

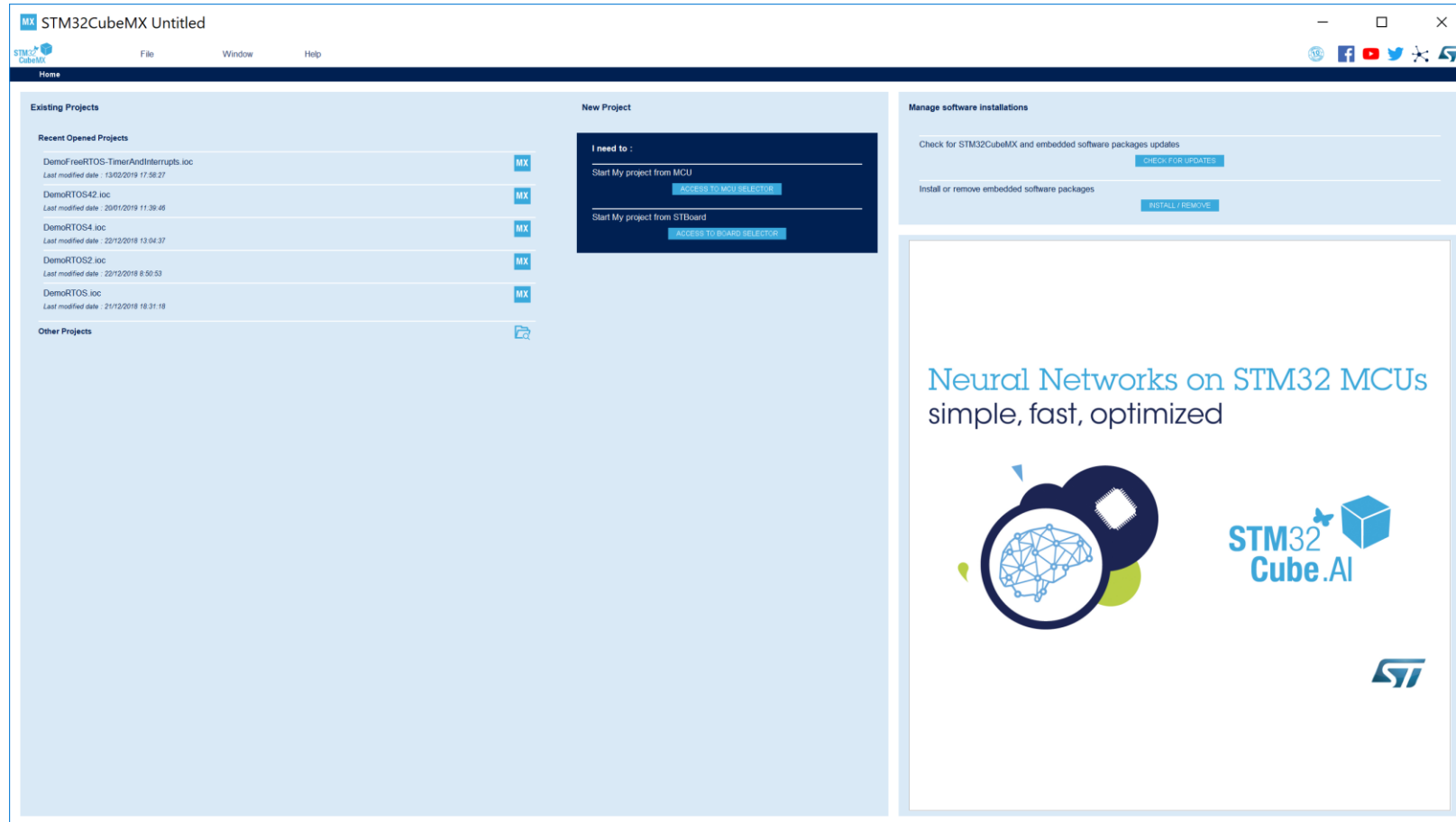
Embedded RTOS Assignment 7

By

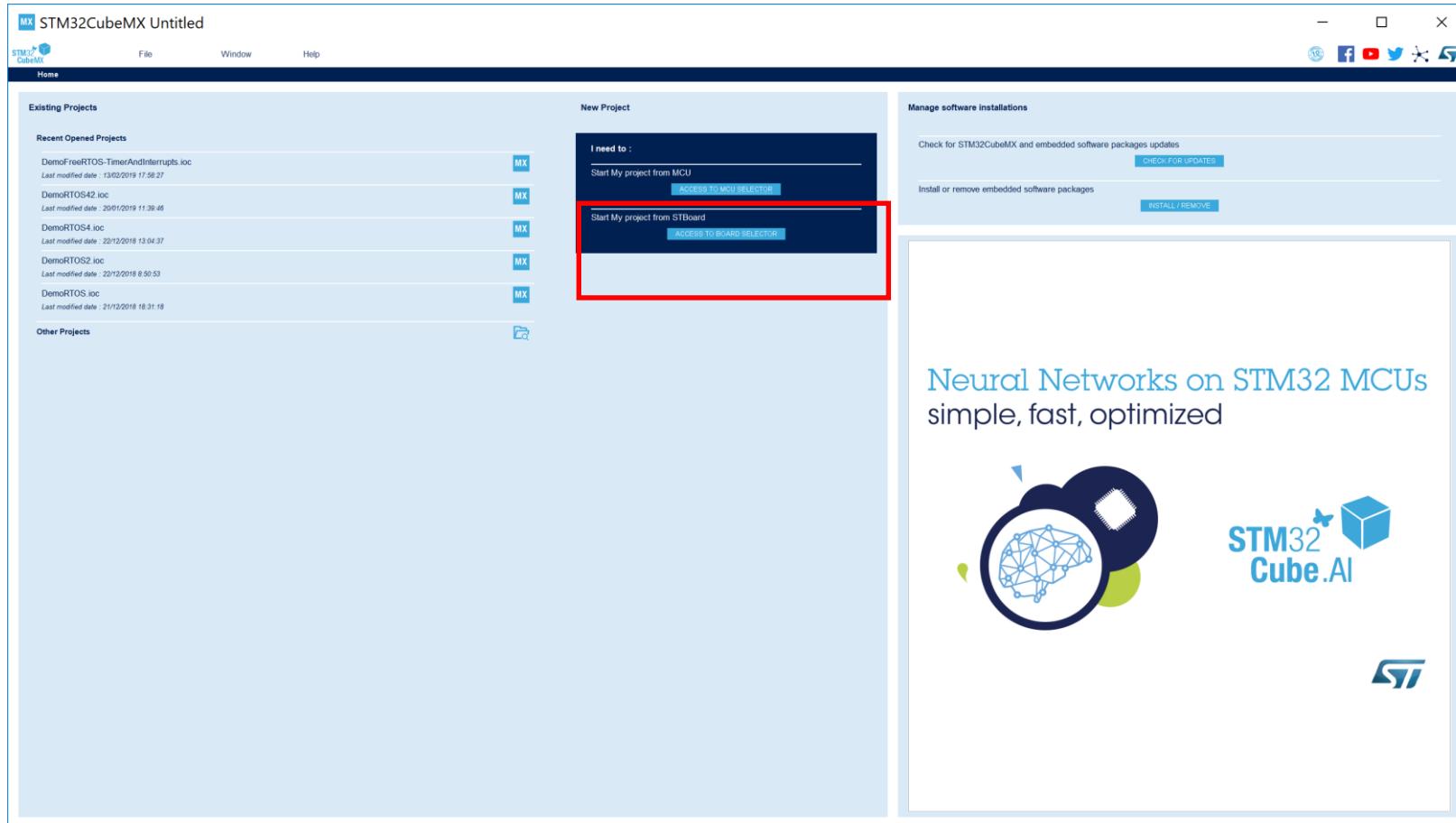
Norman McEntire

Norman.mcentire@gmail.com

Step 1. Startup STM32CubeMX



Step 2. Access Board Selector



Step 3. Select “B-L475E-IOT01A” Board

The screenshot shows the 'New Project from a Board' dialog in STM32CubeIDE. The 'Board Selection' tab is active, displaying a list of boards on the left and a detailed view of the selected board on the right.

Board Selection Panel (Left):

- Part Number Search: B-L475E-IOT01A
- Vendor: STMicroelectronics
- Type: Discovery
- MCU Series: STM32L4
- Price: ~ \$3.8
- Peripheral list (checked): Accelerometer, Gyroscope, Magnetometer, Microphone, On-board LED, Power Source, Pressure Sensor, ROM, RS-232, RTC, Temperature Sensor, USB.

Board Details Panel (Right):

B-L475E-IOT01A

STM32Cube B-L475E-IOT01A IOT Discovery Board Support and Examples

ACTIVE Active
Product is in mass production

Unit Price (USD): \$3.8
Mounted device: STM32L475G6Z

The B-L475E-IOT01A Discovery kit for IoT node allows users to develop applications with direct connection to cloud servers. The Discovery kit enables a wide diversity of applications by exploiting the power communication, multi-sensing and ARM Cortex-M4 core-based STM32L4 Series features. The support for Arduino Uno V3 and PMOD connectivity provides unlimited expansion capabilities with a large choice of specialized add-on boards.

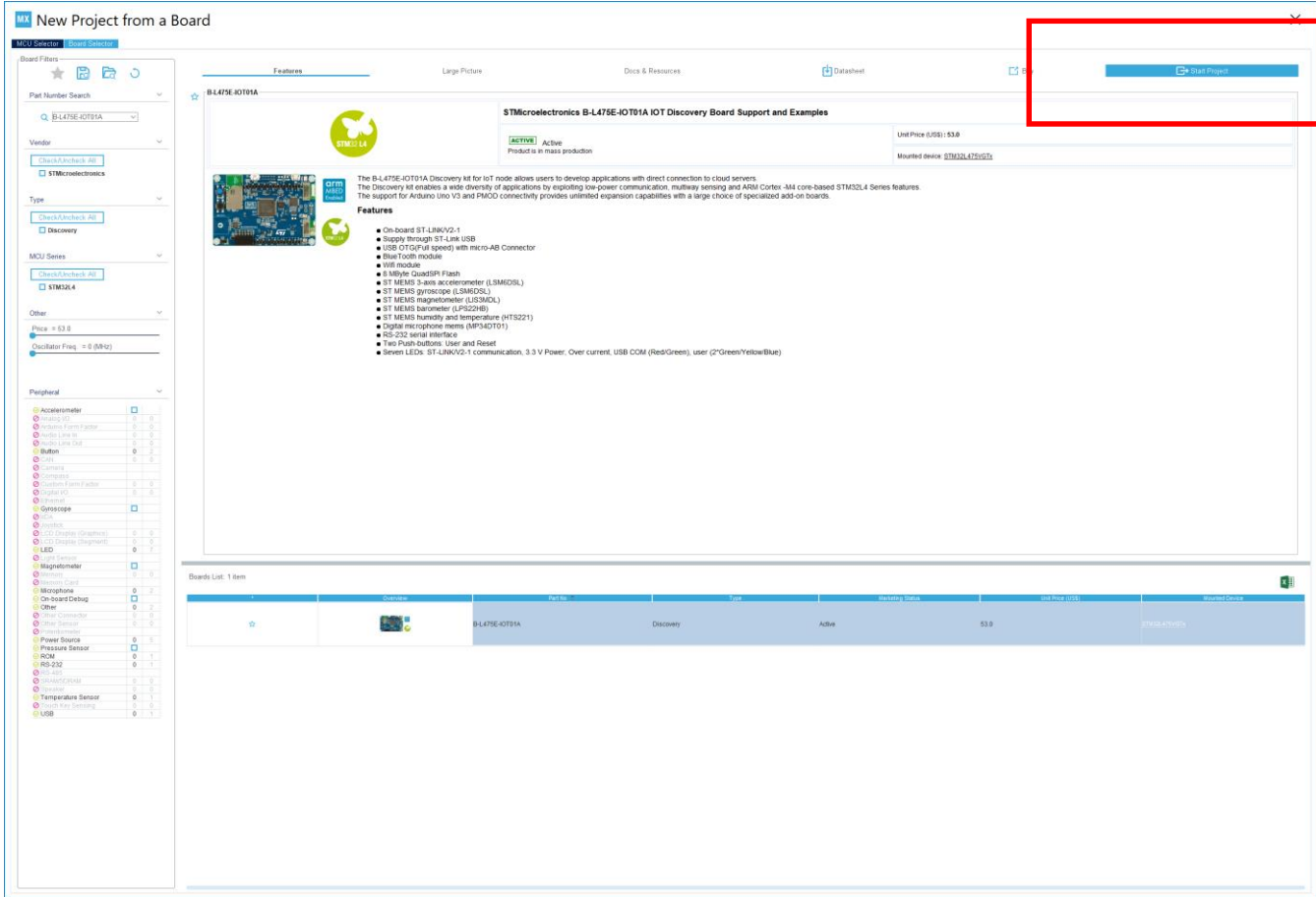
Features

- On-board ST-LINKV2-1
- Supply through ST-Link USB
- USB OTG (Full speed) with micro-AB Connector
- Blue Tooth module
- IoT module
- 8 KByte QuadSPI Flash
- ST MEMS 3-axis accelerometer (LSM2DS1)
- ST MEMS gyroscope (LSM2DS1)
- ST MEMS magnetometer (LSM2DS1)
- ST MEMS barometer (LPS22DH)
- ST MEMS humidity and temperature (HTS221)
- Digital microphone (MP34DT01)
- RS-232 serial interface
- Two Push-buttons: User and Reset
- Seven LEDs: ST-LINKV2-1 communication, 3.3 V Power, Over current, USB COM (Red/Green), User (2*Green/Yellow/Blue)

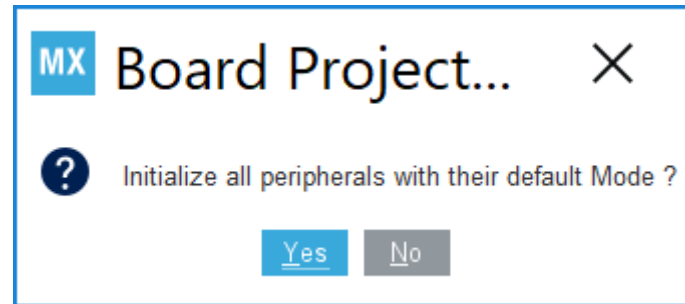
Boards List: 1 item

Image	Part Number	Type	Status	Unit Price (USD)	Mounted device
	B-L475E-IOT01A	Discovery	Active	\$3.8	STM32L475G6Z

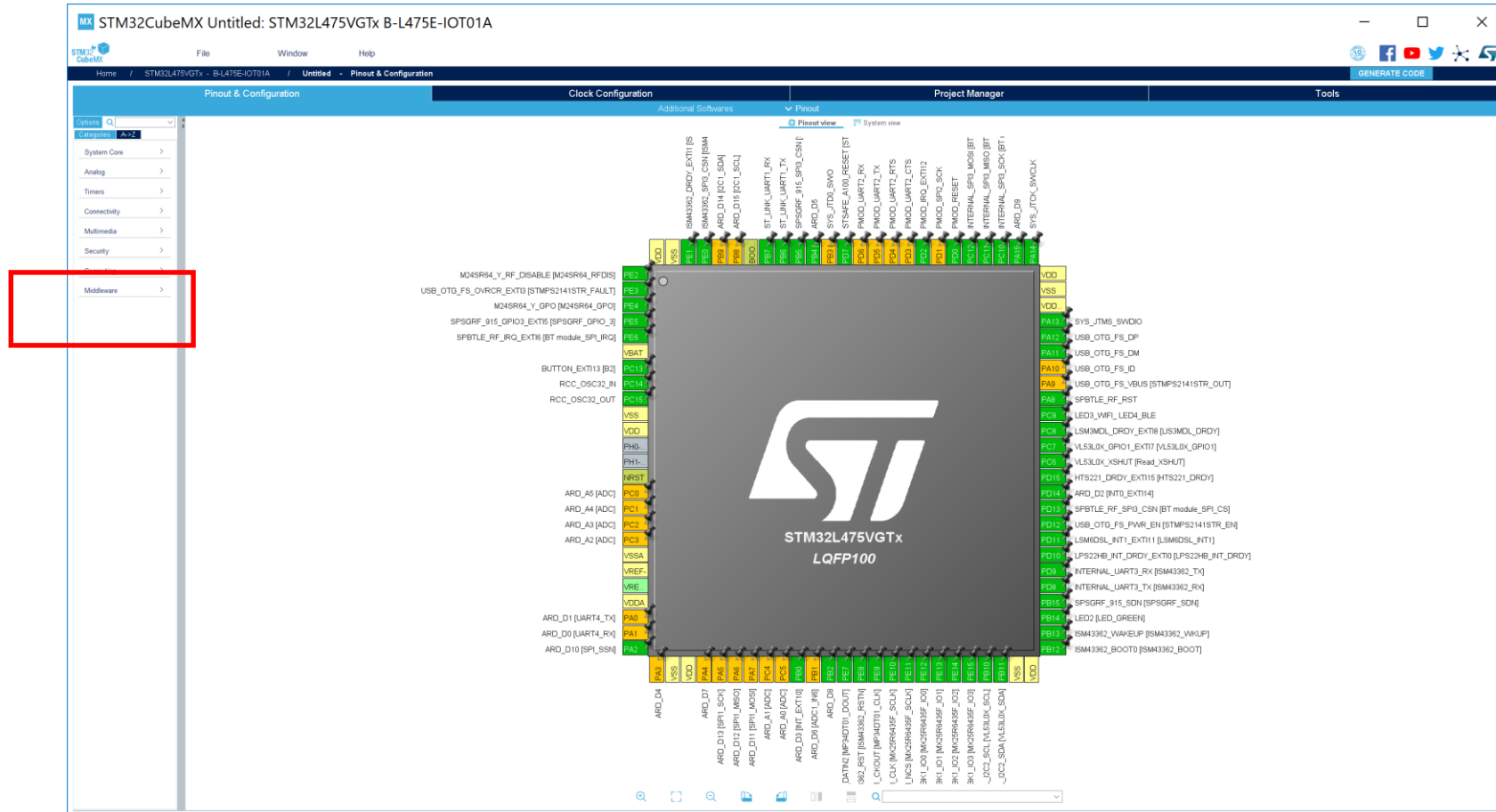
Step 4. Select “Start Project”



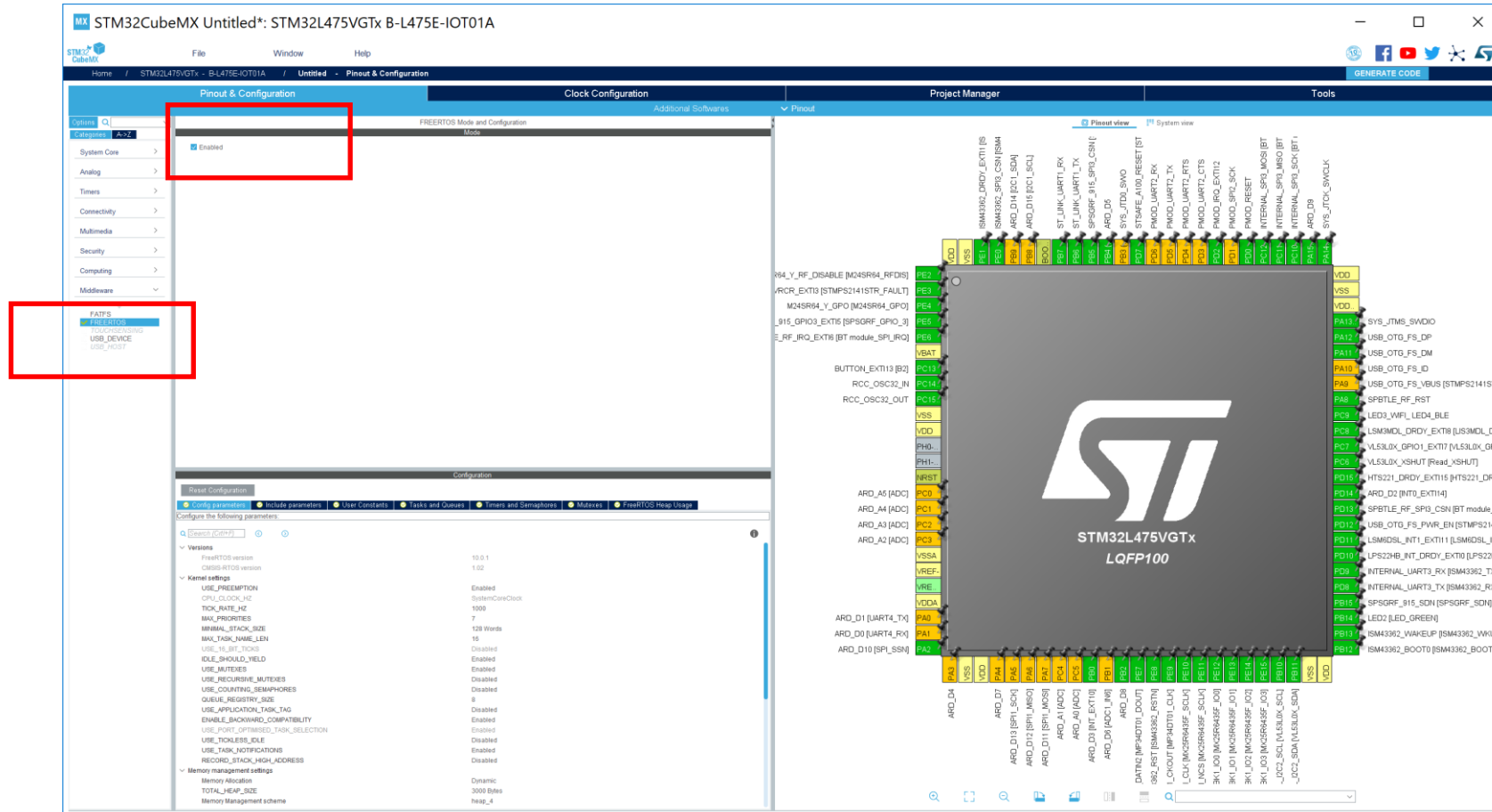
Step 5. Select YES: “Initialize all peripherals with their default Mode”



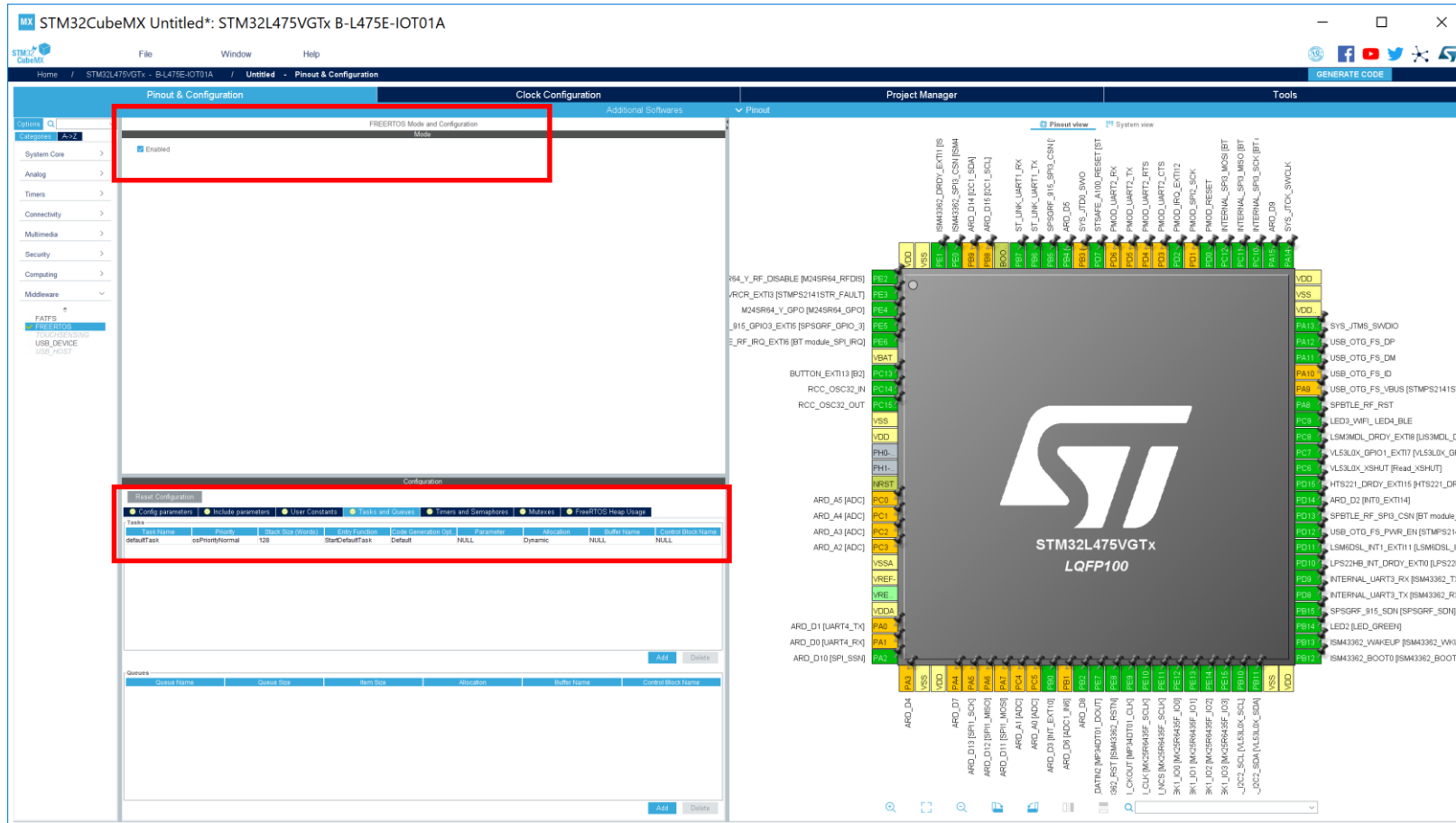
Step 6. Select “Middleware”



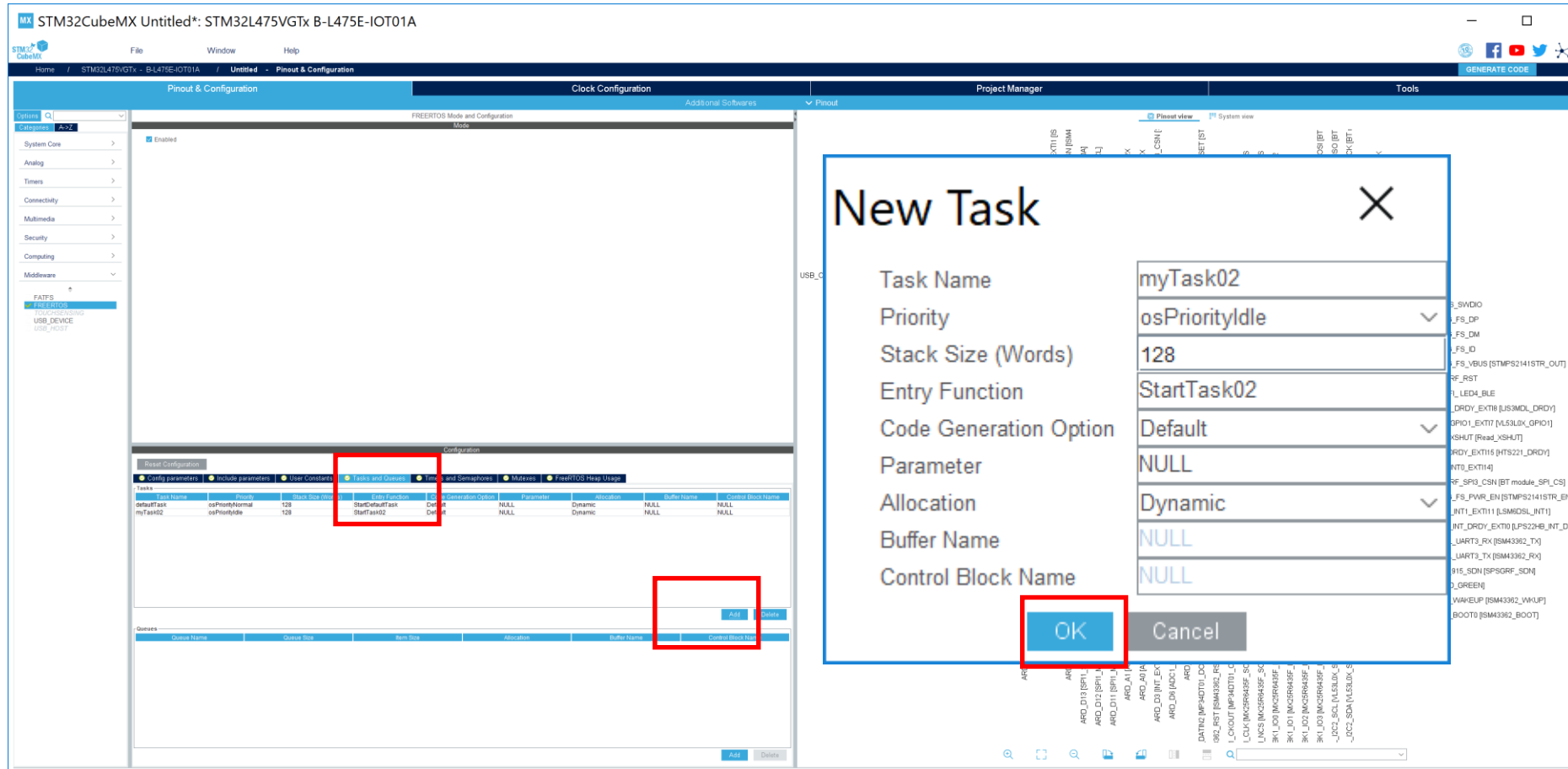
Step 7. Select “FreeRTOS”, then select “Enable”



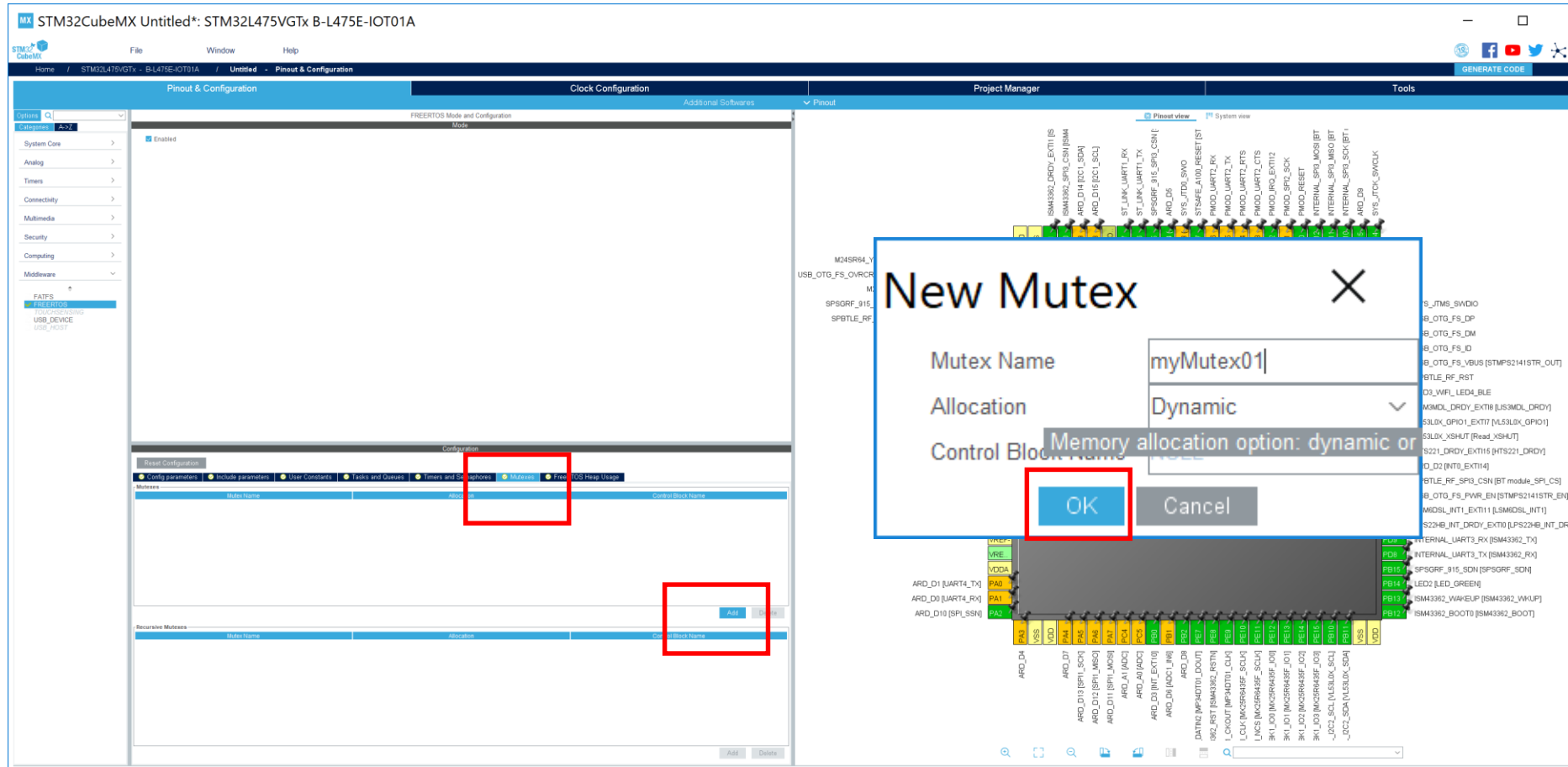
Step 8. Select “Task and Queues”, and observe 1 default task created



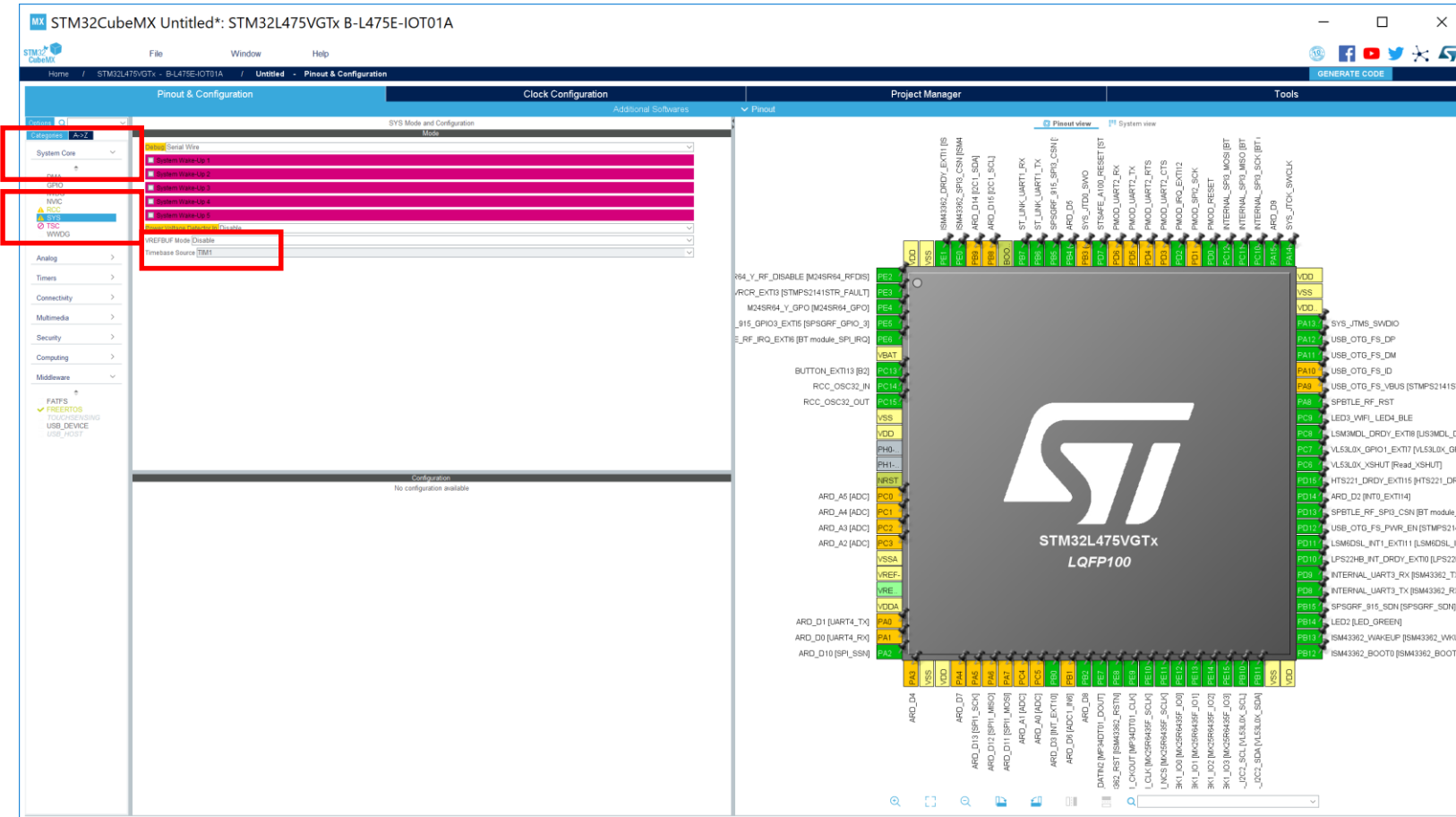
Step 9. Click Add to add 2nd Task



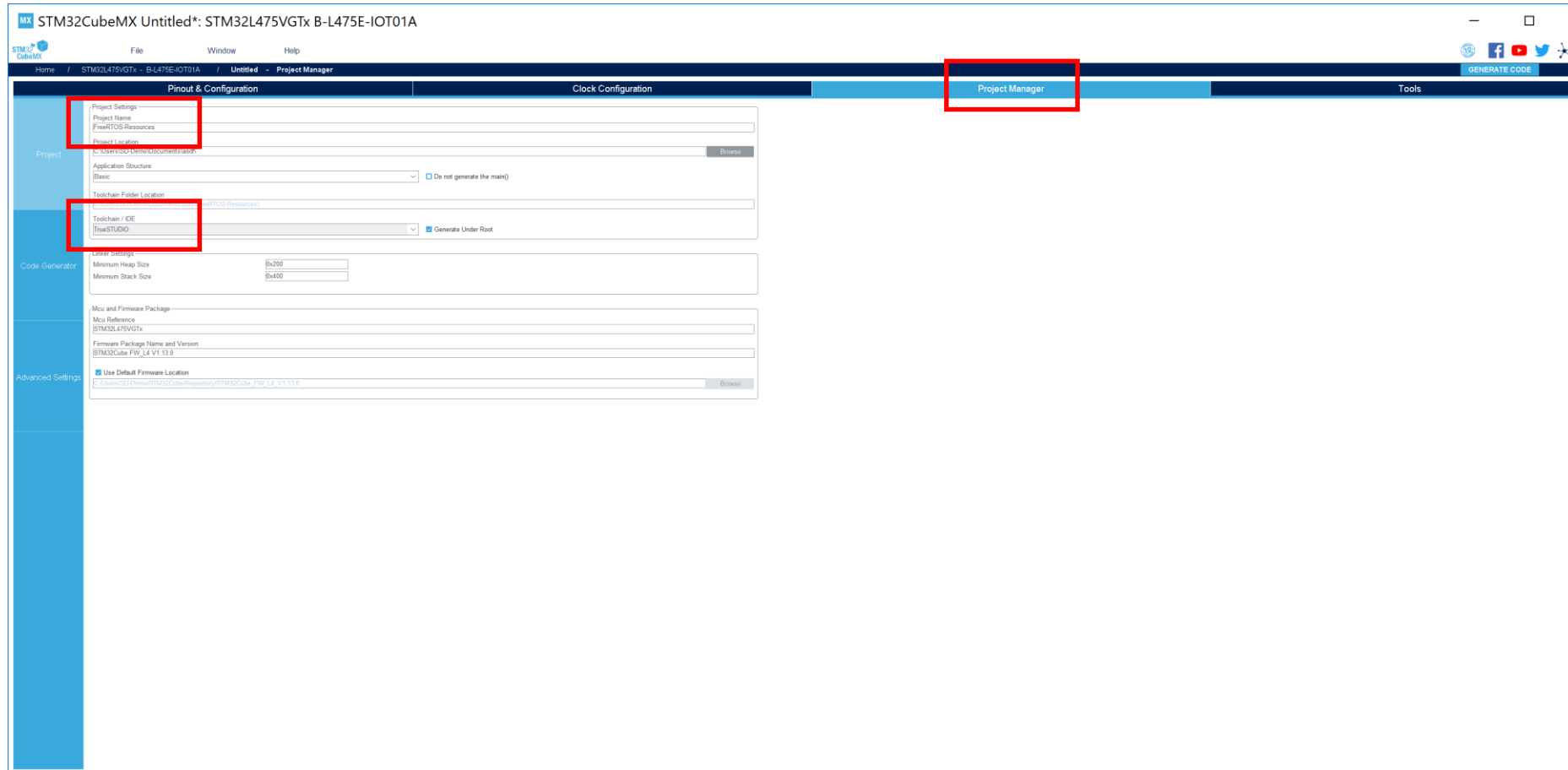
Step 10. Select Mutexes, then Add, then OK



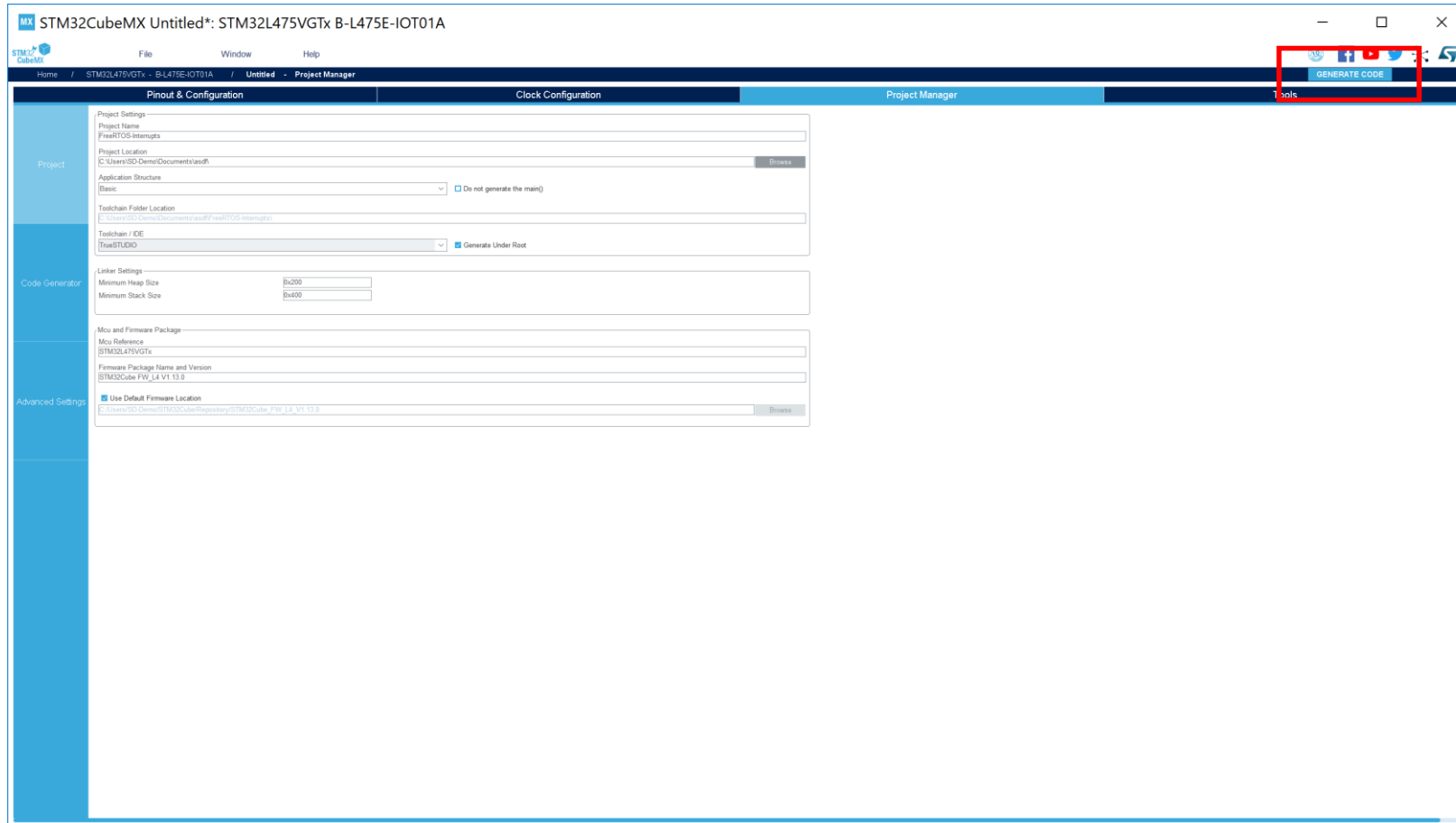
Step 11. Select System Core, Sys, Timebase Source, TIM1



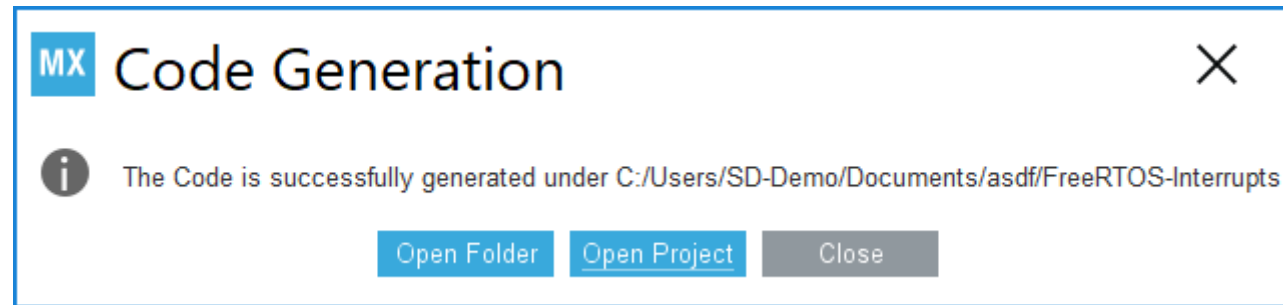
Step 12. Enter Project Name: “FreeRTOS-Resources” and Toolchain/IDE: TrueStudio



Step 13. Select “Generate Code”



Step 14 Select “Open Project”



Step 15. Add interrupt code to increment counter

```
211
212 int count;
213
214 void EXTI15_10_IRQHandler(void)
215 {
216     /* USER CODE BEGIN EXTI15_10_IRQn 0 */
217     int ret = taskENTER_CRITICAL_FROM_ISR();
218     count++;
219     taskEXIT_CRITICAL_FROM_ISR(ret);
220
221     /* USER CODE END EXTI15_10_IRQn 0 */
222     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_10);
223     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_11);
224     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_13);
225     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_14);
226     HAL_GPIO_EXTI_IRQHandler(GPIO_PIN_15);
227     /* USER CODE BEGIN EXTI15_10_IRQn 1 */
228
229     /* USER CODE END EXTI15_10_IRQn 1 */
230 }
231
```

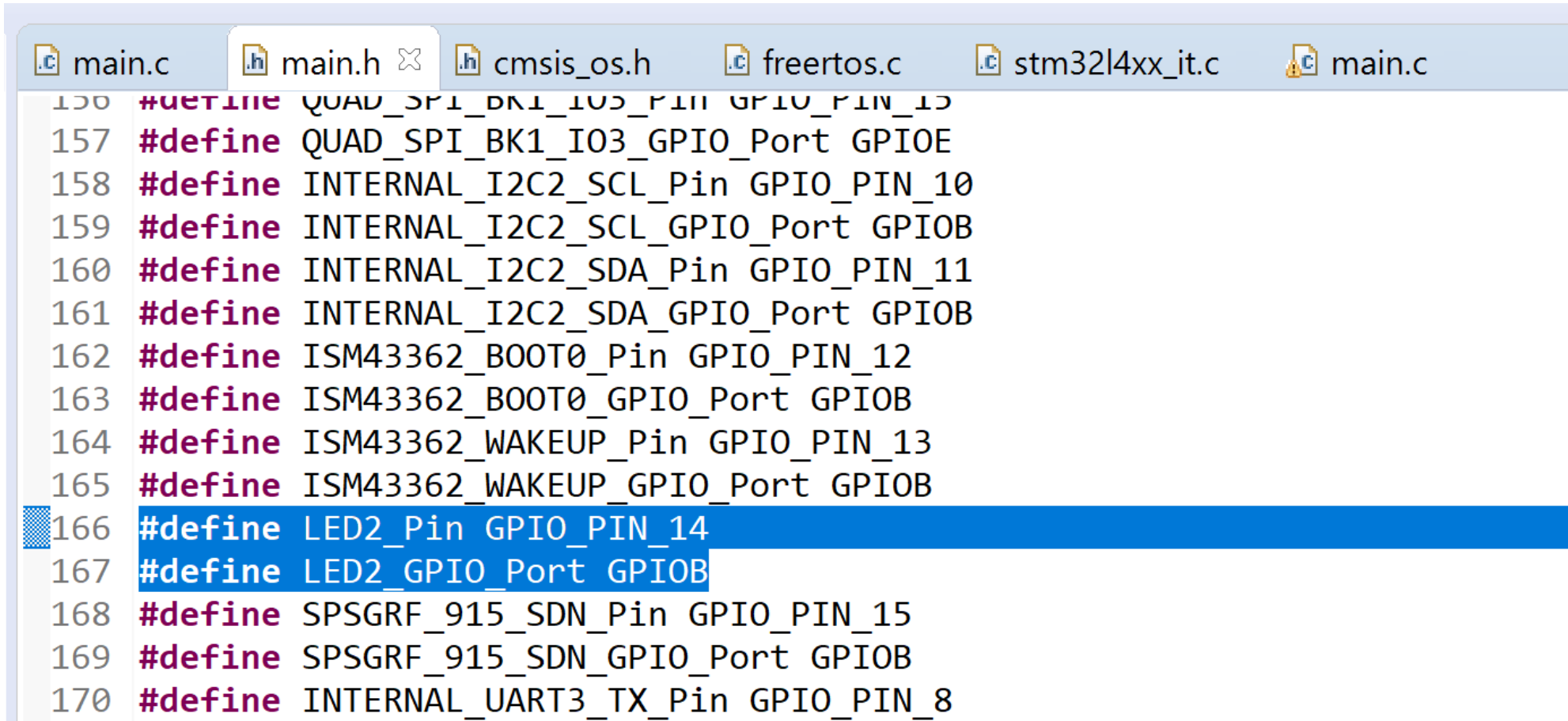

Step 16. Add Task 1 Code to read count and flash LED that many times after grabbing mutex

```
stm32l4xx_it.c  *main.c  portmacro.h  task.h  stm32l4xx_hal.h
737 extern int count;
738
739 void StartDefaultTask(void const * argument)
740 {
741     /* USER CODE BEGIN 5 */
742     for(;;)
743     {
744         osDelay(2000); // Sleep 2 seconds
745         // Read count value
746         int flashCount = 0;
747         taskENTER_CRITICAL();
748         flashCount = count;
749         // If count is 0 then nothing to do
750         if (count == 0) {
751             taskEXIT_CRITICAL();
752             continue;
753         }
754         // Else clear the count
755         count = 0;
756         taskEXIT_CRITICAL();
757
758         // Grab the mutex
759         osMutexWait (myMutex01Handle, osWaitForever);
760
761         // Flash the LED flashCount times
762         while (flashCount != 0) {
763             flashCount--;
764             osDelay(500);
765             HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin );
766         }
767         //Always end with LED off
768         HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 0);
769
770         // Release the mutex
771         osMutexRelease (myMutex01Handle);
772     }
```

Step 17. Add code for 2nd Task

```
stm32l4xx_it.c  *main.c  portmacro.h  task.h  stm32l4xx_hal_gp
785 /* USER CODE END Header_StartTask02 */
786 void StartTask02(void const * argument)
787 {
788     for(;;)
789     {
790         osDelay(5000); // Sleep 2 seconds
791         // Read count value
792         int flashCount = 0;
793         taskENTER_CRITICAL();
794         flashCount = count;
795         // If count is 0 then nothing to do
796         if (count == 0) {
797             taskEXIT_CRITICAL();
798             continue;
799         }
800         // Else clear the count
801         count = 0;
802         taskEXIT_CRITICAL();
803
804         // Grab the mutex
805         osMutexWait (myMutex01Handle, osWaitForever);
806
807         //Always end with LED ON
808         HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 1);
809
810         // Flash the LED flashCount times
811         while (flashCount != 0) {
812             flashCount--;
813             osDelay(1000);
814             HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin );
815         }
816         //Always end with LED off
817         HAL_GPIO_WritePin(LED2_GPIO_Port, LED2_Pin, 0);
818
819         // Release the mutex
820         osMutexRelease (myMutex01Handle);
821     }
}
```

Screenshot of LED2 #defines



The screenshot shows an IDE window with several tabs: main.c, main.h, cmsis_os.h, freertos.c, stm32l4xx_it.c, and another main.c. The main.h tab is active, displaying a list of #define statements. Line 166, which defines LED2_Pin as GPIO_PIN_14, is highlighted with a blue background. The code is as follows:

```
156 #define QUAD_SPI_BK1_IO3_Pin GPIO_PIN_13
157 #define QUAD_SPI_BK1_IO3_GPIO_Port GPIOE
158 #define INTERNAL_I2C2_SCL_Pin GPIO_PIN_10
159 #define INTERNAL_I2C2_SCL_GPIO_Port GPIOB
160 #define INTERNAL_I2C2_SDA_Pin GPIO_PIN_11
161 #define INTERNAL_I2C2_SDA_GPIO_Port GPIOB
162 #define ISM43362_BOOT0_Pin GPIO_PIN_12
163 #define ISM43362_BOOT0_GPIO_Port GPIOB
164 #define ISM43362_WAKEUP_Pin GPIO_PIN_13
165 #define ISM43362_WAKEUP_GPIO_Port GPIOB
166 #define LED2_Pin GPIO_PIN_14
167 #define LED2_GPIO_Port GPIOB
168 #define SPSGRF_915_SDN_Pin GPIO_PIN_15
169 #define SPSGRF_915_SDN_GPIO_Port GPIOB
170 #define INTERNAL_UART3_TX_Pin GPIO_PIN_8
```

Screenshot of BLUE Button Interrupt Defines

```
main.c main.h cmsis_os.h freertos.c stm32l4xx_it.c main.c
96 #define M24SR04_Y_GPIO_GPIO_Port GPIOE
97 #define SPSGRF_915_GPIO3_EXTI5_Pin GPIO_PIN_5
98 #define SPSGRF_915_GPIO3_EXTI5_GPIO_Port GPIOE
99 #define SPSGRF_915_GPIO3_EXTI5_EXTI_IRQn EXTI9_5_IRQn
100 #define SPBTLE_RF_IRQ_EXTI6_Pin GPIO_PIN_6
101 #define SPBTLE_RF_IRQ_EXTI6_GPIO_Port GPIOE
102 #define SPBTLE_RF_IRQ_EXTI6_EXTI_IRQn EXTI9_5_IRQn
103 #define BUTTON_EXTI13_Pin GPIO_PIN_13
104 #define BUTTON_EXTI13_GPIO_Port GPIOC
105 #define BUTTON_EXTI13_EXTI_IRQn EXTI15_10_IRQn
106 #define ARD_A5_Pin GPIO_PIN_0
107 #define ARD_A5_GPIO_Port GPIOC
108 #define ARD_A4_Pin GPIO_PIN_1
109 #define ARD_A4_GPIO_Port GPIOC
```