STM32 CubeIDE con STM32F401RETx Núcleo-64

STM32CubelDE es una herramienta de desarrollo todo-en-uno para diversos sistemas operativos, y forma parte del ecosistema de STM32Cube.

Es una plataforma de desarrollo de C/C++ avanzada con configuración periférica, generación de código, compilación de código, compilación de código, y características para microcontroladores y microprocesadores STM32. Esta basado en el sistema ECLIPSE/CDT y la herramienta de desarrollo GCC, y GDB para depurar. Permite la integración de cientos de plugins que completan las características de ECLIPSE-IDE. STM32CubeIDE integra también todas las funcionalidades de STM32CubeMX para ofrecer una herramienta todo en uno que ahorre tiempo de instalación y de desarrollo. Después de seleccionar una STM32 MCU o MPU, o alguno preconfigurado del tablero de selección, el proyecto es creado y se genera un código de inicialización. En cualquier punto del desarrollo, el usuario puede regresar a la inicialización y configuración de los periféricos o middleware y regenerar el código de inicialización, sin ningún impacto al código del usuario.

De igual forma, este descargara los drivers necesarios para ST-Link/V2-1, que es una interfaz de depuración embebida en todas las STM32 Nucleo.

Primero se debe instalar desde la página de ST Electronics, en https://www.st.com/en/development-tools/stm32cubeide.html, para entonces proceder a configurar el proyecto, haciendo clic en "Start new STM32project".

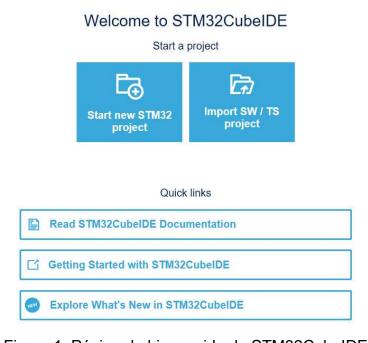


Figura 1: Página de bienvenida de STM32CubeIDE

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.

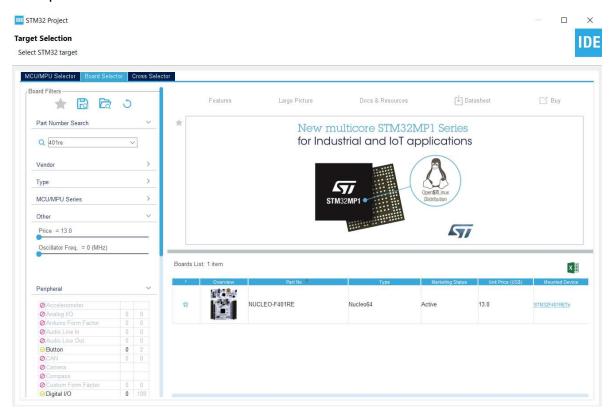


Figura 2: Selector de tarjeta

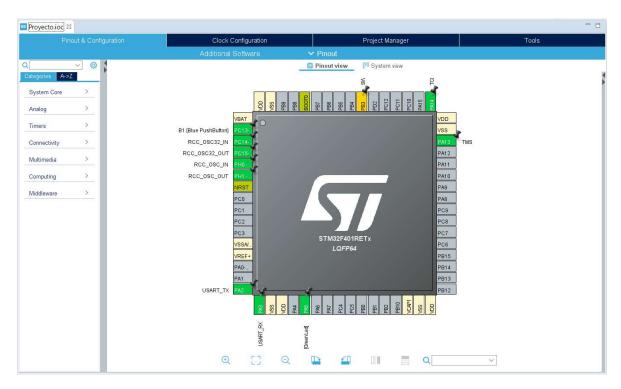


Figura 3: vista de CubeMX, herramienta de configuración del dispositivo

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

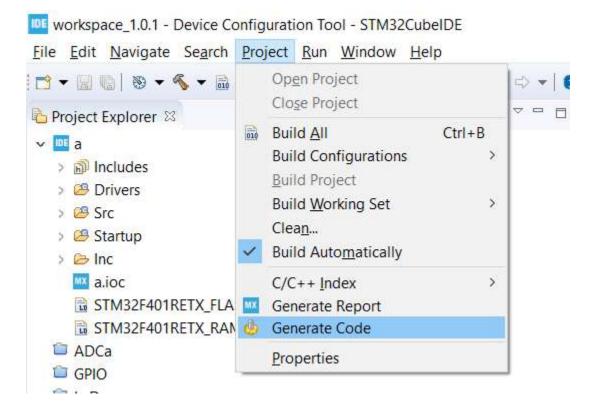


Figura 4: Generate Code.

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.

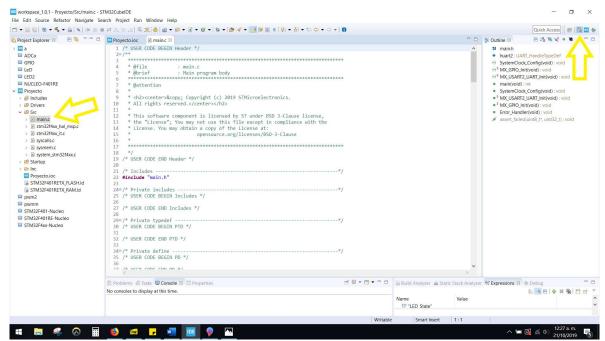
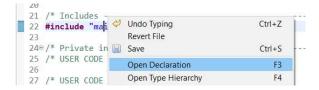


Figura 5: Vista en C/C++

Después se desplaza en el código hasta encontrar un while infinito junto al comentario de USER CODE BEGIN WHILE, y agregaremos las siguientes líneas para crear un programa que haga que el LED parpadee.

```
HAL_Delay(500);
HAL_GPIO_TogglePin(GPIOx, GPIO_Pin);
LED State = HAL GPIO ReadPin(GPIOx, GPIO Pin);
```

Para saber los puertos donde se encuentran ubicados, hay que dirigirse a #include "main.h", para dar clic derecho en main.h y seleccionar "open declaration", donde se pueden apreciar los puertos declarados y sus nombres.



```
🖻 main.c 🚨 *main.c 🕩 main.h 🖾 🔭 🗖 🔡 Outline 🕮
                                     # _MAIN_H
 51 /* USER CODE END EM */
                                      stm32f4xx_hal.h
                                     # Error_Handler(void)
 53 /* Exported functions prototypes
 54 void Error_Handler(void);
                                      # B1_Pin
                                     # B1_GPIO_Port
 56 /* USER CODE BEGIN EFP */
                                     # USART_TX_Pin
                                     # USART_TX_GPIO_Por
 58 /* USER CODE END EFP */
                                     # USART RX Pin
                                     # USART_RX_GPIO_Por
 60 /* Private defines -----
                                      # LD2_Pin
 61 #define B1_Pin GPIO_PIN_13
 62 #define B1_GPIO_Port GPIOC
                                      # LD2_GPIO_Port
 63 #define USART_TX_Pin GPIO_PIN_2
                                      # TMS_Pin
 64 #define USART_TX_GPIO_Port GPIOA
                                     # TMS_GPIO_Port
65 #define USART_RX_Pin GPIO_PIN_3
                                      # TCK_Pin
 66 #define USART_RX_GPIO_Port GPIOA
                                      # TCK_GPIO_Port
67 #define LD2_Pin GPIO_PIN_5
68 #define LD2_GPIO_Port GPIOA
                                      # SWO_Pin
                                      # SWO_GPIO_Port
 69 #define TMS Pin GPIO PIN 13
 70 #define TMS_GPIO_Port GPIOA
 71 #define TCK Pin GPIO PIN 14
 72 #define TCK_GPIO_Port GPIOA
 73 #define SWO Pin GPIO PIN 3
 74 #define SWO GPIO Port GPIOB
                                                      Figuras 6 y 7: Abrir declaración y ver
75 /* USER CODE BEGIN Private define
nombres de pines
    while (1)
          HAL_Delay(500);
          HAL_GPIO_TogglePin(LD2_GPIO_Port, LD2_Pin);
          LED State = HAL GPIO ReadPin(LD2 GPIO Port, LD2 Pin);
       /* USER CODE END WHILE */
        /* USER CODE BEGIN 3 */
```

Figura 8: funciones con puertos configurados

Ya que este listo, se da clic a F11, o a Run>Debug, lo cual llevará a la vista de depuración, y el depurador se detendrá en la primer declaración, entonces se da clic al botón de "Resume" o a la tecla F8, y el LED de usuario, el verde debajo del botón negro de la tarjeta, debe de comenzar a parpadear.

```
workspace_1.0.1 - LED2/Src/main.c - STM32CubeIDE
<u>File Edit Source Refactor Navigate Search Project Run Window Help</u>
/* MCU Configuration----*/
                🕹 | 🛬 | i 🕶 🔻
                             75
                             76

✓ ■ LED2.elf [STM32 MCU Debugging]

                                 /* Reset of all peripherals, Initializes the Flash interface and the Systick. */

✓ 

R LED2.elf [cores: 0]

                             78 HAL_Init();
    79
       main() at main.c:78 0x800187
                                 /* USER CODE BEGIN Init */
                             80
   C:/ST/STM32CubeIDE_1.0.1/STM32
                             81
   Jacobs St. LINK (ST-LINK GDB server)
                             82 /* USER CODE END Init */
                             83
                                 /* Configure the system clock */
                             84
                             85
                                SystemClock_Config();
                             86
                                 /* USER CODE BEGIN SysInit */
                             87
                             88
                                 /* USER CODE END SysInit */
                             89
                             90
                                 /* Initialize all configured peripherals */
                             91
                             92
                                 MX_GPIO_Init();
                                 MX_USART2_UART_Init();
                             93
                                 /* USER CODE BEGIN 2 */
                             94
                             95
                             96
                                 /* USER CODE END 2 */
                             97
                                /* Infinite loop */
                             98
                                 /* USER CODE BEGIN WHILE */
                             99
                            100
                                 while (1)
                            101
                            102
                                    HAL_Delay(500);
                                    HAL GPIO TogglePin(LD2 GPIO Port, LD2 Pin);
                            103
                            101
```

Figura 9: Vista de depuración

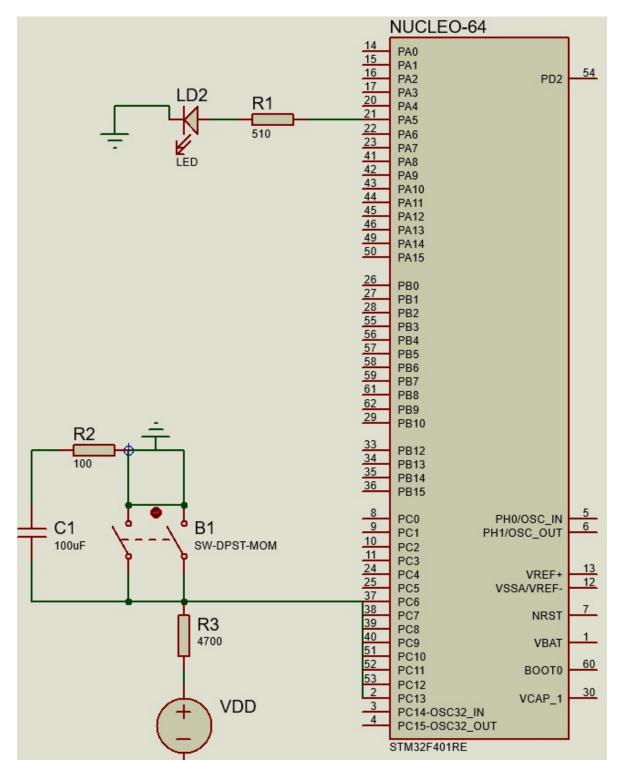


Figura 10: esquemático de la conexión del botón y el LED en el procesador.