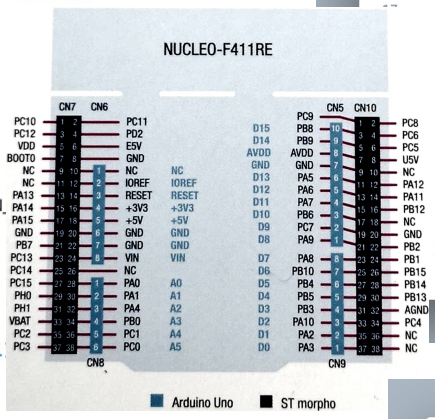
```c
// ---------- Lab3 - 01 ----------

// GPIO -> PC13: GPIO Mode: Rising
// NVIC -> 2bit -> EXTI interupt enable priority 2

/* USER CODE BEGIN 0 */
uint16_t blinktime = 200;
void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin) {
  HAL_Delay(100);
  while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET);
  if (blinktime == 200) blinktime = 1000;
  else if (blinktime == 1000) blinktime = 5000;
  else blinktime = 200;
  __HAL_GPIO_EXTI_CLEAR_IT(GPIO_Pin);
}
/* USER CODE END 0 */
/* Infinite loop */
/* USER CODE BEGIN WHILE */
while (1)
{
    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_
    HAL_Delay(blinkTime);
}

// ---------- Lab3 - 02 ----------

Desire Time Freq = SysClock / ( (PSC+1

// ---------- Lab3 - 03 ----------

// UART Interupt Timer
// ClkConfig: HSE -> 100 MHz
// Timers: Tim2 Tim3 Tim4 -> ClkSource: Internal
// NVIC: Tim2 Tim3 Tim4 global interupt enable
// adjust PSC&CP: Period 500ms: 4999 9999, 490.5ms: 4904 9999, 999.9ms: 9998 9

/* USER CODE BEGIN 0 */
uint8_t green_cnt = 0; // PA5
uint8_t red_cnt = 0; // PA6
// Timer interrupt handlers (controller)
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef* htim){
    if (htim->Instance == TIM2){
        HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
        green_cnt++;
    }
    else if (htim->Instance == TIM3){
        HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_6);
        red_cnt++;
    }
    else if (htim->Instance == TIM4){
//      HAL_UART_Transmit(&huart2, &green_cnt, sizeof(green_cnt), HAL_MAX_DELA
//      HAL_UART_Transmit(&huart2, &red_cnt, sizeof(red_cnt), HAL_MAX_DELAY);
        char green_cnt_str[10];
        char red_cnt_str[10];
//      sprintf(green_cnt_str, "%d", green_cnt/2);
//      sprintf(rS, "%d", red_cnt/2);
        itoa(green_cnt/2, green_cnt_str, 10); // Convert green_cnt to ASCII st
        itoa(red_cnt/2, red_cnt_str, 10);   // Convert red_cnt to ASCII string
        HAL_UART_Transmit(&huart2, green_cnt_str, strlen(green_cnt_str), 10);
        HAL_UART_Transmit(&huart2," " , 1, 10);
        HAL_UART_Transmit(&huart2, red_cnt_str, strlen(red_cnt_str), 10);
        HAL_UART_Transmit(&huart2,"\r\n" , 2, 10); // \r: cursor to begin of l
    }
}
/* USER CODE END 0 */
/* USER CODE BEGIN 2 */
  // Start timers
      HAL_TIM_Base_Start_IT(&htim2);
      HAL_TIM_Base_Start_IT(&htim3);
      HAL_TIM_Base_Start_IT(&htim4);
/* USER CODE END 2 */


// ---------- Lab3 - 04 ----------
// echo back (transmit the receive data) USART2 interrupt
// NVIC -> UART2 global interupt prio2

/* USER CODE BEGIN 2 */
      HAL_UART_Receive_IT(&huart2, &ch, 1);
/* USER CODE END 2 */
/* USER CODE BEGIN 0 */
char ch;
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart){
    HAL_UART_Transmit(&huart2, &ch, 1, 1000);
    HAL_UART_Receive_IT(&huart2, &ch, 1);
}
/* USER CODE END 0 */
```




$f_c = SysClk / [(PSC+1)*(CP+1)]$

```c
// Real Skill Test 1 sec 1 2024
// Print "6532142421" individual by pressing blue button

/* USER CODE BEGIN 2 */
char sequence[10] = "6532142421"; // array of char size 10
uint8_t index = 0;
uint8_t digit;
/* USER CODE END 2 */
while (1)
{
  if (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
  {
    HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5); // Show status
    // Debounce delay
    HAL_Delay(100);
    while (HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13) == GPIO_PIN_RESET)
      ;

    // char digit[2];
    digit = sequence[index];
    // digit[1] = '\0';
    HAL_UART_Transmit(&huart2, &digit, 1, 1000); // Print the digit over UART
                                                 // Print newline
    if (index == 9)
      HAL_UART_Transmit(&huart2, (uint8_t *)"\n", 1, HAL_MAX_DELAY);

    // Increment index and loop back to 0 after reaching the end
    index = (index + 1) % sizeof(sequence);
}
```

```c
// ---------- skill test 2566 01 ----------

/* USER CODE BEGIN 0 */
char input[32];
int index = 0;
char c;
void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    if (htim->Instance == TIM3)
    {
        HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
    }
}
/* USER CODE END 0 */
/* USER CODE BEGIN 2 */
HAL_TIM_Base_Start_IT(&htim3);
/* USER CODE END 2 */
while (1)
{
    if (HAL_UART_Receive(&huart2, (uint8_t *)&c, 1, HAL_MAX_DELAY) == HAL_OK)
    {
        HAL_UART_Transmit(&huart2, (uint8_t *)&c, 1, HAL_MAX_DELAY);

        if (c == '\r' || c == '\n')
        {
            HAL_UART_Transmit(&huart2, (uint8_t *)"\r\n", 2, HAL_MAX_DELAY);
            for (int i = index - 1; i >= 0; i--)
            {
                HAL_UART_Transmit(&huart2, (uint8_t *)&input[i], 1, HAL_MAX_DELAY);
            }
            HAL_UART_Transmit(&huart2, (uint8_t *)"\r\n", 2, HAL_MAX_DELAY);
            index = 0;
        }
        else
        {
            input[index++] = c;
            if (index >= sizeof(input))
            {
                index = 0;
            }
        }
    }
}

// ---------- skill test 2566 02 ----------

/* USER CODE BEGIN 0 */
int density = 0;
int timer_counter = 0;

void HAL_GPIO_EXTI_Callback(uint16_t GPIO_Pin)
{
    HAL_Delay(100);
    density += 10;
    density = density % 100;
    __HAL_GPIO_EXTI_CLEAR_IT(GPIO_Pin);
}

void HAL_TIM_PeriodElapsedCallback(TIM_HandleTypeDef *htim)
{
    if (htim->Instance == TIM2)
    {
        timer_counter++;
        if (timer_counter == 1000)
        { // 10 seconds have elapsed
            timer_counter = 0;
            density = 0; // Reset the density variable
        }
    }
}
/* USER CODE END 0 */
/* USER CODE BEGIN 2 */
HAL_TIM_PWM_Start(&htim2, TIM_CHANNEL_1);
TIM2->CCR1 = 50;
/* USER CODE END 2 */
while (1)
{
    TIM2->CCR1 = density;
}
```

```c
// ---------- Lab4 - 01 ----------
// DIM an LED by increasing the duty cycle from 0% duty cycle
// by 1% every 0.01 second. When the PWM reaches the 100% duty cycle,
// decrease the duty cycle by 1% every 0.01 second to 0% duty cycle.
// Repeat the following step forever. (use 100 microseconds period for P

// CLKconfig -> TIM2 -> CLKsource: Internal -> PWM CH1 -> Adjust PSC&CP
// -> Auto-reload preload: Enable -> PWM Generation Channel 1 -> Adj Pul

/* USER CODE BEGIN 2 */
HAL_TIM_PWM_Start(&htim2, TIM_CHANNEL_1);
TIM2->CCR1 = 50;
// TIMx->CNT - Clock counter // TIMx->PSC - Prescaler values
// TIMx->ARR - Period values // TIMx->CCR1 - PWM for channel 1
/* USER CODE END 2 */
while (1)
{
    for (int i = 0; i < 100; i++)
    {
        TIM2->CCR1 = i;
        HAL_Delay(10);
    }
    for (int i = 100; i >= 0; i--)
    {
        TIM2->CCR1 = i;
        HAL_Delay(10);
    }
}

// ---------- Lab4 - 02 ----------
// Connect LDR read the value from LDR and display serial terminal via U

// 2.1 Setup a PIN to accept an analog input: IOC -> Analog
// pin in INx // ADC1_IN0 is corresponding to PIN PA0.

/* USER CODE BEGIN 2 */
int adcval = 0;
char buf[256];
/* USER CODE END 2 */
while (1)
{
    HAL_ADC_Start(&hadc1);
    // Wait for 1000ms or when the conversion is finished.
    if (HAL_ADC_PollForConversion(&hadc1, 1000) == HAL_OK)
    {
        // Read the ADC value
        adcval = HAL_ADC_GetValue(&hadc1);
        // Write integer to buffer
        sprintf(buf, "%d\r\n", adcval);
        // Transmitted with UART
        HAL_UART_Transmit(&huart2, buf, strlen(buf), 1000);
    }
}

// ---------- Lab4 - 03 ----------
// green LED brightness with PWM from the value of LDR.
// PWM Duty Cycle = ( 1 - (C / P) ) * 100 %
// C = (max environment brightness) - (current environment brightness)
// P = (max environment brightness) - (min environment brightness)

/* Initialize all configured peripherals */
MX_GPIO_Init();
MX_USART2_UART_Init();
MX_ADC1_Init();
MX_TIM2_Init();
/* USER CODE BEGIN 2 */
HAL_TIM_PWM_Start(&htim2, TIM_CHANNEL_1);
TIM2->CCR1 = 0;
int adcval = 0;
double c = 700, p = 150;
char buf[256];
/* USER CODE END 2 */
while (1)
{
    HAL_ADC_Start(&hadc1);
    if (HAL_ADC_PollForConversion(&hadc1, 1000) == HAL_OK)
    {
        adcval = HAL_ADC_GetValue(&hadc1);
        c = 700 - adcval;
        p = 700 - 150;
        int b = (1 - (c / p)) * 100;
        TIM2->CCR1 = b;
        sprintf(buf, "%d\r\n", b);
        HAL_UART_Transmit(&huart2, buf, strlen(buf), 1000);
    }
}
```

```c
// Lab 5.1 Two UARTs communicating
// UART2's TX sends to UART1's RX (wire PA2(TX) to resistor to PA10(RX))
// check UART1 global interrupt in NVIC
void HAL_UART_RxCpltCallback(UART_HandleTypeDef *huart) {
    if(huart == &huart1) {
        HAL_UART_Transmit(&huart2, "b", 1, 100);
        HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
    }
}
char c[1];
while (1) {
    if(!HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)) {
        HAL_UART_Transmit(&huart2, "a", 1, 100);
        HAL_Delay(50);
        HAL_UART_Receive_IT(&huart1, c, 1); //must be _IT (interrupt)
        while(!HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)) {}
    }
}

// Lab 5.2 Two SPIs communicating
// SPI2 Transmit Only Master, SPI3 Receive Only Slave + global interrupt
// SPI2 sends to SPI3 (wire MOSI(PC3, PC12), CLK(PB12, PB10) to resistor to each other)
void HAL_SPI_RxCpltCallback(SPI_HandleTypeDef *hspi) {
    if(hspi == &hspi3) {
        HAL_UART_Transmit(&huart2, "b", 1, 100);
        HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
    }
}
char c;
while (1) {
    if(!HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)) {
        HAL_SPI_Transmit(&hspi2, "a", 1, 100);
        HAL_Delay(50);
        HAL_SPI_Receive_IT(&hspi3, &c, 1); //must be _IT (interrupt)
        while(!HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)) {}
    }
}

// Lab 5.3 Two I2Cs communicating
// I2C1 event, error interrupt + (optional: can change slave addr from 0,
// but have to shift 1 bits i.e. uint16_t slaveADDR = 0x12<<1;)
// GPIO > I2C > both I2C1_SDA, I2C1_SCL GPIO Pull-up
// I2C2 no need to customize, it is master
// I2C2 sends to I2C1 (wire SDA(PB9, PB7), SCL(PB6, PB10) to resistor to each other)
void HAL_I2C_SlaveRxCpltCallback(I2C_HandleTypeDef *hi2c) {
    if(hi2c == &hi2c1) {
        HAL_UART_Transmit(&huart2, "b", 1, 100);
        HAL_GPIO_TogglePin(GPIOA, GPIO_PIN_5);
    }
}
char c;
while (1) {
    if(!HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)) {
        HAL_UART_Transmit(&huart2, "a", 1, 100);
//          HAL_I2C_Master_Transmit(&hi2c2, slaveADDR, "a", 1, 100);
        HAL_I2C_Master_Transmit(&hi2c2, 0, "a", 1, 100);
        HAL_Delay(50);
        HAL_I2C_Slave_Receive_IT(&hi2c1, &c, 1); //must be _IT (interrupt)
        while(!HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_13)) {}
    }
}
```

Cmd: Ctrl A then Ctrl \

```
screen /dev/tty.usbmodem1203 115200, onclr
```