Abhishek Kumar Simra
PRN:-220940130002

Program: To create two Tasks and Two two Queue
-The task one will accept data from interrupt using first Queue in form of count of interrupts
-The task two will accept data from task 2 using second Queuein form of counts when count becomes a multiple of 5 the led should toggle

SOLUTIONS:

```
xTaskCreate( task1, "Task1", 200, NULL, 1, NULL );
xTaskCreate( task2, "Task2", 200, NULL, 2, NULL );

xQueue1 = xQueueCreate( QUEUE_LENGTH, QUEUE_ITEM_SIZE );
xQueue2 = xQueueCreate( QUEUE_LENGTH, QUEUE_ITEM_SIZE );

if( xQueue1 != NULL )
  {

  }

if( xQueue2 != NULL )
  {

  }
//xTaskCreate( task3, "Task3", 200, NULL, 3, &xTask3 );

vTaskStartScheduler();
```

The above code shows the creation of Tasks and creation of Queue.

```
286
287 void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
288 {
289     xHigherPriorityTaskWoken = pdFALSE;
290      if(GPIO_Pin=GPIO_PIN_0){
291         {
292             for(int i=2500;i;i--);
293             if(HAL_GPIO_ReadPin(GPIOA, GPIO_PIN_0)){
294                 count1++;
295                 xStatus1 = xQueueSendFromISR(xQueue1, &count1, &xHigherPriorityTaskWoken );
296                 HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_12);
297                 portYIELD_FROM_ISR(xHigherPriorityTaskWoken);
298             }
299         }
300     }
301 }
302
```

Interrupt callback function which count the number of interrupts generrated

The above code shows the interrupt callback from the GPIO Pin PA0, When interrupts c press of the switch the external interupt is occured which initiate the interupt function is generated it send data to the Queue 1which is received to the task 1 on the other ha is used to suppress the debouncing delay fron the switch. In message I am sending inc with every interrupt to the Queue.

```
88
89  void task1(void *ptr1){
90      while(1){
91          xStatus1 = xQueueReceive(xQueue1, &lReceivedValue1, portMAX_DELAY );
92
93                  if( xStatus1 == pdPASS ) {
94
95                      count2++;
96
97                      //lValueToSend2 = (int32_t)count2;
98
99                      xStatus2 = xQueueSendToBack( xQueue2, &count2, 0 );
00                      //      vTaskDelay(1);
01                              if( xStatus2 != pdPASS ) {
02
03                              }
04                              else{
05                                  HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_13);
06
07                              }
08                  }
09
10      }
11  }
```

message Success checked if true count2 incremmented and value of count2 is send as data to Task2 from Queue2

When the data is send in the Queue from the interrupts the Task1 which is the receiver for the Queue1 initiates and accepts the message which is the the form of count. if the massage received is success full then we incremment the count2 and send the value of count 2 as a message from the Queue2 to Task2, and for the successfull message transfer we toggle the led.

```
13  void task2(void *ptr2){
14      while(1){
15
16          xStatus2 = xQueueReceive( xQueue2, &lReceivedValue2, portMAX_DELAY );
17
18          if( xStatus2 == pdPASS ) {
19              HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_14);
20
21                      if(count2%5==0){
22                              HAL_GPIO_TogglePin(GPIOD, GPIO_PIN_15);
23                      }
24          }
25          else {
26
27          }
28
29      }
30  }
31
```

on success ful receive of data the condition is checked for the multiple of 5 and if true led is toggeled

When the data is send to the Queue2 from Task1, the Task2 read that data, and if the Read is success full the condition is checked for the multiple of 5 using if and if the condition is success the led toggle take place

As per problem on the 5 count of the interrupt the led should toggle it self.