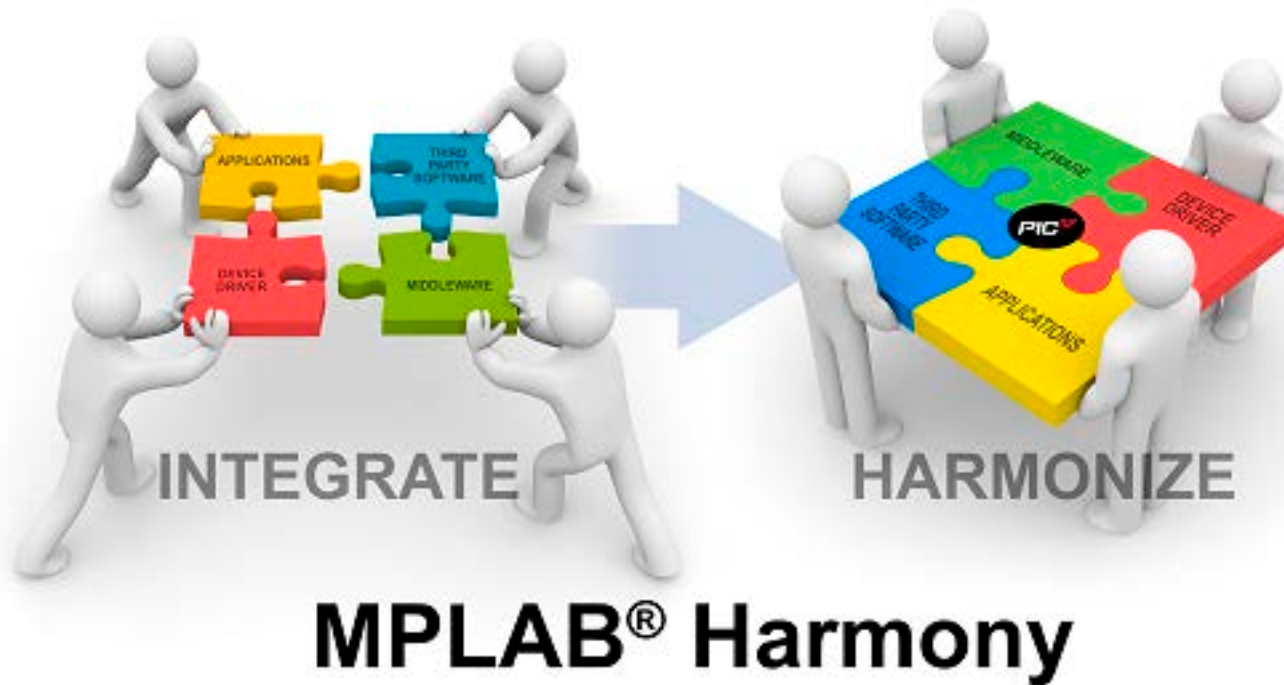




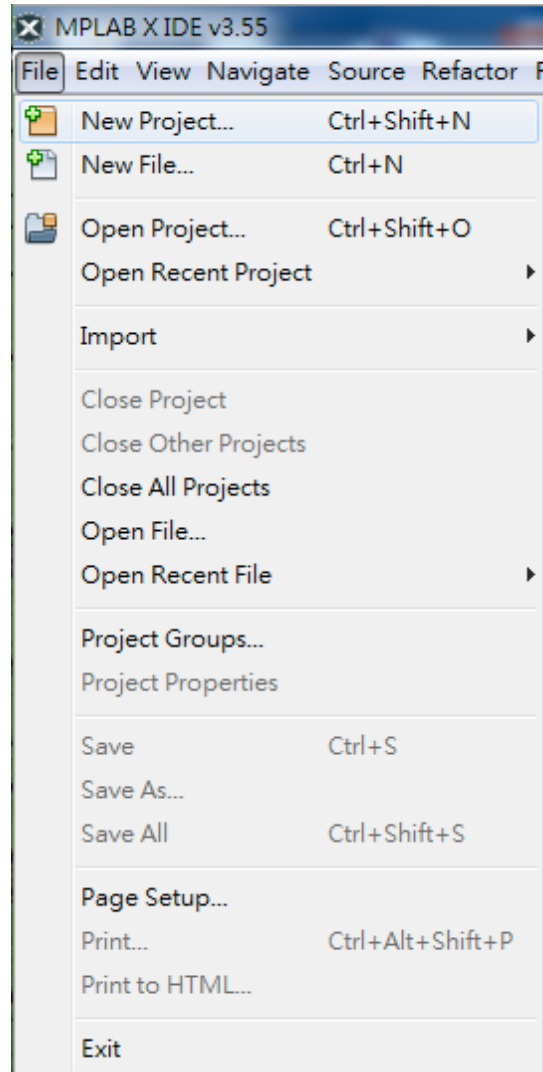
**MICROCHIP**  
**PIC32MX470 Curiosity Board**  
**&**  
**APP043**



- **MCU 32 Harmony 開發環境**
  - MPLBA X v3.55
  - XC32 v1.42
- **MCU32 Curiosity 開發板**
- **APP043**
- **參考資料**
  - PIC32MX330/350/370/430/450/470 (60001185F)

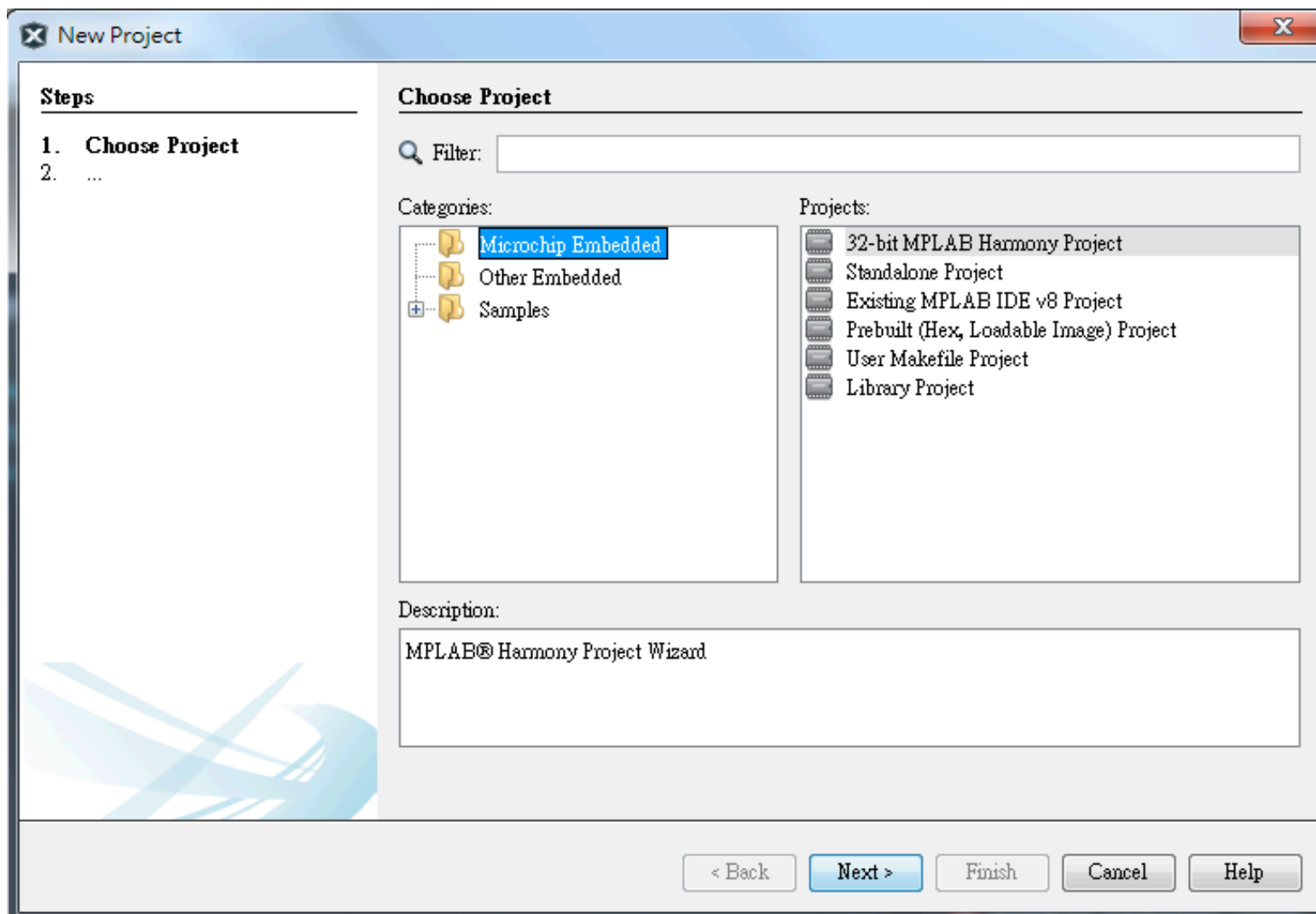
# 建立 Harmony 專案

## Step 1：開啟新的專案



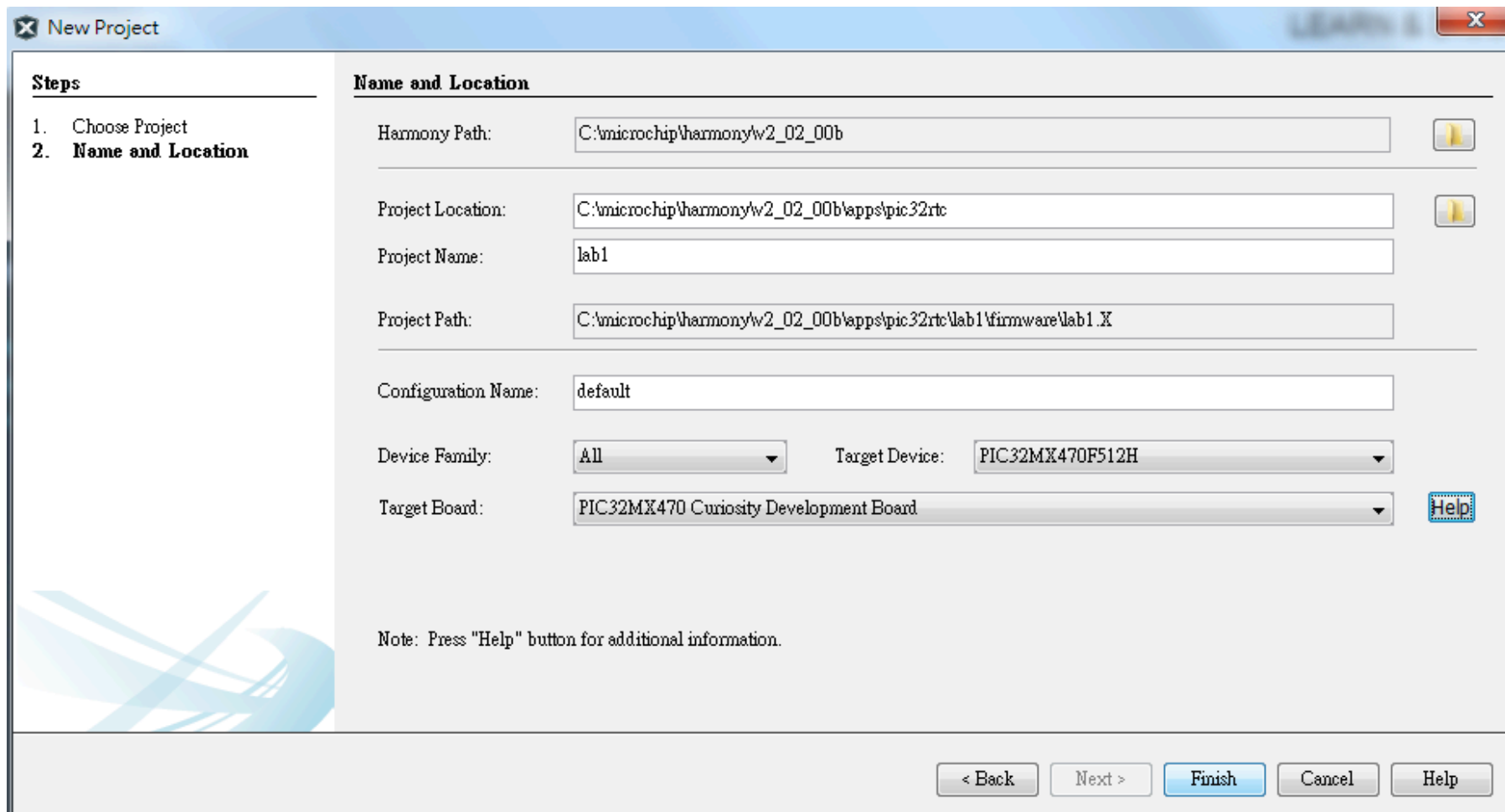
# 建立 Harmony 專案

## Step 2 : 選擇專案型別 “32-bit MPLAB Harmony Project”



# 建立 Harmony 專案

## Step 3：設定 Harmony 版本、專案名稱、路徑、MCU32



**New Project**

**Steps**

1. Choose Project
2. **Name and Location**

**Name and Location**

Harmony Path: C:\microchip\harmony\w2\_02\_00b

Project Location: C:\microchip\harmony\w2\_02\_00b\apps\pic32rtc

Project Name: lab1

Project Path: C:\microchip\harmony\w2\_02\_00b\apps\pic32rtc\lab1\firmware\lab1.X

Configuration Name: default

Device Family: All Target Device: PIC32MX470F512H

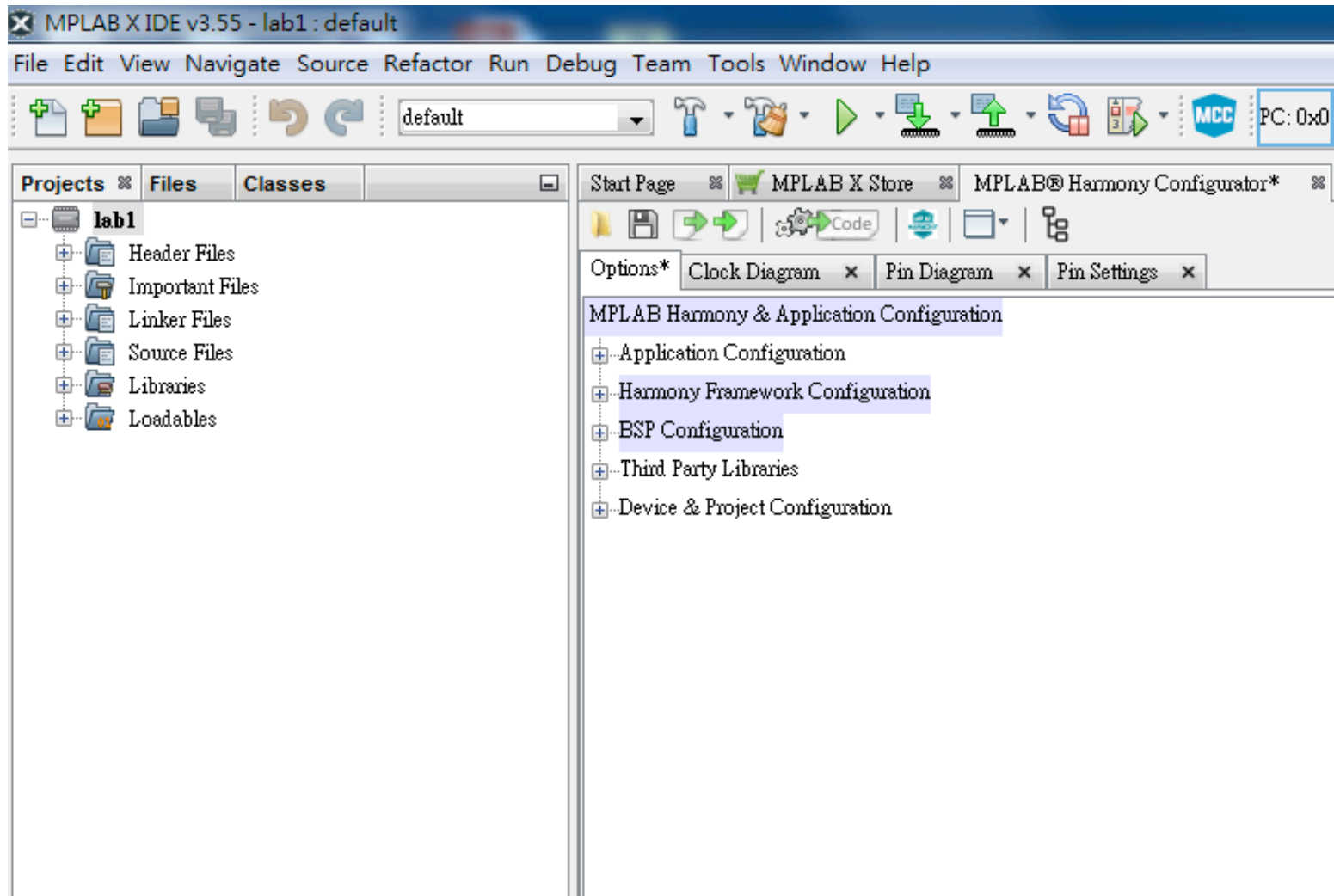
Target Board: PIC32MX470 Curiosity Development Board

Note: Press "Help" button for additional information.

< Back Next > Finish Cancel Help

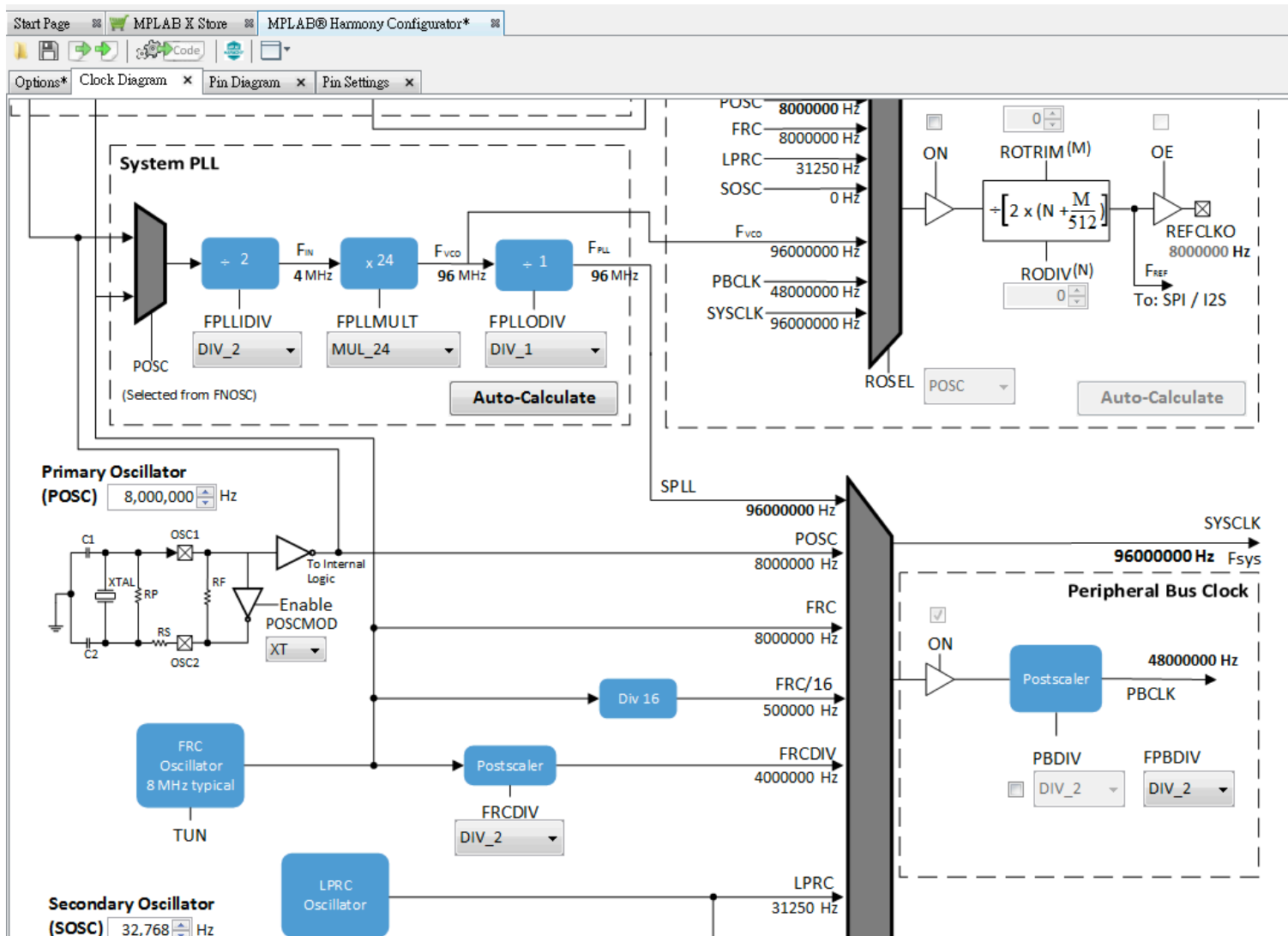
# 建立 Harmony 專案

## Step 4：成功建立Harmony專案框架



# 建立 Harmony 專案

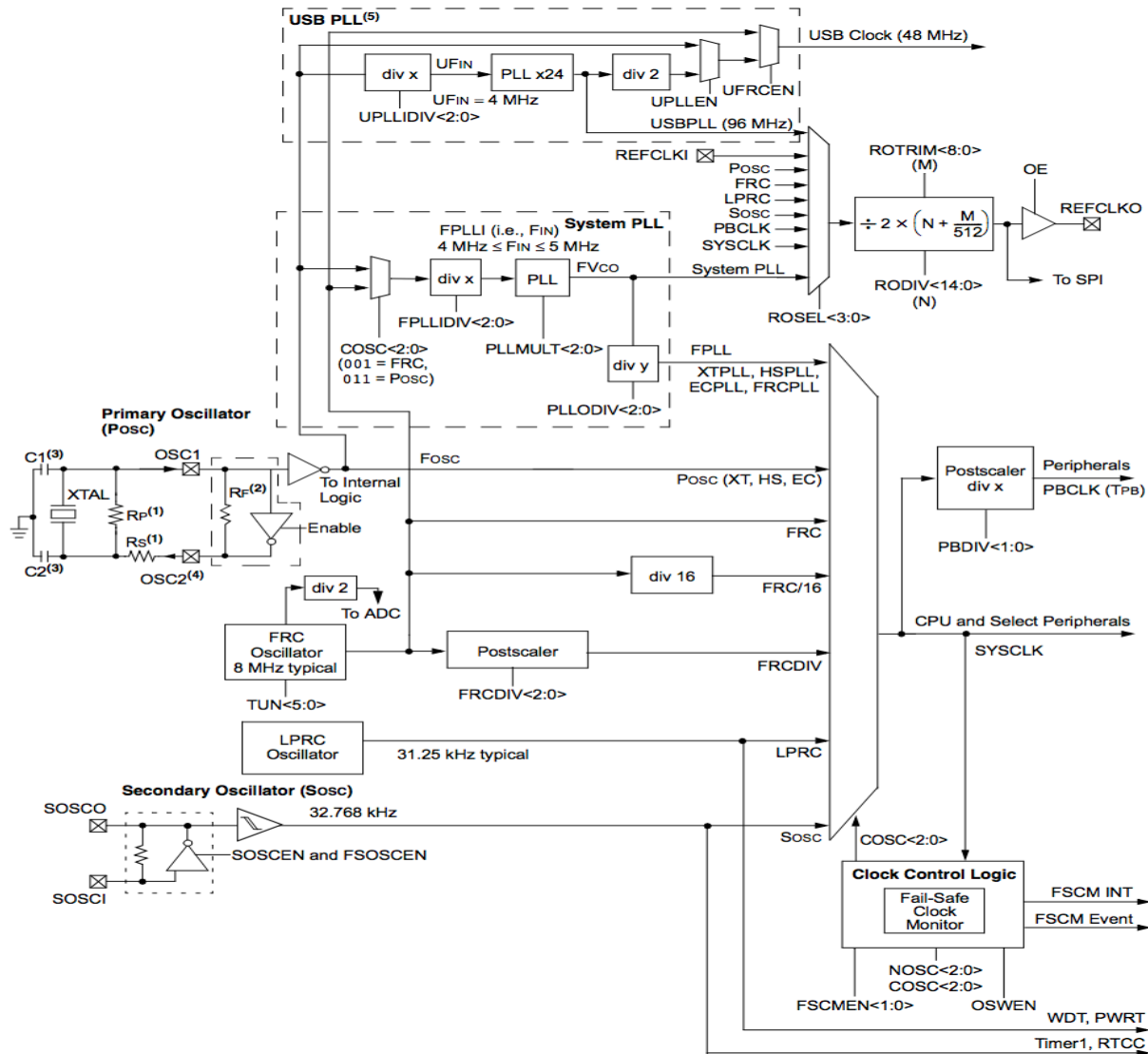
## Step 5 : MHC – Clock Diagram





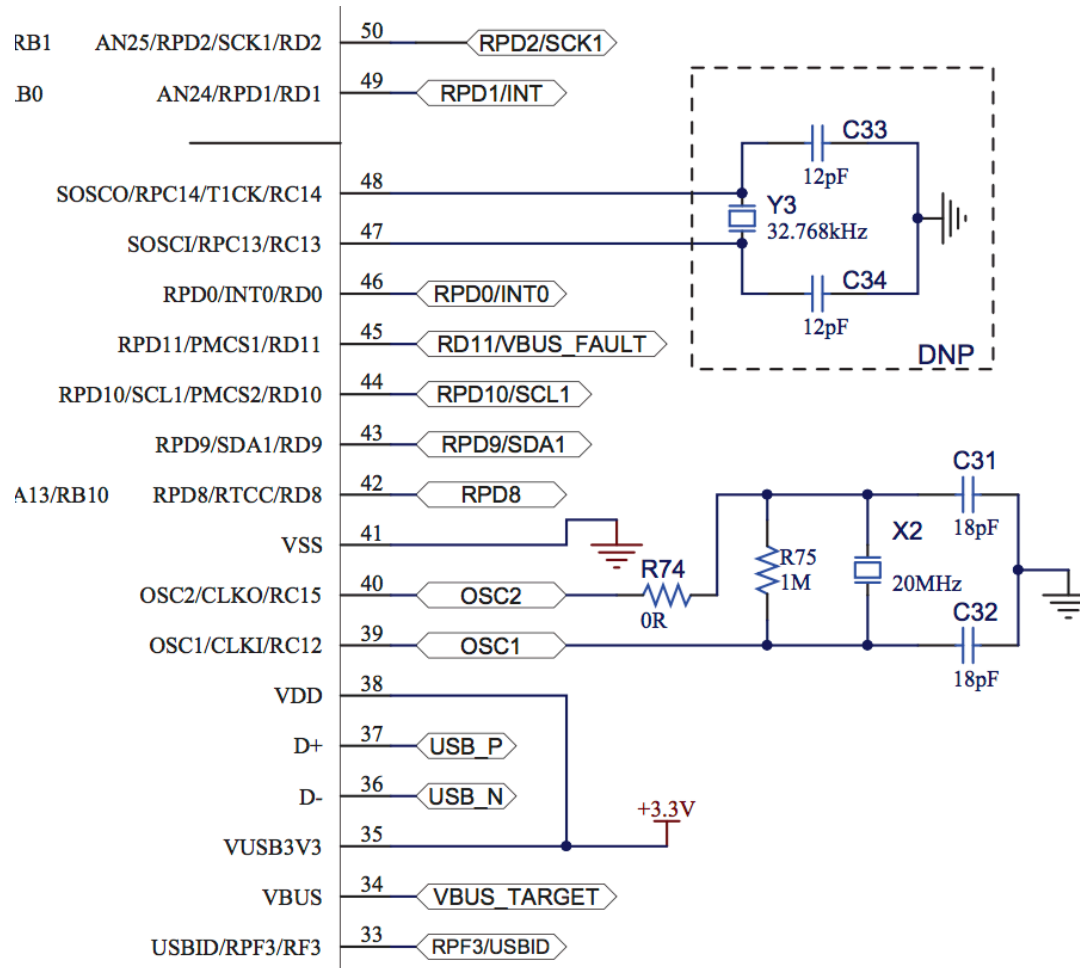
# 建立 Harmony 專案

## Step 5 : PIC32MX470 CLK 方塊圖



# 建立 Harmony 專案

## Step 5 : PIC32 Curiosity board - CLK

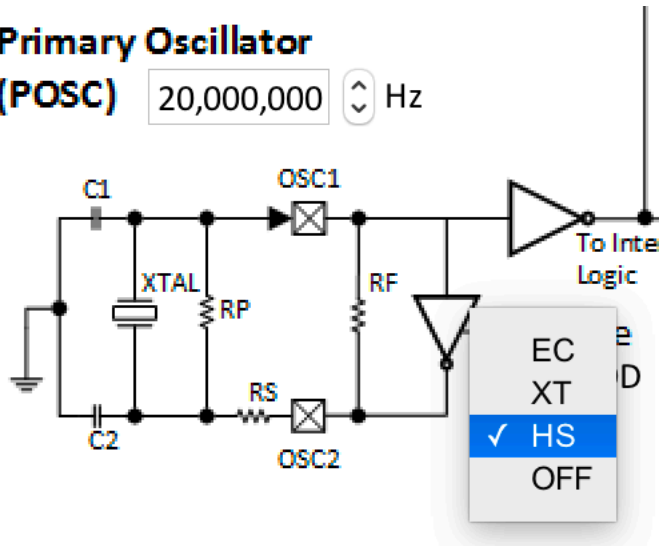


# 建立 Harmony 專案

## Step 5：選擇震盪頻率類別、系統時脈來源

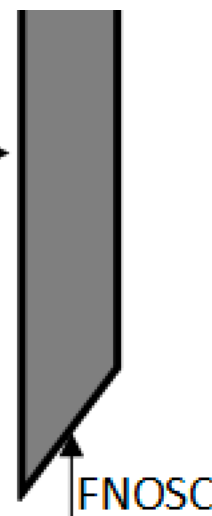
**Primary Oscillator  
(POSC)**

20,000,000 Hz



EC  
XT  
✓ HS  
OFF

OSC  
0 Hz

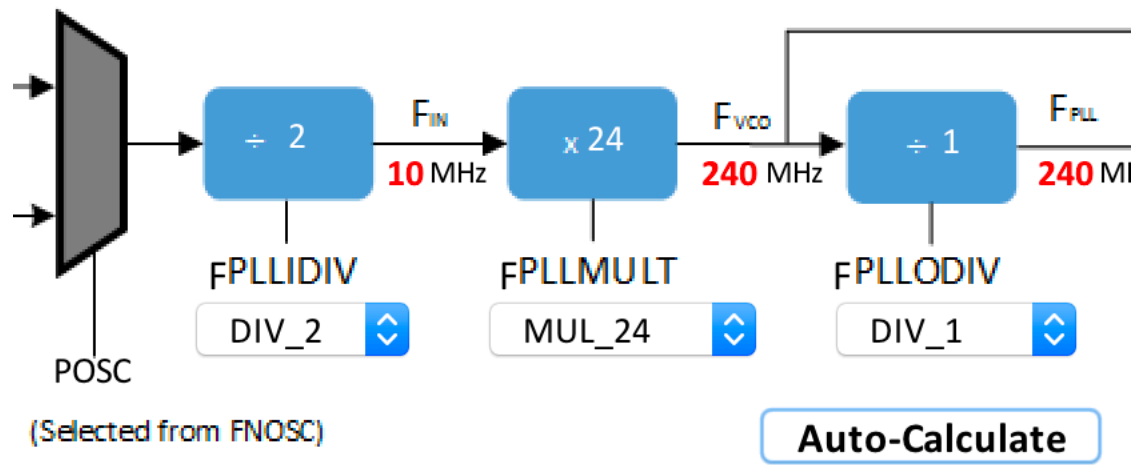


- FRCPLL
  - POSC
  - ✓ PRIPLL
  - SOSC
  - LPFRC
  - FRC/16
  - FRCDIV
- PRIPLL

# 建立 Harmony 專案

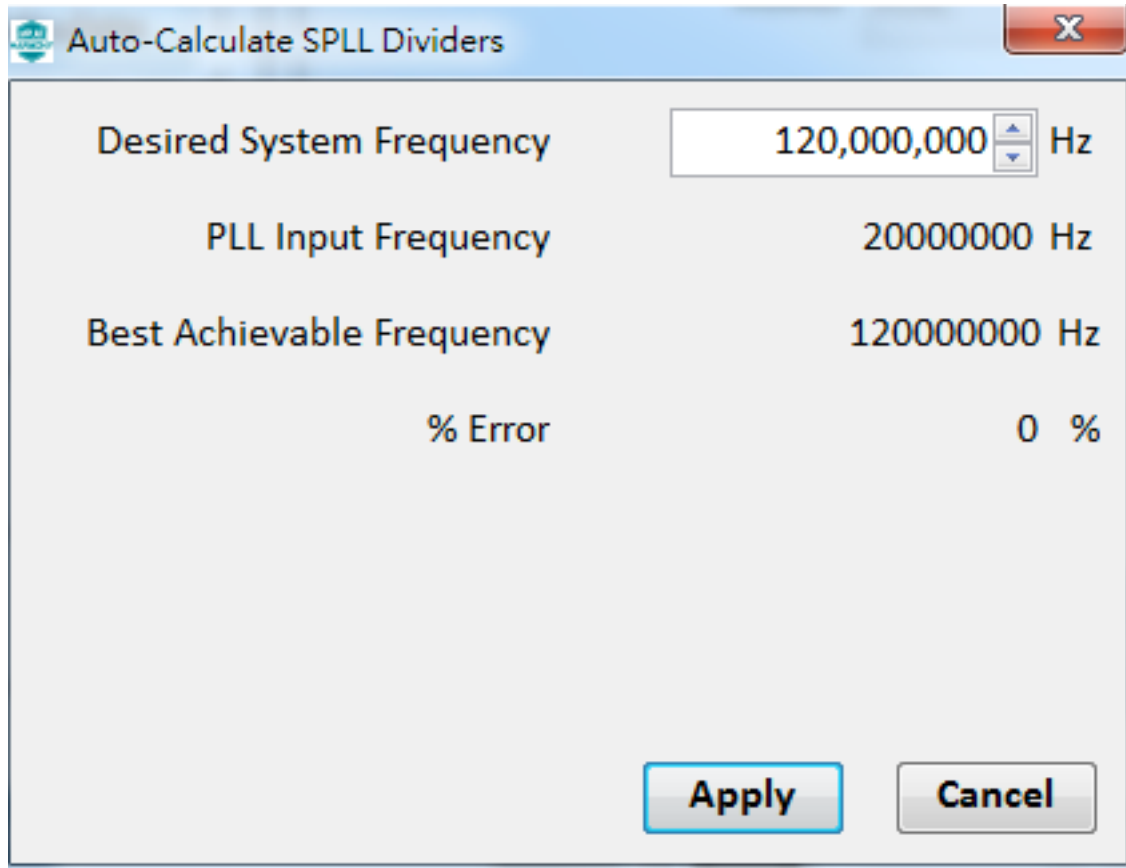
## Step 6：設定系統工作頻率

### System PLL



# 建立 Harmony 專案

## Step 7：設定系統工作頻率為120MHz



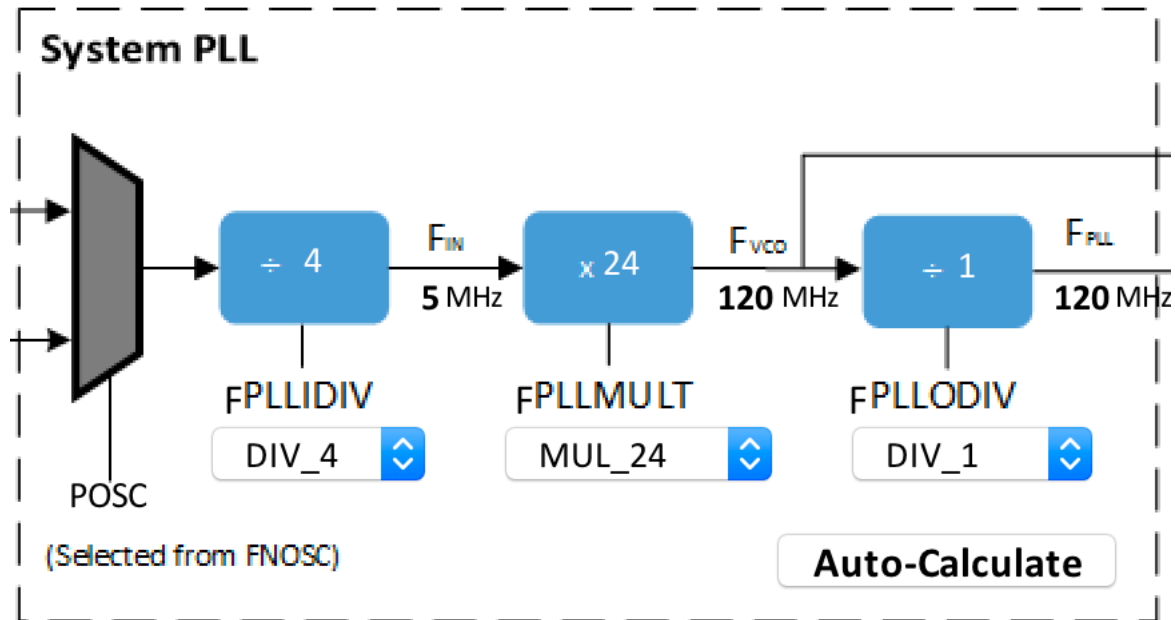
The dialog box titled "Auto-Calculate SPL Dividers" contains the following fields and values:

Field	Value	Unit
Desired System Frequency	120,000,000	Hz
PLL Input Frequency	200000000	Hz
Best Achievable Frequency	1200000000	Hz
% Error	0	%

Buttons: Apply, Cancel

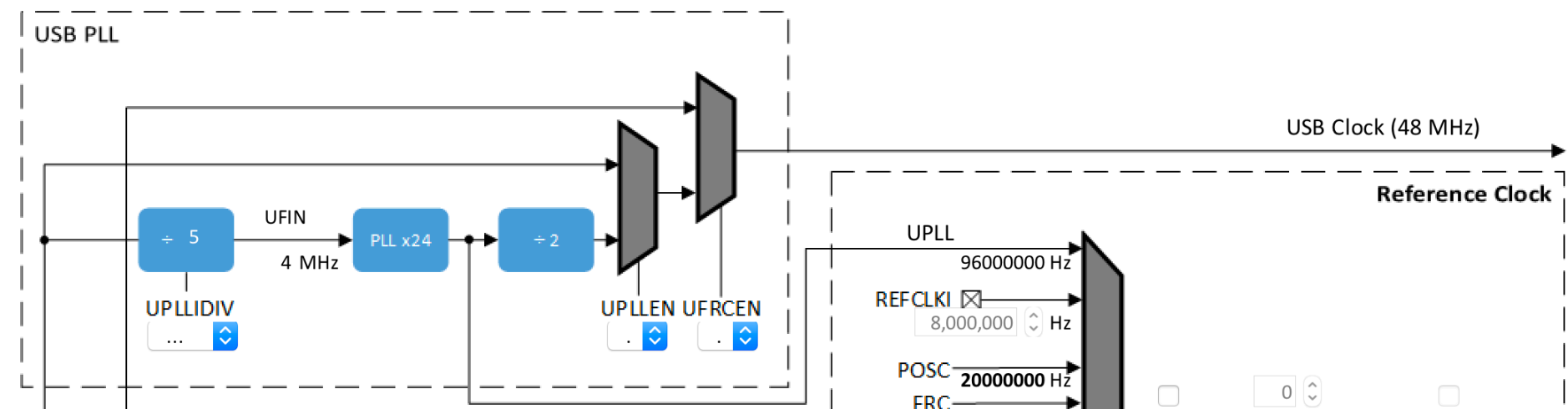
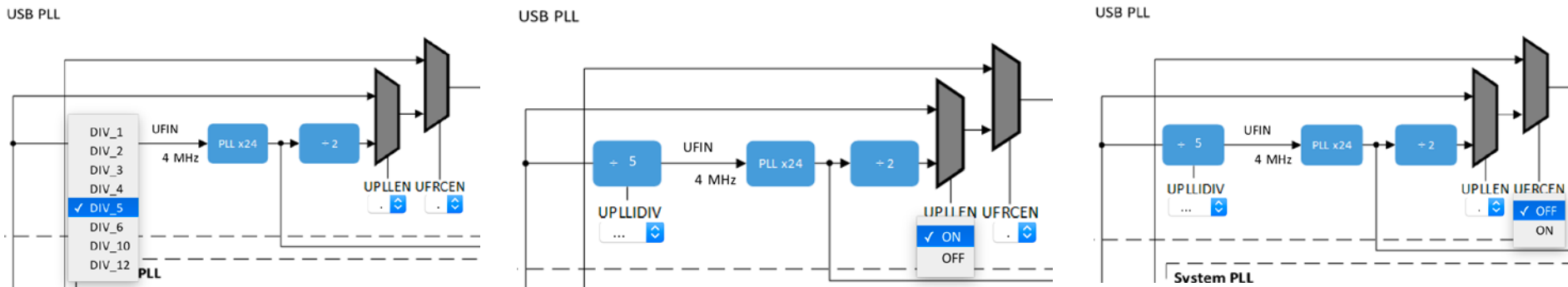
# 建立 Harmony 專案

## Step 8：系統工作頻率為120MHz



# 建立 Harmony 專案

## Step 9：設定USB工作頻率為48MHz

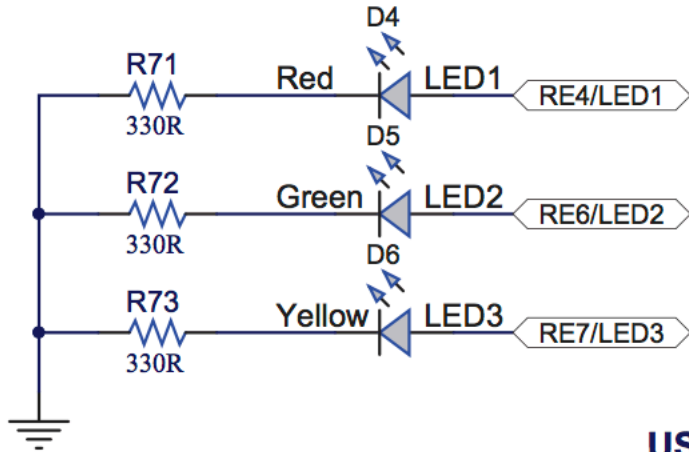






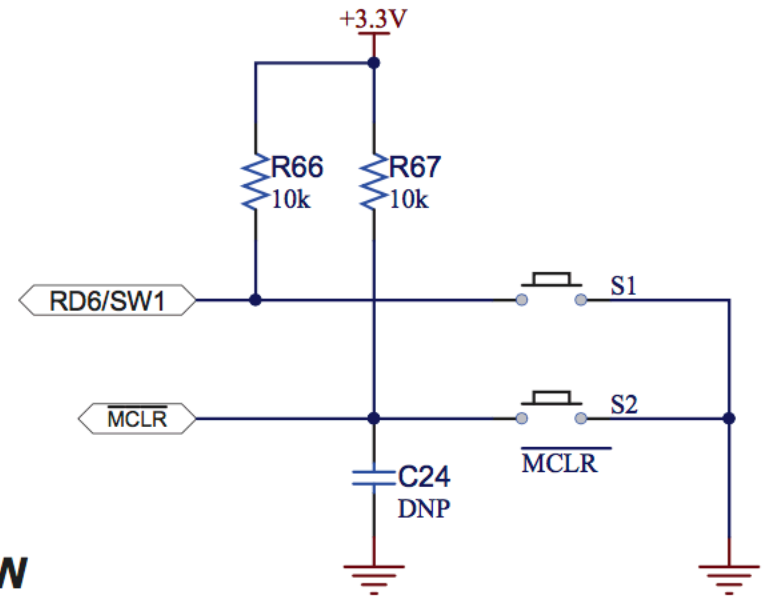
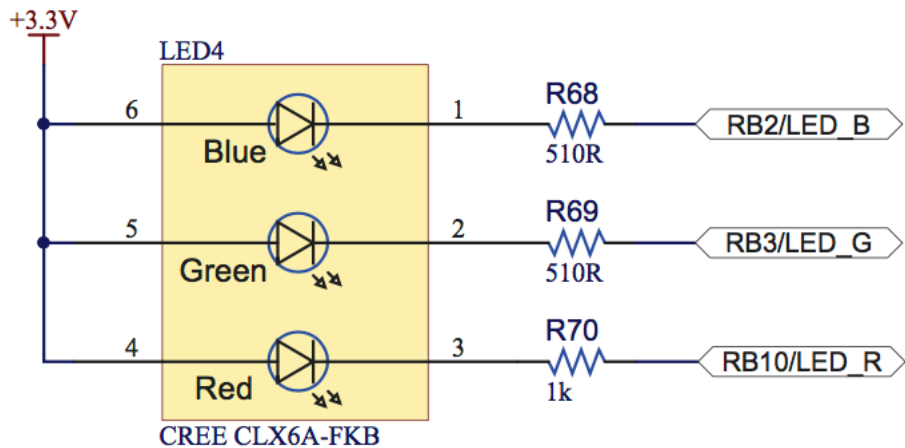
# 建立 Harmony 專案

## Step 10 : PIC32 Curiosity board -> GPIO 設定



**USER LEDs**

**CREE LED**



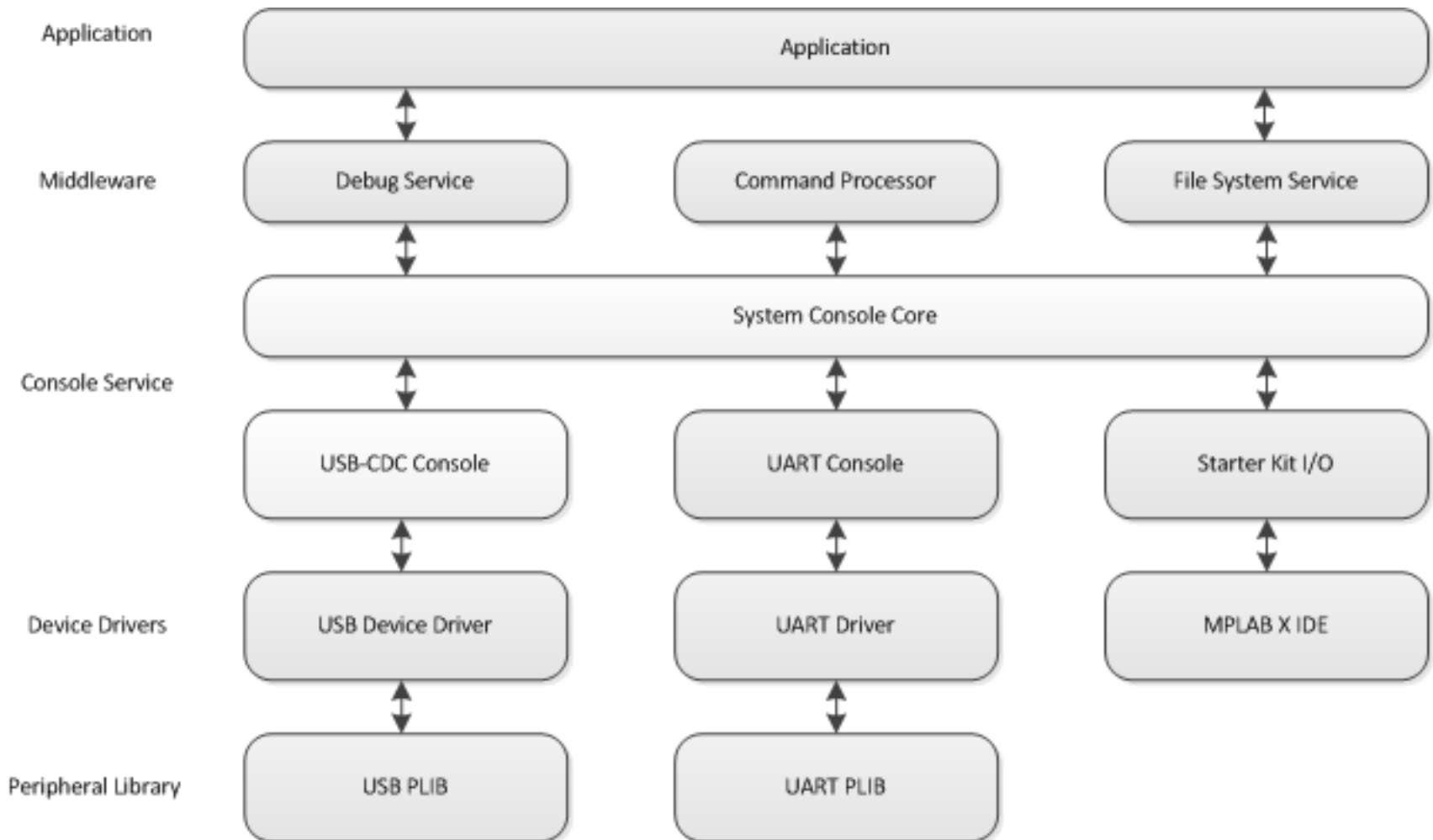
**SW**

# 建立 Harmony 專案

## Step 10 : MHC – Pin Settings -> GPIO 設定

Pin Number	Pin ID	Voltage Tolerance	Name	Function	Direction (TRIS)	Latch (LAT)	Open Drain (ODC)	Mode (ANSEL)	Change Notification (CNEN)	Pull Up (CNPU)	Pull Down (CNPD)
1	RE5			Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Analog"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2	RE6		BSP_LED_2	LED_AH	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3	RE7		BSP_LED_3	LED_AH	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4	RG6			Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Analog"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5	RG7			Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Analog"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
6	RG8			Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Analog"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
7	MCLR	5V			<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
8	RG9			Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Analog"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
9	VSS				<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
10	VDD				<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
11	RB5		USB_VBUS_SWITCH	VBUS	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
12	RB4			Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Analog"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
13	RB3			Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Analog"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
14	RB2			Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Analog"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
15	RB1		BSP_BM64_WAKEUP	GPIO_OUT	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
16	RB0		BSP_BM64_RST	GPIO_OUT	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
54	RD6	5V	BSP_SWITCH_1	SWITCH	<input type="button" value="In"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
55	RD7	5V	BSP_STBY_RST	GPIO_OUT	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
56	VCAP				<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
57	VDD				<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
58	RF0	5V		Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
59	RF1	5V		Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
60	RE0	5V	BSP_RGB_LED_RED	LED_AL	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
61	RE1	5V	BSP_RGB_LED_GREEN	LED_AL	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
62	RE2		BSP_RGB_LED_BLUE	LED_AL	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
63	RE3	5V		Available	<input type="button" value="In"/>	<input type="button" value="n/a"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
64	RE4		BSP_LED_1	LED_AH	<input type="button" value="Out"/>	<input type="button" value="Low"/>	<input type="checkbox"/>	<input type="button" value="Digital"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

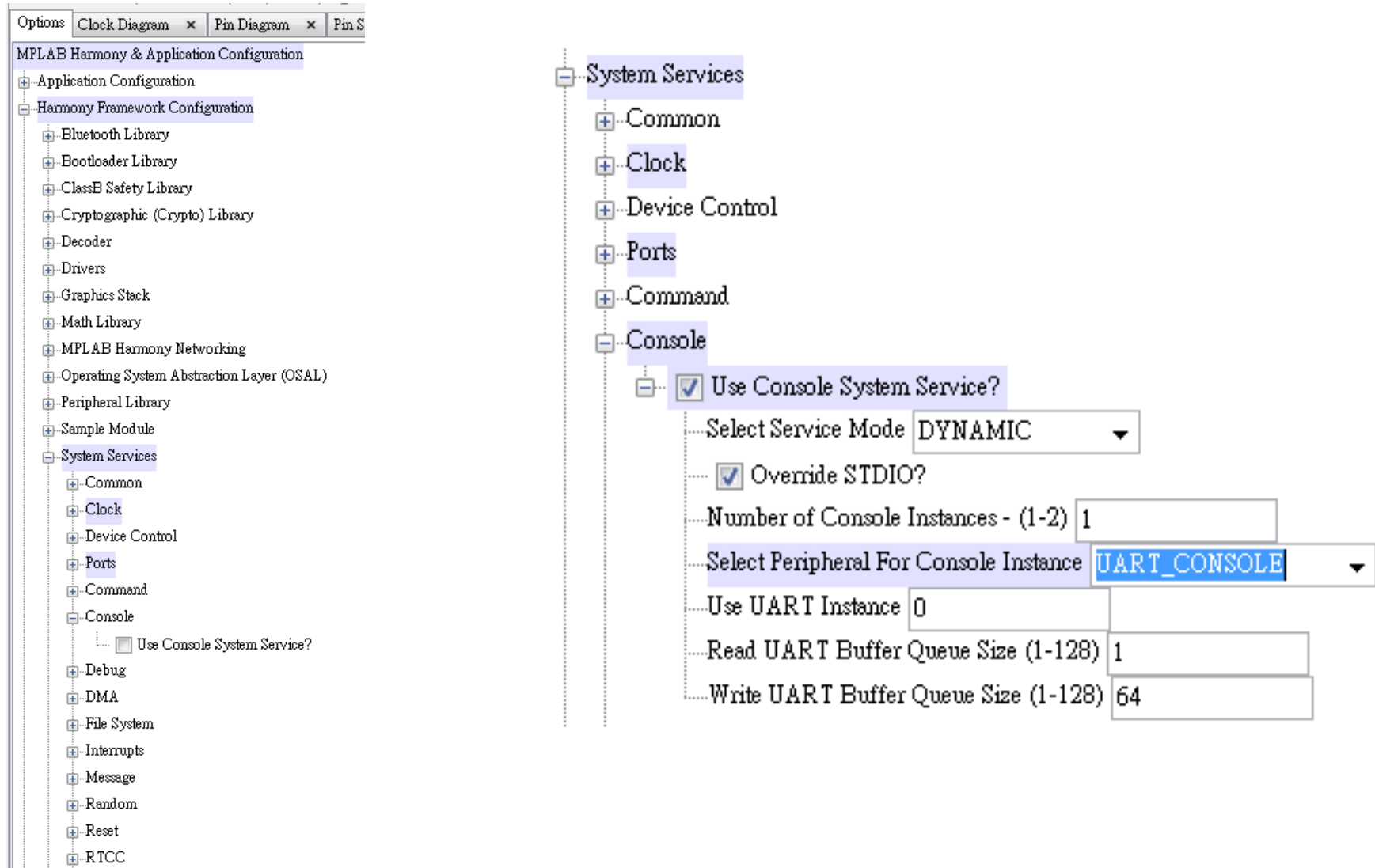
# Console Service Software Abstraction Block Diagram



# Generate “Console system service”

## Step 1 : Harmony Framework Configuration

### -> System Services -> Console



The screenshot displays the MPLAB Harmony & Application Configuration tool interface. The left sidebar shows a tree view of the configuration hierarchy, with 'System Services' expanded to show 'Console'. The main area shows the configuration options for the 'Console' service.

**Options:** Clock Diagram x Pin Diagram x Pin S

**MPLAB Harmony & Application Configuration**

- Application Configuration
- Harmony Framework Configuration
  - Bluetooth Library
  - Bootloader Library
  - ClassB Safety Library
  - Cryptographic (Crypto) Library
  - Decoder
  - Drivers
  - Graphics Stack
  - Math Library
  - MPLAB Harmony Networking
  - Operating System Abstraction Layer (OSAL)
  - Peripheral Library
  - Sample Module
  - System Services
    - Common
    - Clock
    - Device Control
    - Ports
    - Command
    - Console
      - ☒ Use Console System Service?
    - Debug
    - DMA
    - File System
    - Interrupts
    - Message
    - Random
    - Reset
    - RTCC

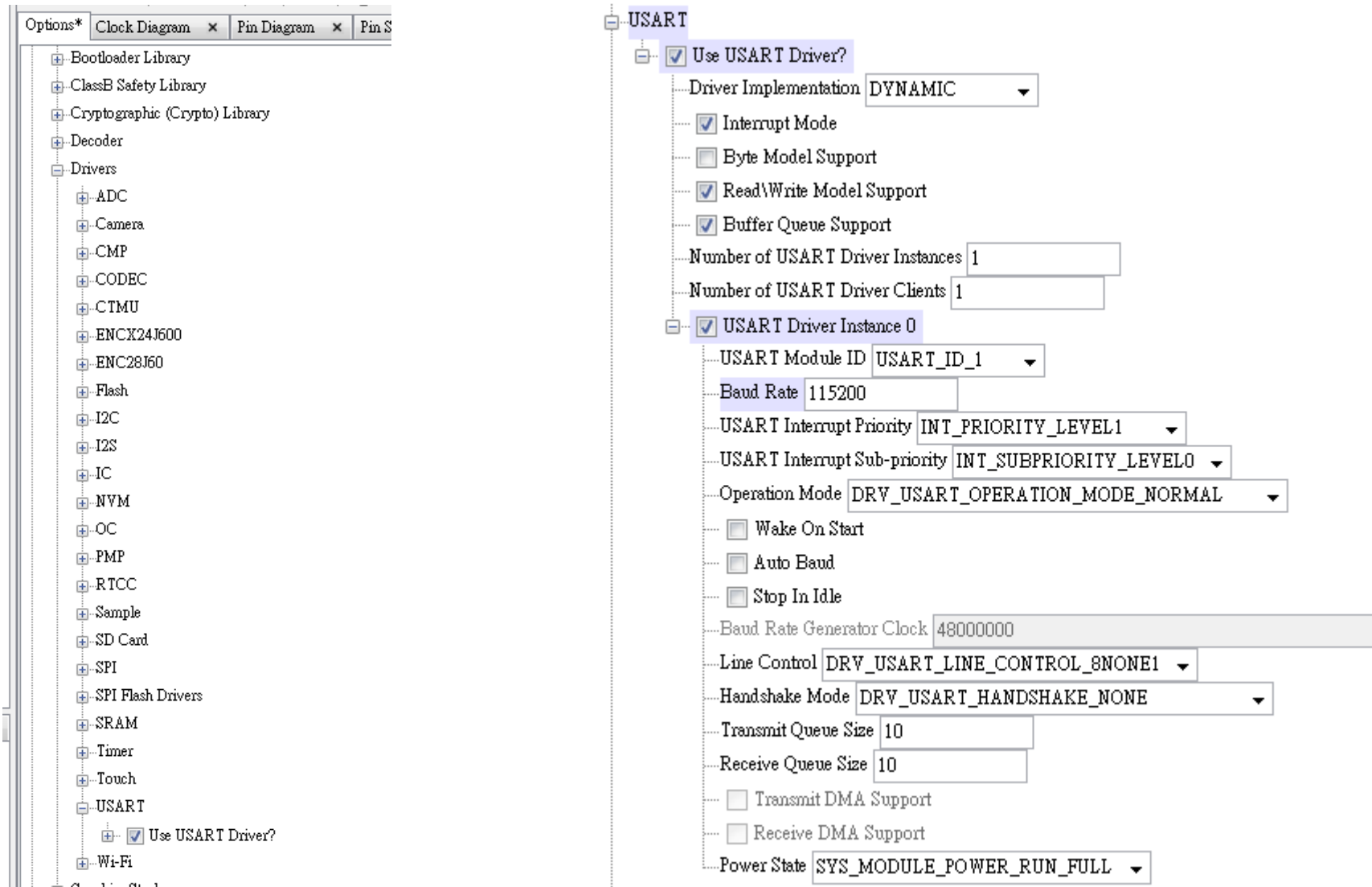
**System Services Configuration:**

- Common
- Clock
- Device Control
- Ports
- Command
- Console
  - ☒ Use Console System Service?
    - Select Service Mode: DYNAMIC
    - ☒ Override STDIO?
    - Number of Console Instances - (1-2): 1
    - Select Peripheral For Console Instance: UART\_CONSOLE
    - Use UART Instance: 0
    - Read UART Buffer Queue Size (1-128): 1
    - Write UART Buffer Queue Size (1-128): 64

# Generate “Console system service”

## Step 2 : Harmony Framework Configuration

### -> Drivers -> USART



The screenshot displays the Microchip Harmony Framework Configuration tool interface. On the left, a tree view shows the project structure, with the 'USART' driver selected under the 'Drivers' category. The 'Use USART Driver?' checkbox is checked. On the right, the configuration details for the USART driver are shown, including the driver implementation, interrupt mode, and various operational parameters.

**Options\*** Clock Diagram x Pin Diagram x Pin S

**Drivers**

- ADC
- Camera
- CMP
- CODEC
- CTMU
- ENCX24J600
- ENC28J60
- Flash
- I2C
- I2S
- IC
- NVM
- OC
- PMP
- RTCC
- Sample
- SD Card
- SPI
- SPI Flash Drivers
- SRAM
- Timer
- Touch
- USART**
  - ☒ Use USART Driver?
- Wi-Fi

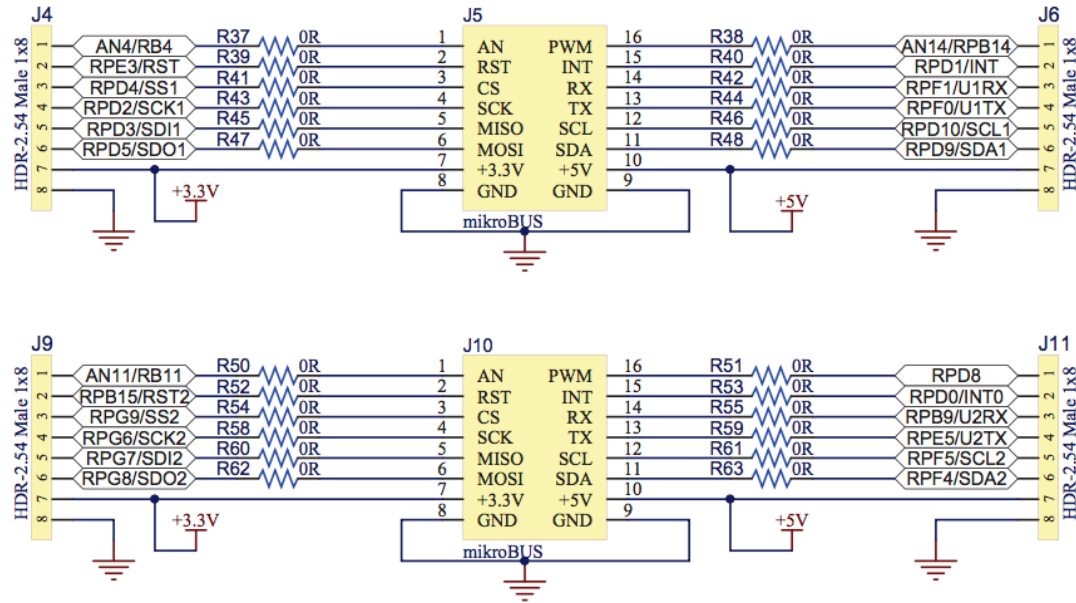
**USART**

- ☒ Use USART Driver?
  - Driver Implementation: DYNAMIC
  - ☒ Interrupt Mode
  - ☐ Byte Model Support
  - ☒ Read/Write Model Support
  - ☒ Buffer Queue Support
  - Number of USART Driver Instances: 1
  - Number of USART Driver Clients: 1
- ☒ USART Driver Instance 0
  - USART Module ID: USART\_ID\_1
  - Baud Rate: 115200
  - USART Interrupt Priority: INT\_PRIORITY\_LEVEL1
  - USART Interrupt Sub-priority: INT\_SUBPRIORITY\_LEVEL0
  - Operation Mode: DRV\_USART\_OPERATION\_MODE\_NORMAL
  - ☐ Wake On Start
  - ☐ Auto Baud
  - ☐ Stop In Idle
  - Baud Rate Generator Clock: 48000000
  - Line Control: DRV\_USART\_LINE\_CONTROL\_8NONE1
  - Handshake Mode: DRV\_USART\_HANDSHAKE\_NONE
  - Transmit Queue Size: 10
  - Receive Queue Size: 10
  - ☐ Transmit DMA Support
  - ☐ Receive DMA Support
  - Power State: SYS\_MODULE\_POWER\_RUN\_FULL

# Generate “Console system service”

## Step 5 : Harmony Framework Configuration

### -> Pin Table



**MikroBUS HDRs**

Search Results

Output

MPLAB® Harmony Configurator\*

Output

Pin Table

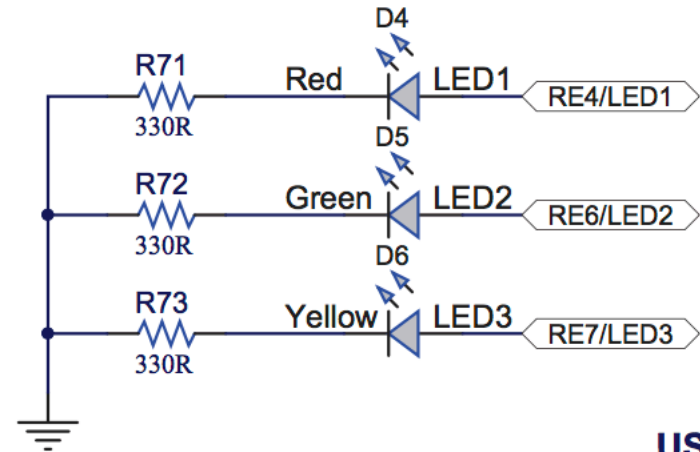
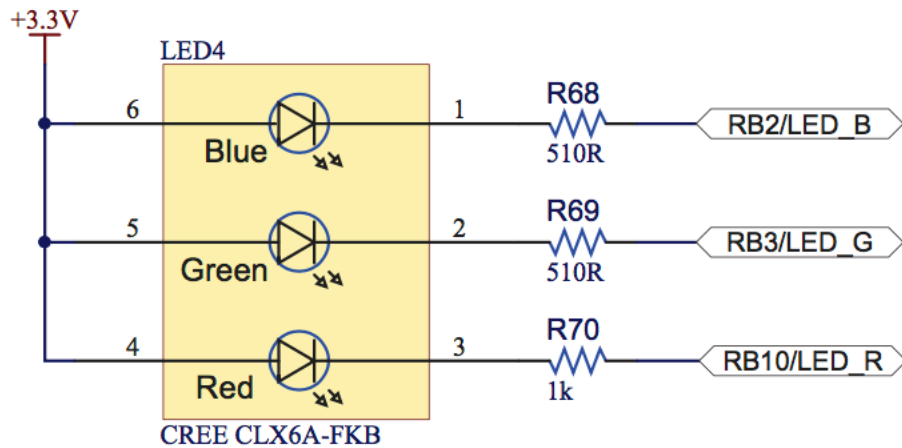
Package: QFN

Module	Function	RB8	RB9	RB10	RB11	VSS	VDD	RB12	RB13	RB14	RB15	RF4	RF5	RF3	VBUS	VUSE3V...	D-	D+	VDD	RC12	RC15	VSS	RD8	RD9	RD10	RD11	RD0	RC13	RC14	RD1	RD2	RD3	RD4	RD5	ESP_SW...	ESP_ST...	VCAP	VDD	U1TX	U1RX	ESP_RC...	
	TRD3																																									
UART 1 (USART_ID_1)	U1RX																																									
	U1TX																																									
	U1CTS																																									
	U1RTS																																									

# LED 點亮及關閉

## Step 3：檢查電路及BSP的設定

### CREE LED



**USER LEDs**

# LED 點亮及關閉

## Step 1 : bsp.c 的函式

```
void BSP_LEDOn(BSP_LED led)
{
    if(led_active_level_map[led] == BSP_LED_ACTIVE_HIGH)
    {
        PLIB_PORTS_PinSet( PORTS_ID_0, led_port_channel_map[led], led_port_bit_pos_map[led] );
    }
    else
    {
        PLIB_PORTS_PinClear( PORTS_ID_0, led_port_channel_map[led], led_port_bit_pos_map[led] );
    }
}

void BSP_LEDOff(BSP_LED led)
{
    if(led_active_level_map[led] == BSP_LED_ACTIVE_HIGH)
    {
        PLIB_PORTS_PinClear( PORTS_ID_0, led_port_channel_map[led], led_port_bit_pos_map[led] );
    }
    else
    {
        PLIB_PORTS_PinSet( PORTS_ID_0, led_port_channel_map[led], led_port_bit_pos_map[led] );
    }
}
```



# LED 點亮及關閉

## Step 1 : bsp.h 的函式

---

```
typedef enum
{
    BSP_LED_2 = 0,
    BSP_LED_3 = 1,
    BSP_RGB_LED_GREEN = 2,
    BSP_RGB_LED_BLUE = 3,
    BSP_RGB_LED_RED = 4,
    BSP_LED_1 = 5
} BSP_LED;
```

```
static const PORTS_BIT_POS led_port_bit_pos_map[] =
{
    PORTS_BIT_POS_6,
    PORTS_BIT_POS_7,
    PORTS_BIT_POS_3,
    PORTS_BIT_POS_2,
    PORTS_BIT_POS_10,
    PORTS_BIT_POS_4
};
```

# LED 點亮及關閉

## Step 1 : main.c 加入 BSP\_LEDOn() / BSP\_LEDOff()

```
#include <stddef.h>           // Defines NULL
#include <stdbool.h>          // Defines true
#include <stdlib.h>           // Defines EXIT_FAILURE
#include "system/Common/sys_module.h" // SYS function prototypes
#include "bsp.h"

int main ( void )
{
    /* Initialize all MPLAB Harmony modules, including application(s). */
    SYS_Initialize ( NULL );

    while ( true )
    {
        /* Maintain state machines of all polled MPLAB Harmony modules. */
        BSP_LEDOn(BSP_LED_1);
        BSP_LEDOn(BSP_LED_2);
        BSP_LEDOn(BSP_LED_3);
        BSP_LEDOff(BSP_LED_1);
        BSP_LEDOff(BSP_LED_2);
        BSP_LEDOff(BSP_LED_3);

        BSP_LEDOn(BSP_RGB_LED_RED);
        BSP_LEDOn(BSP_RGB_LED_GREEN);
        BSP_LEDOn(BSP_RGB_LED_BLUE);
        BSP_LEDOff(BSP_RGB_LED_RED);
        BSP_LEDOff(BSP_RGB_LED_GREEN);
        BSP_LEDOff(BSP_RGB_LED_BLUE);

        SYS_Tasks ( );
    }
    /* Execution should not come here during normal operation */
    return ( EXIT_FAILURE );
}
```

# Generate “Console system service”

## Step 2 : app.c -> APP\_Tasks

```
ssize_t nr;  
char myBuffer[] = "message";  
  
case APP_STATE_SERVICE_TASKS:  
{  
    nr = SYS_CONSOLE_Write( SYS_CONSOLE_INDEX_0, STDOUT_FILENO, myBuffer, strlen(myBuffer) );  
    if (nr != strlen(myBuffer))  
    {  
        // Handle error  
    }  
    appData.state = APP_STATE_IDLE;  
    break;  
}  
case APP_STATE_IDLE:  
  
    break;
```

# Generate “Console system service”

## Step 2 : app.h -> APP\_STATES

---

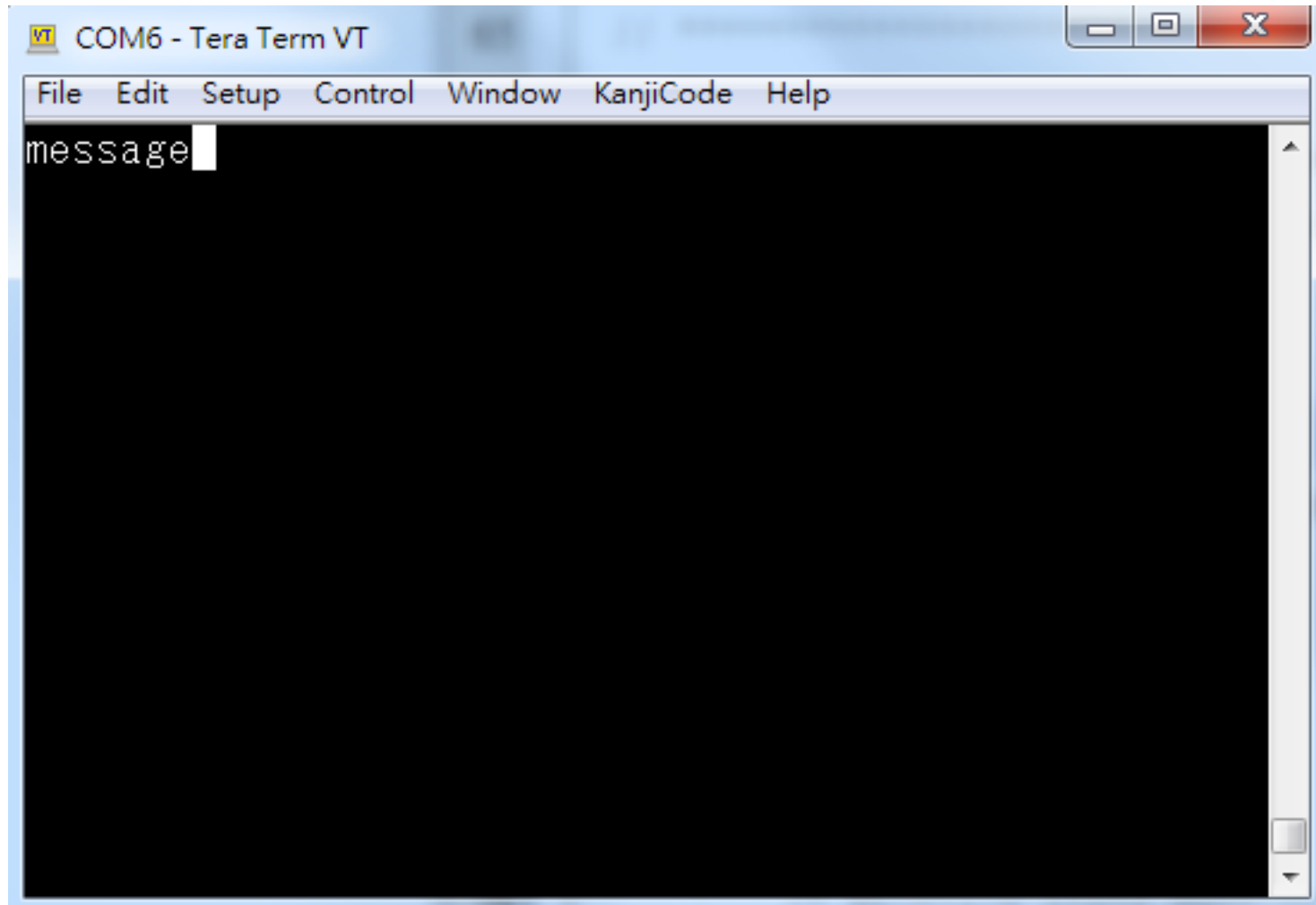
```
typedef enum
{
    /* Application's state machine's initial state. */
    APP_STATE_INIT=0,
    APP_STATE_SERVICE_TASKS,
    APP_STATE_IDLE,
    /* TODO: Define states used by the application state machine. */

} APP_STATES;
```

---

# Generate “Console system service”

## Step 8 : 編譯、執行





**MICROCHIP**

**Thank You**

**Any Questions?**