Lab1 ~ Lab5 共用規定

- 上課時間: 14:10~17:00；地點 @新館一樓 65105
- 每一個 lab 最晚都會在上課當天中午12:00前上傳投影片到 moodle，

  為避免教室網路訊號不好，請同學在14:00上課前先下載投影片至電腦中。
- 每一個 lab 佔總分 8%, 獨立計分. (Final Project 佔總分 60%)
- Lab 完成後, 要在 7 天內寫好 lab report 上傳moodle。
- 要來上 lab 課簽到，我們才會為你的 lab 成果評分。
- 若 lab 下課前有做完，我們會現場幫你評分。
- 若 lab 下課前沒做完，會有補交機制 (各 lab 規定方式可能不同)，

  期限內有完成就不會扣分 (期限為 7 天內，超過不計分)。

Lab2 規定

- Lab2 補交機制 (各 lab 規定方式可能不同)

  本次lab不開放補交

- 寫 lab report (上傳moodle)

Lab 2: TaskMonitor

OS lab email : oslab@mail.csie.ncku.edu.tw
TA: 鄭宇辰

OUTLINE

01 Lab requirement

02 UART

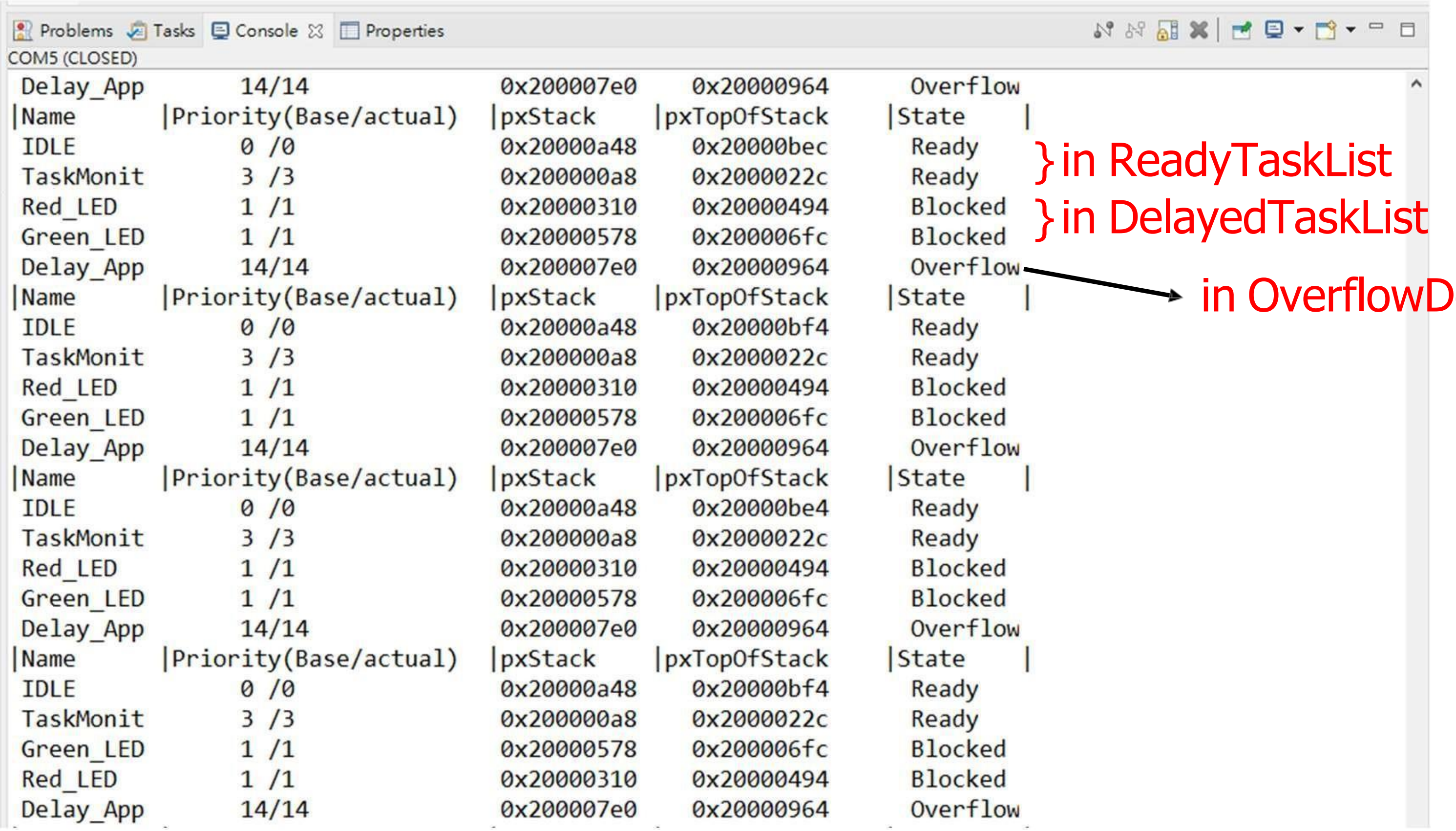
03 Lab2

04 Marco about Lab2

Lab requirement

- Create four task // 不用自己寫
  - Red_LED_App、Green_LED_App、Delay_App、TaskMonitor_App
- TaskMonitor_App will call **Taskmonitor()** periodicity
- **TaskMonitor()**
  - Traverse ReadyTaskList, DelayedTaskList, OverflowDelayedTaskList
  - Print TCB information by **UART**
    - Task Name、Priority(Base/actual)、Stack Pointer、Topofstack Pointer、Task State

Demo



```
 Delay_App       14/14              0x200007e0    0x20000964      Overflow
|Name        |Priority(Base/actual) |pxStack     |pxTopOfStack    |State    |
 IDLE             0 /0              0x20000a48    0x20000bec      Ready
 TaskMonit        3 /3              0x200000a8    0x2000022c      Ready
 Red_LED          1 /1              0x20000310    0x20000494      Blocked
 Green_LED        1 /1              0x20000578    0x200006fc      Blocked
 Delay_App       14/14              0x200007e0    0x20000964      Overflow
|Name        |Priority(Base/actual) |pxStack     |pxTopOfStack    |State    |
 IDLE             0 /0              0x20000a48    0x20000bf4      Ready
 TaskMonit        3 /3              0x200000a8    0x2000022c      Ready
 Red_LED          1 /1              0x20000310    0x20000494      Blocked
 Green_LED        1 /1              0x20000578    0x200006fc      Blocked
 Delay_App       14/14              0x200007e0    0x20000964      Overflow
|Name        |Priority(Base/actual) |pxStack     |pxTopOfStack    |State    |
 IDLE             0 /0              0x20000a48    0x20000be4      Ready
 TaskMonit        3 /3              0x200000a8    0x2000022c      Ready
 Red_LED          1 /1              0x20000310    0x20000494      Blocked
 Green_LED        1 /1              0x20000578    0x200006fc      Blocked
 Delay_App       14/14              0x200007e0    0x20000964      Overflow
|Name        |Priority(Base/actual) |pxStack     |pxTopOfStack    |State    |
 IDLE             0 /0              0x20000a48    0x20000bf4      Ready
 TaskMonit        3 /3              0x200000a8    0x2000022c      Ready
 Green_LED        1 /1              0x20000578    0x200006fc      Blocked
 Red_LED          1 /1              0x20000310    0x20000494      Blocked
 Delay_App       14/14              0x200007e0    0x20000964      Overflow
```

} in ReadyTaskList

} in DelayedTaskList

→ in OverflowDelayedTaskList

Lab2 grading

- (2%) 有印出**五個**TASK
- (2%) TCB 資料正確
- (1%) 不會多印出其他資料
- (3%) Lab report (一定要交)

Add New code

- Create four task
  - Red_LED_App priority = 1
  - Green_LED_App priority = 1
  - Delay_App priority = 14
  - TaskMonitor_App priority = 3 (把Stack調大一點，至少256)

Add New code

- **task.h**
  - #include "stm32f4xx_hal.h"
  - UART_HandleTypeDef huart2; // for USART2
  - void Taskmonitor(void);
- **Task.c**
  - void Taskmonitor(void){...}
- **List.h**
  - #define listGET_ITEM_OF_HEAD_ENTRY(pxList) ((&((pxList)->xListEnd))->pxNext)
- **FreeRTOSConfig.h**
  - configMAX_PRIORITIES **15**

Add New code

```
void Taskmonitor(void)
{
        /* Initialize string */
        char Monitor_data[130];
        memset(Monitor_data,'\0',sizeof(Monitor_data));

        /* Stop scheduler */
        /* Taskmonitor() will block when UART is transmitting data */
        /* Scheduler will change list data when Taskmonitor() is blocked */
        vTaskSuspendAll();

        /* Print title */
        sprintf(Monitor_data,"|Name      |Priority(Base/actual)  |pxStack     |pxTopOfStack    |State    |\n\r");
        HAL_UART_Transmit(&huart2,(uint8_t *)Monitor_data,strlen(Monitor_data),0xffff);

        /* pxReadyTasksLists  */
        /* pxDelayedTaskList*/
        /* pxOverflowDelayedTaskList  */

        /* Resume  scheduler */
        xTaskResumeAll();
}
```

Add New code

- **Main.c**

```
void TaskMonitor_App(void *pvParameters){
        for(;;){
        Taskmonitor();
        vTaskDelay(1000);
        }
 }

void Red_LED_App(void *pvParameters){
        uint32_t Redtimer = 800;
        for(;;){
                  HAL_GPIO_TogglePin(GPIOD,Red_LED_Pin);
                  vTaskDelay(Redtimer);
                  Redtimer+=1;
        }
  }
```

Add New code

- **Main.c**

```
void Green_LED_App(void *pvParameters){
        uint32_t G reentimer = 1000;
        for(;;){
                HAL_GPIO_TogglePin(GPIOD,Green_LED_Pin);
                vTaskDelay(Greentimer);
                Greentimer+=2;
        }
}
void Delay_App(void *pvParameters){
        int delayflag=0;
        uint32_t delaytime;
        while(1){
                if(delayflag==0){
                        delaytime = 1000;
                        delayflag=1;
                }else{
                        delaytime=0xFFFFFFFF;
                }
                vTaskDelay(delaytime);
        }
}
```

OUTLINE

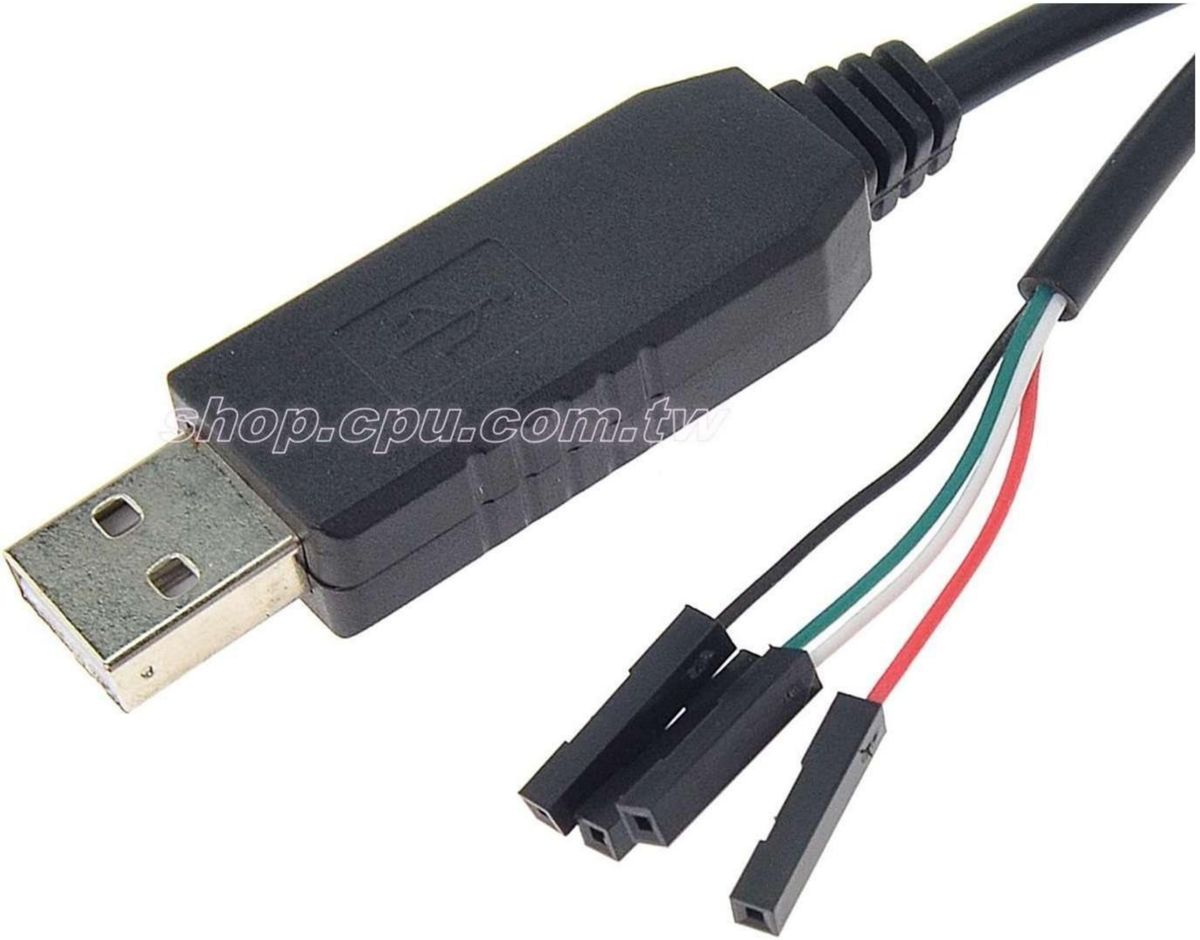
01 Lab requirement

02 UART

03 Lab2

04 Marco about Lab2

USB to TTL

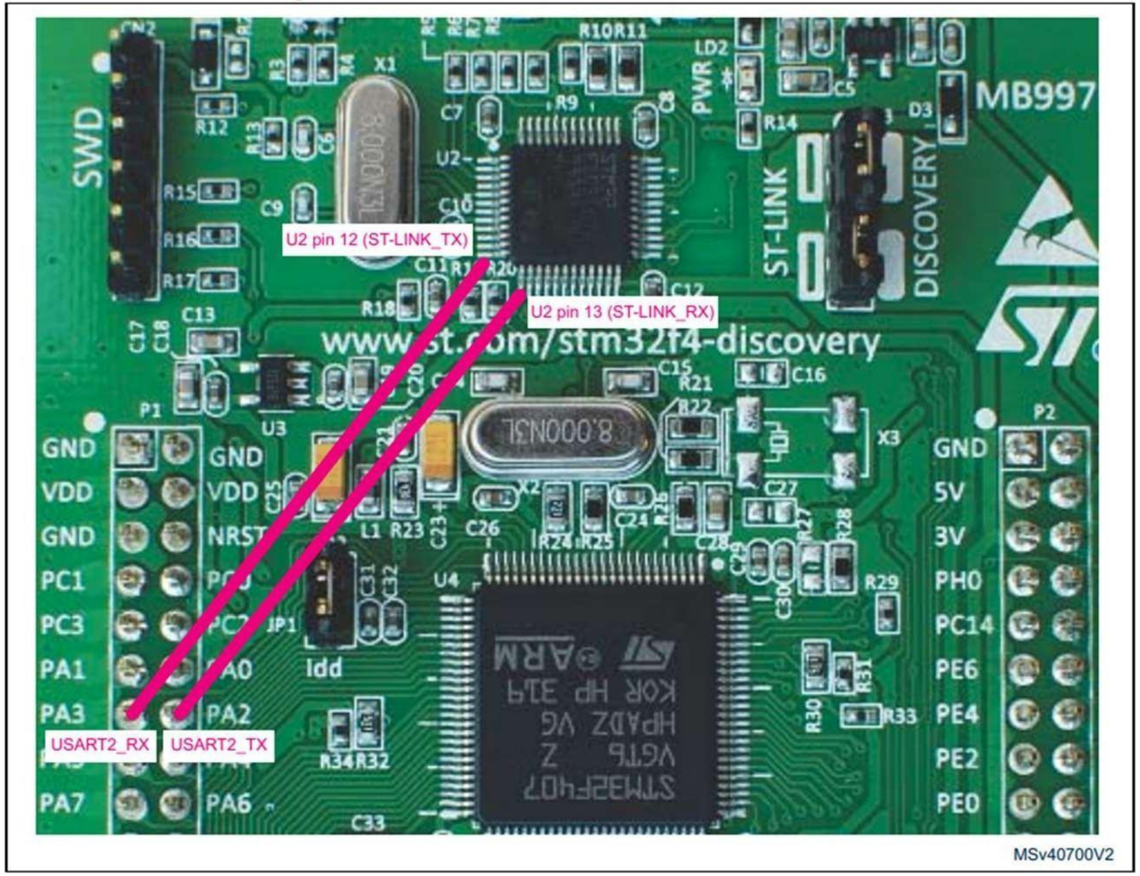


⬛	GND
🟩	TXD
⬜	RXD
🟥	+5V

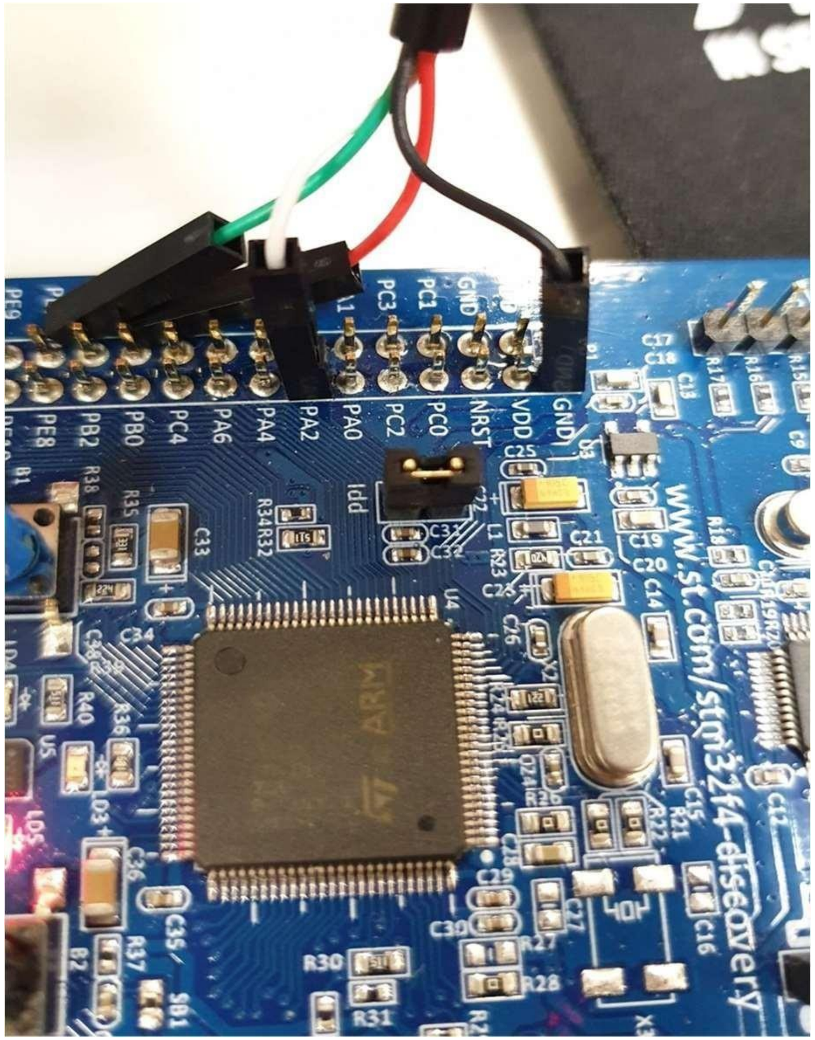
USART pin

Using an USART to USB dongle from the market connected for instance to STM32F407 USART2 available on connector P1 pin 14 (PA2: **USART2**_TX) and P1 pin 13 (PA3: **USART2**_RX).

Figure 6. ST-LINK VCP connection to USART2



USB to TTL



F407G-DISC transmit data(TXD) to PC. therefore, we need use PA2.

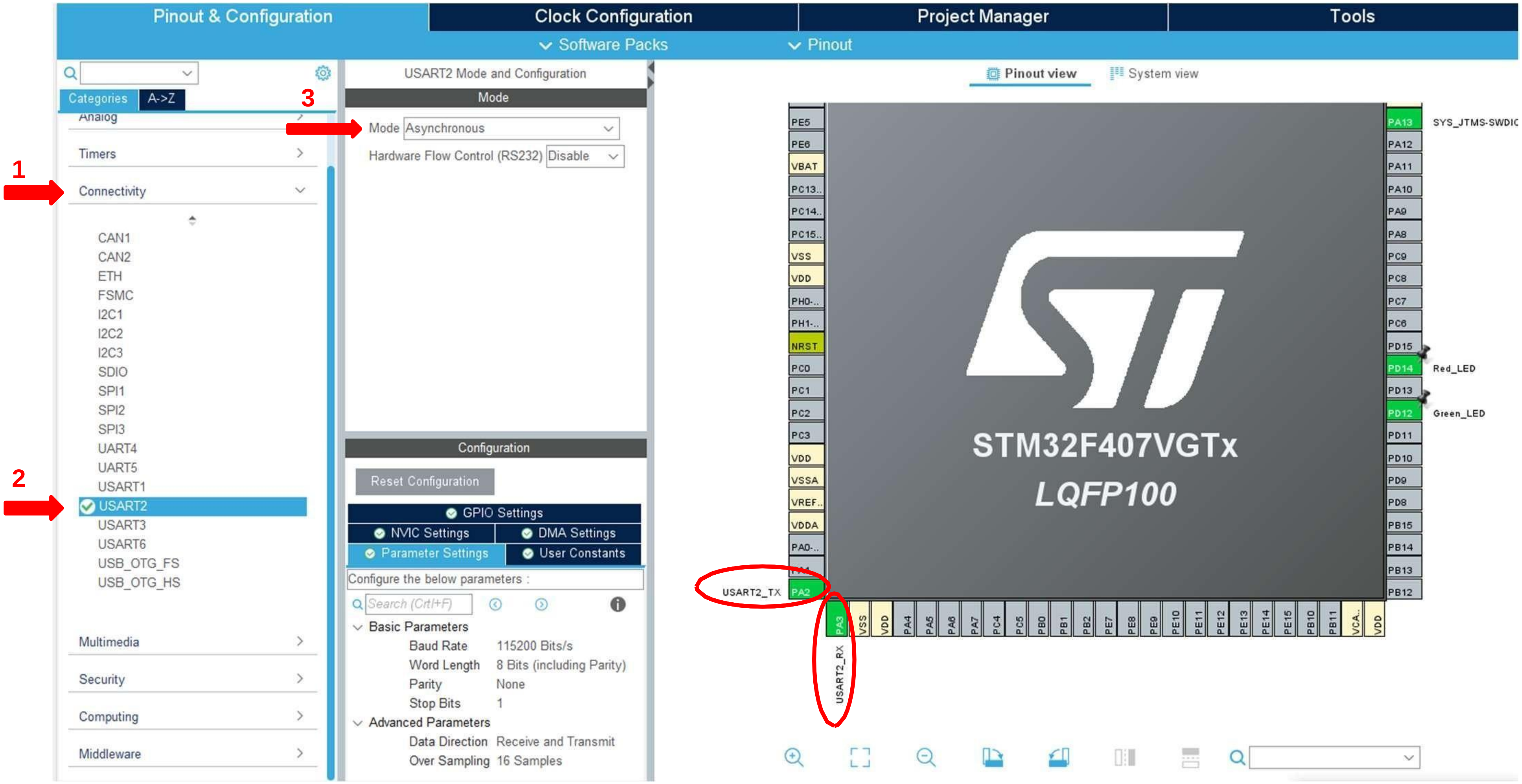
PC will receive external data(RXD), so we need use **white line** connect to PA2.

PA2(TXD) ↔ ⬜ **RXD**

F407G-DISC GND ↔ ⬛ **GND**

USART

USART set up



USART2 Mode and Configuration

Mode

Mode Asynchronous

Hardware Flow Control (RS232) Disable

Configuration

Configure the below parameters :

Basic Parameters

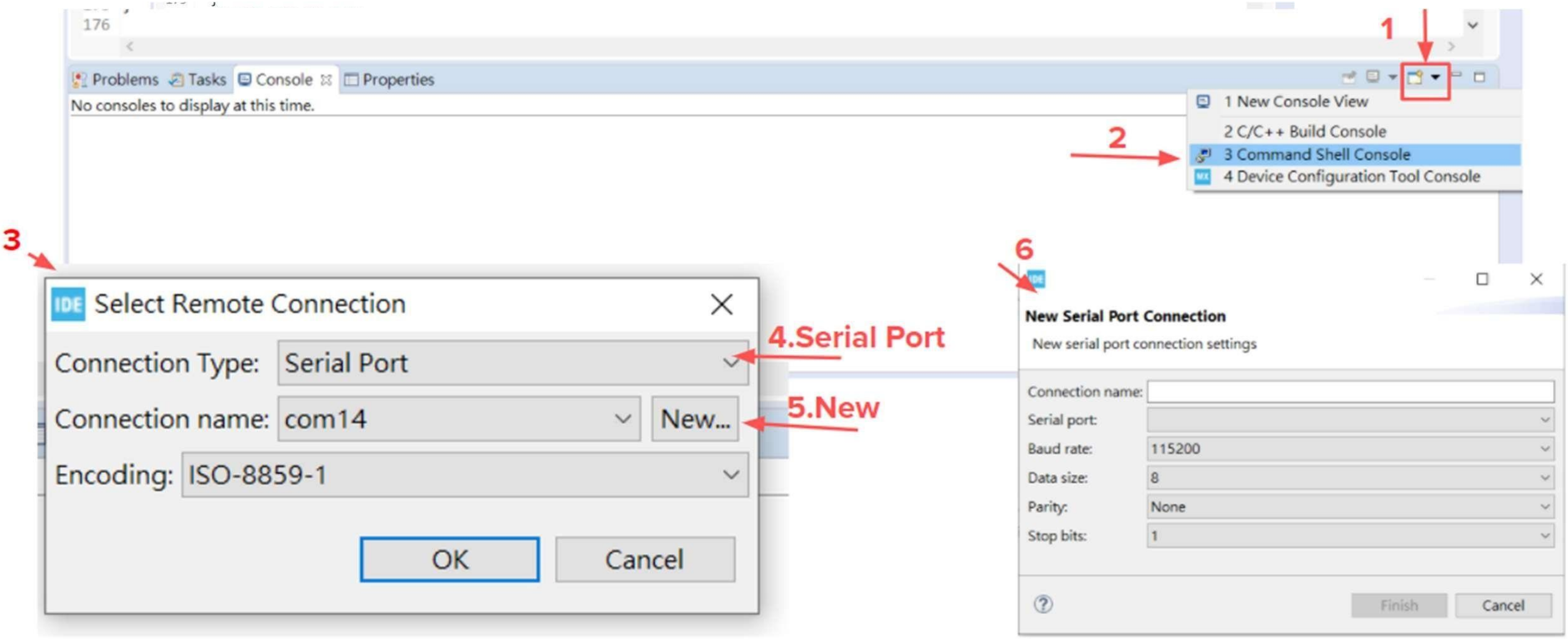
Baud Rate	115200 Bits/s
Word Length	8 Bits (including Parity)
Parity	None
Stop Bits	1

Advanced Parameters

Data Direction	Receive and Transmit
Over Sampling	16 Samples

Command shell set up

Serial port指的是電腦的USB port編號



1

2

3

4.Serial Port

5.New

6

Connection Type: Serial Port

Connection name: com14

Encoding: ISO-8859-1

Connection name:

Serial port:

Baud rate: 115200

Data size: 8

Parity: None

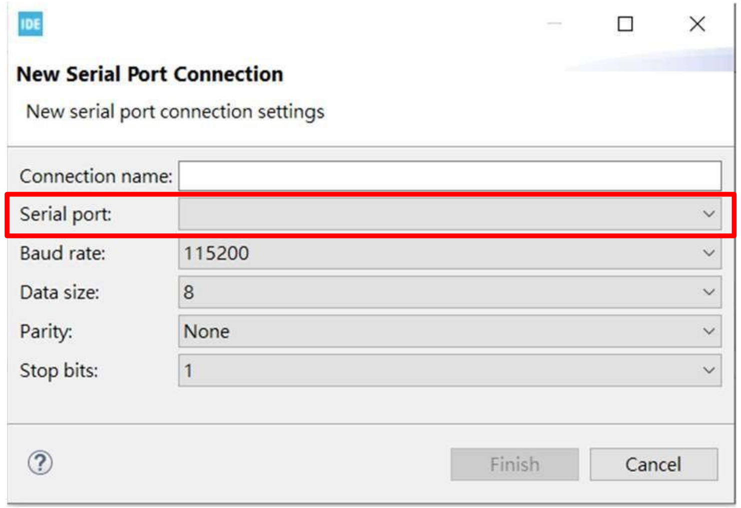
Stop bits: 1

How to find serial port

先開啟裝置管理員

USB to TTL插上電腦時，裝置管理員的連接埠會跳出一個Prolific USB-to-Serial Comm port

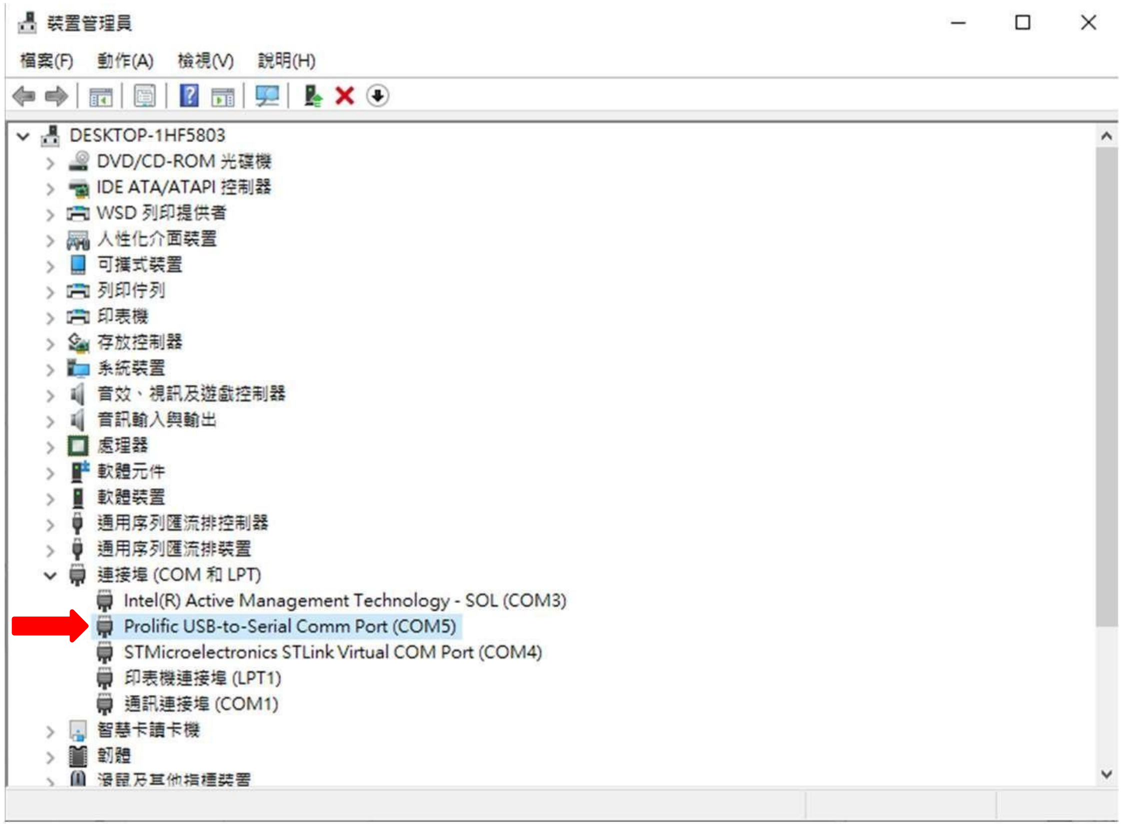
Serial port就會是括弧裡的編號



New Serial Port Connection

New serial port connection settings

Connection name:	
Serial port:	
Baud rate:	115200
Data size:	8
Parity:	None
Stop bits:	1



- DESKTOP-1HF5803
  - DVD/CD-ROM 光碟機
  - IDE ATA/ATAPI 控制器
  - WSD 列印提供者
  - 人性化介面裝置
  - 可攜式裝置
  - 列印佇列
  - 印表機
  - 存放控制器
  - 系統裝置
  - 音效、視訊及遊戲控制器
  - 音訊輸入與輸出
  - 處理器
  - 軟體元件
  - 軟體裝置
  - 通用序列匯流排控制器
  - 通用序列匯流排裝置
  - 連接埠 (COM 和 LPT)
    - Intel(R) Active Management Technology - SOL (COM3)
    - Prolific USB-to-Serial Comm Port (COM5)
    - STMicroelectronics STLink Virtual COM Port (COM4)
    - 印表機連接埠 (LPT1)
    - 通訊連接埠 (COM1)
  - 智慧卡讀卡機
  - 韌體
  - 滑鼠及其他指標裝置

HAL_UART_Transmit

```
HAL_UART_Transmit(
                    USART_HandleTypeDef * huart,
                    uint8_t * pTxData,
                    uint16_t Size,
                    uint32_t Timeout
                   );
```

huart: Pointer to a UART_HandleTypeDef structure that contains the configuration information for the specified UART module

PTxData: Pointer to data buffer

Size: Amount of data elements to be sent

Timeout: Timeout duration

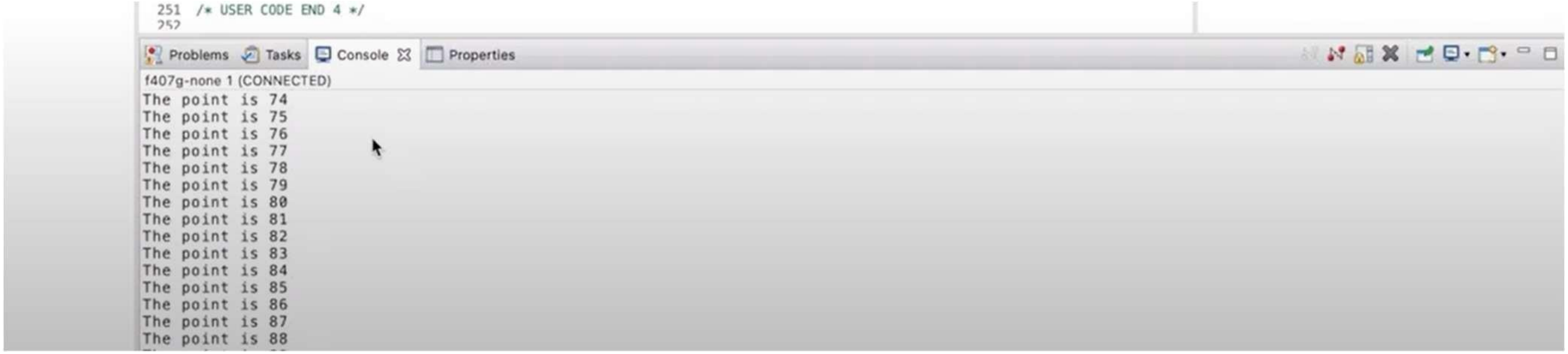
USART Tx Test

創建一個 Task 以測試USART 。

```
UART_HandleTypeDef huart2;
void USART_Test(void *pvParameters){
        uint32_t Monitortimer = 400;
        char MonitorTset[30];
        char num[15];
        int i = 0;
        while(1){
                memset(MonitorTset,'\0',sizeof(MonitorTset));
                memset(num,'\0',sizeof(num));
                itoa(i,num,10);
                strcat(num," ");
                sprintf(MonitorTset,"The point is %s\n\r",num);
                HAL_UART_Transmit(&huart2,(uint8_t *)MonitorTset,strlen(MonitorTset),0xffff);
                vTaskDelay(Monitortimer);
                Monitortimer += 1;
                i += 1;
        }
}
```

USART Tx Test

- 理論上如果設定沒有問題，運行後會看到console 顯示你transmit 的 string
- 如下圖所示
- 連接你TTL的port的Console 只要創建一個就好，或是只connect 一個。



```
/* USER CODE END 4 */
```

```
f407g-none 1 (CONNECTED)
The point is 74
The point is 75
The point is 76
The point is 77
The point is 78
The point is 79
The point is 80
The point is 81
The point is 82
The point is 83
The point is 84
The point is 85
The point is 86
The point is 87
The point is 88
```

OUTLINE

01 Lab requirement

02 UART

03 Lab2

04 Marco about Lab2

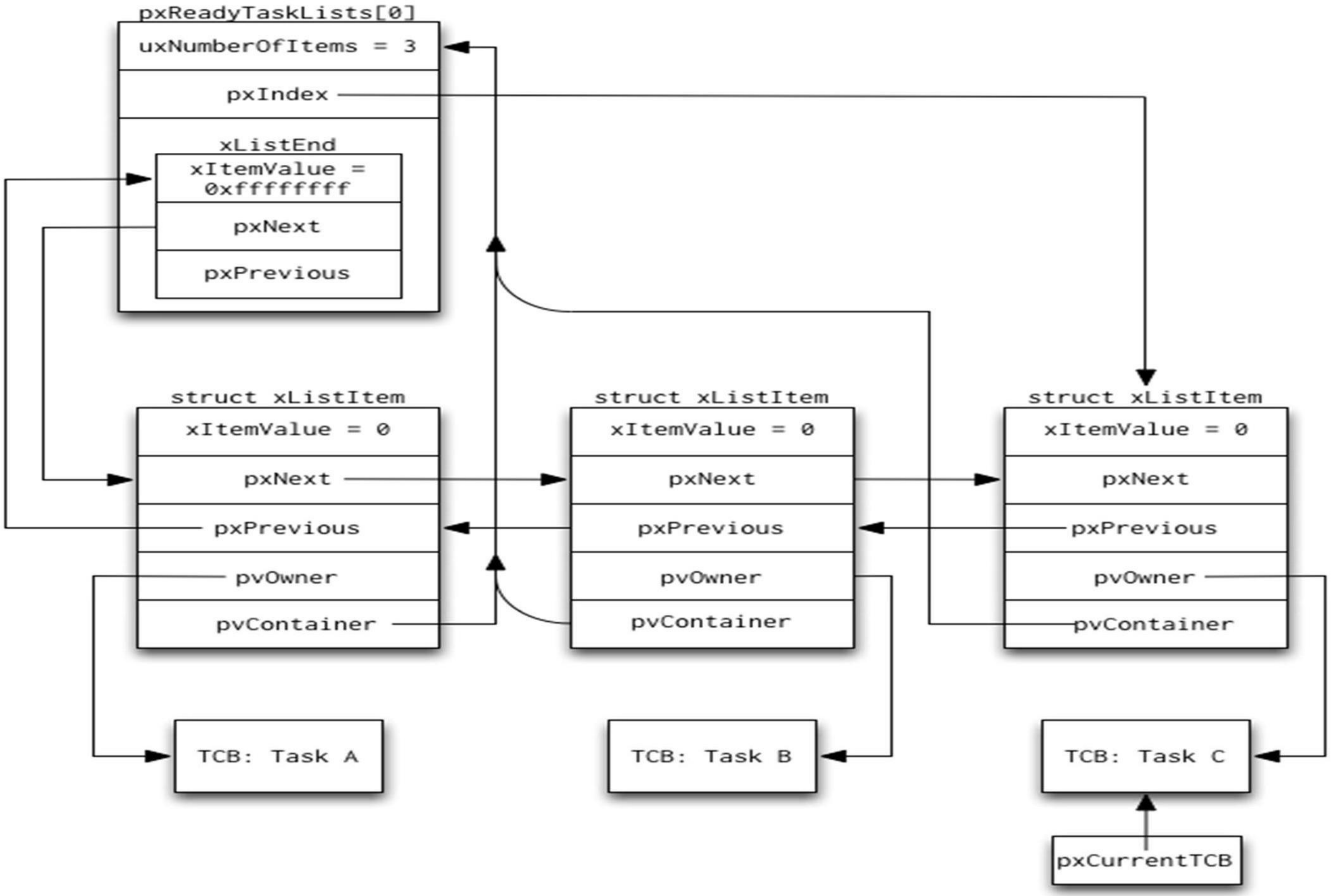
List

tasks.c

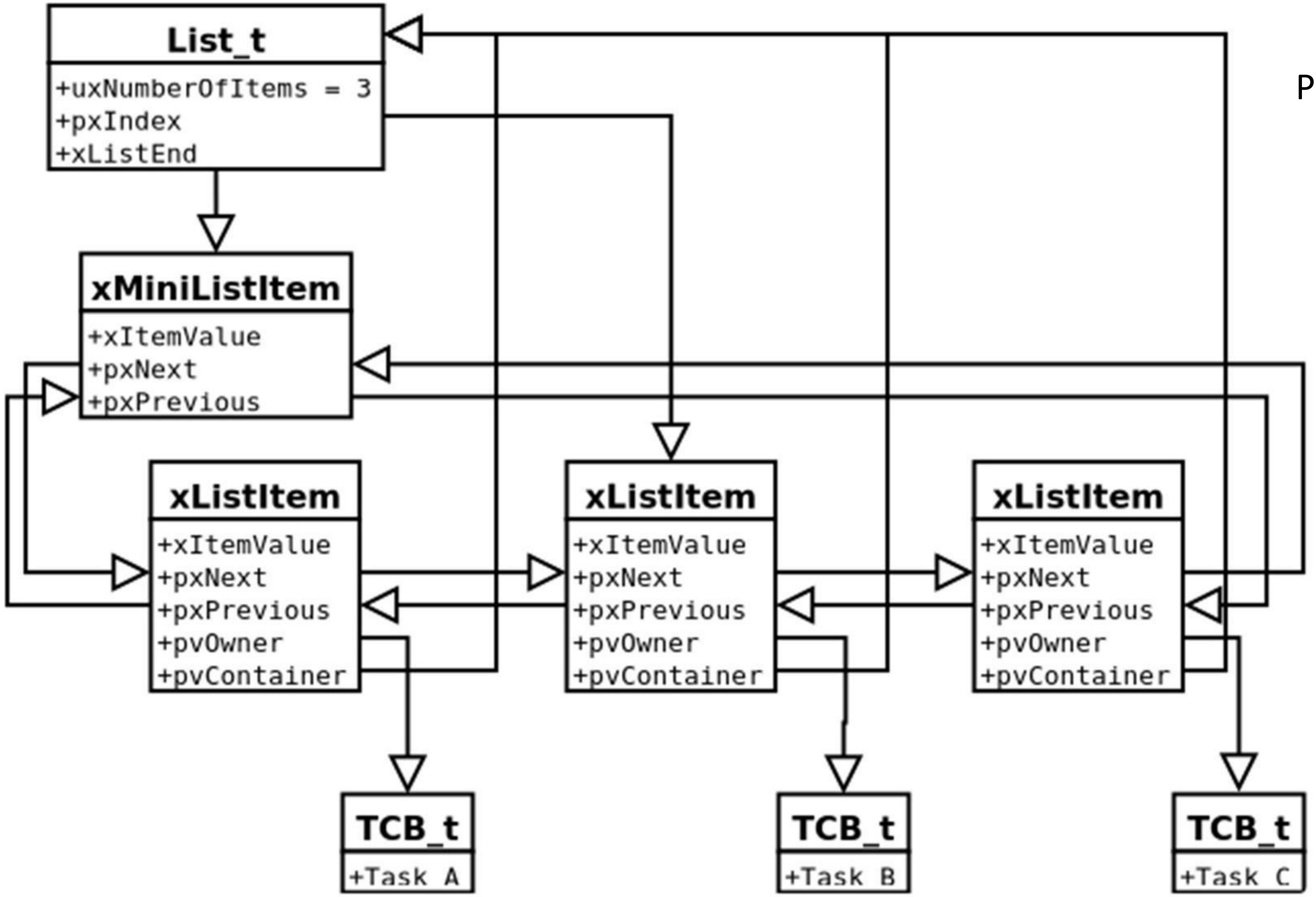
```
PRIVILEGED_DATA static List_t pxReadyTasksLists[ configMAX_PRIORITIES ];/*<Prioritised ready tasks. */
PRIVILEGED_DATA static List_t xDelayedTaskList1; /*<Delayed tasks. */
PRIVILEGED_DATA static List_t xDelayedTaskList2; /*<Delayed tasks (two lists are used - one for delays that have overflowed the current tick count. */
PRIVILEGED_DATA static List_t * volatile pxDelayedTaskList; /*<Points to the delayed task list currently being used. */
PRIVILEGED_DATA static List_t *volatile pxOverflowDelayedTaskList; /*<Points to the delayed task list currently being used to hold tasks that have overflowed the current tick count. */
```

- Only need access
  a. pxReadyTasksLists
  b. pxDelayedTaskList
  c. pxOverflowDelayedTaskList

List


pxReadyTaskLists[0]
uxNumberOfItems = 3
pxIndex
xListEnd
xItemValue =
0xffffffff
pxNext
pxPrevious
struct xListItem
xItemValue = 0
pxNext
pxPrevious
pvOwner
pvContainer
struct xListItem
xItemValue = 0
pxNext
pxPrevious
pvOwner
pvContainer
struct xListItem
xItemValue = 0
pxNext
pxPrevious
pvOwner
pvContainer
TCB: Task A
TCB: Task B
TCB: Task C
pxCurrentTCB

List



Pxindex: 會指到其中一個list item

xLIST_ITEM

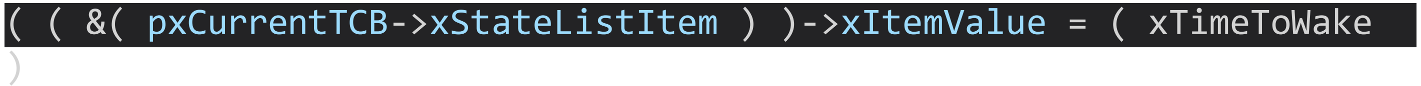
```
struct xLIST_ITEM
{
        configLIST_VOLATILE  TickType_t  xItemValue;
        struct xLIST_ITEM  * configLIST_VOLATILE  pxNext;
        struct  xLIST_ITEM  * configLIST_VOLATILE  pxPrevious;
        void * pvOwner;
        struct xLIST  * configLIST_VOLATILE  pxContainer;
};
typedef struct xLIST_ITEM ListItem_t;
```

Task Control Block

```
typedef struct tskTaskControlBlock  /* The old naming convention is used to prevent breaking kernel aware debuggers. */
{
            volatile StackType_t *pxTopOfStack; /*< Points to the location of the last item placed on the tasks stack. THIS MUST BE THE
FIRST MEMBER OF THE TCB STRUCT.  */

             ListItem_t xStateListItem; /*< The list that the state list item of a task is reference from denotes the state of that task (Ready,
Blocked, Suspended ). */
             ListItem_t xEventListItem; /*< Used to reference a task from an event list. */
             UBaseType_t uxPriority; /*< The priority of the task. 0 is the lowest priority. */
             StackType_t *pxStack; /*< Points to the start of the stack. */
             Char pcTaskName[ configMAX_TASK_NAME_LEN ]; /*< Descriptive name given to the task when created.  Facilitates
debugging only. */ /*lint !e971 Unqualified char types are allowed for strings and single characters only. */

             #if ( ( portSTACK_GROWTH > 0 ) || ( configRECORD_STACK_HIGH_ADDRESS == 1 ) )
                          StackType_t              *pxEndOfStack; /*< Points to the highest valid address for the stack. */
             #endif

             #if ( configUSE_MUTEXES == 1 )
                          UBaseType_t              uxBasePriority; /*< The priority last assigned to the task - used by the priority
inheritance mechanism. */
                          UBaseType_t uxMutexesHeld;
             #endif
} tskTCB;
```

tasks.c

```
static void prvAddCurrentTaskToDelayedList( TickType_t xTicksToWait, const BaseType_t xCanBlockIndefinitely ){

                /* Calculate the time at which the task should be woken if the event
                does not occur. This may overflow but this doesn't matter, the
                kernel will manage it correctly. */
                xTimeToWake = xConstTickCount + xTicksToWait;

                /* The list item will be inserted in wake time order. */
                listSET_LIST_ITEM_VALUE( &( pxCurrentTCB->xStateListItem ), xTimeToWake );
```



```
( ( &( pxCurrentTCB->xStateListItem ) )->xItemValue = ( xTimeToWake
)
```

```
                if( xTimeToWake < xConstTickCount )
                {
                        /*Wake time has overflowed. Place this item in the overflow
                        list. */
                        vListInsert( pxOverflowDelayedTaskList, &( pxCurrentTCB->xStateListItem ) );
                }
                else
                {
                        /* The wake time has not overflowed, so the current block list
                        is used. */
                        vListInsert( pxDelayedTaskList, &( pxCurrentTCB->xStateListItem ) )


}
```

OUTLINE

01 Lab requirement

02 UART

03 Lab2

04 Marco about Lab2

Macro

- list.h
    - **#define** listLIST_IS_EMPTY(pxList) (((pxList)->uxNumberOfItems == (UBaseType_t) 0) ? pdTRUE : pdFALSE)
        - listLIST_IS_EMPTY(&(pxReadyTasksLists[xPriority]);

    - **#define** listCURRENT_LIST_LENGTH(pxList) ((pxList)->uxNumberOfItems)
        - listCURRENT_LIST_LENGTH(&(pxReadyTasksLists[xPriority]));