MICROCHIP

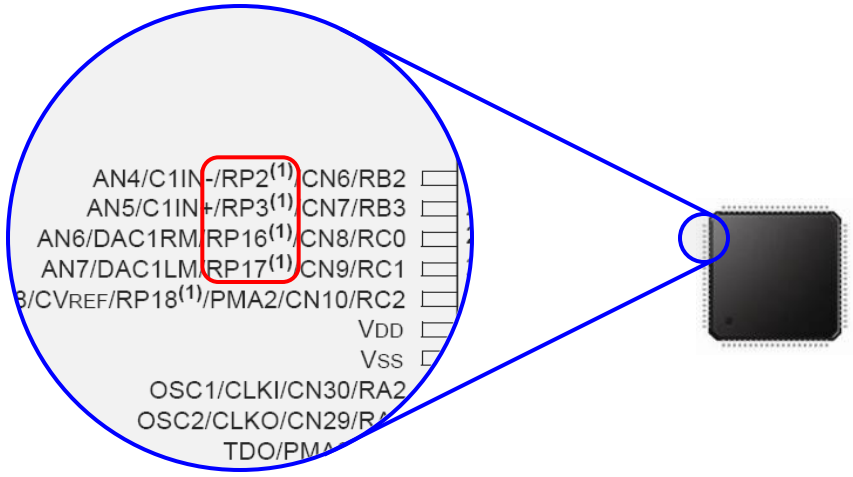
Regional Training Centers

Section 10
PPS and UART

MICROCHIP
Regional Training Centers

What's PPS

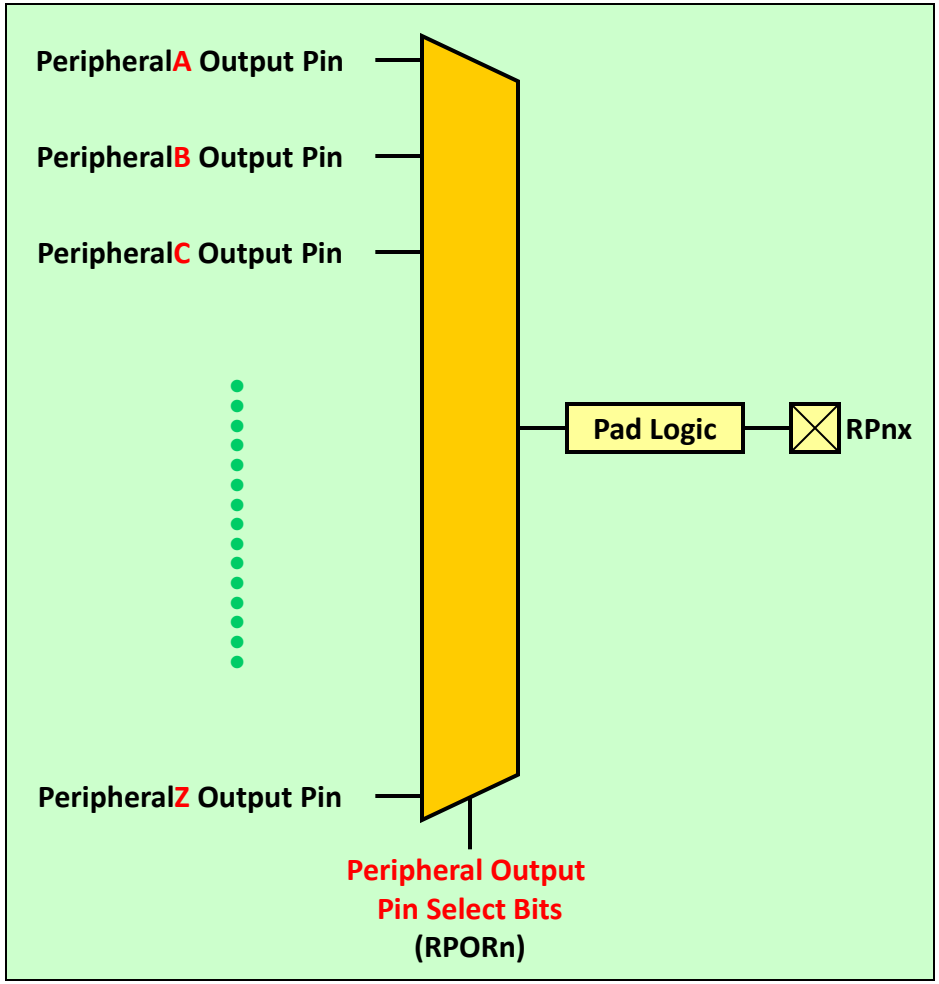
- **PPS：Peripheral Pin Select, 周邊接腳選擇**
  一種可以重新改變周邊訊號接腳位置的功能。讓使用者可以自行安排(Remapping)周邊訊號接腳位置, 達到最彈性化的接腳使用。
- 在接腳上有標示RPn/RPIn的, 都是可以Remapping的接腳。可已透過設定將UART, SPI,etc..的訊號接腳透過該Pin連接。
- PPS支援大部分的數位週邊,如:
  UART,SPI,OC,IC等。**但需要較複雜狀態處理的數位週邊及類比功能則不支援,其接腳為固定不可變更的,如:**
  I²C,USB,PMP及類比數輸入等。



MICROCHIP
Regional Training Centers

PPS's Output

- 輸出多工器的輸入端連接到各個周邊的輸出訊號,輸出則接到特定接腳(RP*nx*)。
- 每個接腳(RP*nx*)都有專屬的多工器,可單獨指定其所要接的輸出訊號。

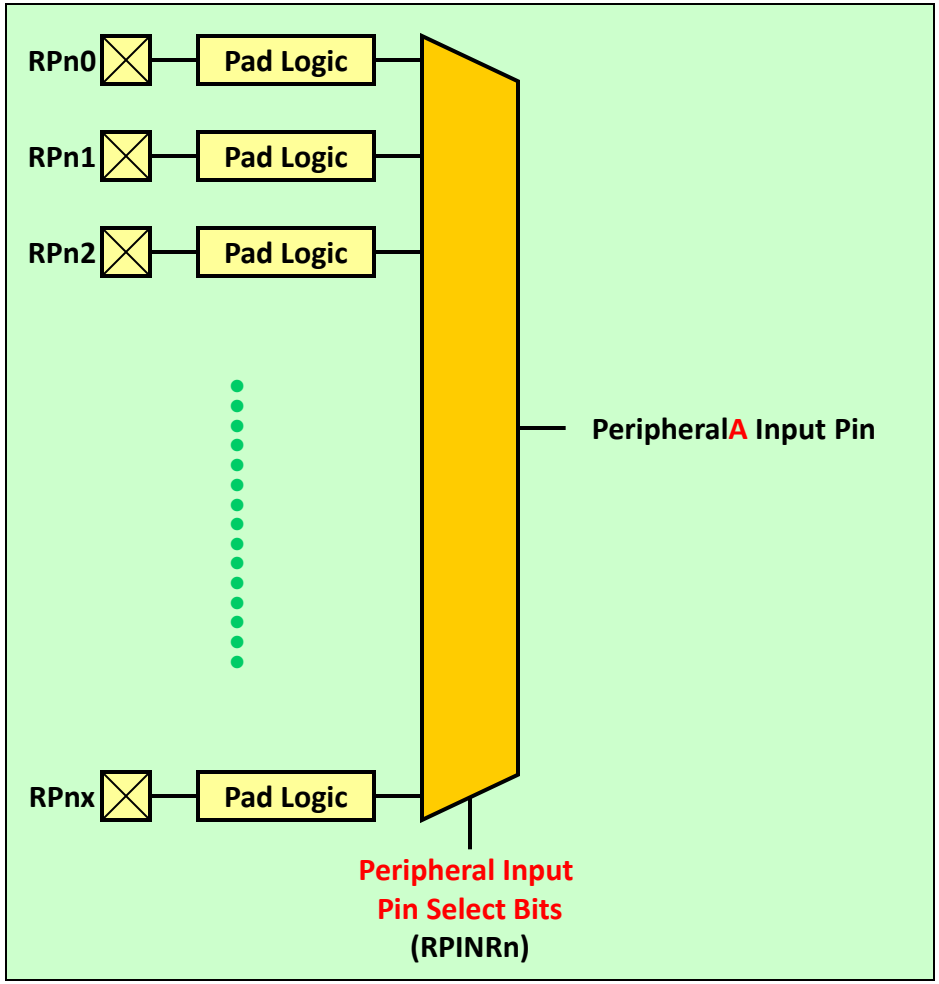


PPS基本輸出架構

MICROCHIP
Regional Training Centers

PPS's Input

- 輸入多工器的輸入端連接到各個輸入接腳(RP*nx*,RPI*nx*),輸出則接到某個周邊的輸入接腳。
- 每個周邊都有專屬的多工器,可單獨指定其所要連接的接腳(RP*nx*,RPI*nx*)。



PPS基本輸入架構

MICROCHIP
Regional Training Centers

支援PPS功能的周邊

- External Interrupts
- Timer External Clocks
- Input Captures
- Output Compares
- PWM Fault Input pins
- SPI,UART Functions
- QEI Inputs
- Data Converter Interface
- CAN

Table 30-1: Selectable Input Sources (Maps Input to Function)

Input Name(1)	Function Name	Register	Configuration Bits
External Interrupt 1	INT1	RPINR0	INT1R<4:0>
External Interrupt 2	INT2	RPINR1	INT2R<4:0>
Timer2 External Clock	T2CK	RPINR3	T2CKR<4:0>
Timer3 External Clock	T3CK	RPINR3	T3CKR<4:0>
Input Capture 1	IC1	RPINR7	IC1R<4:0>
Input Capture 2	IC2	RPINR7	IC2R<4:0>
Input Capture 7	IC7	RPINR10	IC7R<4:0>
Input Capture 8	IC8	RPINR10	IC8R<4:0>
Output Compare Fault A	OCFA	RPINR11	OCFAR<4:0>
PWM1 Fault	$\overline{\text{FLTA1}}$	RPINR12	FLTA1R<4:0>
PWM2 Fault	$\overline{\text{FLTA2}}$	RPINR13	FLTA2R<4:0>
QEI Phase A	QEA	RPINR14	QEAR<4:0>
QEI Phase B	QEB	RPINR14	QEBR<4:0>
QEI Index	INDX	RPINR15	INDXR<4:0>
UART1 Receive	U1RX	RPINR18	U1RXR<4:0>
UART1 Clear t			
SPI1 Data Inpu			
SPI1 Clock Inp			
SPI1 Slave Sel			

Note 1: Unles

Table 30-2: Output Selection for Remappable Pin (RPn)

Function	RPnR<4:0>	Output Name
NULL	00000	RPn tied to default port pin
U1TX	00011	RPn tied to UART1 Transmit
$\overline{\text{U1RTS}}$	00100	RPn tied to UART1 Ready to Send
SDO1	00111	RPn tied to SPI1 Data Output
SCK1OUT	01000	RPn tied to SPI1 Clock Output
SS1OUT	01001	RPn tied to SPI1 Slave Select Output
OC1	10010	RPn tied to Output Compare 1
OC2	10011	RPn tied to Output Compare 2
UPDN	11010	RPn tied to QEI direction (UPDN) status

MICROCHIP
Regional Training Centers

PPS Write Protection

- **PPS 設定前必須先進行解鎖, 設定完後再上鎖。**
  以避免非預期的動作更動PPS的設定。
  任何位元的非預期改變都會造成 MCU Reset。
- 解鎖動作必須將OSCCON的IOLOCK清除為0;
  OSCCONbits.IOLOCK = 0;
  鎖定動作必須必須將OSCCON的IOLOCK設為1。
  OSCCONbits.IOLOCK = 1;
- PPS 的鎖定模式, 在Configuration Bits中設定
  IOL1WAY:IOLOCK One-Way Set Enable bit
  #pragma config IOL1WAY= ON, PPS只能設定一次。
  #pragma config IOL1WAY= OFF, PPS可重複的設定。

MICROCHIP
Regional Training Centers

XC16's PPS Function Support

- **Output Mapping**

  iPPSOutput(*pin* , *fn*);

  pin:PPS的接腳編號

  OUT_PIN_PPS_RPx

  fn:周邊模組輸出訊號

  OUT_FN_PPS_NULL , OUT_FN_PPS_CxOUT ,
  OUT_FN_PPS_UxTX , OUT_FN_PPS_SDOx ,
  OUT_FN_PPS_OCx ,

- **Input Mapping**

  iPPSInput(*fn* , *pin*);

  fn:周邊模組輸入訊號

  IN_FN_PPS_INTx , IN_FN_PPS_ICx , IN_FN_PPS_OCFx ,
  IN_FN_PPS_UxRX , IN_FN_PPS_SDIx ,

  pin:PPS的接腳編號

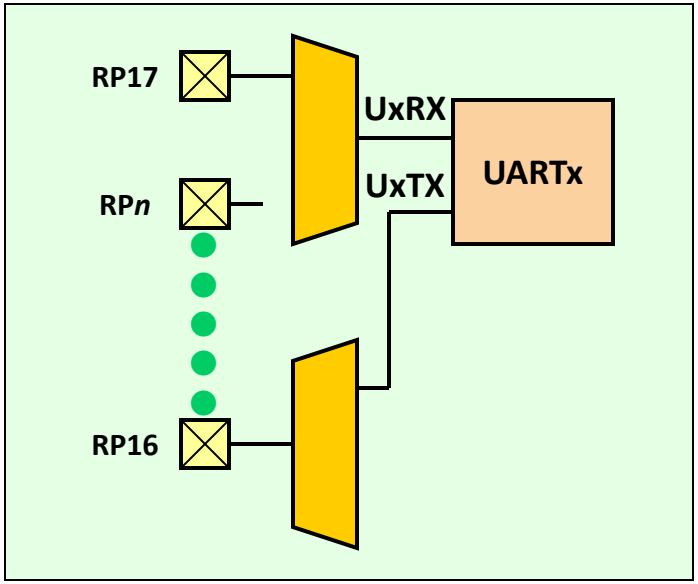
  IN_PIN_PPS_RPx

PPS UART Mapping Example

- PPS設定UART2, Tx, Rx的Mapping:

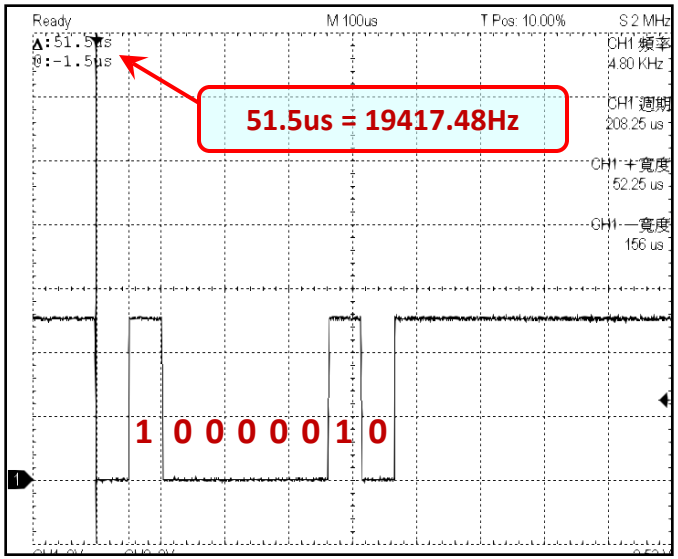
  Ex:

```
#include <PPS.h>
int main( void )
{
    ...
    ...
    ...
    ...
    PPSUnLock;
    iPPSOutput( OUT_PIN_PPS_RP16 , OUT_FN_PPS_U2TX );
    iPPSInput( IN_FN_PPS_U2RX , IN_PIN_PPS_RP17 );
    PPSLock;
    ...
}
```

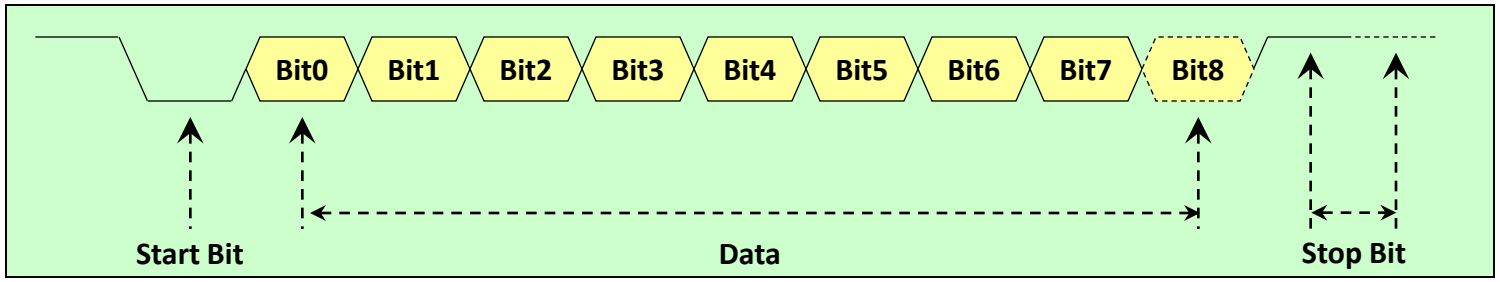


What's UART

- **UART:Universal Asynchronous Receiver Transmitter,**
  泛用非同步接收傳送模組。
  一個以串列方式進行資料交換與溝通的通訊模組。
- UART 傳輸的資料由LSB先傳。
  格式包含一個起始位元, 八或九位元的資料, 一至二個停止位元。

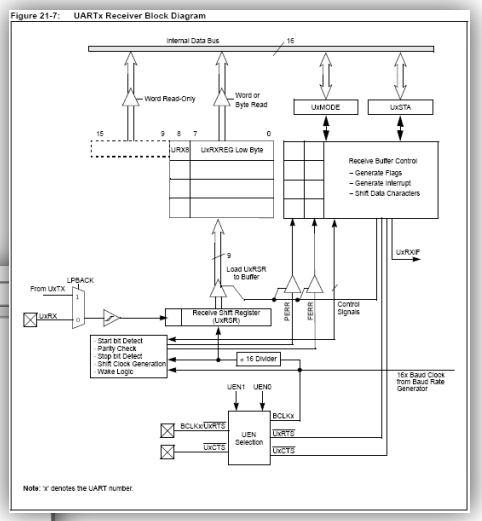


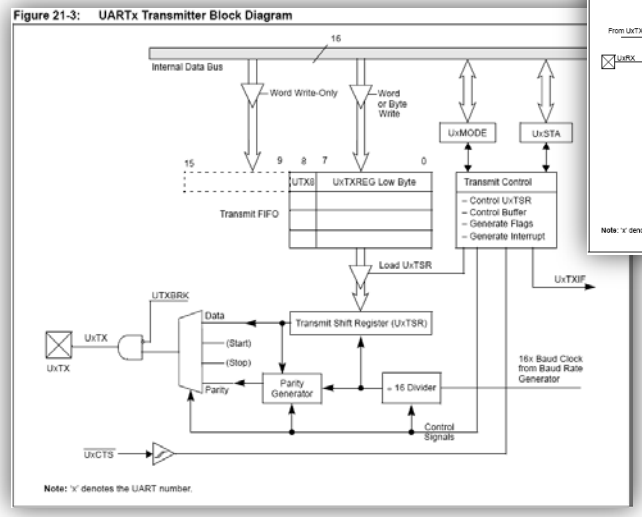
UART傳送範例'A'(0x41), 19200 bps



PIC24 UART Tran./Rece. Block

- PIC24的UART Module支援, 8 or 9-Bits Data, 同位檢查及硬體流量控制(DTS, RTS)。
- 採全雙工, 非同步通訊。支援RS-232, RS-485, IrDA及LIN Bus。
- 四層深度的輸出入緩衝器(Buffer) 。
- 具有錯誤檢查機制
  (Parity Error, Frame Error, Overflow, etc..),
  Loopback及
  Address Detect
  (RS-485), Auto Baud
  Detect(LIN bus)功能。





MICROCHIP
Regional Training Centers

UART's Baud Rate

- PIC24的鮑率(Baud Rate)的計算分為高低兩種速率, 透過UxMODE中BRGH來設定。BRGH=0為低速;反之則為高速。
- 低速率時(BRGH=0), 計算方式如下:

$$UxBRG = \frac{SystemClock}{16 \times IdeaBaulRate} - 1 \qquad ActualBaudRate = \frac{SystemClock}{16 \times (UxBRG + 1)}$$

- 高速率時(BRGH=1),計算方式如下:

$$UxBRG = \frac{SystemClock}{4 \times IdeaBaulRate} - 1 \qquad ActualBaudRate = \frac{SystemClock}{4 \times (UxBRG + 1)}$$

MICROCHIP
Regional Training Centers

Baud Rate Error

- UART模組鮑率計算誤差?
  假設System Clock為16MHz。預設鮑率115,200 bps。計算鮑率與誤差。
  低速率時(BRGH=0),鮑率與誤差計算:

$$\frac{16MHz}{16\times 115200}-1=7.68\cong 8 \qquad \frac{16MHz}{16\times(8+1)}\cong 111111 \qquad \frac{111111-115200}{115200}\times 100\%\cong -3.55\%$$

  高速率時(BRGH=1),鮑率與誤差計算:

$$\frac{16MHz}{4\times 115200}-1=33.7\cong 34 \qquad \frac{16MHz}{4\times(34+1)}\cong 114286 \qquad \frac{114286-115200}{115200}\times 100\%\cong -0.79\%$$

- UART誤差容忍度約為3%,在範例中,以低速率設定時,產生的誤差過高。此時就算UART的硬體設定無誤,也會因為過高的誤差導致UART動作不正常。

MICROCHIP
Regional Training Centers

UART Transmit Example

- **UART Transmit(Polling, String) Example**

```
unsigned char UARTString[ 100 ];
sprintf( ( char * ) UARTString , “AD Value %4d\r\n”, ADValue );
 // 格式化輸出函數,功能與printf相同, 不同的是,
 // sprintf會將結果輸出到字串陣列中。
```

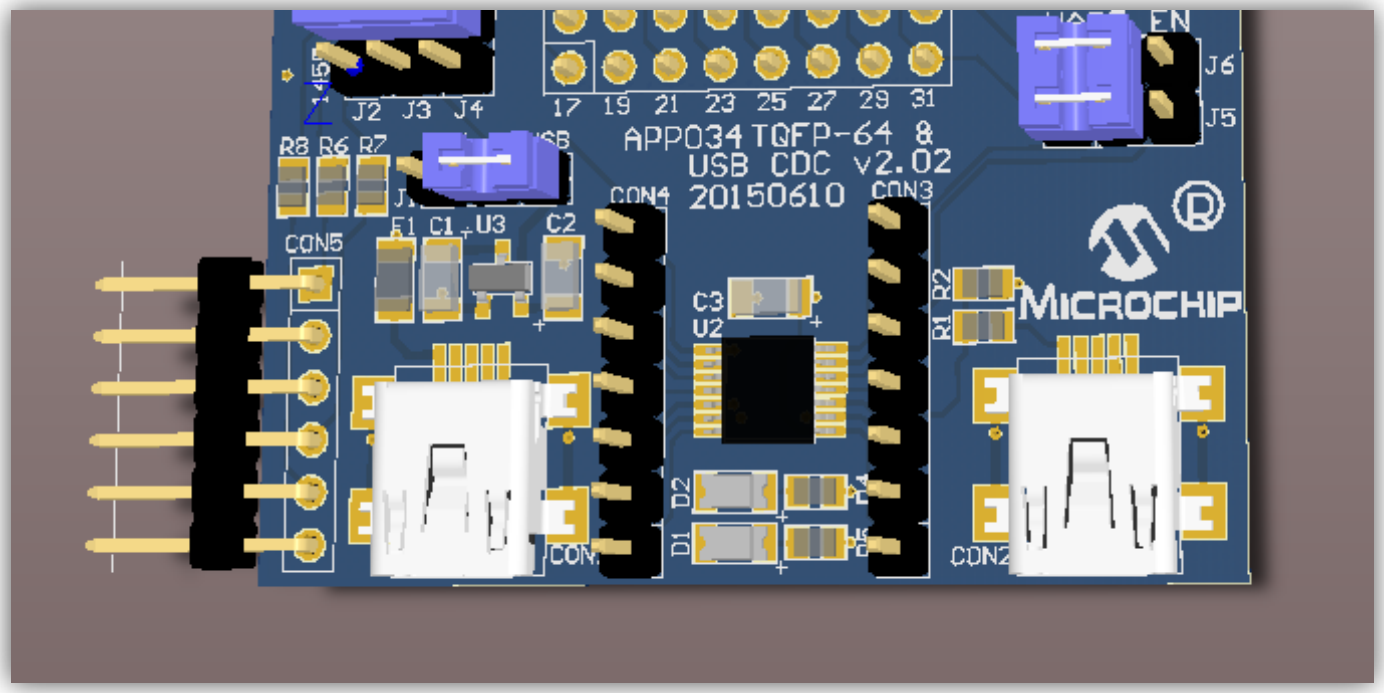
- **void UART_PutRAMString(unsigned char * String)**

```
{
        while( *String != 0x00 )
        {
                while(U2STAbits.UTXBF );
                U2TXREG = *String++;
        }
}
```

 // 將字串陣列的字元,逐一透過UART2傳送出去,直到字串結尾(0x00) 。

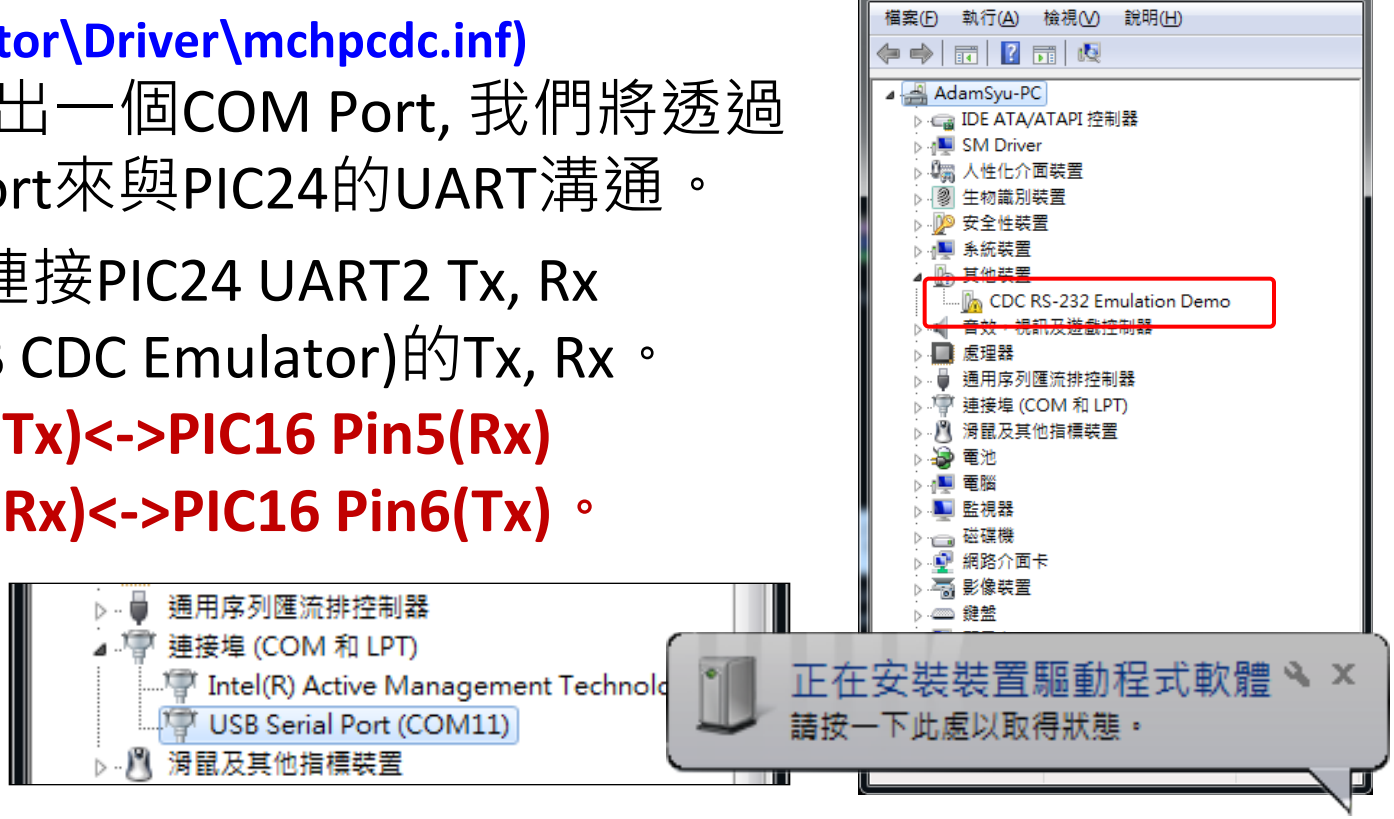
USB CDC Emulator

- PIC16F1455-I/ST內建有USB的Module, 我們利用此MCU實作了一個USB CDC類別的USB to UART Bridge。
- D1, D2是傳送與接收指示燈。指示目前資料傳輸的情形。



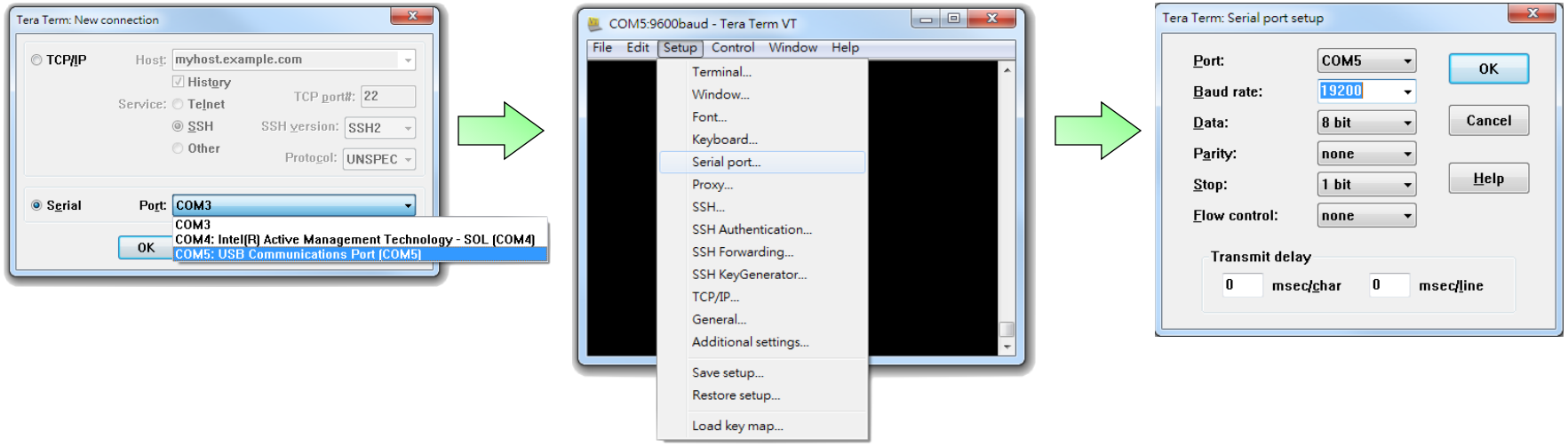
Driver Installation

- PIC16F1455已經預先燒錄USB Serial Emulator的firmware。
- 使用Mini USB Cable連接CON2, 此時PC會出現”找到新硬體”的訊息, 接著請安裝該裝置的驅動程式, **(\Serial Emulator\Driver\mchpcdc.inf)** 安裝後會多出一個COM Port, 我們將透過這個COM Port來與PIC24的UART溝通。
- J5與J6用來連接PIC24 UART2 Tx, Rx與PIC16(USB CDC Emulator)的Tx, Rx。
  PIC24 Pin32(Tx)<->PIC16 Pin5(Rx)
  PIC24 Pin31(Rx)<->PIC16 Pin6(Tx)。



Tera Term

- 由於Windows XP之後的作業系統, 不再內建終端機軟體, 因此必須另行安裝替代軟體, 才能監控COM Port的狀態。
- 請先安裝Tera Term, 然後開啟然後Tera Term, 指定COM Port, 設定為**9600 , N , 8 , 1**。



Lab10 UART Tx Polling

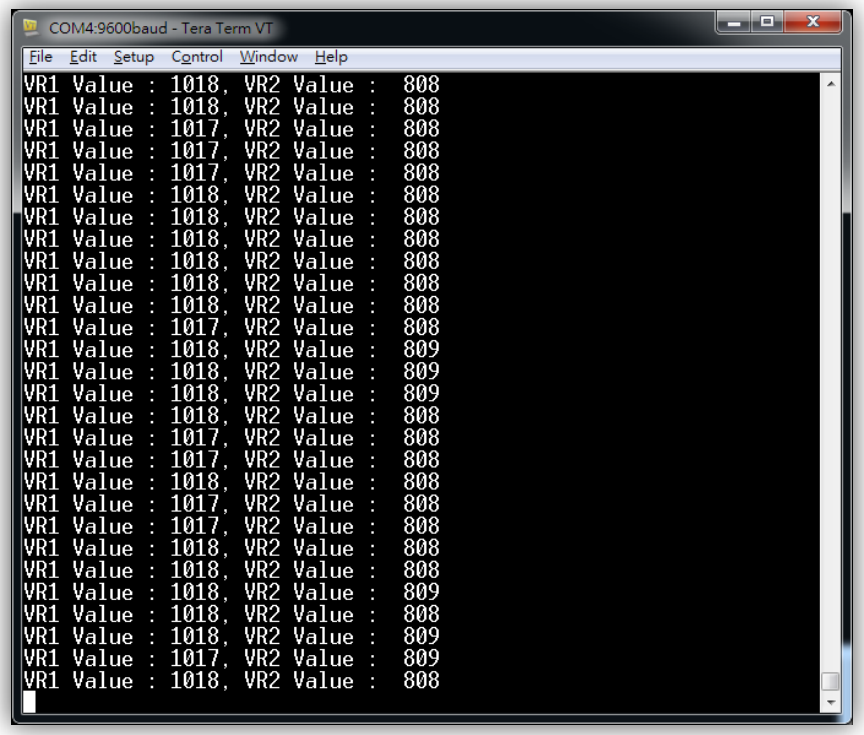
目 標

- 嘗試透過MCC的設定, 在Lab9架構上加入**UART2**的設定。並透過Timer定期的將ADC的數值, 透過**UART2**傳送至PC顯示。
- **UART2的規格請設定為9600 , N , 8 , 1, no flow Control。**
- **UART2 的RX, TX請指定至Pin31(RF4, U2RX), Pin32(RF5, U2TX) 。**

- 該如何開始？

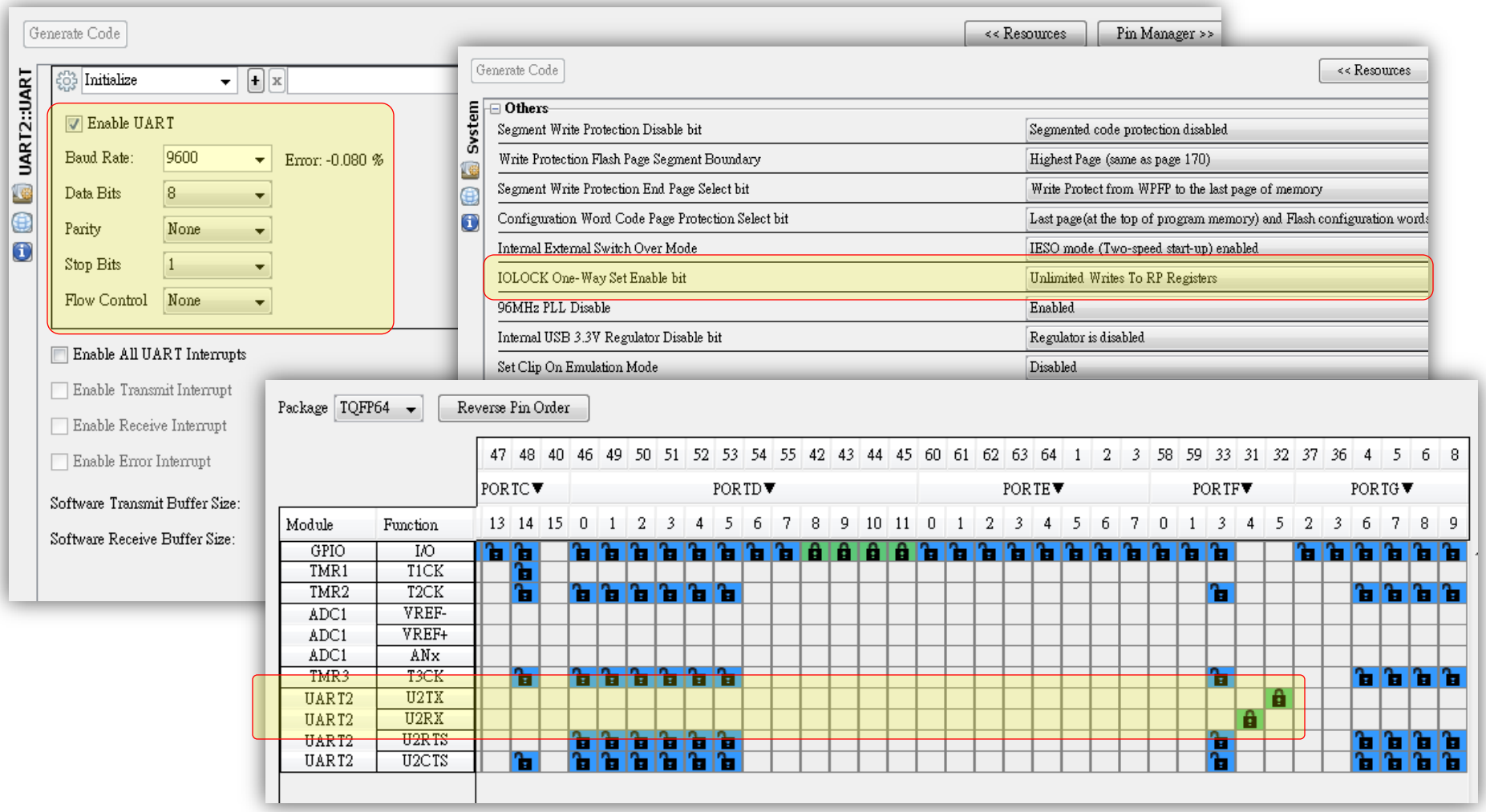
Lab10 UART Tx Polling Step

- 開啟既有專案(C:\Exercises\Lab10 UART Polling.X)
- 開啟MCC, 新增**UART2**資源,根據題目要求完成設定。
- Configuration Bits中的IO On-Way Set 記得設定為Unlimited Writes 。
- 期望能在終端機畫面看到VR1與VR2的ADC數值。



Lab10 UART Tx Polling

MCC's Setting



MICROCHIP
Regional Training Centers

Lab10 UART Tx Polling Code Example

main.c

```
#include <stdio>
void UART_PutRAMString(unsigned char * String);
unsigned char UARTString[ 100 ];

int main(void)
{
   while(1)
   {
      ...
      if(AD1Flag)
      {
         AD1Flag = 0;
         ...
         sprintf( ( char * ) UARTString , "VR1 Value:%4d, VR2 Value:%4d\r\n“,
                                                AD1Buffer[0], AD1Buffer[1]);
         UART_PutRAMString(unsigned char * String);
      }
   }
}
void UART_PutRAMString(unsigned char * String)
{
   while( *String != 0x00 )
   {
      while(U2STAbits.UTXBF );
      U2TXREG = *String++;
   }
}
```

MCC's UART Function & Buffer

- **MCC中的UART Function已經預先建構了傳送與接收的環狀佇列。開啟中斷可以更靈活的運用這些機制, 幾個主要的核心函數如下:**

```
void UART2_Initialize(void)
    // 啟用UART2, 設定UART2工作模式。
void UART2_TasksTransmit(void);
    // 傳送任務維護函數, 在傳送中斷內呼叫, 用以維護傳送工作。
void UART2_TasksReceive(void);
    // 接收任務維護函數, 在接收中斷內呼叫, 用以維護接收工作。
void UART2_Read();
unsigned int UART2_ReadBuffer(*buffer, bufLen)
    // 傳送函數
void UART2_Write(byte);
unsigned int UART2_WriteBuffer(*buffer, bufLen)
    // 接收函數
```

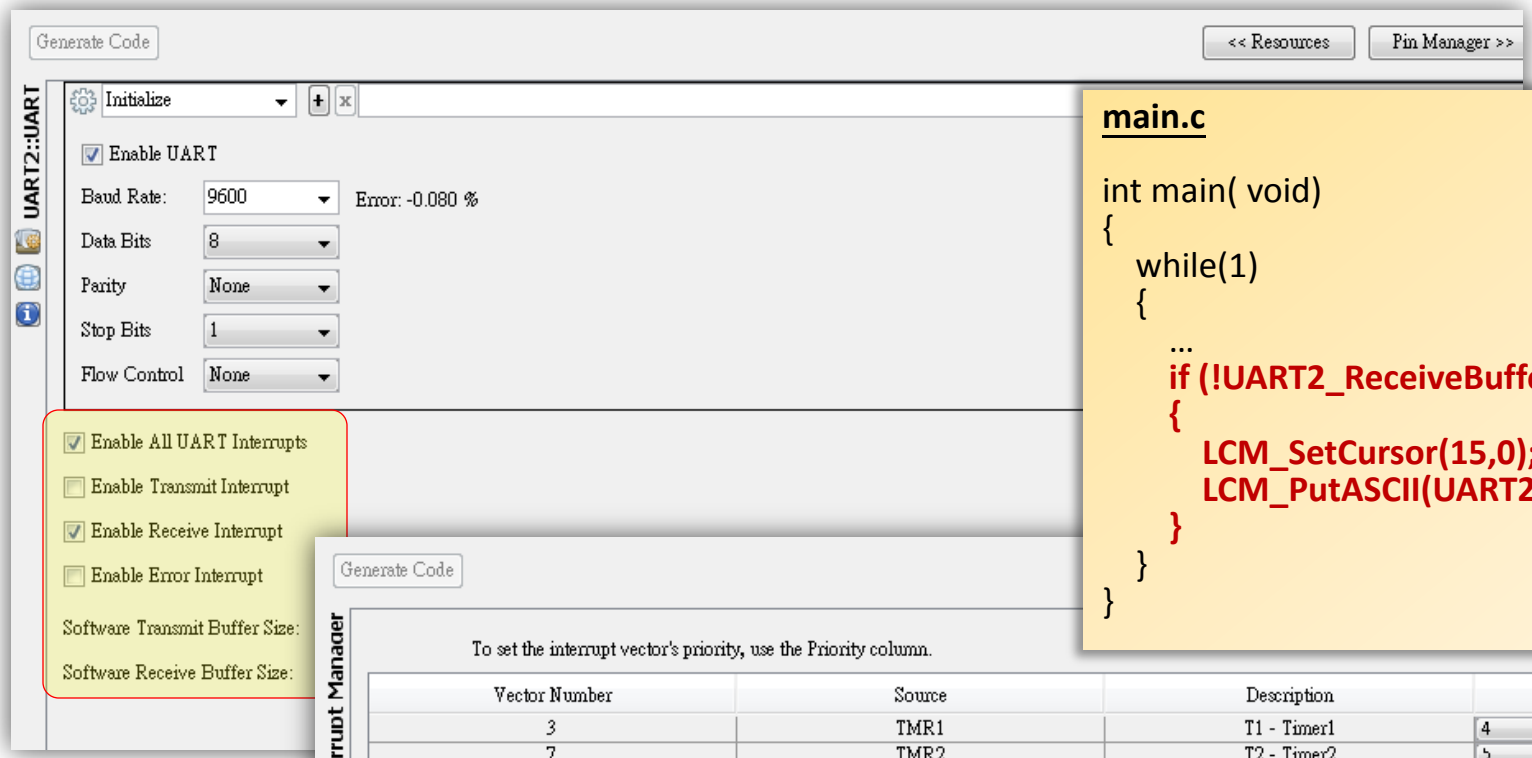
Lab11 UART Tx Polling Rx Interrupt

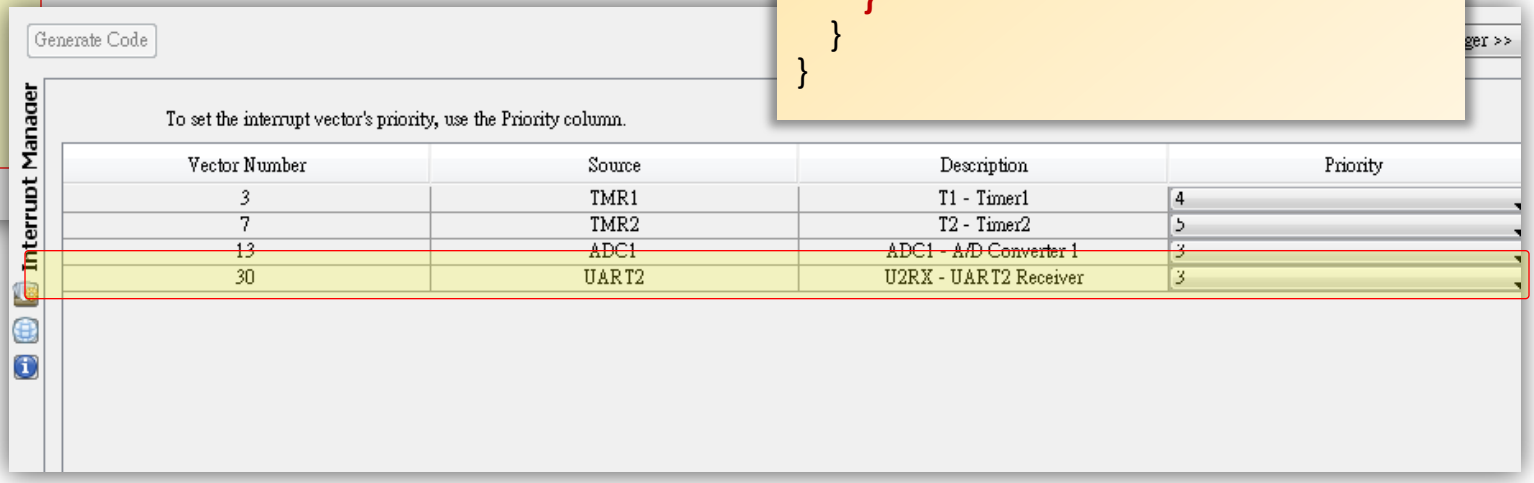
目 標

- 嘗試透過MCC的設定, 在Lab10架構上修改**UART2**的設定。開啟**UART2**的接收中斷。中斷優先權修改為”3”。
- 除了透過**UART2**將文字傳送至PC顯示外。也可接收PC端所輸入的字元, 再將該字元顯示在顯示在LCD Module上。

Lab11 UART Tx Polling Rx Interrupt

MCC's Setting & Code Example



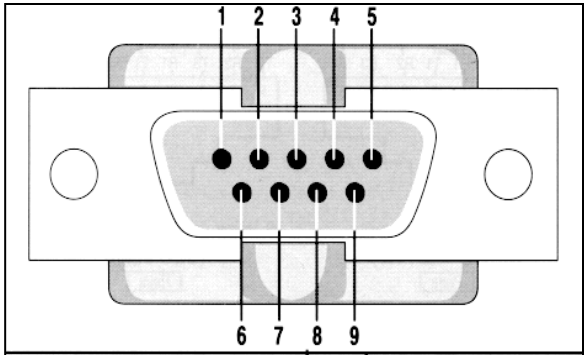


main.c

```
int main( void)
{
   while(1)
   {
      ...
      if (!UART2_ReceiveBufferIsEmpty())
      {
         LCM_SetCursor(15,0);
         LCM_PutASCII(UART2_Read());
      }
   }
}
```

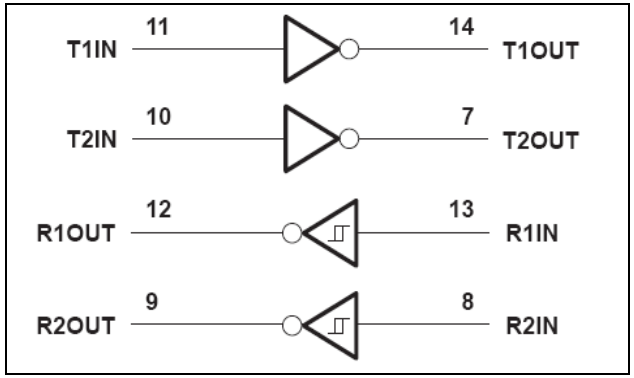
UART & MAX232 Connection

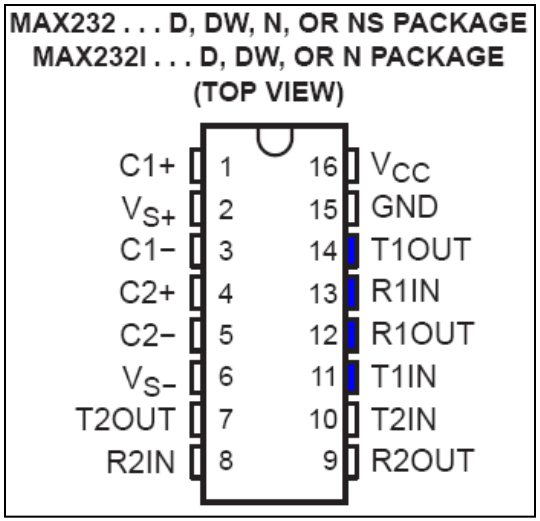
- 也可以使用MAX232來與PC的COM Port連接。
- **PC上的RS232所使用邏輯系統與MCU不同。MCU使用正邏輯系統。其邏輯"1"的電壓準位為0.8 V_{DD} ~ V_{DD};邏輯" 0"則為V_{SS} ~ 0.2V_{DD} 。RS232,使用負邏輯,邏輯"1"的電壓準位為-3~-12V;邏輯"0"則為+3 ~ +12V。**
- MCU與RS232因為邏輯系統與電壓位準均不同,因此無法直接連接,必須透過MAX232作電壓位準的調整。



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

http://www.aggsoft.com/rs232-pinout-cable/serial-cable-connections.htm





UART & MAX232 Connection

- UART, MAX232與RS232連接器之間的線路連接可參考下圖。在APP026-3x上, MAX232的電源, 及電容等均以事先連接完成。但訊號線須由使用者自行以杜邦線連接適當訊號。
- 先使用PPS指定UART2的Rx(RP17,Pin31), Tx(RP10,Pin32)位置, 再利用杜邦線連接PIC24與MAX232, (Rx-R1OUT) , (Tx-T1IN)。
- 接著,利用杜邦線連接MAX232與DB9 Connecter, (R1IN- Pin3), (T1OUT-Pin2) 。
- CTS (Clear to Send),RTS (Request to Send)為進行流量控制時所必須使用之訊號。如果需要流量控制時,則必須將CTS, RTS依參考線路所示連接。