

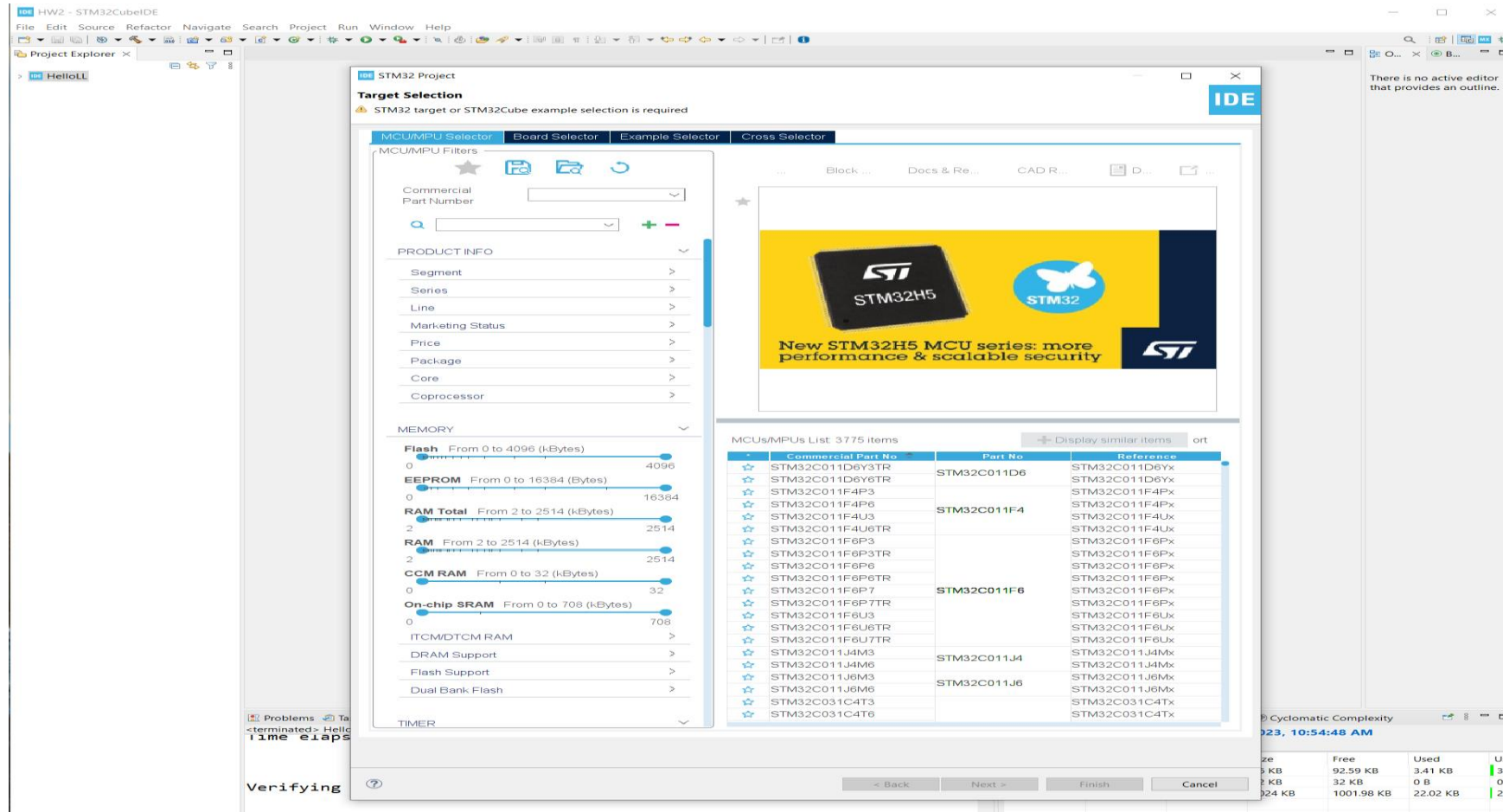
UCSD Embedded C Final Assignment

By

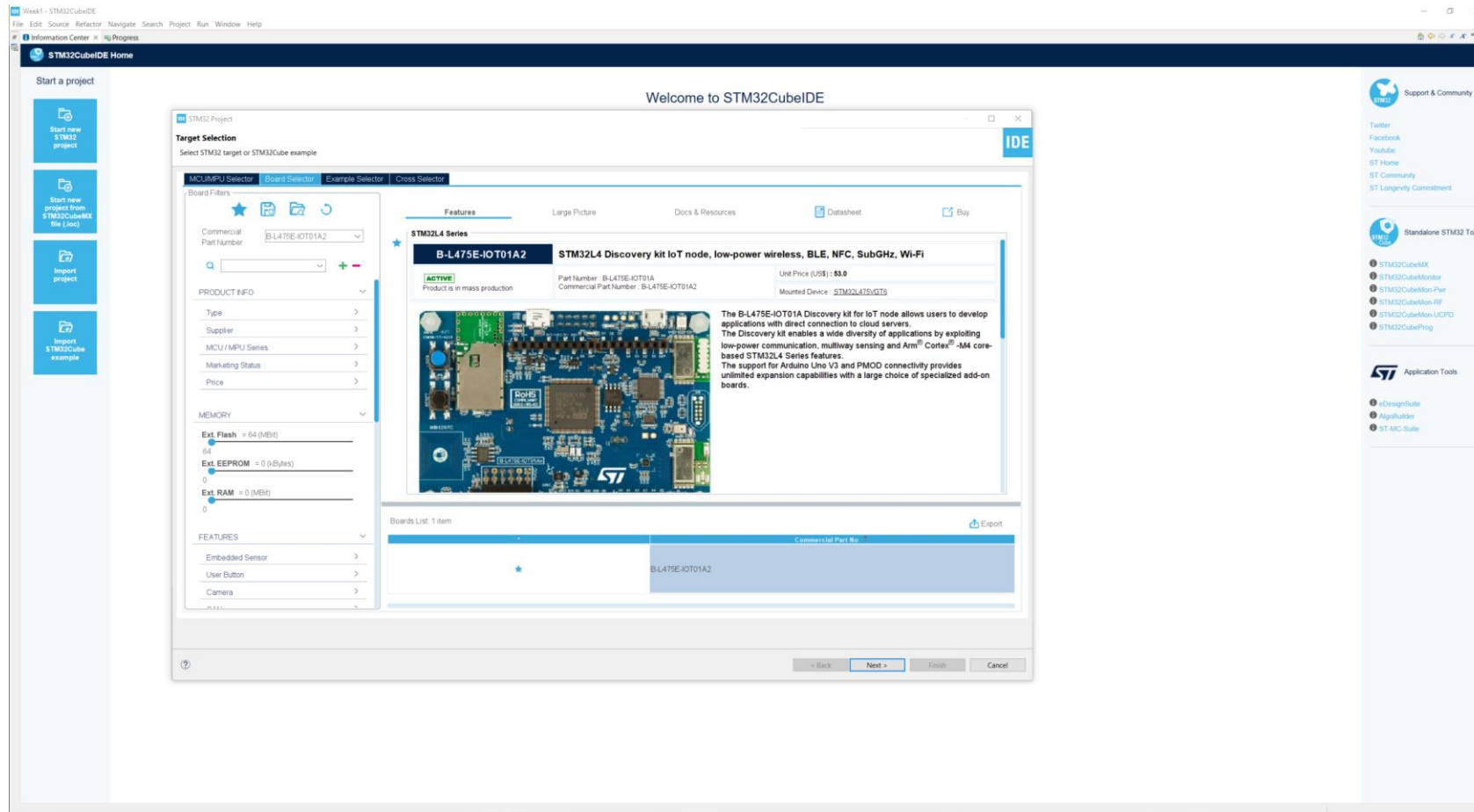
Hsuankai Chang

hsuankac@umich.edu

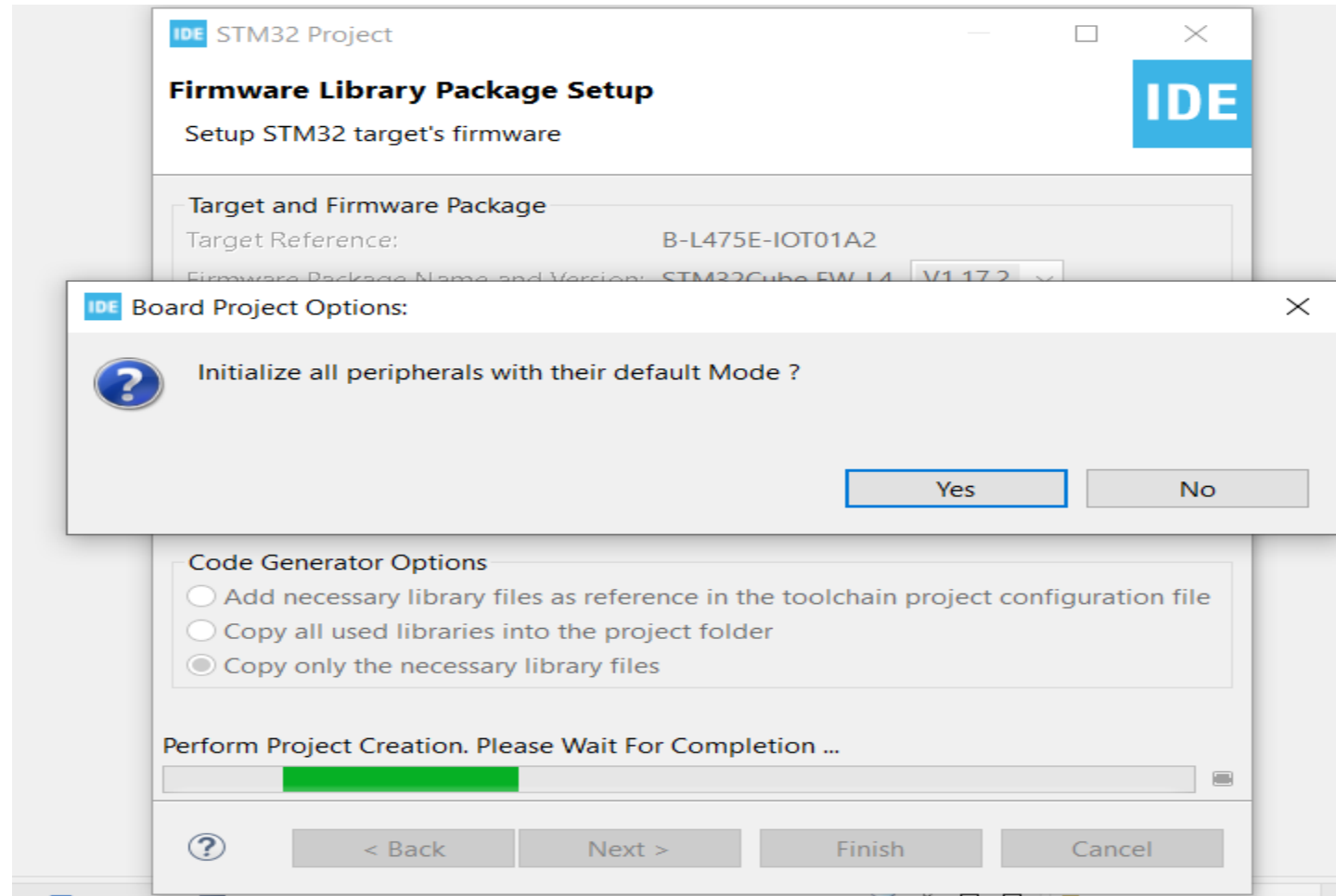
Step 1. Startup STM32CubeIDE and create new STM32 project



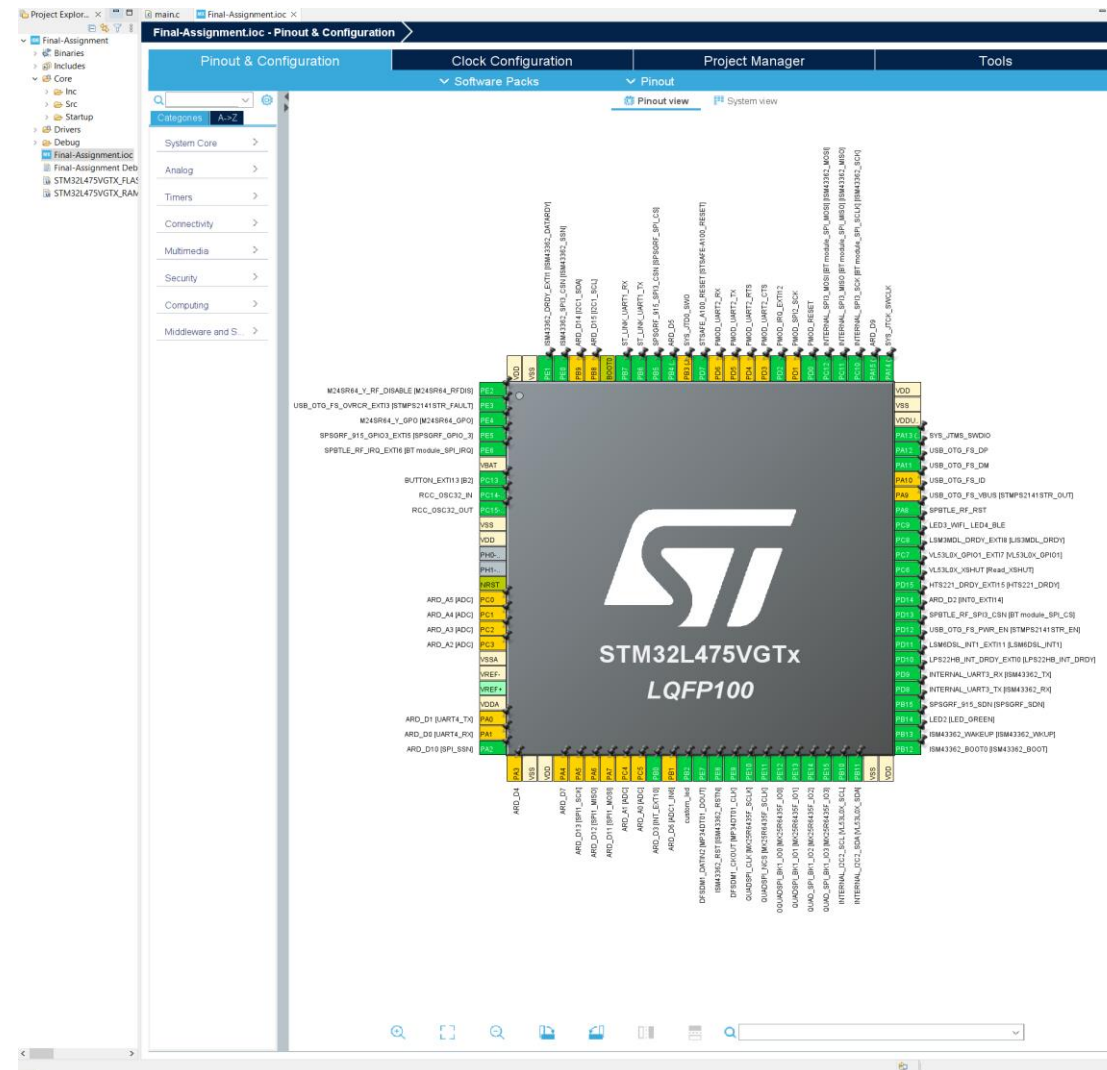
Step 2. Access board selector and type in the board you use, enter project name and click Next



Step 3. Click yes to initialize all peripherals to default



Step 4. When in .ioc file, view the overall default settings



Step 5. Click GPIO and set PB2 pin as custom_led output push pull pin, we will use this pin to toggle external LED for demo 4

The screenshot shows the STM32CubeIDE Pinout & Configuration window. The 'Pinout & Configuration' tab is active, displaying a table of pins and their configurations. The 'PB2' pin is highlighted, and its configuration is shown in the 'PB2 Configuration' section below the table.

Pinout & Configuration

Group By Peripherals: Configuration

Search Signals: Search (Ctrl+F)

Pin Name | Signal on Pin | GPIO output le. | GPIO mode | GPIO Pull-up/ | Maximum outp. | Fast Mode | User Label | Modified

Pin Name	Signal on Pin	GPIO output le.	GPIO mode	GPIO Pull-up/	Maximum outp.	Fast Mode	User Label	Modified
PA2	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	ARD_D10 [S...	✓
PA8	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	SPBTLT_RF...	✓
PA15 (JTDI)	n/a	Low	Output Push P...	No pull-up and...	n/a	n/a	ARD_D9	✓
PB0	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	ARD_D3 [INT...	✓
PB2	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	custom_led	✓
PB4 (NJTRST)	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	ARD_D5	✓
PB5	n/a	High	Output Push P...	No pull-up and...	Low	n/a	SPSGRF_91...	✓
PB12	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	ISM43362_B...	✓
PB13	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	ISM43362_W...	✓
PB14	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	LED2 [LED...	✓
PB15	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	SPSGRF_91...	✓
PC6	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	VL53L0X_XS...	✓
PC7	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	VL53L0X_GP...	✓
PC8	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	LSM3MDL_D...	✓
PC9	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	LED3_WIFI...	✓
PC13	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	BUTTON_EX...	✓
PD0	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	PMOD_RESET	✓
PD2	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	PMOD_IRQ...	✓
PD7	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	STSAFE_A1...	✓
PD10	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	LPS22HB_IN...	✓
PD11	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	LSM6DSL_IN...	✓
PD12	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	USB_OTG_F...	✓
PD13	n/a	High	Output Push P...	No pull-up and...	Low	n/a	SPBTLT_RF...	✓
PD14	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	ARD_D2 [INT...	✓
PD15	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	HTS221_DR...	✓
PE0	n/a	High	Output Push P...	No pull-up and...	Low	n/a	ISM43362_S...	✓
PE1	n/a	n/a	External Interru...	No pull-up and...	n/a	n/a	ISM43362_D...	✓
PE2	n/a	Low	Output Push P...	No pull-up and...	Low	n/a	M24SR64_Y...	✓

PB2 Configuration:

GPIO output level: Low

GPIO mode: Output Push Pull

GPIO Pull-up/Pull-down: No pull-up and no pull-down

Maximum output speed: Low

User Label: custom_led

Pinout view

System view

Pinout view shows the physical pin connections for the STM32F407VGT6. The pins are color-coded and labeled with their functions. The PB2 pin is highlighted in green, indicating its configuration as a custom_led output push pull pin.

Step 6. Confirm that blue button has set up as external interrupt trigger

The left screenshot shows the STM32CubeMX Pinout & Configuration window. The 'Pinout & Configuration' tab is active. The 'GPIO Mode and Configuration' section is expanded, showing a table of pins and their configurations. The pin PC13 is highlighted in blue, indicating it is selected as an external interrupt. The configuration for PC13 is: Pin Name: PC13, Signal on Pin: n/a, GPIO output level: Low, GPIO mode: External Interrupt, GPIO Pull-up/Pull-down: No pull-up and no pull-down, Maximum output current: n/a, Fast Mode: n/a, User Label: BUTTON_EXTI13.

Pin Name	Signal on Pin	GPIO output level	GPIO mode	GPIO Pull-up/Pull-down	Maximum output current	Fast Mode	User Label	Modified
PA2	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	ARD_D10 (SPI)	✓
PA8	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	SPBTLT_RF_RST	✓
PA15 (JTDI)	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	ARD_D9	✓
PB0	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	ARD_D3 (INT_E)	✓
PB2	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	custom_led	✓
PB4 (NTRST)	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	ARD_D5	✓
PB5	n/a	High	Output Push Pull	No pull-up and no pull-down	Low	n/a	SPSGRF_915_	✓
PB12	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	ISM43362_BOO	✓
PB13	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	ISM43362_WAK	✓
PB14	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	LED2 (LED_GR)	✓
PB15	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	SPSGRF_915_	✓
PC6	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	VL53L0X_KSHU	✓
PC7	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	VL53L0X_GPIO	✓
PC8	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	LSM3MDL_DRD	✓
PC9	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	LED3_WIFI_LE	✓
PC13	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	BUTTON_EXTI13	✓
PD0	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	PMOD_RESET	✓
PD2	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	PMOD_IRQ_EX	✓
PD7	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	STSAFE_A100_	✓
PD10	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	LPS22HB_INT	✓
PD11	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	LSM6DSL_INT1	✓
PD12	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	USB_OTG_FS_	✓
PD13	n/a	High	Output Push Pull	No pull-up and no pull-down	Low	n/a	SPBTLT_RF_S	✓
PD14	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	ARD_D2 (INT0)	✓
PD15	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	HTS221_DRDY	✓
PE0	n/a	High	Output Push Pull	No pull-up and no pull-down	Low	n/a	ISM43362_SPI3	✓
PE1	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	ISM43362_DRD	✓
PE2	n/a	Low	Output Push Pull	No pull-up and no pull-down	Low	n/a	M24SR64_Y_RF	✓
PE3	n/a	n/a	External Interrupt	No pull-up and no pull-down	n/a	n/a	USB_OTG_FS_	✓

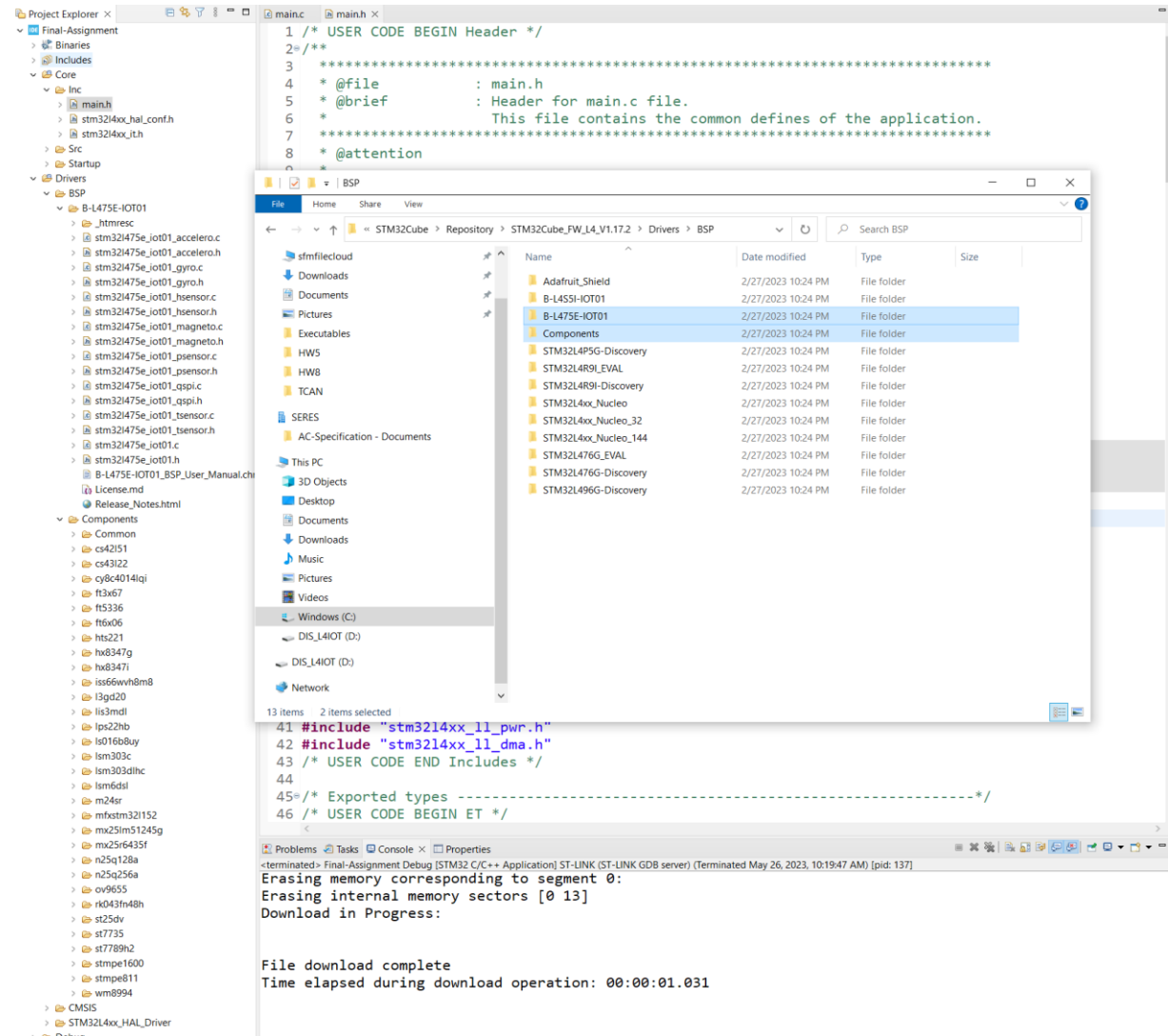
The right screenshot shows the same window, but the 'NVIC interrupt Table' is expanded. The table shows the configuration for the PC13 interrupt. The 'Enabled' column is checked, and the 'Preemption Priority' and 'Sub Priority' are both set to 0.

Interrupt	Enabled	Preemption Priority	Sub Priority
EXTI line0 interrupt	<input type="checkbox"/>	0	0
EXTI line1 interrupt	<input type="checkbox"/>	0	0
EXTI line2 interrupt	<input type="checkbox"/>	0	0
EXTI line3 interrupt	<input type="checkbox"/>	0	0
EXTI line[9-5] interrupts	<input checked="" type="checkbox"/>	0	0
EXTI line[15-10] interrupts	<input checked="" type="checkbox"/>	0	0

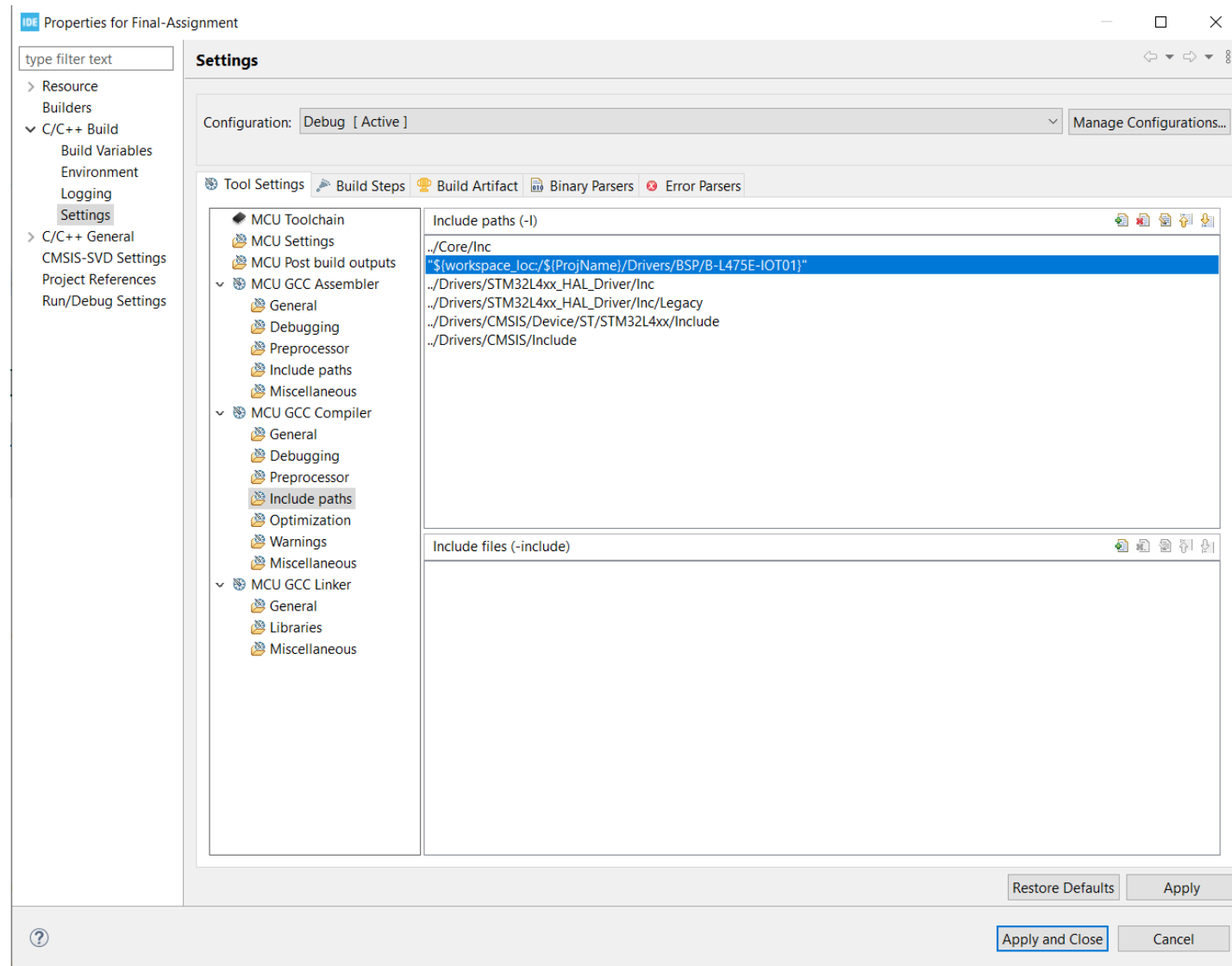
Step 7. Generate the code, and in main.h file, and the required low layer header file for use in demo 1

```
1 /* USER CODE BEGIN Header */
2 /**
3  *
4  * @file          : main.h
5  * @brief         : Header for main.c file.
6  *                : This file contains the common defines of the application.
7  *
8  * @attention
9  *
10 * Copyright (c) 2023 STMicroelectronics.
11 * All rights reserved.
12 *
13 * This software is licensed under terms that can be found in the LICENSE file
14 * in the root directory of this software component.
15 * If no LICENSE file comes with this software, it is provided AS-IS.
16 *
17 ****
18 */
19 /* USER CODE END Header */
20
21 /* Define to prevent recursive inclusion -----*/
22 #ifndef MAIN_H
23 #define MAIN_H
24
25 #ifdef __cplusplus
26 extern "C" {
27 #endif
28
29 /* Includes -----*/
30 #include "stm32l4xx_hal.h"
31
32 /* Private includes -----*/
33 /* USER CODE BEGIN Includes */
34 #include "stm32l4xx_ll_system.h"
35 #include "stm32l4xx_ll_gpio.h"
36 #include "stm32l4xx_ll_exti.h"
37 #include "stm32l4xx_ll_bus.h"
38 #include "stm32l4xx_ll_cortex.h"
39 #include "stm32l4xx_ll_rcc.h"
40 #include "stm32l4xx_ll_utils.h"
41 #include "stm32l4xx_ll_pwr.h"
42 #include "stm32l4xx_ll_dma.h"
43 /* USER CODE END Includes */
44
45 /* Exported types -----*/
46 /* USER CODE BEGIN ET */
```


Step 8. Create BSP folder under Drivers folder, and copy and paste the local Components and B-L475E-IOT01 folder to it. This is for using BSP package for temp sensor in demo 3



Step 9. Add the BSP folder to the include path



Step 10. In main.c file, add the required header files

```
main.c x main.h
1 /* USER CODE BEGIN Header */
2 /**
3  *
4  * @file      : main.c
5  * @brief     : Main program body
6  *
7  * @attention
8  *
9  * Copyright (c) 2023 STMicroelectronics.
10 * All rights reserved.
11 *
12 * This software is licensed under terms that can be found in the LICENSE file
13 * in the root directory of this software component.
14 * If no LICENSE file comes with this software, it is provided AS-IS.
15 *
16 *
17 */
18 /* USER CODE END Header */
19 /* Includes -----*/
20 #include "main.h"
21
22 /* Private includes -----*/
23 /* USER CODE BEGIN Includes */
24 #include <string.h>
25 #include <stdio.h>
26 #include <math.h>
27 #include "stm32l475e_iot01.h"
28 #include "stm32l475e_iot01_tsensor.h"
29 /* USER CODE END Includes */
30
31 /* Private typedef -----*/
32 /* USER CODE BEGIN PTD */
33
34 /* USER CODE END PTD */
35
36 /* Private define -----*/
37 /* USER CODE BEGIN PD */
38
39 /* USER CODE END PD */
40
41 /* Private macro -----*/
42 /* USER CODE BEGIN PM */
43
44 /* USER CODE END PM */
45
46 /* Private variables -----*/
47 DFSDM_Channel1_HandleTypeDef hdfsdm1_channel1;
48
49 I2C_HandleTypeDef hi2c2;
50
51 QSPI_HandleTypeDef hqspi;
52
53 SPI_HandleTypeDef hspi3;
54
55 UART_HandleTypeDef huart1;
56 UART_HandleTypeDef huart2;
```

Step 11. In main.c, add the needed variables

```
main.c x main.h
37 /* USER CODE BEGIN PD */
38
39 /* USER CODE END PD */
40
41 /* Private macro -----*/
42 /* USER CODE BEGIN PM */
43
44 /* USER CODE END PM */
45
46 /* Private variables -----*/
47 DFSDM_Channel_HandleTypeDef hdfsdm1_channel1;
48
49 I2C_HandleTypeDef hi2c2;
50
51 QSPI_HandleTypeDef hqspi;
52
53 SPI_HandleTypeDef hspi3;
54
55 UART_HandleTypeDef huart1;
56 UART_HandleTypeDef huart3;
57
58 PCD_HandleTypeDef hpcd_USB_OTG_FS;
59
60 /* USER CODE BEGIN PV */
61 uint8_t demo0_count = 0;
62 uint8_t demo1_count = 0;
63 uint8_t demo_count = 0;
64 char buf[100];
65 /* USER CODE END PV */
66
67 /* Private function prototypes -----*/
68 void SystemClock_Config(void);
69 static void MX_GPIO_Init(void);
70 static void MX_DFSDM1_Init(void);
71 static void MX_I2C2_Init(void);
72 static void MX_QUADSPI_Init(void);
73 static void MX_SPI3_Init(void);
74 static void MX_USART1_UART_Init(void);
75 static void MX_USART3_UART_Init(void);
76 static void MX_USB_OTG_FS_PCD_Init(void);
77 /* USER CODE BEGIN PFP */
78
79 /* USER CODE END PFP */
80
81 /* Private user code -----*/
82 /* USER CODE BEGIN 0 */
83
84 /* USER CODE END 0 */
85
86 /**
87  * @brief The application entry point.
88  * @retval int
89  */
90 int main(void)
91 {
92     /* USER CODE BEGIN 1 */
```

Step 12. Add the function definition for EXTI call back function

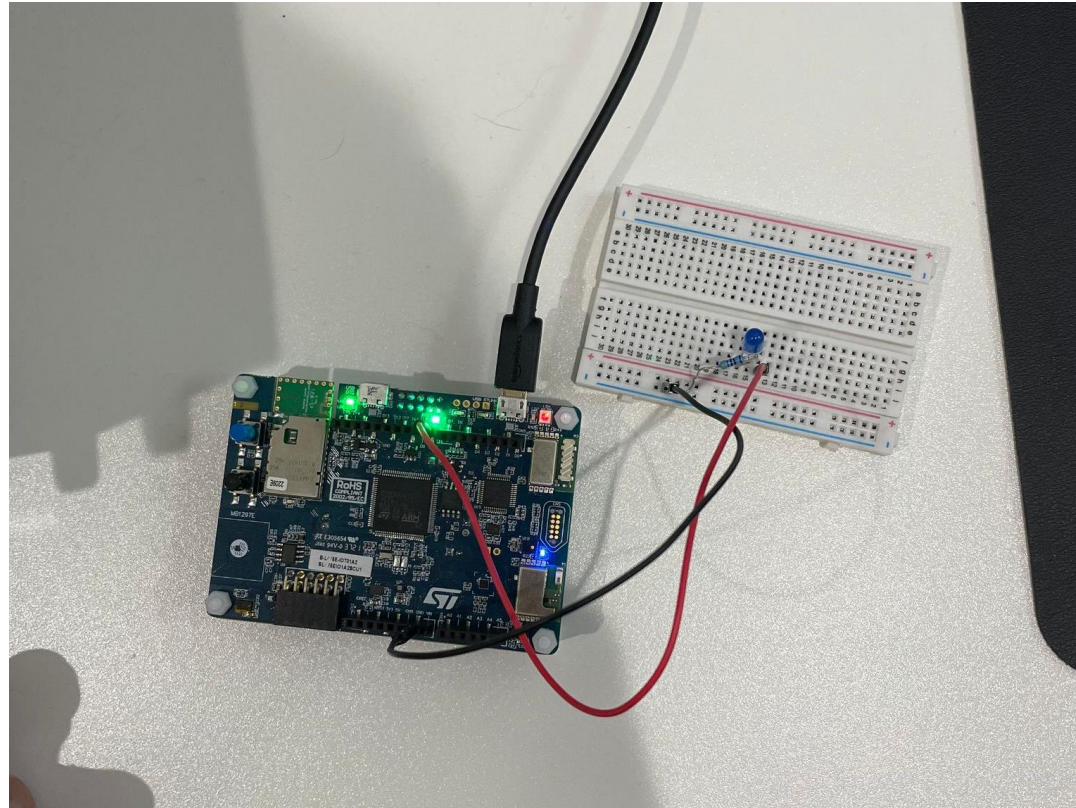
```
719 /* USER CODE BEGIN 4 */
720 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
721 {
722     if(GPIO_Pin == BUTTON_EXTI13_Pin)
723     {
724         demo_count = (demo_count + 1) % 4;
725         if(demo_count == 0){
726             demo0_count = 0;
727         }else if (demo_count == 1){
728             demo1_count = 0;
729         }
730         sprintf(buf, "UART1: current demo is: %d\r\n", demo_count + 1);
731         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
732     }
733 }
734 /* USER CODE END 4 */
735
736 /**
```

Step 13. Add the code for demo 1 to demo 4

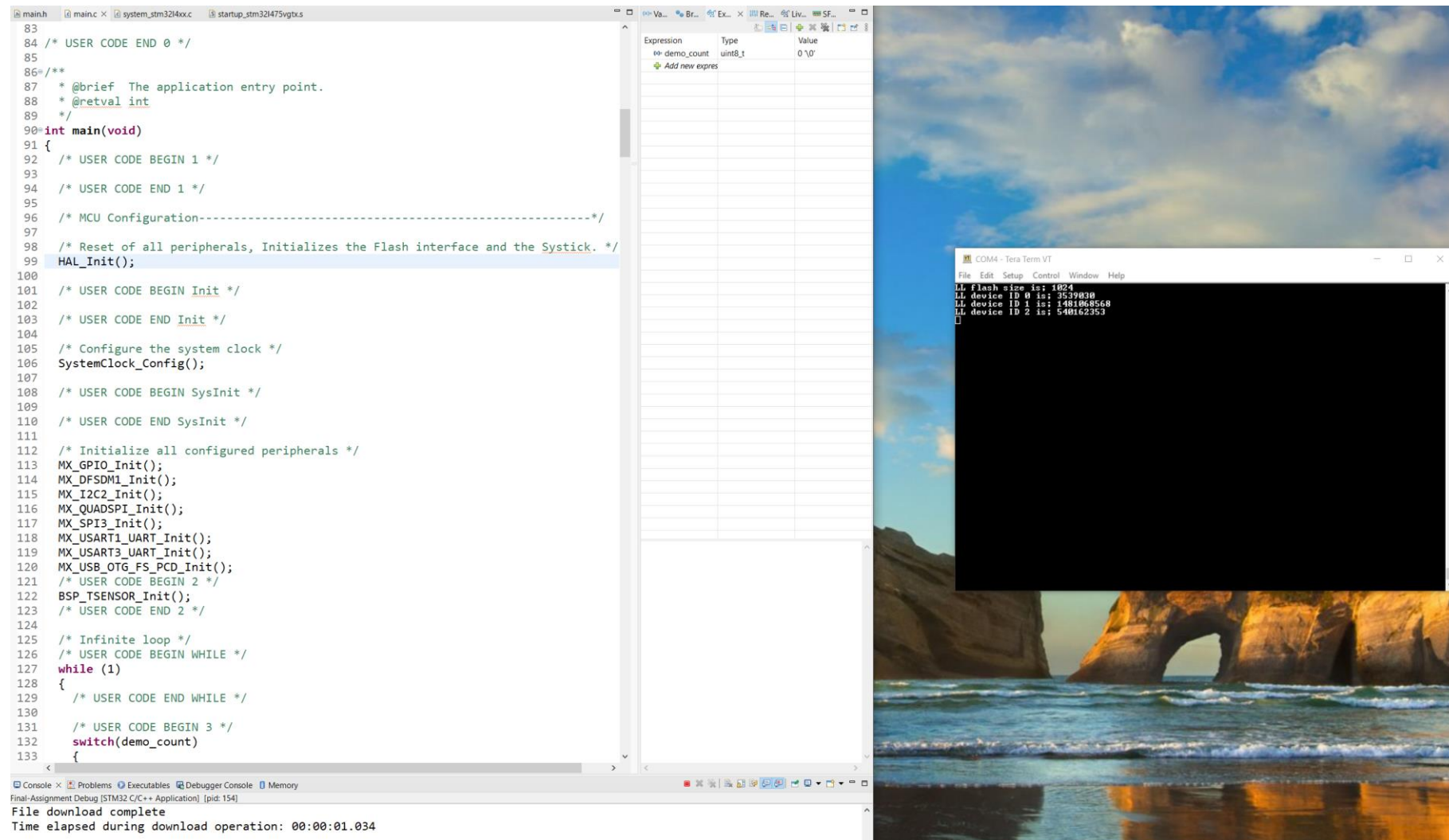
```
127 while (1)
128 {
129     /* USER CODE END WHILE */
130
131     /* USER CODE BEGIN 3 */
132     switch(demo_count)
133     {
134         case 0:
135         {
136             if(demo0_count == 0)
137             {
138                 uint32_t flashSize = LL_GetFlashSize();
139                 uint32_t deviceID0 = LL_GetUID_Word0();
140                 uint32_t deviceID1 = LL_GetUID_Word1();
141                 uint32_t deviceID2 = LL_GetUID_Word2();
142                 sprintf(buf, "LL flash size is; %lu\r\n", flashSize);
143                 HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
144                 sprintf(buf, "LL device ID 0 is; %lu\r\n", deviceID0);
145                 HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
146                 sprintf(buf, "LL device ID 1 is; %lu\r\n", deviceID1);
147                 HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
148                 sprintf(buf, "LL device ID 2 is; %lu\r\n", deviceID2);
149                 HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
150                 demo0_count++;
151             }
152             LL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin);
153             HAL_Delay(1000);
154             break;
155         }
156         case 1:
157         {
158             if(demo1_count == 0)
159             {
160                 uint32_t DeviceID = HAL_GetDEVID();
161                 uint32_t deviceUID0 = HAL_GetUIDw0();
162                 uint32_t deviceUID1 = HAL_GetUIDw1();
163                 uint32_t deviceUID2 = HAL_GetUIDw2();
164
165                 sprintf(buf, "HAL device ID is; %lu\r\n", DeviceID);
166                 HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
167                 sprintf(buf, "HAL unique device ID 0 is; %lu\r\n", deviceUID0);
168                 HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
169                 sprintf(buf, "HAL unique device ID 1 is; %lu\r\n", deviceUID1);
170                 HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
171                 sprintf(buf, "HAL unique device ID 2 is; %lu\r\n", deviceUID2);
172                 HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
173                 demo1_count++;
174             }
175             HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin);
176             HAL_Delay(2000);
177             break;
178         }
179     }
```

```
179         case 2:
180         {
181             float temp = BSP_TSENSOR_ReadTemp();
182             int tempInt1 = temp;
183             float tempFrac = temp - tempInt1;
184             int tempInt2 = trunc(tempFrac * 100);
185             sprintf(buf, "Temperature is : %d.%02d\r\n", tempInt1, tempInt2);
186             HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
187             BSP_LED_Toggle(LED2);
188             HAL_Delay(3000);
189             break;
190         }
191         case 3:
192         {
193             HAL_GPIO_TogglePin(custom_led_GPIO_Port, custom_led_Pin);
194             HAL_Delay(4000);
195             break;
196         }
197         default:
198             break;
199     }
200 }
201 /* USER CODE END 3 */
202 }
```

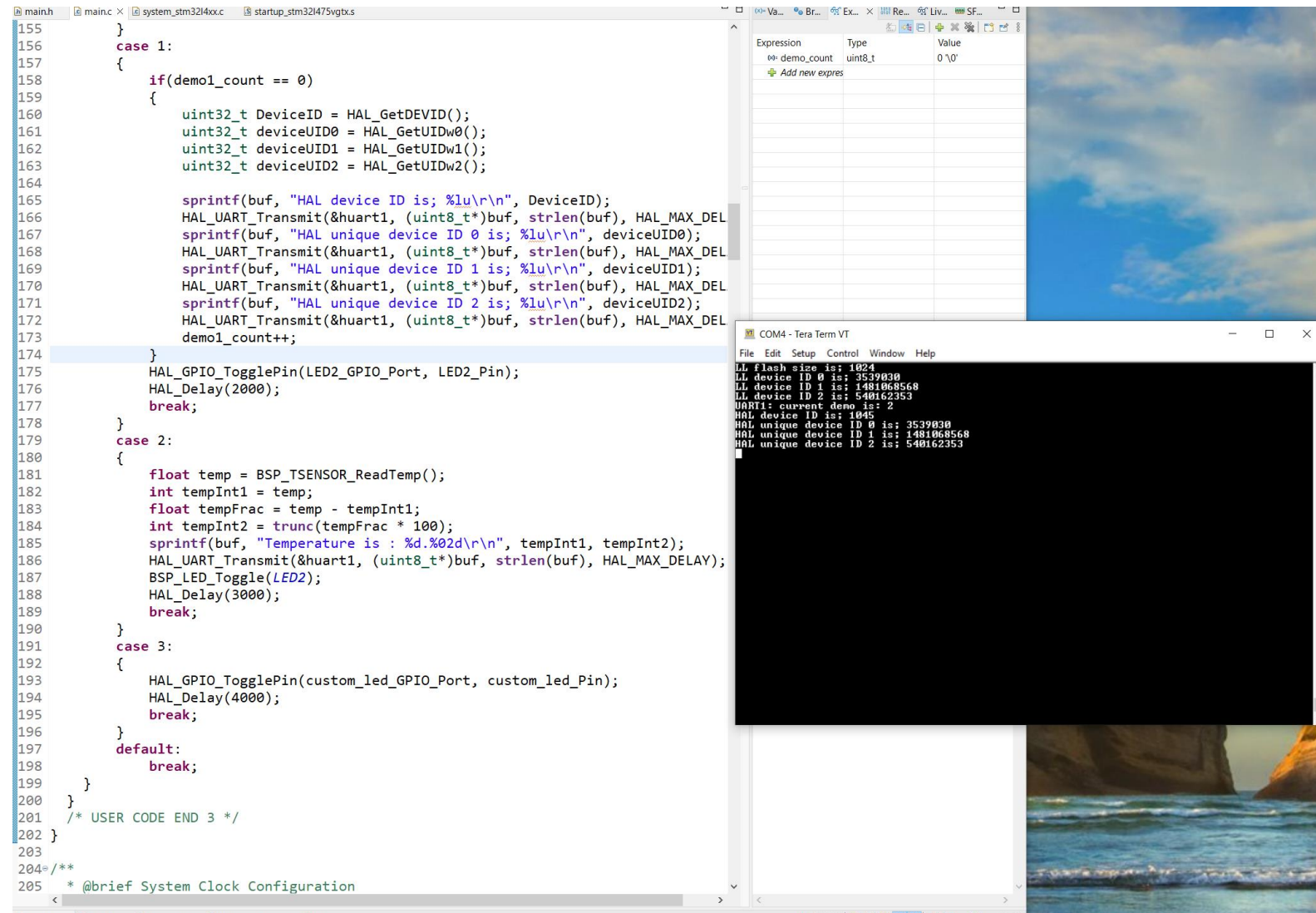
Step 14. Setup the wire connection for demo 4 LED blinking



Step 15. Compile and run the code in debug mode, open tera term and test from demo 1. test is successful, print the flash size and device ID once when entering demo 1 code, and continue blinking LED2 at 1 second interval



Step 16. Press the user button, we will see HAL device ID and unique ID only print once, and continue blinking LED for 2 seconds interval. Demo 2 success



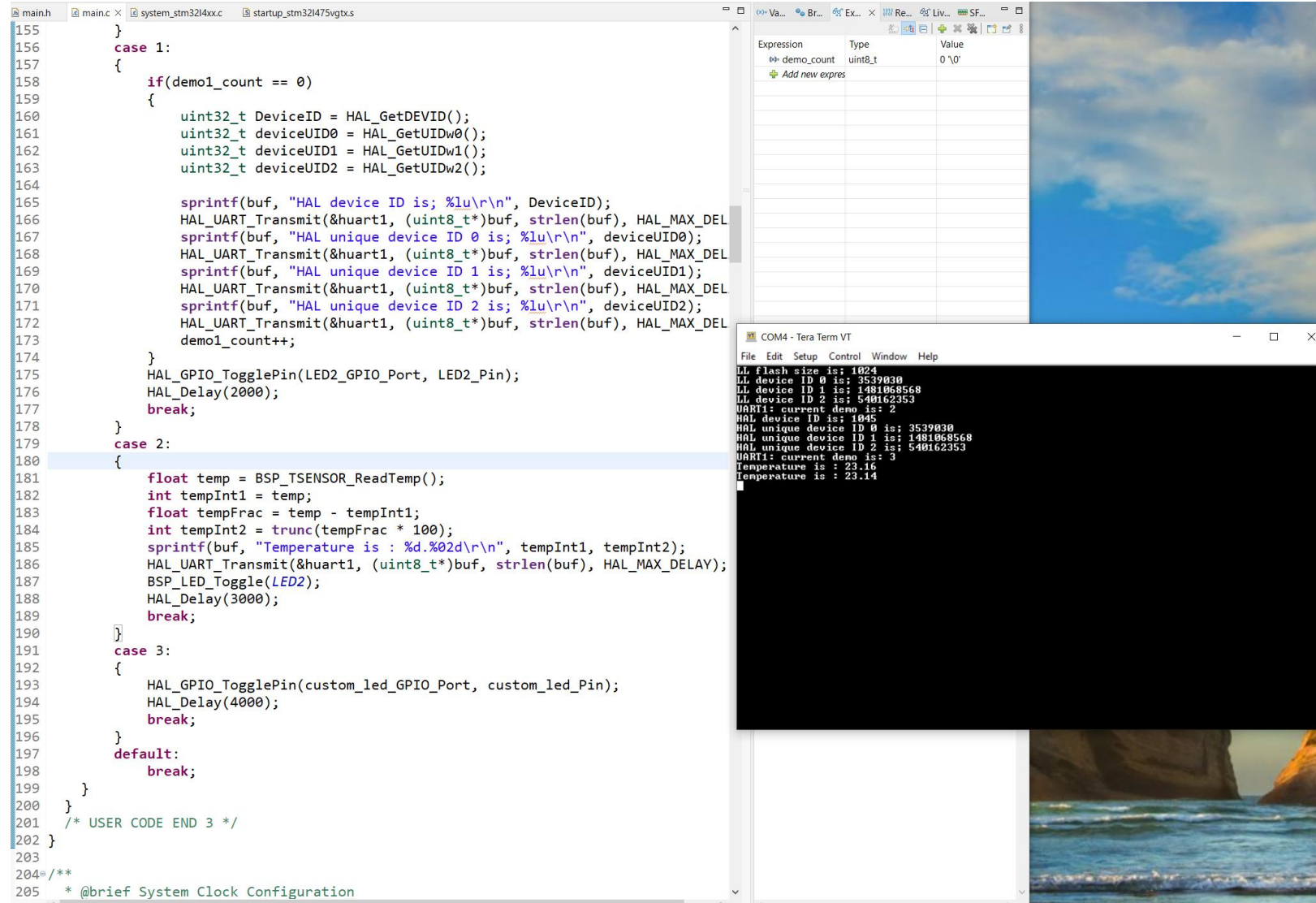
The screenshot displays a development environment with a C source file and a terminal window. The code is a switch-case statement with three cases. Case 1 prints device and unique IDs. Case 2 reads temperature and toggles an LED. Case 3 toggles a custom LED. The terminal shows the output of Case 1, displaying device and unique IDs for three different devices.

```
155 }
156 case 1:
157 {
158     if(demo1_count == 0)
159     {
160         uint32_t DeviceID = HAL_GetDEVID();
161         uint32_t deviceUID0 = HAL_GetUIDw0();
162         uint32_t deviceUID1 = HAL_GetUIDw1();
163         uint32_t deviceUID2 = HAL_GetUIDw2();
164
165         sprintf(buf, "HAL device ID is; %lu\r\n", DeviceID);
166         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DEL
167         sprintf(buf, "HAL unique device ID 0 is; %lu\r\n", deviceUID0);
168         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DEL
169         sprintf(buf, "HAL unique device ID 1 is; %lu\r\n", deviceUID1);
170         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DEL
171         sprintf(buf, "HAL unique device ID 2 is; %lu\r\n", deviceUID2);
172         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DEL
173         demo1_count++;
174     }
175     HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin);
176     HAL_Delay(2000);
177     break;
178 }
179 case 2:
180 {
181     float temp = BSP_TSENSOR_ReadTemp();
182     int tempInt1 = temp;
183     float tempFrac = temp - tempInt1;
184     int tempInt2 = trunc(tempFrac * 100);
185     sprintf(buf, "Temperature is : %d.%02d\r\n", tempInt1, tempInt2);
186     HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
187     BSP_LED_Toggle(LED2);
188     HAL_Delay(3000);
189     break;
190 }
191 case 3:
192 {
193     HAL_GPIO_TogglePin(custom_led_GPIO_Port, custom_led_Pin);
194     HAL_Delay(4000);
195     break;
196 }
197 default:
198     break;
199 }
200 }
201 /* USER CODE END 3 */
202 }
203
204 /**
205  * @brief System Clock Configuration
```

Expression	Type	Value
demo_count	uint8_t	0 \0

```
LL flash size is: 1024
LL device ID 0 is: 3539030
LL device ID 1 is: 1481068568
LL device ID 2 is: 540162353
UART1: current demo is: 2
HAL device ID is: 1045
HAL unique device ID 0 is: 3539030
HAL unique device ID 1 is: 1481068568
HAL unique device ID 2 is: 540162353
```

Step 17. Press the user button again, now we can read the temp value and send it through UART1.
Demo 3 success



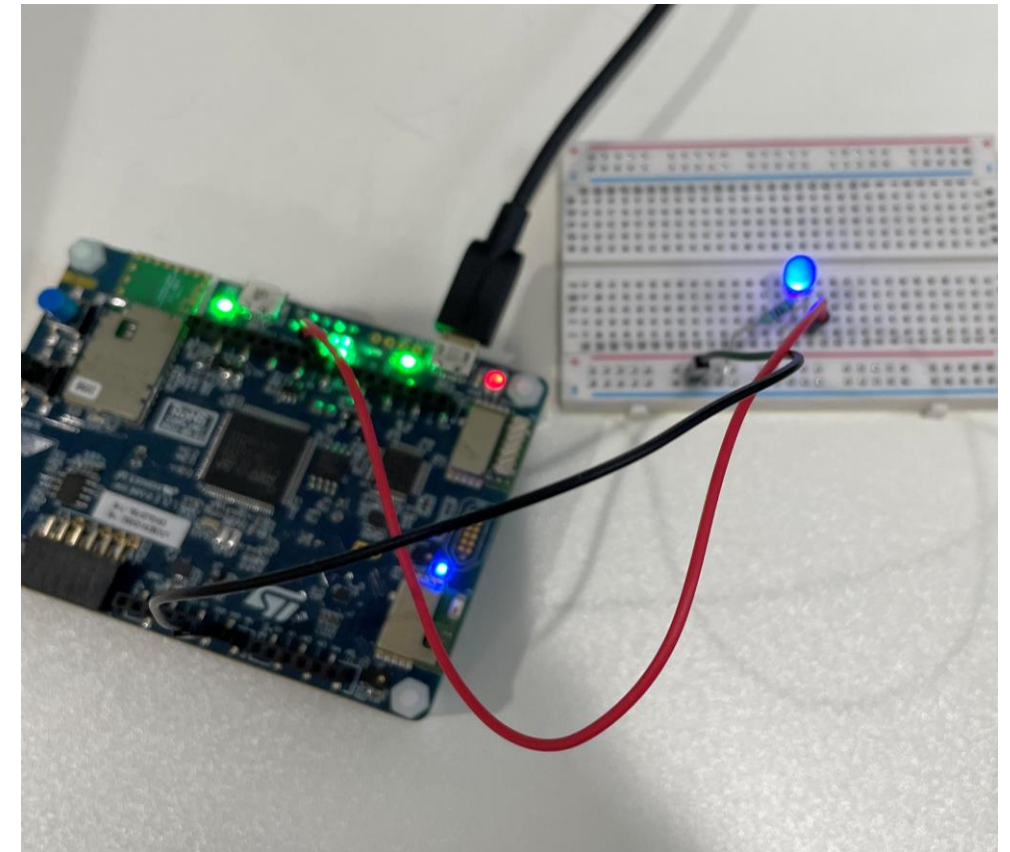
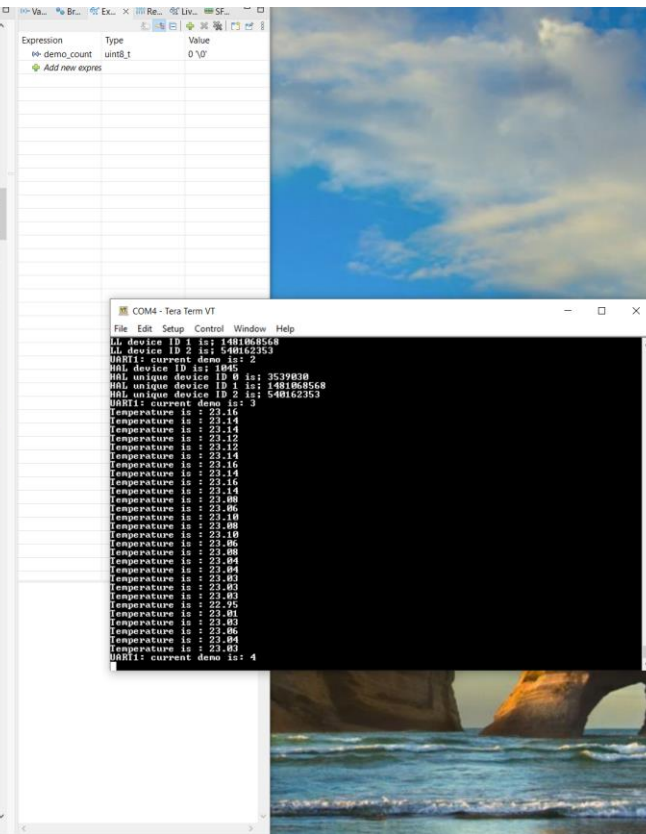
```
155 }
156 case 1:
157 {
158     if(demo1_count == 0)
159     {
160         uint32_t DeviceID = HAL_GetDEVID();
161         uint32_t deviceUID0 = HAL_GetUIDw0();
162         uint32_t deviceUID1 = HAL_GetUIDw1();
163         uint32_t deviceUID2 = HAL_GetUIDw2();
164
165         sprintf(buf, "HAL device ID is; %lu\r\n", DeviceID);
166         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
167         sprintf(buf, "HAL unique device ID 0 is; %lu\r\n", deviceUID0);
168         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
169         sprintf(buf, "HAL unique device ID 1 is; %lu\r\n", deviceUID1);
170         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
171         sprintf(buf, "HAL unique device ID 2 is; %lu\r\n", deviceUID2);
172         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
173         demo1_count++;
174     }
175     HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin);
176     HAL_Delay(2000);
177     break;
178 }
179 case 2:
180 {
181     float temp = BSP_TSSENSOR_ReadTemp();
182     int tempInt1 = temp;
183     float tempFrac = temp - tempInt1;
184     int tempInt2 = trunc(tempFrac * 100);
185     sprintf(buf, "Temperature is : %d.%02d\r\n", tempInt1, tempInt2);
186     HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
187     BSP_LED_Toggle(LED2);
188     HAL_Delay(3000);
189     break;
190 }
191 case 3:
192 {
193     HAL_GPIO_TogglePin(custom_led_GPIO_Port, custom_led_Pin);
194     HAL_Delay(4000);
195     break;
196 }
197 default:
198     break;
199 }
200 }
201 /* USER CODE END 3 */
202 }
203
204 /**
205 * @brief System Clock Configuration
```

Expression	Type	Value
demo_count	uint8_t	0 \0
Add new expres		

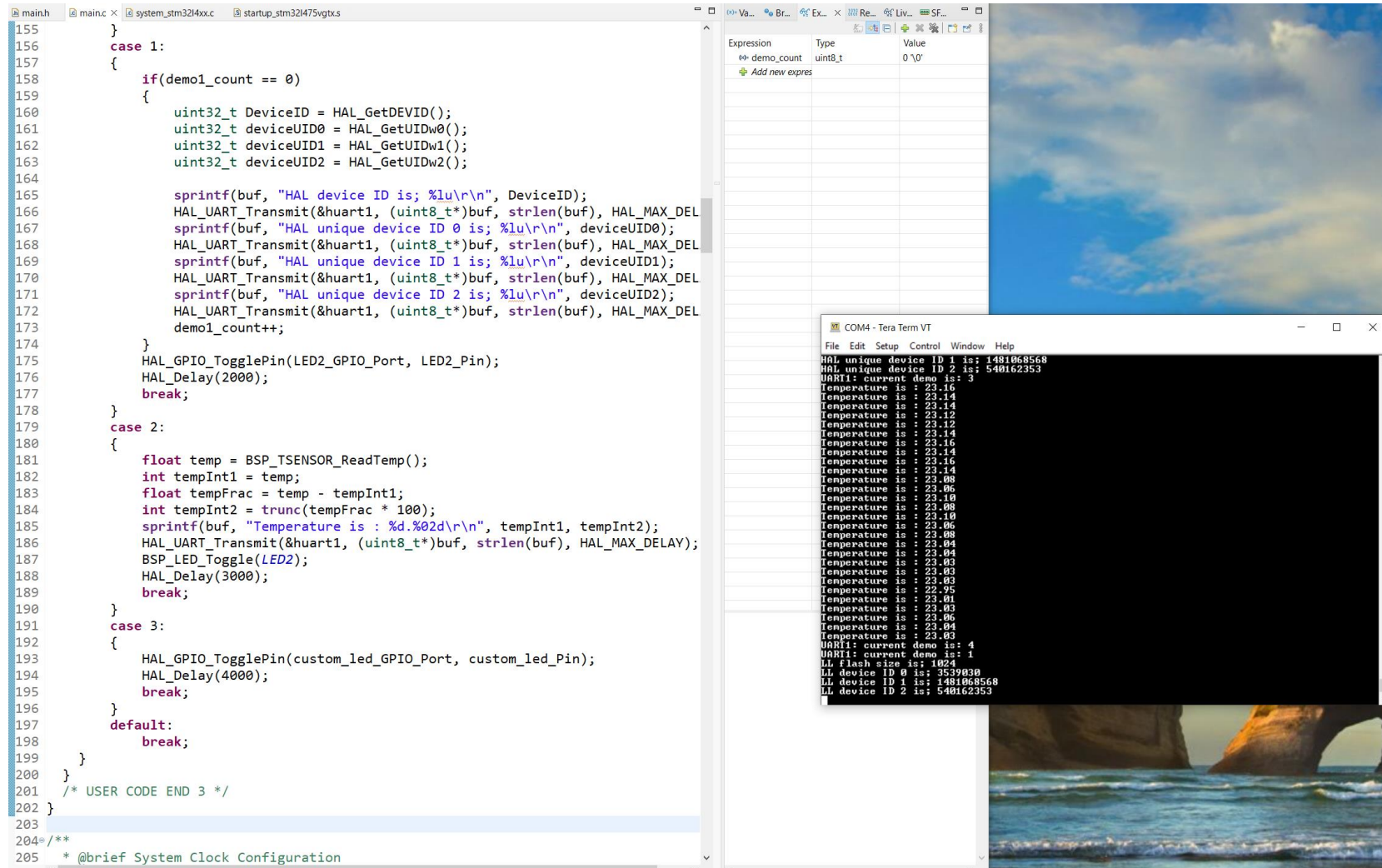
```
COM4 - Tera Term VT
File Edit Setup Control Window Help
LL flash size is: 1024
LL device ID 0 is: 3539030
LL device ID 1 is: 1481068560
LL device ID 2 is: 540162353
UART1: current demo is: 2
HAL device ID is: 1045
HAL unique device ID 0 is: 3539030
HAL unique device ID 1 is: 1481068560
HAL unique device ID 2 is: 540162353
UART1: current demo is: 3
Temperature is : 23.16
Temperature is : 23.14
```

Step 18. Press the user button again, we can see the external LED blinking at 4 seconds interval.
Demo 4 success

```
R:\mainh  @ mainc x  @ system_uart3244x.c  @ startup_uart3244x5y6x
155 }
156 case 1:
157 {
158     if(demo1_count == 0)
159     {
160         uint32_t DeviceID = HAL_GetDeviceID();
161         uint32_t deviceUID0 = HAL_GetUIDw0();
162         uint32_t deviceUID1 = HAL_GetUIDw1();
163         uint32_t deviceUID2 = HAL_GetUIDw2();
164
165         sprintf(buf, "HAL device ID is: %lu\r\n", DeviceID);
166         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
167         sprintf(buf, "HAL unique device ID 0 is: %lu\r\n", deviceUID0);
168         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
169         sprintf(buf, "HAL unique device ID 1 is: %lu\r\n", deviceUID1);
170         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
171         sprintf(buf, "HAL unique device ID 2 is: %lu\r\n", deviceUID2);
172         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
173         demo1_count++;
174     }
175     HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin);
176     HAL_Delay(2000);
177     break;
178 }
179 case 2:
180 {
181     float temp = BSP_TSENSOR_ReadTemp();
182     int tempInt1 = temp;
183     float tempFrac = temp - tempInt1;
184     int tempInt2 = trunc(tempFrac * 100);
185     sprintf(buf, "Temperature is : %.02d\r\n", tempInt1, tempInt2);
186     HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
187     BSP_LED_Toggle(LED2);
188     HAL_Delay(3000);
189     break;
190 }
191 case 3:
192 {
193     HAL_GPIO_TogglePin(custom_led_GPIO_Port, custom_led_Pin);
194     HAL_Delay(4000);
195     break;
196 }
197 default:
198     break;
199 }
200 }
201 /* USER CODE END 3 */
202 }
203
204 /**
205  * @brief System Clock Configuration
```



Step 19. Press the user button again, it will loop back to demo 1



The screenshot displays an IDE with a C source file and a terminal window. The code is a switch-case statement with three cases. Case 1 handles device ID printing and a demo counter. Case 2 reads a temperature sensor and prints the value. Case 3 toggles a custom LED. The terminal shows the output of the program, including device IDs, temperature readings, and demo counts.

```
155 }
156 case 1:
157 {
158     if(demo1_count == 0)
159     {
160         uint32_t DeviceID = HAL_GetDEVID();
161         uint32_t deviceUID0 = HAL_GetUIDw0();
162         uint32_t deviceUID1 = HAL_GetUIDw1();
163         uint32_t deviceUID2 = HAL_GetUIDw2();
164
165         sprintf(buf, "HAL device ID is: %lu\r\n", DeviceID);
166         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
167         sprintf(buf, "HAL unique device ID 0 is: %lu\r\n", deviceUID0);
168         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
169         sprintf(buf, "HAL unique device ID 1 is: %lu\r\n", deviceUID1);
170         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
171         sprintf(buf, "HAL unique device ID 2 is: %lu\r\n", deviceUID2);
172         HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
173         demo1_count++;
174     }
175     HAL_GPIO_TogglePin(LED2_GPIO_Port, LED2_Pin);
176     HAL_Delay(2000);
177     break;
178 }
179 case 2:
180 {
181     float temp = BSP_TSSENSOR_ReadTemp();
182     int tempInt1 = temp;
183     float tempFrac = temp - tempInt1;
184     int tempInt2 = trunc(tempFrac * 100);
185     sprintf(buf, "Temperature is : %d.%02d\r\n", tempInt1, tempInt2);
186     HAL_UART_Transmit(&huart1, (uint8_t*)buf, strlen(buf), HAL_MAX_DELAY);
187     BSP_LED_Toggle(LED2);
188     HAL_Delay(3000);
189     break;
190 }
191 case 3:
192 {
193     HAL_GPIO_TogglePin(custom_led_GPIO_Port, custom_led_Pin);
194     HAL_Delay(4000);
195     break;
196 }
197 default:
198     break;
199 }
200 }
201 /* USER CODE END 3 */
202 }
203
204 /**
205  * @brief System Clock Configuration
```

COM4 - Tera Term VT

```
HAL unique device ID 1 is: 1481068568
HAL unique device ID 2 is: 540162353
UART1: current demo is: 3
Temperature is : 23.16
Temperature is : 23.14
Temperature is : 23.14
Temperature is : 23.12
Temperature is : 23.12
Temperature is : 23.14
Temperature is : 23.16
Temperature is : 23.14
Temperature is : 23.16
Temperature is : 23.08
Temperature is : 23.06
Temperature is : 23.10
Temperature is : 23.08
Temperature is : 23.10
Temperature is : 23.06
Temperature is : 23.08
Temperature is : 23.04
Temperature is : 23.04
Temperature is : 23.03
Temperature is : 23.03
Temperature is : 23.03
Temperature is : 22.95
Temperature is : 23.01
Temperature is : 23.03
Temperature is : 23.06
Temperature is : 23.04
Temperature is : 23.03
UART1: current demo is: 4
UART1: current demo is: 1
LL flash size is: 1024
LL device ID 0 is: 3539030
LL device ID 1 is: 1481068568
LL device ID 2 is: 540162353
```