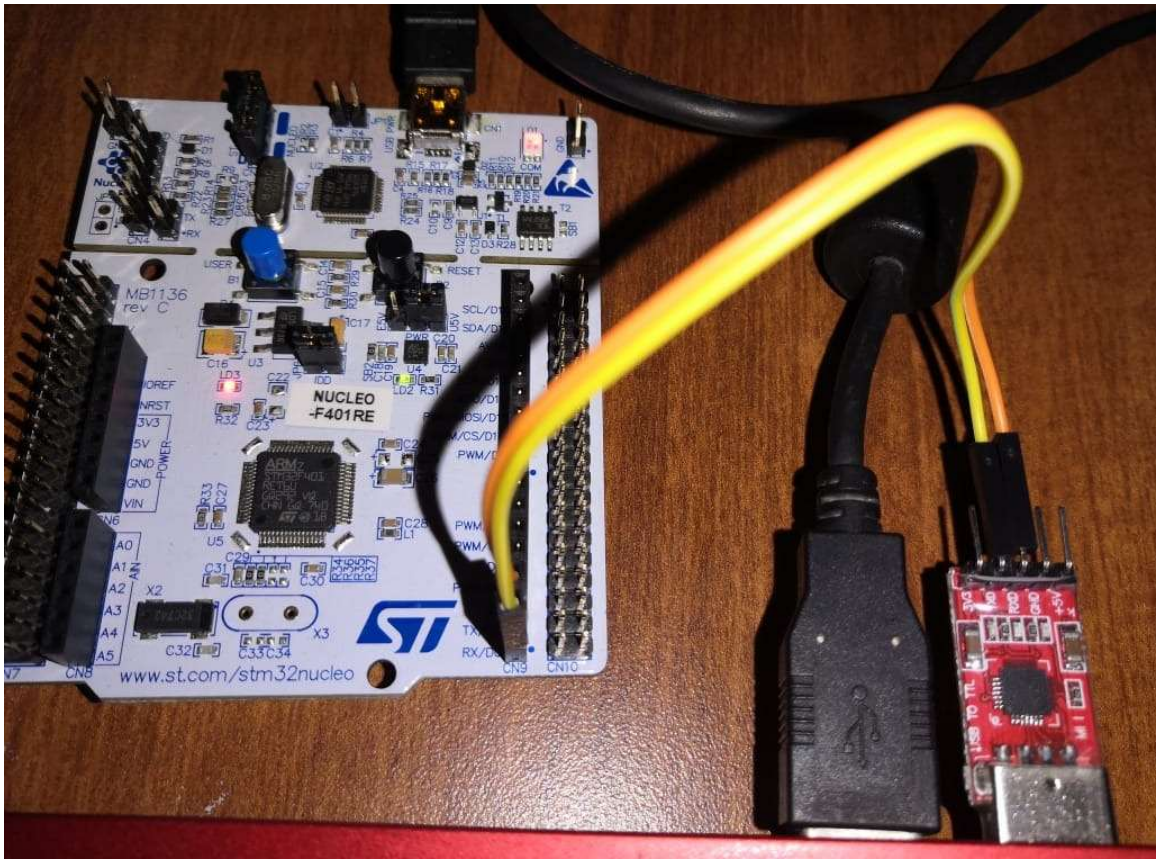

Figura 12: ejemplo de conexión.

Para poder ver los datos enviados por el procesador, y poderle enviar datos, se hará uso de TeraTerm, una terminal virtual muy útil para pruebas de serial. Solo hay que ir al sitio de TeraTerm, bajarlo y configurarlo.

Cuando abrimos TeraTerm, ofrece dos opciones: TCP/IP y Serial, se selecciona serial y luego en las opciones, la que corresponda al COM de la tarjeta de desarrollo.

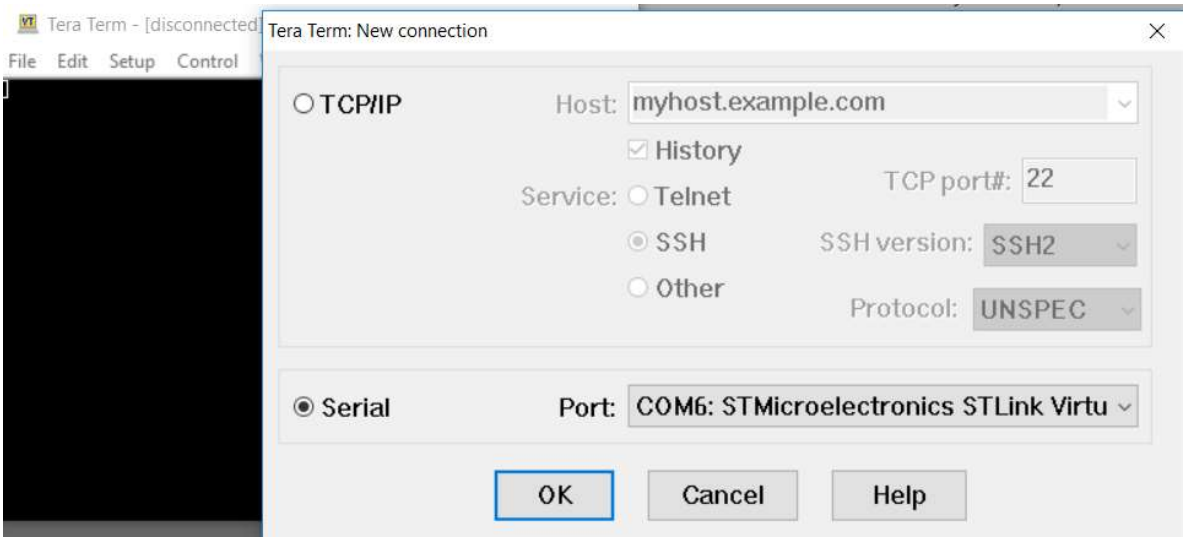

Figura 13: Configuración de Tera Term.

Ya que esta lista la terminal, se depura el código, se corre, y se debería ver algo como esto:

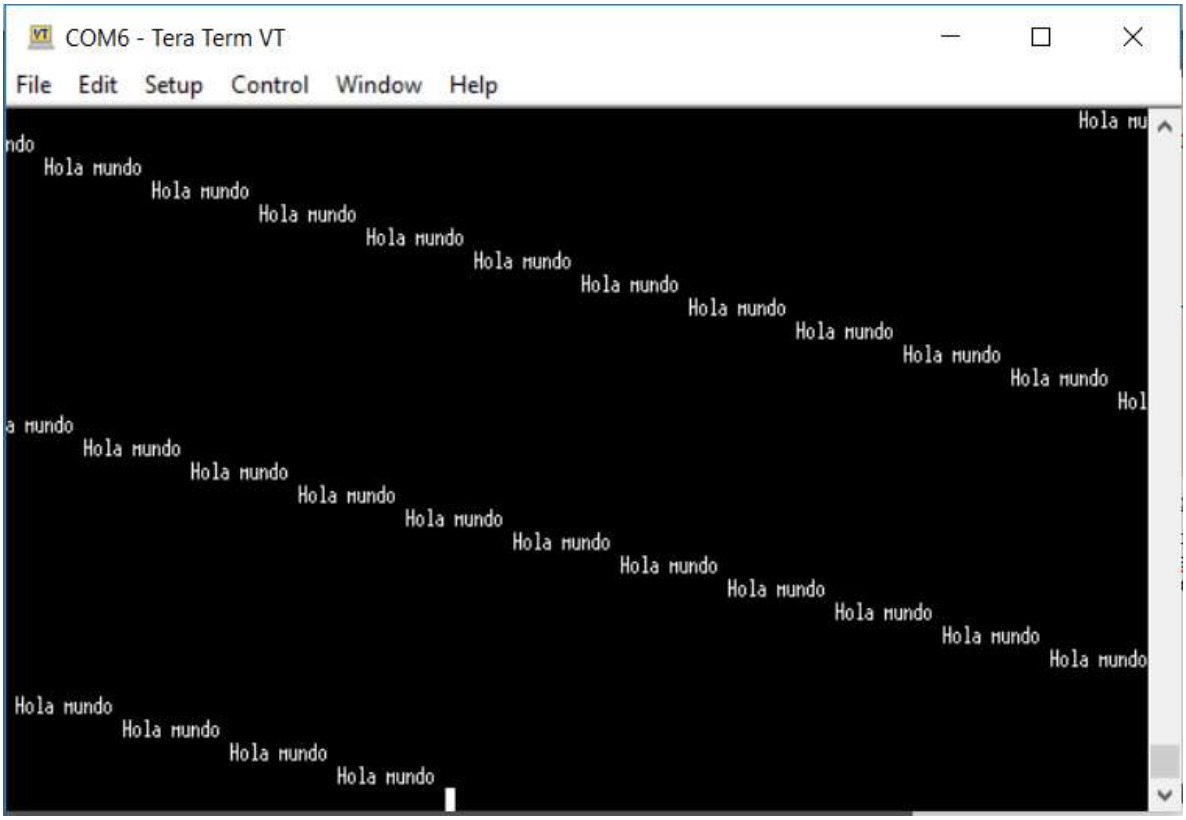

Figura 14: datos recibidos por TeraTerm

Se aprecia que no deja de llegar el dato, es debido a que el envío se encuentra en un while infinito.