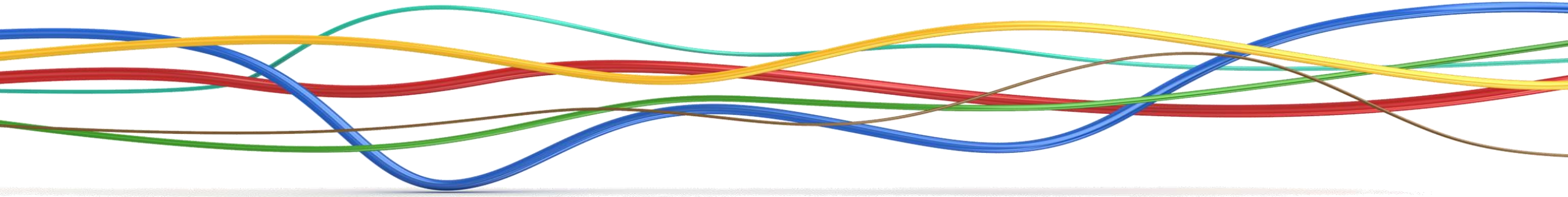
Digital Power Supply from Scratch in 60 Minutes



MICROCHIP

A Leading Provider of Smart, Connected and Secure Embedded Control Solutions

SMART | CONNECTED | SECURE

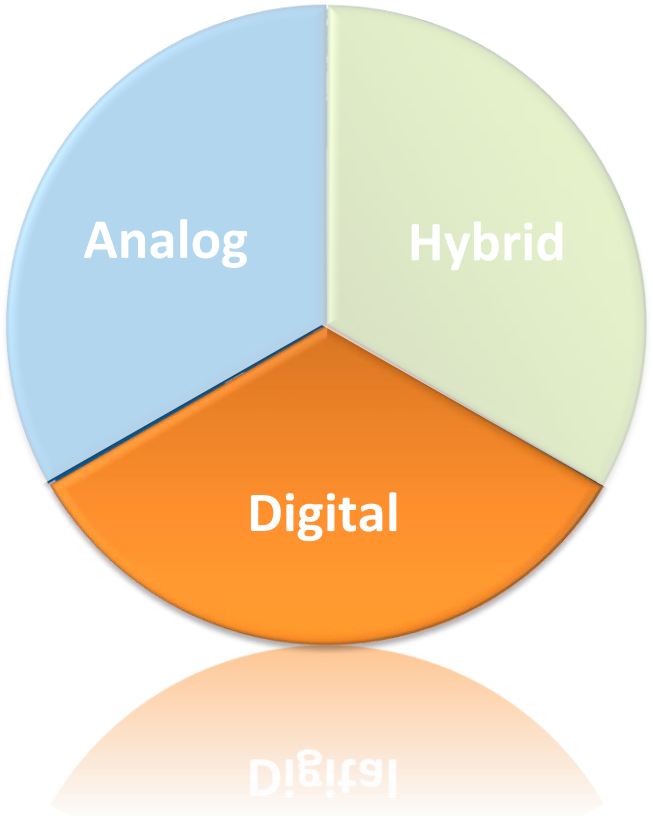
Power Conversion

Presented by Andy Reiter

15th November 2022

The World of Power Control

Standard Analog Controllers
Integrated solutions for dedicated topologies, power levels and applications



Programmable Digitally Enhanced Power Analog
Integrated solutions for various topologies, power levels and applications

Fully Digitally Controlled Systems
High-end control solutions for various topologies, power levels and applications providing highest flexibility

MICROCHIP

Agenda

Digital Power Supply from Scratch in 60 Minutes*

*: most probably not true ☺

- **Overview Digital Power Starter Kit 3 (DPSK3)**
- **Requirements and Scope**
- **Hands-On**
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- **Q&A**

Agenda

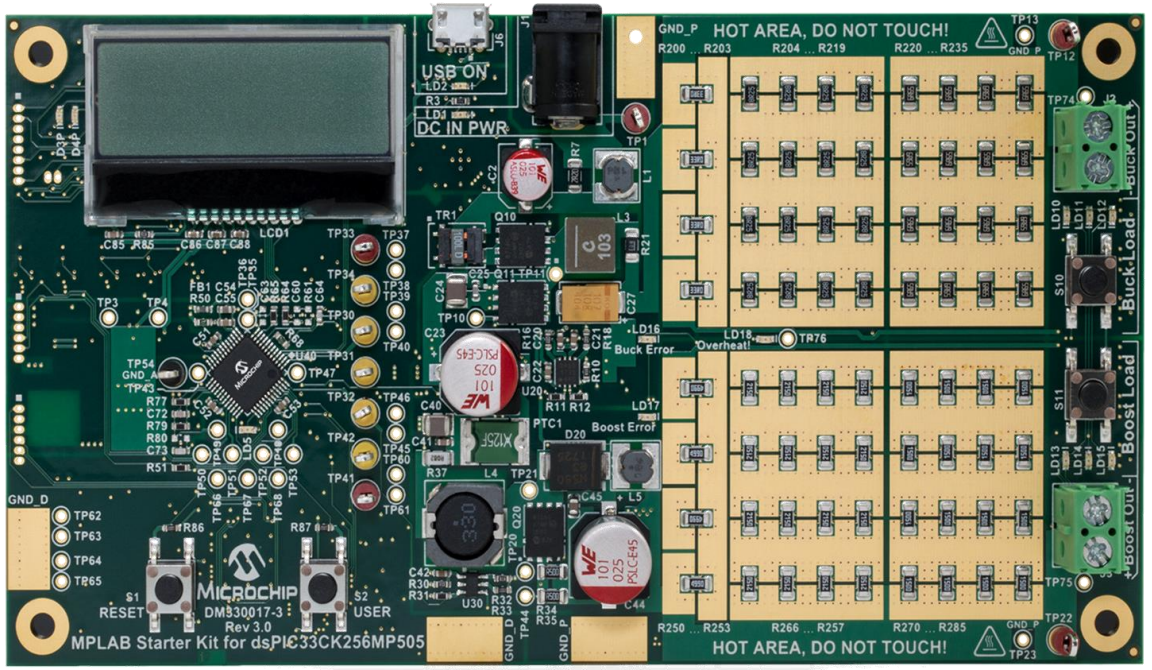
Digital Power Supply from Scratch in 60 Minutes*

*: no, honestly - it's a lie ☺

- **Overview Digital Power Starter Kit 3 (DPSK3)**
- **Requirements and Scope**
- **Hands-On**
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- **Q&A**

dsPIC33C Digital Power Starter Kit (DPSK3)

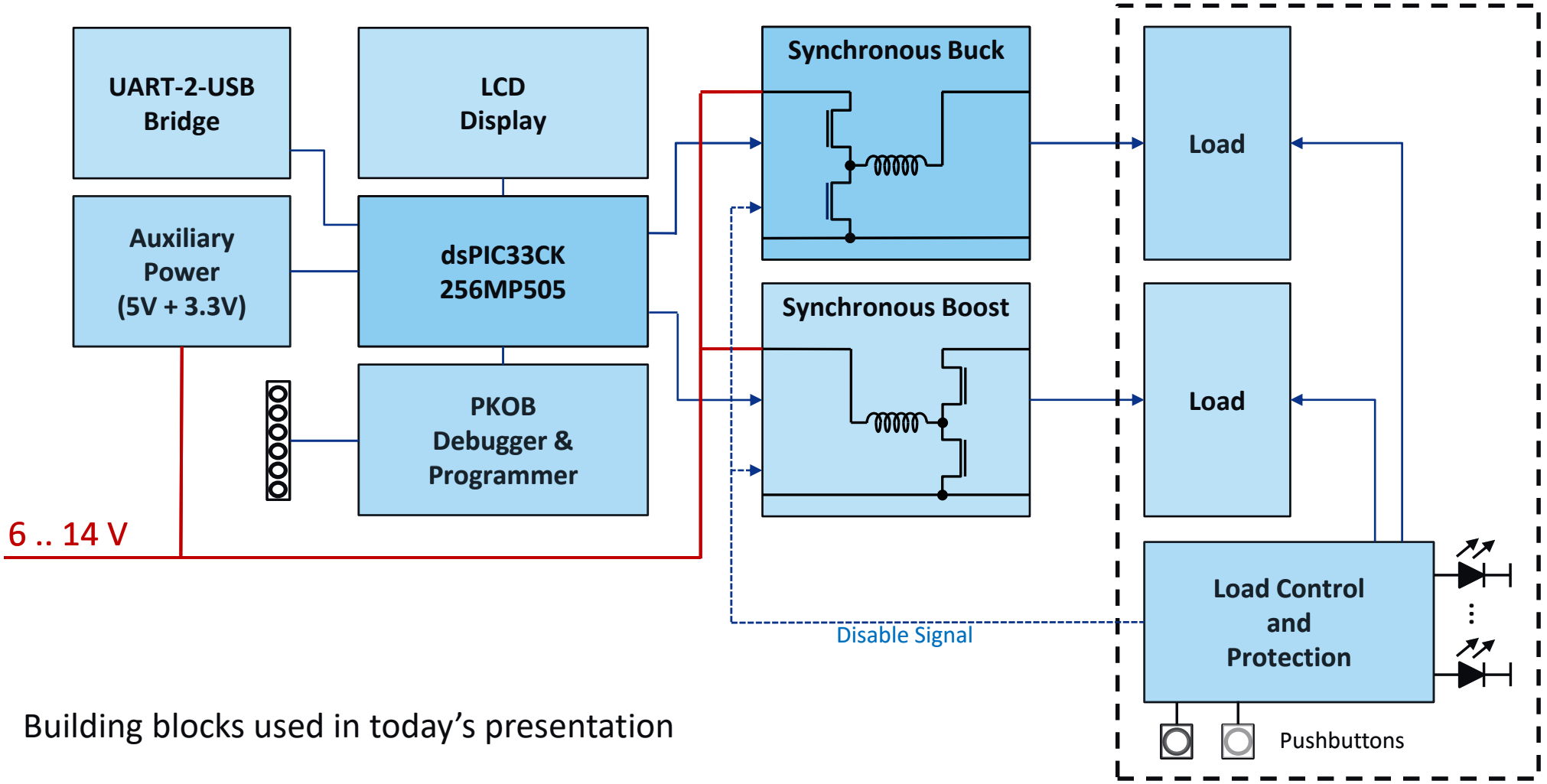
- **On-board dsPIC33CK256MP505 DSC**
- **PIC24F Auxiliary Microcontroller managing loads and protection circuit auto recovery**
- **Two Independent DC/DC Converter Topologies:**
  - Synchronous Buck Converter
  - Asynchronous Boost Converter
- **Independent resistive loads**
  - Four selectable Constant Load Levels
  - Three Selectable Step Load Levels
- **Protection circuitry**
  - Over Current Protection (OCP)
  - Over Voltage Protection (OVP)
  - Over Temperature Protection (OTP)
- **Development Features**
  - PKOB4 On-Board Programmer/Debugger
  - LC Display User Interface
  - USB/UART Bridge (Standard VCP)



Available Now:

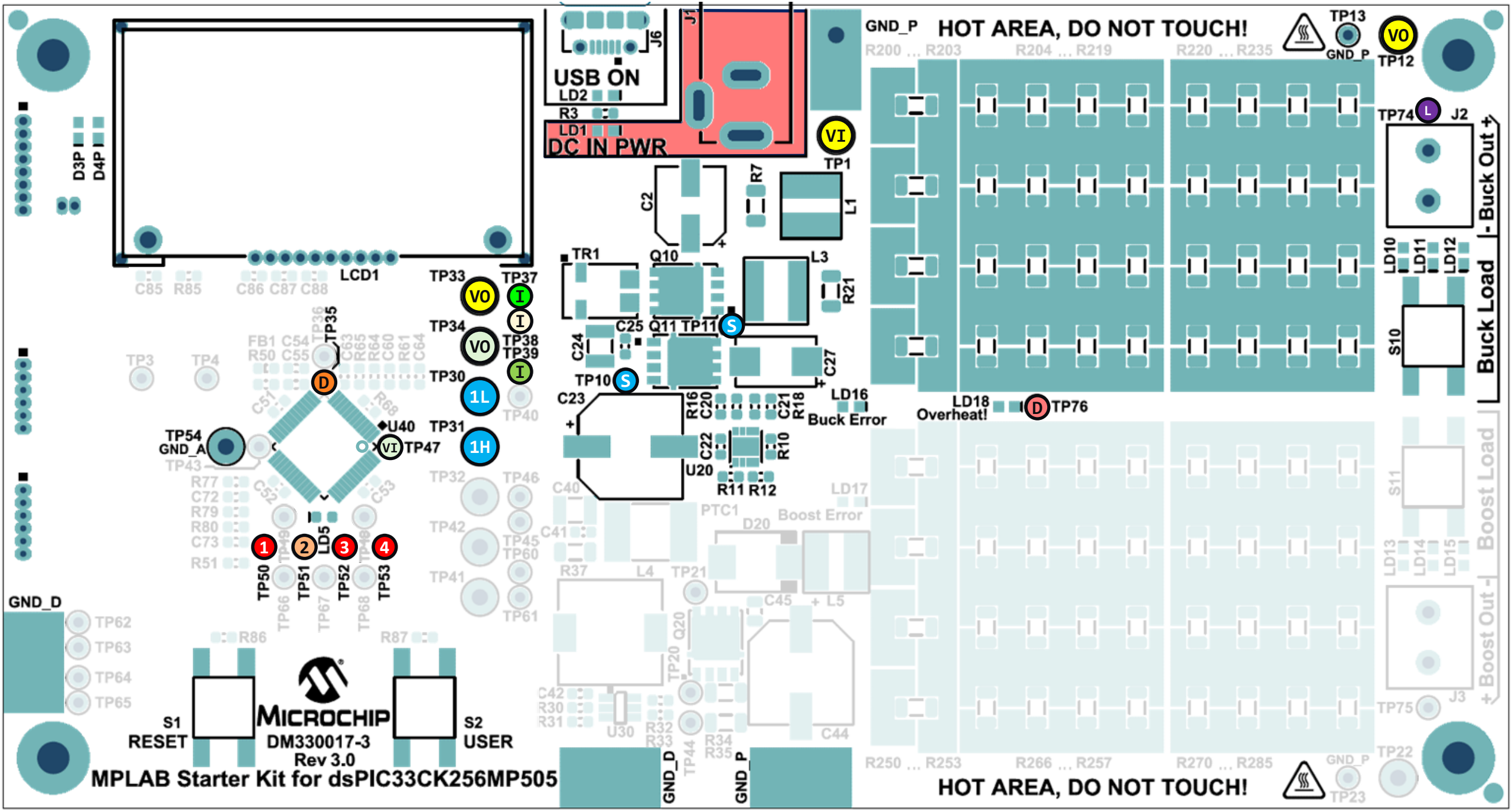
Part-No: DM330017-3

dsPIC33C Digital Power Starter Kit 3 (DPSK3)



Building blocks used in today's presentation

MICROCHIP


HOT AREA, DO NOT TOUCH!
USB ON
DC IN PWR
LCD1
Buck Load
- Buck Out +
Boost Load
+ Boost Out -
Buck Error
Boost Error
Overheat!
GND_P
GND_D
GND_A
S1
RESET
S2
USER
MICROCHIP
DM330017-3
Rev 3.0
MPLAB Starter Kit for dsPIC33CK256MP505

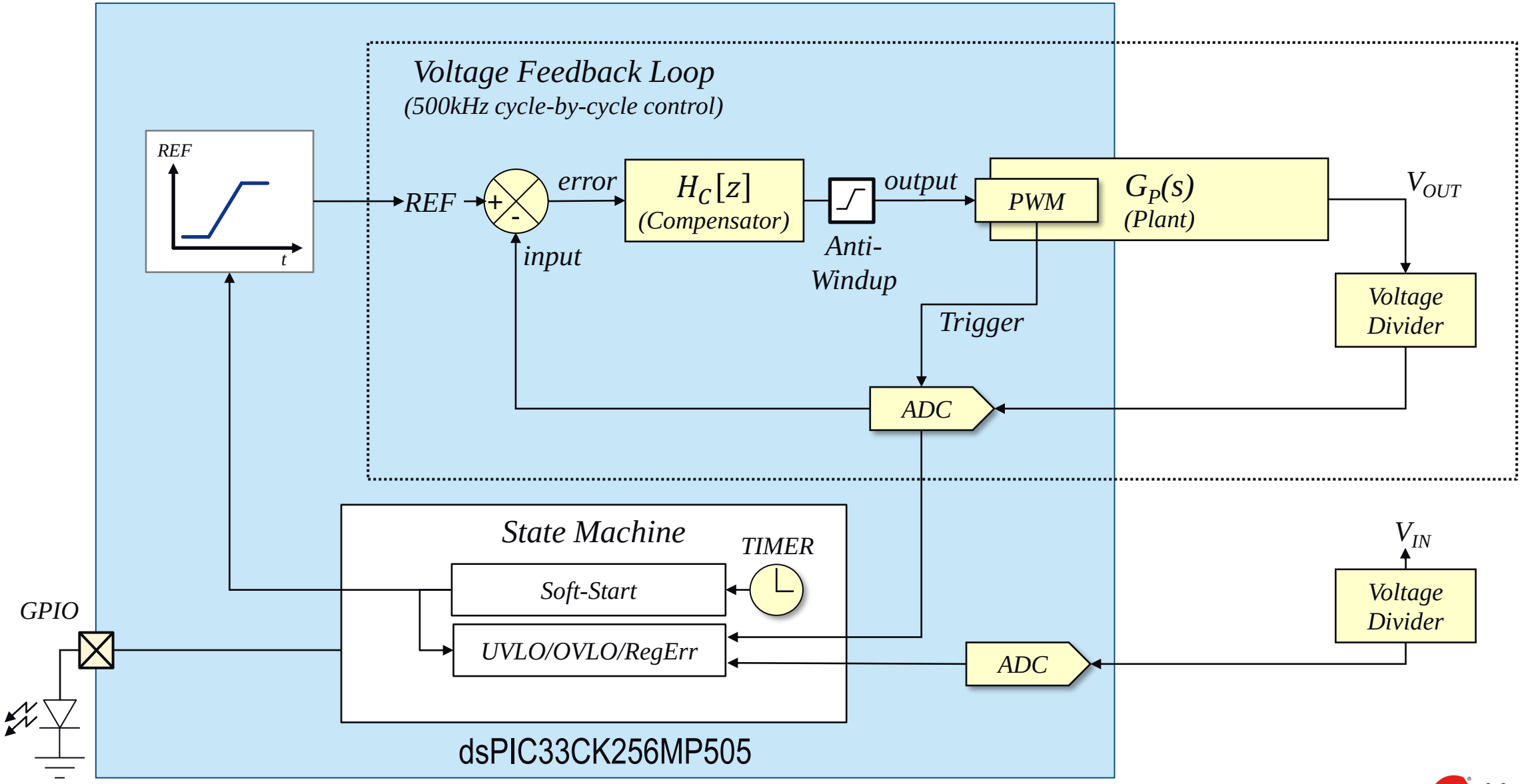
Agenda

Digital Power Supply from Scratch in 60 Minutes*

*: who had the idea to put this in the title?

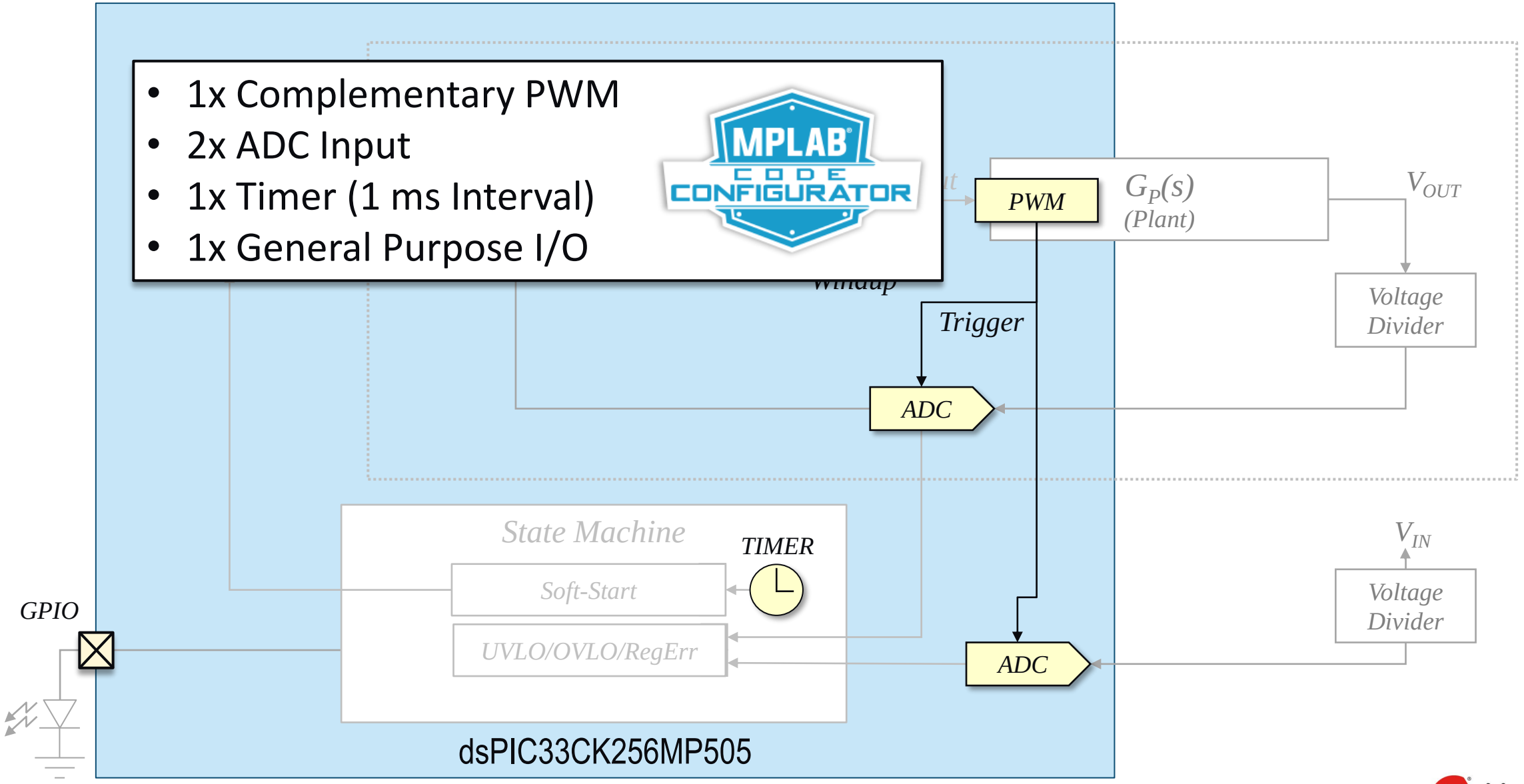
- Overview Digital Power Starter Kit 3 (DPSK3)
- **Requirements and Scope**
- Hands-On
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- Q&A

Requirements



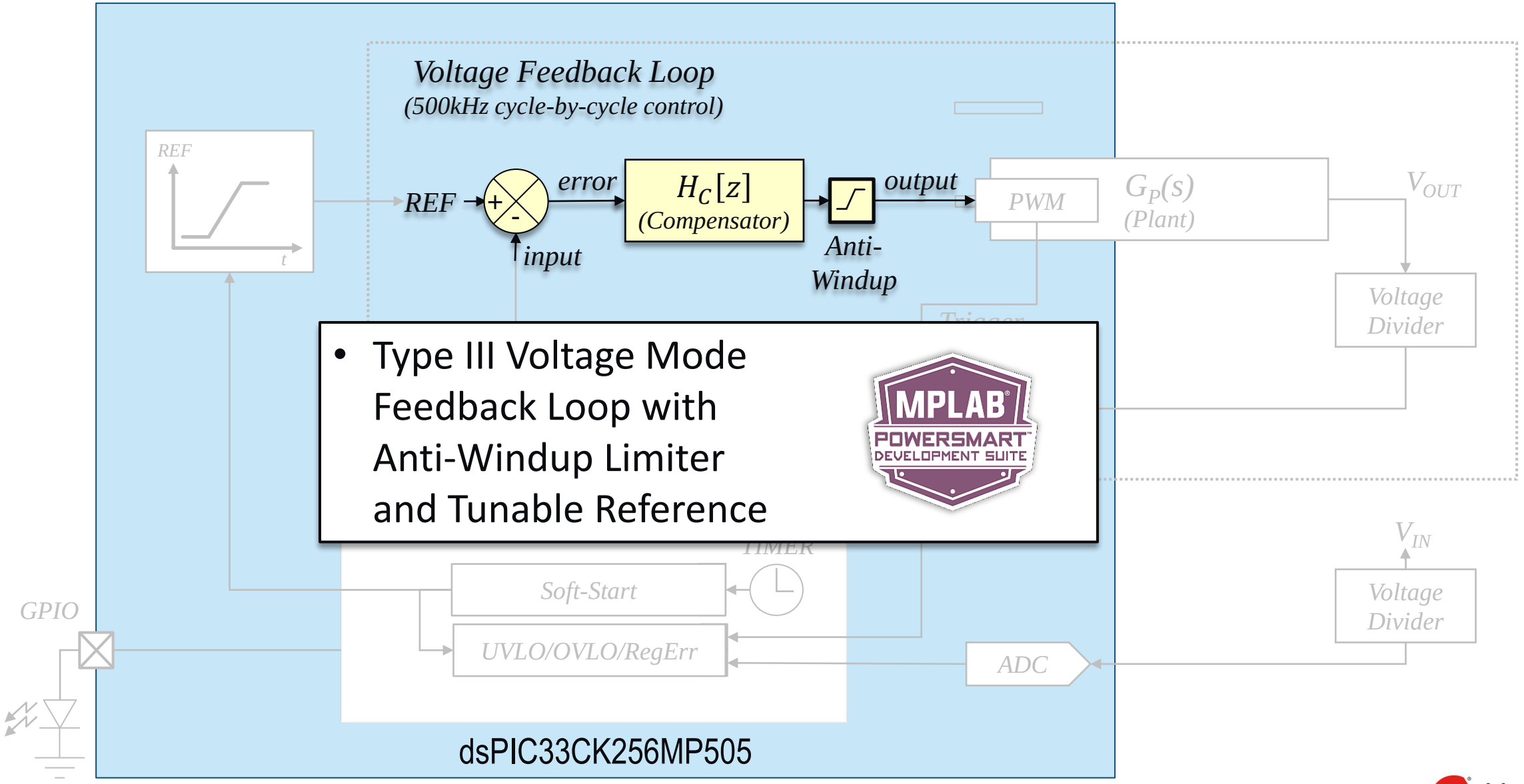
MICROCHIP

Requirements



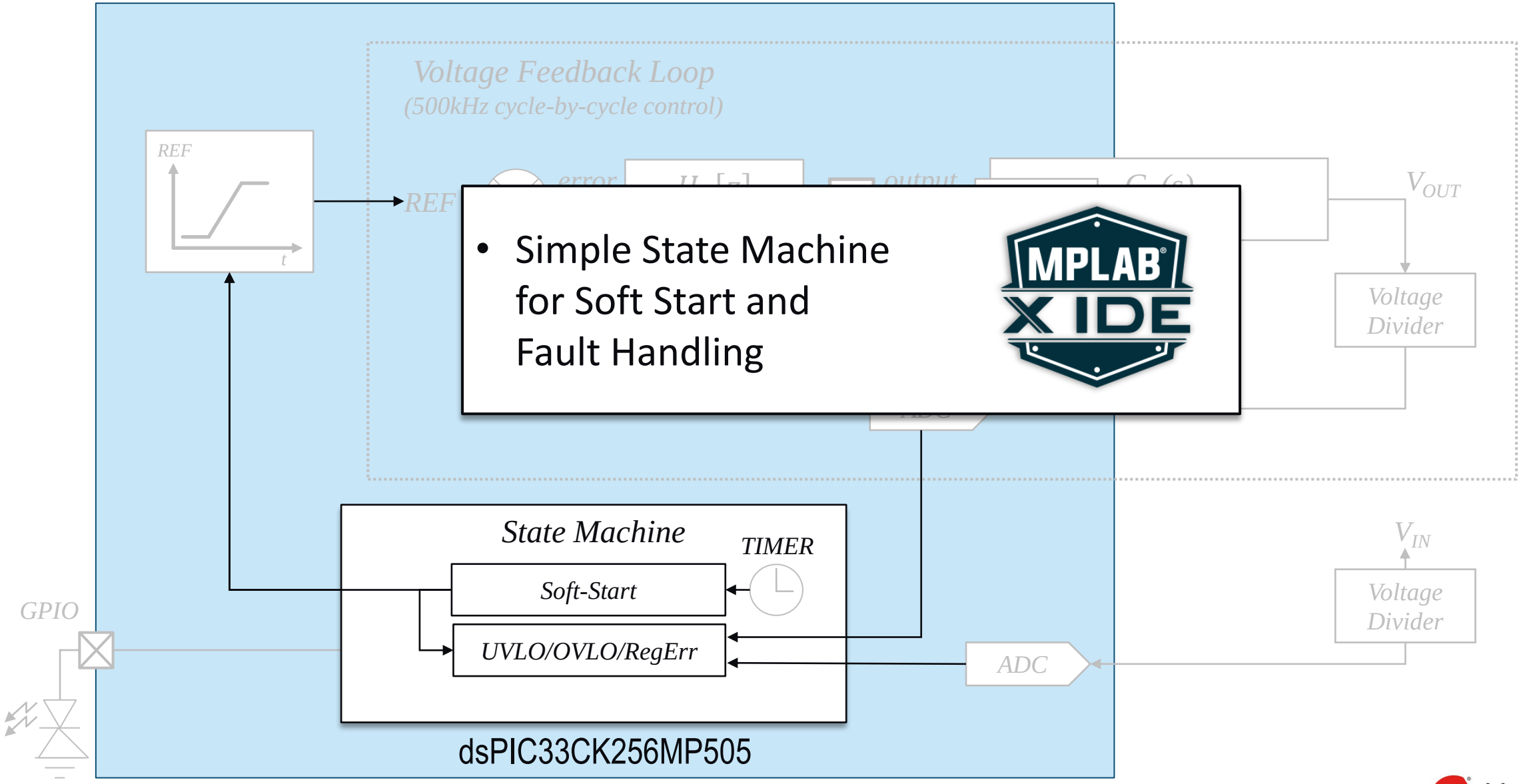
MICROCHIP

Requirements



MICROCHIP

Requirements

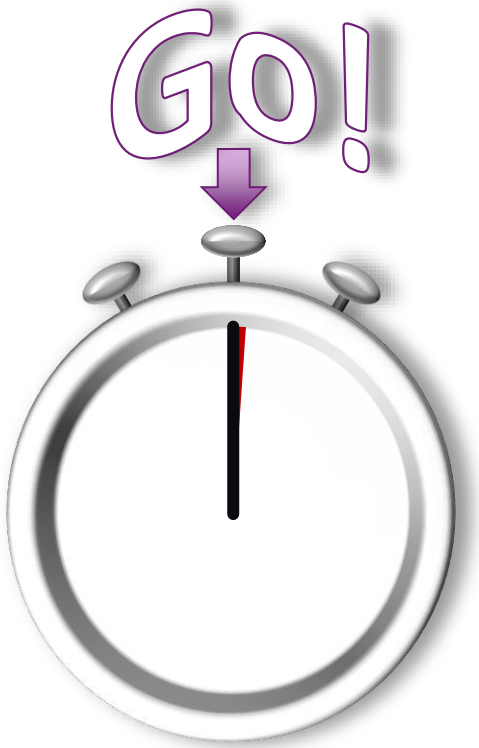


MICROCHIP

This is from where we start…

Estimated Time to Output Voltage (ETOV): 1:00:00 *

*: Ah! Now you are starting to cheat!



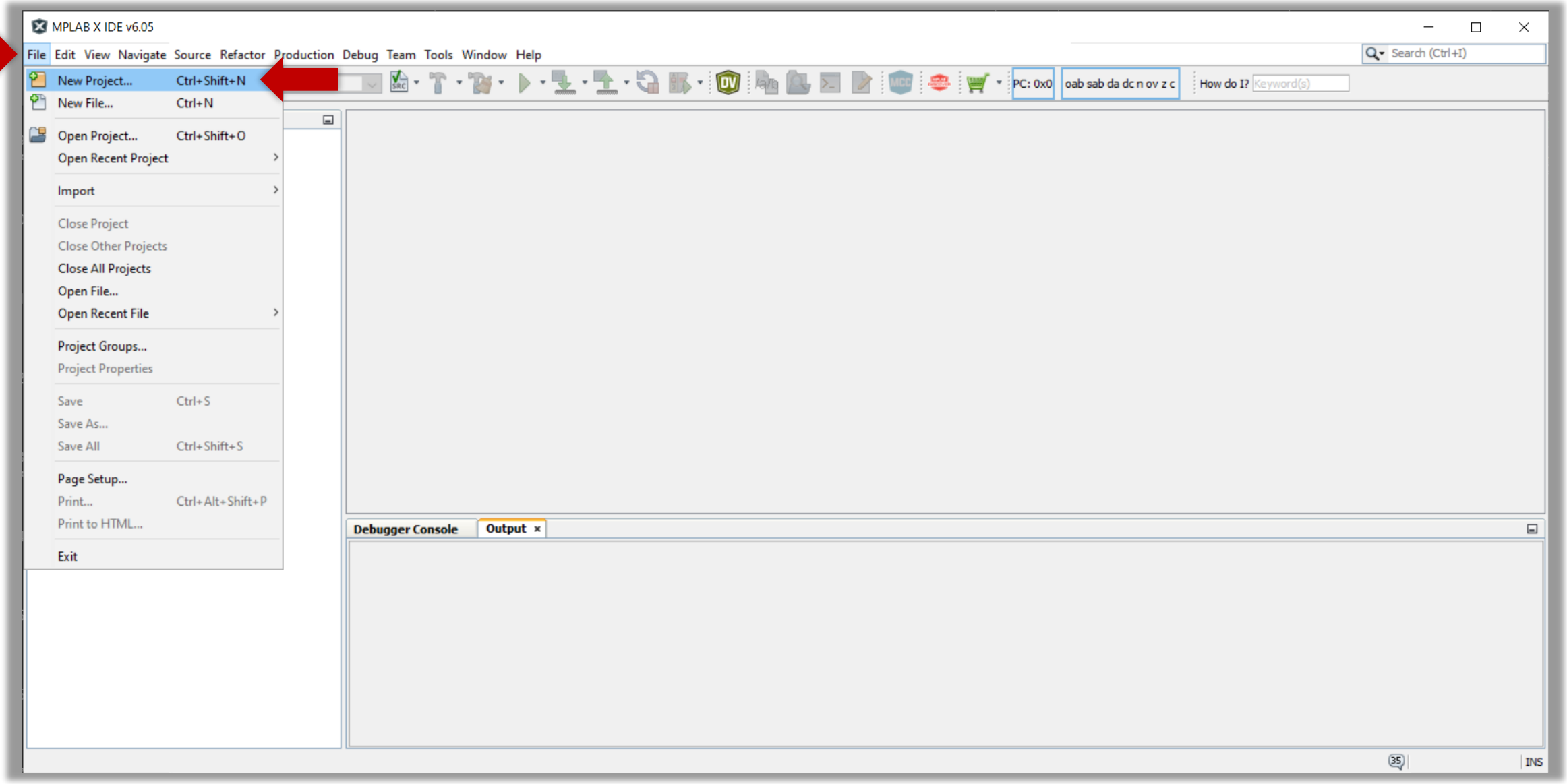
Agenda

Digital Power Supply from Scratch in 60 Minutes*

*: ok, now it's getting real….

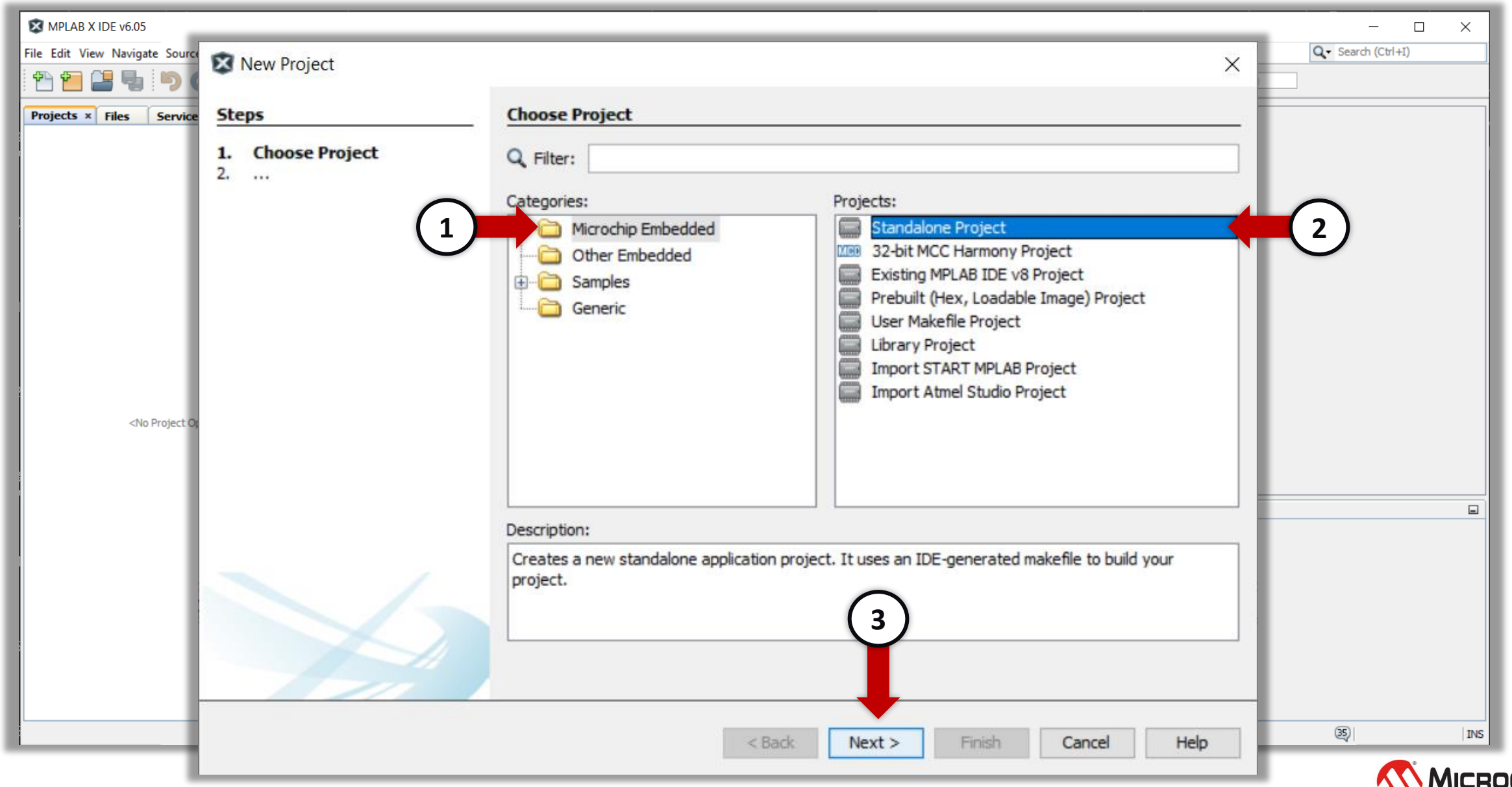
- Overview Digital Power Starter Kit 3 (DPSK3)
- Requirements and Scope
- **Hands-On**
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- Q&A

Creating a New Project



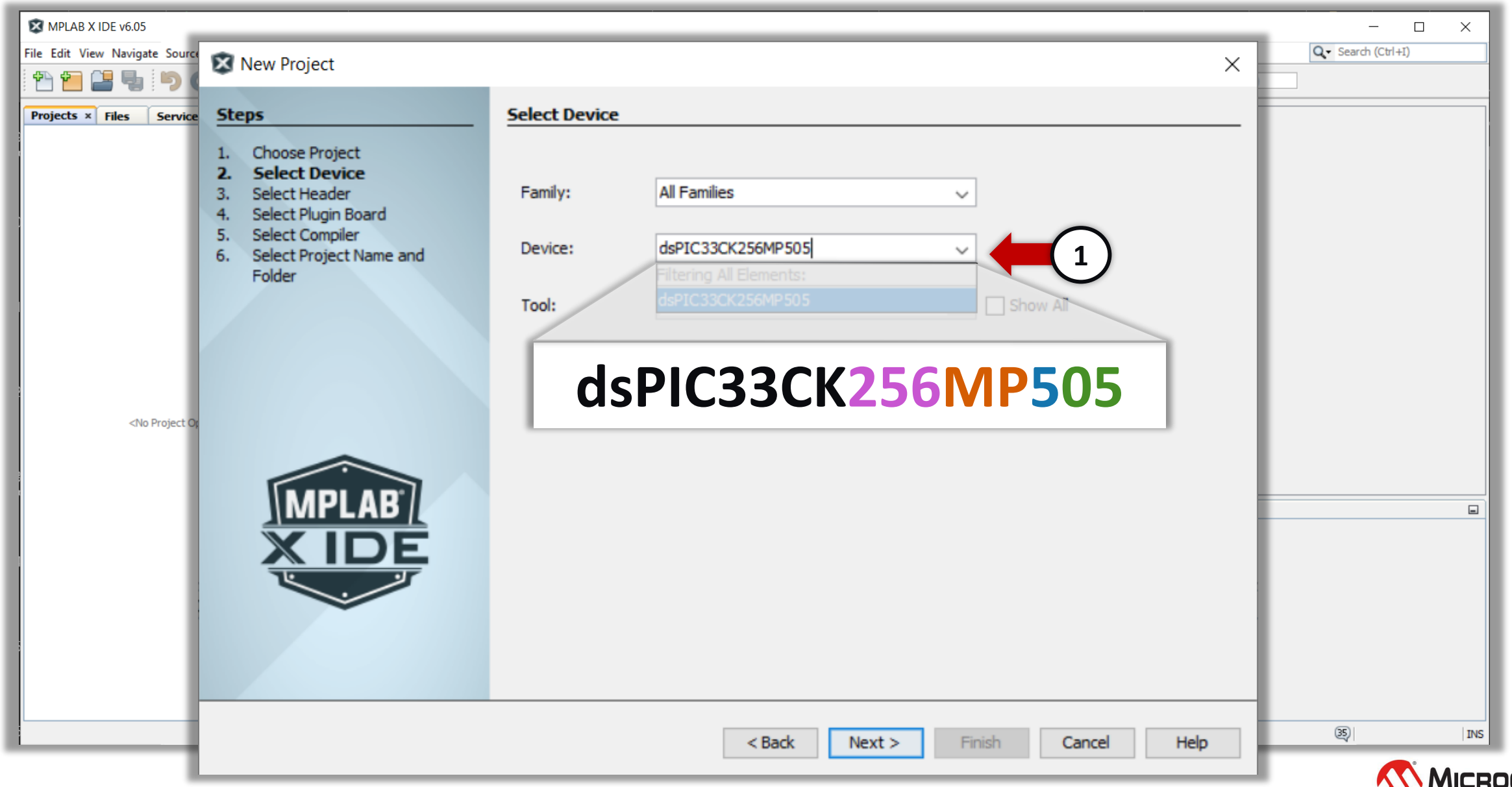
MICROCHIP

Create a Standalone Project



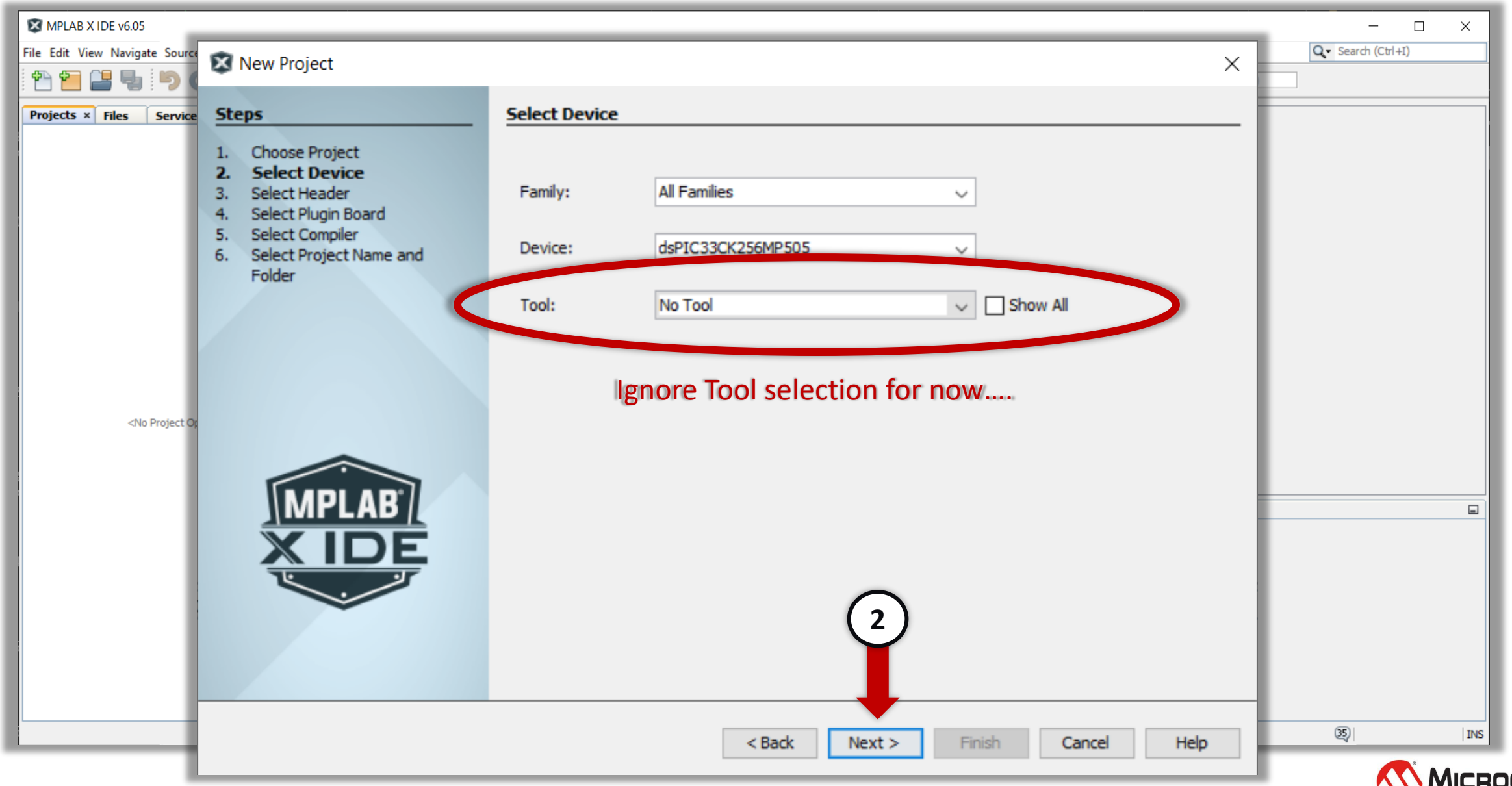
MICROCHIP

Select Target Device & Programming Tool



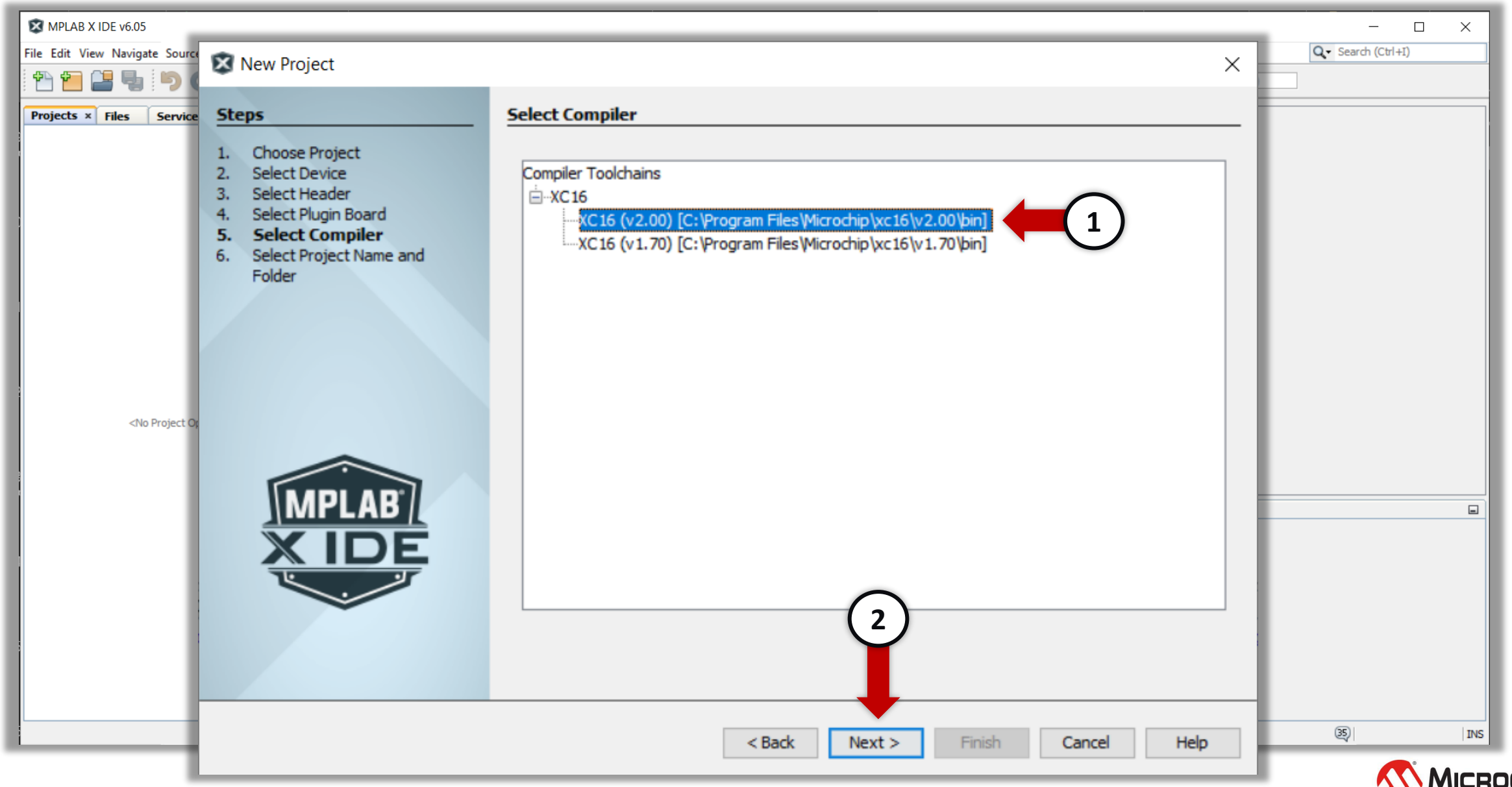
MICROCHIP

Select Target Device & Programming Tool



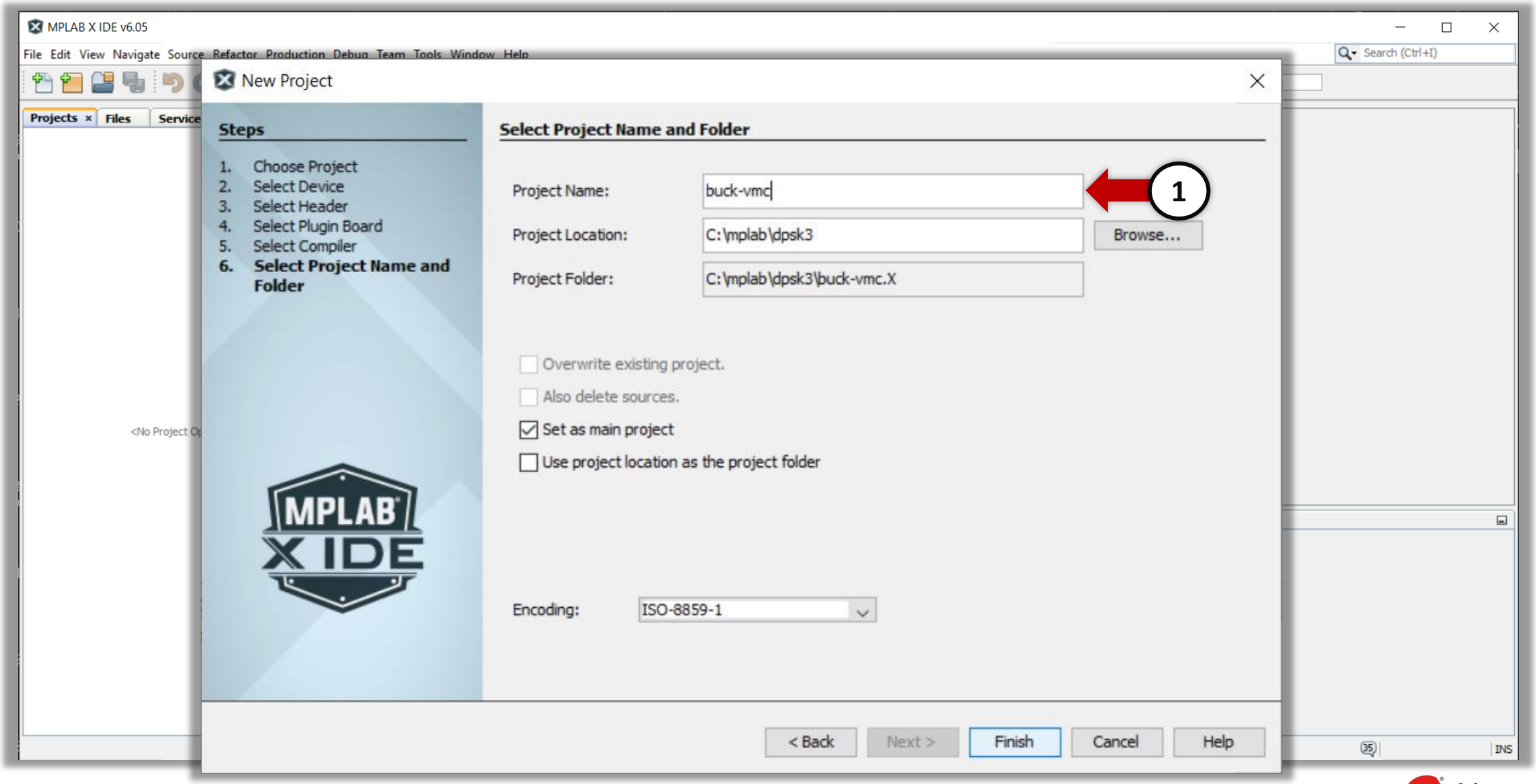
MICROCHIP

Select C Compiler



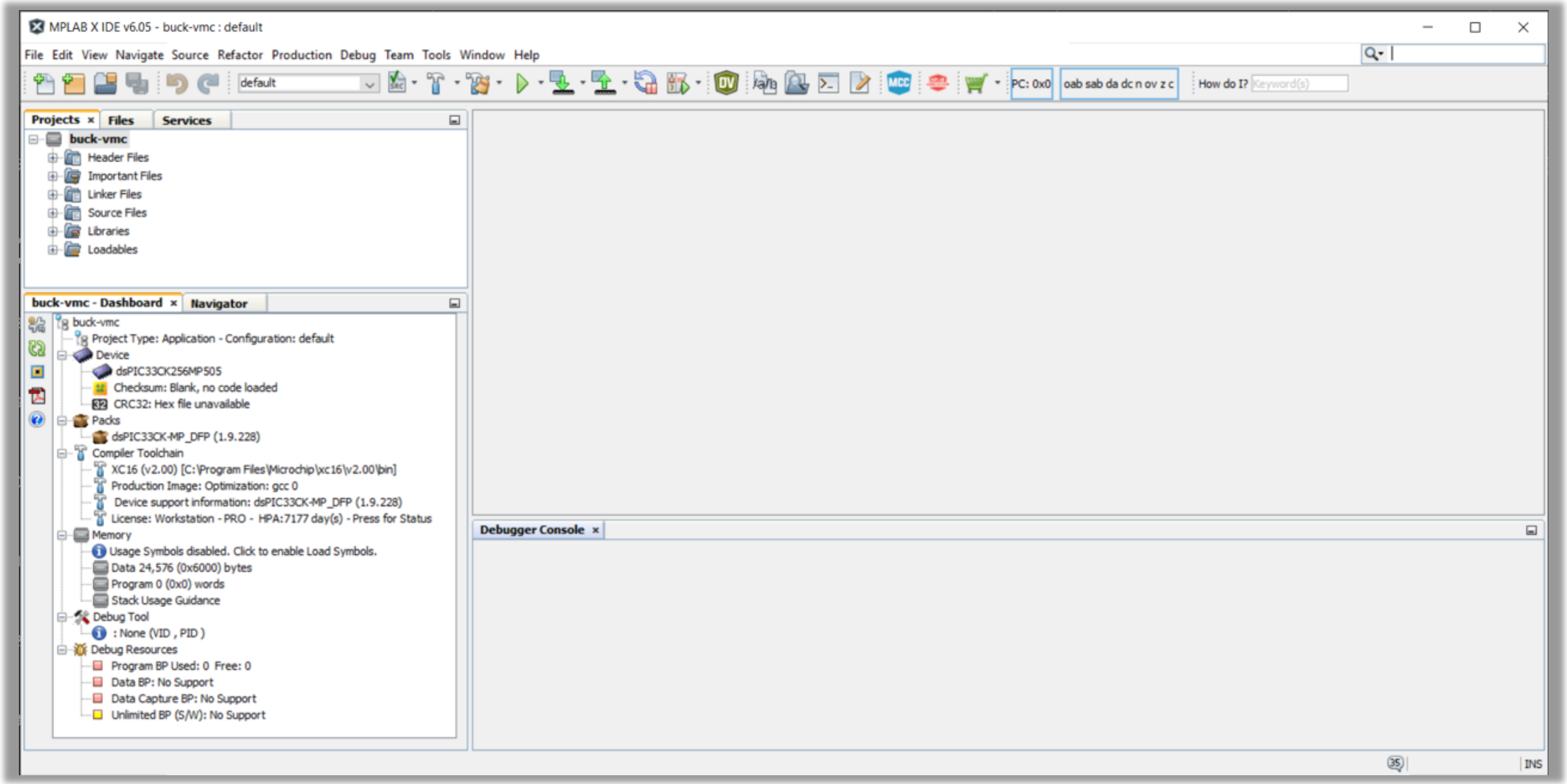
MICROCHIP

Select Project Name & Location



MICROCHIP

Empty Project Has Been Created!



MICROCHIP

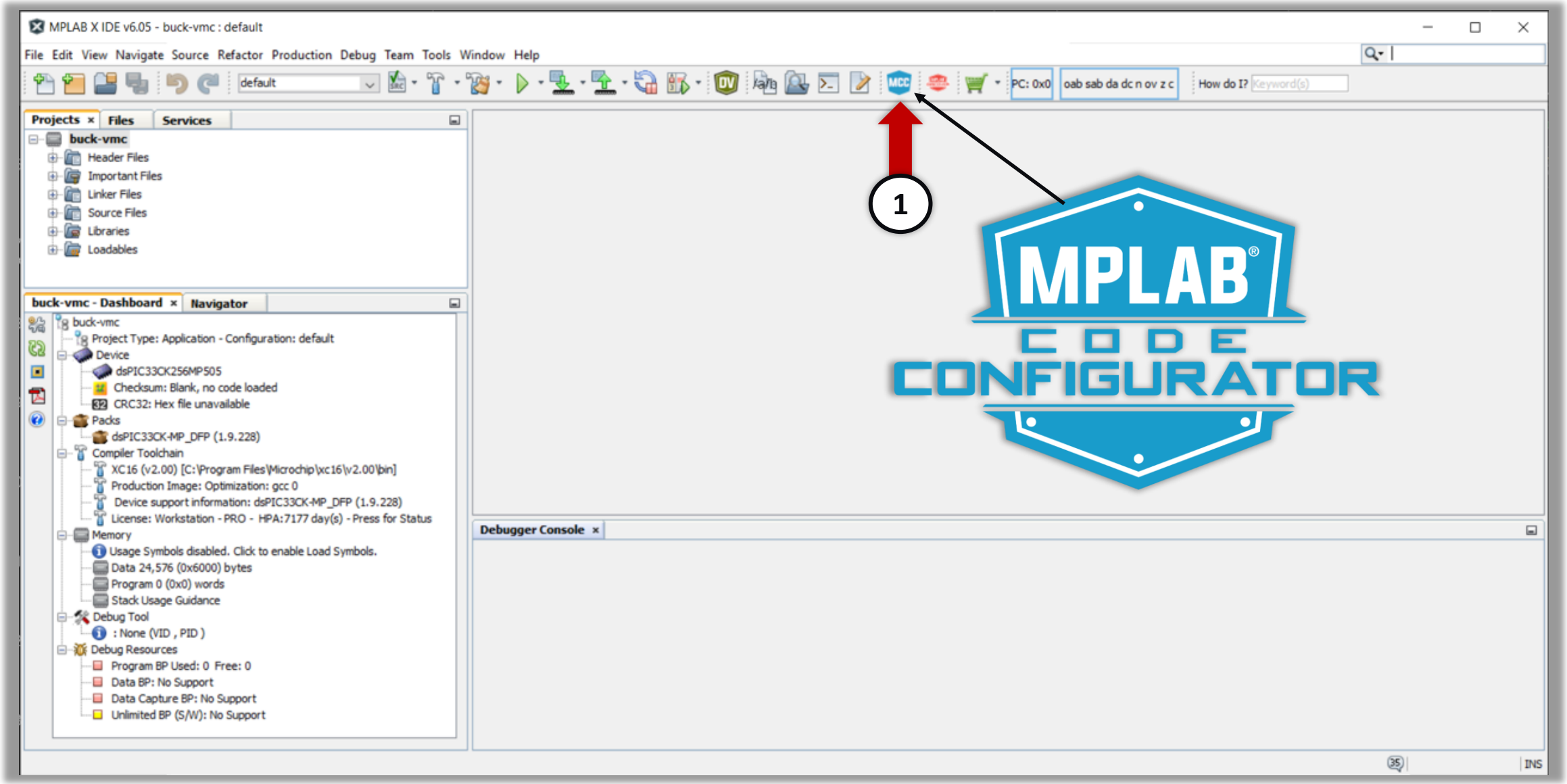
Agenda

Digital Power Supply from Scratch in 60 Minutes*

*: Curious to see how this is playing out ….

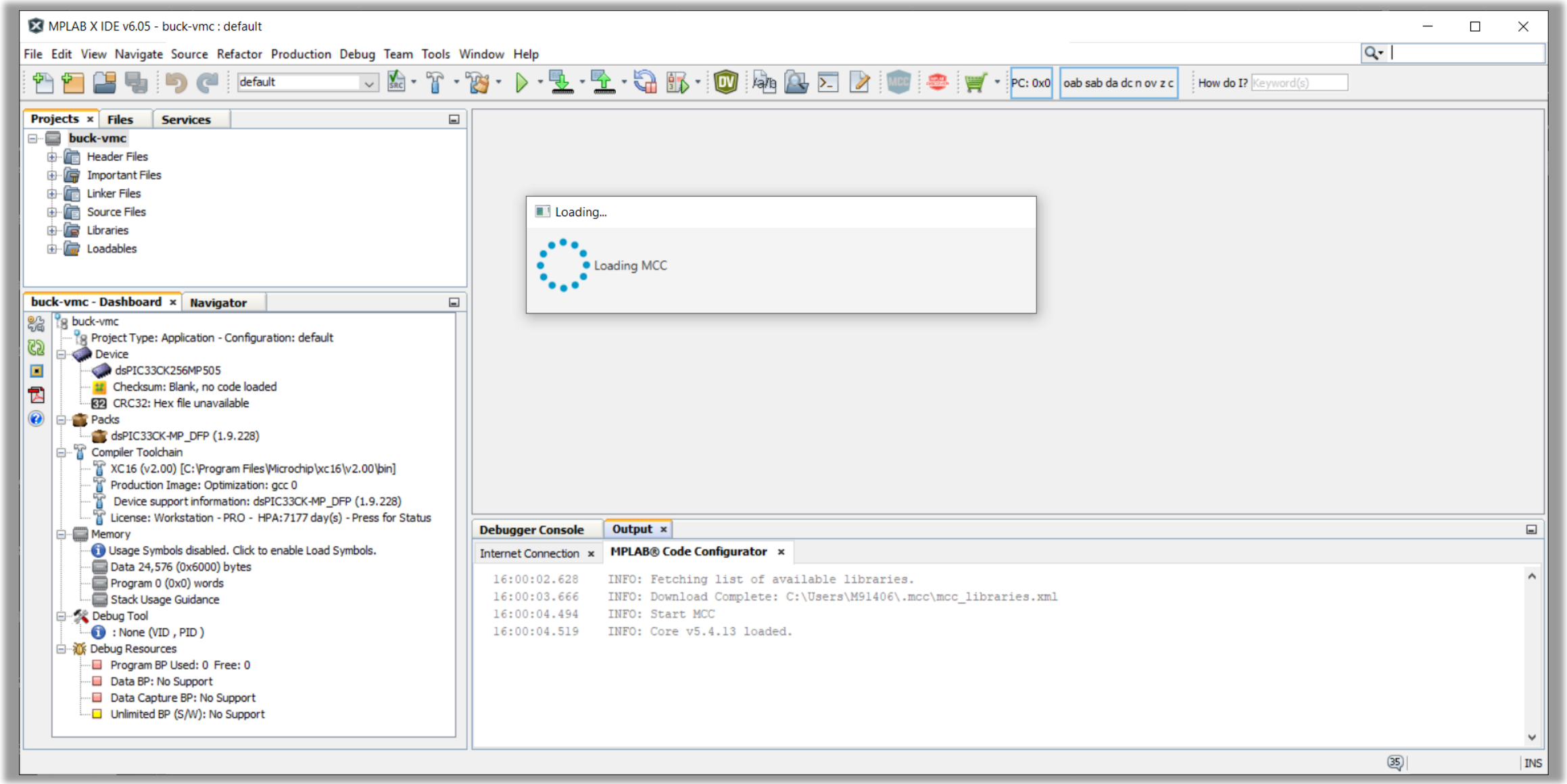
- Overview Digital Power Starter Kit 3 (DPSK3)
- Requirements and Scope
- **Hands-On**
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- Q&A

Open MPLAB® Code Configurator



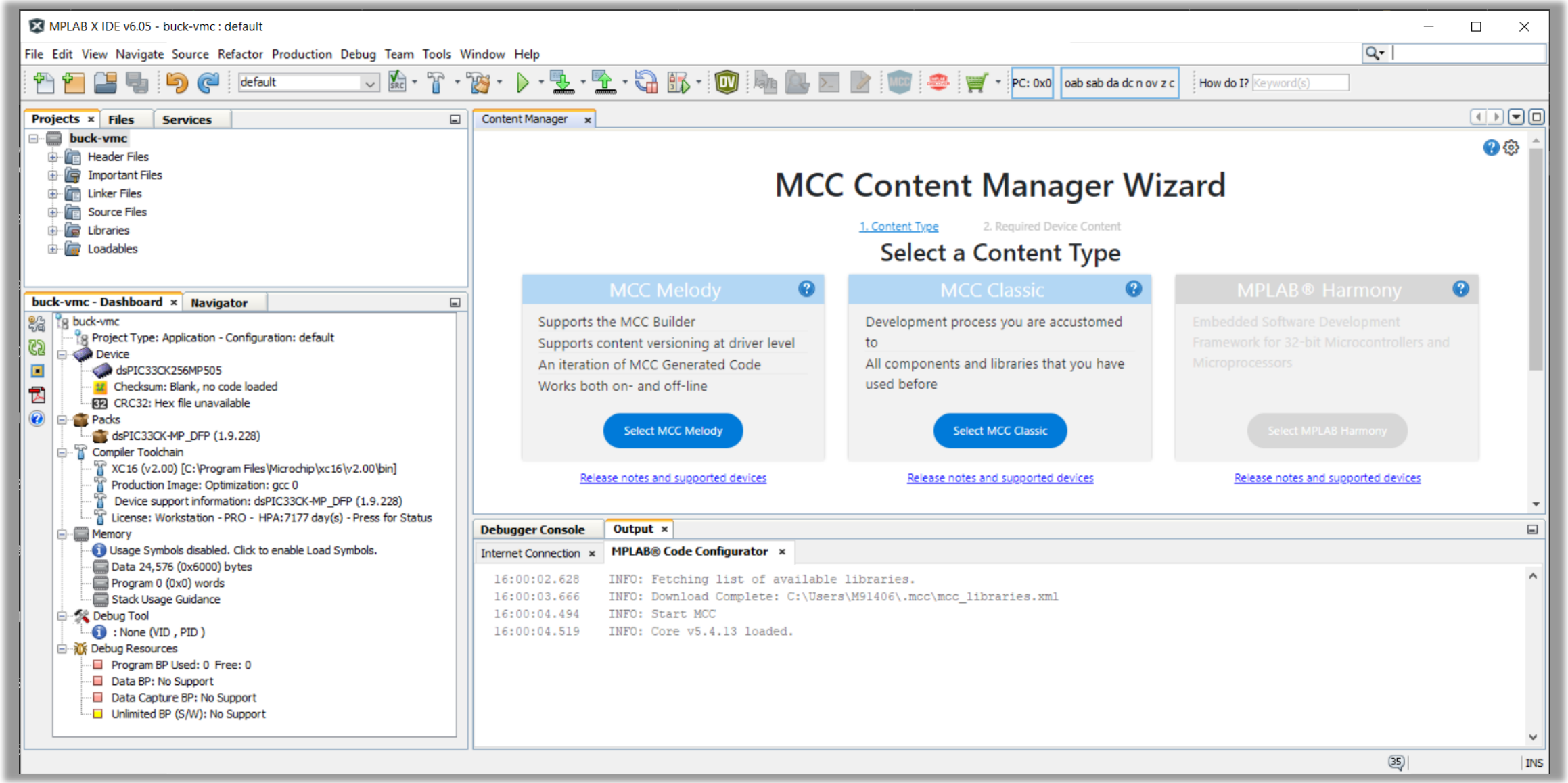
MICROCHIP

Open MPLAB® Code Configurator



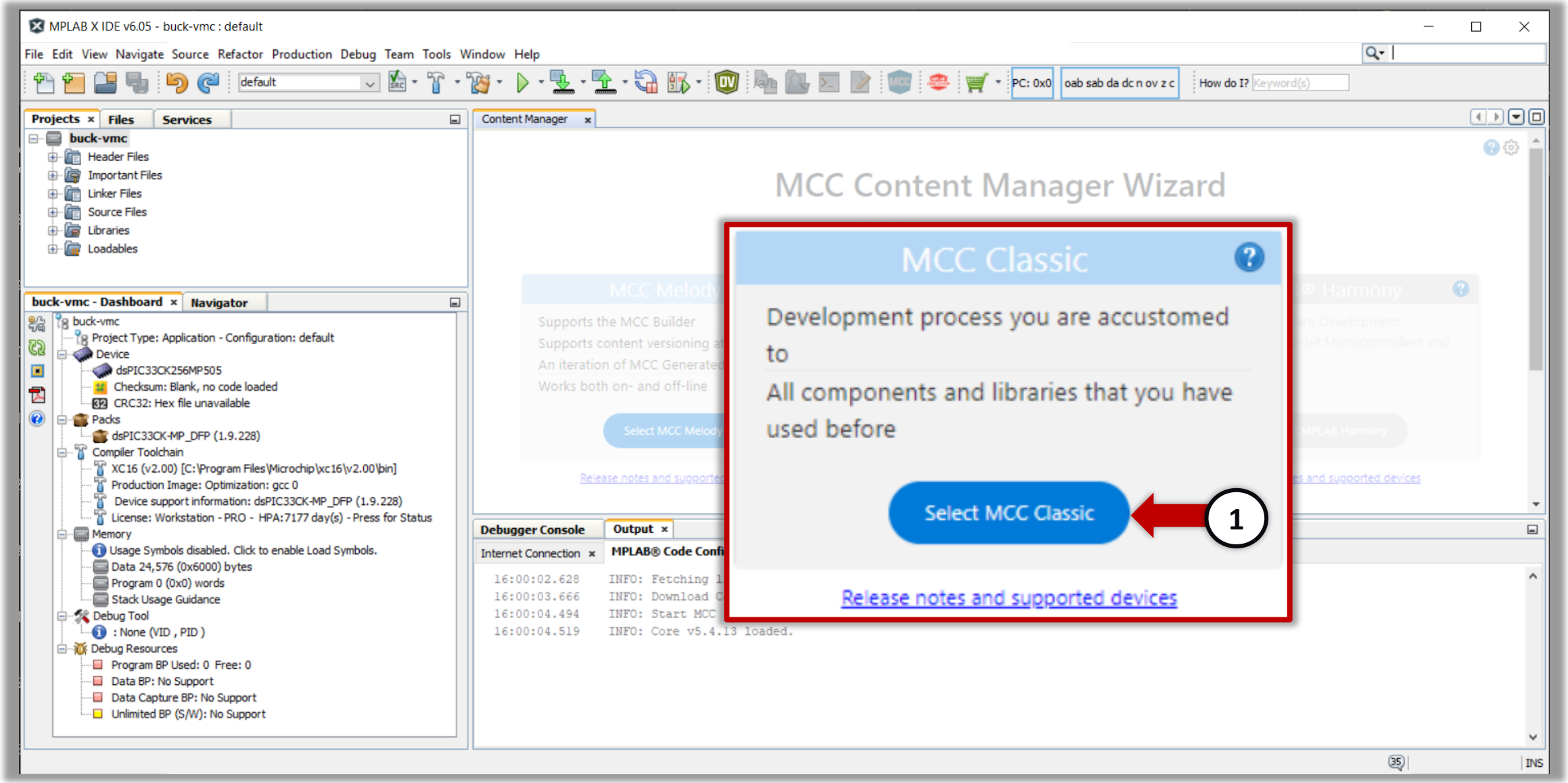
MICROCHIP

Open MPLAB® Code Configurator

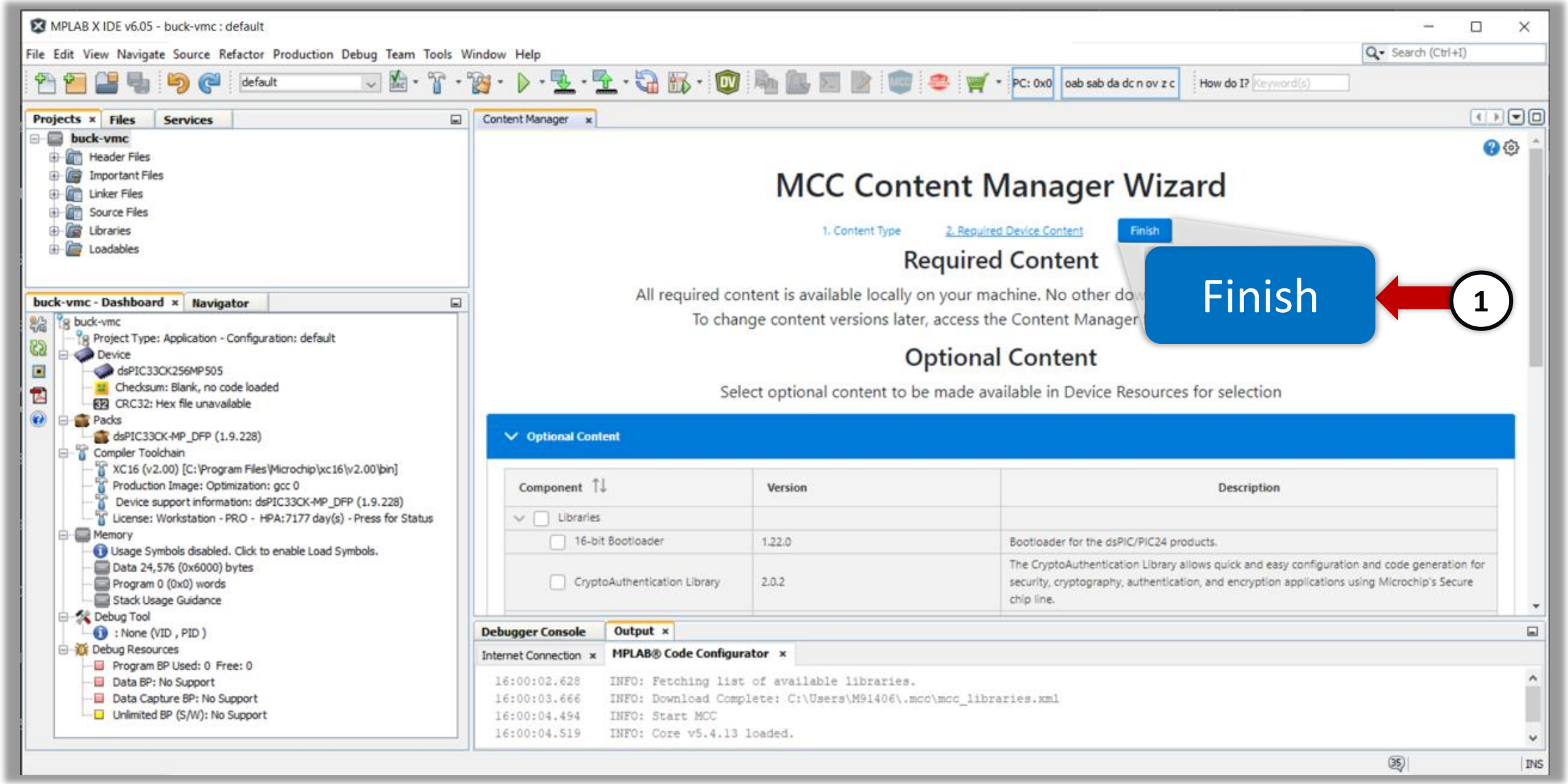


MICROCHIP

Open MPLAB® Code Configurator

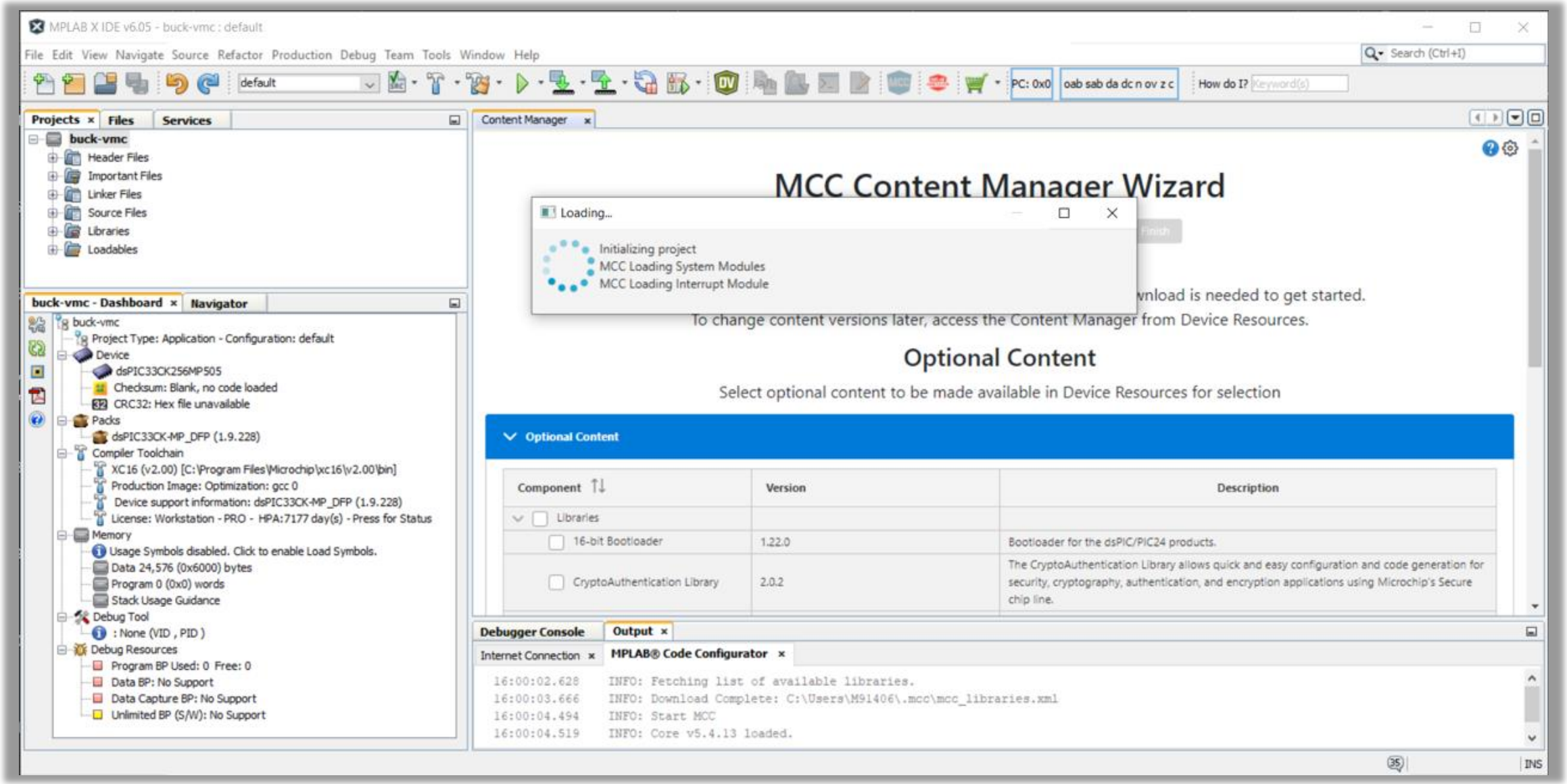


Open MPLAB® Code Configurator



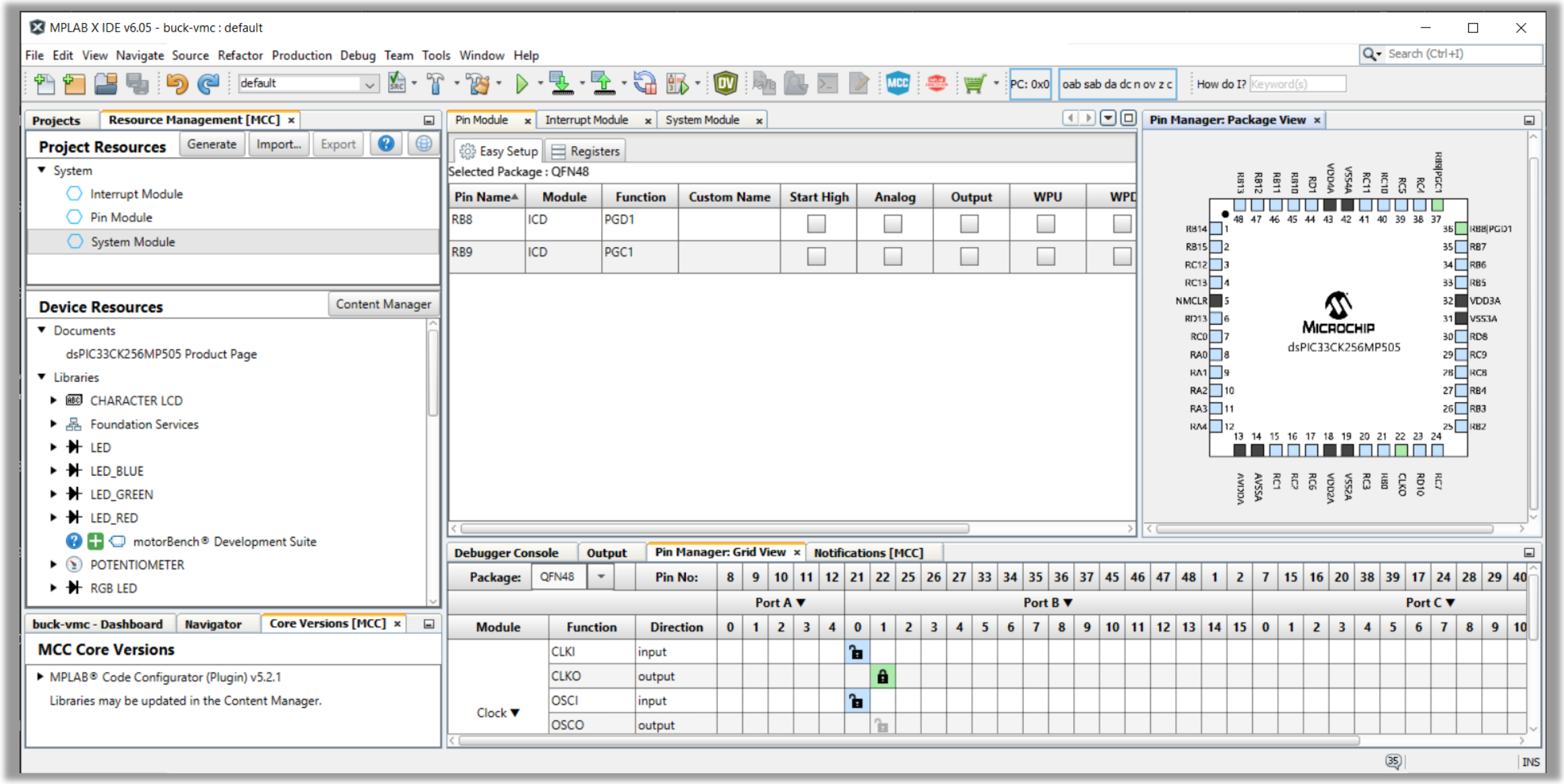
MICROCHIP

Open MPLAB® Code Configurator

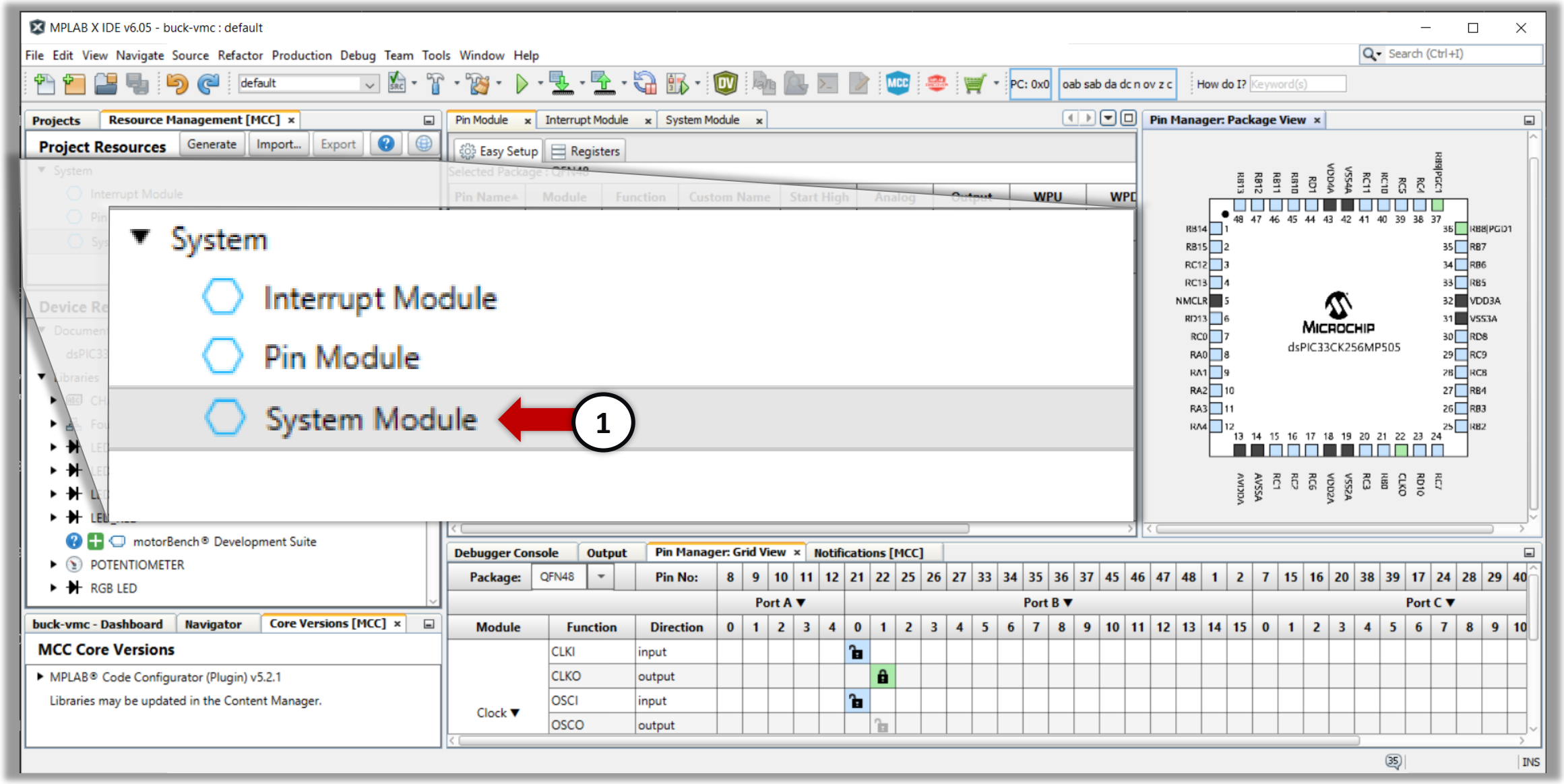


MICROCHIP

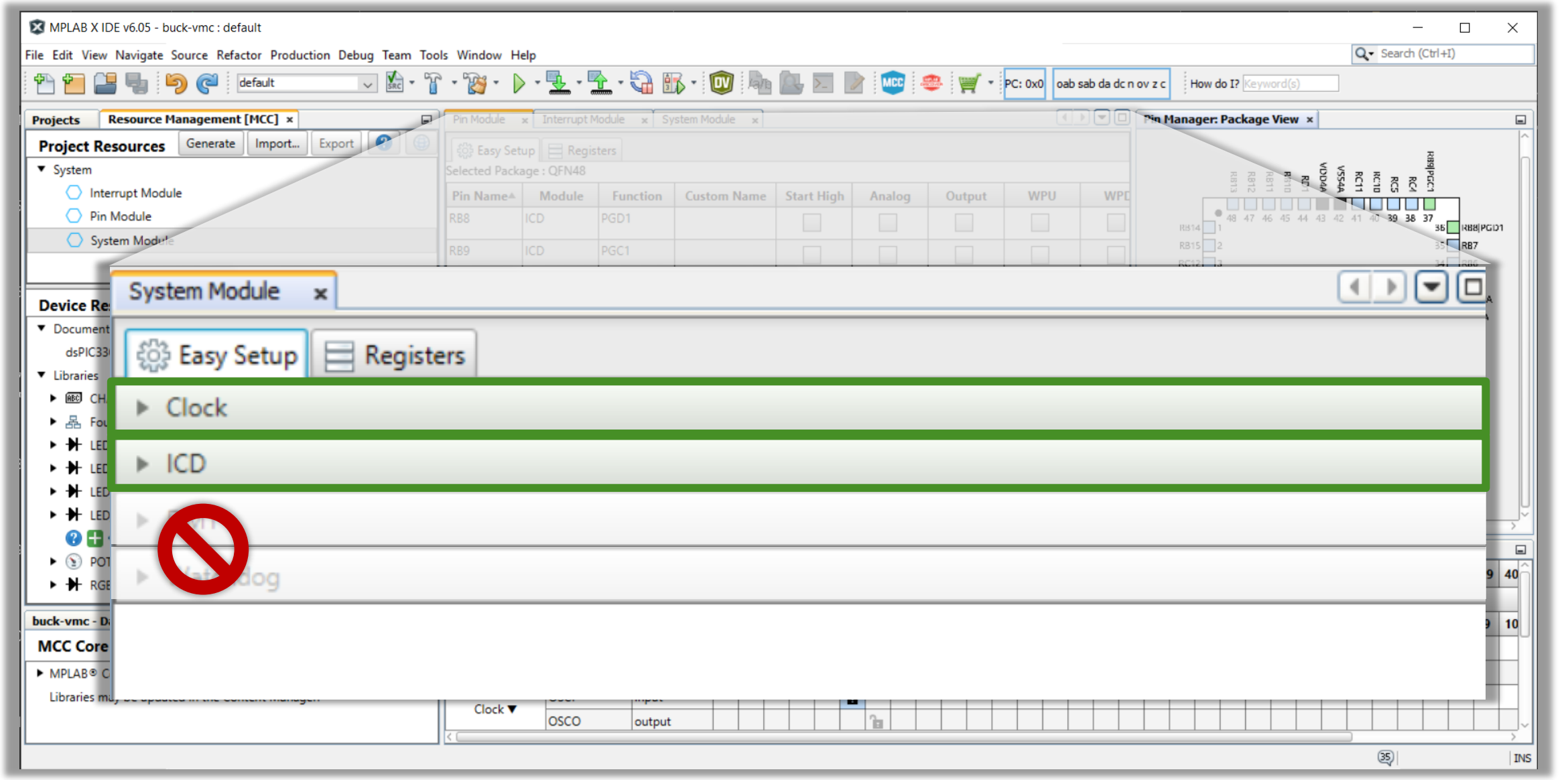
Open MPLAB® Code Configurator



Configuring the System Oscillator



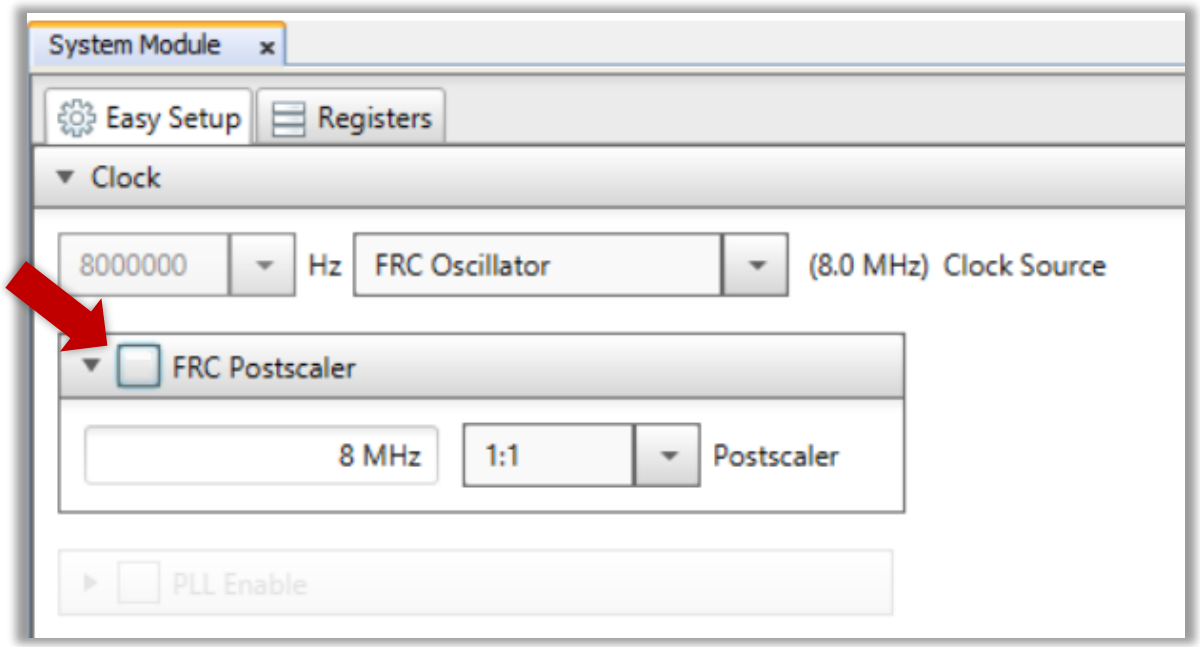
Configuring the System Oscillator



MICROCHIP

Configuring the System Oscillator

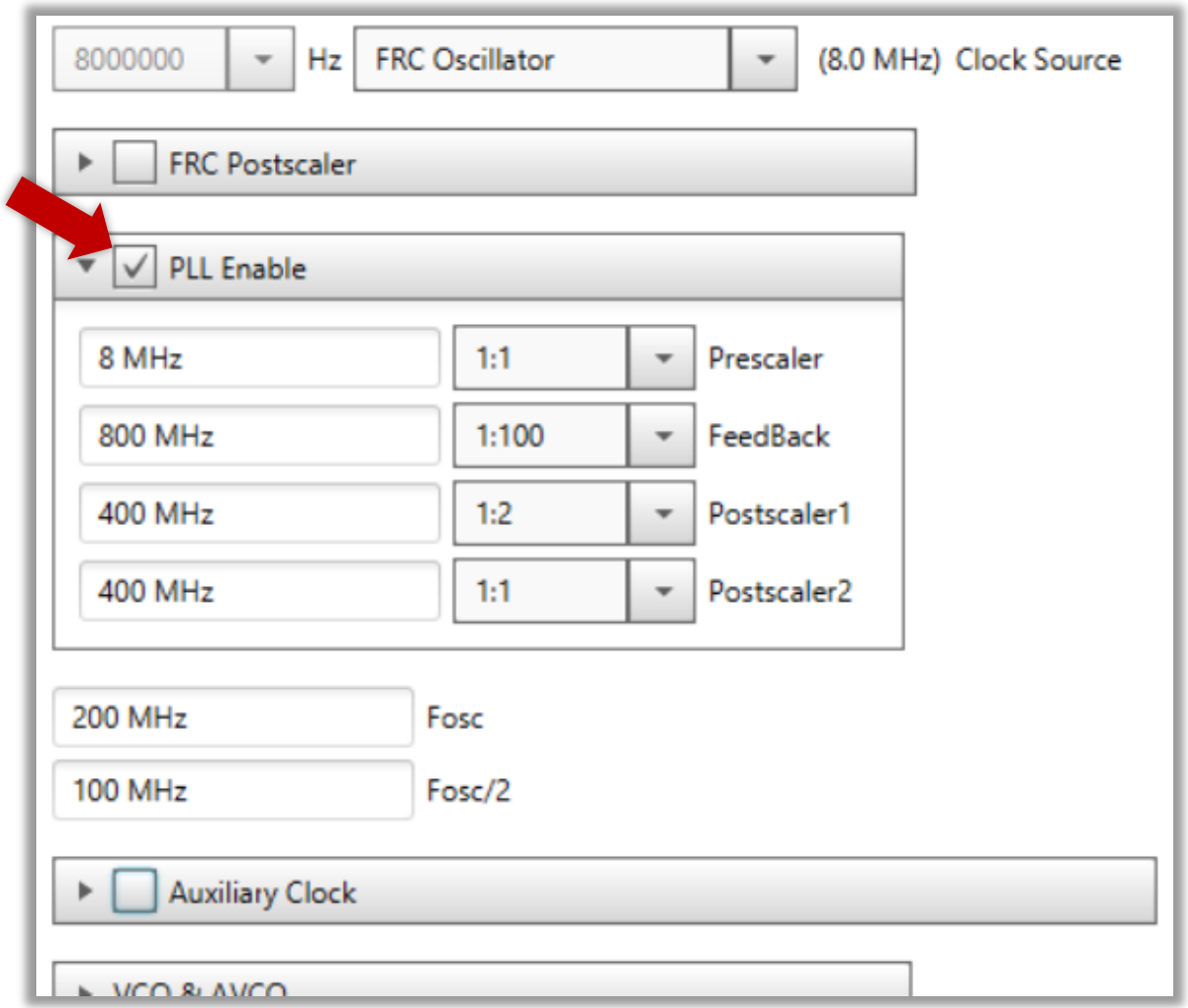
- **Disable FRC Postscaler**
- Enable PLL
- Enable Auxiliary Clock
- Set VCO/AVCO Dividers
- Set OSC2 as General Purpose IO
- Select ICD Port PGC1/PGD1



MICROCHIP

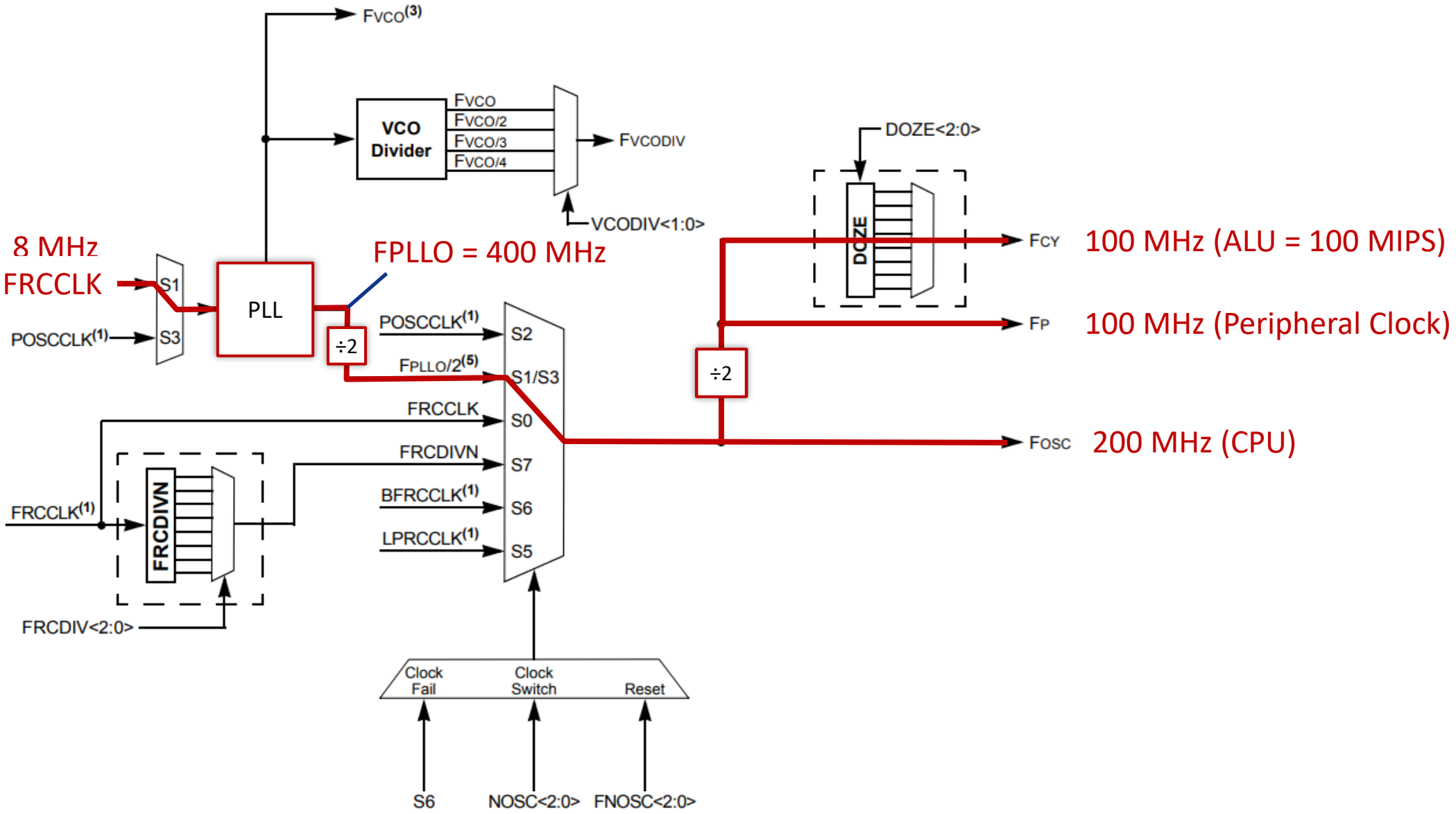
Configuring the System Oscillator

- Disable FRC Postscaler
- **Enable PLL**
- Enable Auxiliary Clock
- Set VCO/AVCO Dividers
- Set OSC2 as General Purpose IO
- Select ICD Port PGC1/PGD1



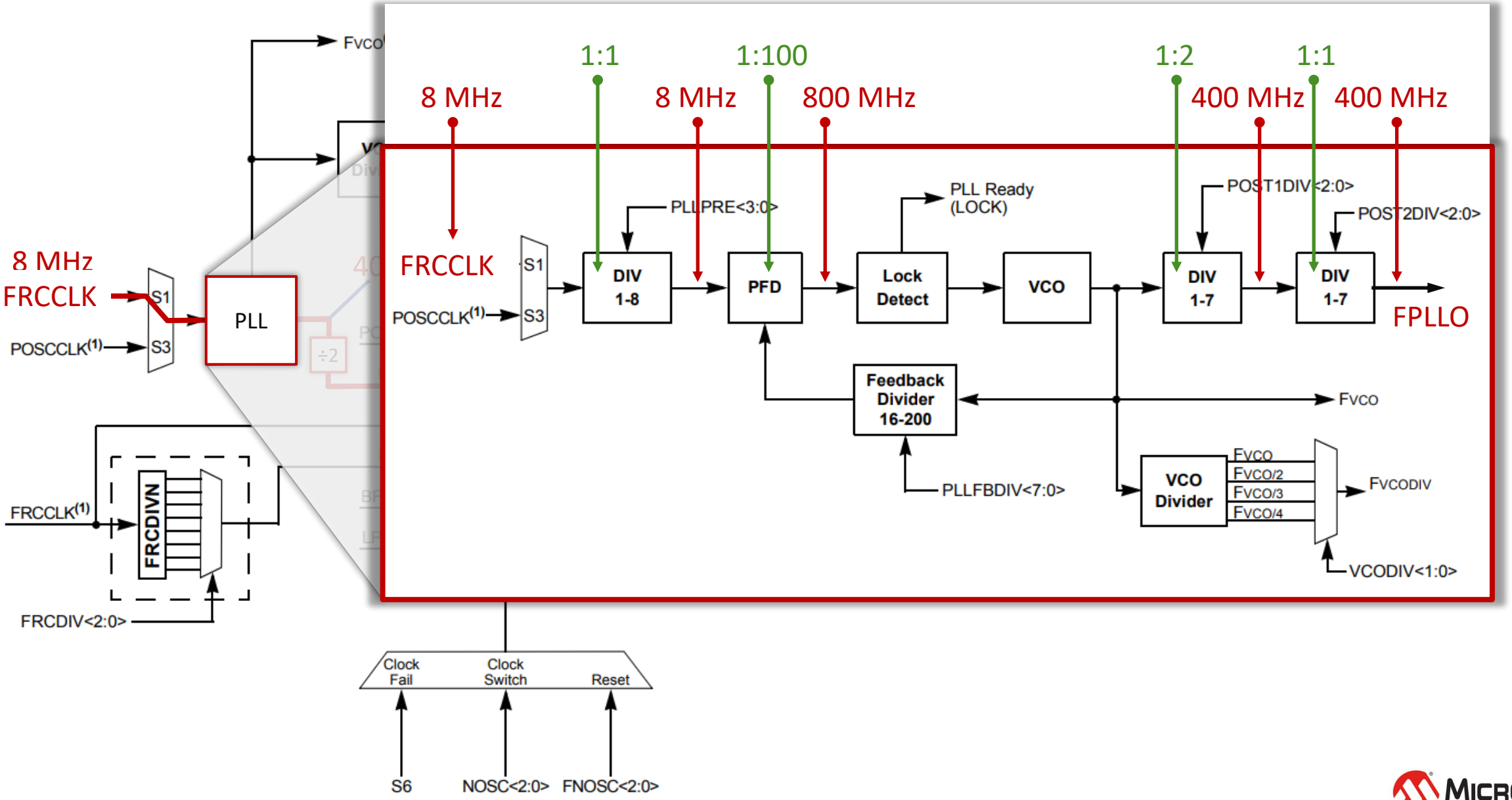
MICROCHIP

Configuring the System Oscillator



MICROCHIP

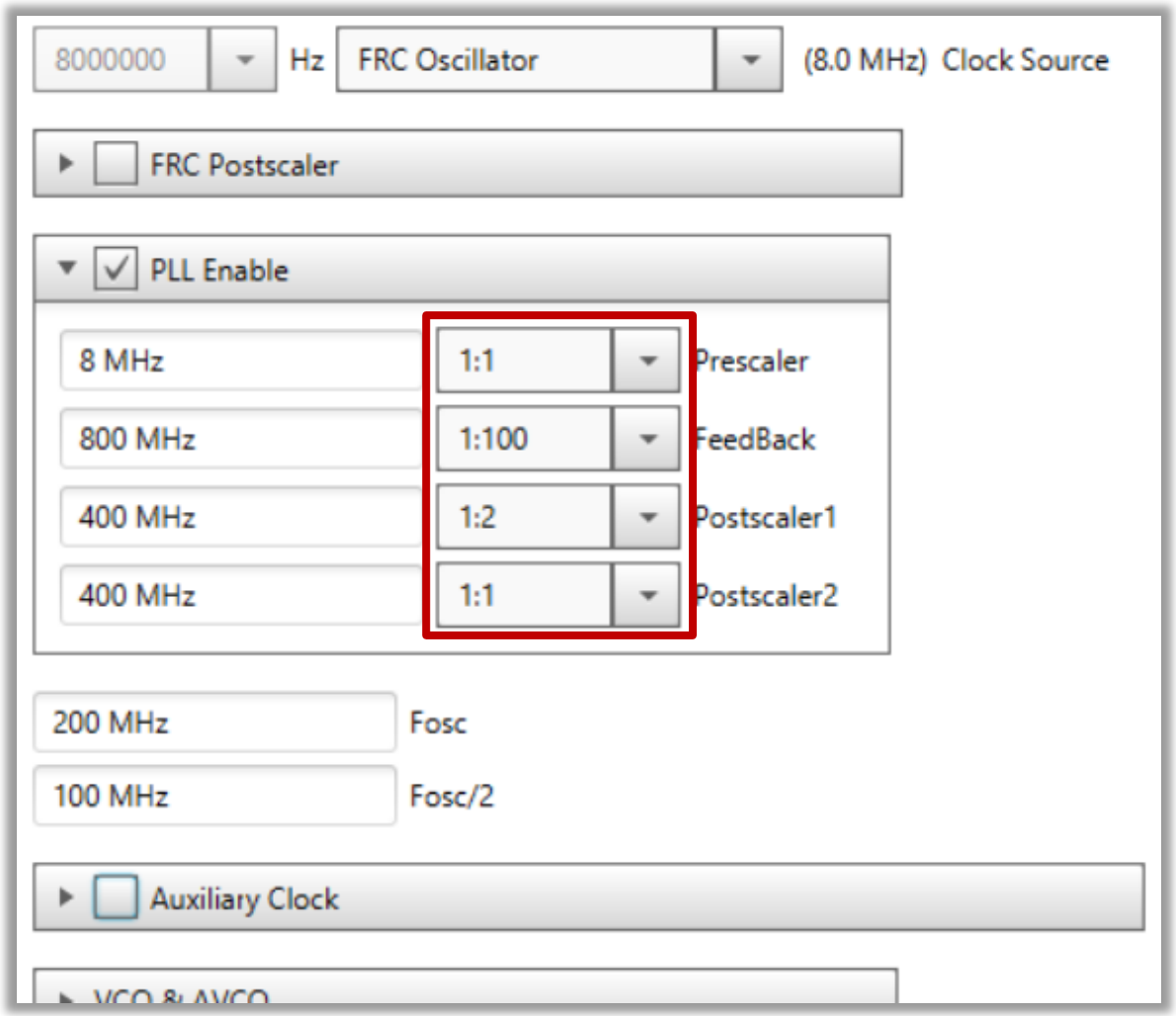
Configuring the System Oscillator



MICROCHIP

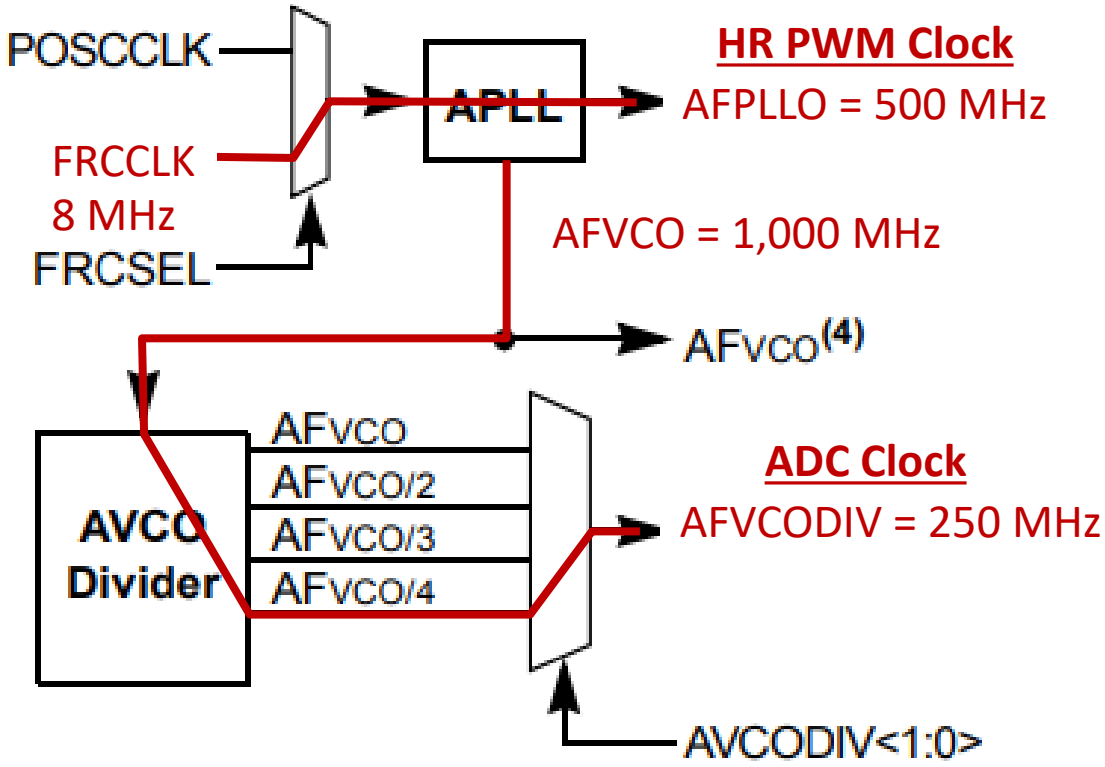
Configuring the System Oscillator

- Disable FRC Postscaler
- **Enable PLL**
  - Prescaler = 1:1
  - Feedback = 1:100
  - Postscaler 1 = 1:2
  - Postscaler 2 = 1:1
- Enable Auxiliary Clock
- Set VCO/AVCO Dividers
- Set OSC2 as General Purpose IO
- Select ICD Port PGC1/PGD1



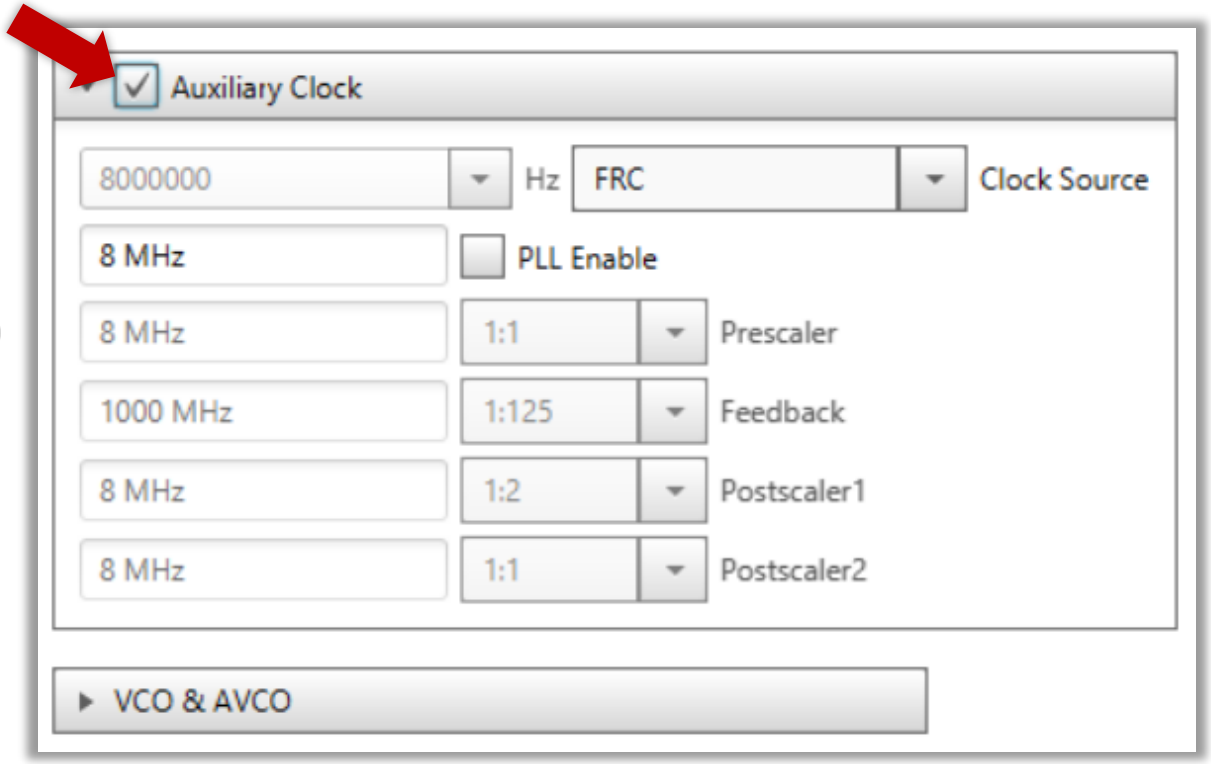
Configuring the System Oscillator

- **Auxiliary Clock Configuration**
  - PWM and ADC must be set on same clock domain to reduce clock jitter
  - Both, PWM and ADC, have their own, internal clock dividers



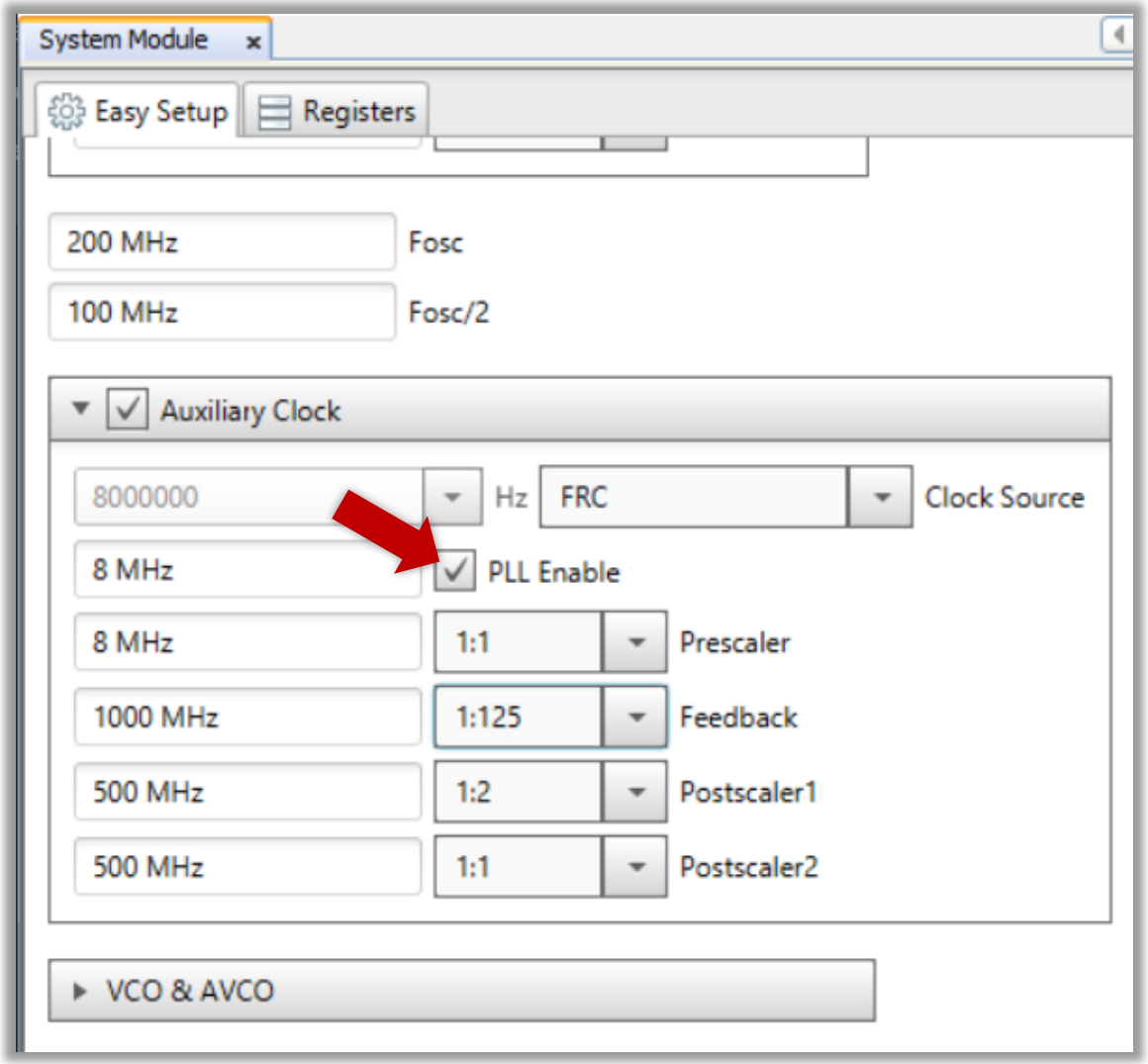
Configuring the System Oscillator

- Disable FRC Postscaler
- Enable PLL
- **Enable Auxiliary Clock**
- Set VCO/AVCO Dividers
- Set OSC2 as General Purpose IO
- Select ICD Port PGC1/PGD1



Configuring the System Oscillator

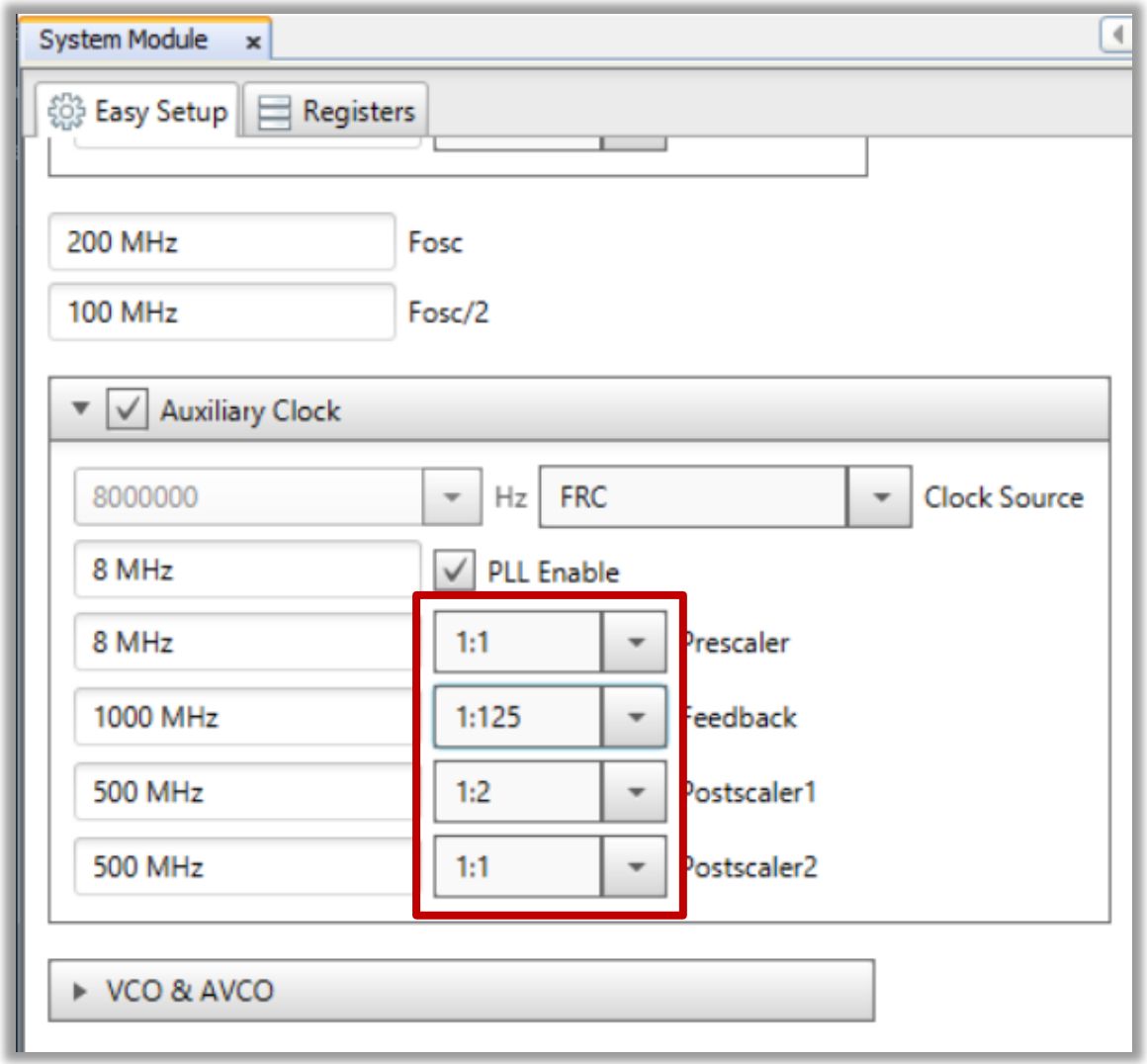
- Disable FRC Postscaler
- Enable PLL
- **Enable Auxiliary Clock**
  - Enable PLL
  - Prescaler = 1:1
  - Feedback = 1:125
  - Postscaler 1 = 1:2
  - Postscaler 2 = 1:1
- Set VCO/AVCO Dividers
- Set OSC2 as General Purpose IO
- Select ICD Port PGC1/PGD1



MICROCHIP

Configuring the System Oscillator

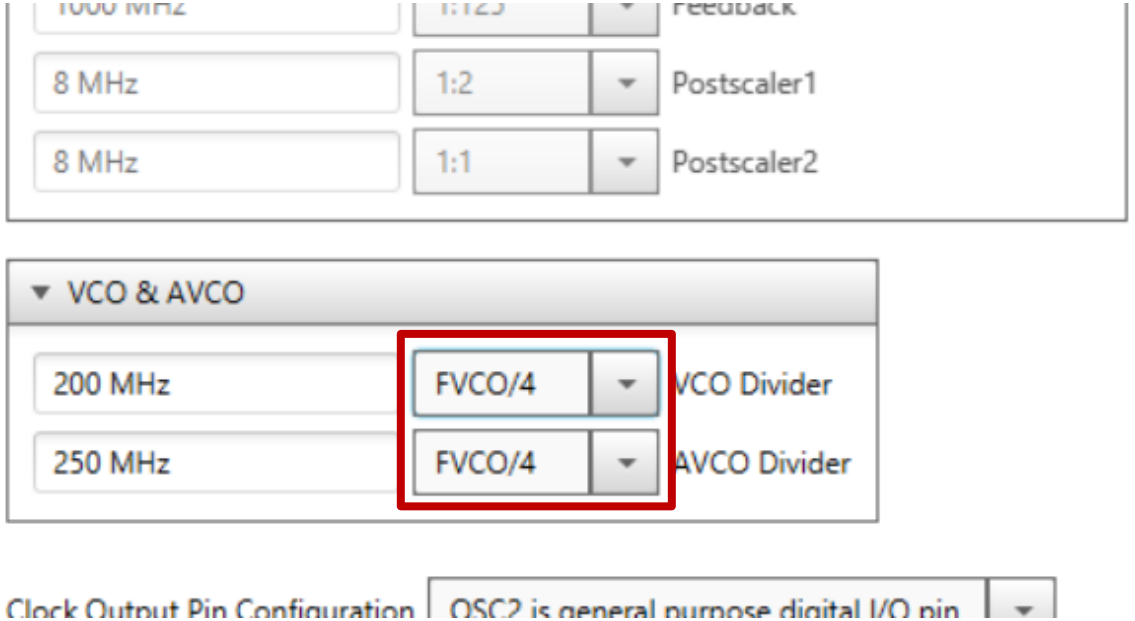
- Disable FRC Postscaler
- Enable PLL
- **Enable Auxiliary Clock**
  - Enable PLL
  - Prescaler = 1:1
  - Feedback = 1:125
  - Postscaler 1 = 1:2
  - Postscaler 2 = 1:1
- Set VCO/AVCO Dividers
- Set OSC2 as General Purpose IO
- Select ICD Port PGC1/PGD1



MICROCHIP

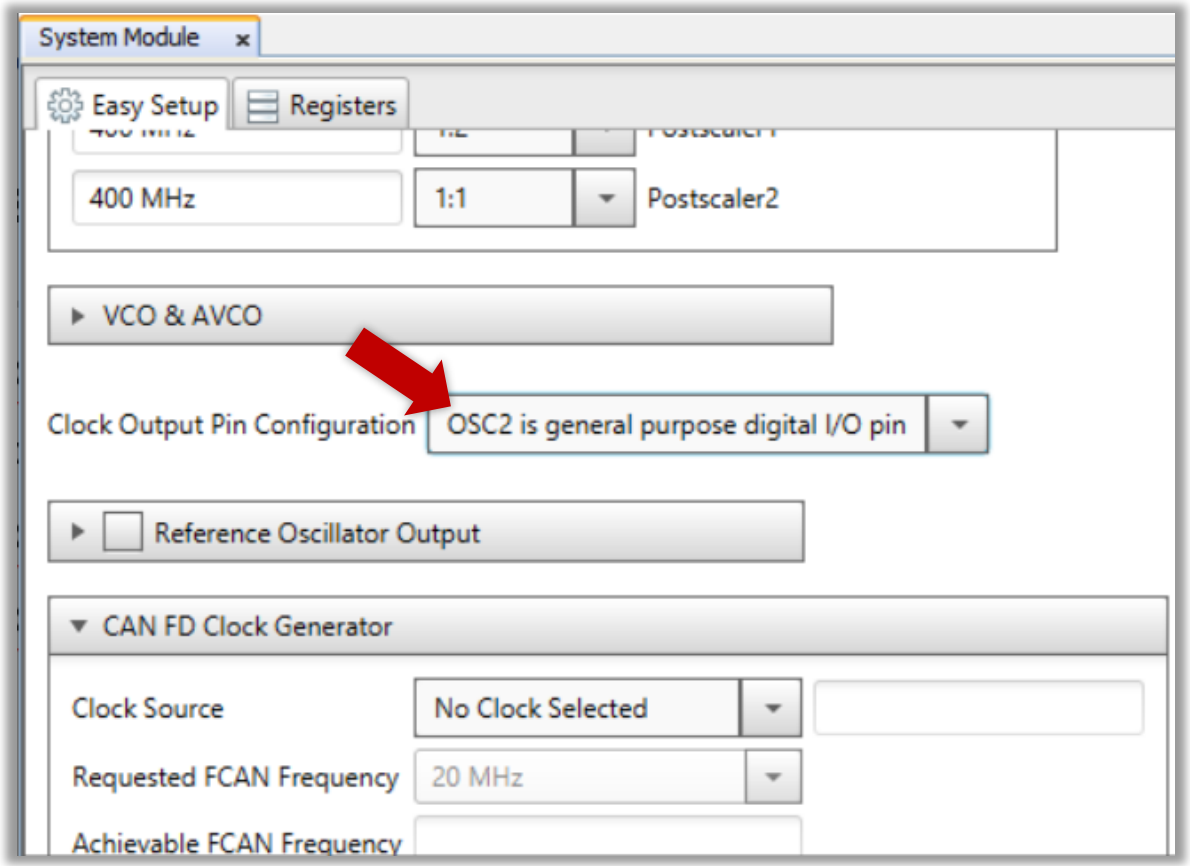
Configuring the System Oscillator

- Disable FRC Postscaler
- Enable PLL
- Enable Auxiliary Clock
- **Set VCO/AVCO Dividers**
- Set OSC2 as General Purpose IO
- Select ICD Port PGC1/PGD1



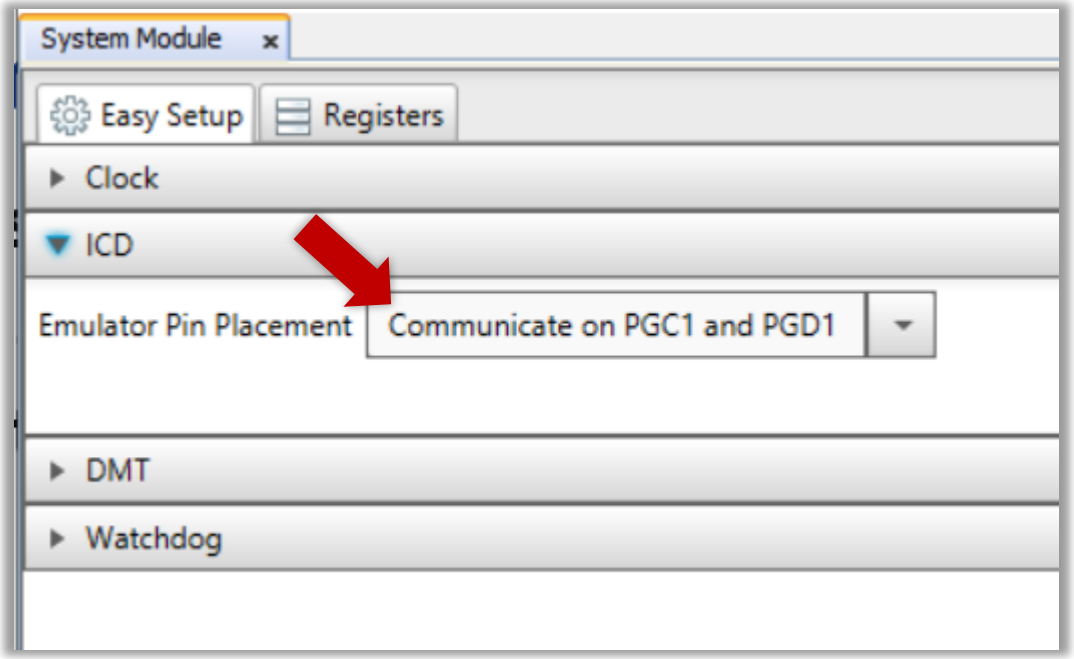
Configuring the System Oscillator

- Disable FRC Postscaler
- Enable PLL
- Enable Auxiliary Clock
- Set VCO/AVCO Dividers
- **Set OSC2 as General Purpose IO**
- Select ICD Port PGC1/PGD1



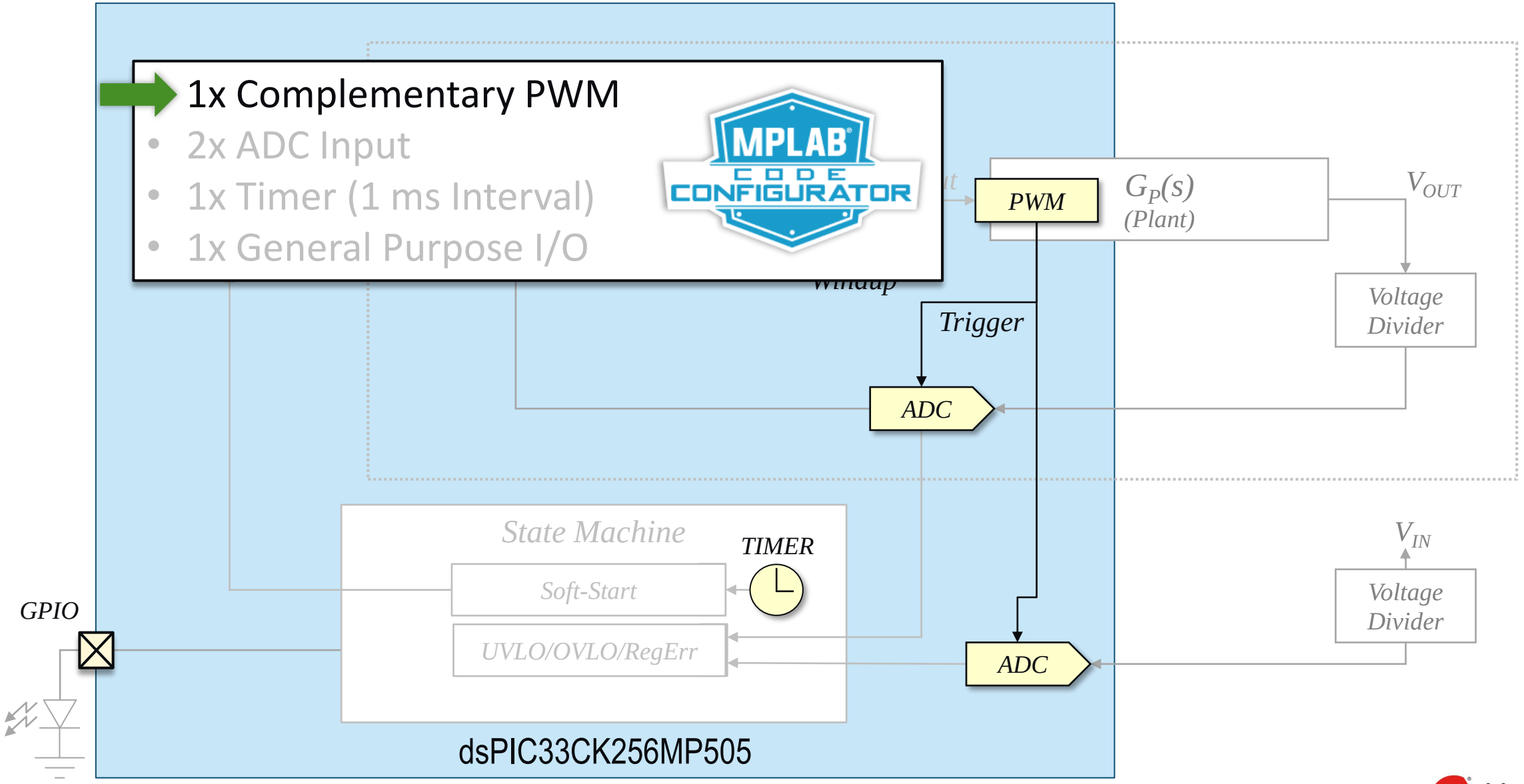
Configuring the System Oscillator

- Disable FRC Postscaler
- Enable PLL
- Enable Auxiliary Clock
- Set VCO/AVCO Dividers
- Set OSC2 as General Purpose IO
- **Select ICD Port PGC1/PGD1**



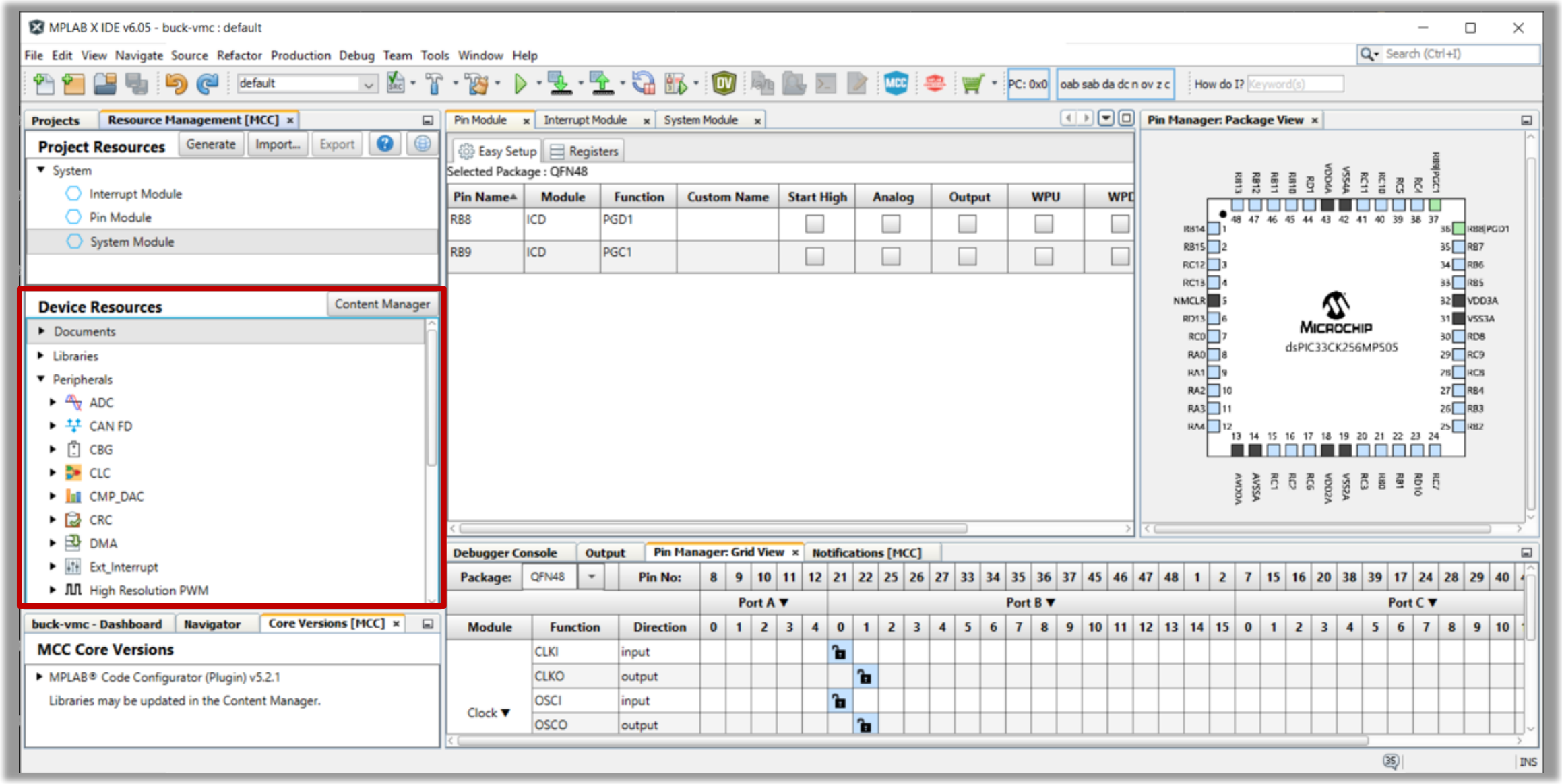
MICROCHIP

Requirements

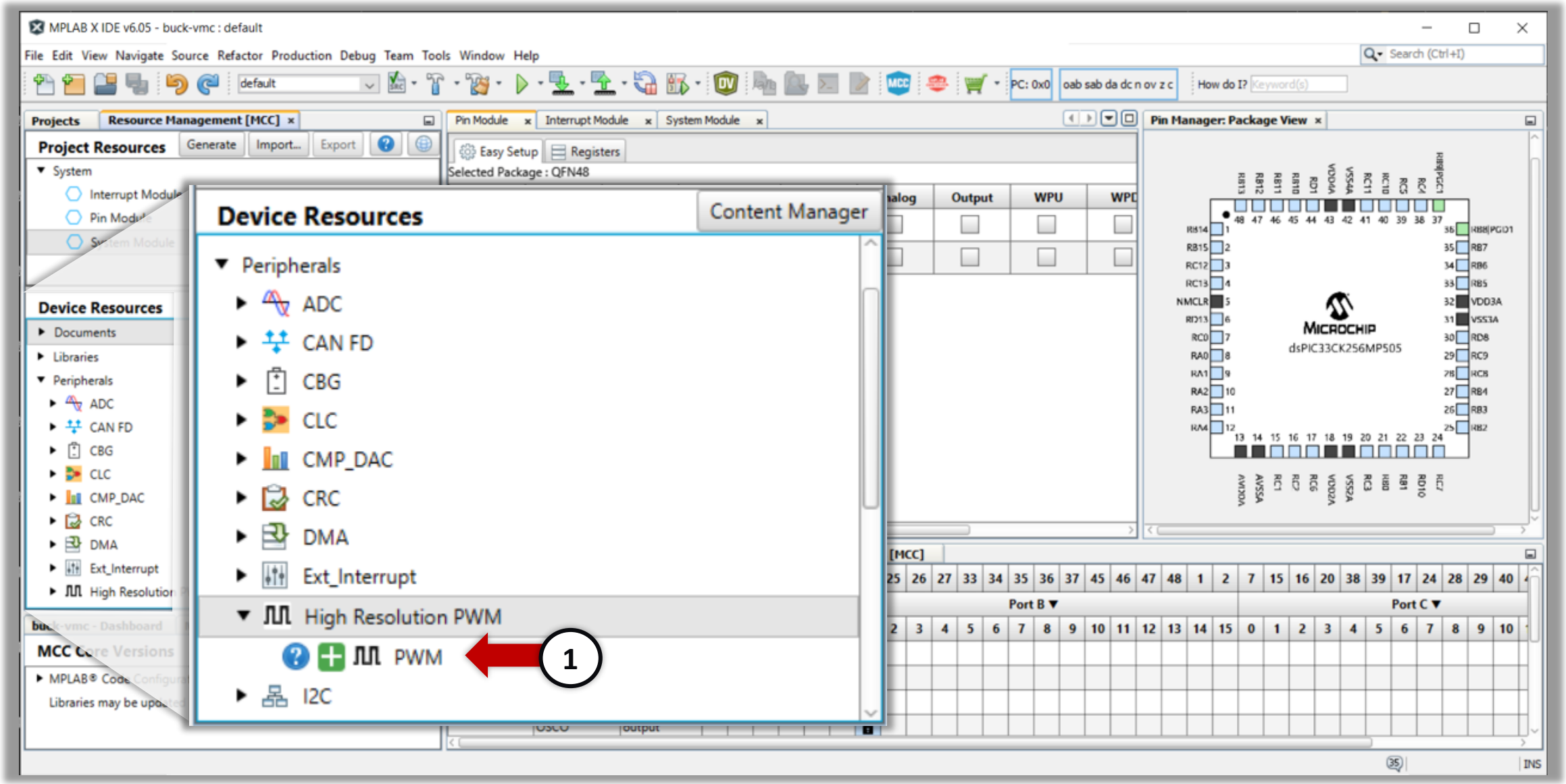


MICROCHIP

Configuring the PWM

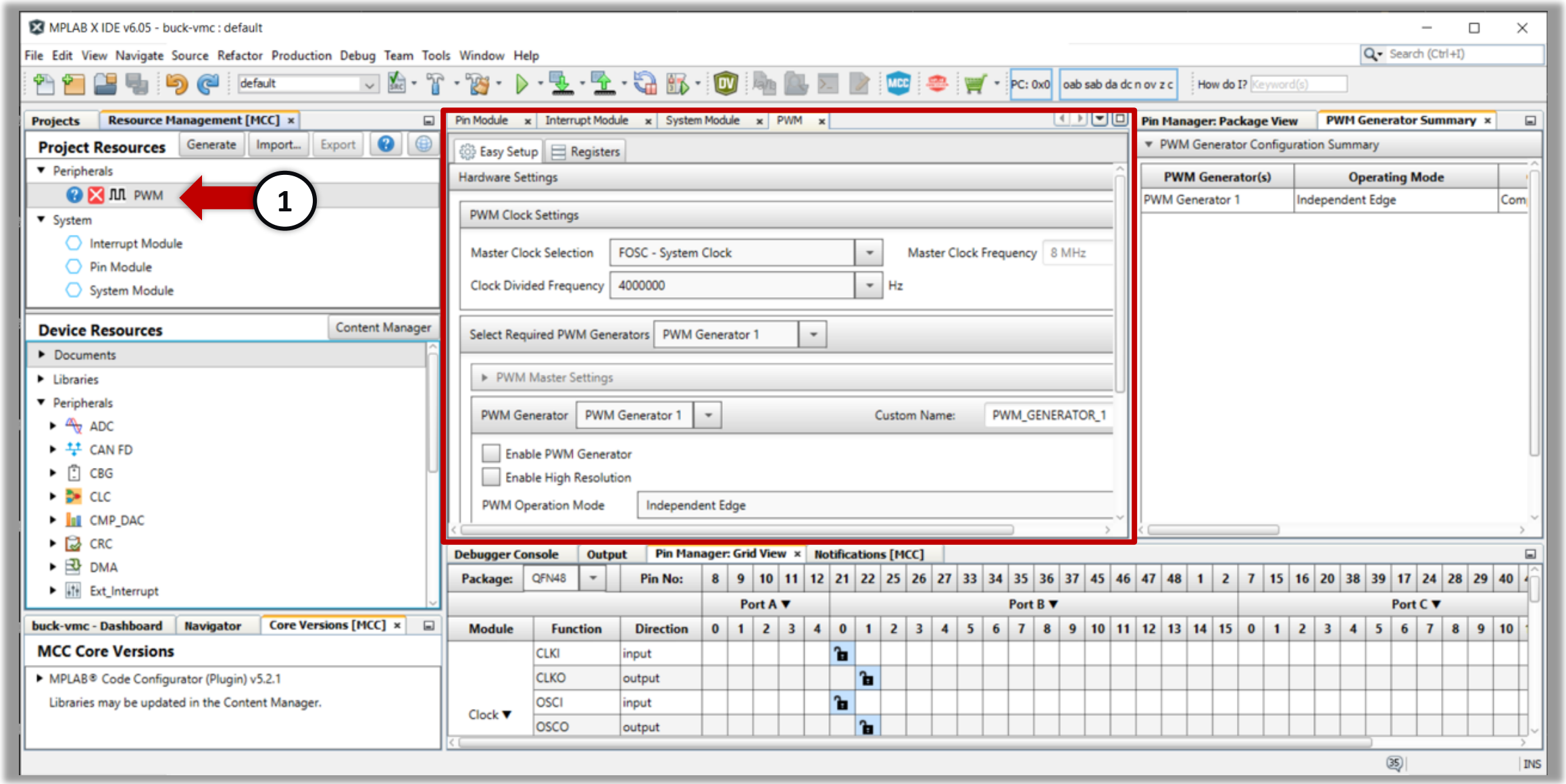


Configuring the PWM



MICROCHIP

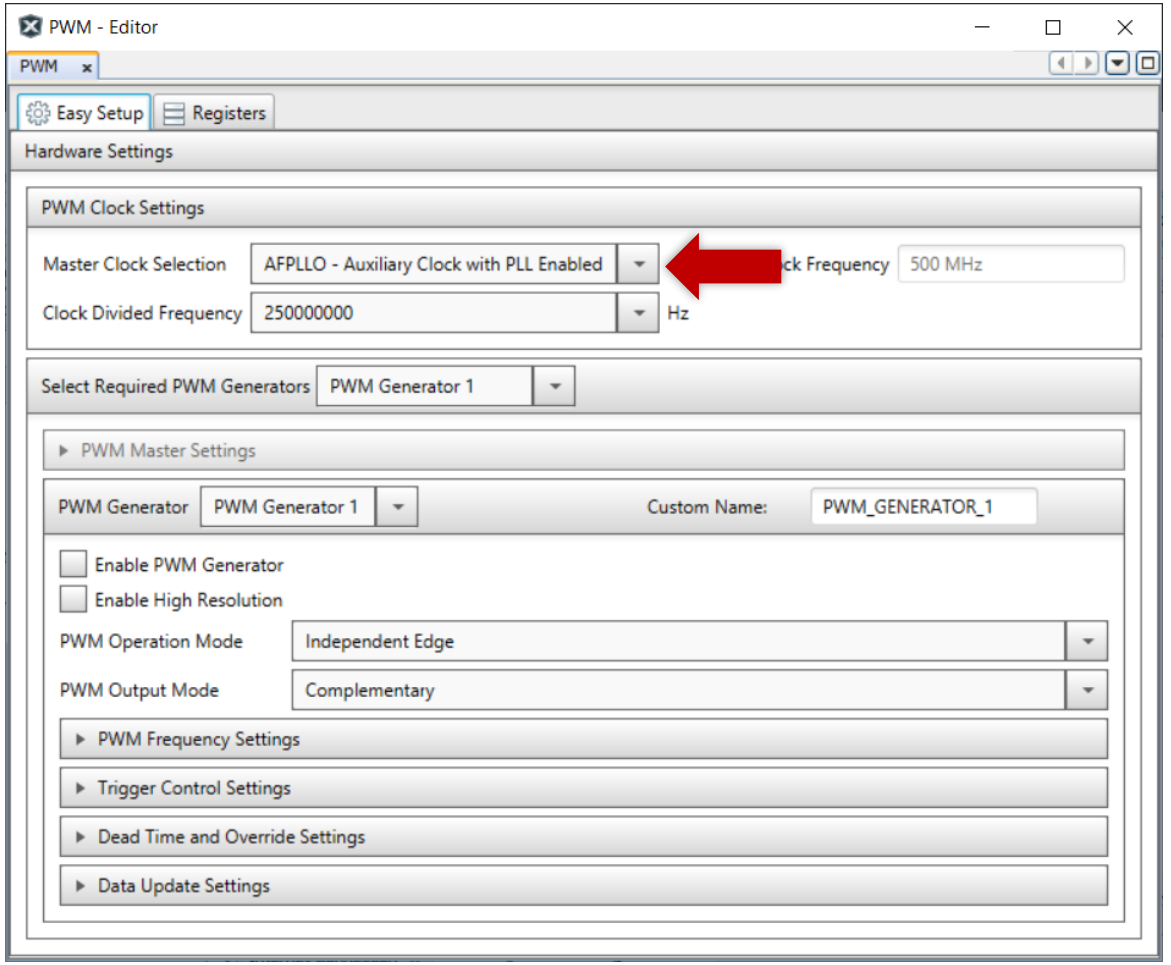
Configuring the PWM



MICROCHIP

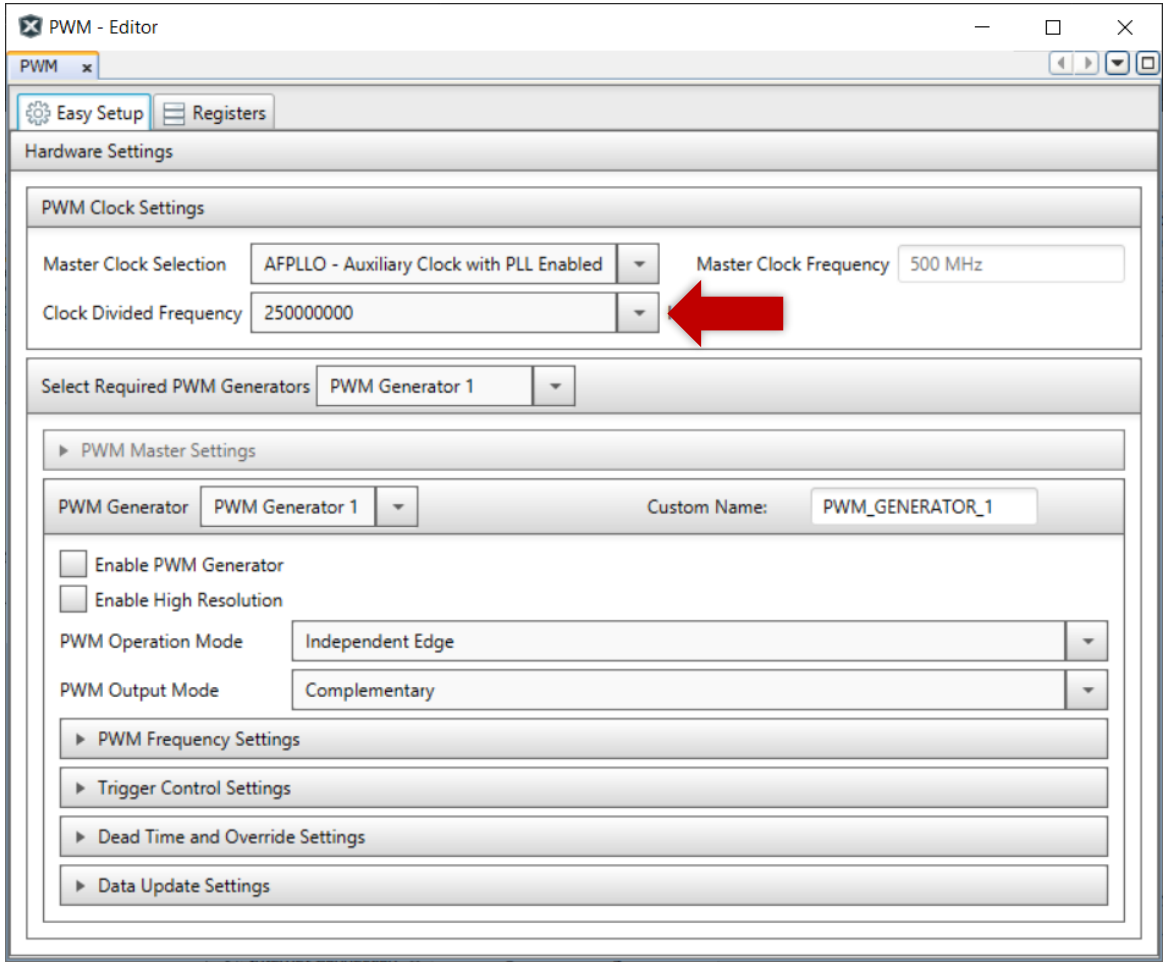
Configuring the PWM

- **Set Input Clock (AFPLLO=500 MHz)**



Configuring the PWM

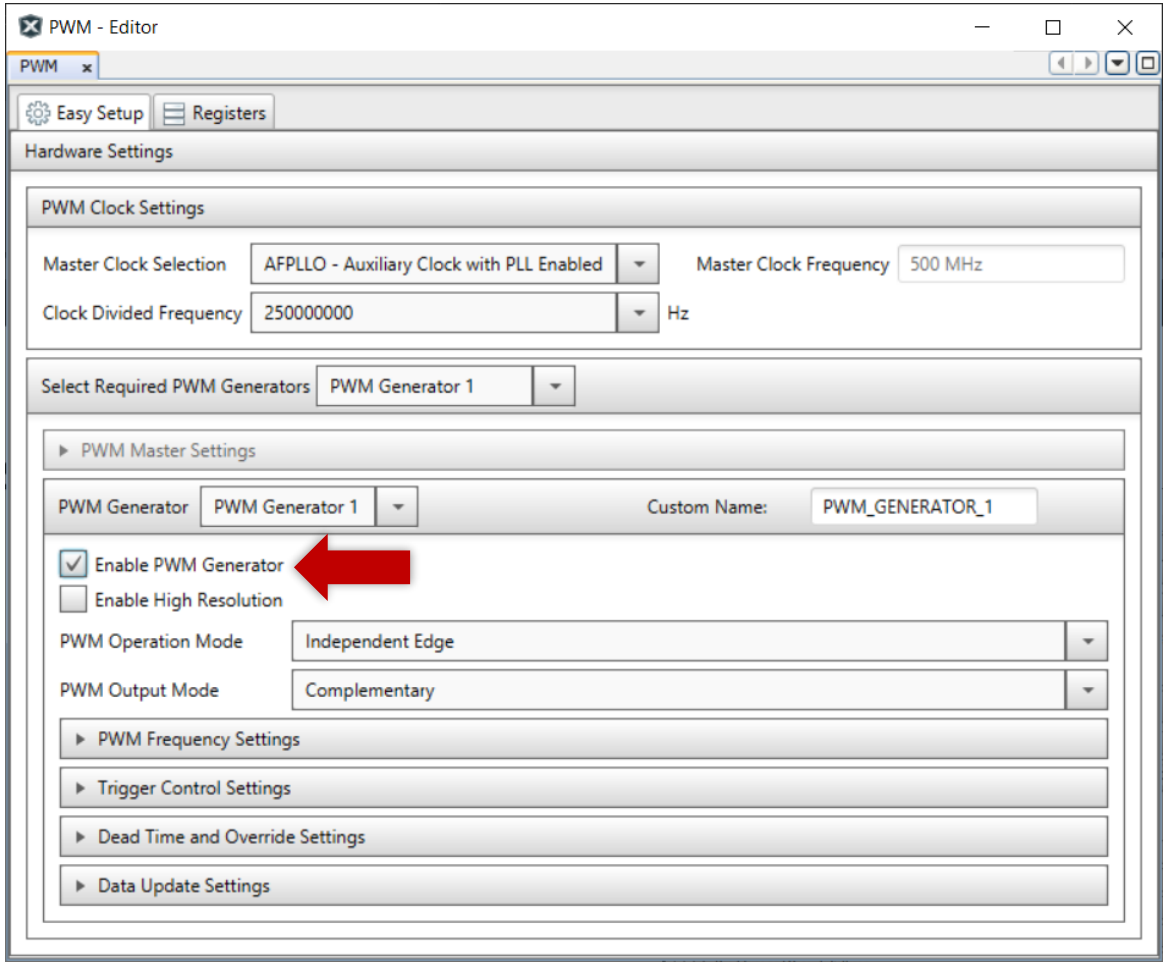
- Set Input Clock (AFPLLO=500 MHz)
- **Select Clock Divider Frequency (=250 MHz)**



MICROCHIP

Configuring the PWM

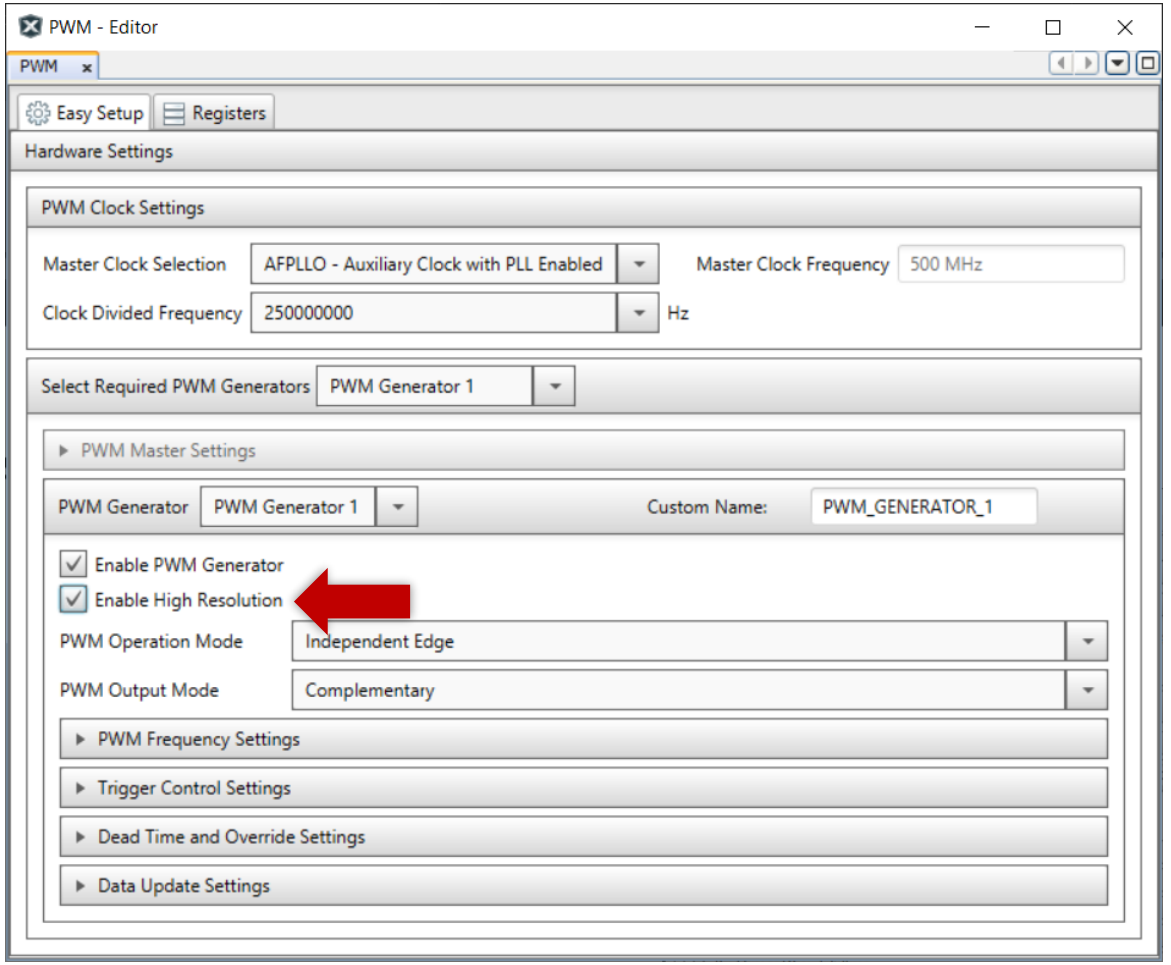
- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- **Enable PWM Generator #1**



MICROCHIP

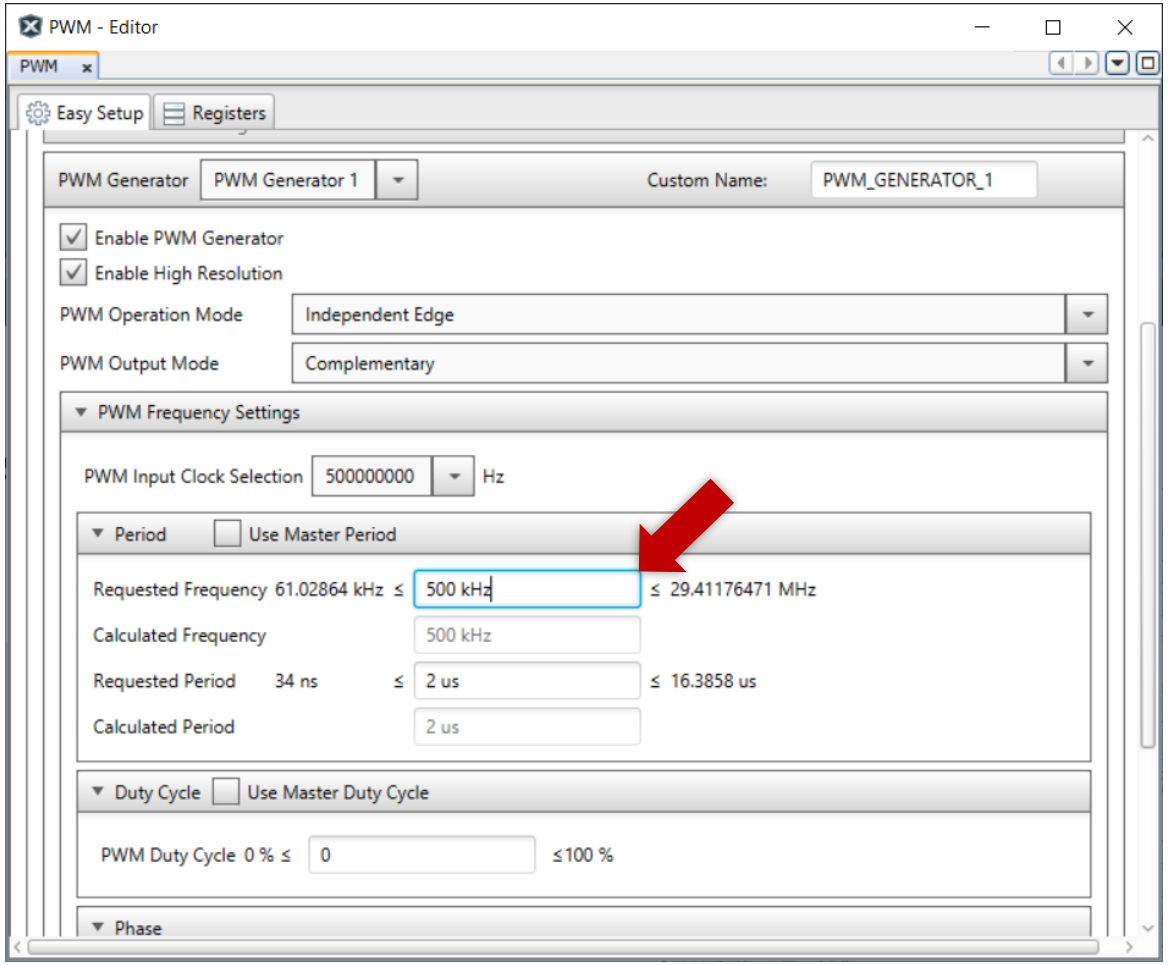
Configuring the PWM

- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- **Enable High Resolution**



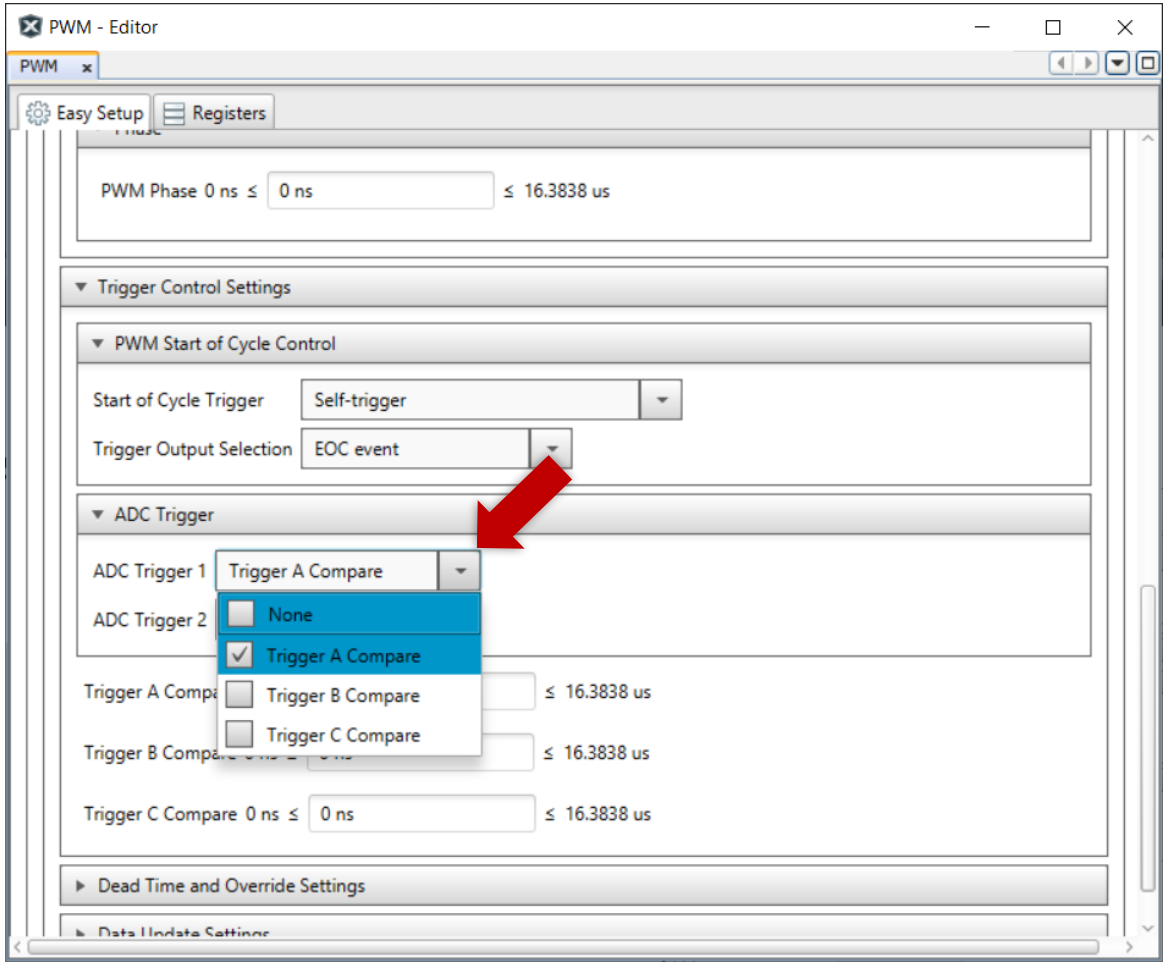
Configuring the PWM

- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- Enable High Resolution
- **Set PWM Frequencies (=500 kHz)**



Configuring the PWM

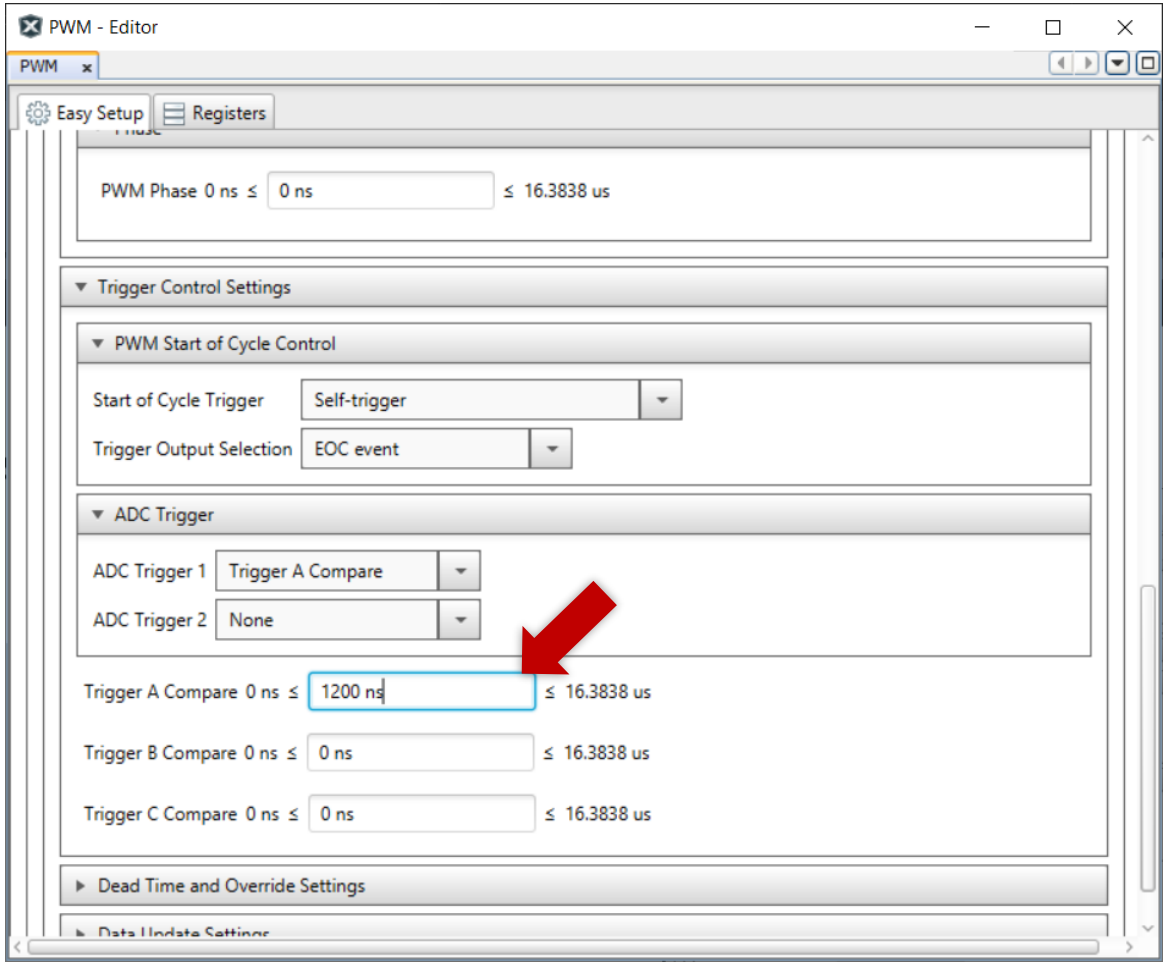
- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- Enable High Resolution
- Set PWM Frequencies (=500 kHz)
- **Set ADC Trigger 1 (=Trigger A Compare)**



MICROCHIP

Configuring the PWM

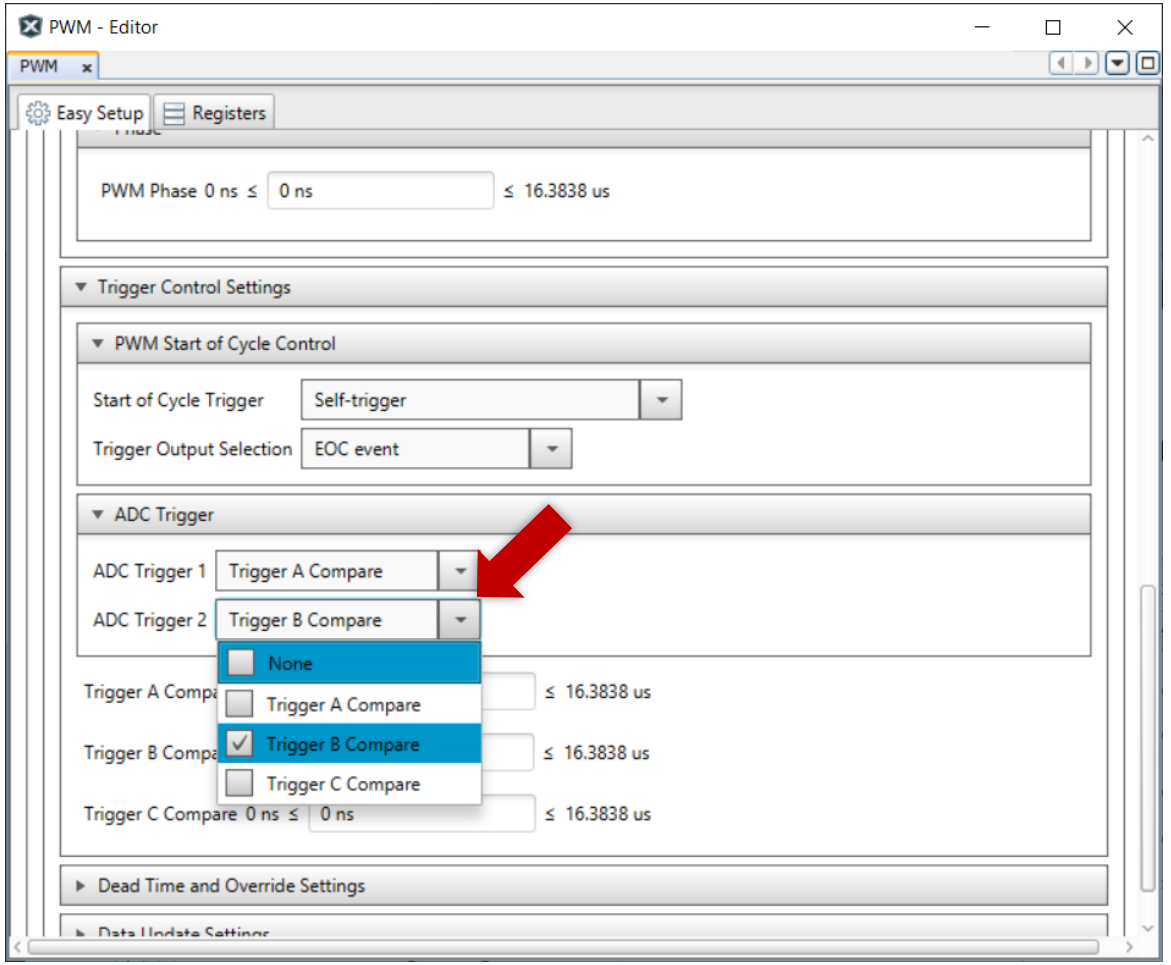
- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- Enable High Resolution
- Set PWM Frequencies (=500 kHz)
- Set ADC Trigger 1 (=Trigger A Compare)
- **Set Trigger A Compare (=1200 ns)**



MICROCHIP

Configuring the PWM

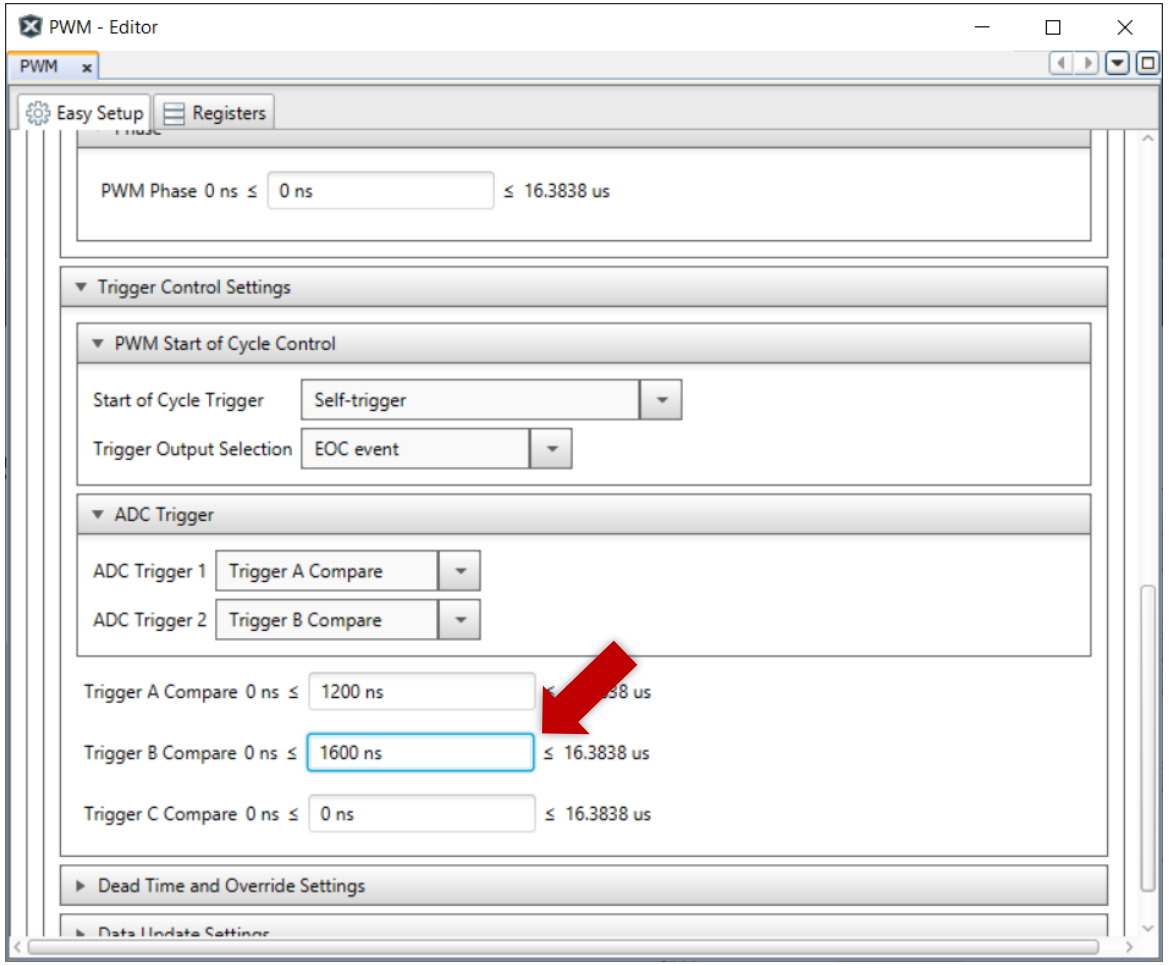
- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- Enable High Resolution
- Set PWM Frequencies (=500 kHz)
- Set ADC Trigger 1 (=Trigger A Compare)
- Set Trigger A Compare (=1200 ns)
- **Set ADC Trigger 2 (=Trigger B Compare)**



MICROCHIP

Configuring the PWM

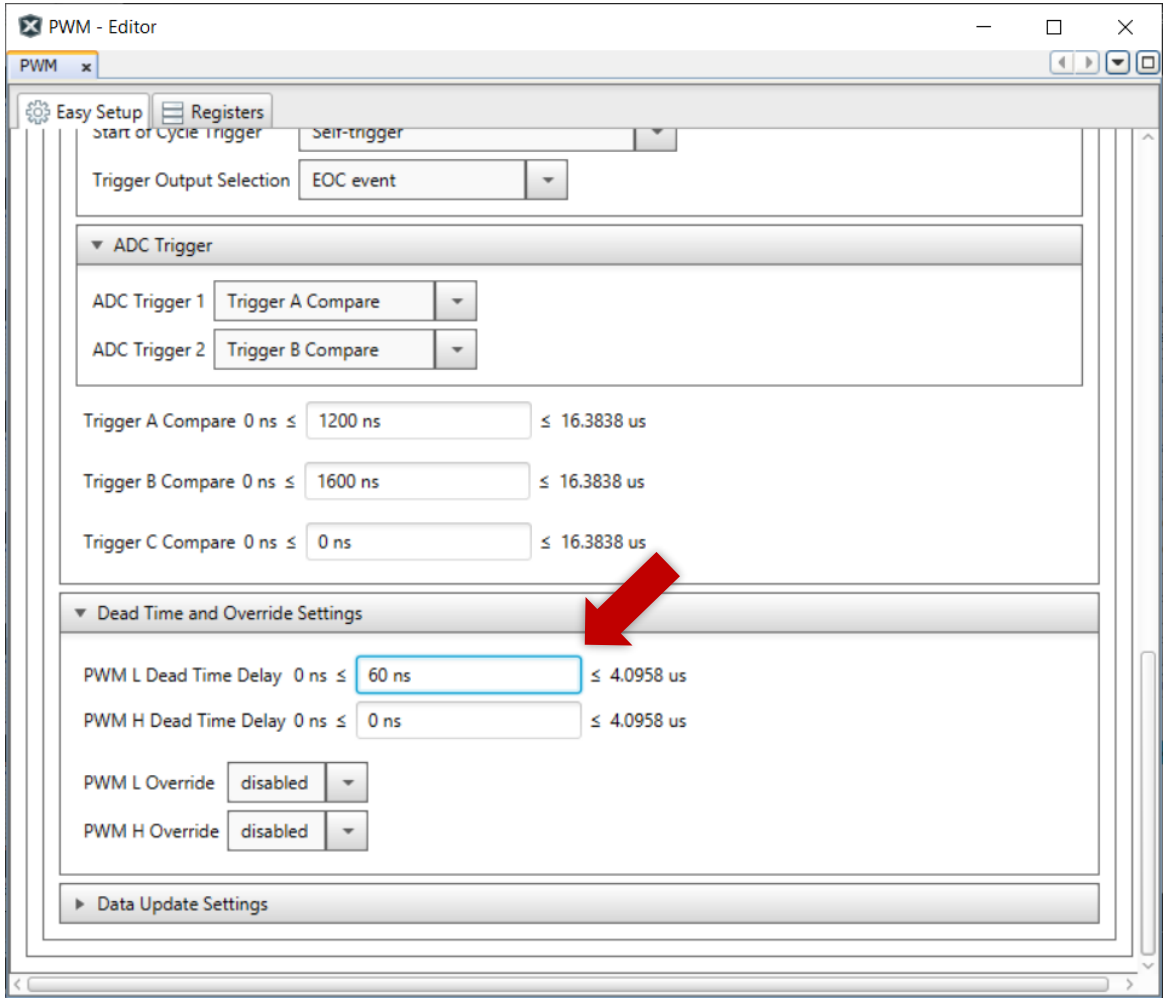
- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- Enable High Resolution
- Set PWM Frequencies (=500 kHz)
- Set ADC Trigger 1 (=Trigger A Compare)
- Set Trigger A Compare (=1200 ns)
- Set ADC Trigger 2 (=Trigger B Compare)
- **Set Trigger B Compare (=1600 ns)**



MICROCHIP

Configuring the PWM

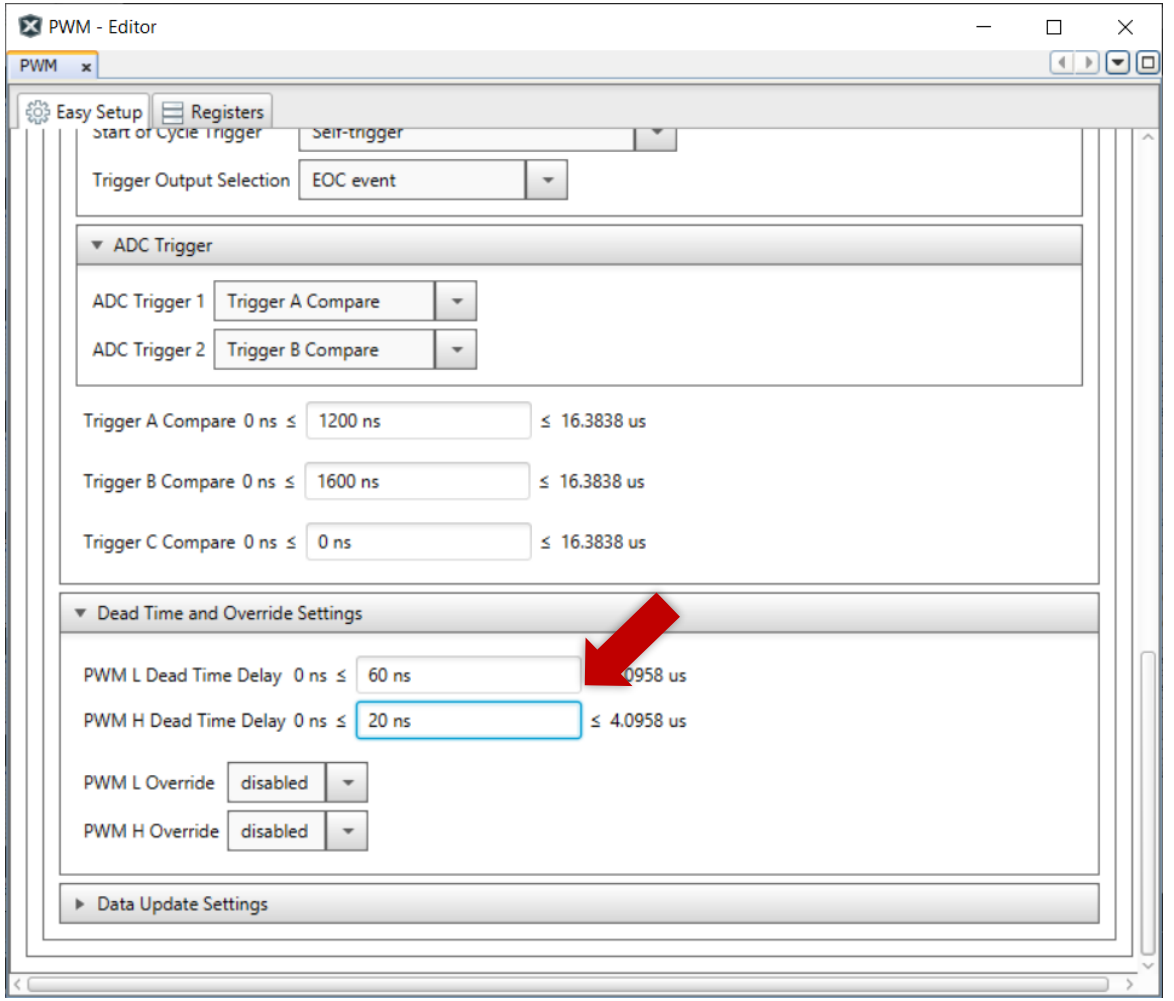
- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- Enable High Resolution
- Set PWM Frequencies (=500 kHz)
- Set ADC Trigger 1 (=Trigger A Compare)
- Set Trigger A Compare (=1200 ns)
- Set ADC Trigger 2 (=Trigger B Compare)
- Set Trigger B Compare (=1600 ns)
- **Set PWM L Dead Time (=60 ns)**



MICROCHIP

Configuring the PWM

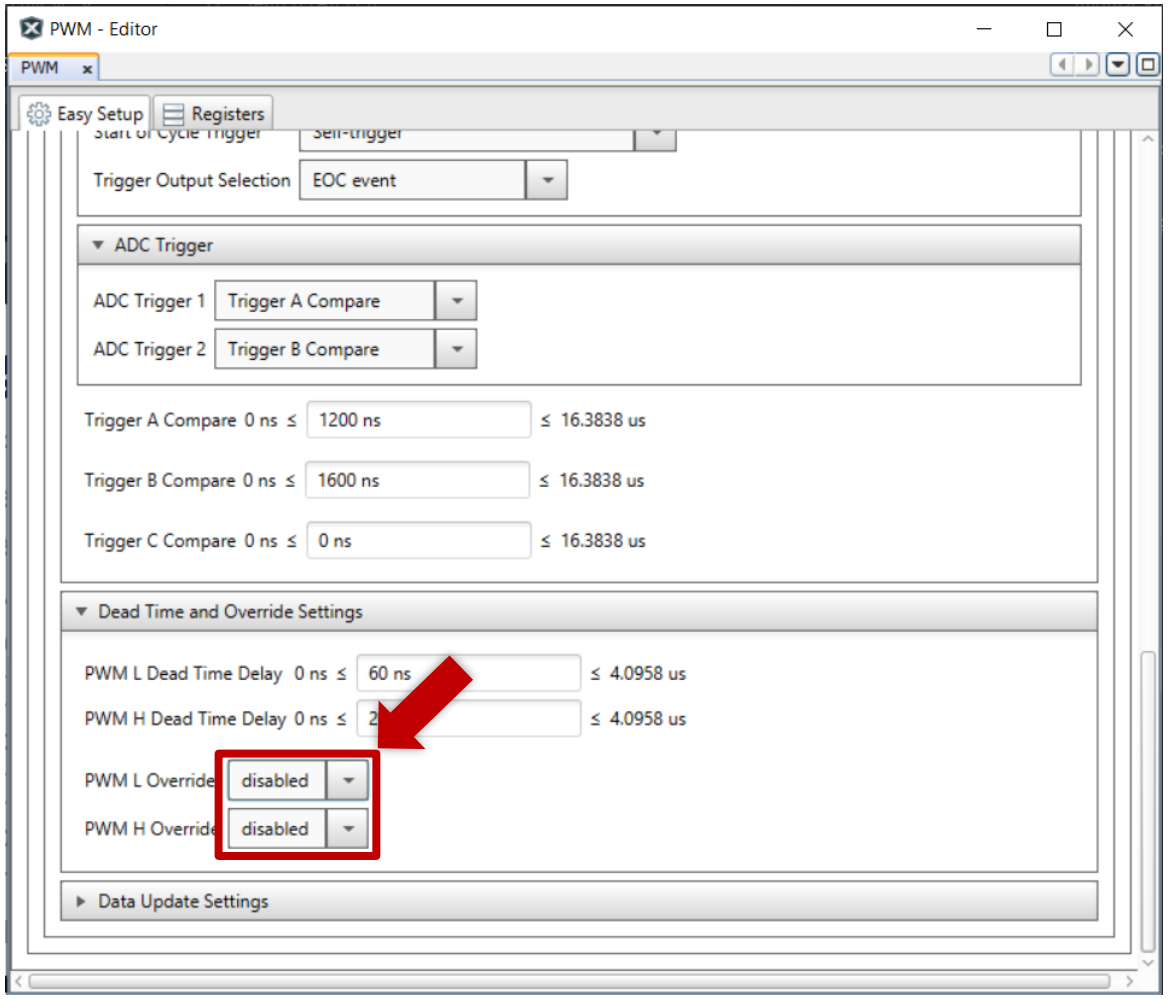
- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- Enable High Resolution
- Set PWM Frequencies (=500 kHz)
- Set ADC Trigger 1 (=Trigger A Compare)
- Set Trigger A Compare (=1200 ns)
- Set ADC Trigger 2 (=Trigger B Compare)
- Set Trigger B Compare (=1600 ns)
- Set PWM L Dead Time (=60 ns)
- **Set PWM H Dead Time (=20 ns)**



MICROCHIP

Configuring the PWM

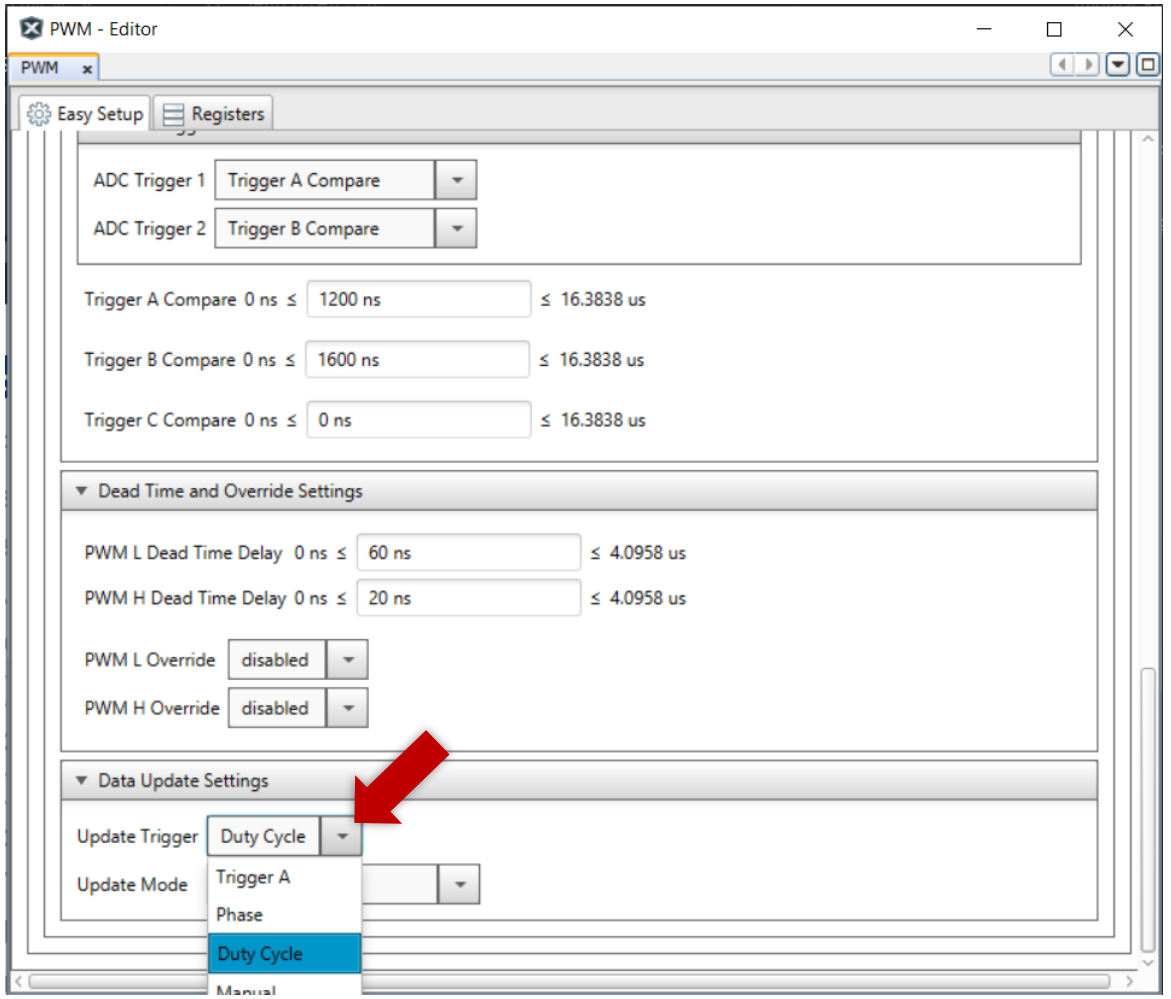
- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- Enable High Resolution
- Set PWM Frequencies (=500 kHz)
- Set ADC Trigger 1 (=Trigger A Compare)
- Set Trigger A Compare (=1200 ns)
- Set ADC Trigger 2 (=Trigger B Compare)
- Set Trigger B Compare (=1600 ns)
- Set PWM L Dead Time (=60 ns)
- Set PWM H Dead Time (=20 ns)
- **Set PWM L/H Override (=disabled)**



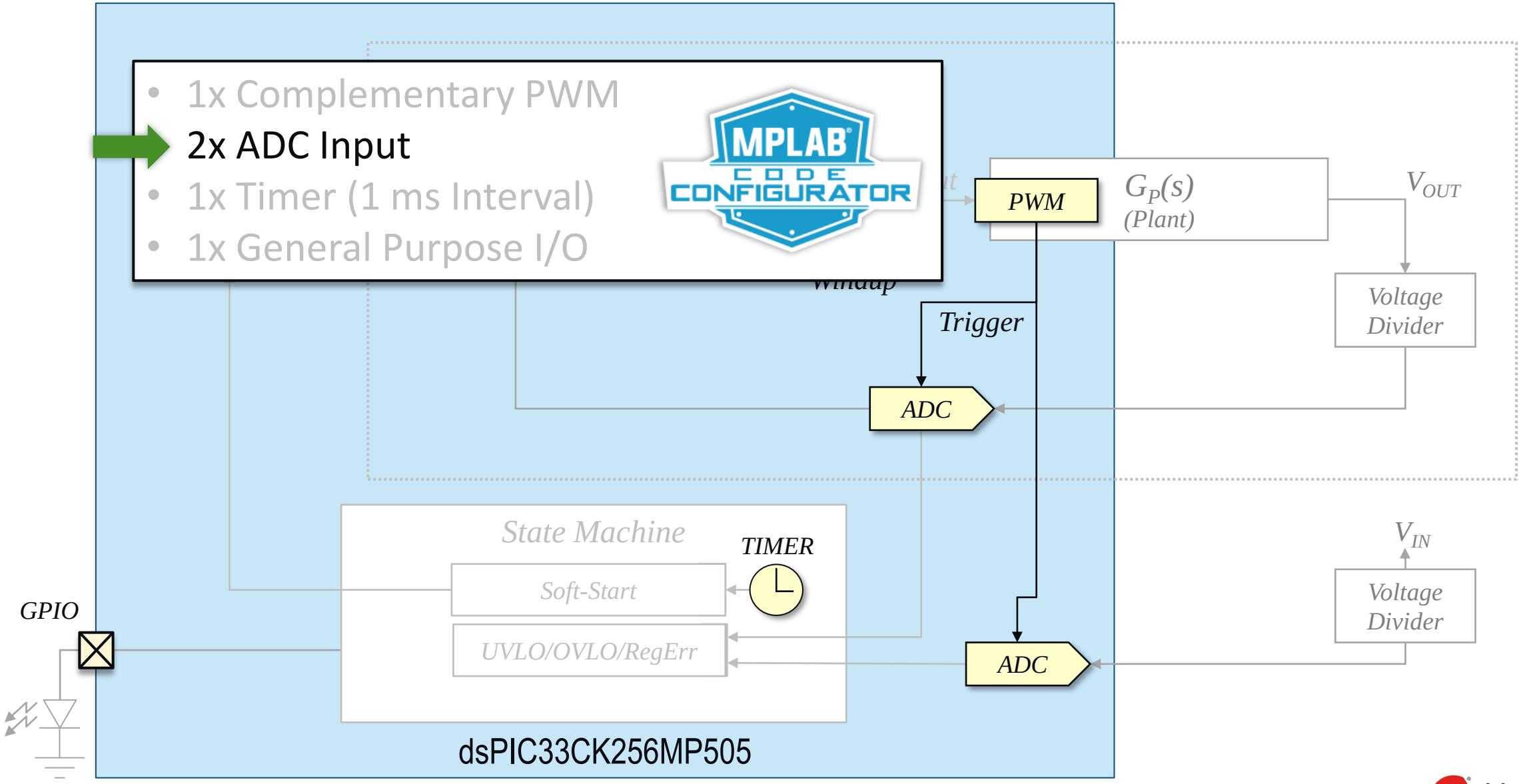
MICROCHIP

Configuring the PWM

- Set Input Clock (AFPLLO=500 MHz)
- Select Clock Divider Frequency
- Enable PWM Generator #1
- Enable High Resolution
- Set PWM Frequencies (=500 kHz)
- Set ADC Trigger 1 (=Trigger A Compare)
- Set Trigger A Compare (=1200 ns)
- Set ADC Trigger 2 (=Trigger B Compare)
- Set Trigger B Compare (=1600 ns)
- Set PWM L Dead Time (=60 ns)
- Set PWM H Dead Time (=20 ns)
- Set PWM L/H Override (=disabled)
- **Set Data Update Trigger (=Duty Cycle)**

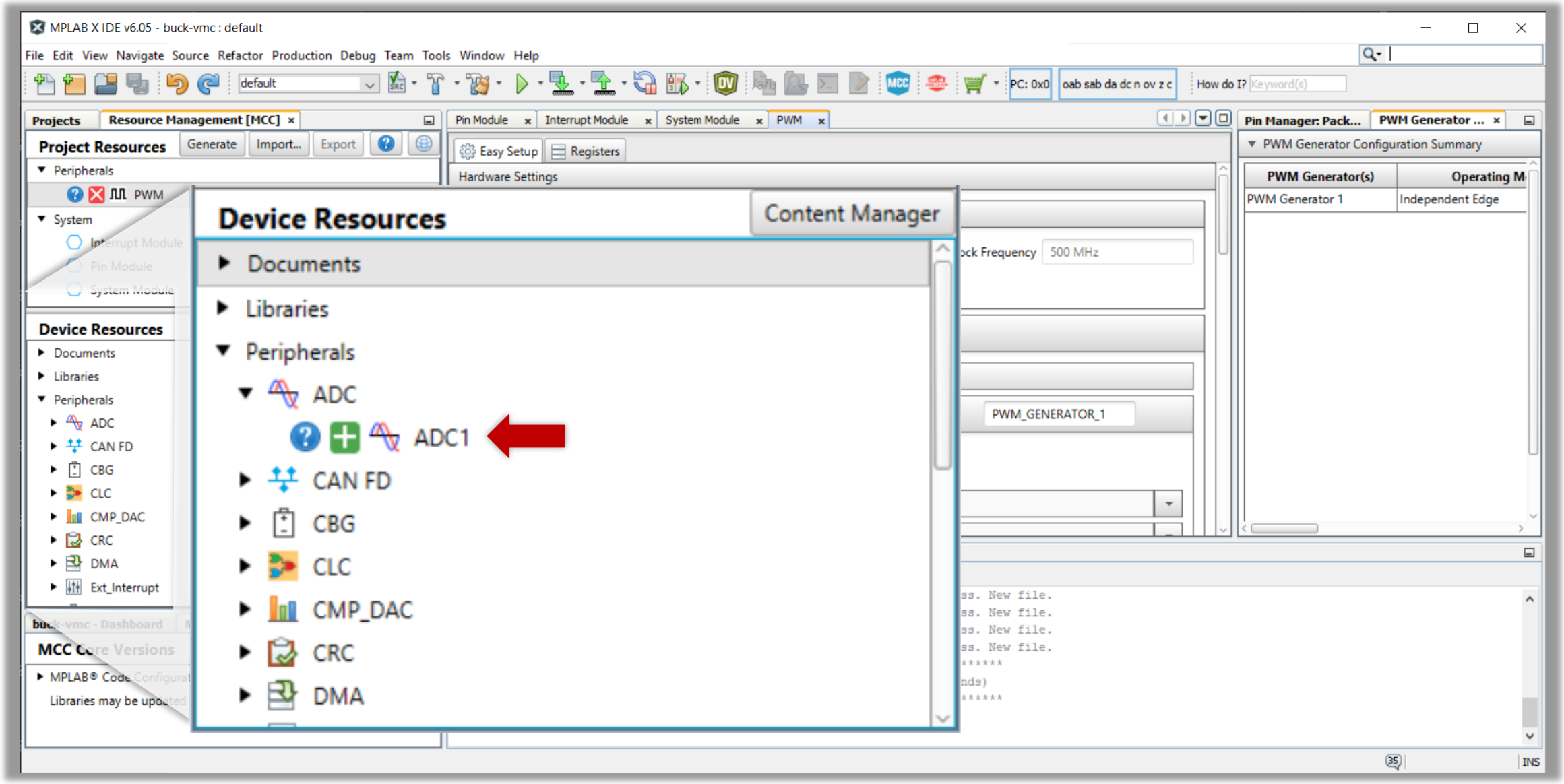


Requirements



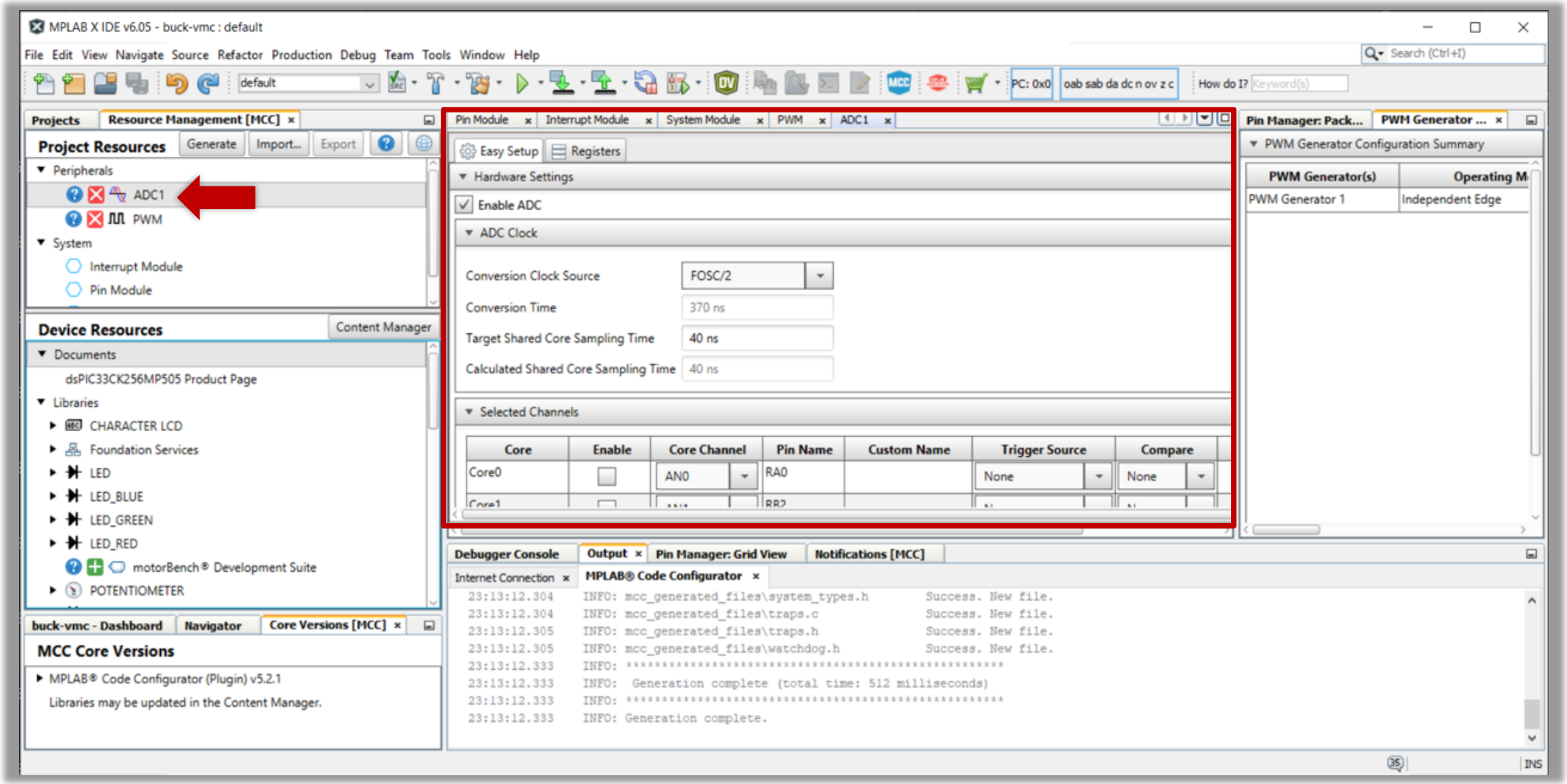
MICROCHIP

Configuring the ADC



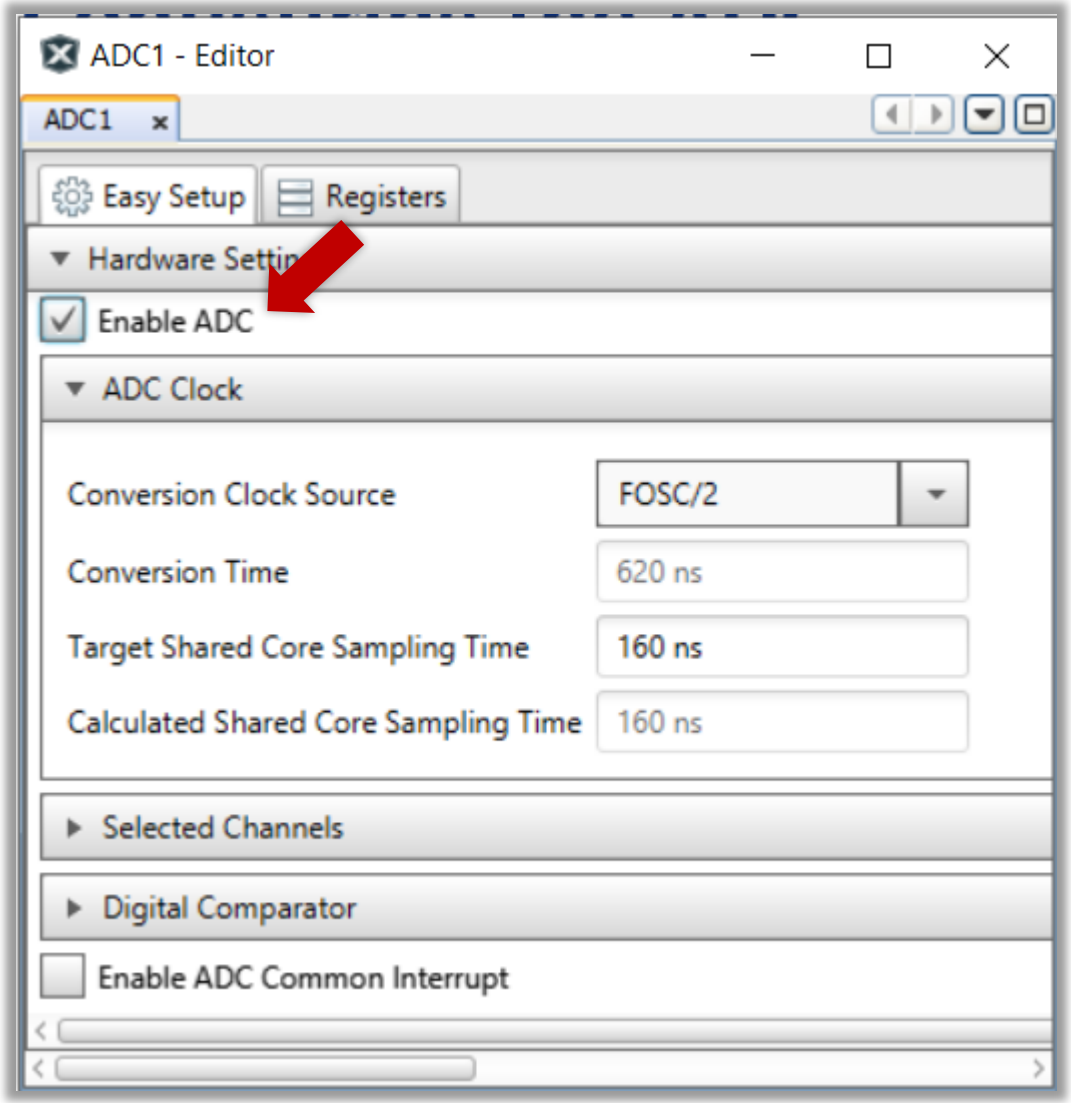
MICROCHIP

Configuring the ADC



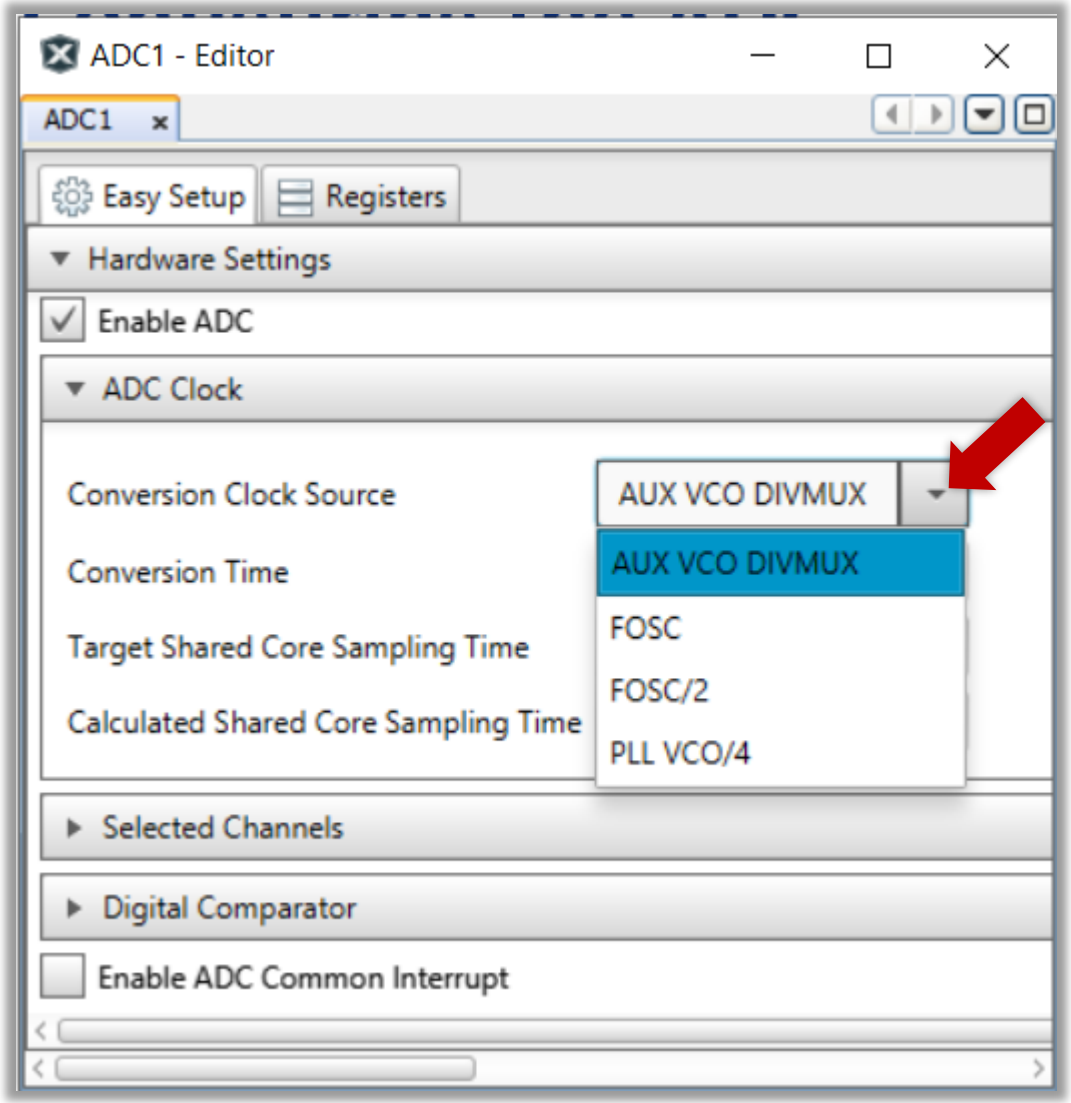
Configuring the ADC

- **Enable ADC**



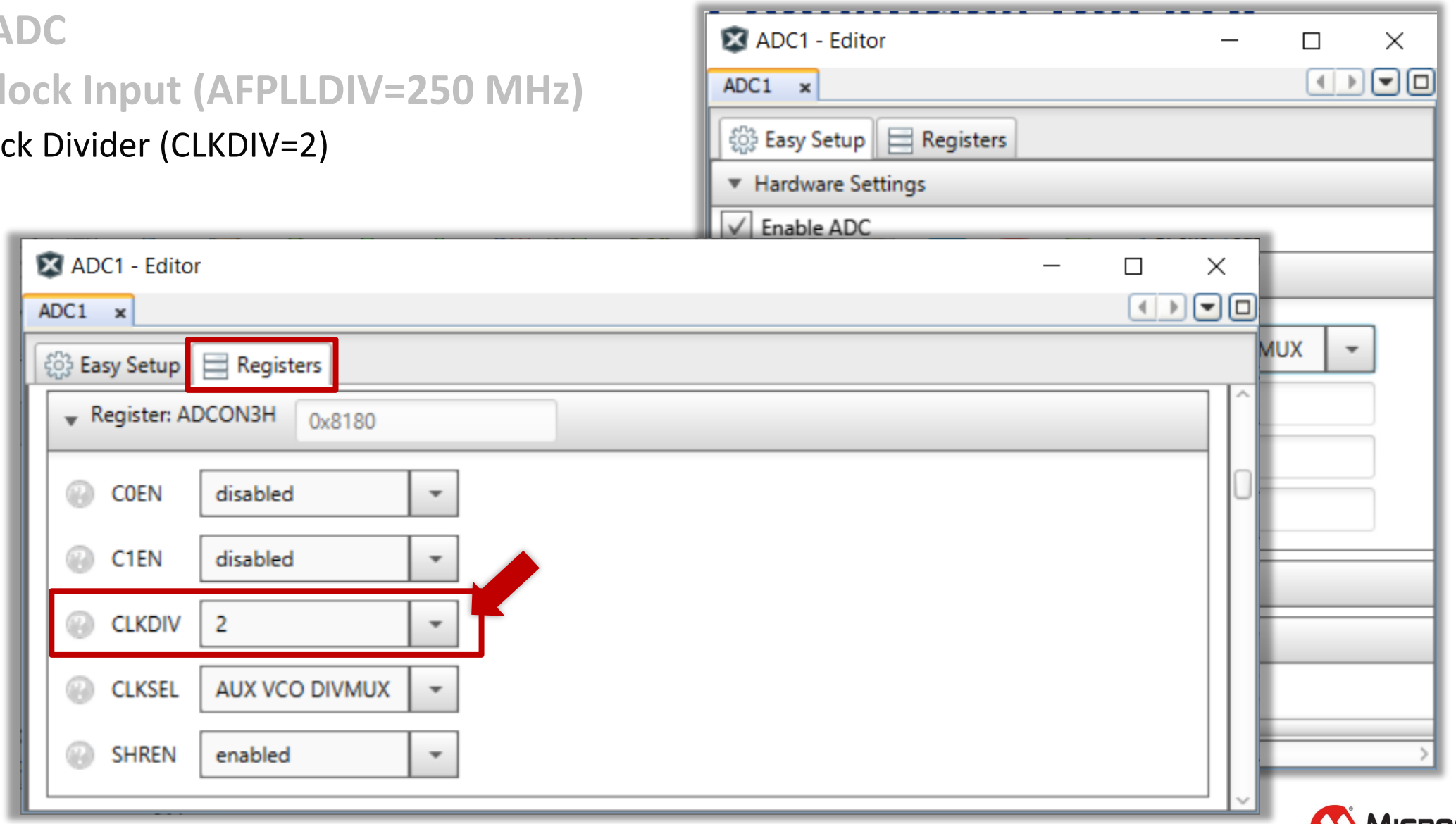
Configuring the ADC

- Enable ADC
- **Select Clock Input (AFPLLDIV=250 MHz)**



Configuring the ADC

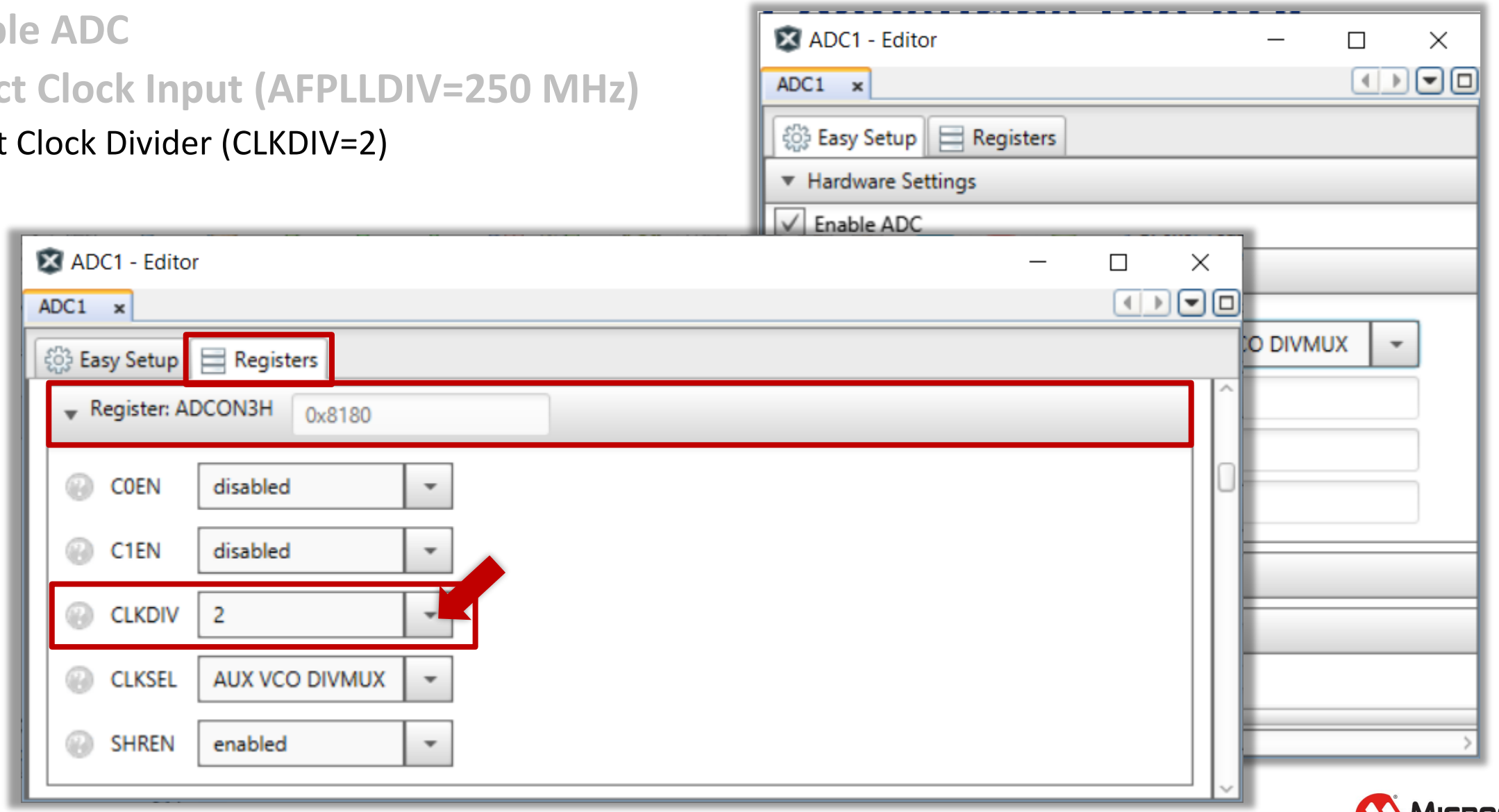
- Enable ADC
- Select Clock Input (AFPLLDIV=250 MHz)
  - Set Clock Divider (CLKDIV=2)



MICROCHIP

Configuring the ADC

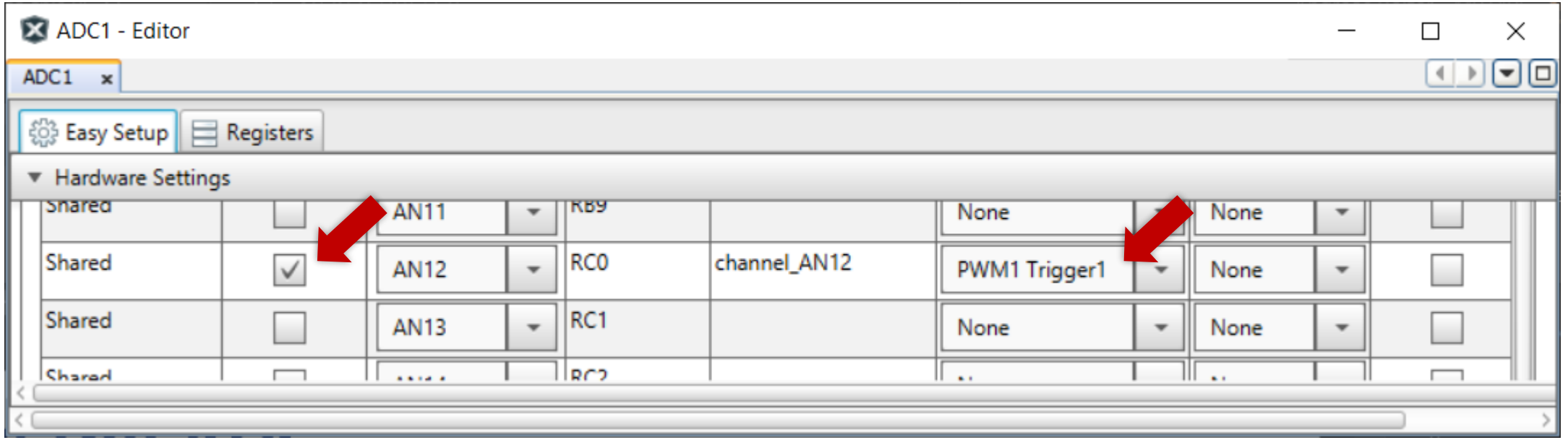
- Enable ADC
- Select Clock Input (AFPLLDIV=250 MHz)
  - Set Clock Divider (CLKDIV=2)



MICROCHIP

Configuring the ADC

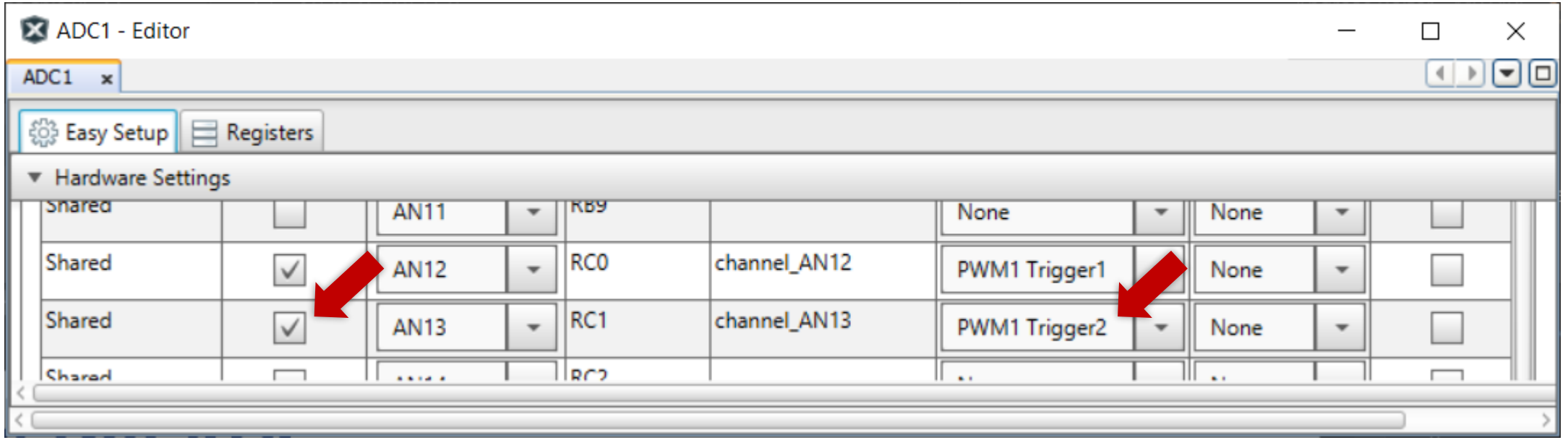
- Enable ADC
- Select Clock Input (AFPLLDIV=250 MHz)
- **Enable ADC Input AN12, Select Trigger Source (=PWM1 Trigger 1)**



MICROCHIP

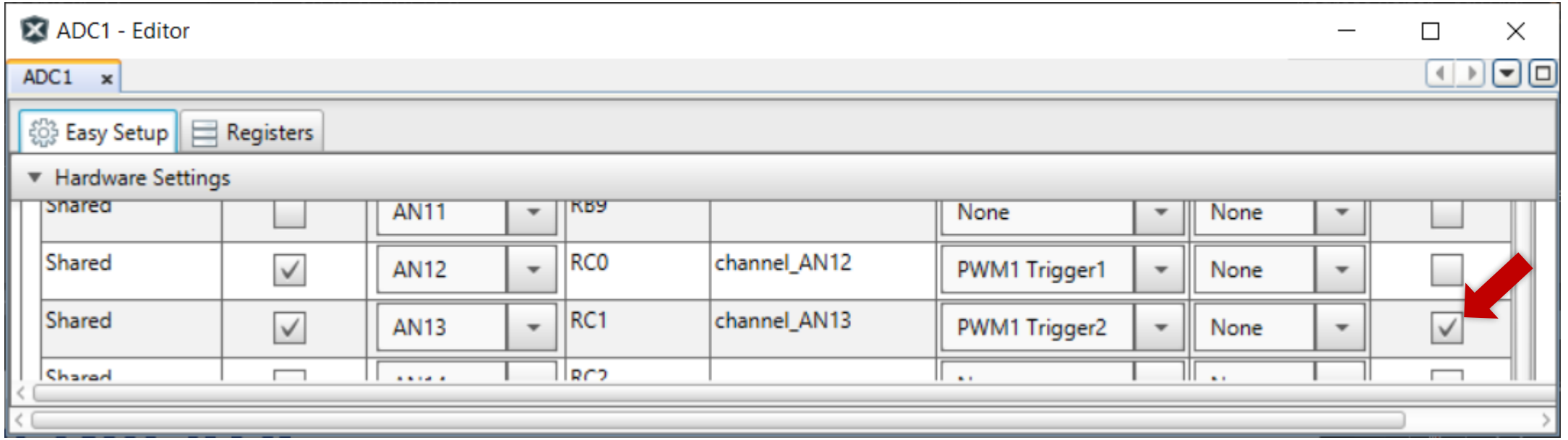
Configuring the ADC

- Enable ADC
- Select Clock Input (AFPLLDIV=250 MHz)
- Enable ADC Input AN12, Select Trigger Source (=PWM1 Trigger 1)
- **Enable ADC Input AN13, Select Trigger Source (=PWM1 Trigger 2)**

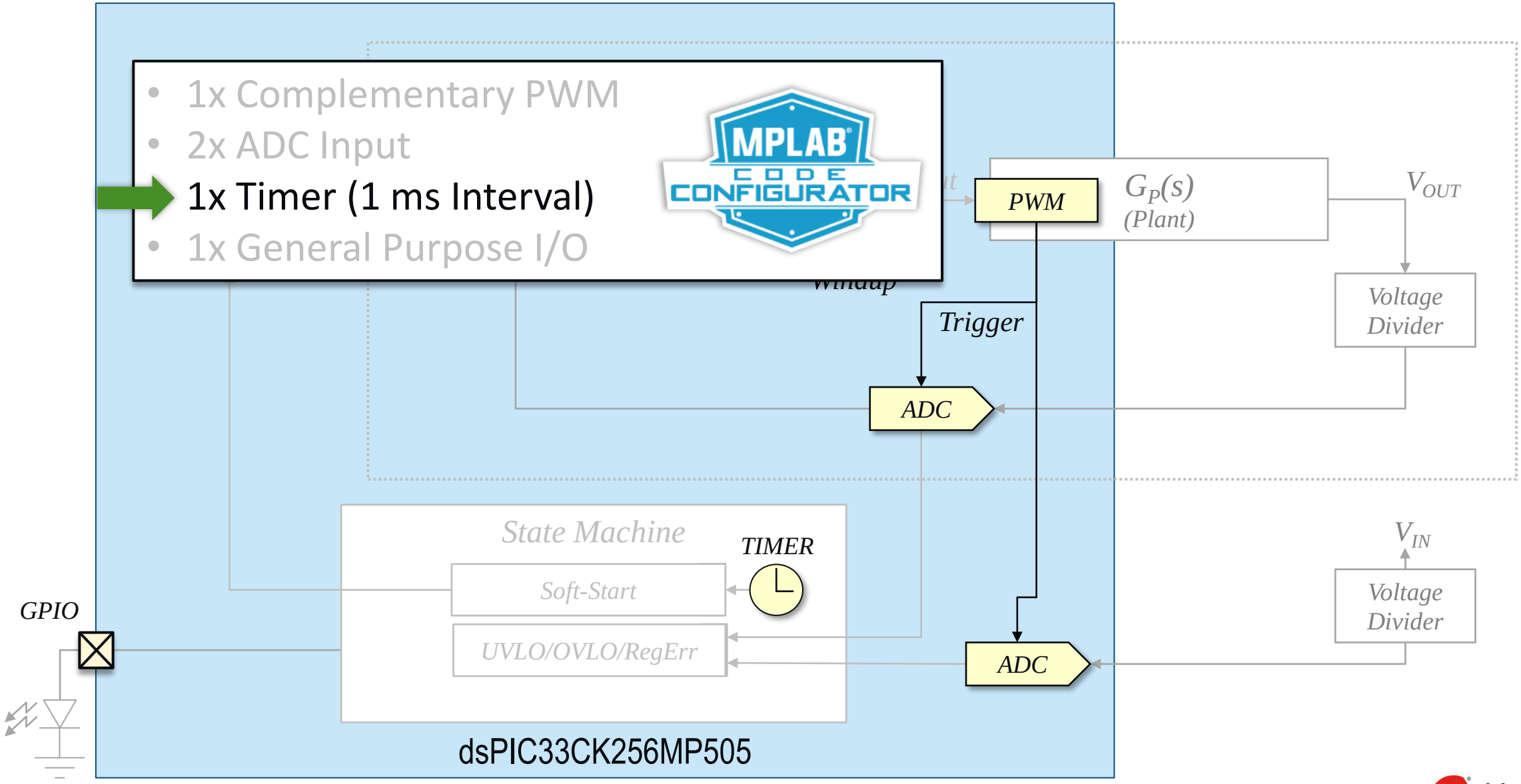


Configuring the ADC

- Enable ADC
- Select Clock Input (AFPLLDIV=250 MHz)
- Enable ADC Input AN13
- Select Trigger Source (=PWM1 Trigger 2)
- Enable ADC Input AN13 and Select Trigger Source (=PWM1 Trigger 2)
- **Enable AN13 Interrupt**

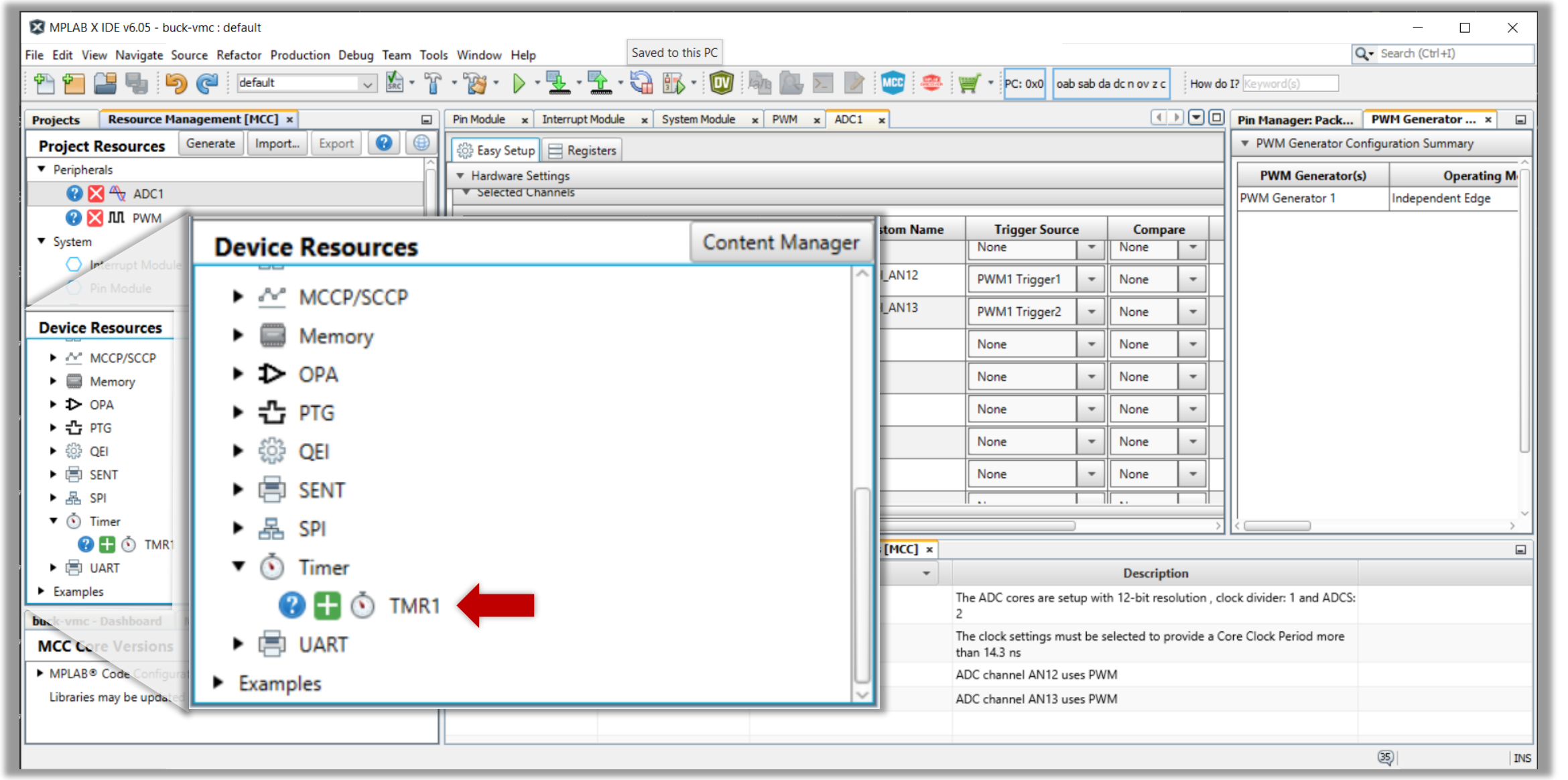


Requirements



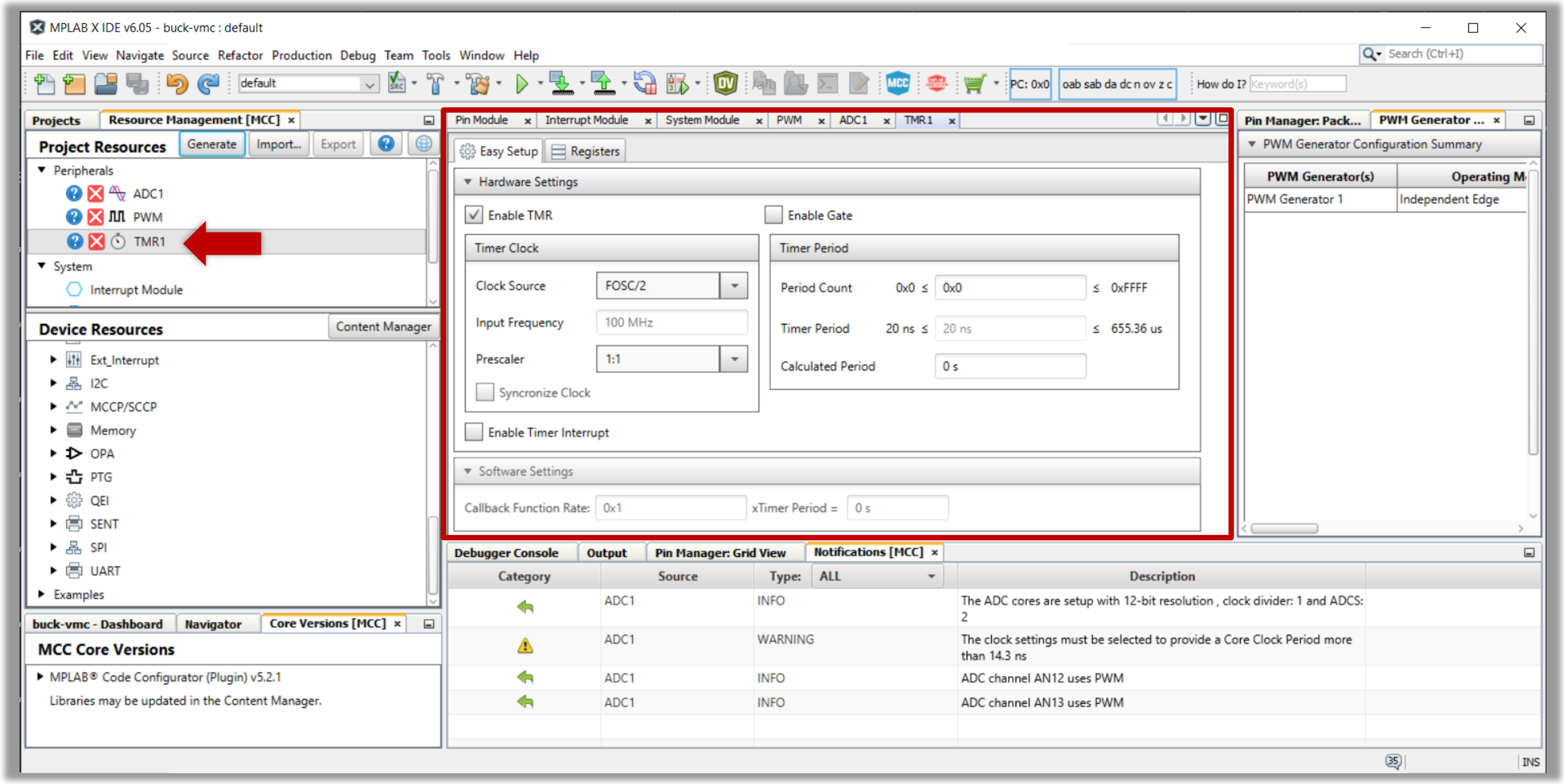
MICROCHIP

Configuring the Timer



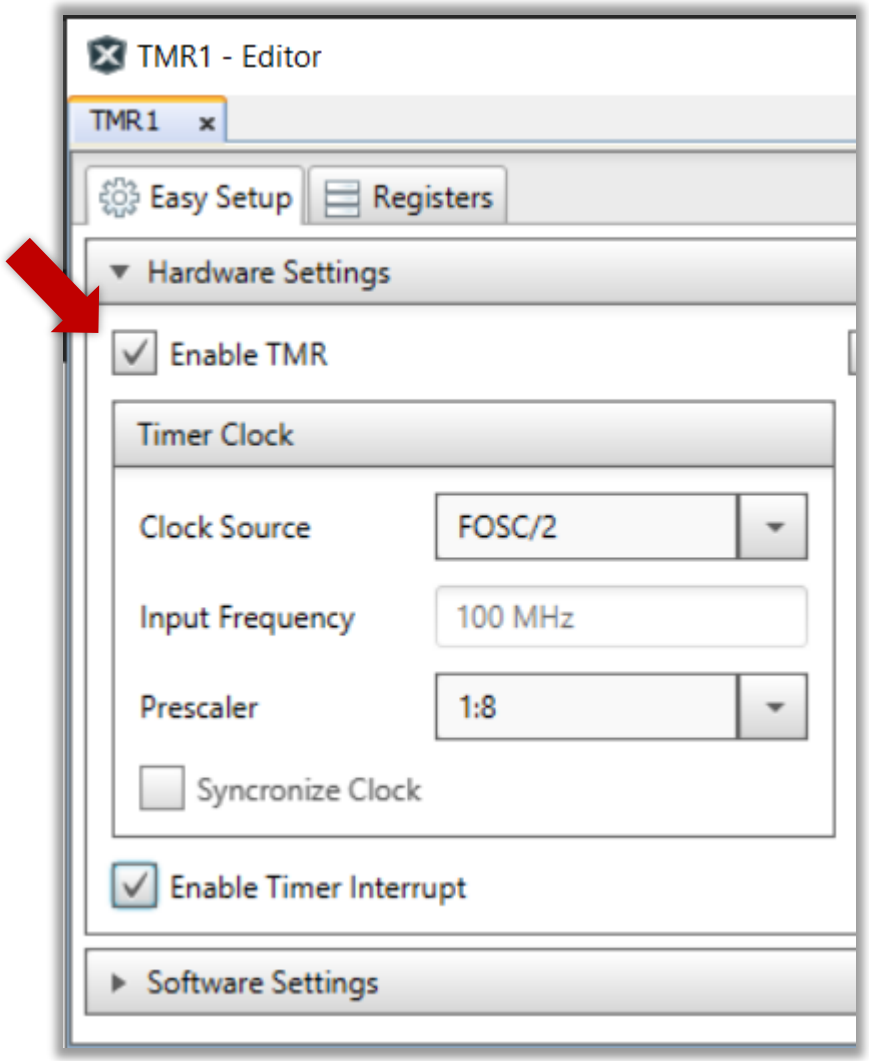
MICROCHIP

Configuring the Timer



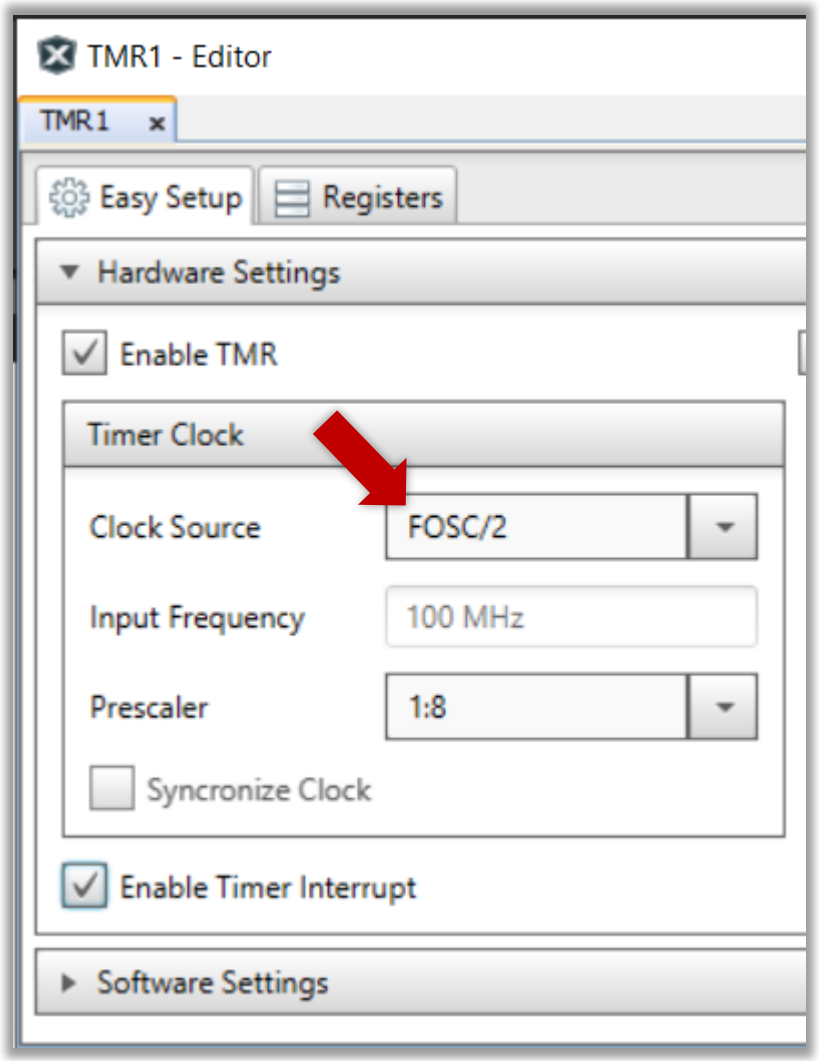
MICROCHIP

Configuring the Timer



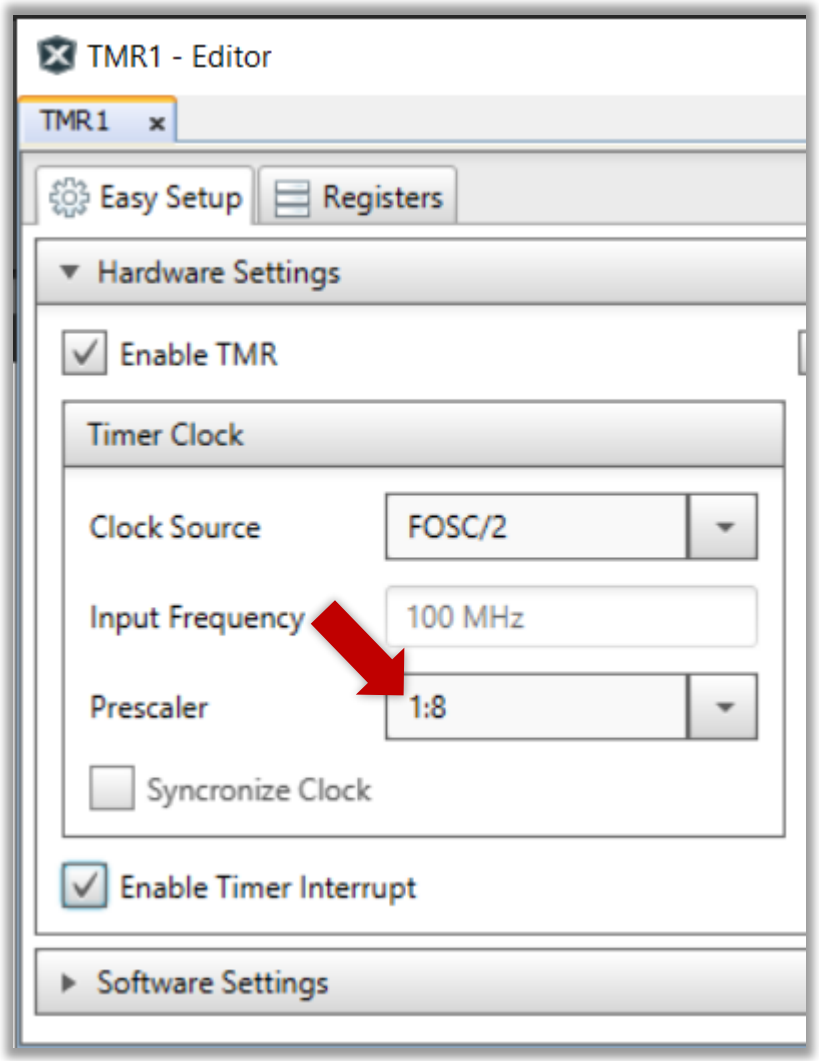
- **Enable Timer 1**

Configuring the Timer



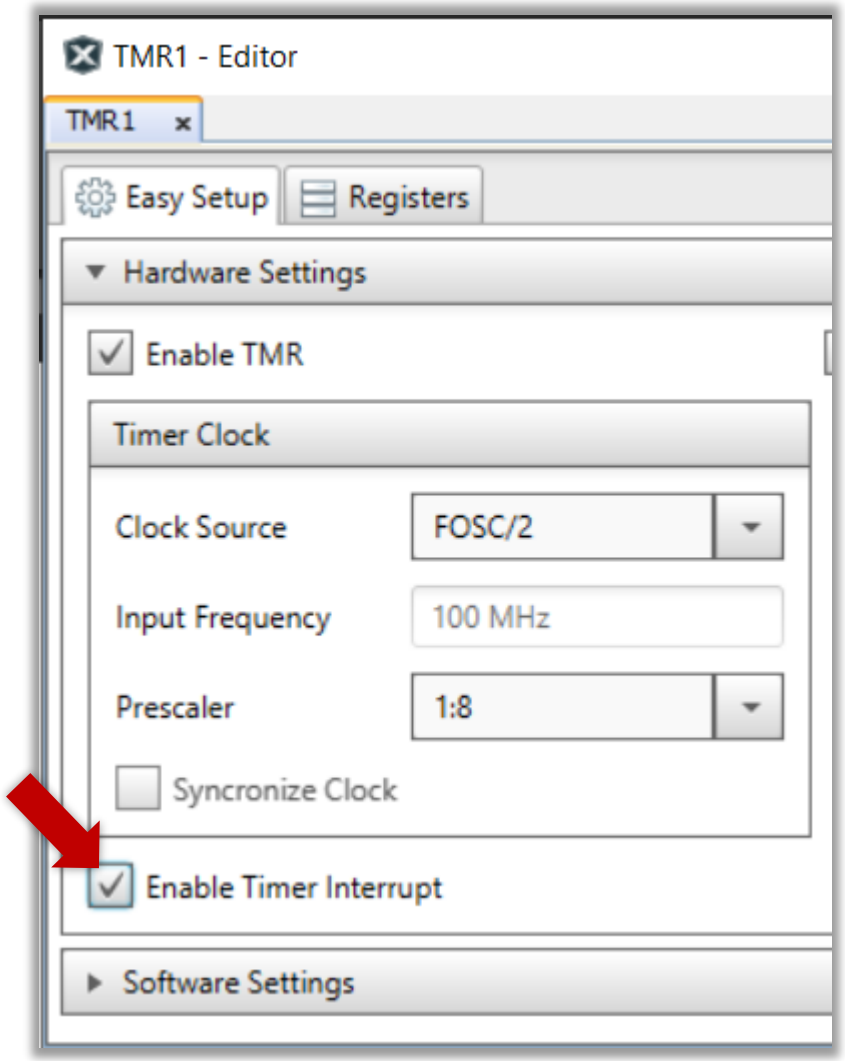
- Enable Timer 1
- **Set Clock Source to FOSC/2**

Configuring the Timer



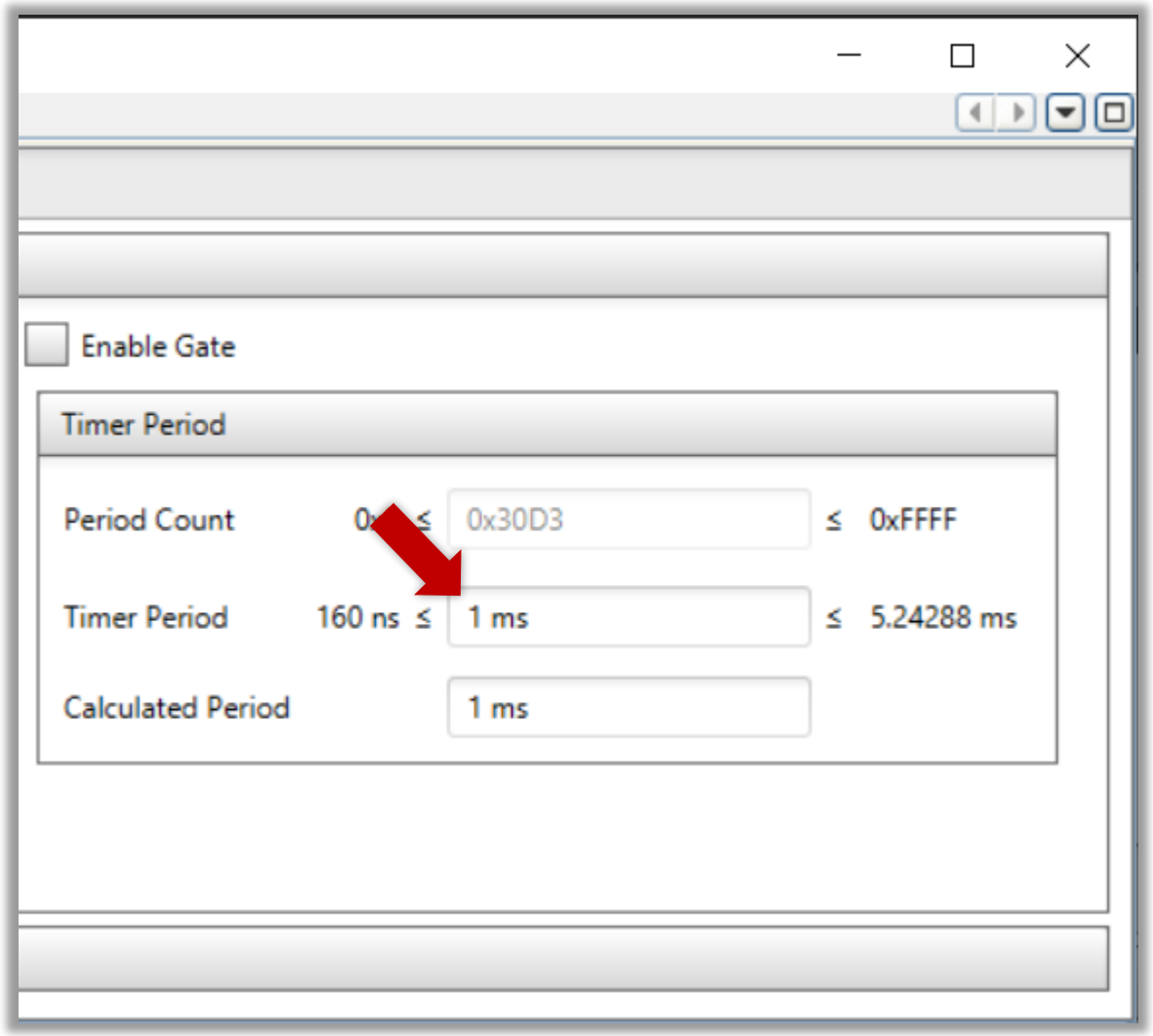
- Enable Timer 1
- Set Clock Source to FOSC/2
- **Set Prescaler (=1:8)**

Configuring the Timer



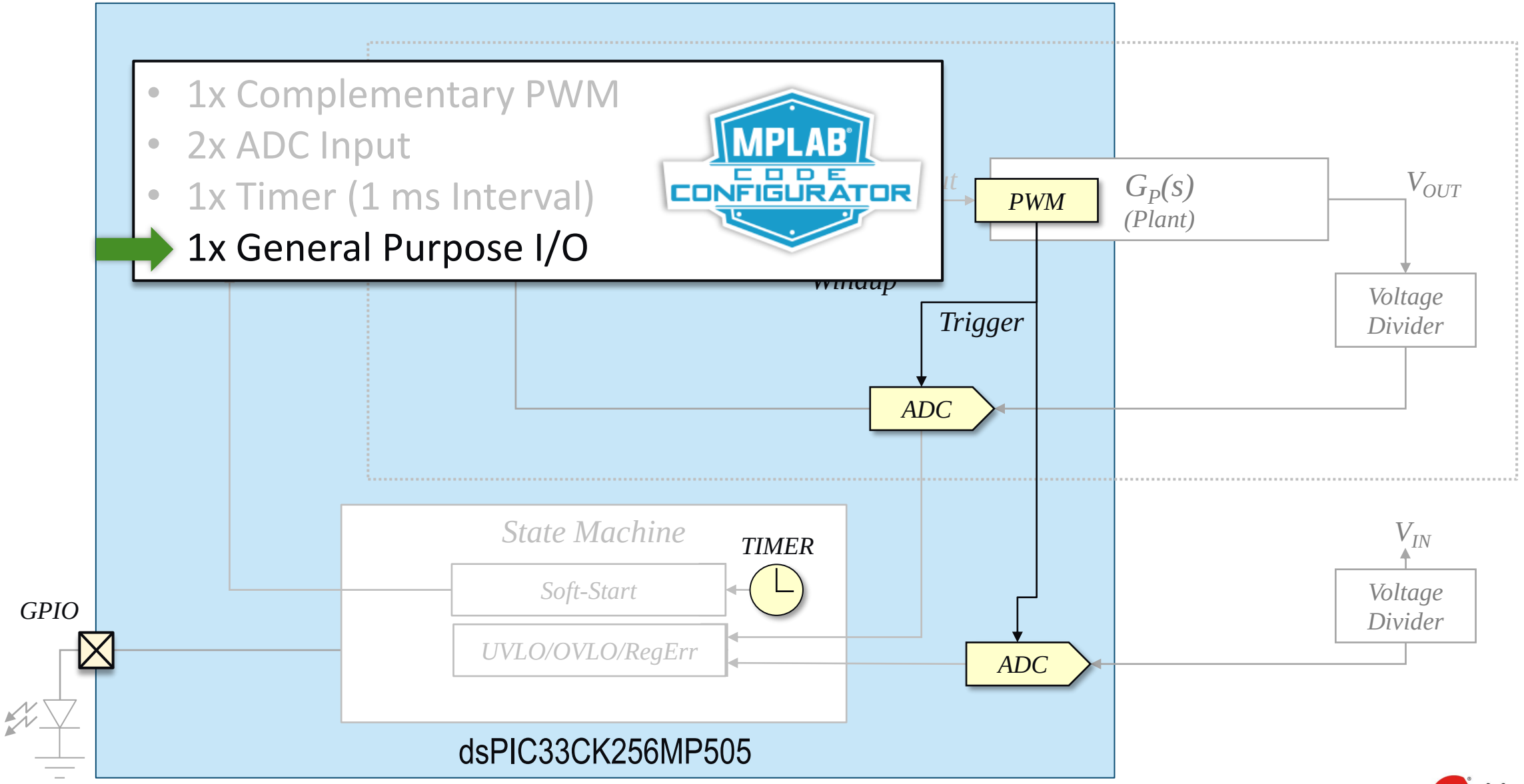
- Enable Timer 1
- Set Clock Source to FOSC/2
- Set Prescaler (=1:8)
- **Enable Timer Interrupt**

Configuring the Timer



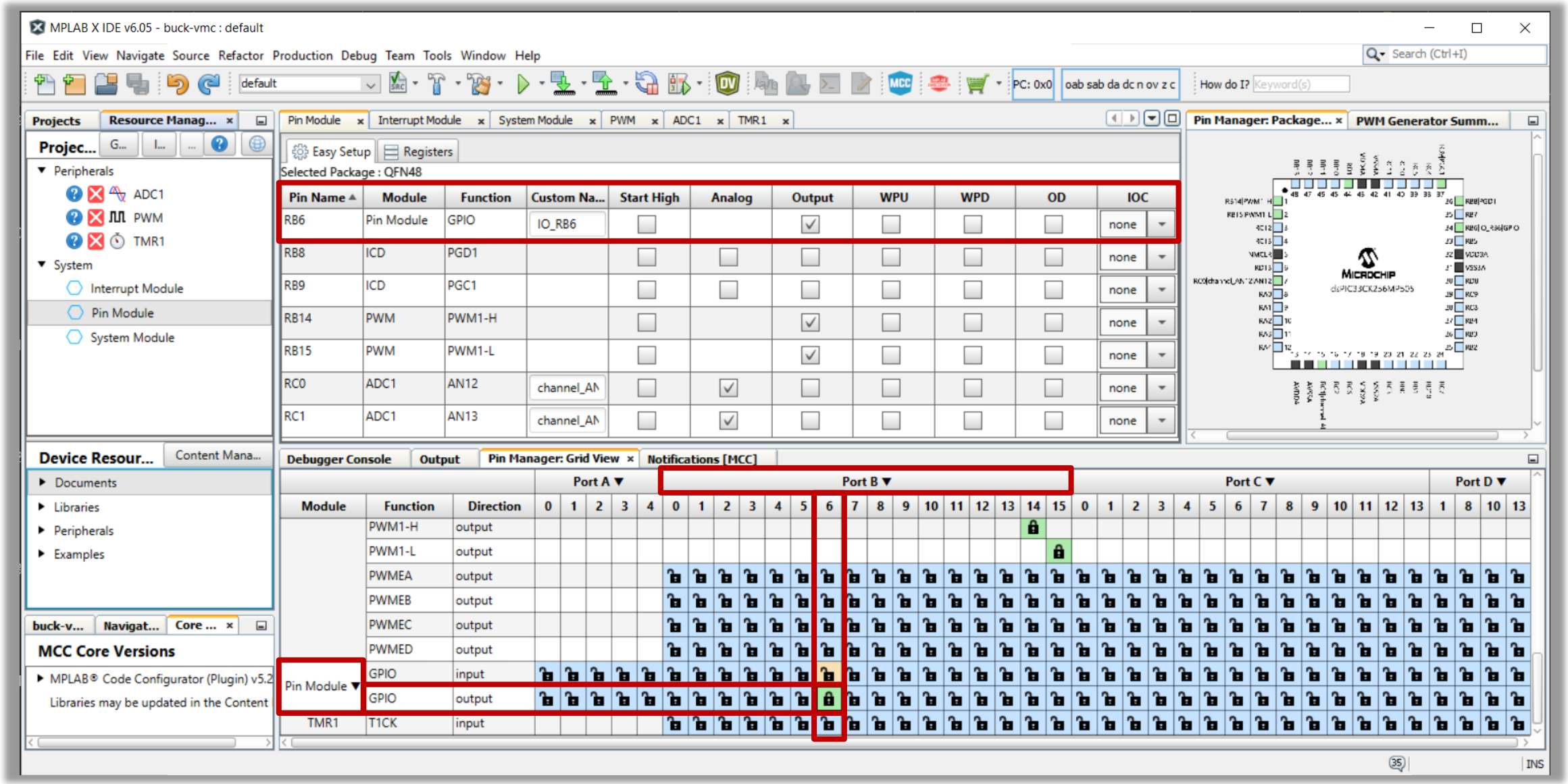
- Enable Timer 1
- Set Clock Source to FOSC/2
- Set Prescaler (=1:8)
- Enable Timer Interrupt
- **Set Timer Period (=1 ms)**

Requirements



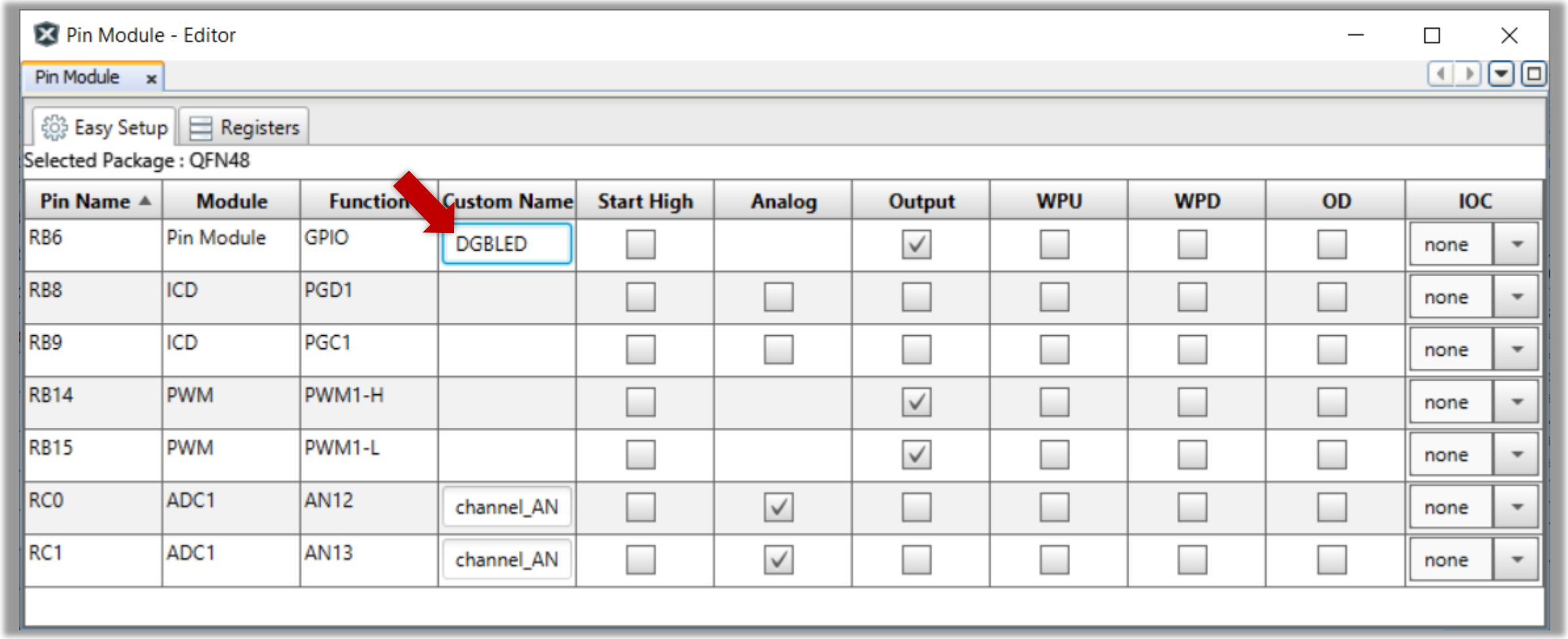
MICROCHIP

Configuring the General Purpose I/O



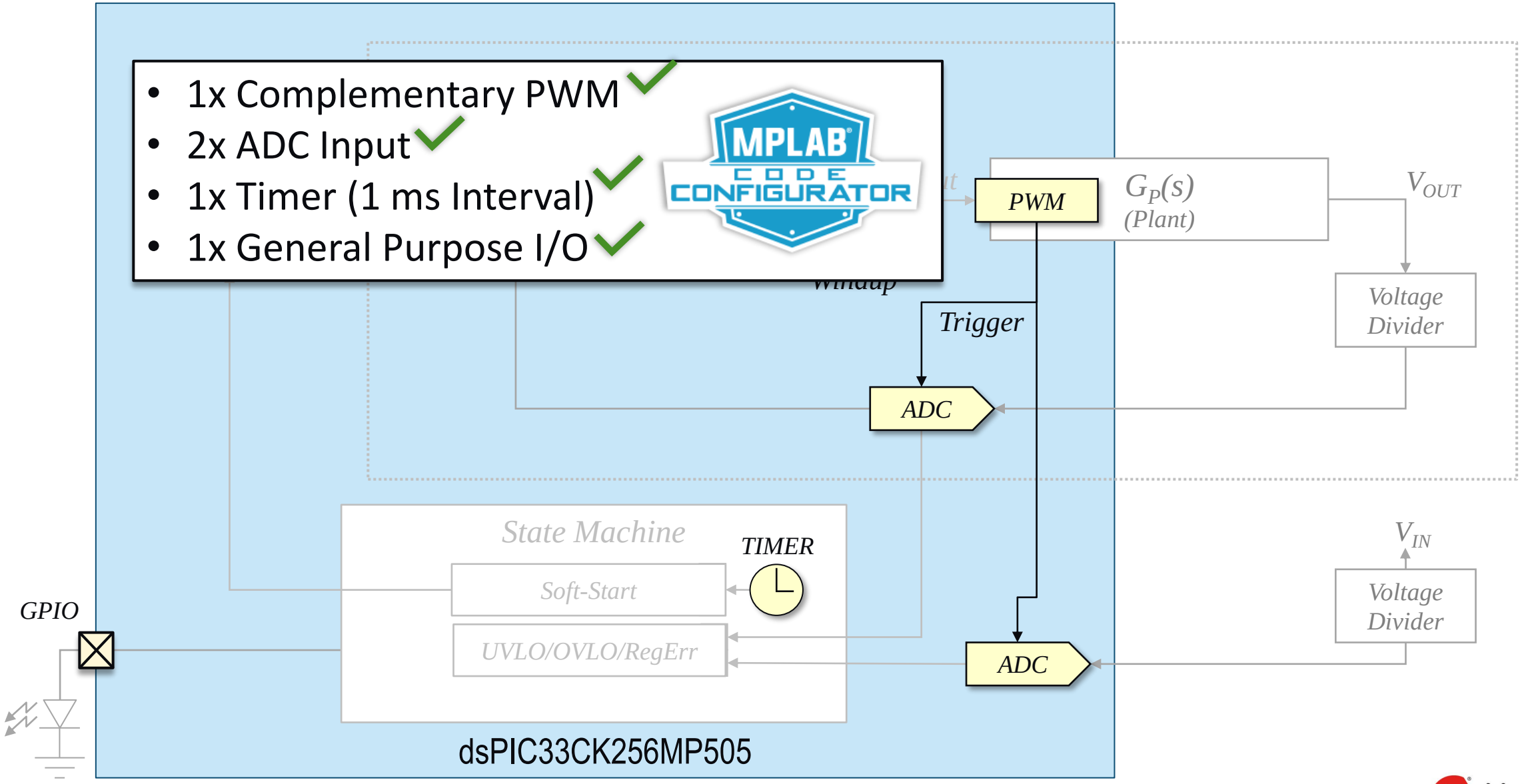
MICROCHIP

Configuring the General Purpose I/O



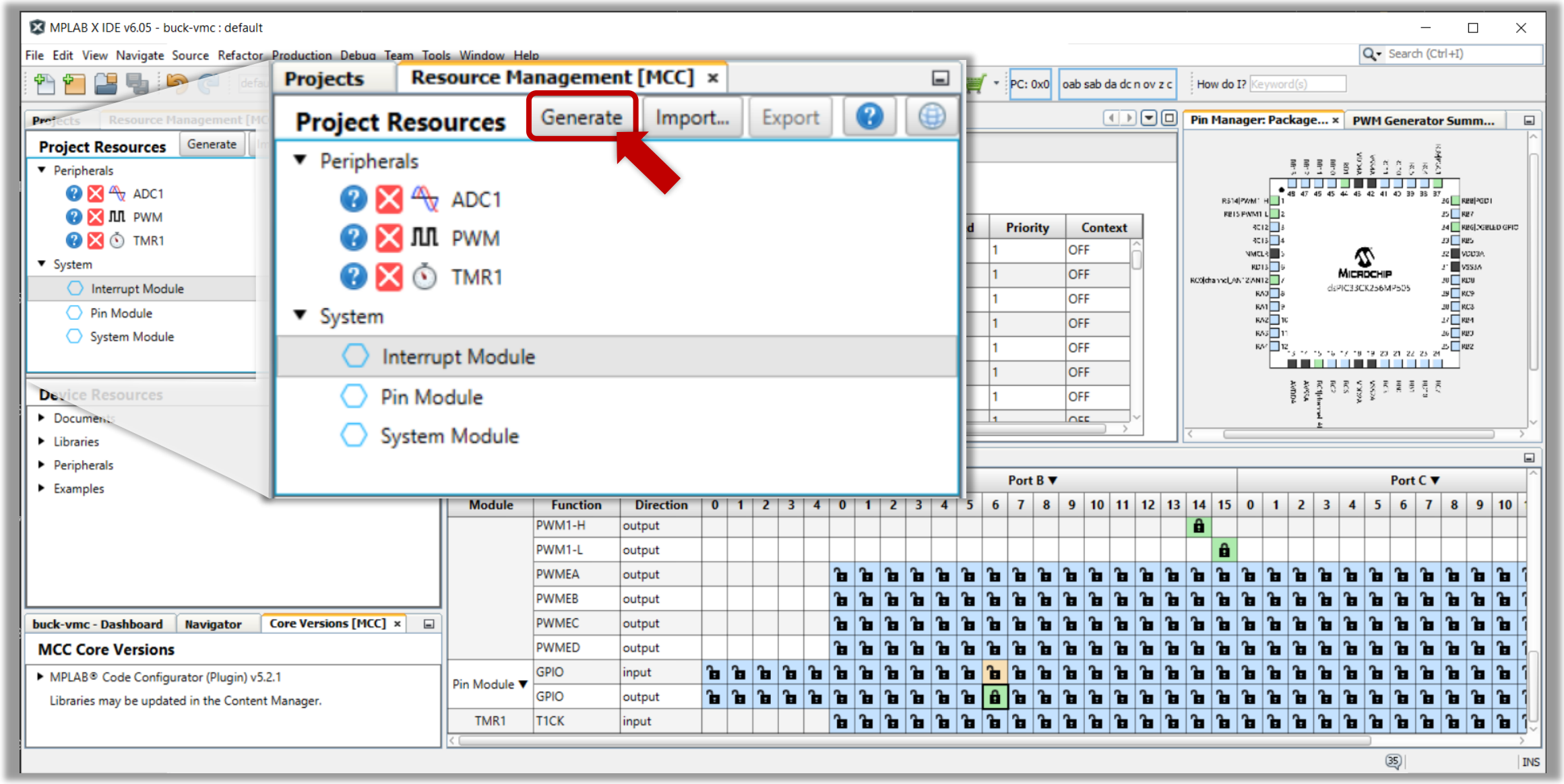
Pin Name	Module	Function	Custom Name	Start High	Analog	Output	WPU	WPD	OD	IOC
RB6	Pin Module	GPIO	DGBLED	☐		☑	☐	☐	☐	none
RB8	ICD	PGD1		☐	☐	☐	☐	☐	☐	none
RB9	ICD	PGC1		☐	☐	☐	☐	☐	☐	none
RB14	PWM	PWM1-H		☐		☑	☐	☐	☐	none
RB15	PWM	PWM1-L		☐		☑	☐	☐	☐	none
RC0	ADC1	AN12	channel_AN	☐	☑	☐	☐	☐	☐	none
RC1	ADC1	AN13	channel_AN	☐	☑	☐	☐	☐	☐	none

Requirements

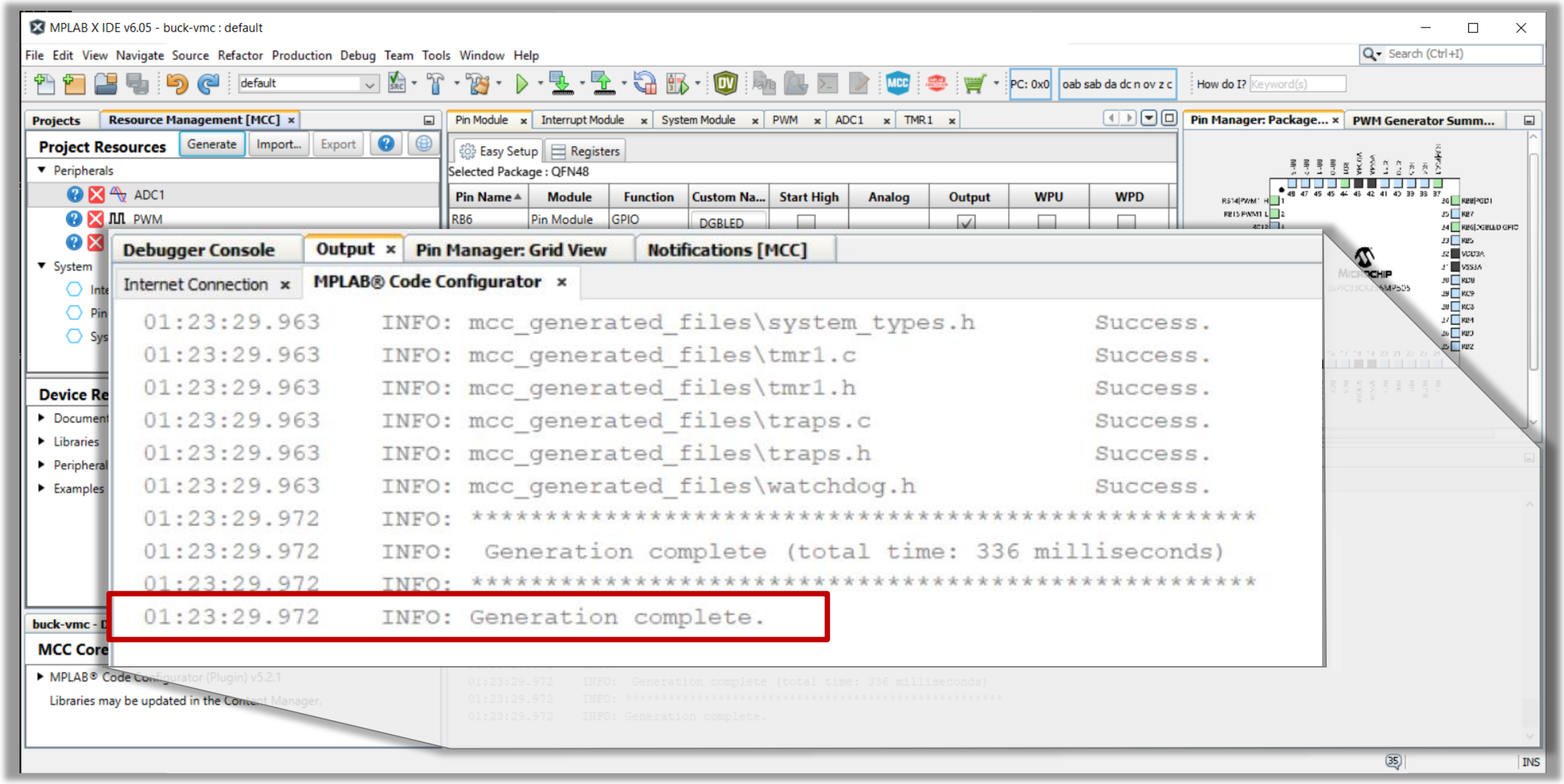


MICROCHIP

Time to Generate Code!

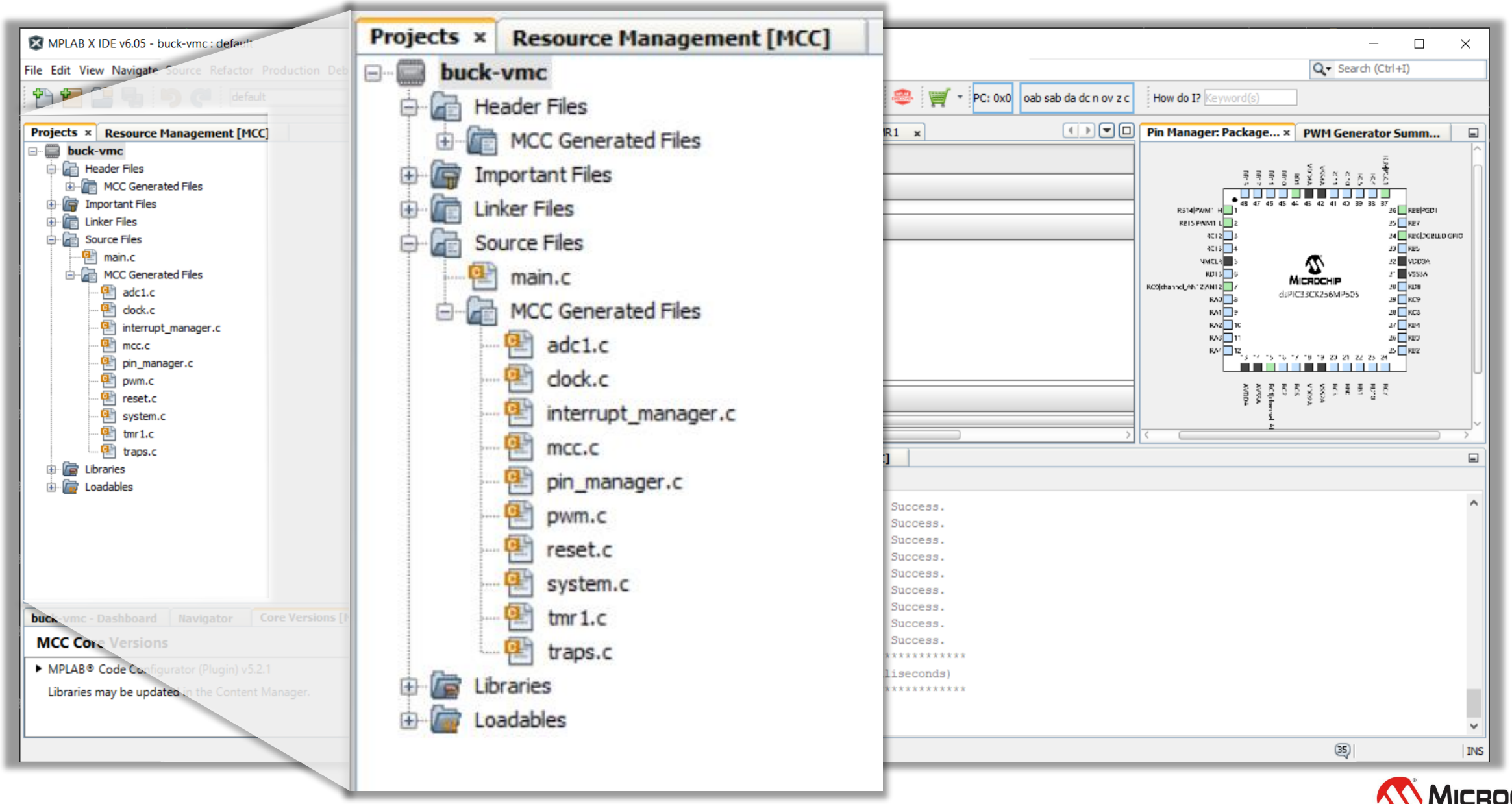


Time to Generate Code!



MICROCHIP

Time to Generate Code!



Including MCC header files in main.c

```
/**
  Section: Included Files
*/
#include "mcc_generated_files/system.h"
#include "mcc_generated_files/pin_manager.h"
#include "mcc_generated_files/pwm.h"
#include "mcc_generated_files/adc1.h"
#include "mcc_generated_files/tmr1.h"

[…]

int main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    while (1)
    {
        // Add your application code
    }
    return 1;
}
```

Declare User Variables in main.c

```
/**
  Section: Included Files
*/
#include "mcc_generated_files/system.h"
#include "mcc_generated_files/pin_manager.h"
#include "mcc_generated_files/pwm.h"
#include "mcc_generated_files/adc1.h"
#include "mcc_generated_files/tmr1.h"

// Define Global Variables:
int adc_vin = 0;    // Scaling: 155.1 steps/V
int adc_vbuck = 0;  // Scaling: 620.5 steps/V
int ref_vbuck = 0;  // Scaling: 620.5 steps/V
[…]
int main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    while (1)
    {
        // Add your application code
    }
    return 1;
}
```

Add Timer Interrupt Service Routine to main.c

```
[…]

void TMR1_Int(void)
{
    static int led_tmr = 0;

    // blink LED @ 1Hz
    if(++led_tmr>=500) {
        led_tmr = 0;
        DGBLED_Toggle();
    }
}

int main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    while (1)
    {
        // Add your application code
    }
    return 1;
}
```

Add Timer Interrupt Service Routine to main.c

```
[…]
    if(++led_tmr>=500) {
        led_tmr = 0;
        DGBLED_Toggle();
    }
}

void AN13_Int(void)
{
    adc_vin = ADCBUF12;
    adc_vbuck = ADCBUF13;
}

int main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    while (1)
    {
        // Add your application code
    }
    return 1;
}
```

Set Interrupt Handlers in main() in main.c

```
[…]

void AN13_Int(void)
{
    adc_vin = ADCBUF12;
    adc_vbuck = ADCBUF13;
}

int main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    ADC1_Setchannel_AN13InterruptHandler(&AN13_Int);
    TMR1_SetInterruptHandler(&TMR1_Int);
    while (1)
    {
        // Add your application code
    }
    return 1;
}
```


Testing Time !

Give it a try…

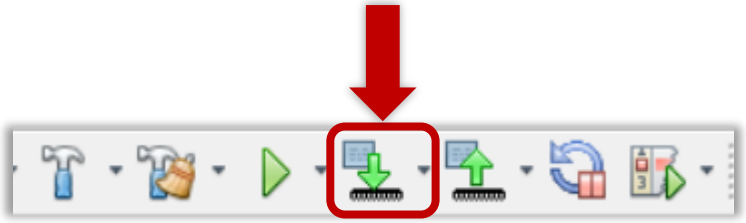
- **Build the Project**

  - If the Build was successful …
  - Or check error message and fix what's broken

- **Connect the DPSK3 via USB to your PC**

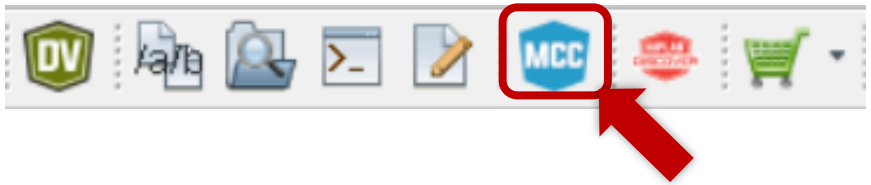
DO NOT ATTACH POWER YET !!!

- **Program the Project**

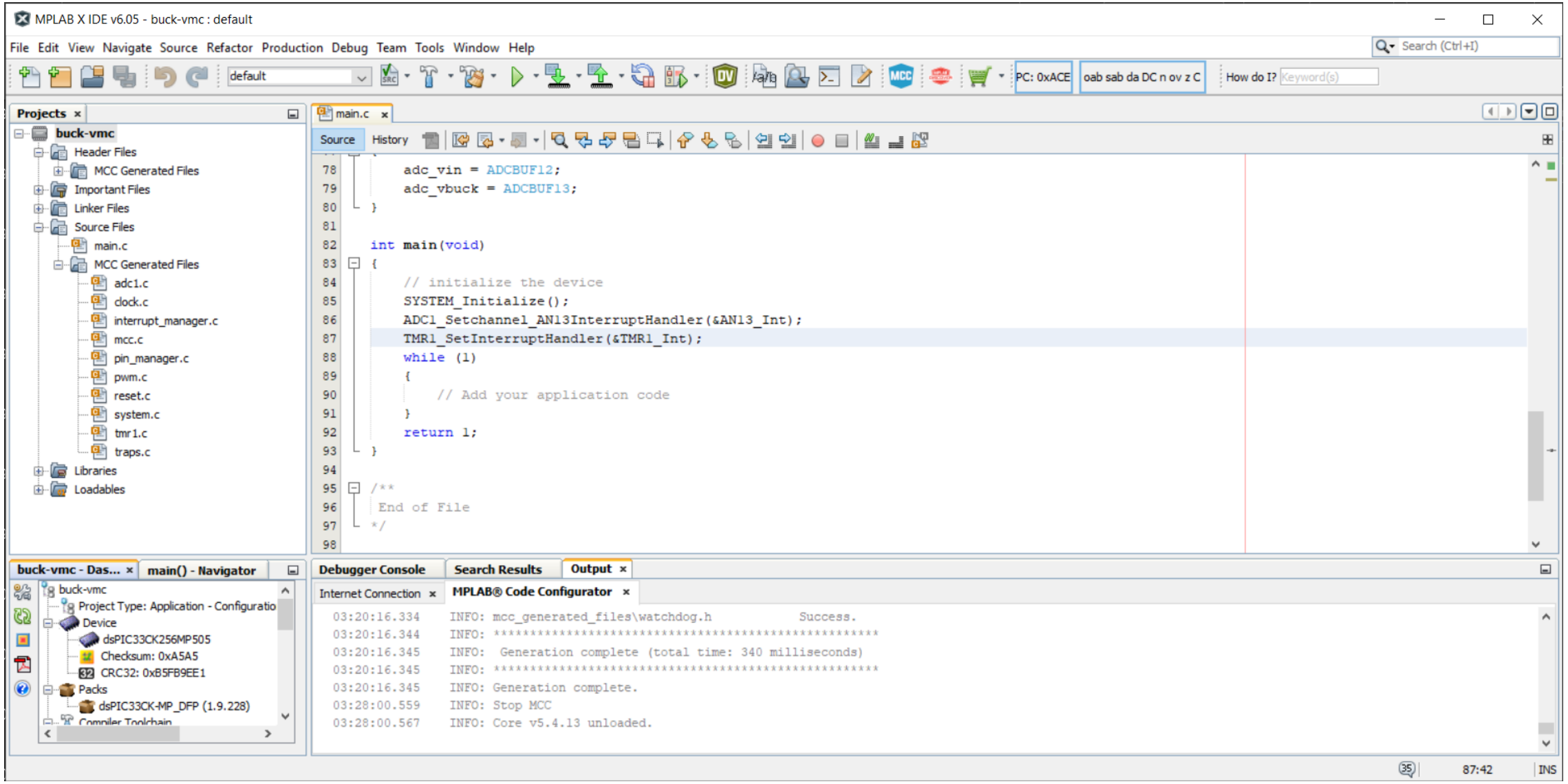


  - When programming was successful, the red LED below dsPIC should blink

- **Close MCC**



Now we can continue coding & testing …



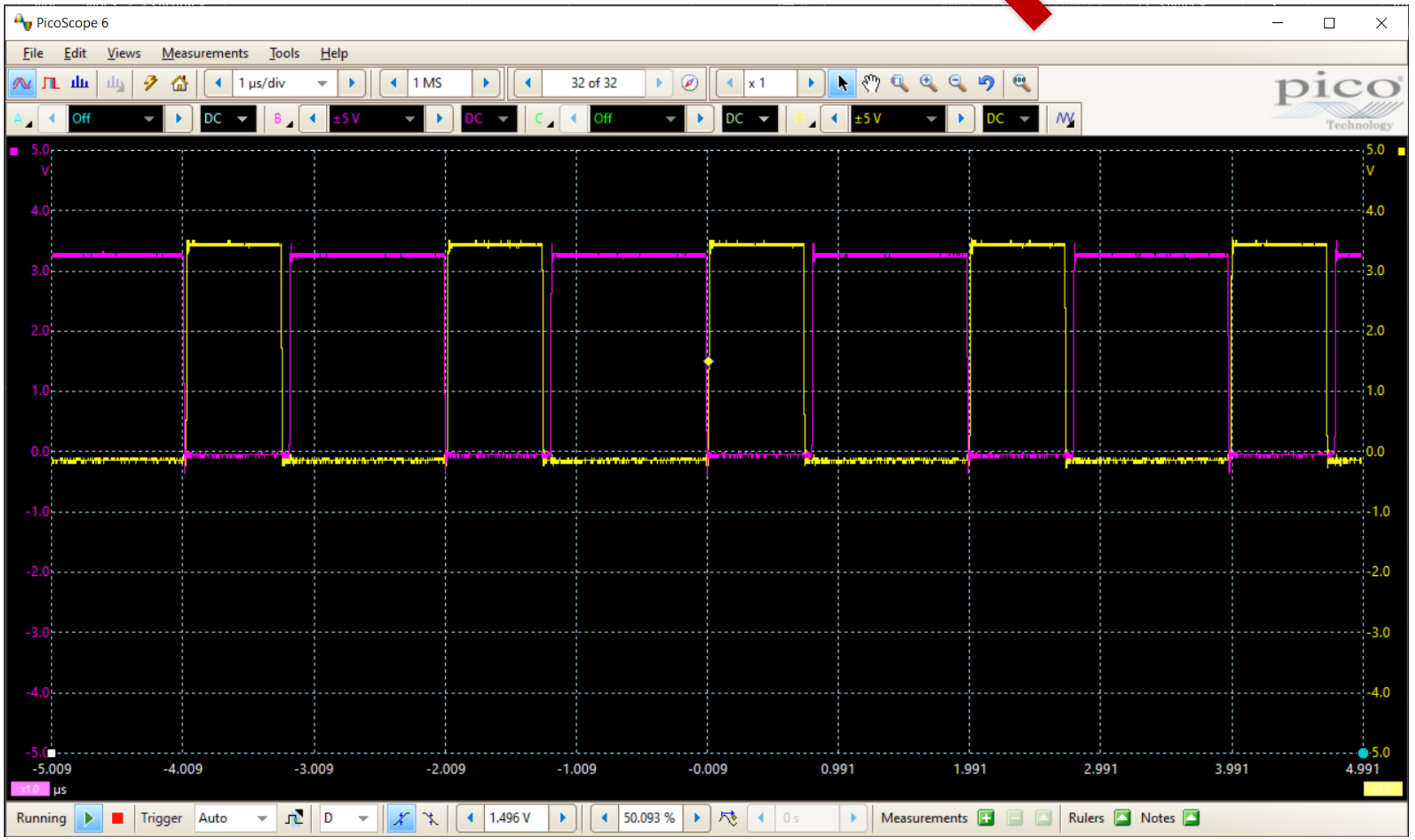
MICROCHIP

Next up: PWM Output Signals

```
void TMR1_Int(void)
{
    static int led_tmr = 0;

    // blink LED @ 1Hz
    if(++led_tmr>=500) {
        led_tmr = 0;
        DGBLED_Toggle();
    }

    // State Machine
    if (PG1DC < 3000) PG1DC++;

}
```

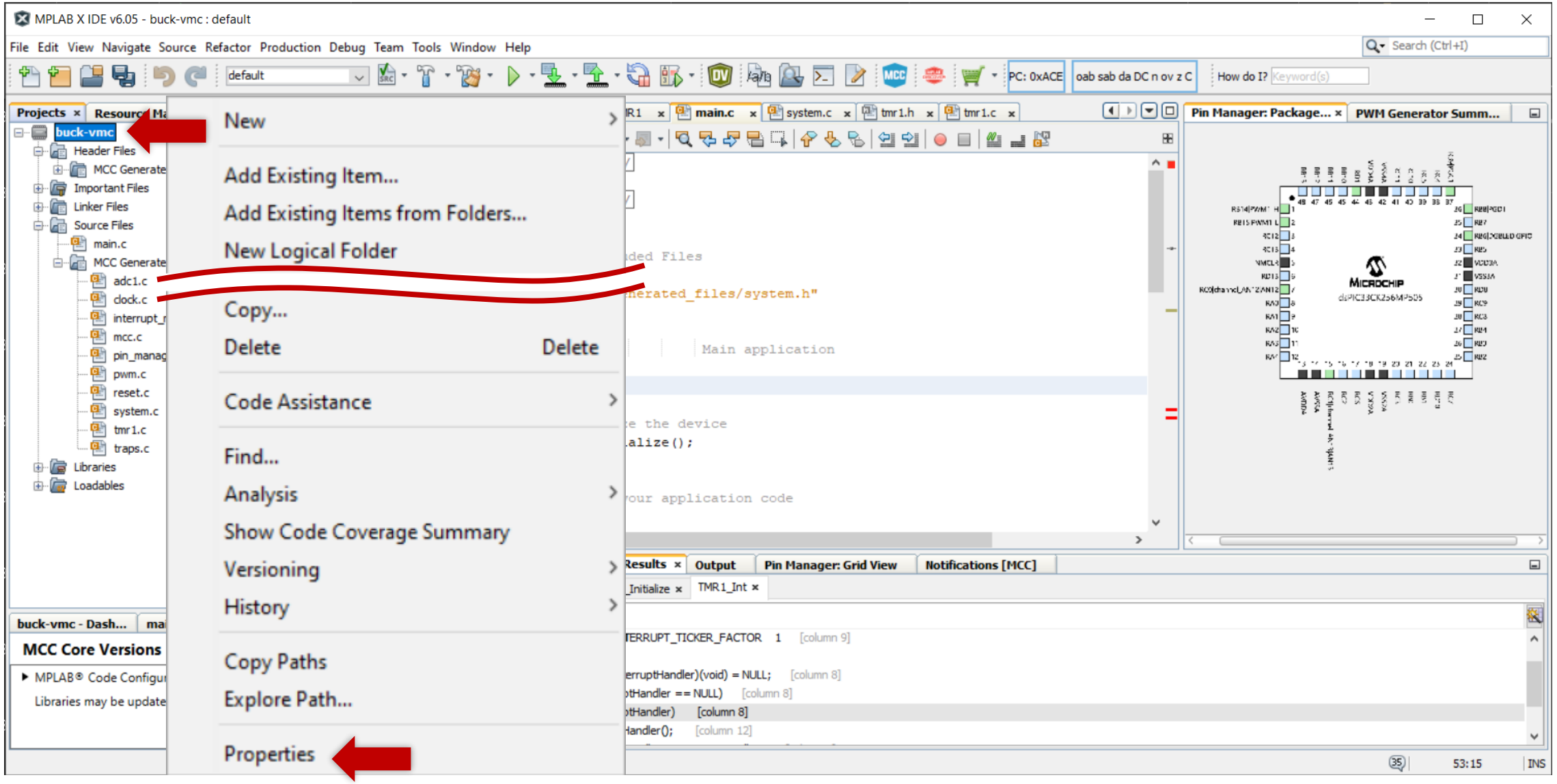
Add Open Loop Soft-Start code to prevent start of PWM from triggering protection circuit

Program Target Device

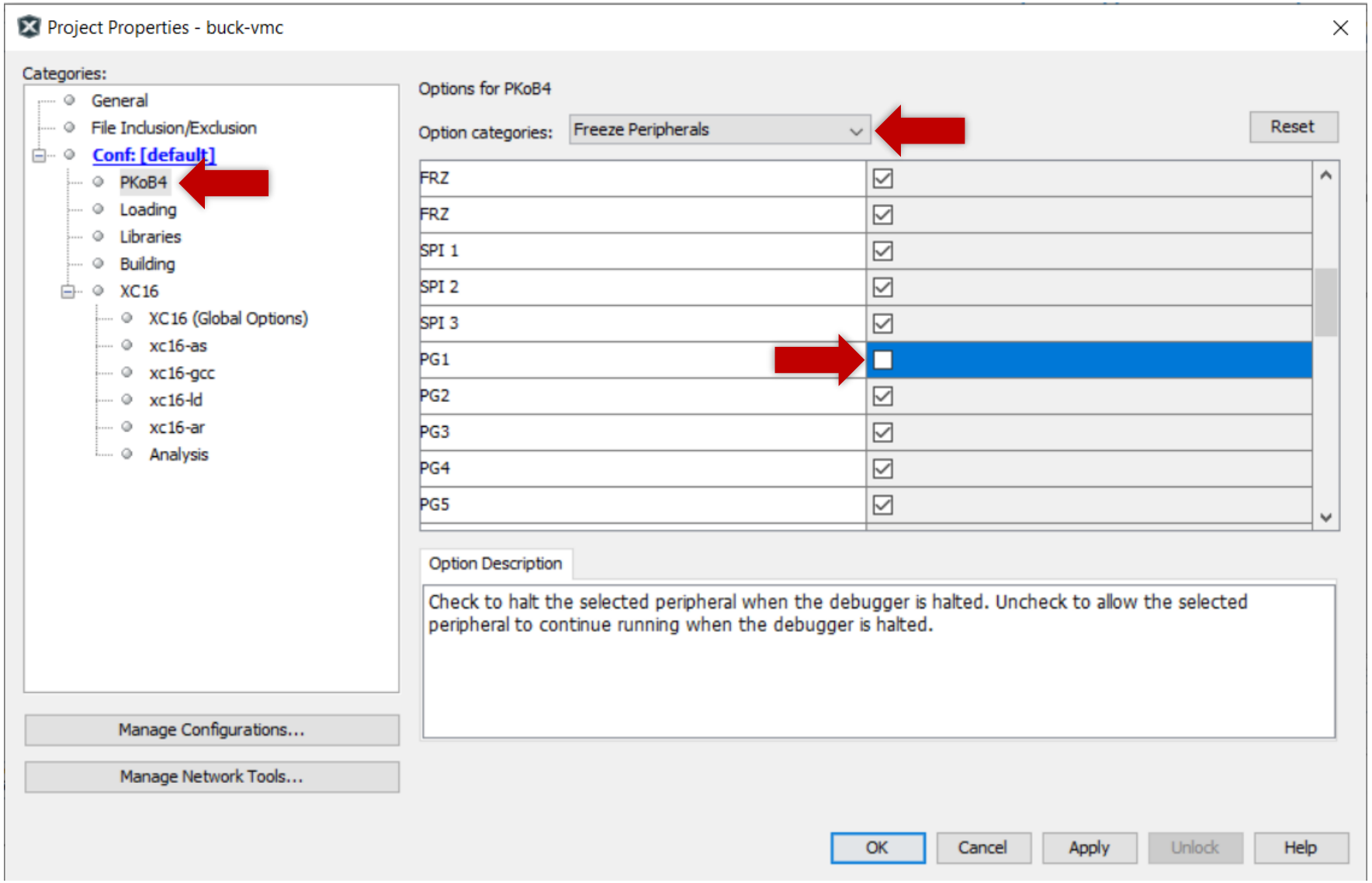


MICROCHIP

“Unfreeze” PWM Generator #1



“Unfreeze” PWM Generator #1

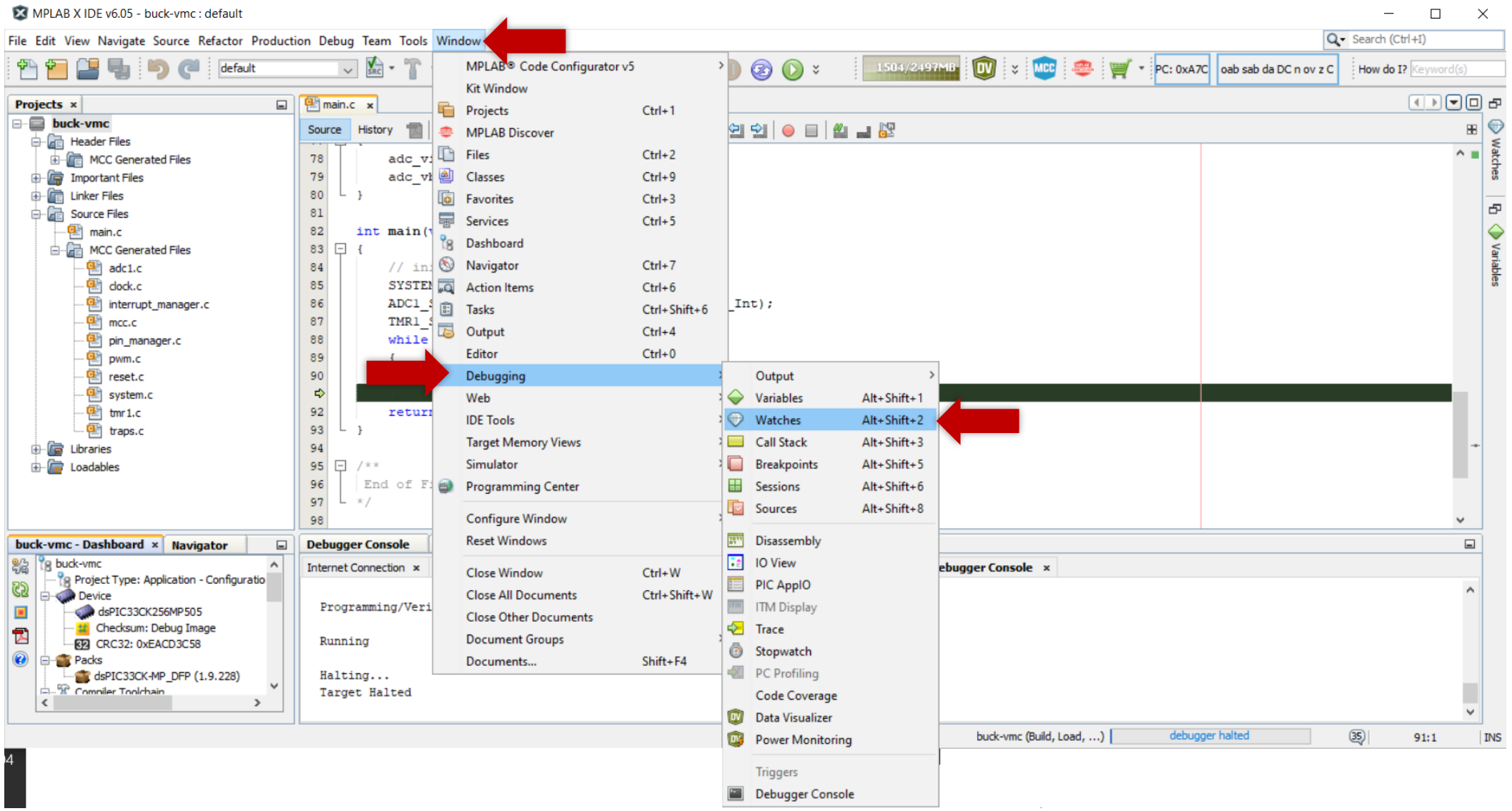


Start a Debug Session for ADC Verification

- **Start a Debug Session**

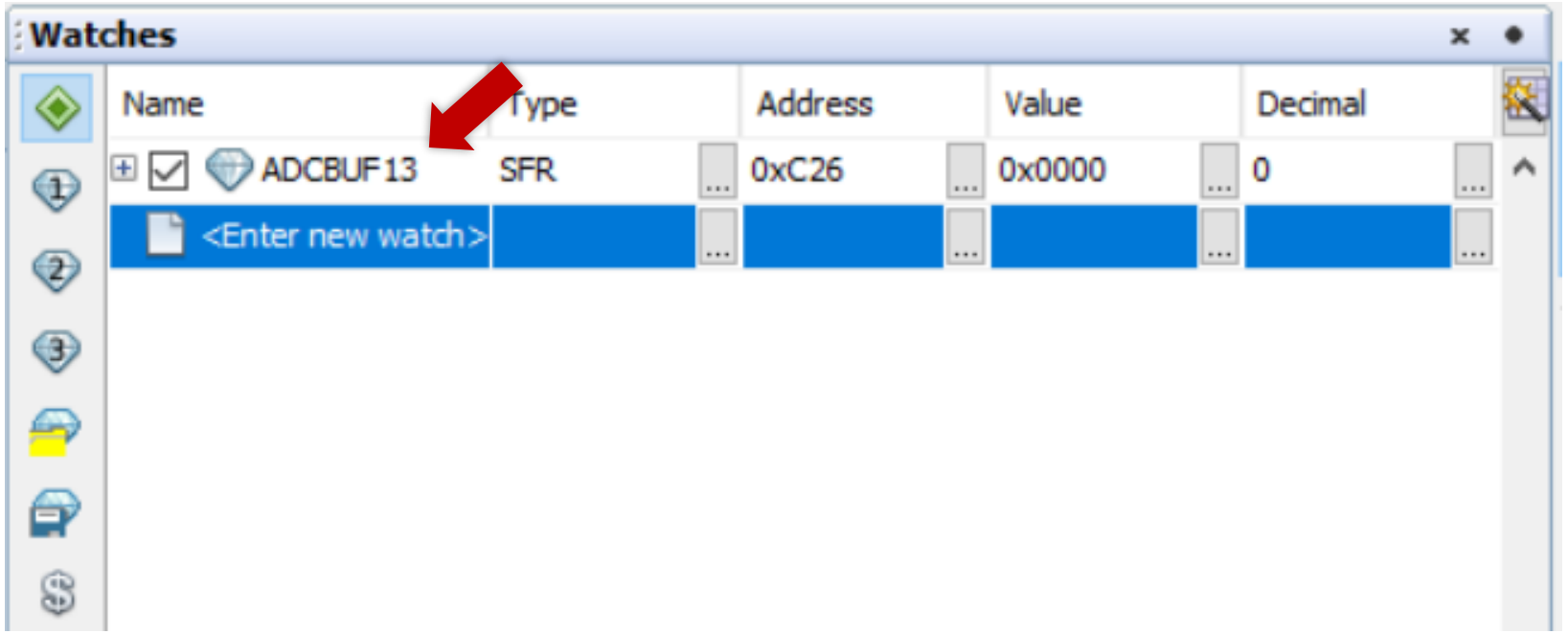
  - Target device is being programmed
  - Software stops at start of main()
  - If not, click Pause and Reset

Open the Watch-Window



Analyzing Variables at Runtime

- Add register name ADCBUF13 to Watches



- Connect 9V Power Supply to DPSK3
- Release target device

MICROCHIP

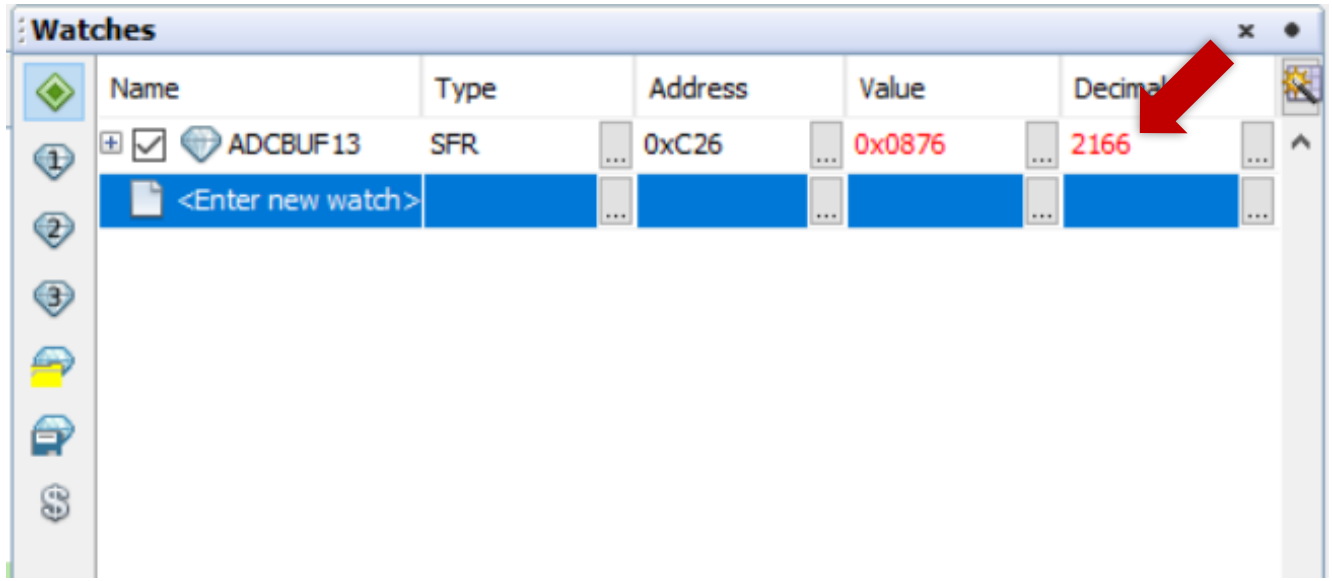
Analyzing Variables at Runtime

- **Connect 9V Power Supply to DPSK3**
- **Release target device**

- **Let it run for some seconds…**
- **Click Pause to halt the code execution**

Analyzing Variables at Runtime

- Check ADCBUF13 value
- Value = 2156 ticks



$$V_{FB} = V_{REF} \times \frac{ADCBUF13}{4095} = 3.3\,V \times \frac{2156}{4095} = 1.737\,V$$

$$V_{OUT} = V_{VB} \times \frac{R_1 + R_2}{R_2} = 1.737\,V \times \frac{1\,k\Omega + 1\,k\Omega}{1\,k\Omega} = 3.475\,V$$

Let’s take a short break…

Remaining Time until Controller Design Completion: 36 min*

*: I am speechless…..

MCC Configuration and Initial Firmware Test

Total time elapsed: 24 min

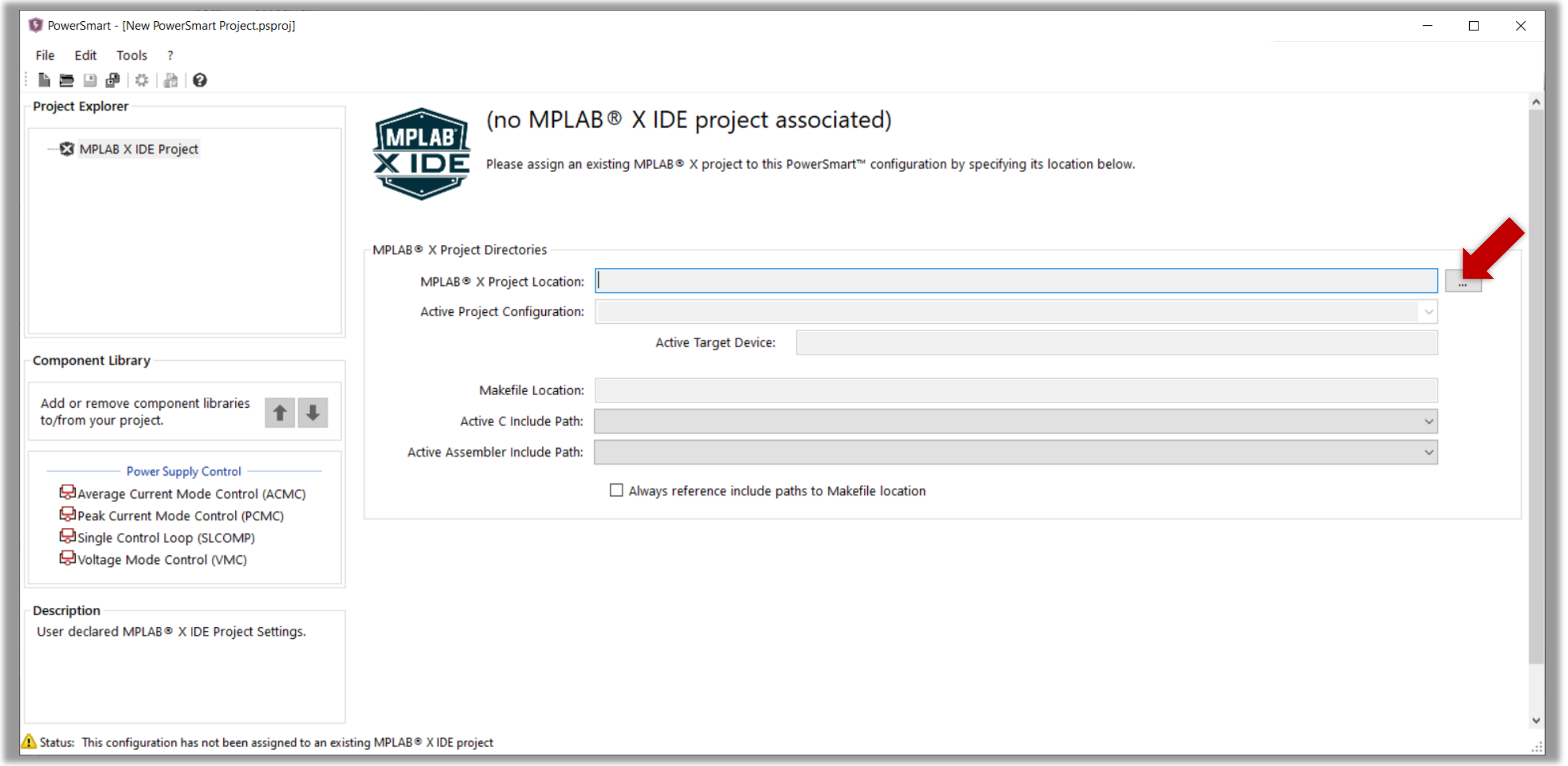
Agenda

Digital Power Supply from Scratch in 60 Minutes*

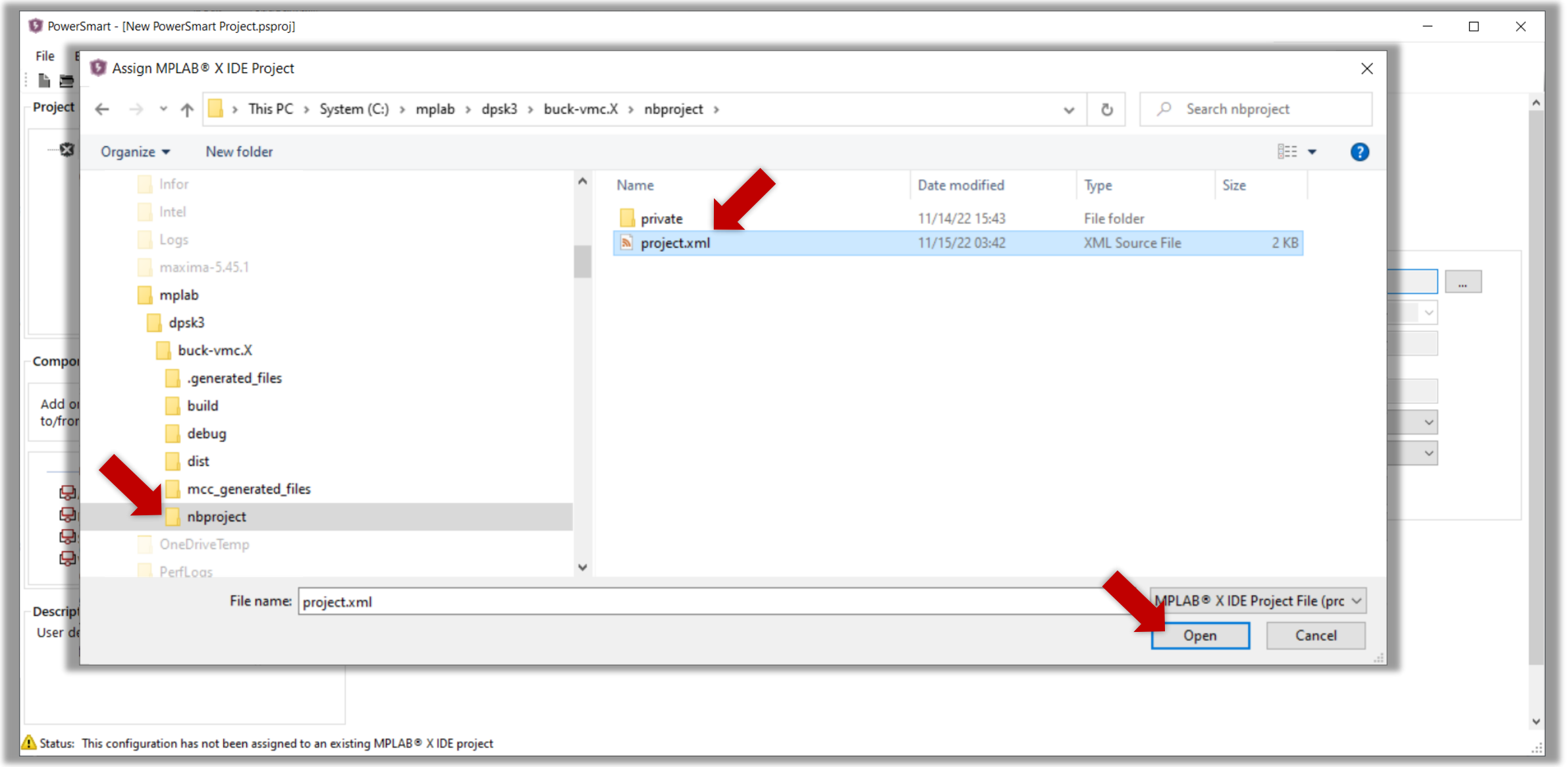
*: still not buying it ….

- Overview Digital Power Starter Kit 3 (DPSK3)
- Requirements and Scope
- **Hands-On**
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- Q&A

Assign MPLAB® X Project

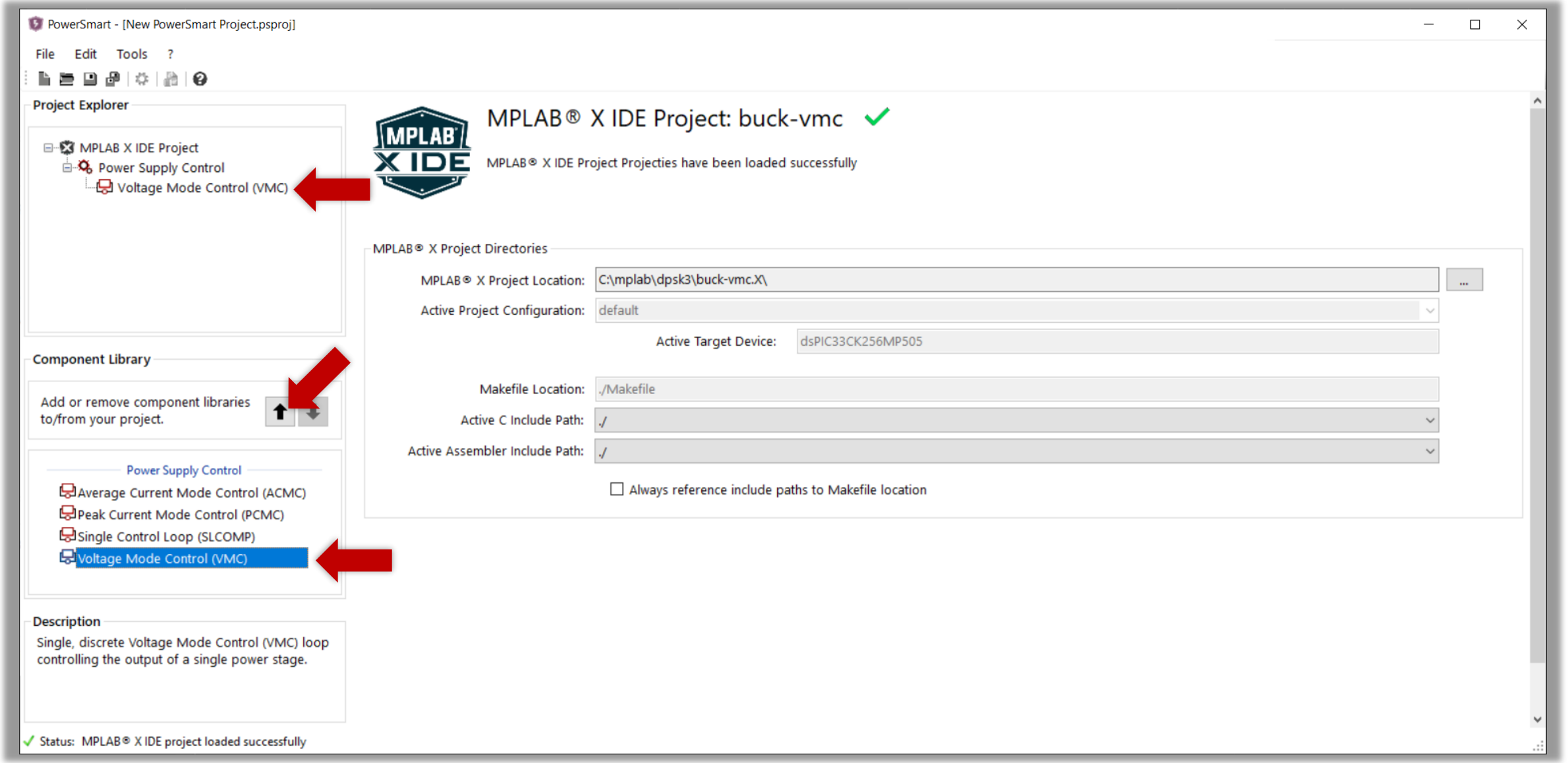


Assign MPLAB® X Project



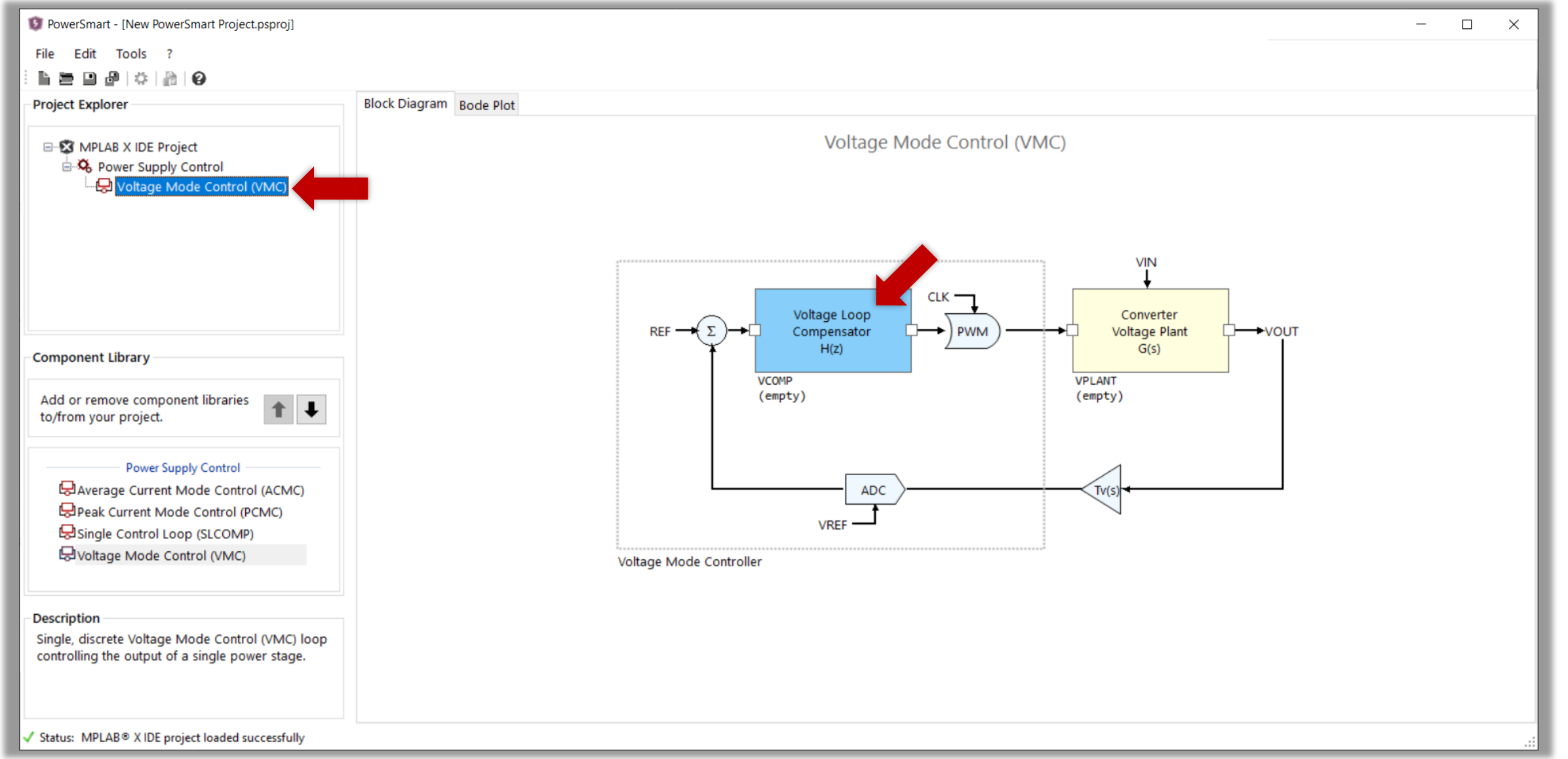
MICROCHIP

Add Voltage Loop Controller to PS™ Project

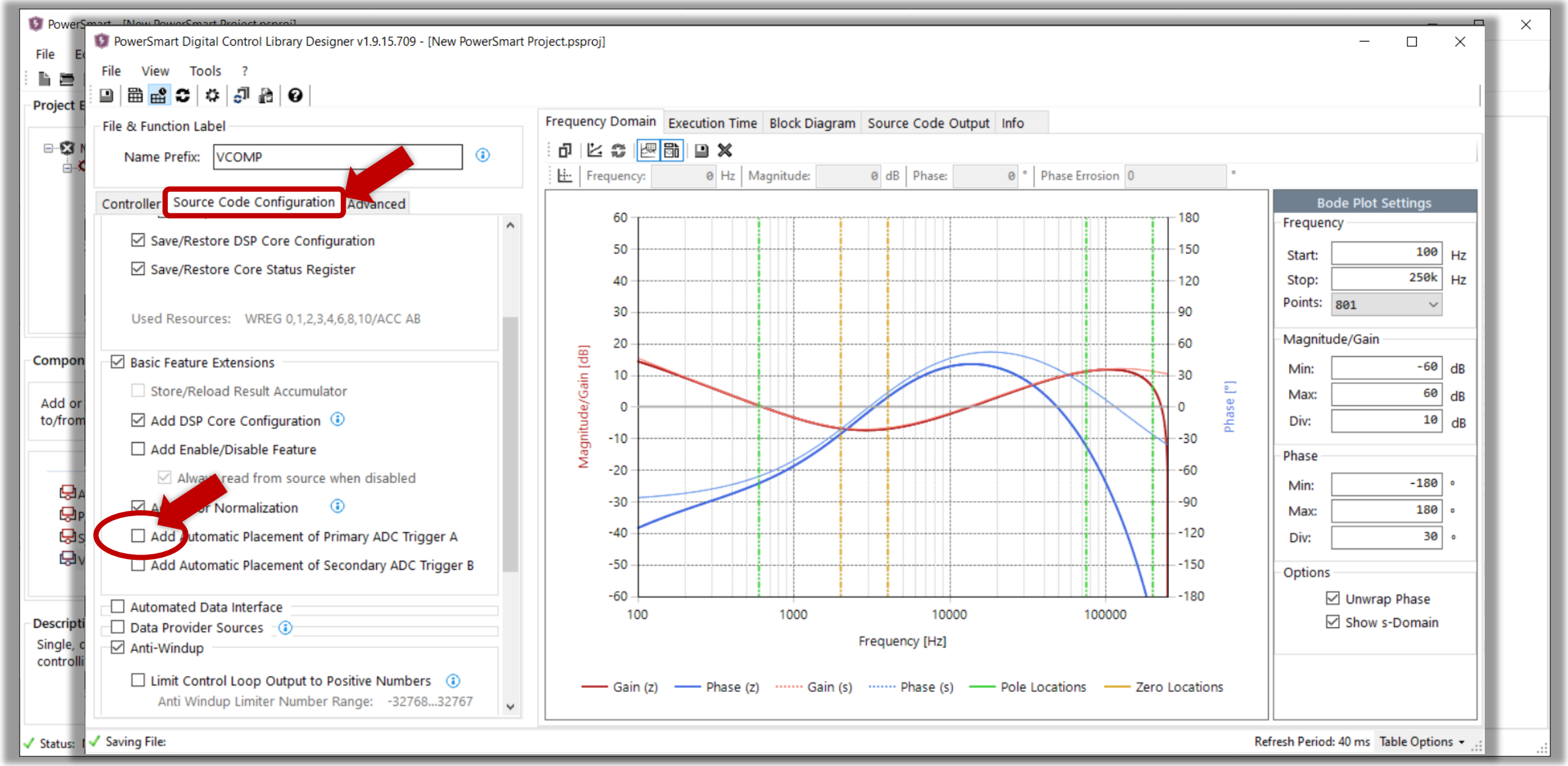


MICROCHIP

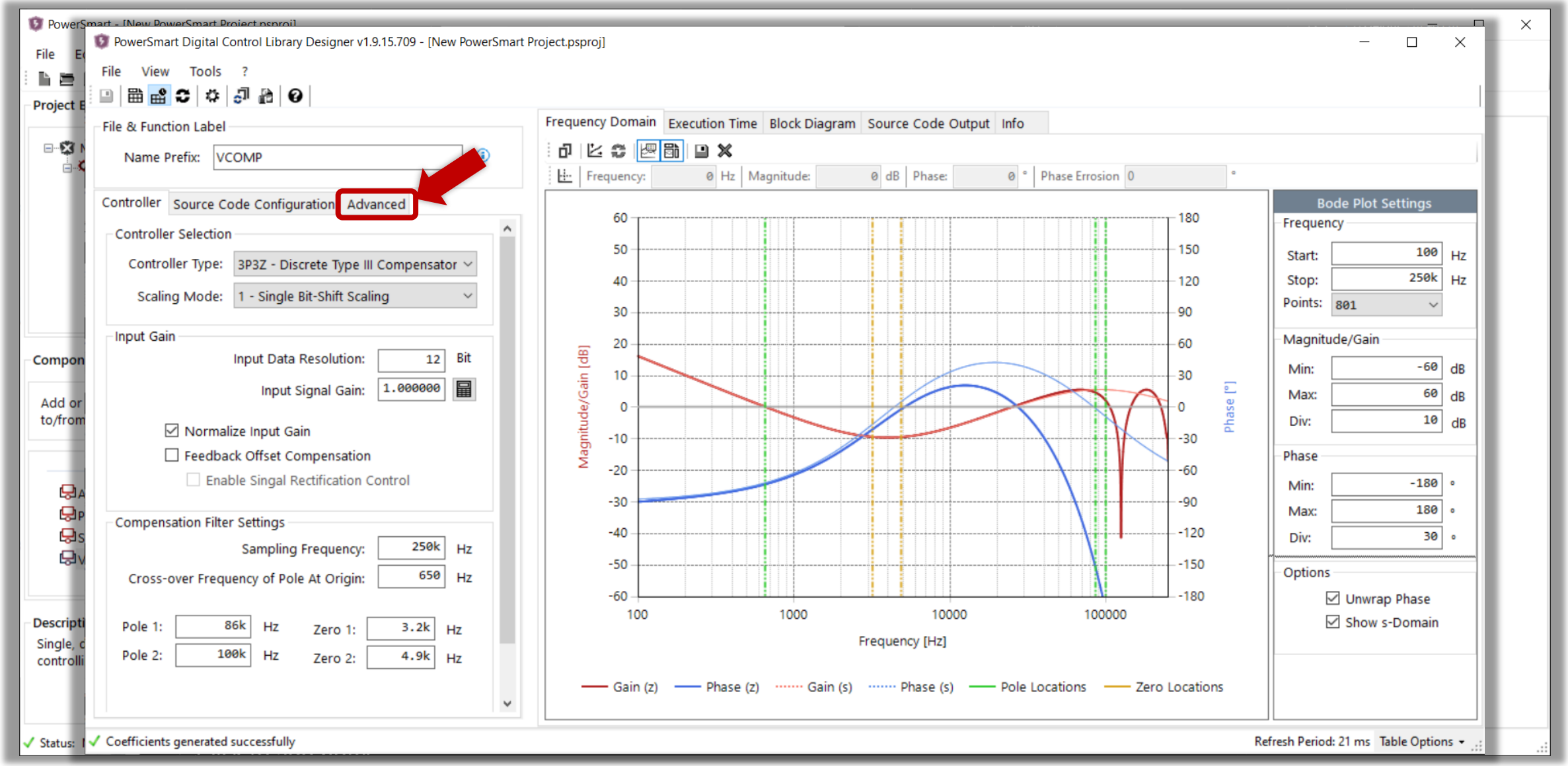
Open Voltage Loop Controller



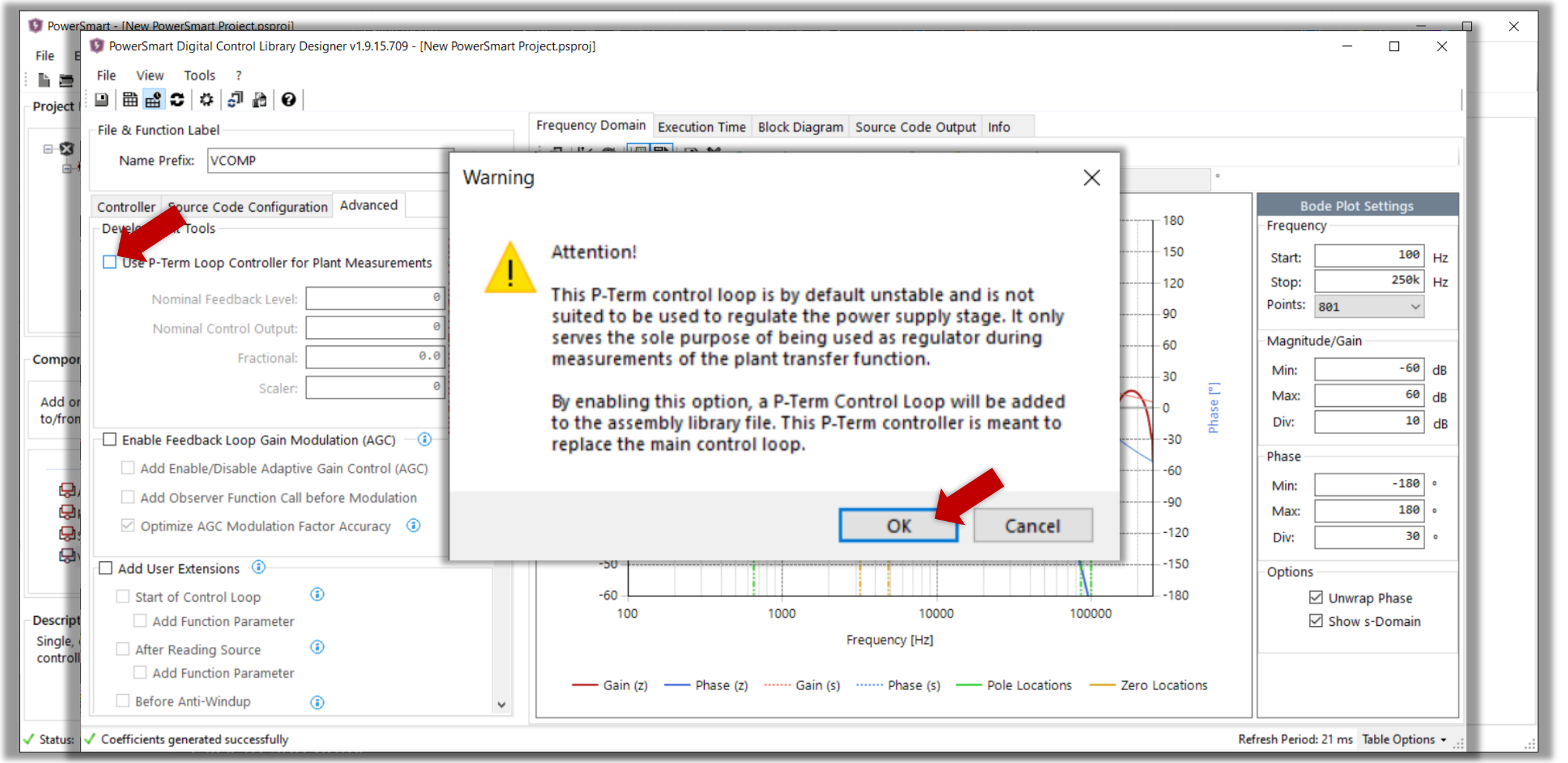
Turn Off Auto ADC Trigger Placement



Switch to Advanced Tab

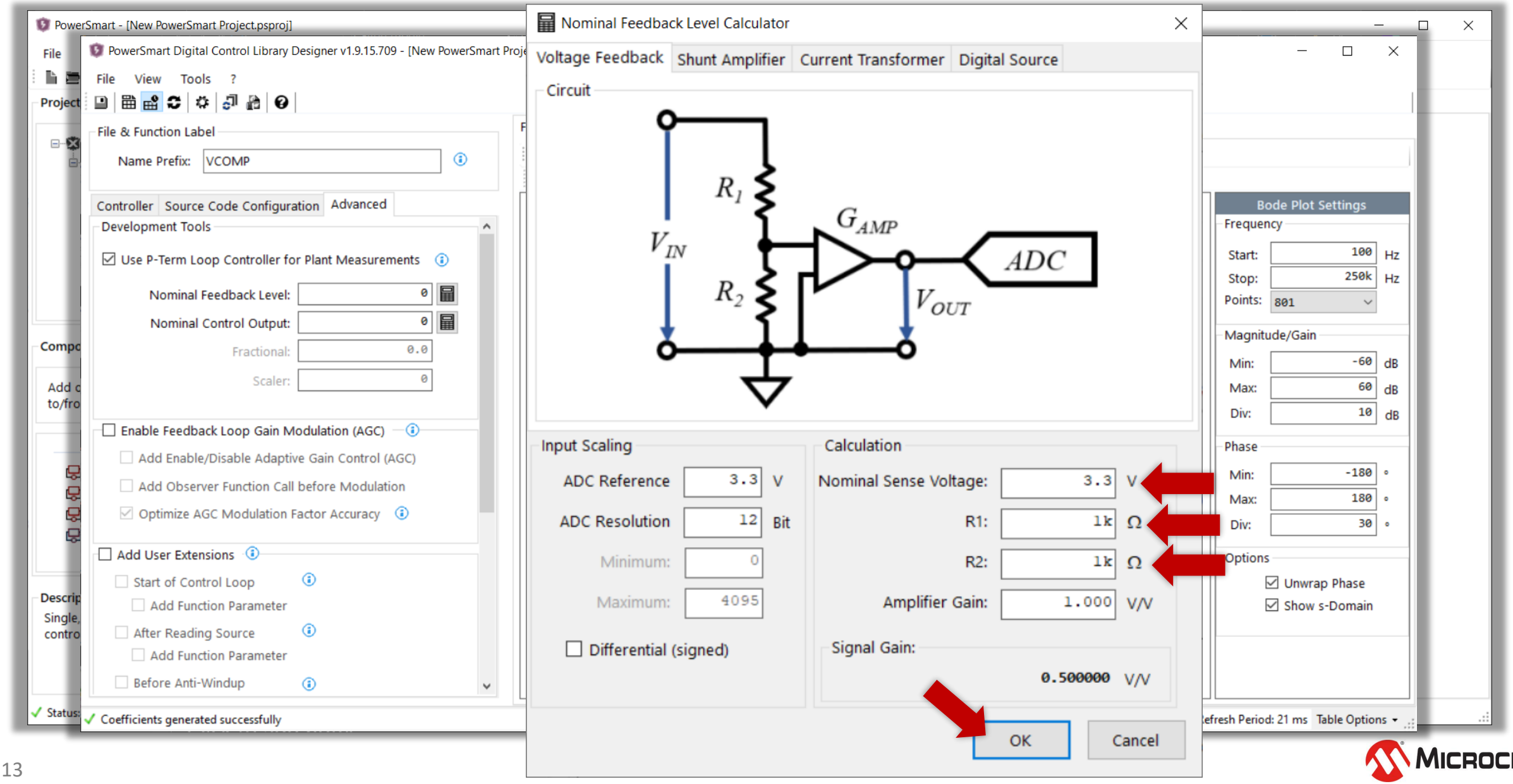


Enable Pterm Controller Generation

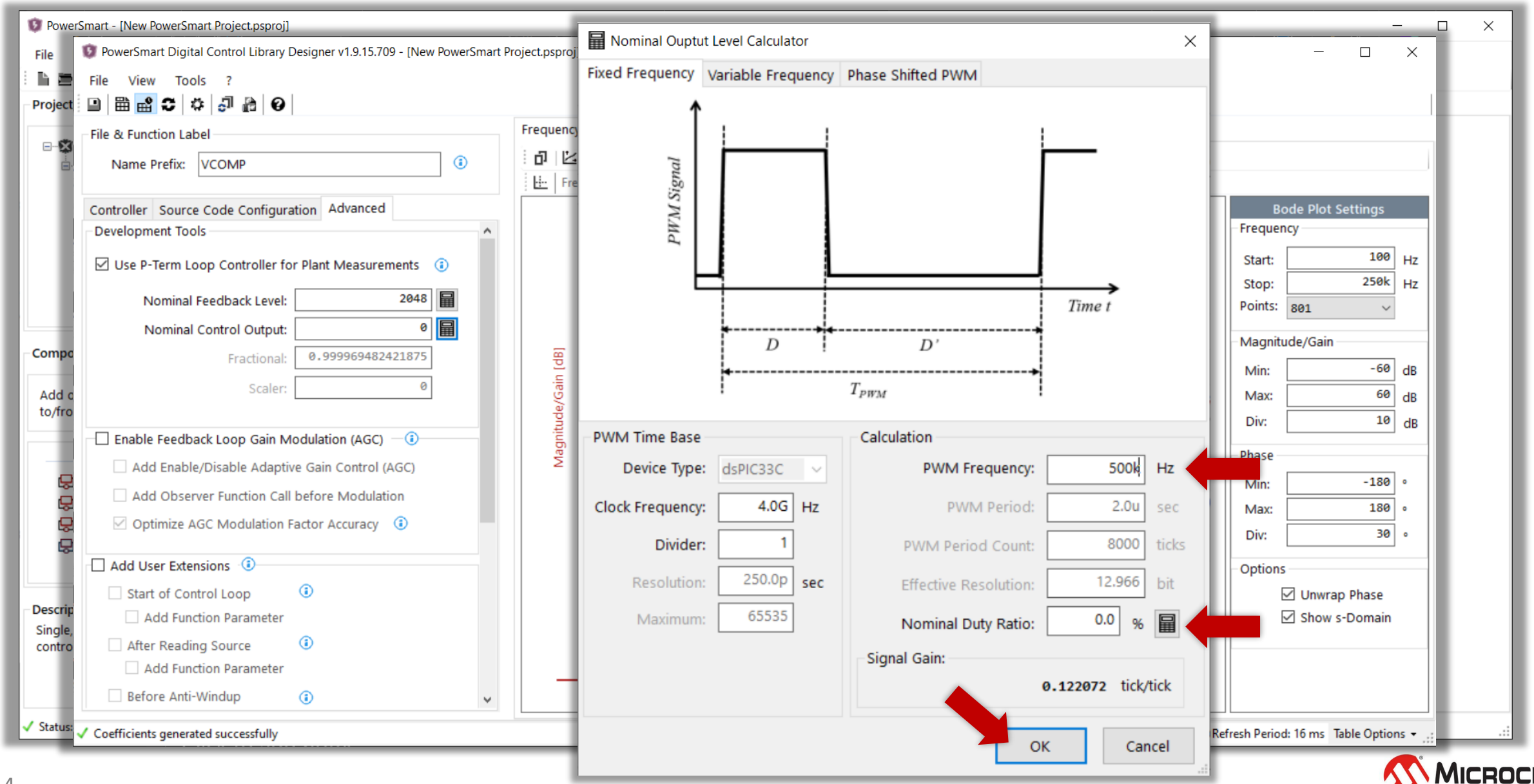


MICROCHIP

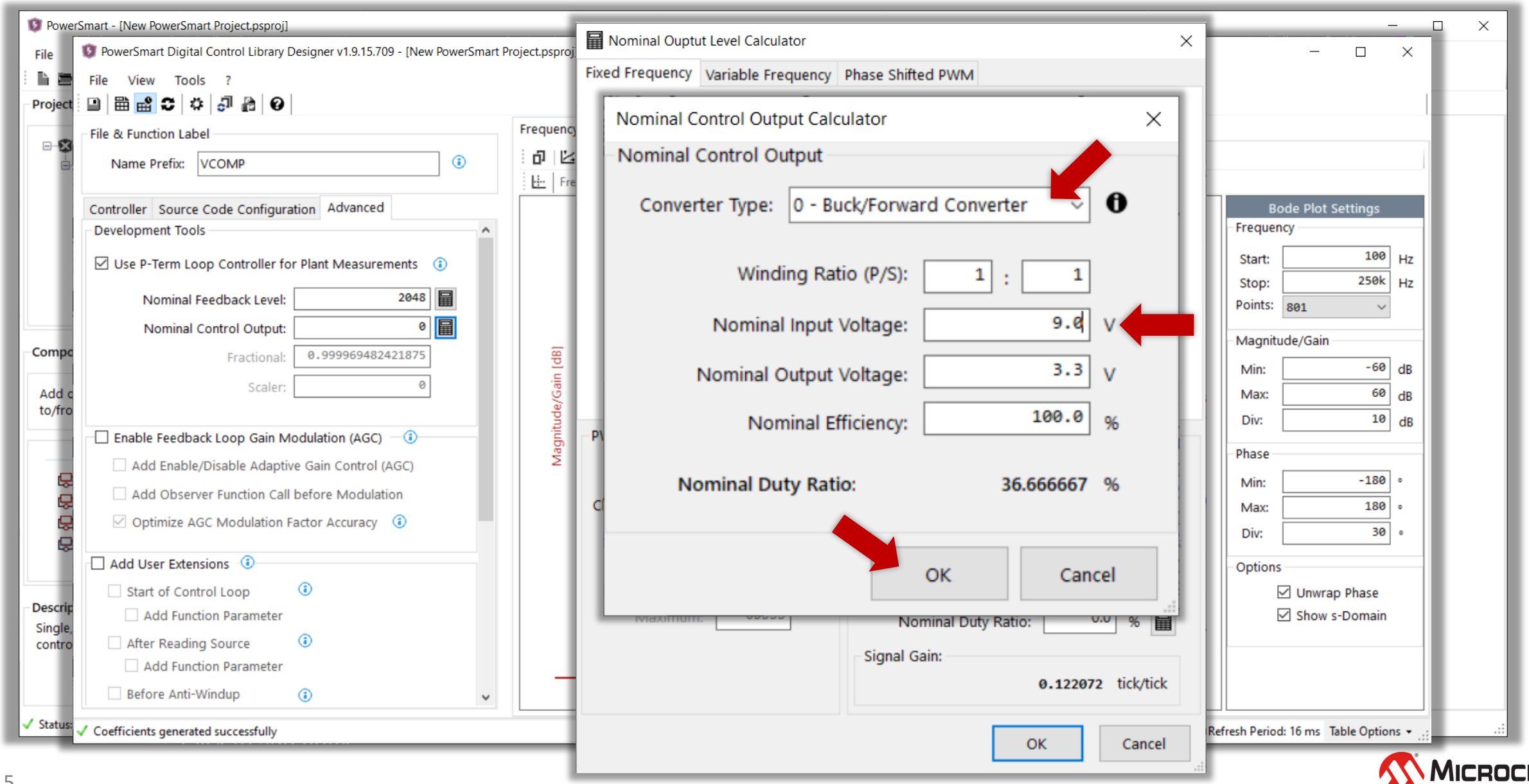
Configure Data Input



Configure Data Output Port

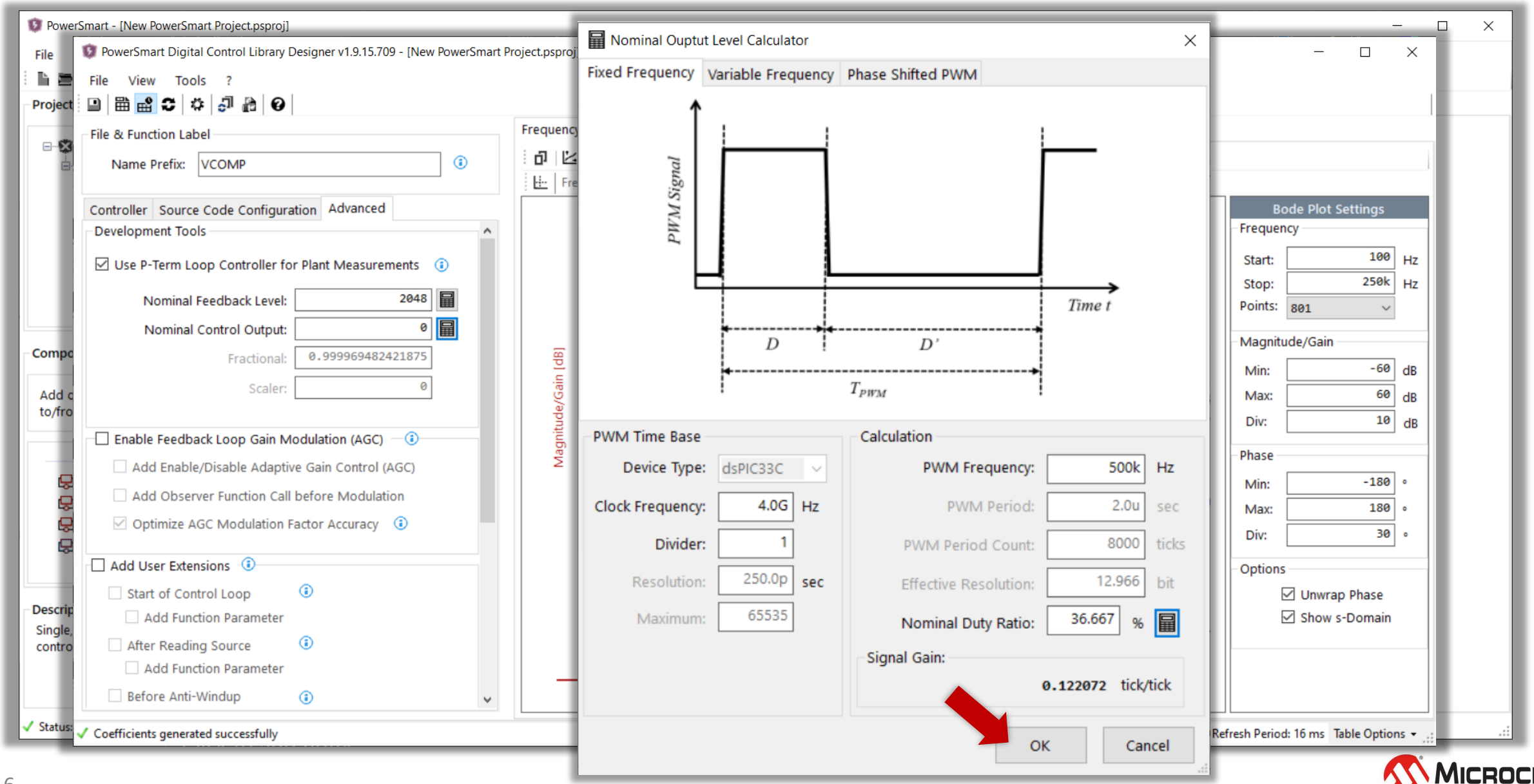


Get Estimated, Ideal Duty Ratio

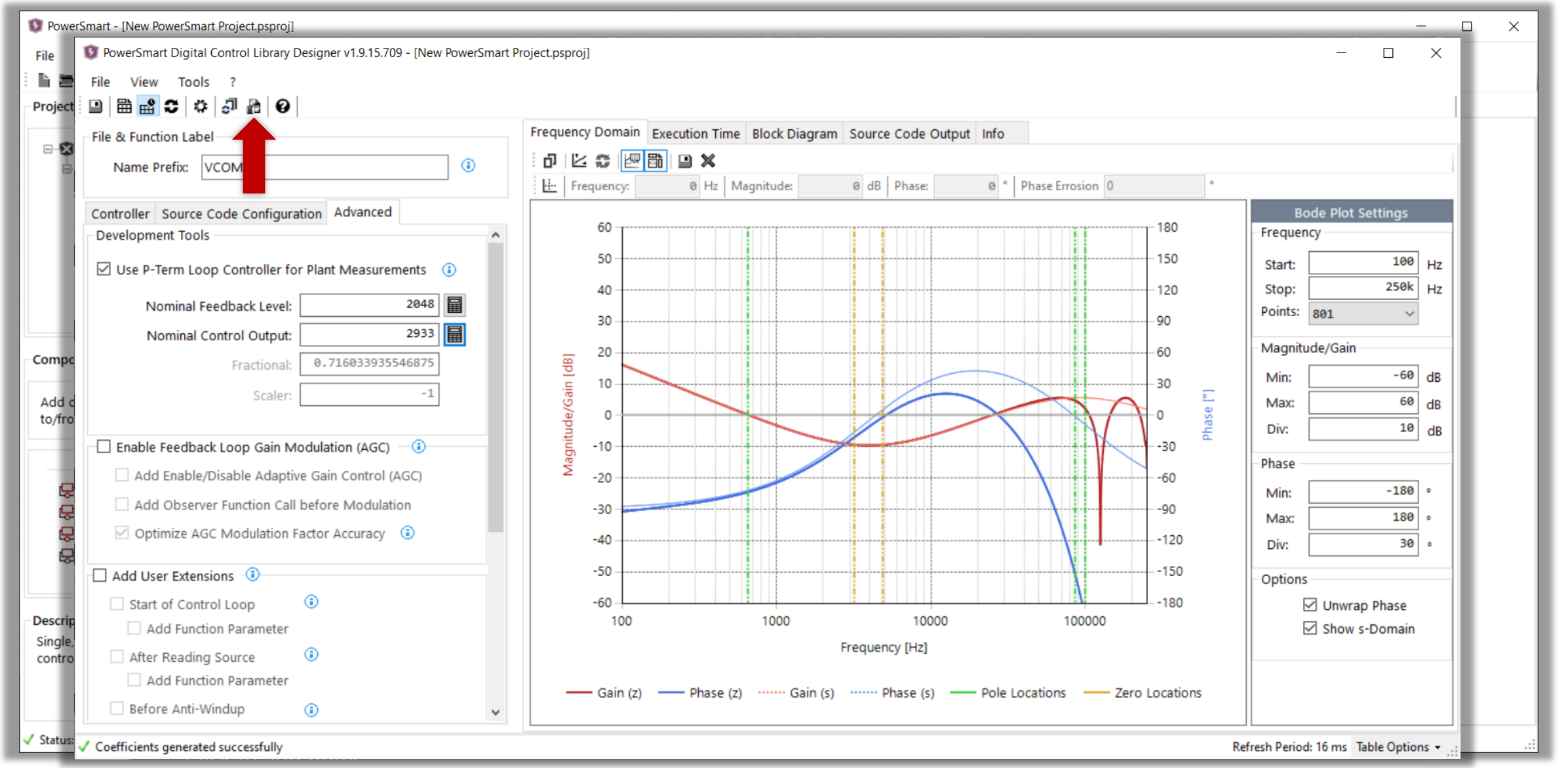


MICROCHIP

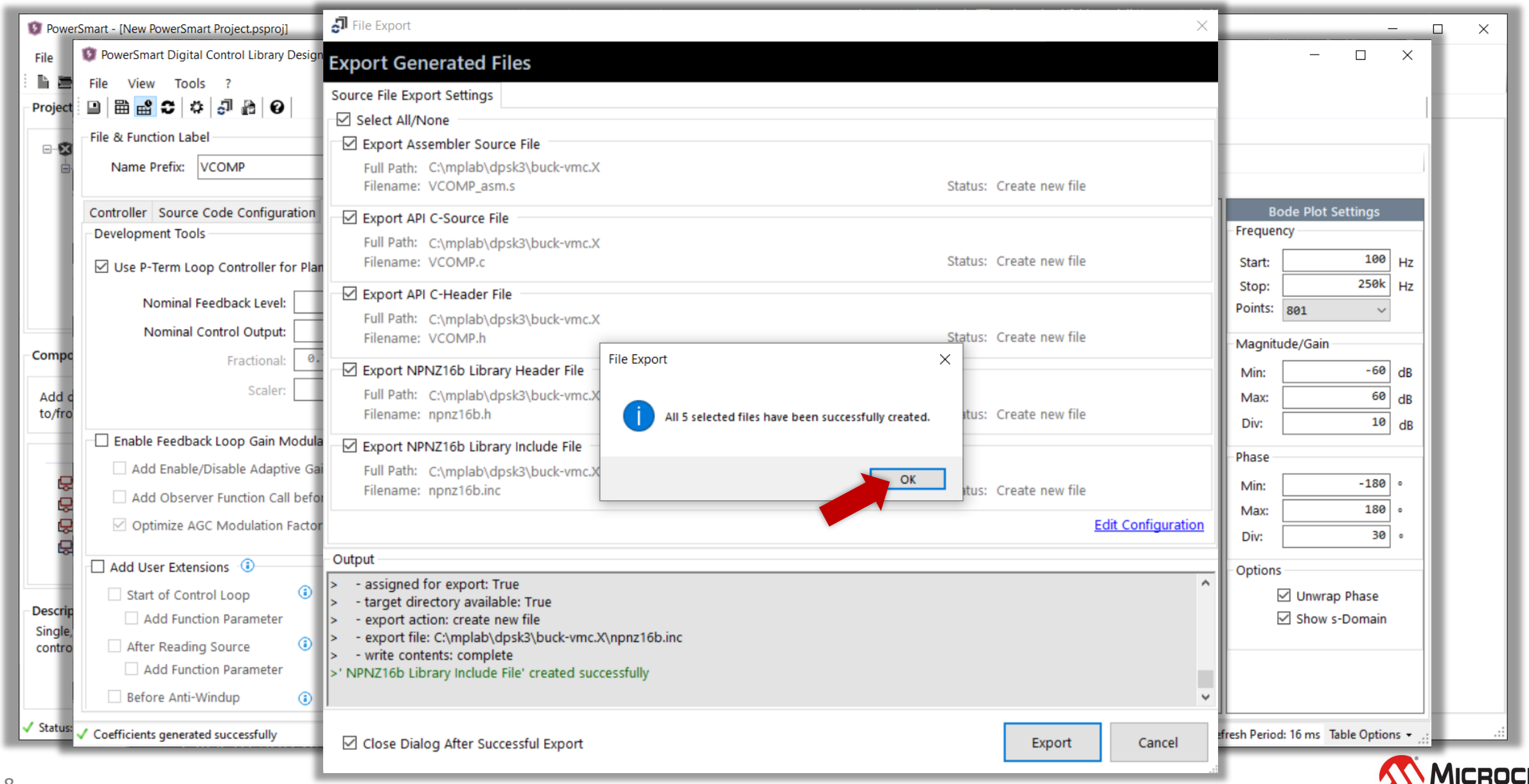
Apply Derived Data



Open Code Generator

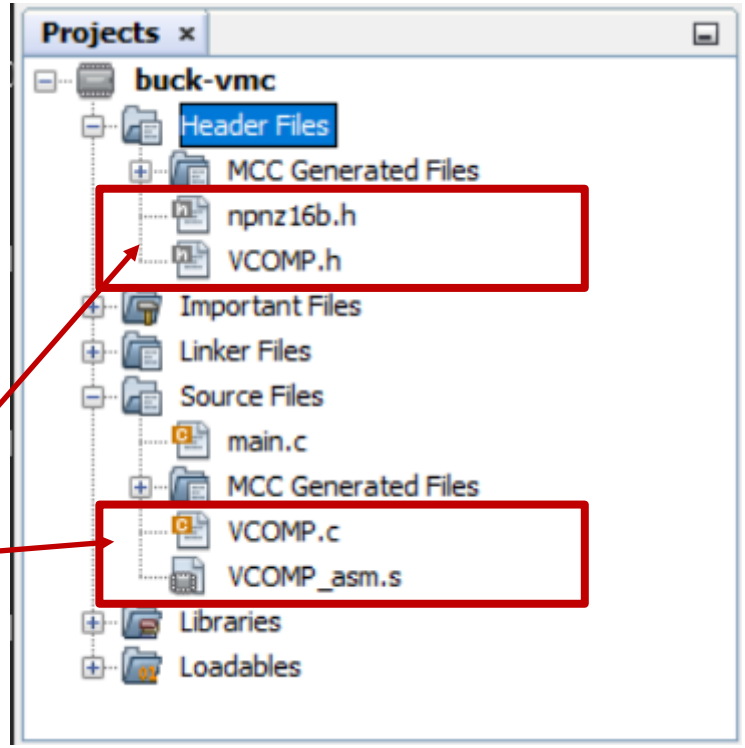


Generate Code



Import Generated Files into MPLAB® X Project

Name	Date modified	Type	Size
.generated_files	11/14/22 15:43	File folder	
build	11/15/22 02:43	File folder	
debug	11/15/22 01:35	File folder	
dist	11/15/22 02:44	File folder	
mcc_generated_files	11/15/22 01:23	File folder	
nbproject	11/14/22 15:43	File folder	
.gitignore	11/14/22 15:43	Git Ignore Source ...	1 KB
buck-vmc.mc3	11/15/22 03:38	MC3 File	2,442 KB
main.c	11/15/22 03:19	C Source File	3 KB
Makefile	11/14/22 15:43	File	4 KB
npnz16b.h	11/15/22 05:19	C Header Source F...	39 KB
npnz16b.inc	11/15/22 05:19	Include File	14 KB
VCOMP.c	11/15/22 05:19	C Source File	9 KB
VCOMP.h	11/15/22 05:19	C Header Source F...	13 KB
VCOMP_asm.s	11/15/22 05:19	Assembler Source	37 KB



MICROCHIP

Removing Test Code from main.c

```
void TMR1_Int(void)
{
    static int led_tmr = 0;

    // blink LED 1Hz
    if(++led_tmr>=500) {
        led_tmr = 0;
        DGBLED_Toggle();
    }

    // State Machine
    if (PG1DC < 3000) PG1DC++;
}
```

Including VCOMP header files in main.c

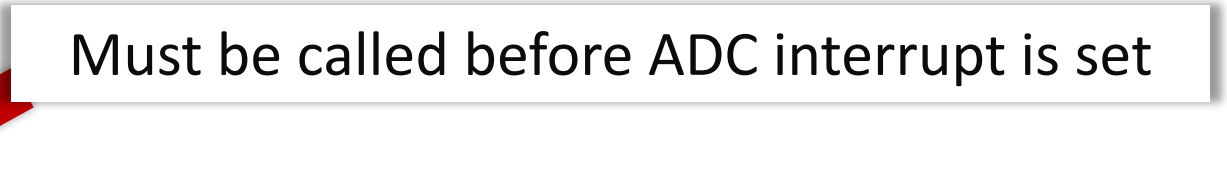
```
/**
  Section: Included Files
*/
#include "mcc_generated_files/system.h"
#include "mcc_generated_files/pin_manager.h"
#include "mcc_generated_files/pwm.h"
#include "mcc_generated_files/adc1.h"
#include "mcc_generated_files/tmr1.h"
#include "VCOMP.h"
[…]
int main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    ADC1_Setchannel_AN13InterruptHandler(&AN13_Int);
    TMR1_SetInterruptHandler(&TMR1_Int);     while (1)
    {
        // Add your application code
    }
```

Add Controller Initialization Code to main.c

```
void AN13_Int(void)
{
    adc_vin = ADCBUF12;
    adc_vbuck = ADCBUF13;
}

void CONTROL_Initialize(void)
{
    // initialize compensator
    VCOMP_Initialize(& VCOMP);
    VCOMP.Ports.Source.ptrAddress = (unsigned*)&adc_vbuck;
    VCOMP.Ports.Target.ptrAddress = (unsigned*)&PG1DC;
    VCOMP.Ports.ptrControlReference = (unsigned*)&ref_vbuck;
    VCOMP.Limits.MinOutput = 0;
    VCOMP.Limits.MaxOutput = (int)(0.8*PG1PER); // 80% DC
}

int main(void)
{
```

Call Controller Initialization from main() in main.c

```
int main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    CONTROL_Initialize();
    ADC1_Setchannel_AN13InterruptHandler(&AN13_Int);
    TMR1_SetInterruptHandler(&TMR1_Int);

    while (1)
    {
        // Add your application code
    }
    return 1;
}
```



Call Controller from ADC Interrupt in main.c

```
void AN13_Int(void)
{
    adc_vin = ADCBUF12;
    adc_vbuck = ADCBUF13;
    VCOMP_PTermUpdate(&VCOMP);
}

void CONTROL_Initialize(void)
{
    // initialize compensator
    VCOMP_Initialize(& VCOMP);
    VCOMP.Ports.Source.ptrAddress = (unsigned*)&adc_vbuck;
    VCOMP.Ports.Target.ptrAddress = (unsigned*)&PG1DC;
    VCOMP.Ports.ptrControlReference = (unsigned*)&ref_vbuck;
    VCOMP.Limits.MinOutput = 0;
    VCOMP.Limits.MaxOutput = (int)(0.8*PG1PER); // 80% DC
}

int main(void)
```

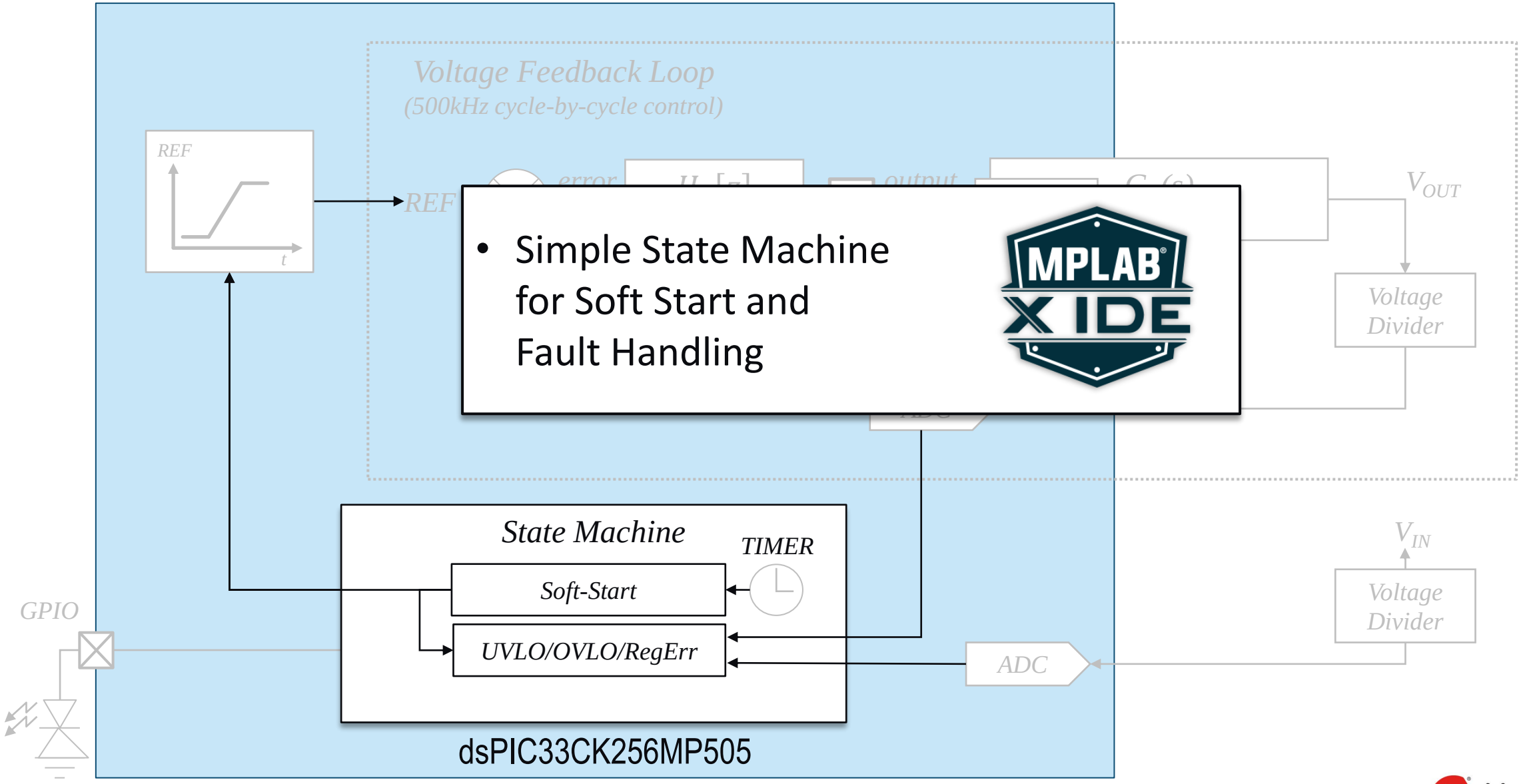
Agenda

Digital Power Supply from Scratch in 60 Minutes*

*: Are we even still within the 60 min?

- Overview Digital Power Starter Kit 3 (DPSK3)
- Requirements and Scope
- **Hands-On**
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- Q&A

Requirements



MICROCHIP

Add Scaling Macros to main.c

```
// Define Global Variables:
int adc_vin = 0;     // Scaling: 155.1 steps/V
int adc_vbuck = 0;   // Scaling: 620.5 steps/V
int ref_vbuck = 0;   // Scaling: 620.5 steps/V


// Add Defines for Hardware Properties
#define AN12_VIN_SCALE   (float)( 1.0/8.0 * 4095.0/3.3 ) // Input voltage: 155 digits/V
#define AN13_VBUCK_SCALE (float)( 1.0/2.0 * 4095.0/3.3 ) // Buck output voltage: 620 ticks/V

#define VIN_TH_ON  (int16_t)( 6.5   * AN12_VIN_SCALE )   // Switch-on threshold: 6.5 V
#define VIN_TH_OFF (int16_t)( 5.5   * AN12_VIN_SCALE )   // Switch-off (undervoltage)
                                                         // threshold: 5.5 V
#define VBUCK_REF  (int16_t)( 3.3   * AN13_VBUCK_SCALE ) // Output voltage set value: 3.3 V
#define VBUCK_DREF (int16_t)( 0.2   * AN13_VBUCK_SCALE ) // max. output voltage error: 200 mV
#define VBUCK_RAMP (int16_t)( 10e-3 * AN13_VBUCK_SCALE ) // soft start ramp: 10 mV/ms

[…]
```

Add Local SM-Variables to Timer ISR in main.c

```
[…]

// Initial TMR1_Int() function, do only LED blinking
void TMR1_Int(void)
{
    static int led_tmr = 0;
    static int fsm_state = 0;
    static int start_tmr = 0;
    static int err_tmr = 0;

    // blink LED 1Hz
    if(++led_tmr>=500) {
        led_tmr = 0;
        USER_LED_Toggle();
    }

}
```

Add State Machine to Timer ISR in main.c

```
// Initial TMR1_Int() function, do only LED blinking
void TMR1_Int(void)
{
    […]
    // output voltage supervision, increment error timer if voltage is too low
    err_tmr = (ref_vbuck-adc_vbuck)>VBUCK_DREF ? err_tmr+1 : 0;

    switch(fsm_state) {
        case 0: // State 0: wait for Vin>threshold (delay time)
            ref_vbuck = 0;
            err_tmr = 0;
            if(adc_vin<VIN_TH_ON) {
                start_tmr = 0;
            } else if(++start_tmr>2000) {
                fsm_state = 1;
            }
        break;
    }
}
```

Add State Machine to Timer ISR in main.c

```
// Initial TMR1_Int() function, do only LED blinking
void TMR1_Int(void)
{
    […]
    switch(fsm_state) {
        case 0: // State 0: wait for Vin>threshold (delay time)
        […]
        break;
        case 1: // State 1: ramp up buck reference value (soft start)
           if(ref_vbuck < VBUCK_REF) {
               ref_vbuck += VBUCK_RAMP;
           }
           if((adc_vin<VIN_TH_OFF)||(err_tmr>200)) { // low input voltage OR error for >200ms
               start_tmr = 0;
               fsm_state = 0;
           }
        break;
    }
}
```

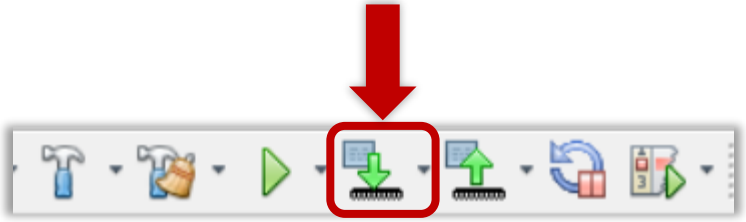

Testing Time !

Give it a try…

- **Build the Project**

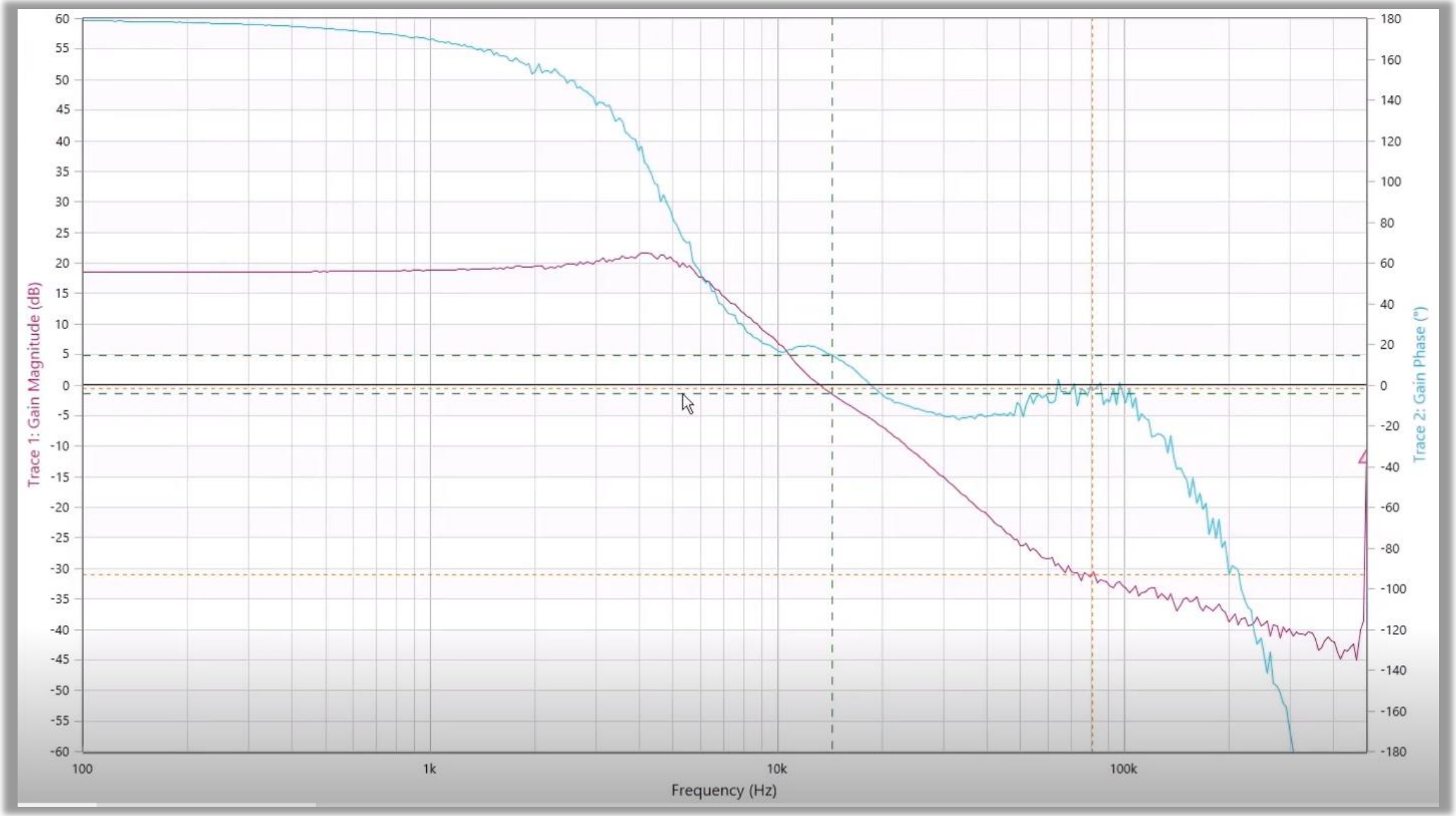
  - If the Build was successful …
  - Or check error message and fix what's broken

- **Program the Project**



  - When programming was successful, the red LED below dsPIC should blink

Bode Plot Results



MICROCHIP

Agenda

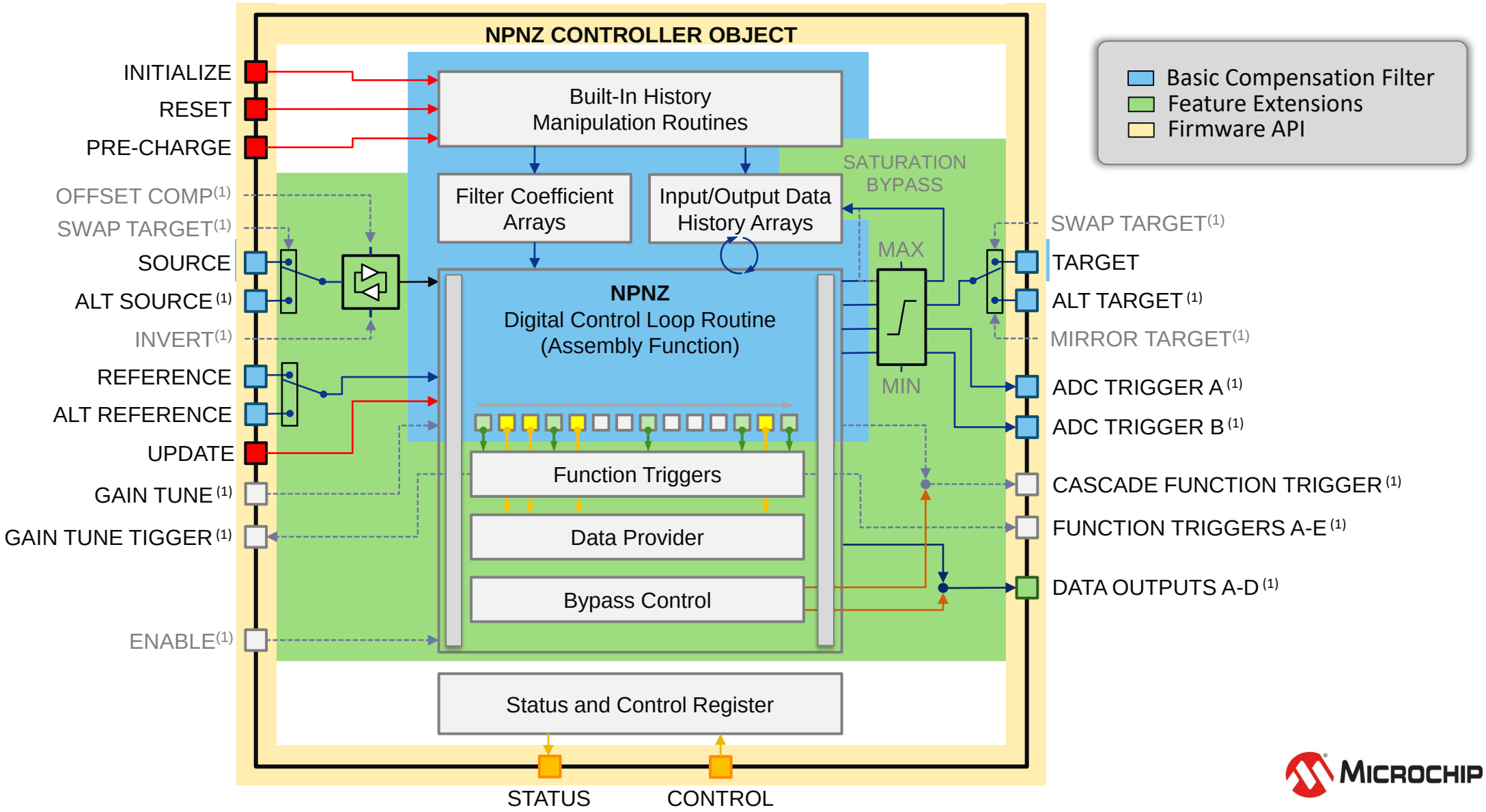
Digital Power Supply from Scratch in 60 Minutes*

*: Is anyone waring a watch?

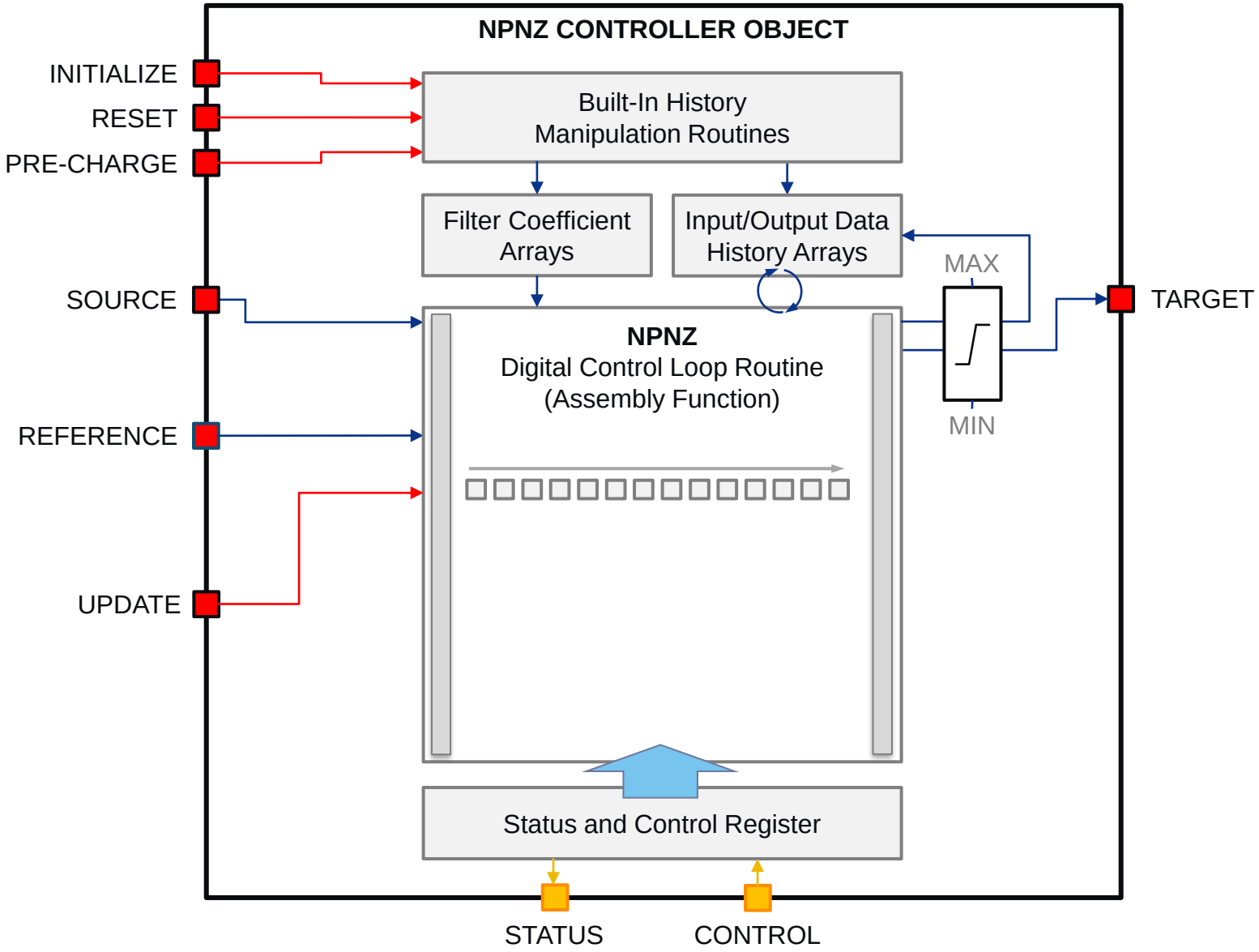
- Overview Digital Power Starter Kit 3 (DPSK3)
- Requirements and Scope
- **Hands-On**
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- Q&A

Design Time !

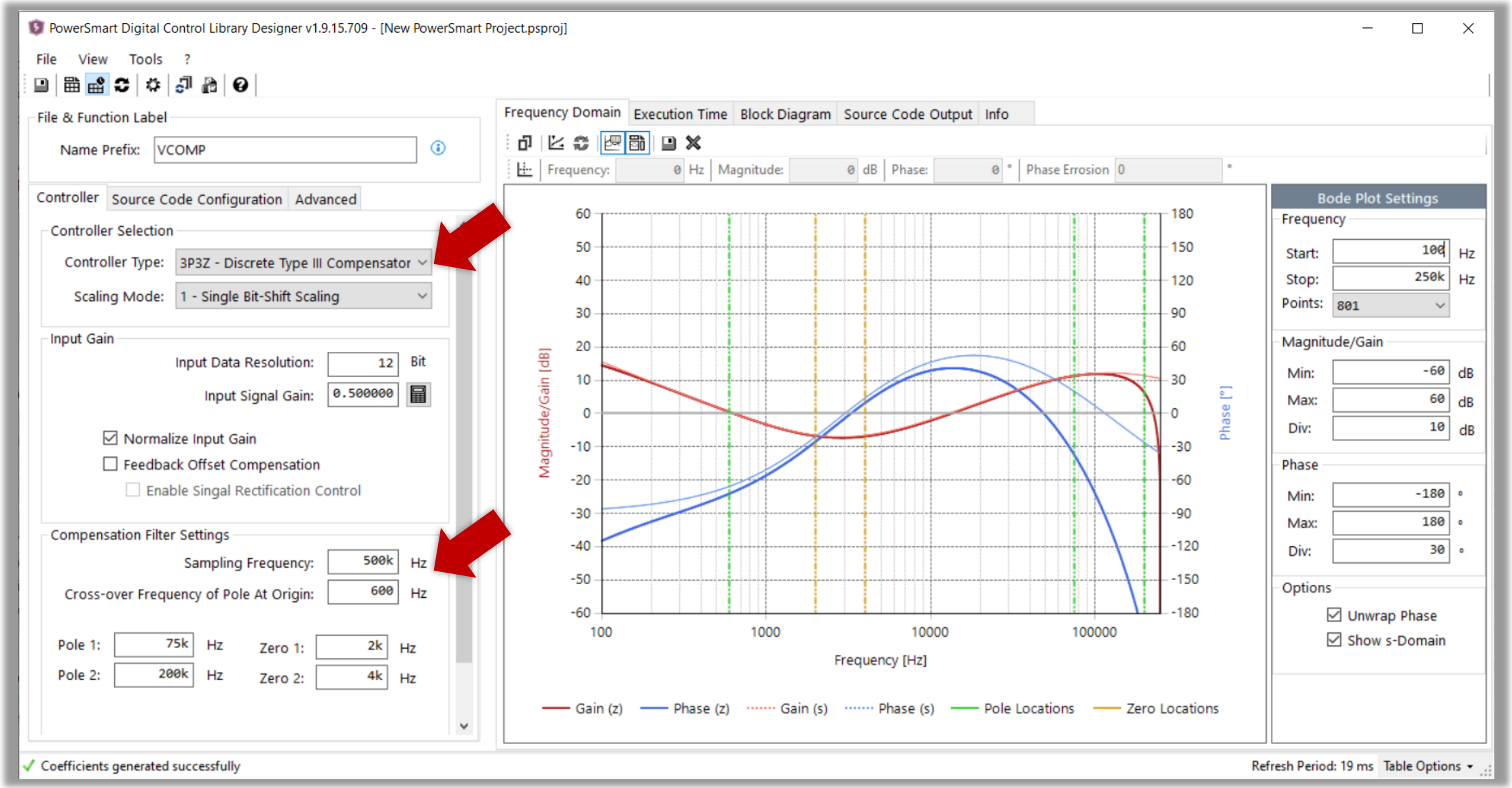
PS-DCLD Feedback Loop Block Diagram



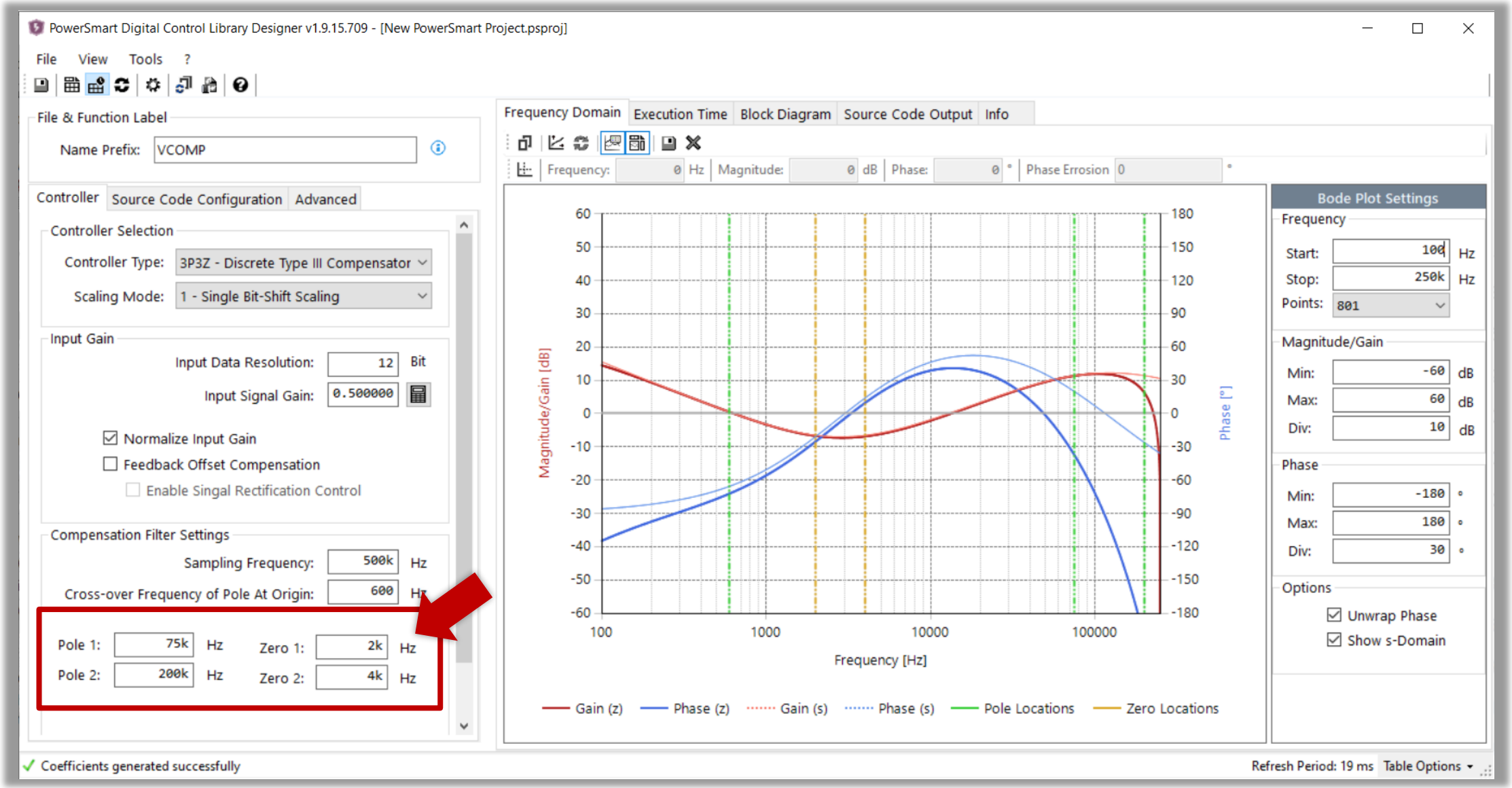
PS-DCLD VMC Feedback Loop Block Diagram



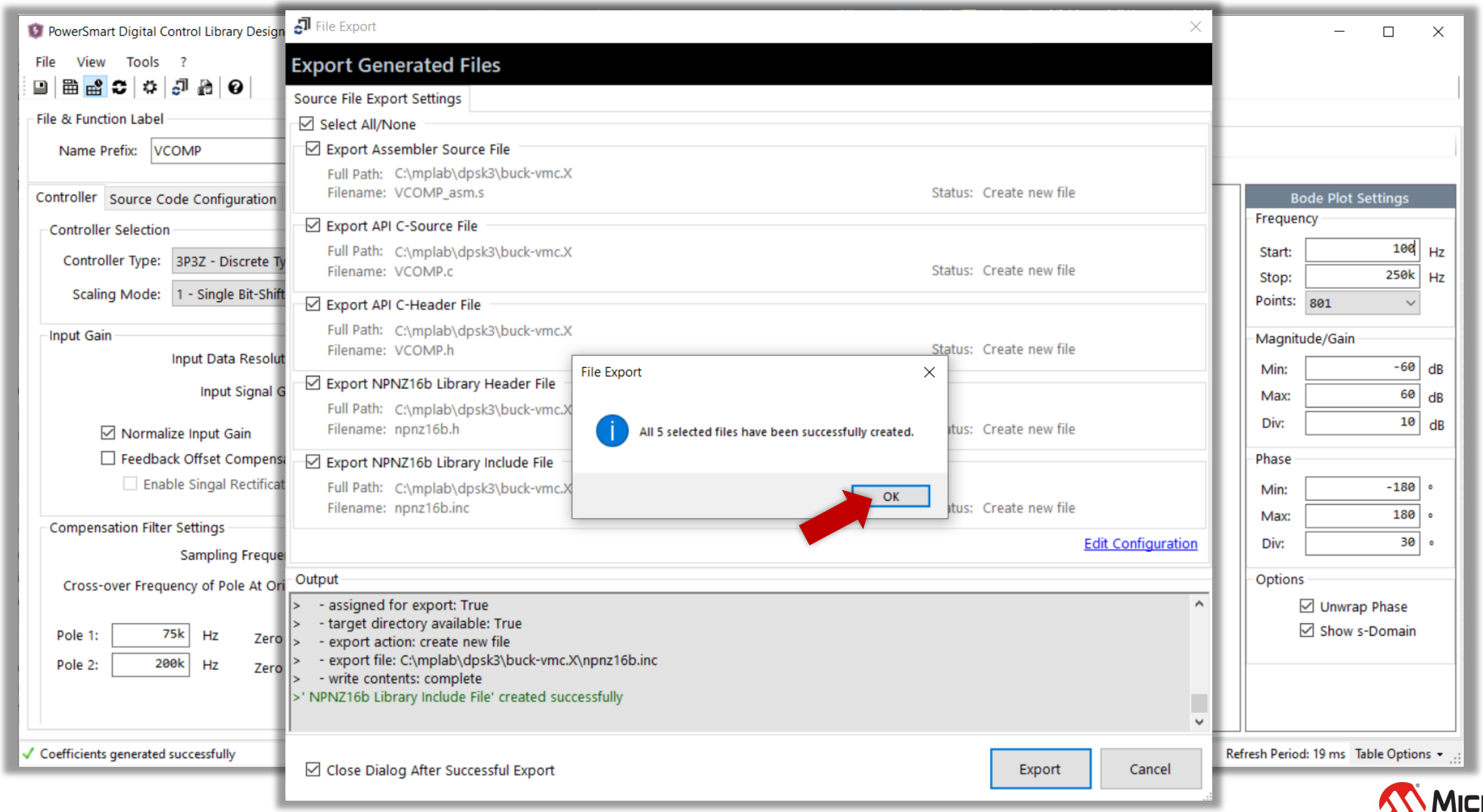
Control Loop Adjustment



Control Loop Adjustment



Control Loop Adjustment



Control Loop Implementation

```
void AN13_Int(void)
{
    adc_vin = ADCBUF12;
    adc_vbuck = ADCBUF13;
    // VCOMP_PTermUpdate(&VCOMP);
    VCOMP_Update(&VCOMP);
}
```

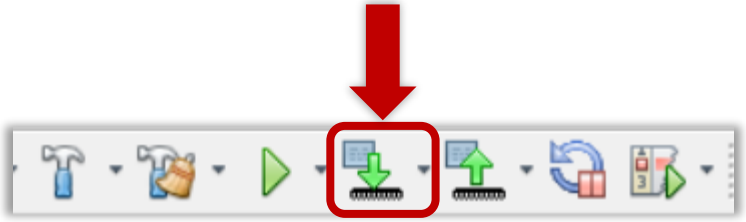

Testing Time !

Give it a try…

- **Build the Project**

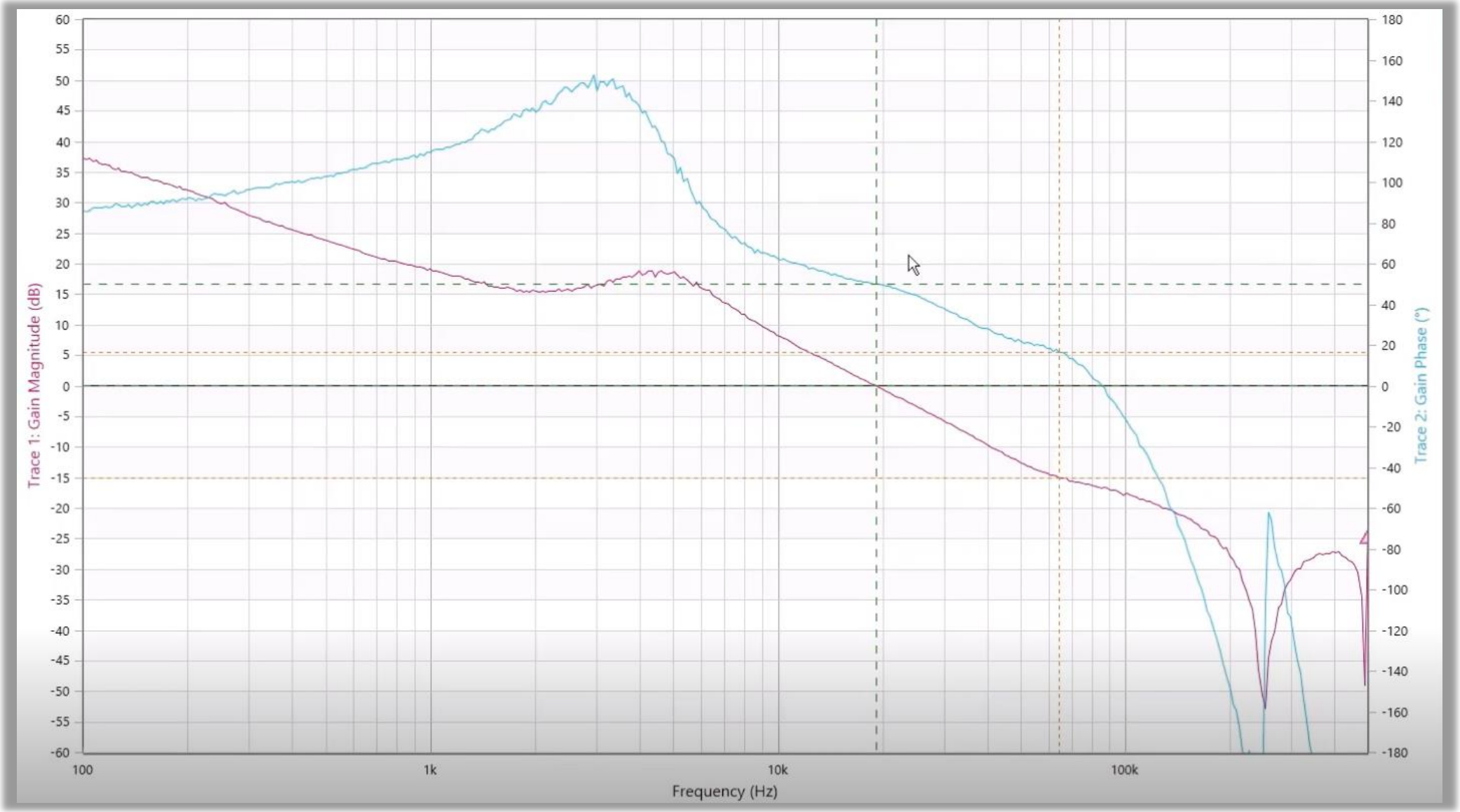
  - If the Build was successful …
  - Or check error message and fix what's broken

- **Program the Project**



  - When programming was successful, the red LED below dsPIC should blink

Bode Plot Results



MICROCHIP

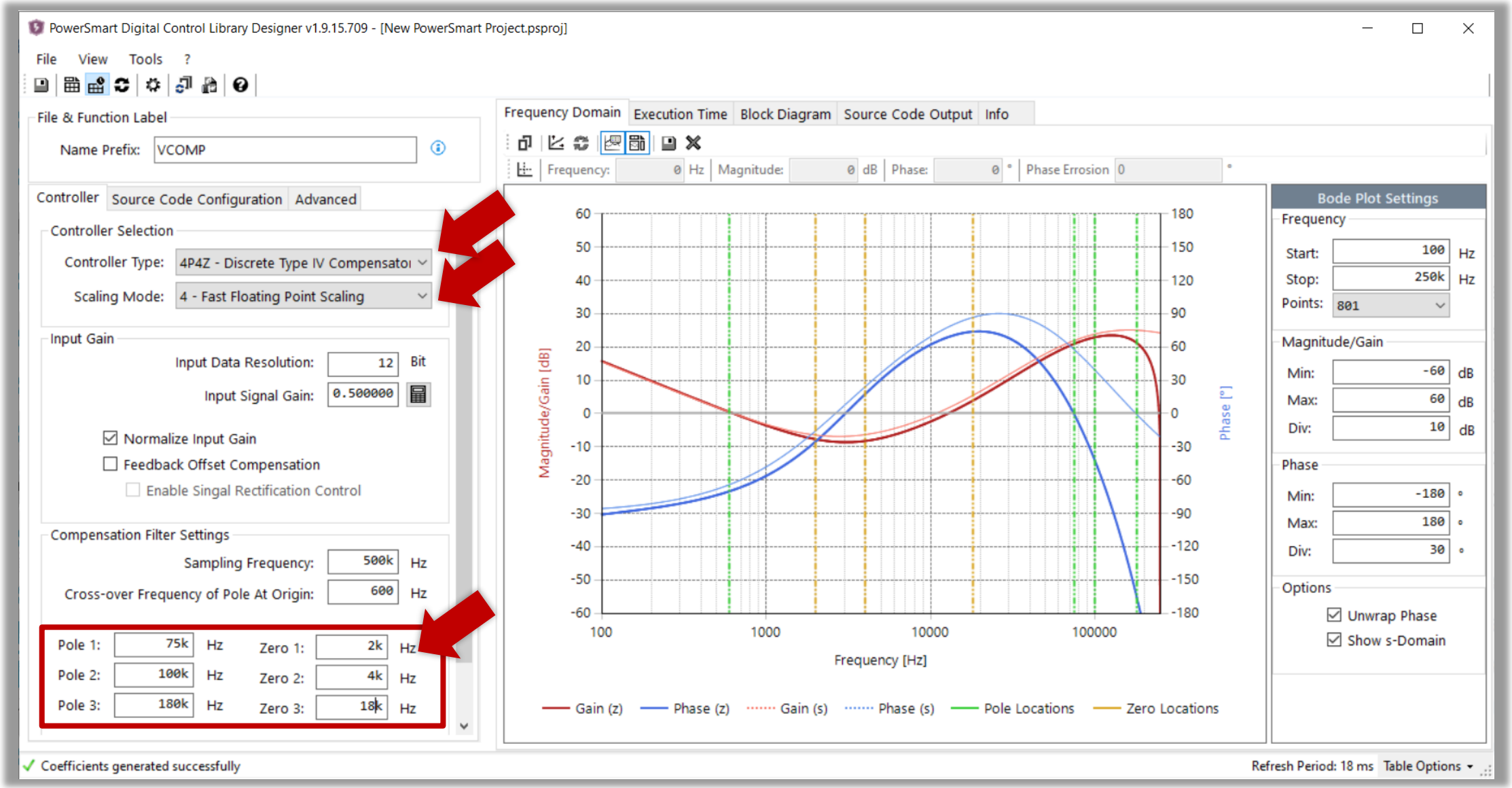
Agenda

Digital Power Supply from Scratch in 60 Minutes*

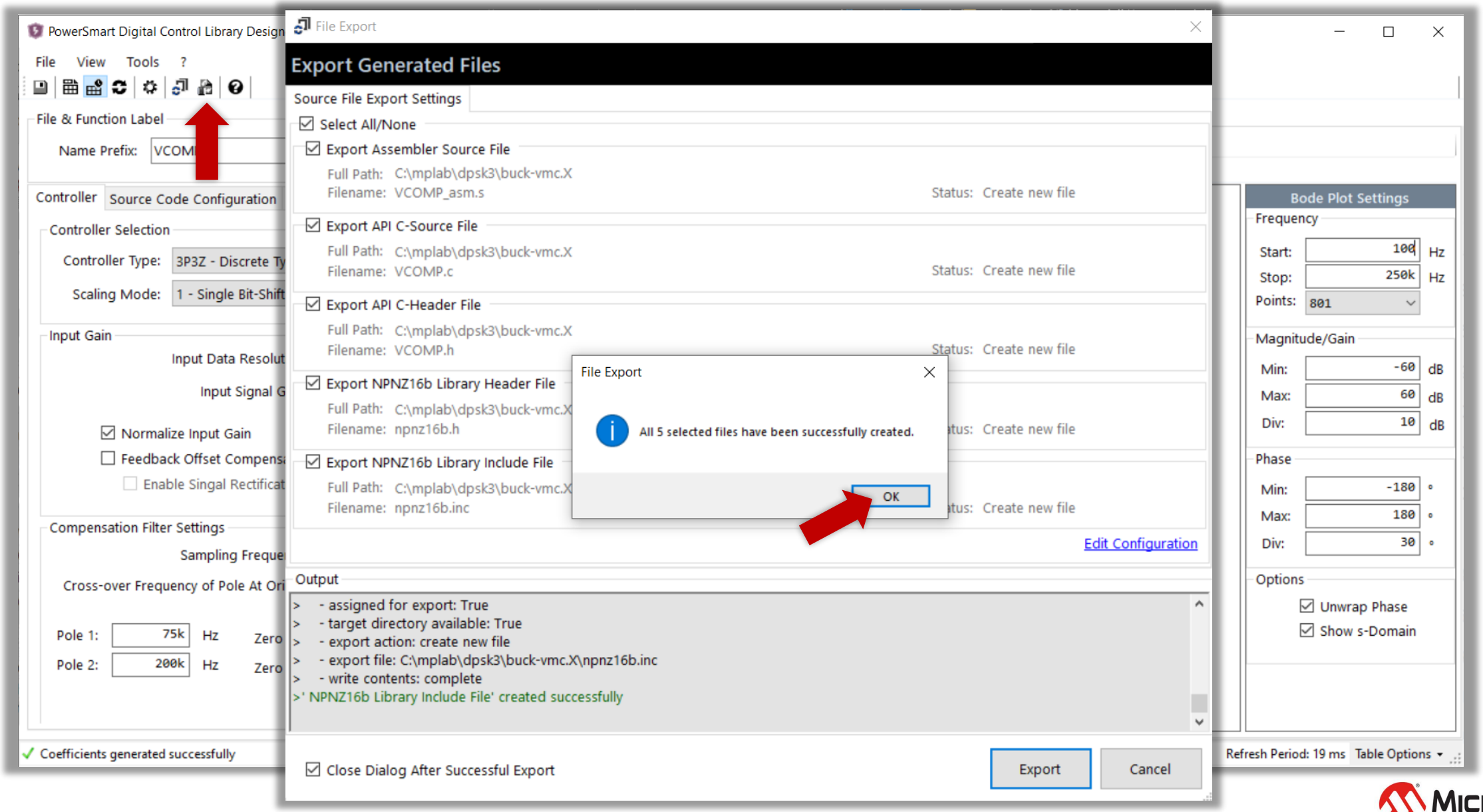
*: This will take forever!

- Overview Digital Power Starter Kit 3 (DPSK3)
- Requirements and Scope
- **Hands-On**
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- Q&A

Control Loop Adjustment



Control Loop Adjustment



MICROCHIP

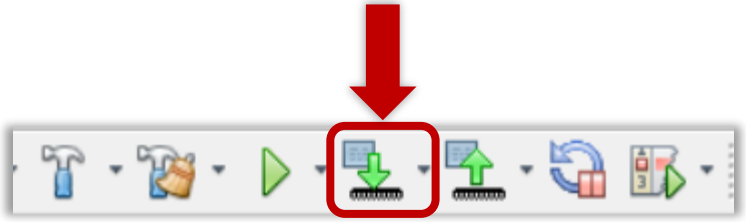

Testing Time !

Give it a try…

- **Build the Project**

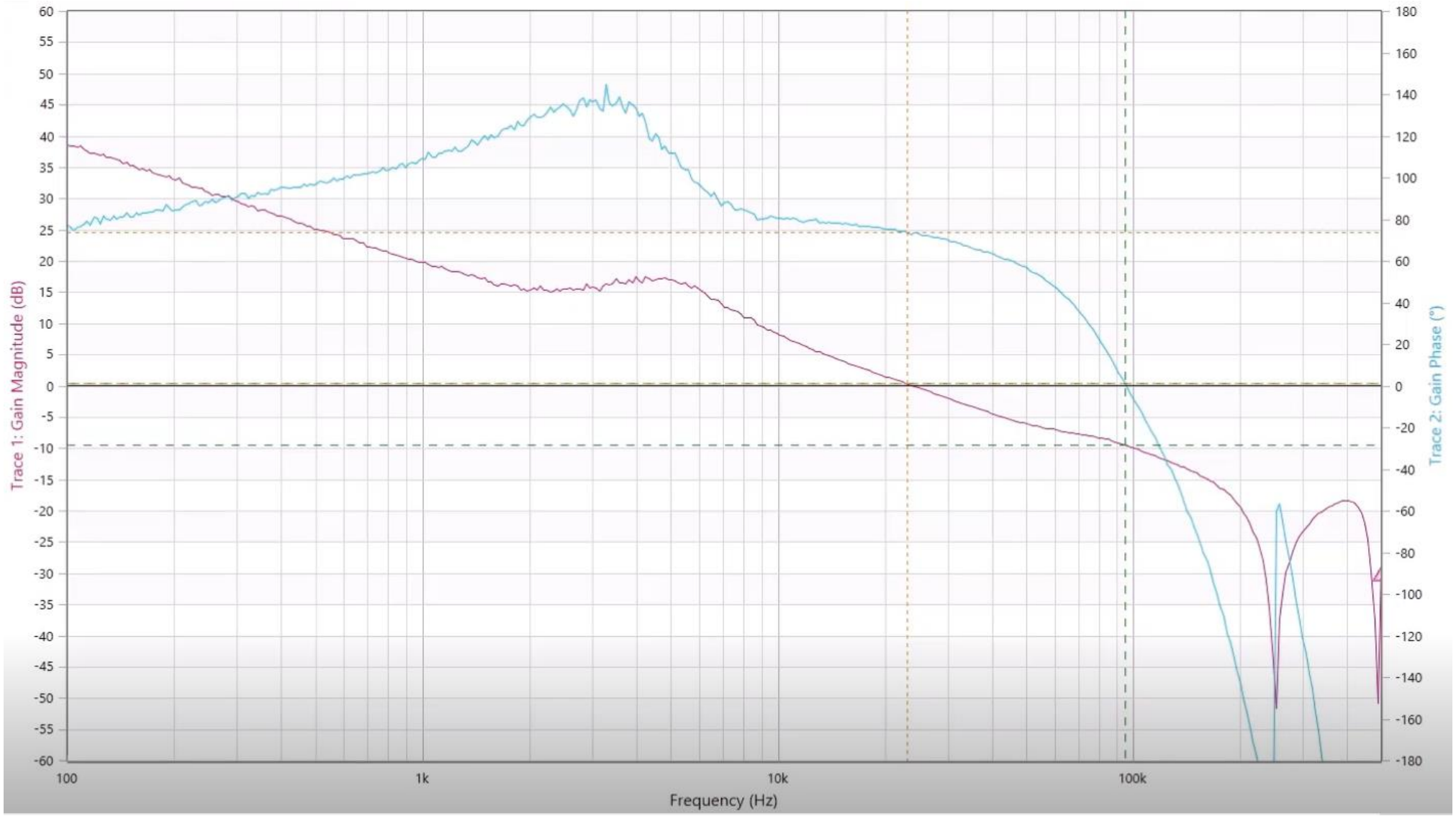
  - If the Build was successful …
  - Or check error message and fix what's broken

- **Program the Project**



  - When programming was successful, the red LED below dsPIC should blink

Bode Plot Results



MICROCHIP

Agenda

Digital Power Supply from Scratch in 60 Minutes*

*: Curious to see how this is playing out ….

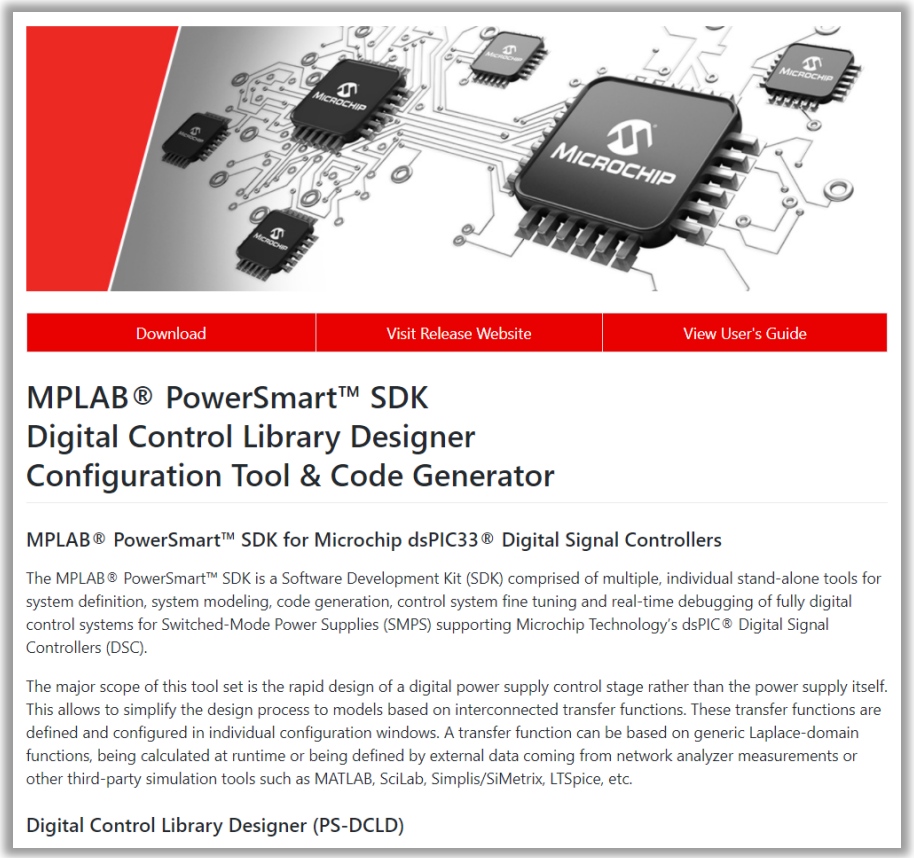
- Overview Digital Power Starter Kit 3 (DPSK3)
- Requirements and Scope
- Hands-On
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB® PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start and Error Handling State Machine to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- **Q&A**

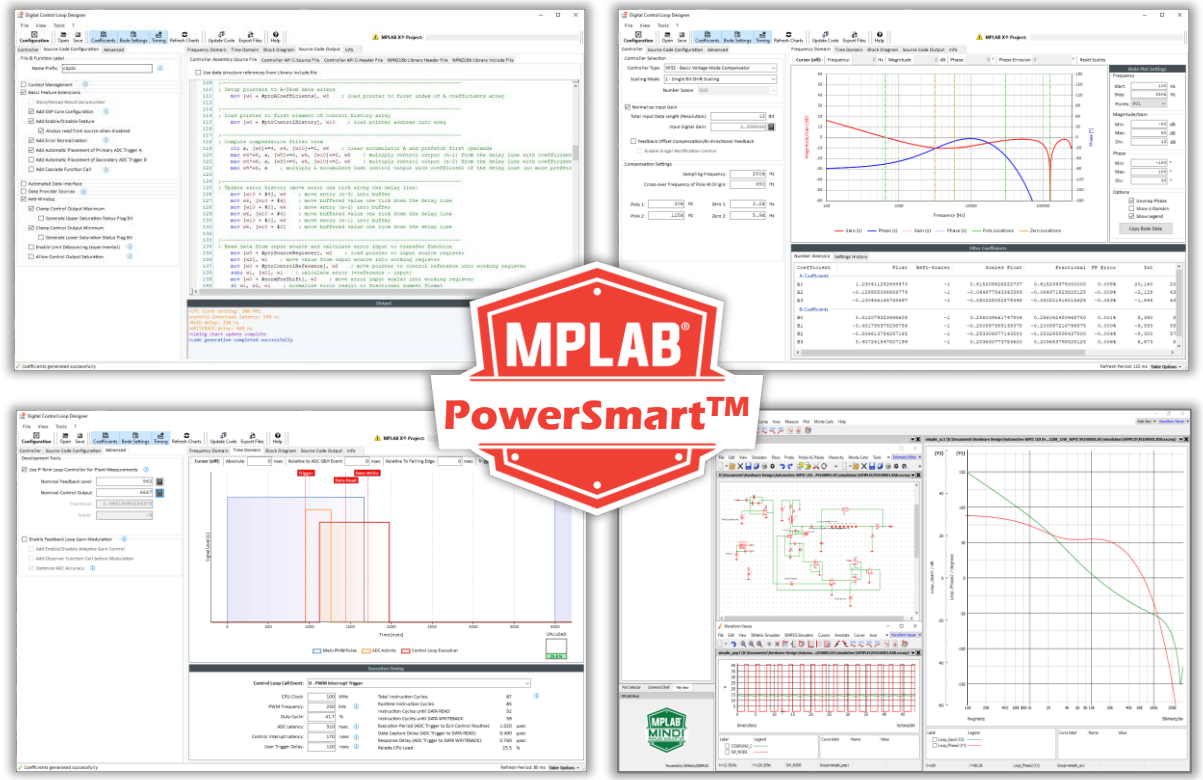
Appendix

MPLAB® PowerSmart™ SDK *Beta Pre-Release Version*

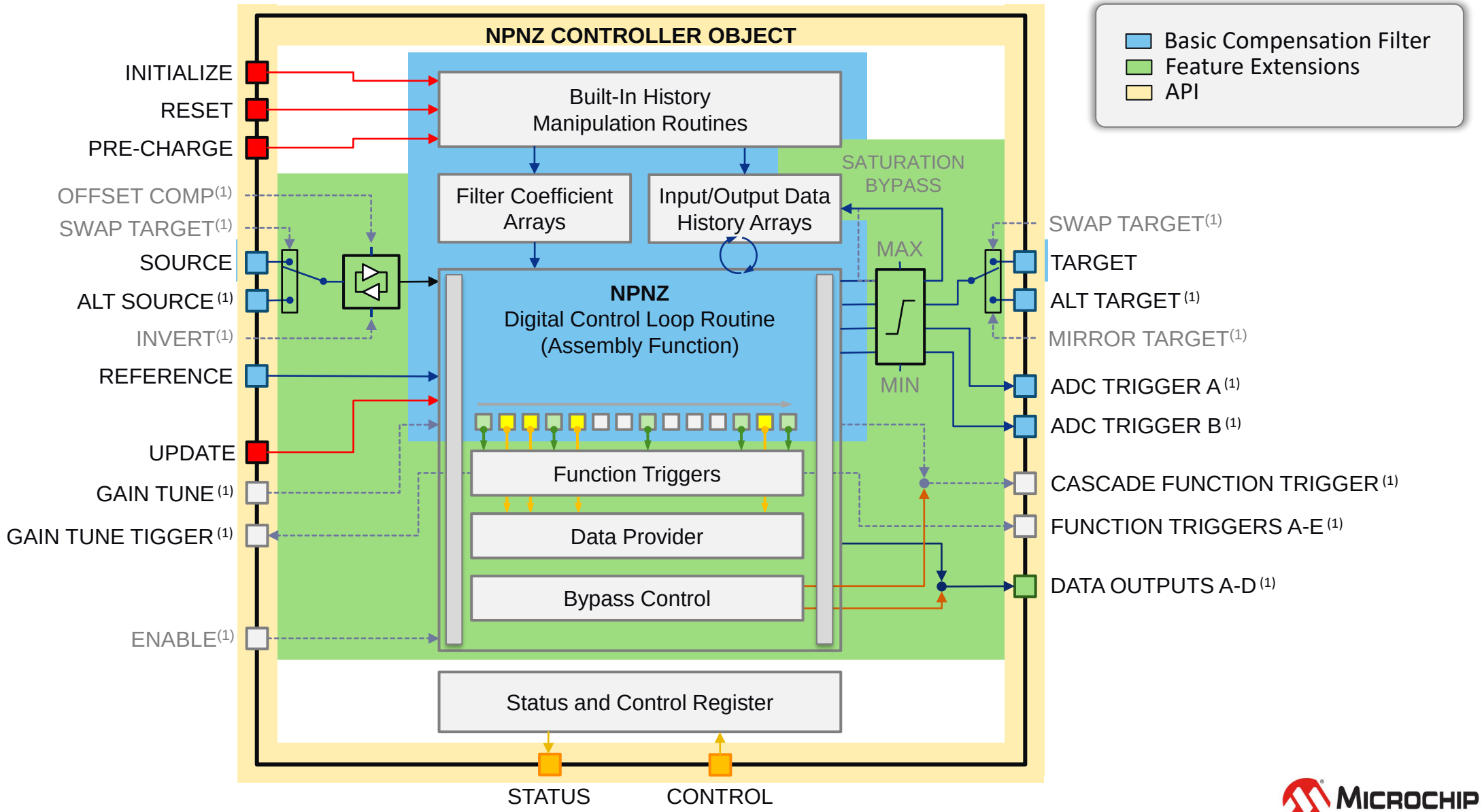
Digital Control Library Designer (PS-DCLD)

https://microchip-pic-avr-tools.github.io/powersmart-dcld/

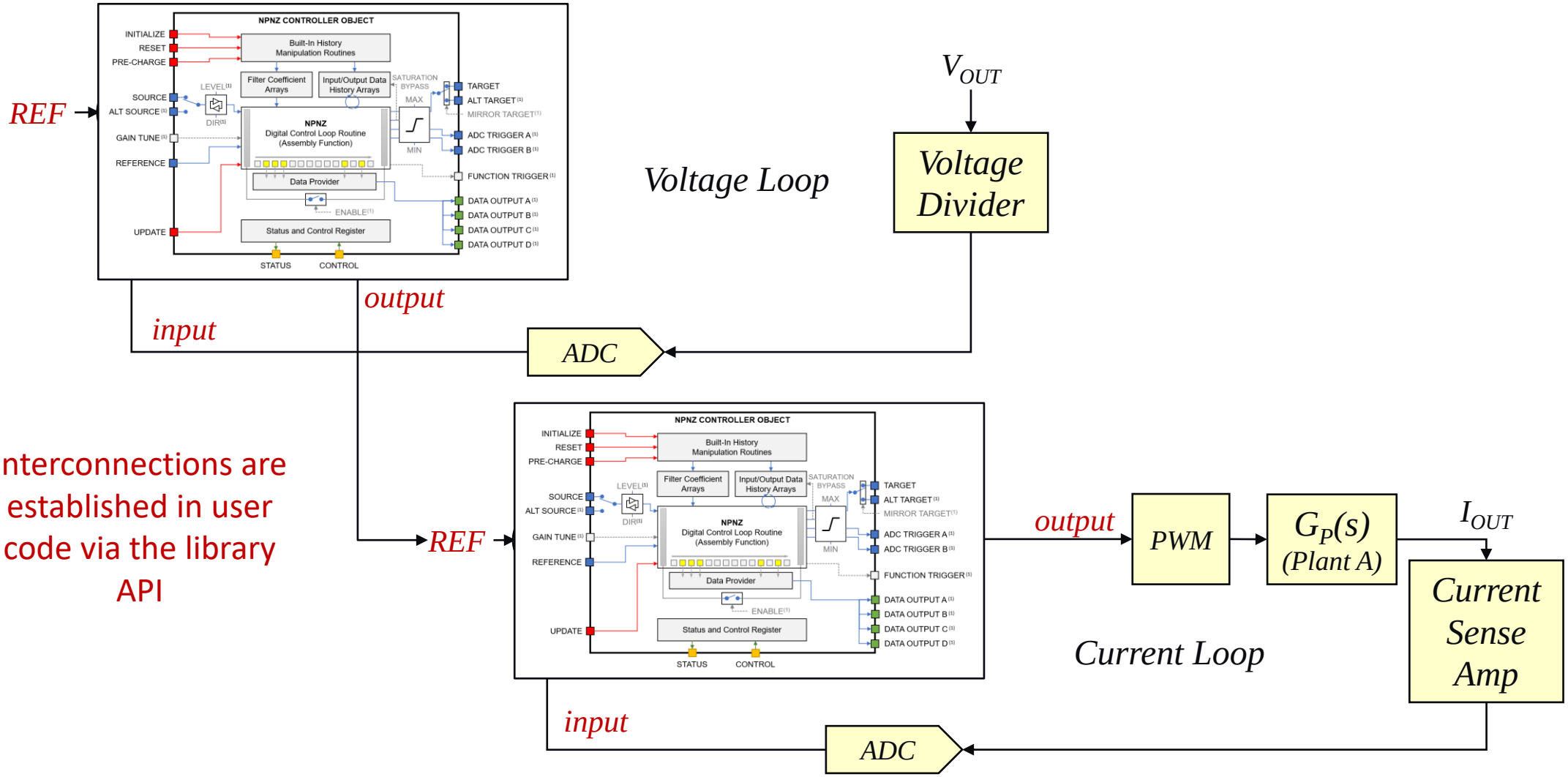




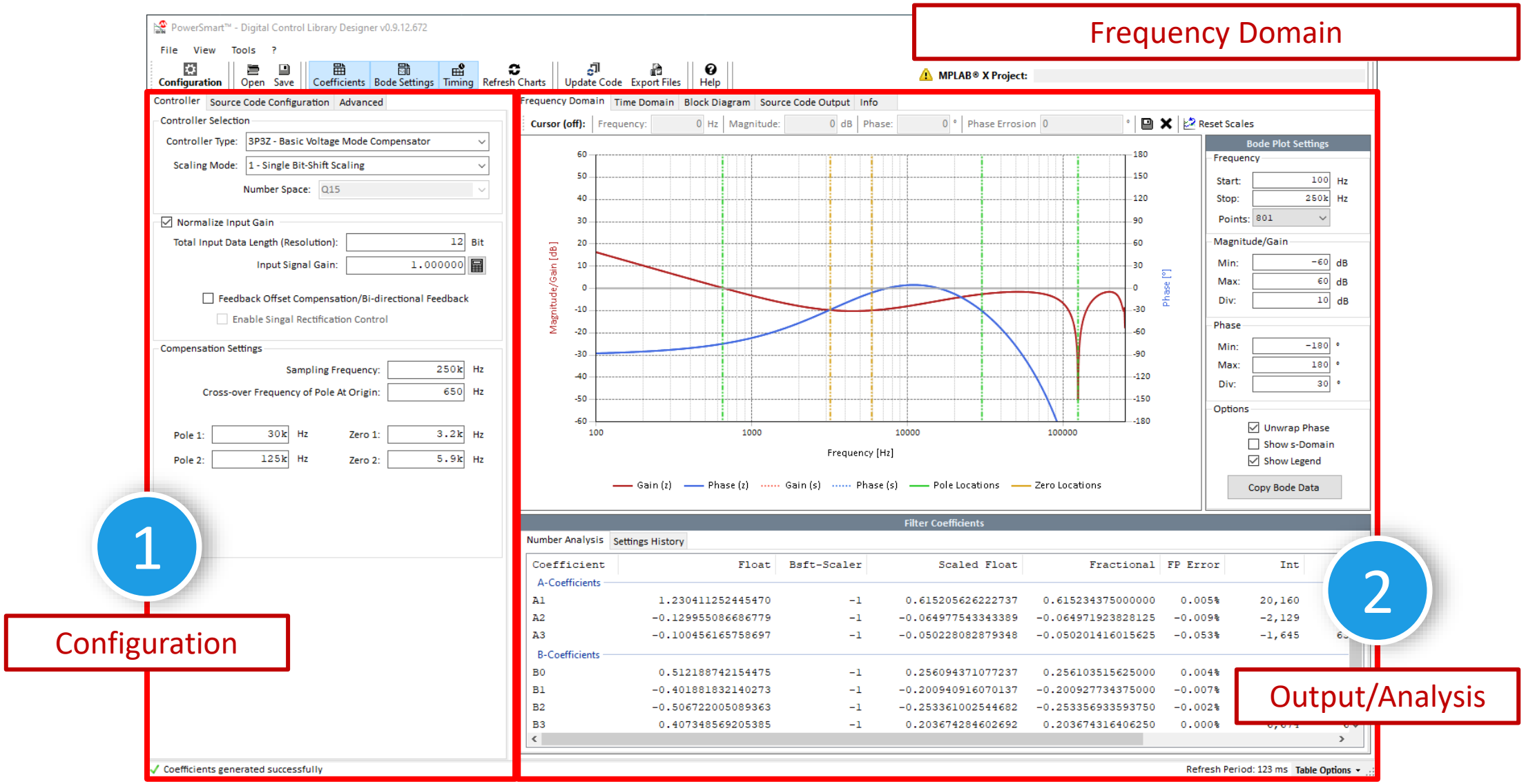
PS-DCLD Feedback Loop Block Diagram



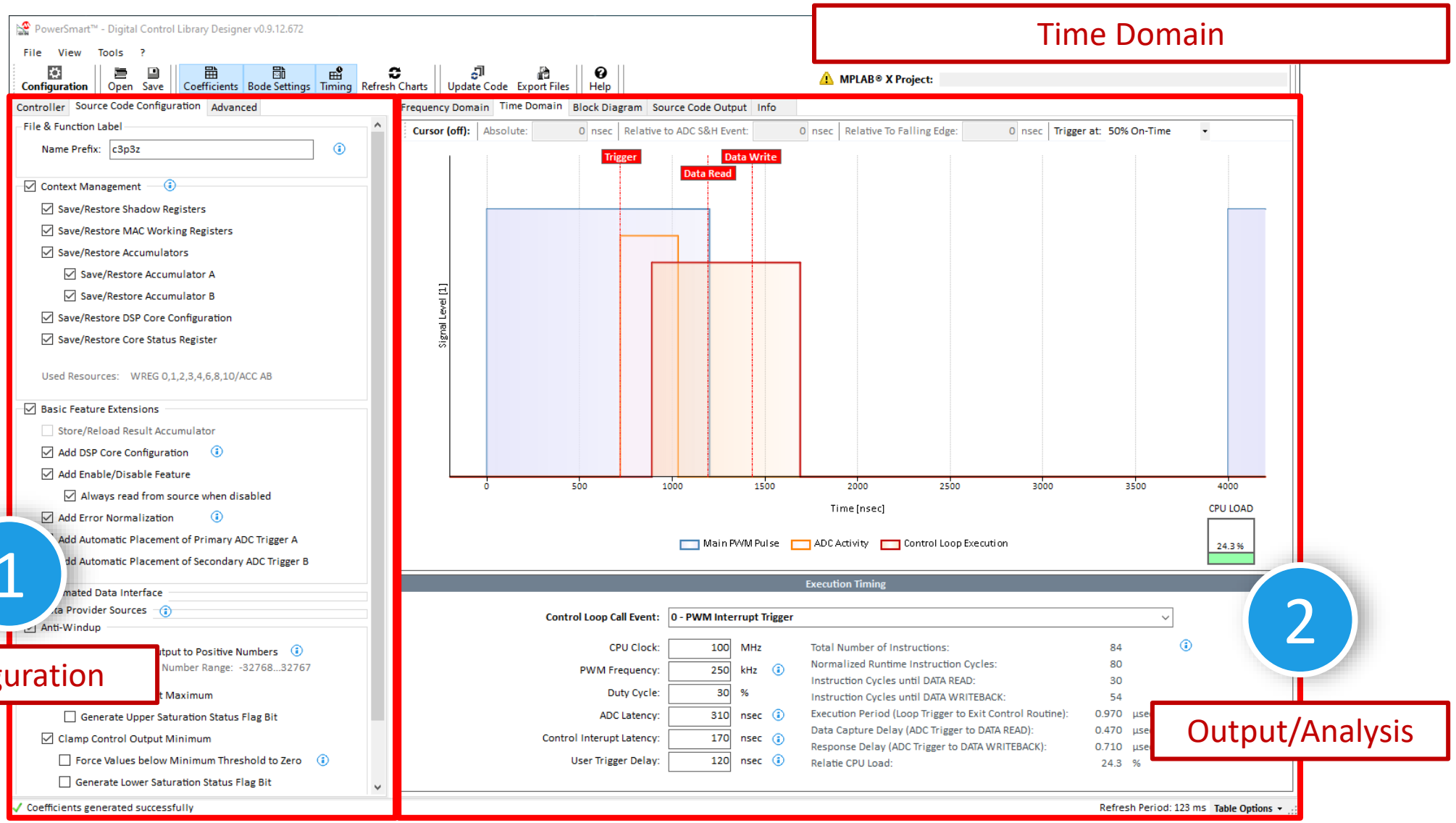
Using DCLD in Multi-Loop Systems



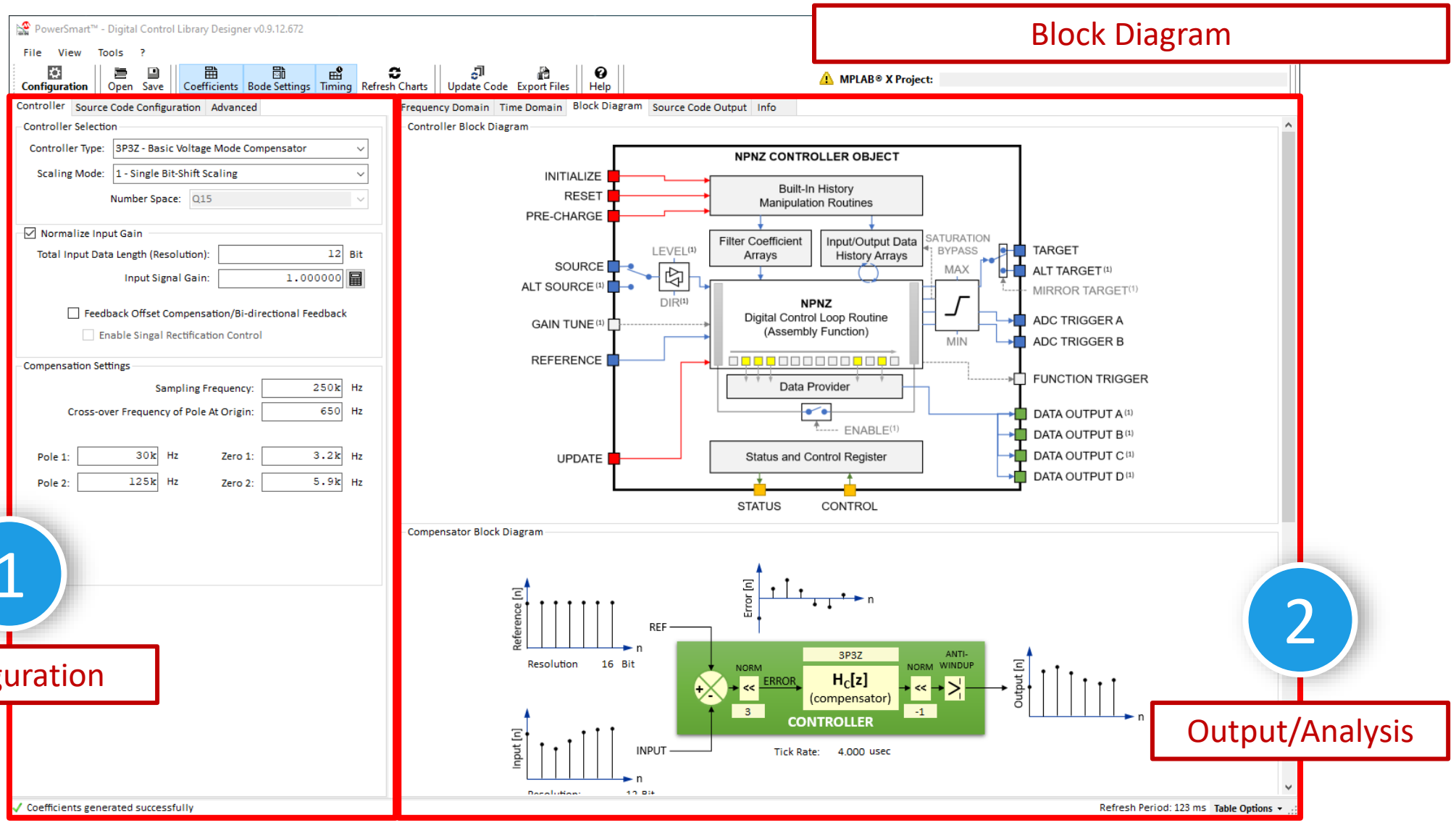
PowerSmart™ DCLD Main Window Overview



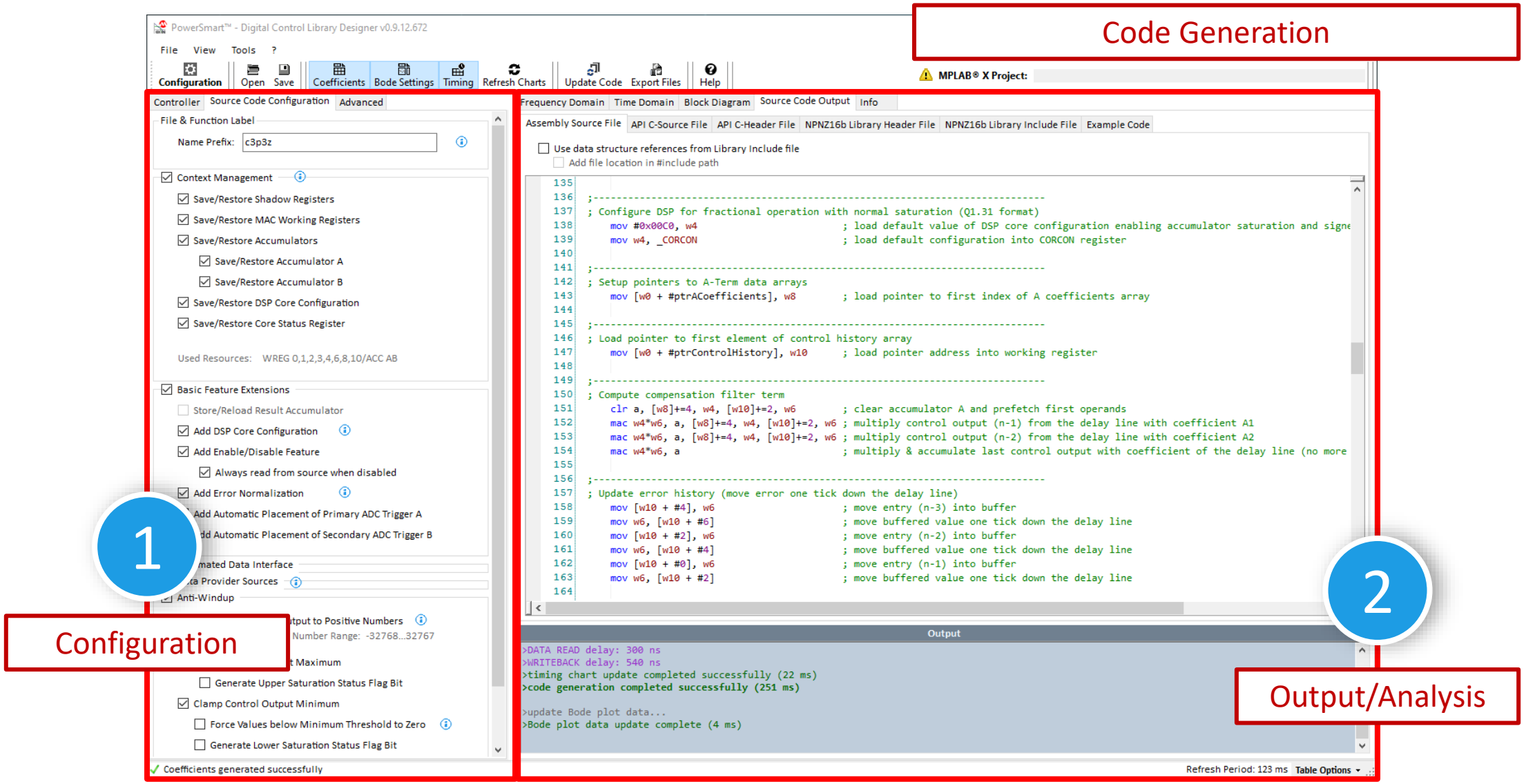
PowerSmart™ DCLD Main Window Overview



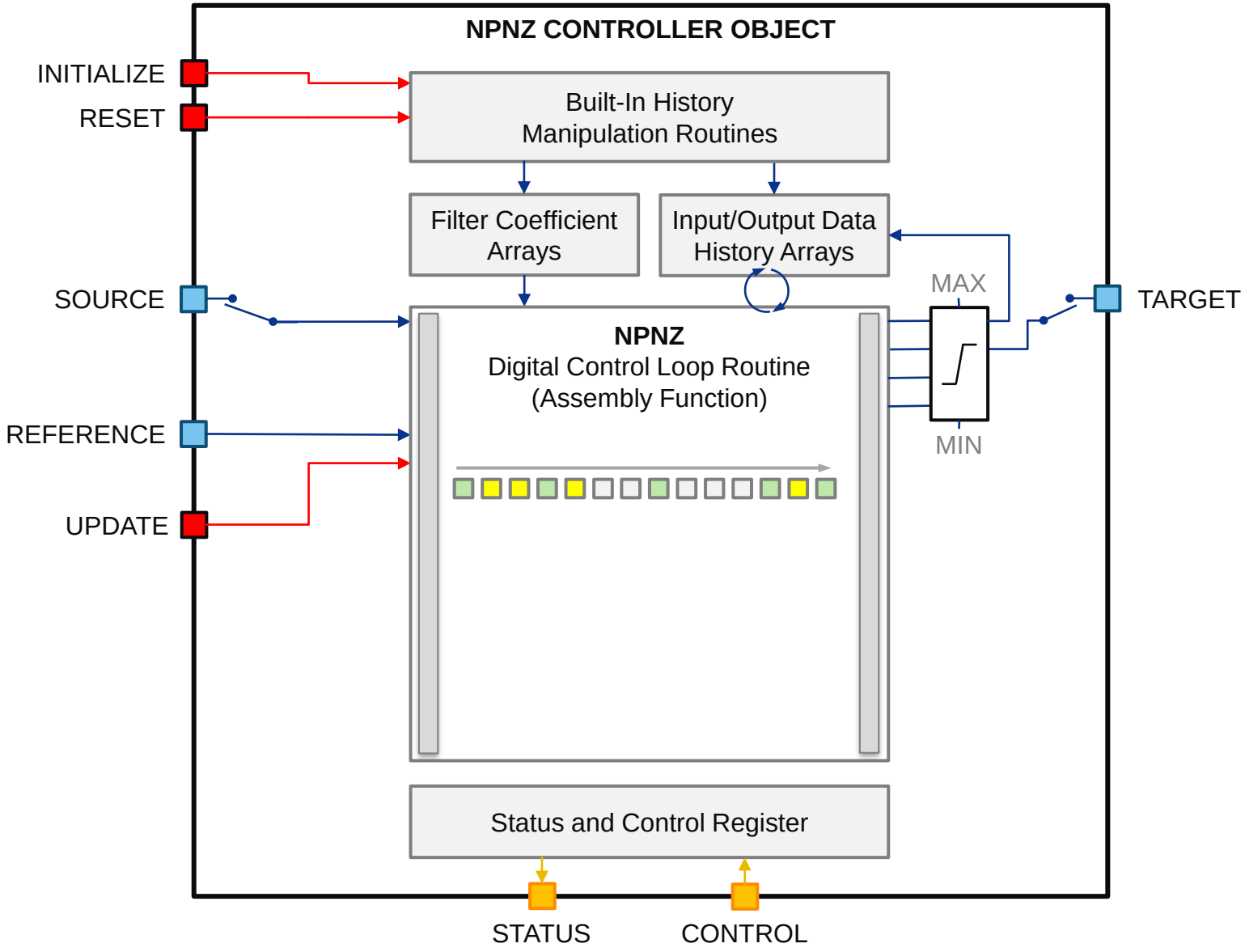
PowerSmart™ DCLD Main Window Overview



PowerSmart™ DCLD Main Window Overview



Buck Example Configuration



MICROCHIP

Let’s continue…

Remaining Time until Controller Design Completion: 21 min

MCC Configuration and
Initial Firmware Test

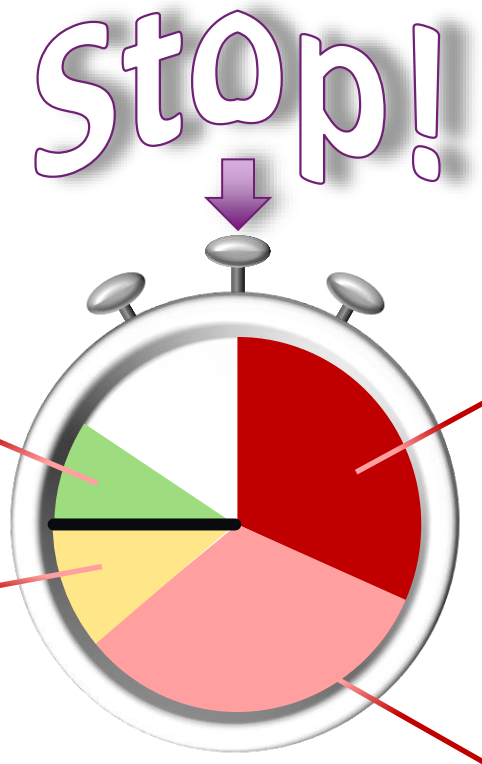
Total time elapsed: 24 min

Control Configuration using MPLAB® PowerSmart™

Building Up a Digital Power Supply
from Scratch in 45 Minutes

Design Time Review

Remaining Time until Controller Design Completion: 0 min



PS-DCLD Configuration of optimized 4P4Z Voltage Loop

Total time elapsed: **51 min**

PS-DCLD Configuration of 3P3Z Voltage Loop

Total time elapsed: **45 min**

MCC Configuration and Initial Firmware Test

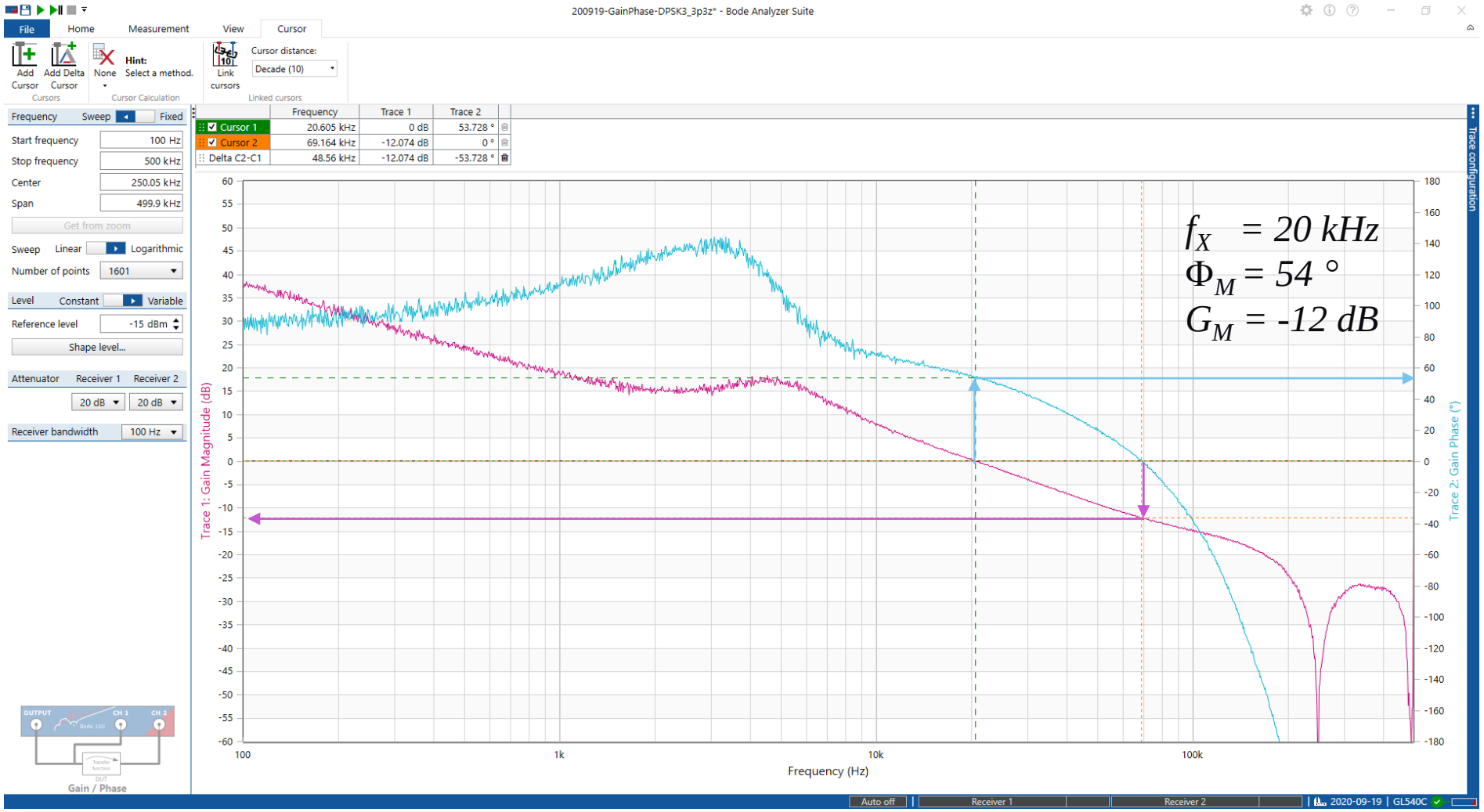
Time to setup device: **24 min**

PS-DCLD Configuration for Plant Measurement

Time to set up feedback loop: **39 min**

Final Test Result

V_{IN}=9 V, V_{OUT}= 3.3 V, I_{LOAD} = 50%

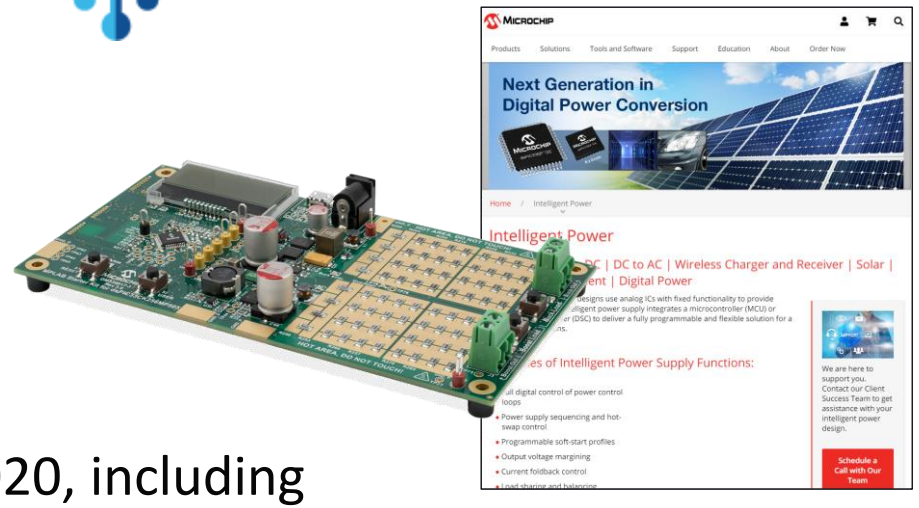


MICROCHIP

Digital Power

- **Getting Started in Digital Power**
  - Intelligent Power Design Center:
    https://www.microchip.com/power
- **How-2 Starter Kits**
  - Digital Power Starter Kit 3 (Part-No. DM330017-3):
    https://www.microchip.com/dm330017-3
- **Training:**
  - Microchip University (Virtual Training Platform):
    https://secure.microchip.com/mu

MICROCHIP UNIVERSITY

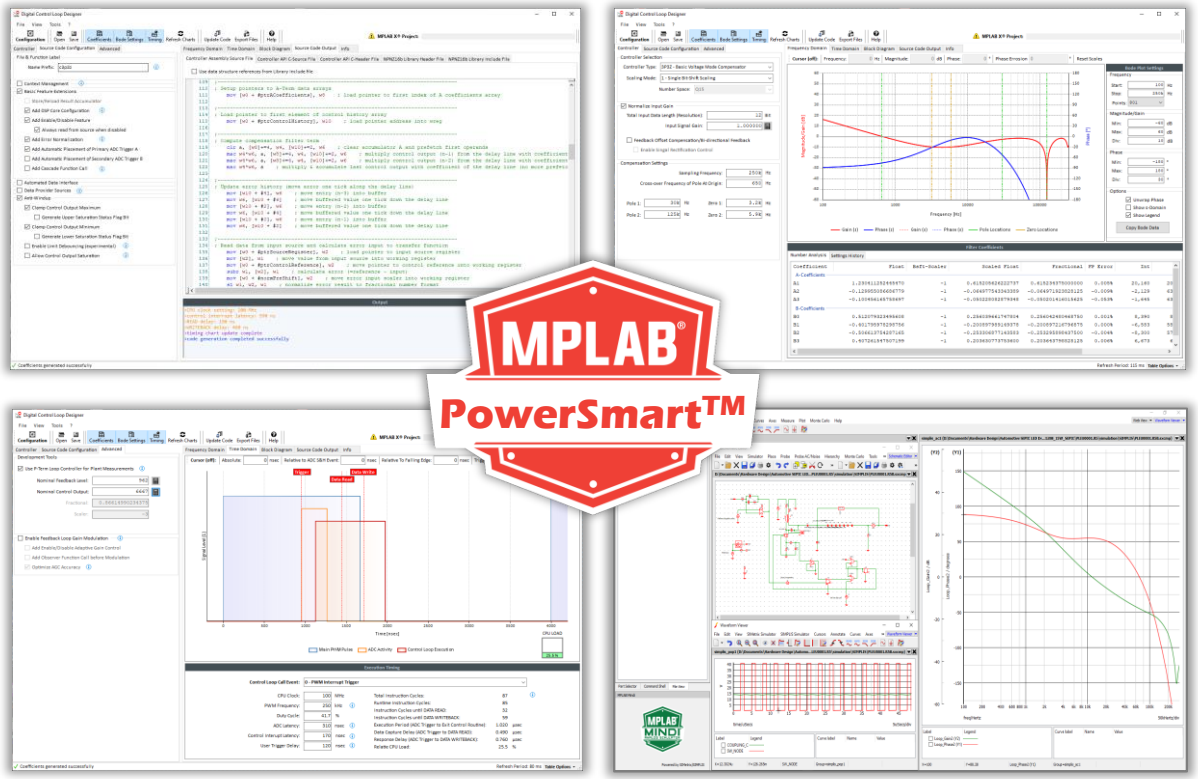


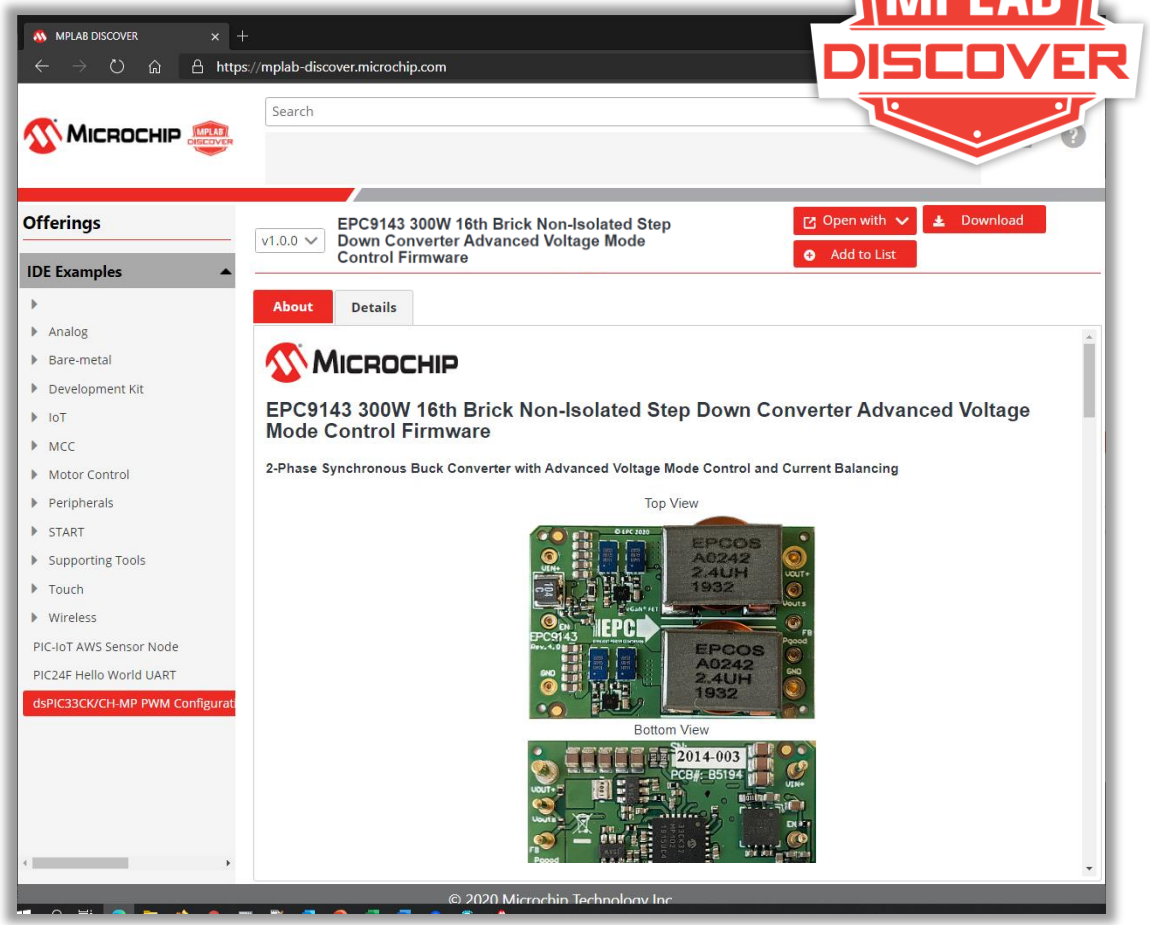
Please note:
All Face-2-Face workshops have been suspended in early 2020, including the well-known MASTERs conferences usually conducted in 9 nations around the globe. For the time being all available trainings have been moved to our new virtual training platform ***Microchip University***.

MICROCHIP

Digital Power Design Resources

- **Find hundreds of code examples and design tools on https://discover.microchip.com**





Agenda

Building Up a Digital Power Supply from Scratch in 45 Minutes

- Introduction EU PFG
- Overview Digital Power Starter Kit 3 (DPSK3)
- Live Demo:
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- **Q&A**

Questions and Answers

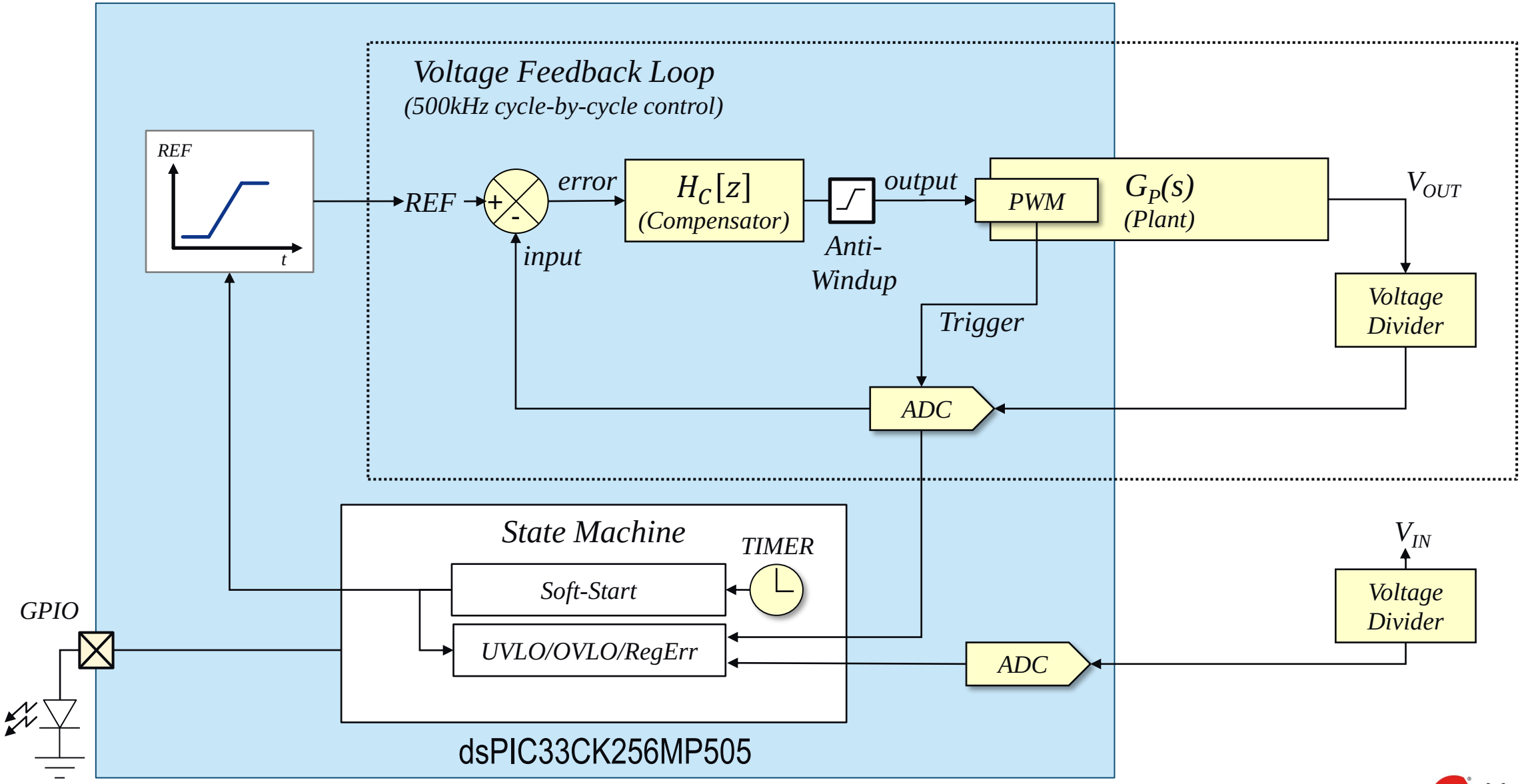
Appendix

Agenda

Building Up a Digital Power Supply from Scratch in 45 Minutes

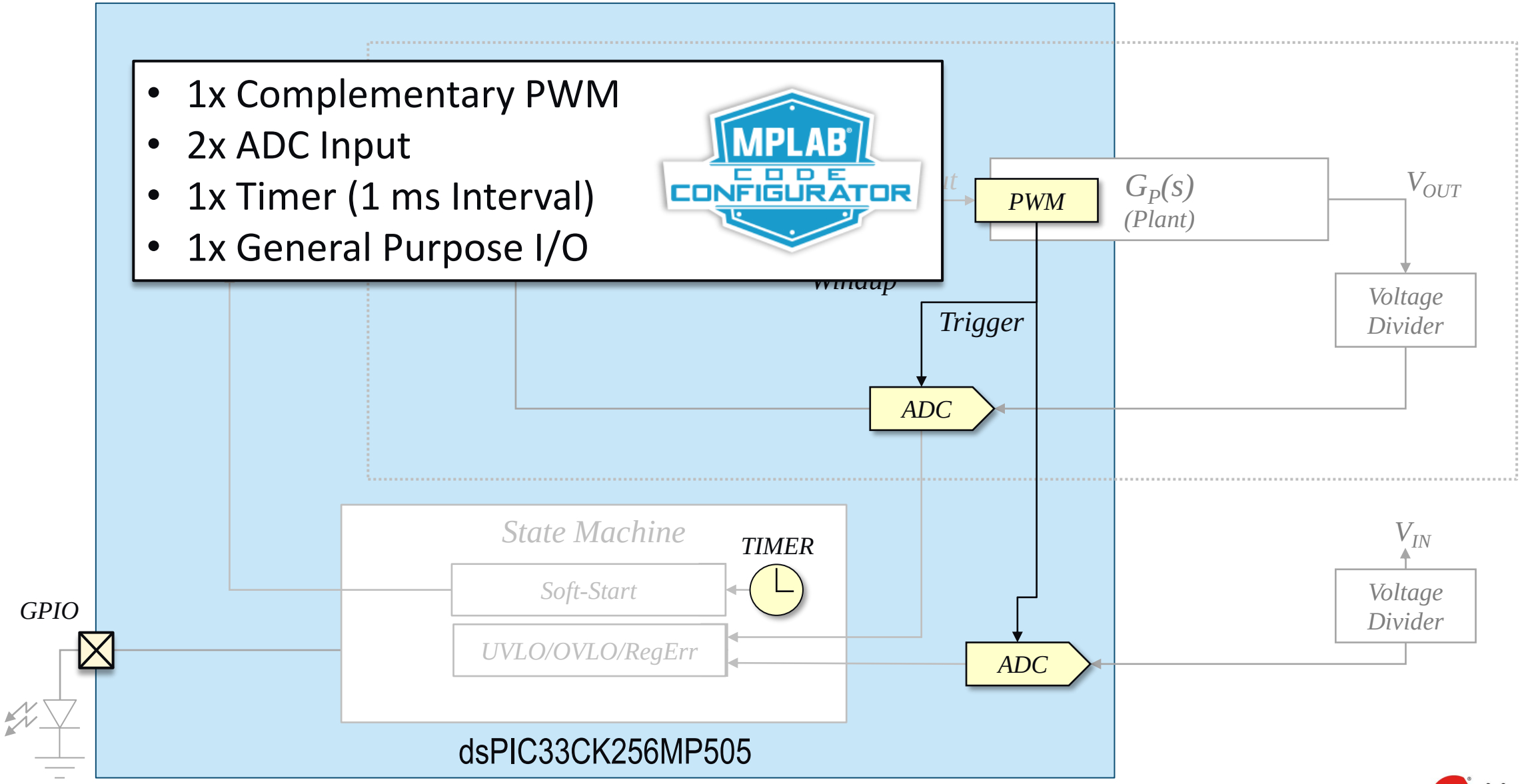
- Introduction EU PFG
- Overview Digital Power Starter Kit 3 (DPSK3)
- **Live Demo:**
  - Setting up the Project
  - Configuring Device and Peripherals
  - Using MPLAB PowerSmart™ to Build an Initial Loop
  - Adding Simple Soft Start to Project
  - Final Controller Selection and Adjustment
  - Loop Measurement & Optimization
- Q&A

Requirements



MICROCHIP

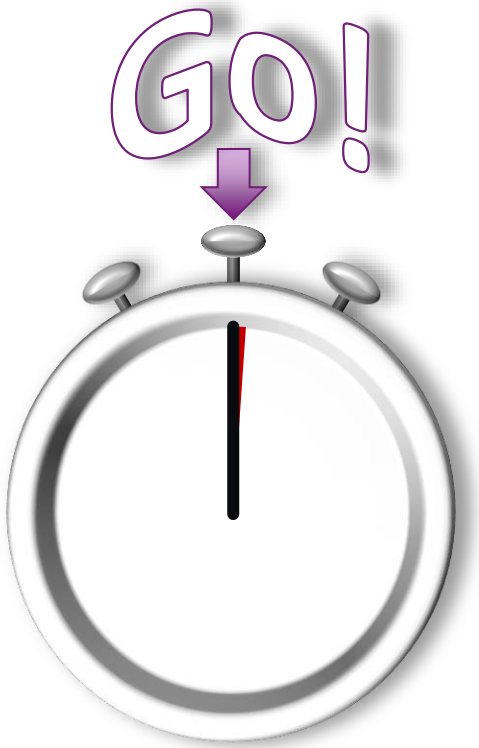
Requirements



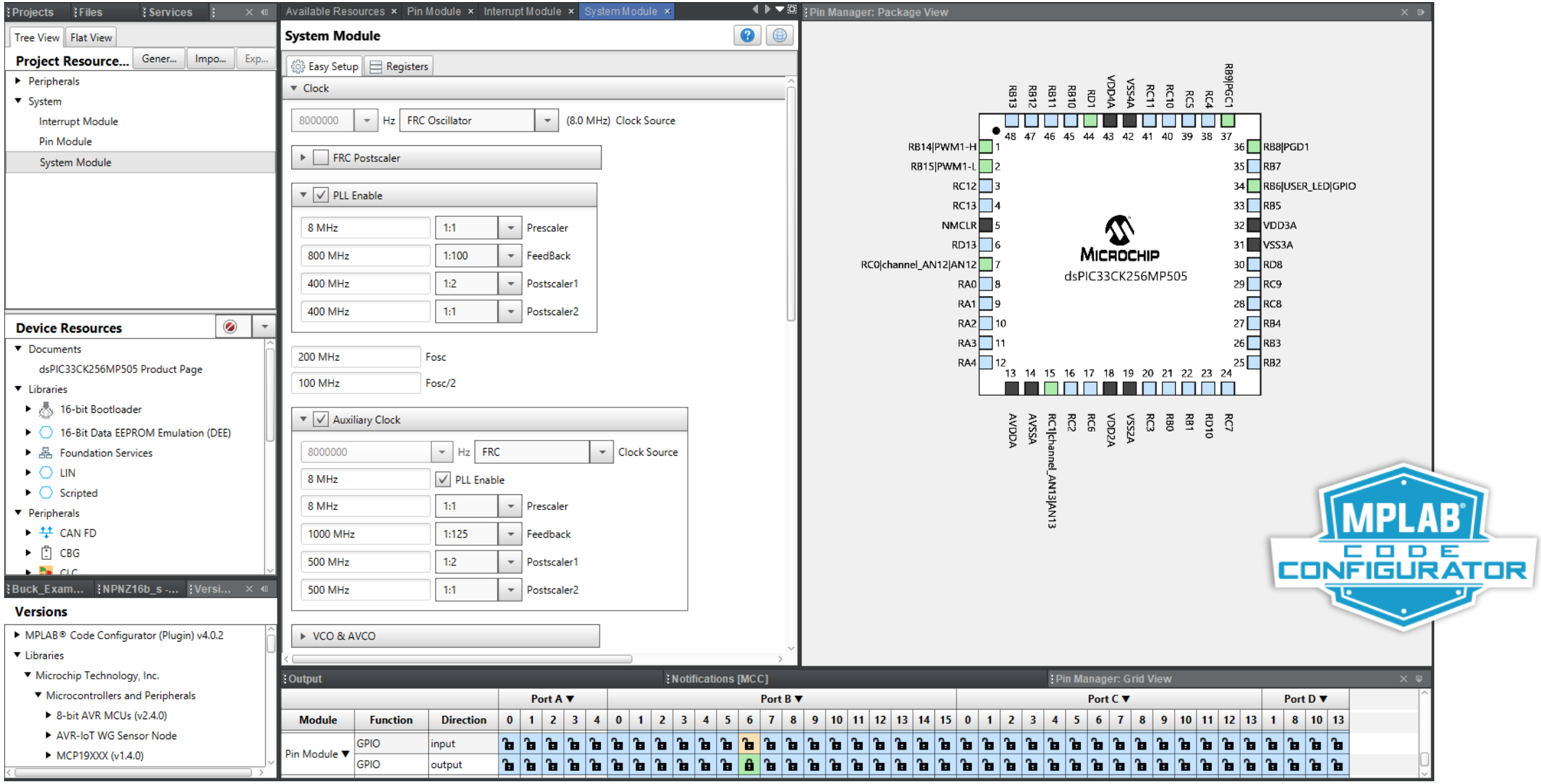
MICROCHIP

This is from where we start…

Remaining Time until Controller Design Completion: 45 min

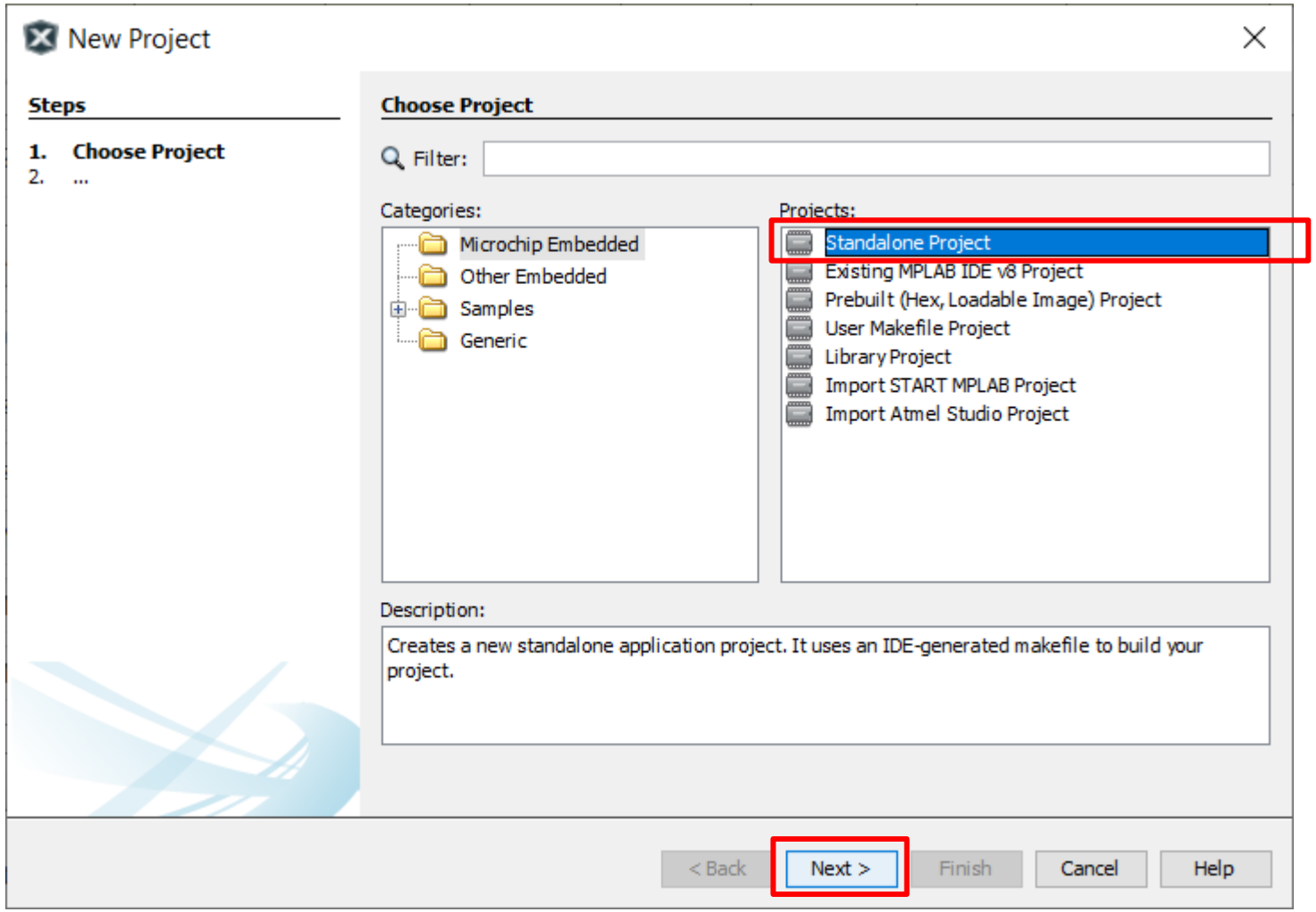


Open MCC in MPLAB® X IDE

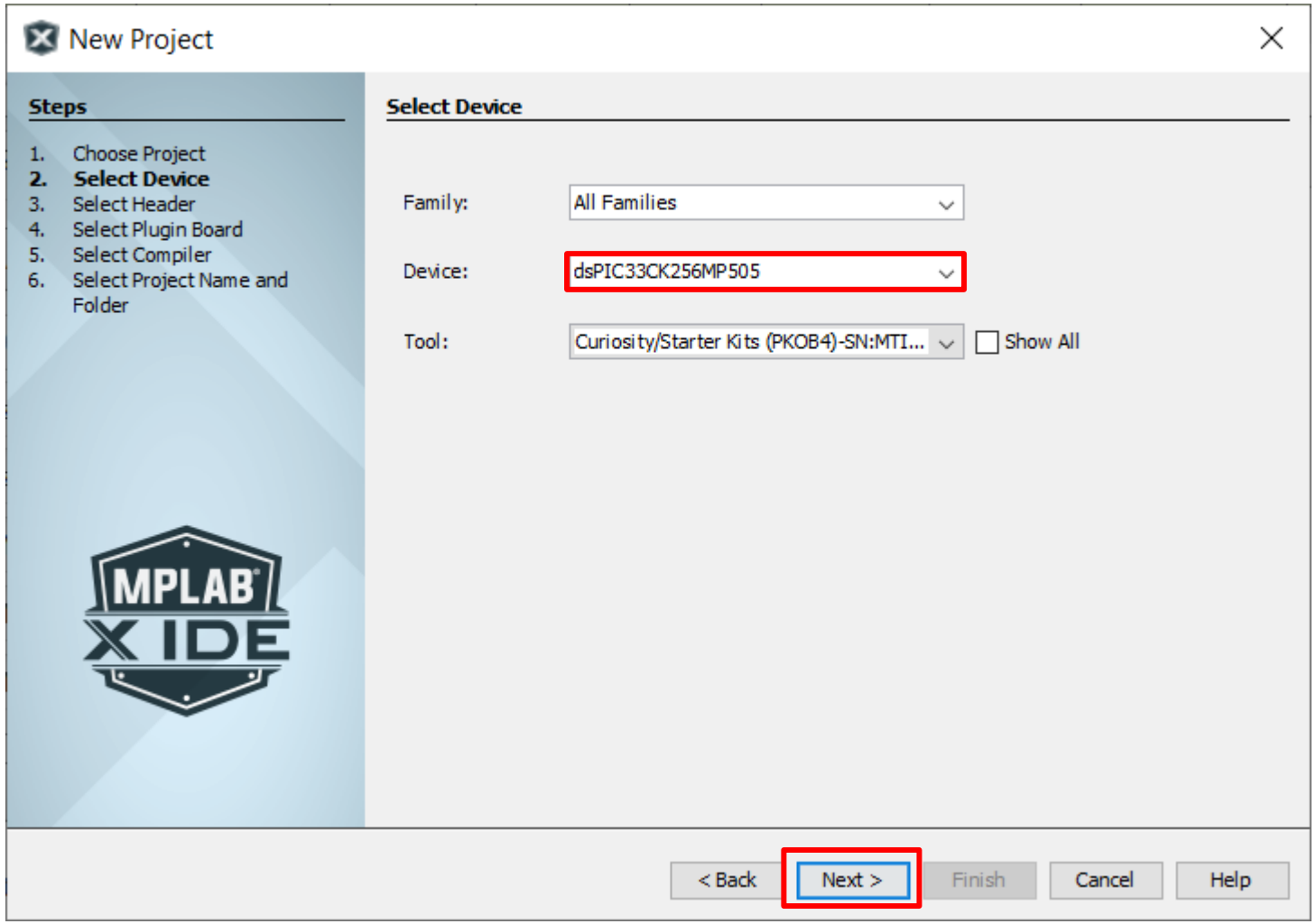


MICROCHIP

Creating the Project

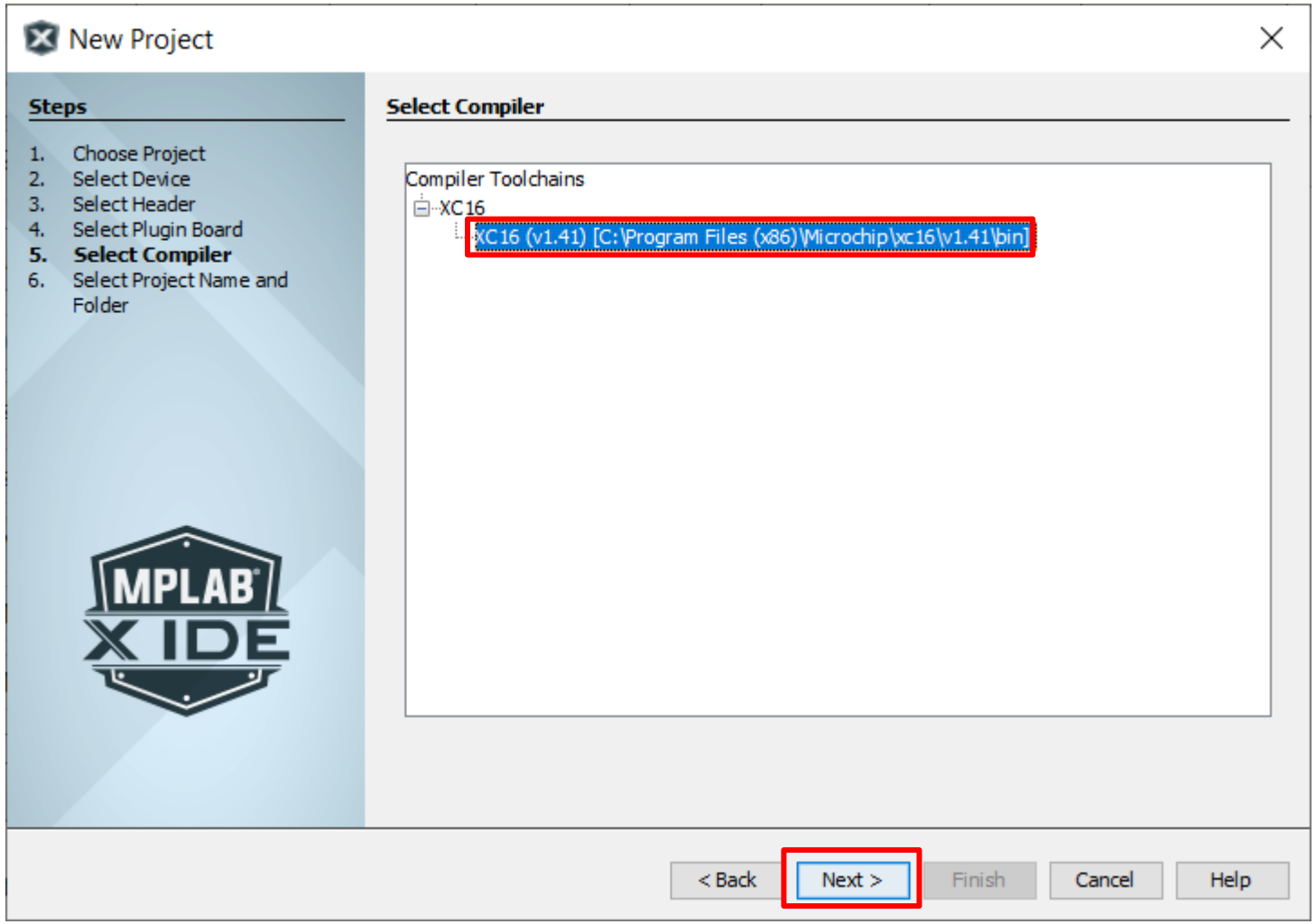


Creating the Project



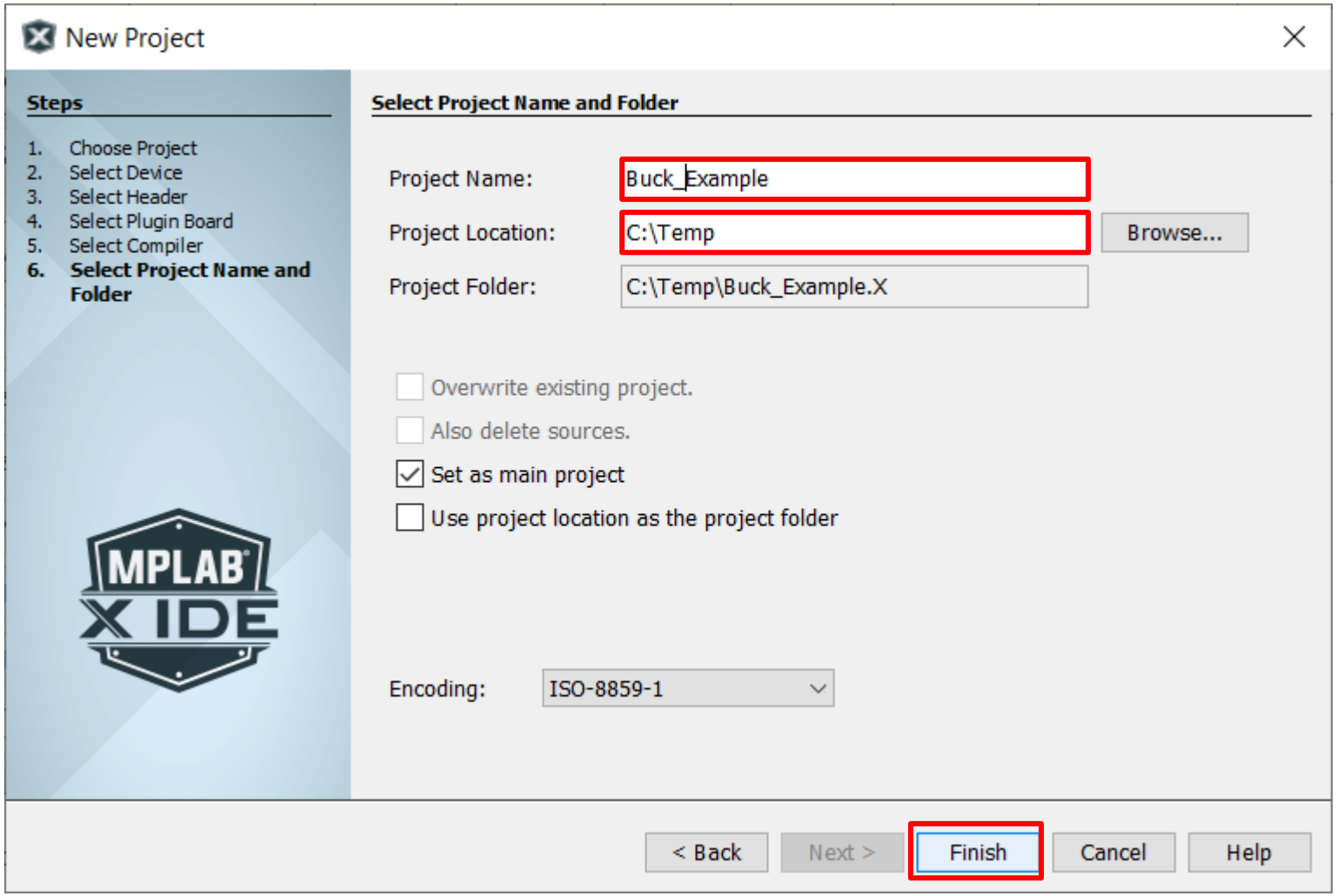
MICROCHIP

Creating the Project



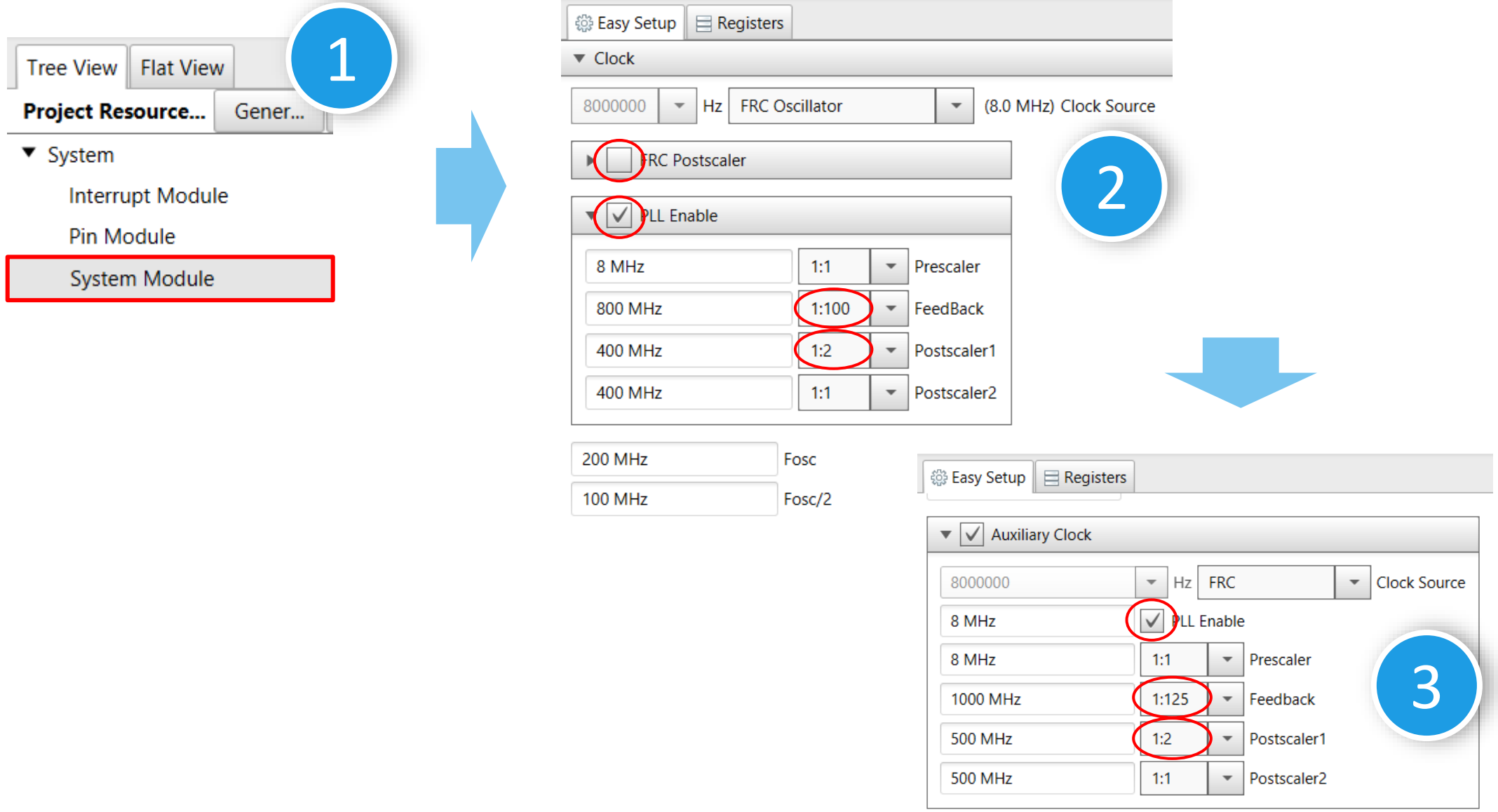
MICROCHIP

Creating the Project



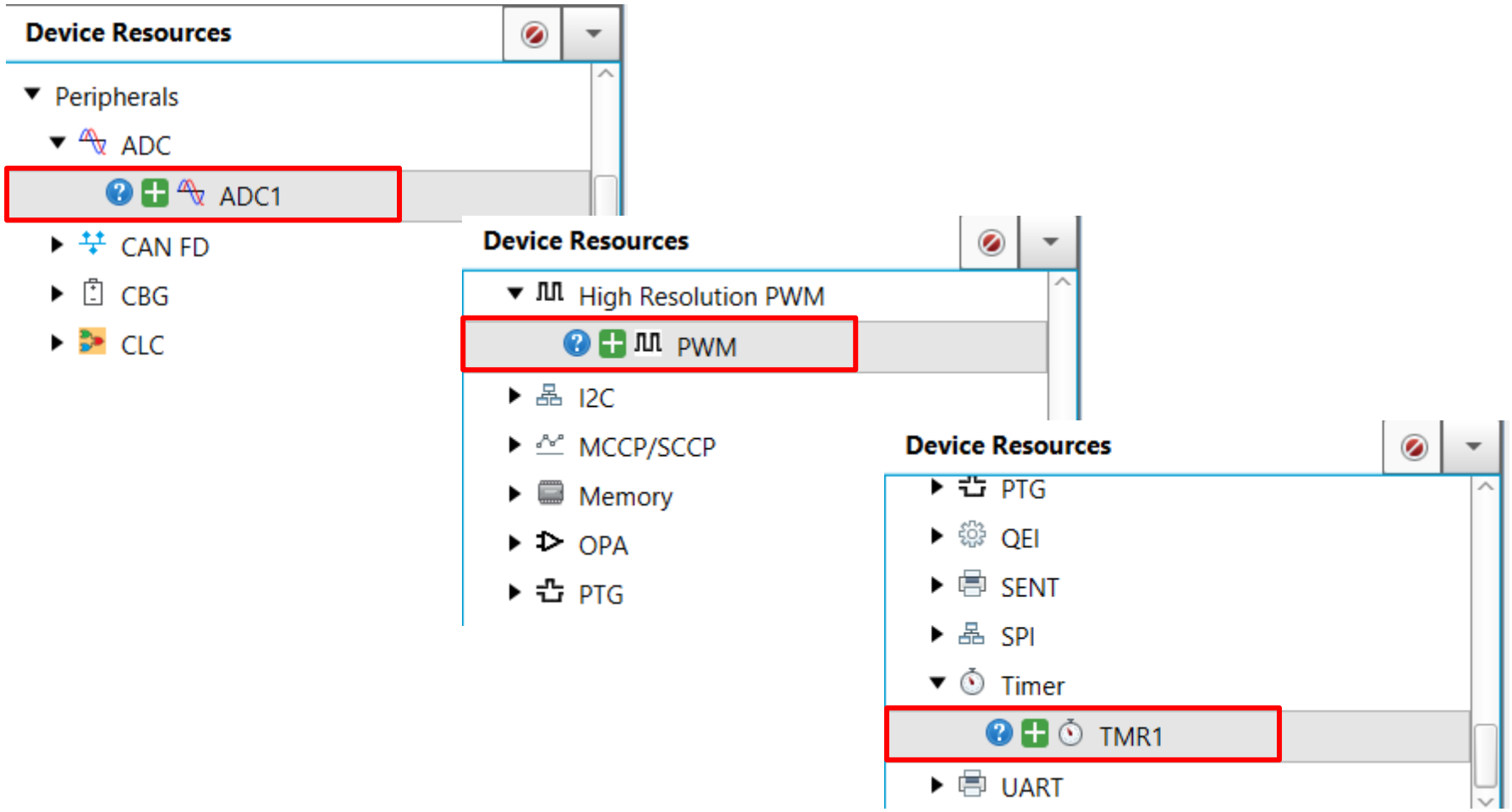
MICROCHIP

System Configuration in MCC



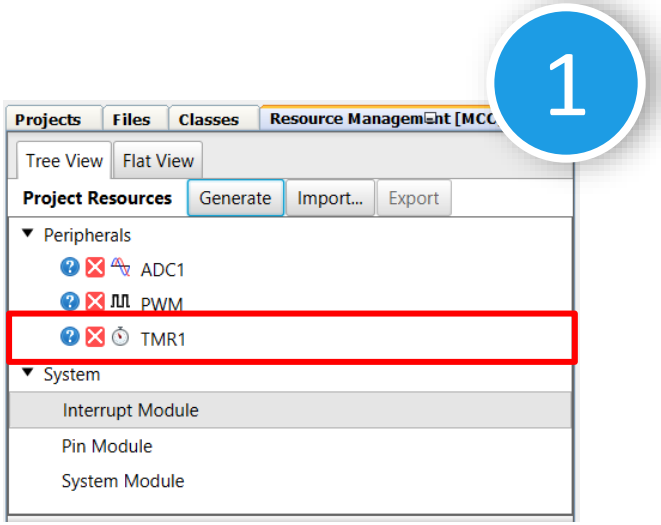
MICROCHIP

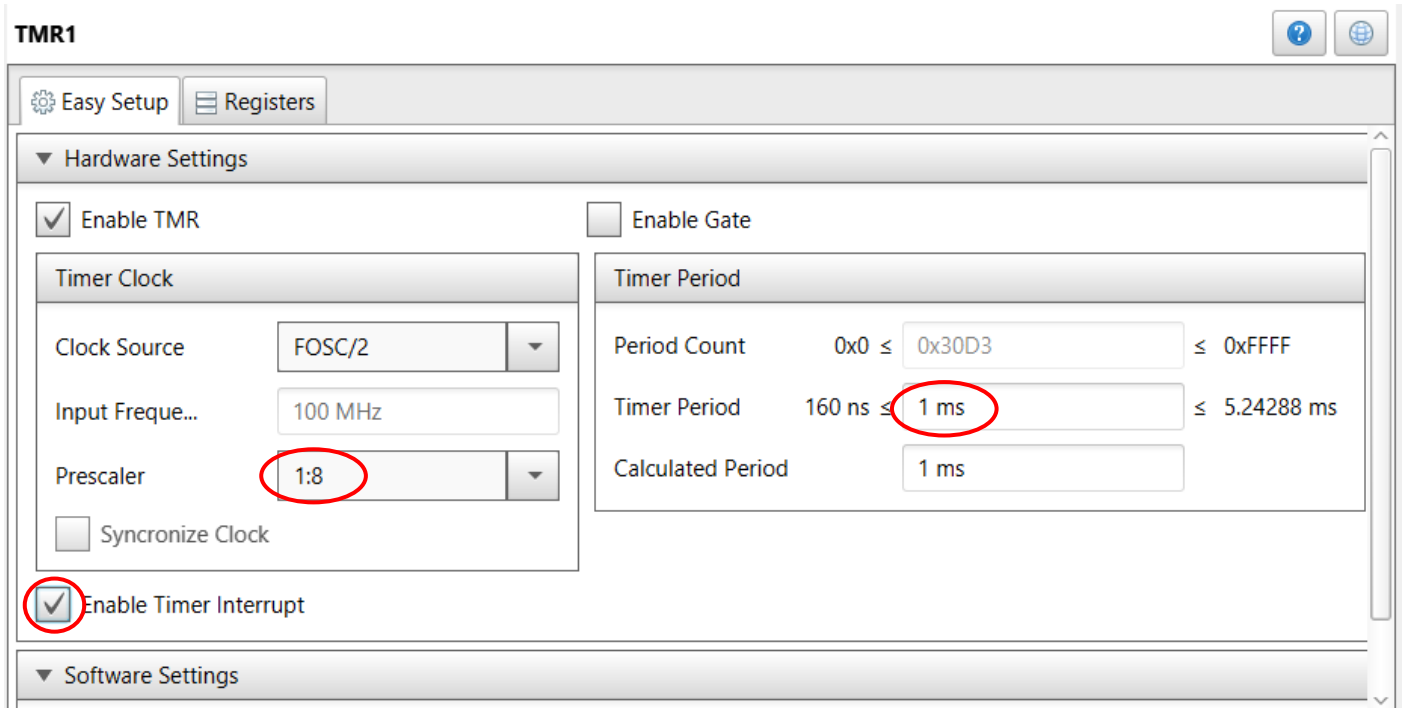
Peripherals Selection in MCC



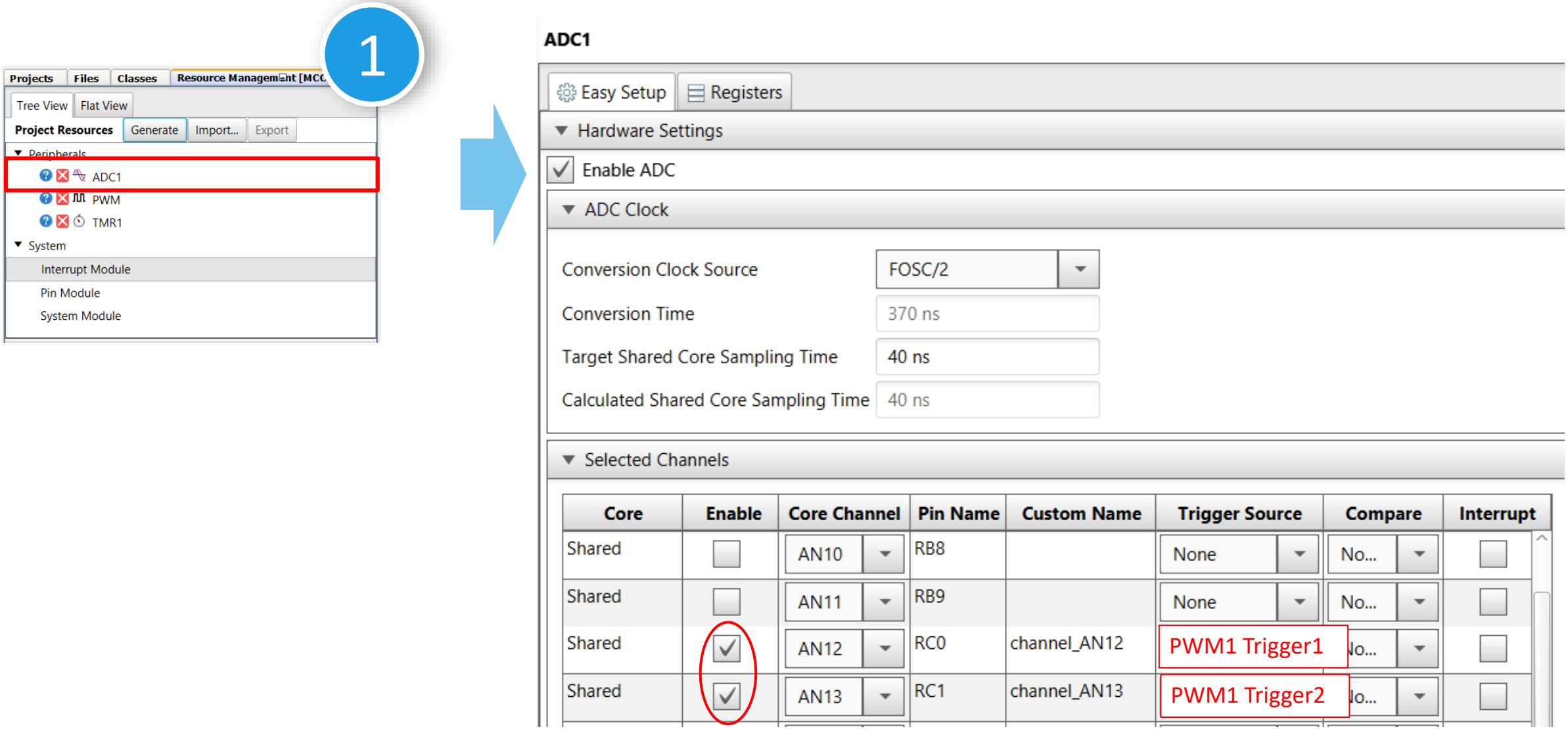
MICROCHIP

Timer1 Configuration in MCC



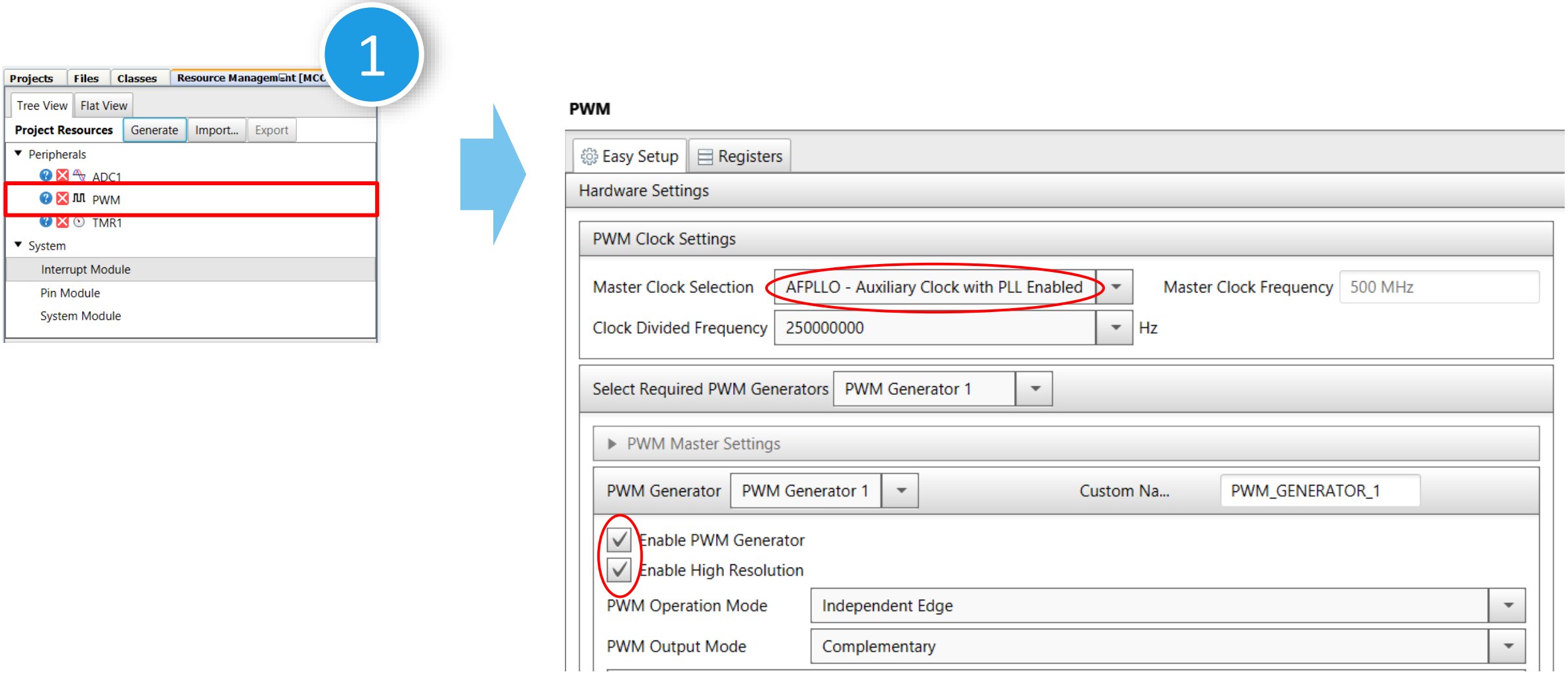


ADC Configuration in MCC

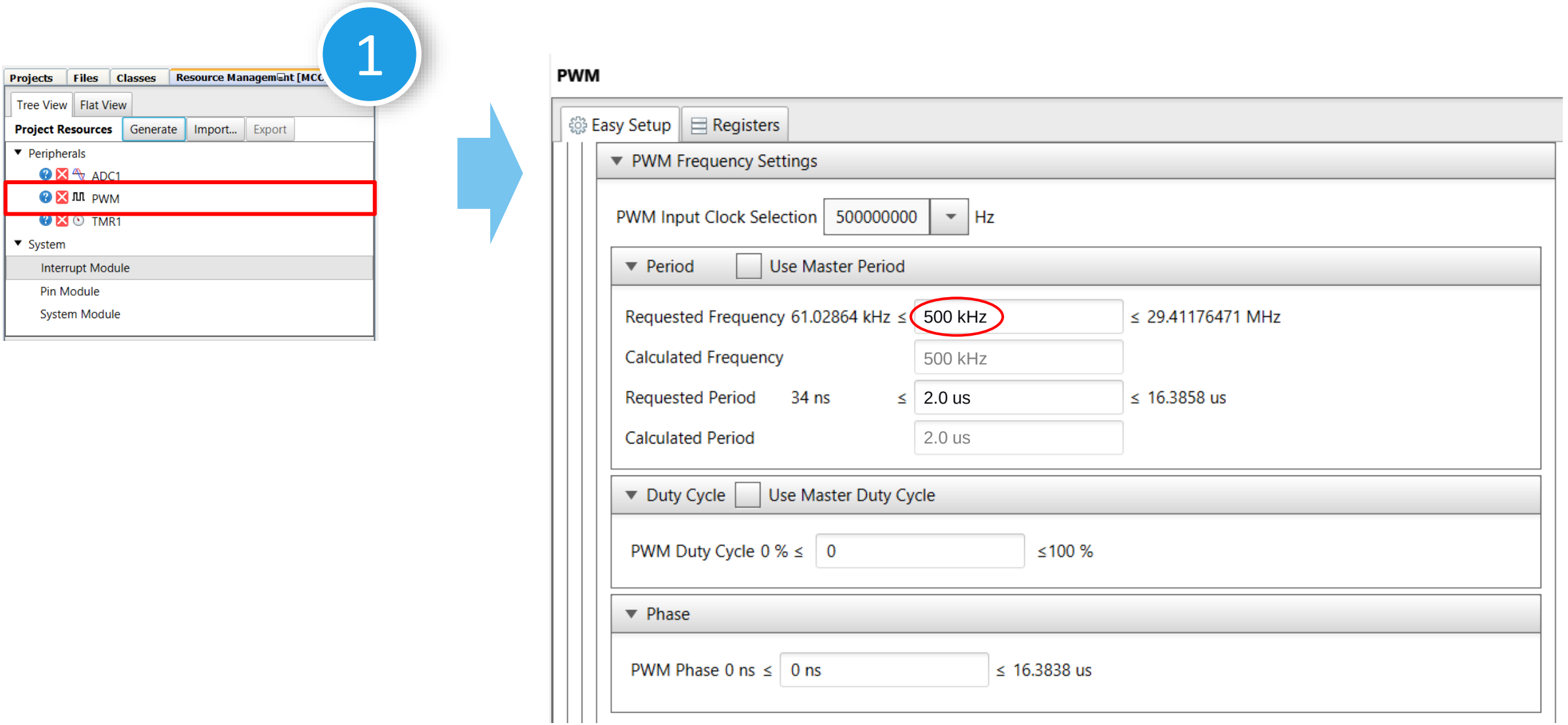


MICROCHIP

PWM Configuration in MCC (1/3)

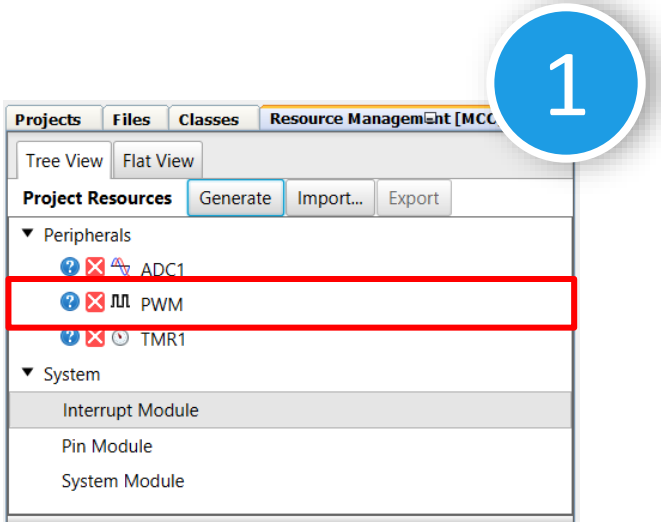


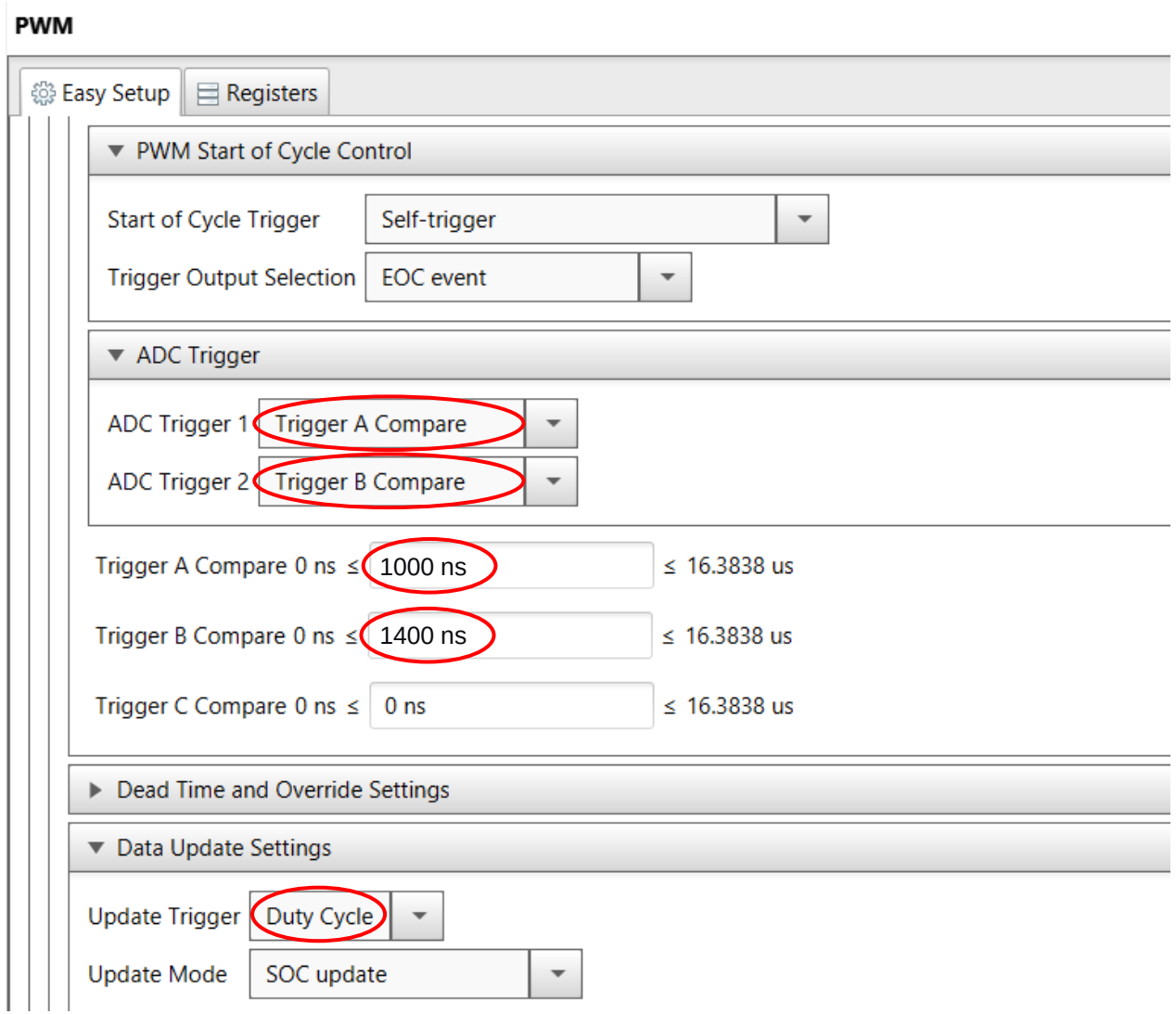
PWM Configuration in MCC (2/3)



MICROCHIP

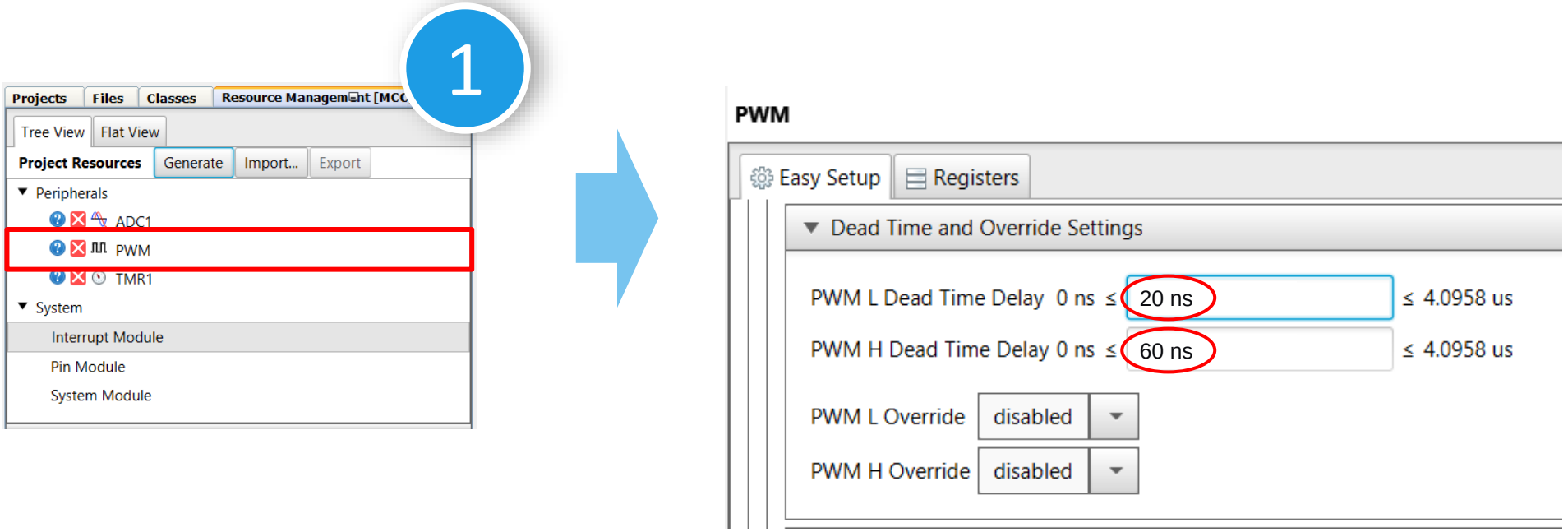
PWM Configuration in MCC (3/3)





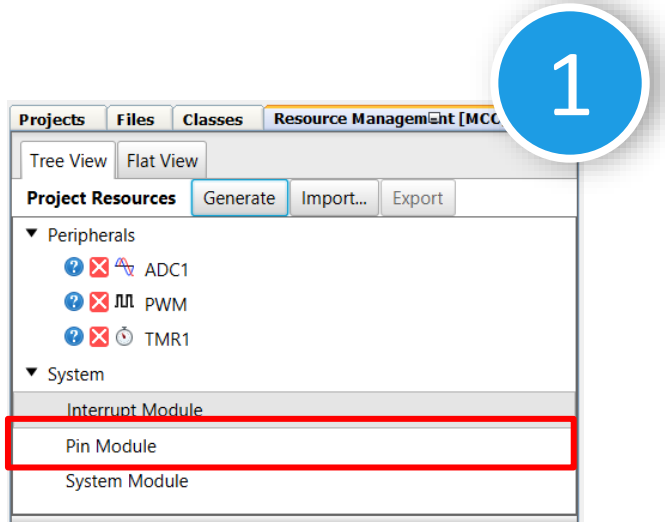
MICROCHIP

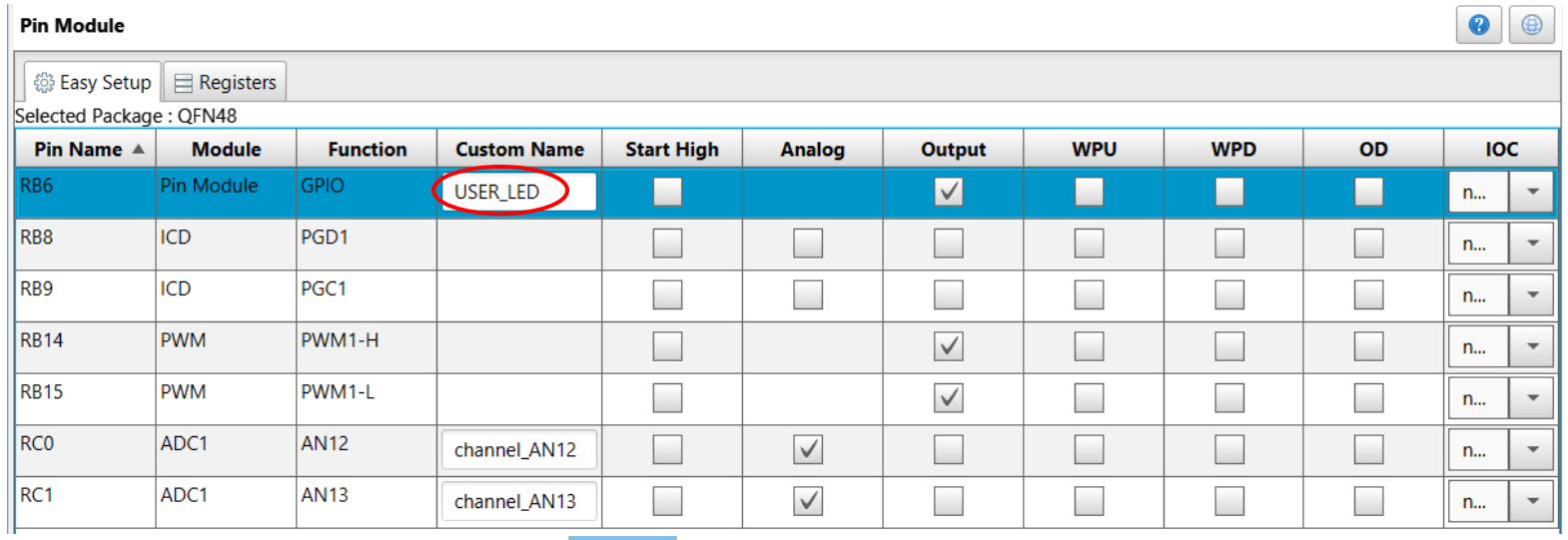
PWM Configuration in MCC (3/3)

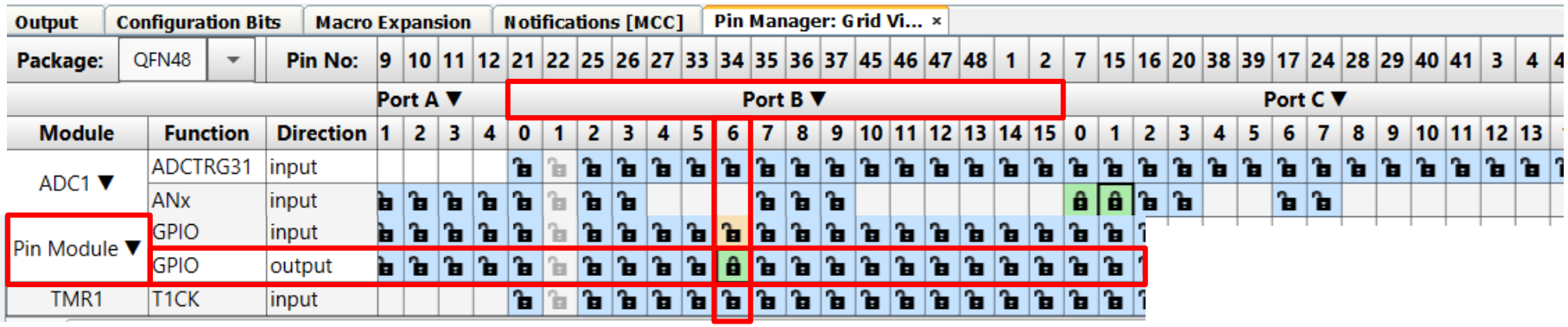


MICROCHIP

IO Configuration in MCC



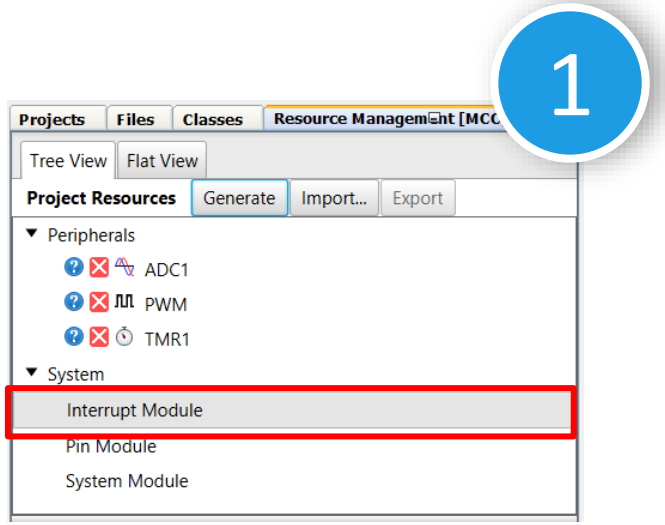




To make device pin RB6 an output, click on LOCK symbol representing device pin function

MICROCHIP

Interrupt Configuration in MCC



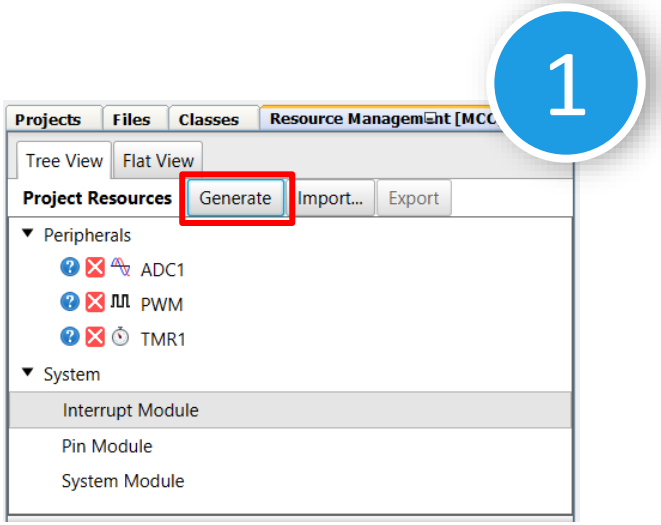
Easy Setup

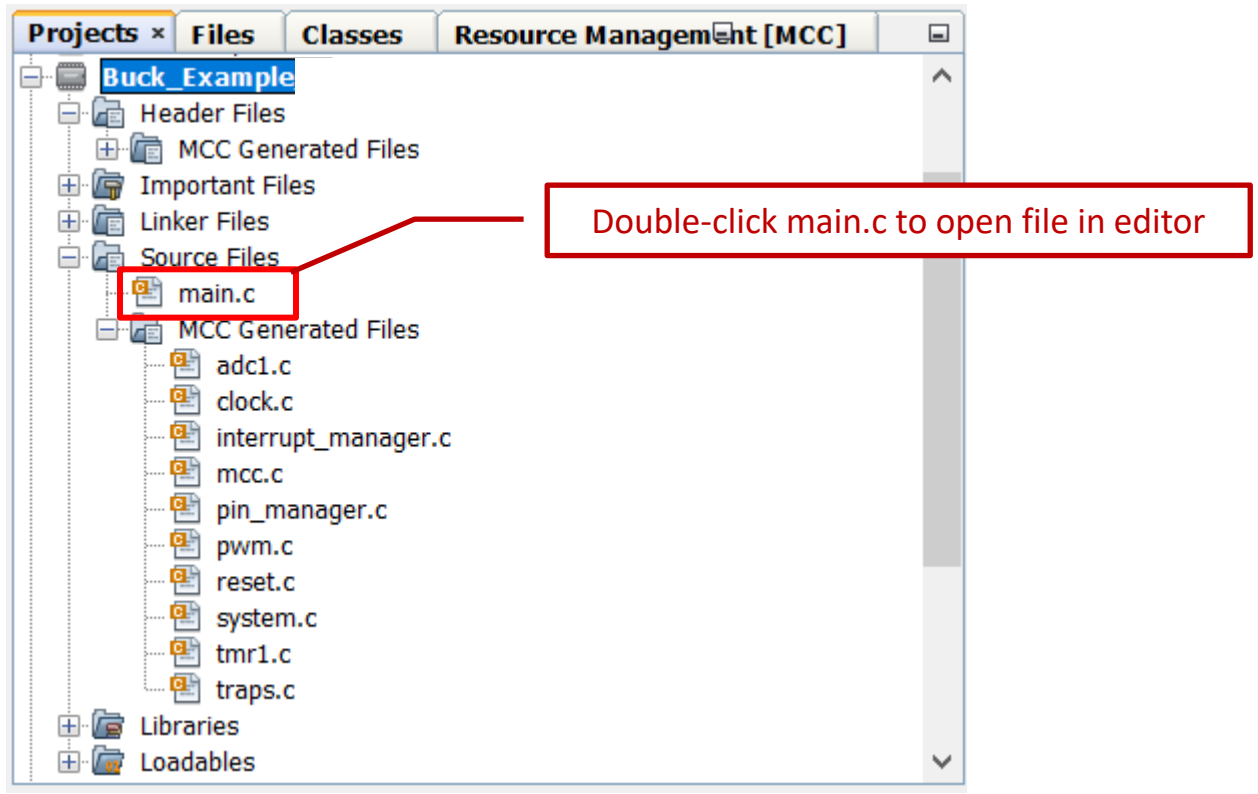
Interrupt Manager

Enable Global Interrupts

Module	Interrupt	Description	IRQ Nu...	Enabled	Priority	Context
Pin Module	CNAI	Change Notification A	2	☐	1	OFF
Pin Module	CNBI	Change Notification B	3	☐	1	OFF
Pin Module	CNCI	Change Notification C	19	☐	1	OFF
Pin Module	CNDI	Change Notification D	75	☐	1	OFF
TMR1	TI	Timer 1	1	☑	1	OFF
ADC1	ADCAN11	ADC AN11 Convert Done	102	☐	1	OFF
ADC1	ADCAN10	ADC AN10 Convert Done	101	☐	1	OFF
ADC1	ADCAN13	ADC AN13 Convert Done	104	☑	6	OFF
ADC1	ADCAN12	ADC AN12 Convert Done	103	☐	1	OFF

Generating the Project





MICROCHIP

Setting Up Interrupts

```
void TMR1_Int(void)
{
    static int led_tmr = 0;

    // blink LED 1Hz
    if(++led_tmr>=500) {
        led_tmr = 0;
        USER_LED_Toggle();
    }
}
```

```
void AN13_Int(void)
{
    adc_vin = ADCBUF12;
    adc_vbuck = ADCBUF13;
}
```

Recommended Intermediary Step:
If you program this code into the target device, The on-board LED will start blinking at a frequency of 1 Hz

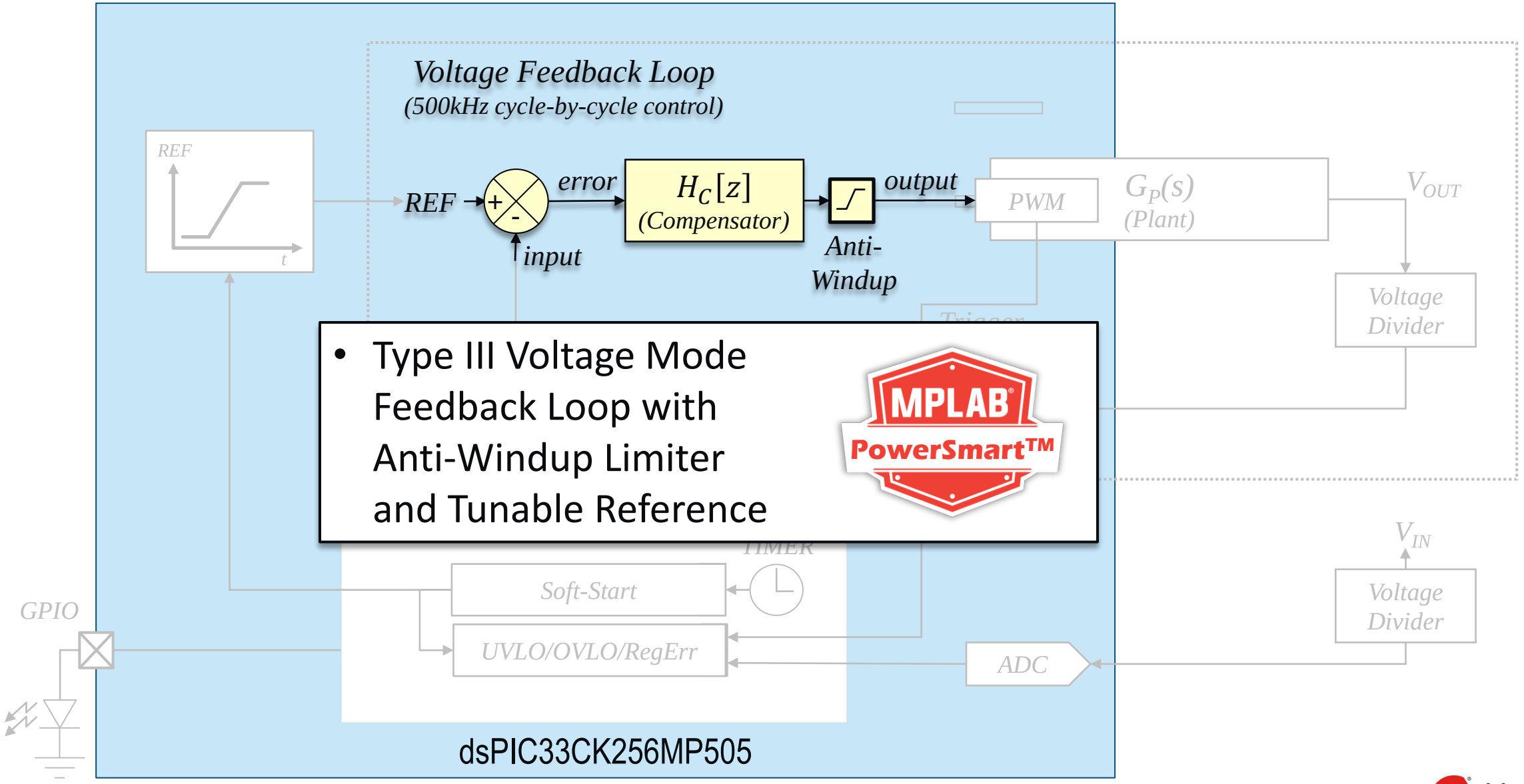
File main.c

```
/**
  Section: Included Files
*/
#include "mcc_generated_files/system.h"
#include "mcc_generated_files/mcc.h"

// Define Global Variables:
int adc_vin = 0;    // Scaling: 155.1 steps/V
int adc_vbuck = 0;  // Scaling: 620.5 steps/V
int ref_vbuck = 0;  // Scaling: 620.5 steps/V
[…]
```

add red colored code to file main.c

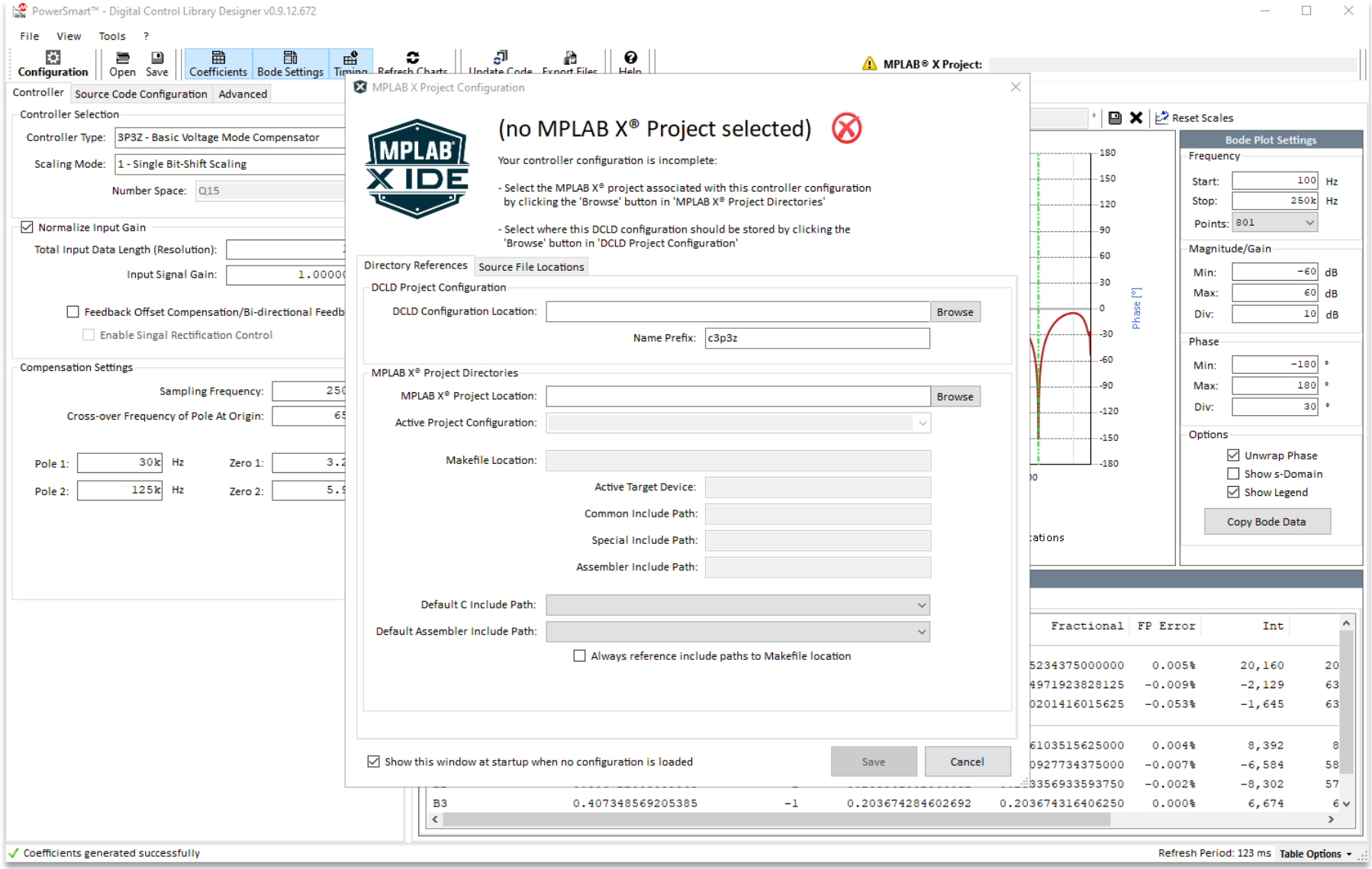
```
int main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    ADC1_SetChannel_AN13InterruptHandler(&AN13_Int);
    TMR1_SetInterruptHandler(&TMR1_Int);
    while (1)
    {
        // Add your application code
    }
    return 1;
}
```

Requirements

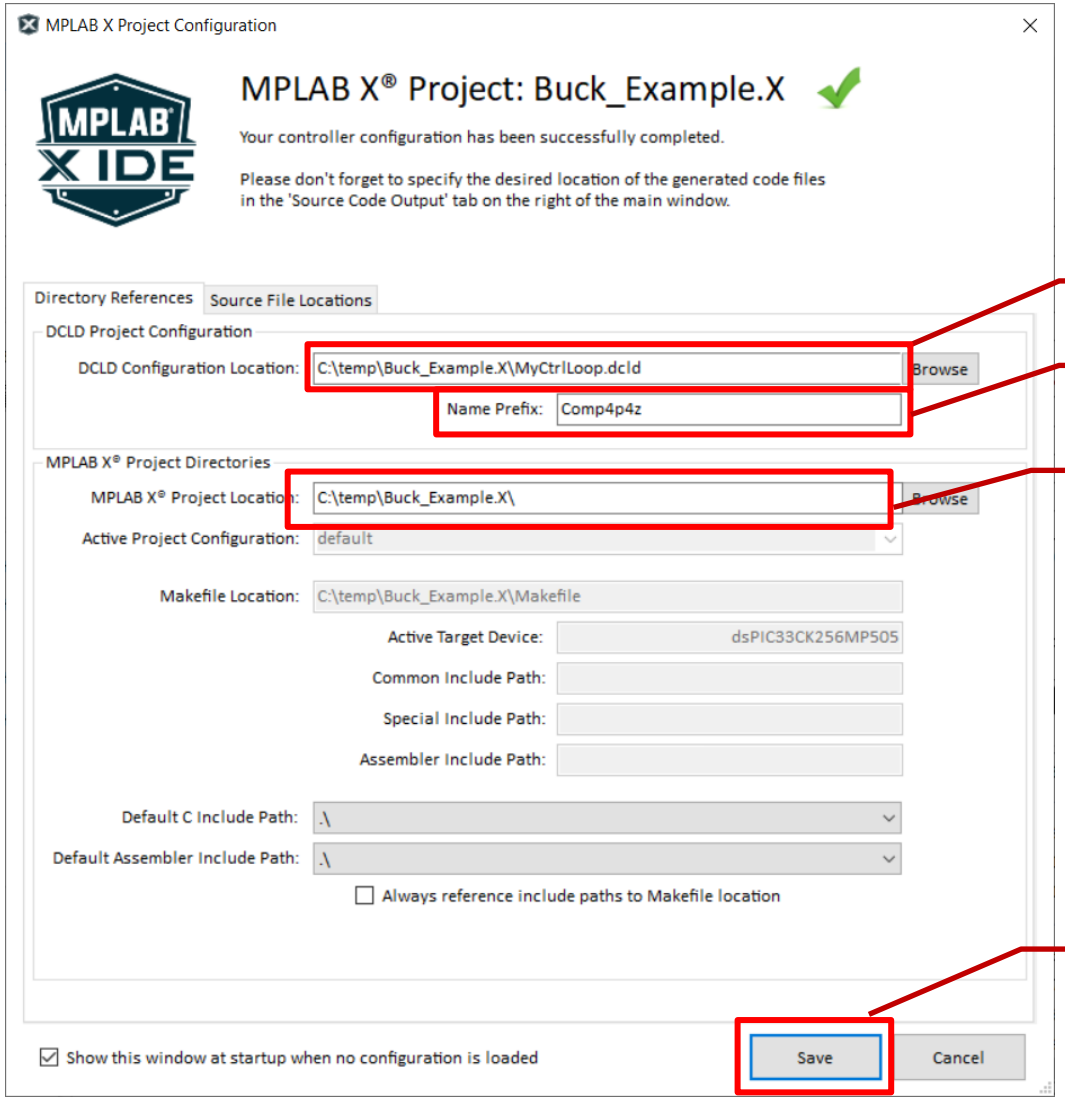


MICROCHIP

MPLAB® PowerSmart™ Digital Control Library Designer



Initial Project Configuration



PowerSmart™ DCLD Configuration Location

Controller Name

MPLAB® X Project File Location

MPLAB® X Project File Location

MICROCHIP

Compensation Filter Selection and Adjustment

Controller | Source Code Configuration | Advanced

Controller Selection

Controller Type: 4P4Z - Advanced High-Q Compensator

Scaling Mode: 4 - Fast Floating Point Coefficient Scaling

Number Space: Q15

☑ Normalize Input Gain

Total Input Data Length (Resolution): 12 Bit

Input Signal Gain: 0.500000

☐ Feedback Offset Compensation/Bi-directional Feedback

☐ Enable Singal Rectification Control

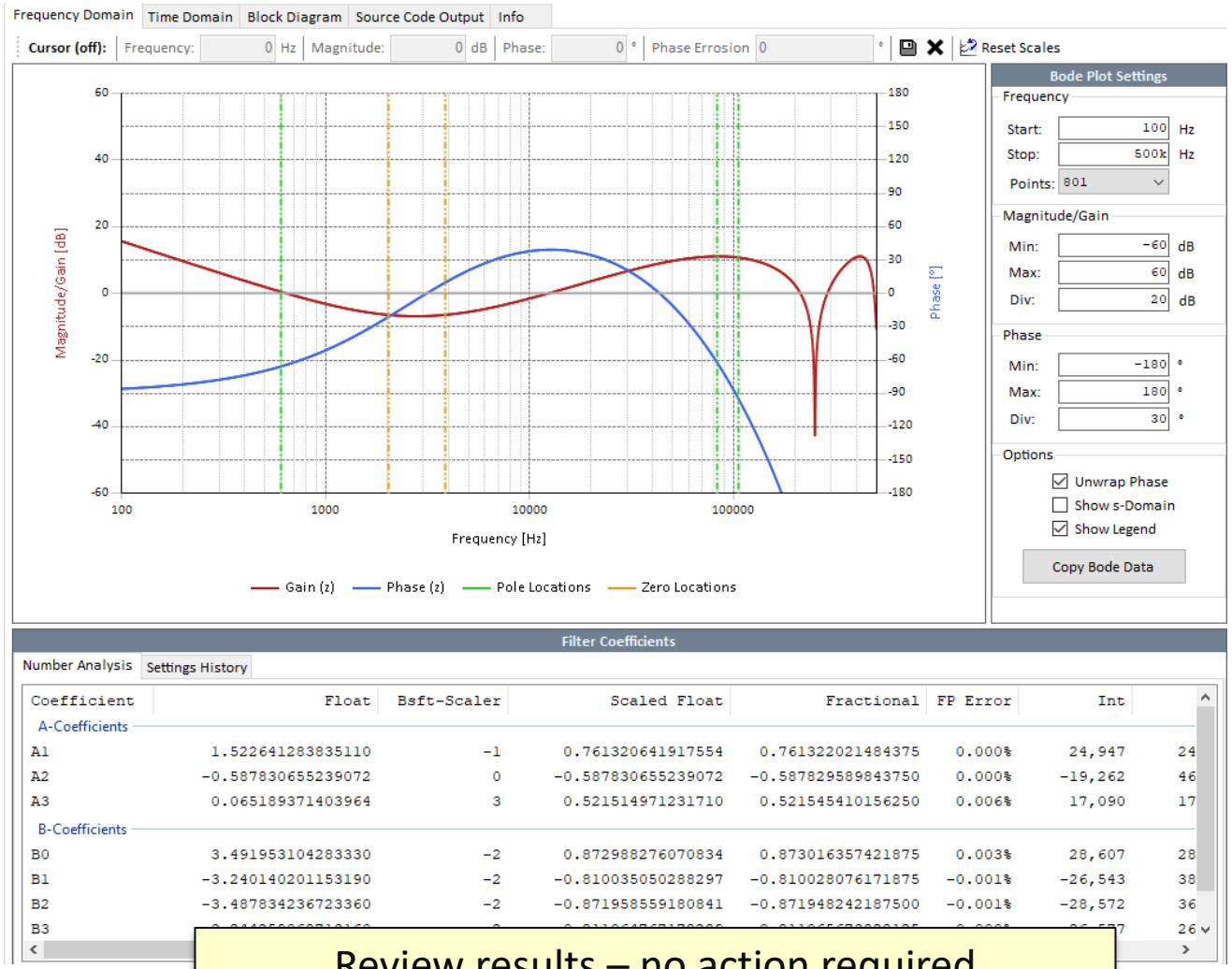
Compensation Settings

Sampling Frequency: 500k Hz

Cross-over Frequency of Pole At Origin: 600 Hz

Pole 1:	50k Hz	Zero 1:	2k Hz
Pole 2:	100k Hz	Zero 2:	4k Hz
Pole 3:	180k Hz	Zero 3:	18k Hz

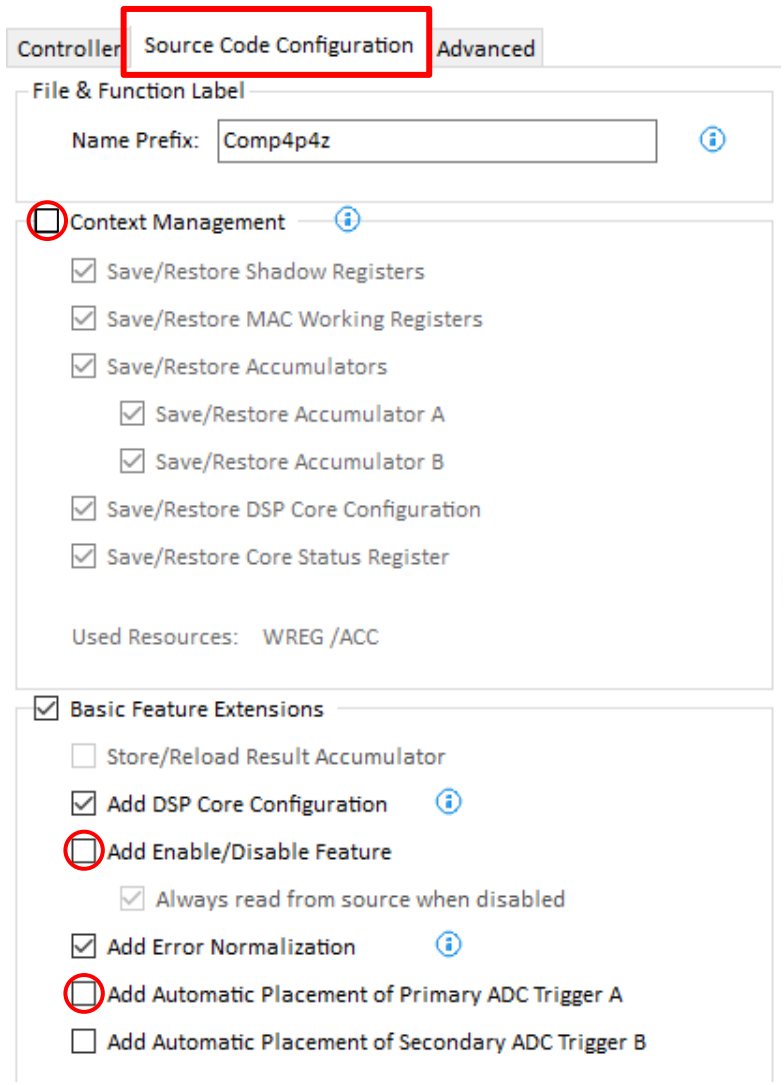
1



Review results – no action required

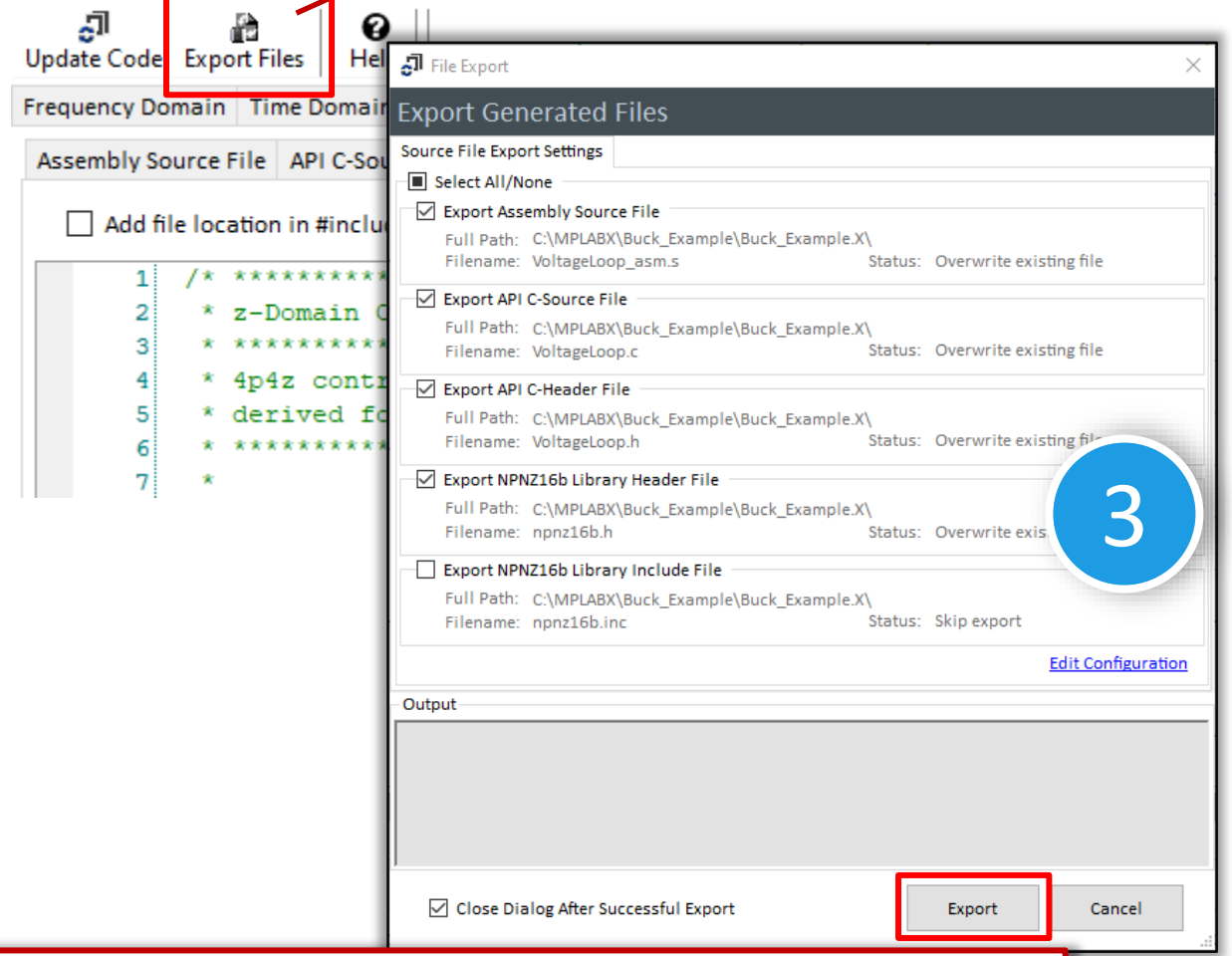
MICROCHIP

Configuring Software-Options



2

Click When Ready

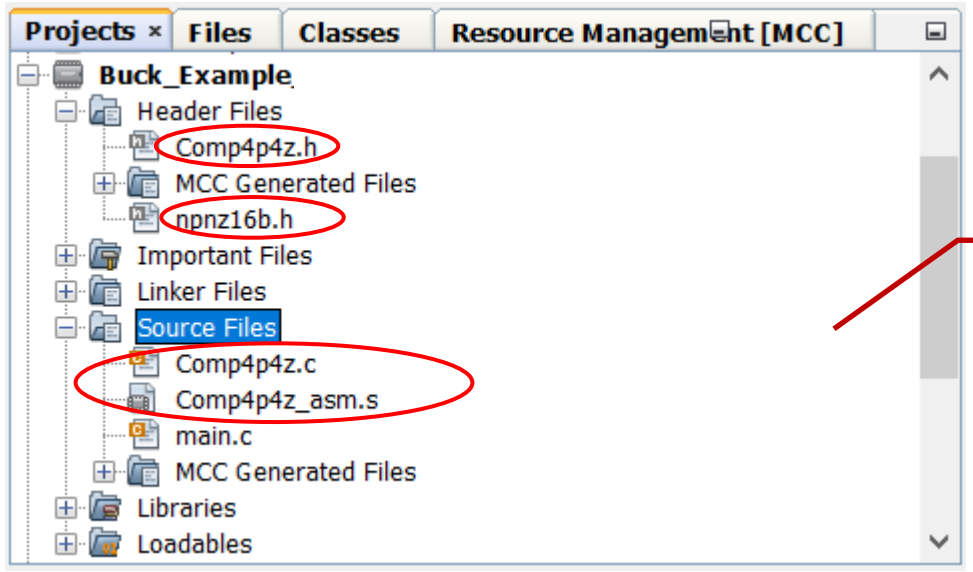


3

MPLAB® PowerSmart™ Digital Control Library Designer is generating files in project directory…

MICROCHIP

Add PS-DCLD Generated Files To Project



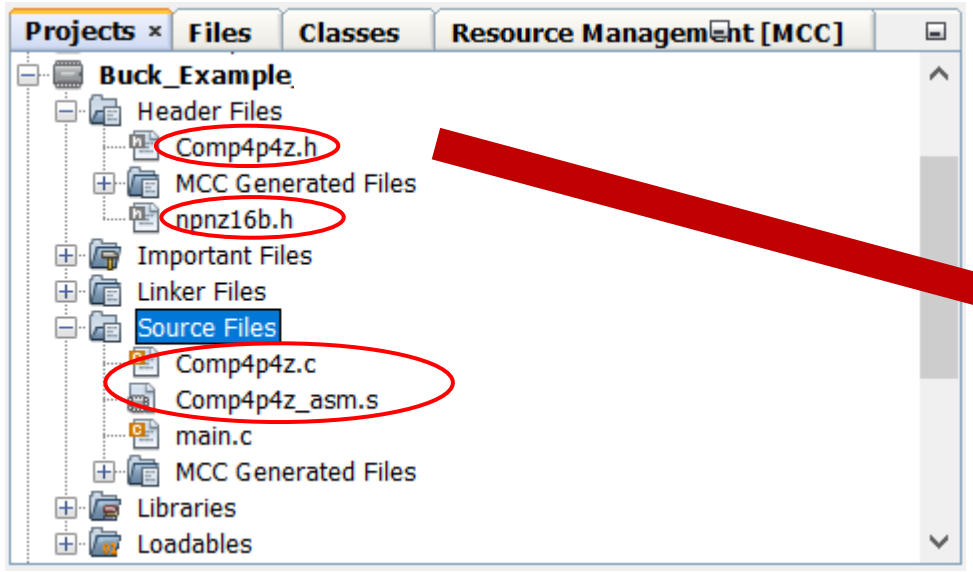
Add generated files to Project by Right-Click on *Header Files → Add Existing File:*

- *Source Files:*
  - *Assembly Library File (<my_controller>_asm.s)*
  - *C-Source File (<my_controller>.c)*
- *Header Files:*
  - *C-Header File (<my_controller>.h)*
  - *Library C-Header File (npnz16b.h)*

Recommended:

Also add the PS-DCLD Configuration file (*.dcld) to folder *Important Files* in the *Project Manager*. Thus, the current configuration can be conveniently opened from MPLAB® X by right-click on the configuration file, selecting *Open In System*

Including DCLD Generated Files




```
/**
  Section: Included Files
*/
#include "mcc_generated_files/system.h"
#include "mcc_generated_files/pin_manager.h"
#include "mcc_generated_files/pwm.h"
#include "mcc_generated_files/adc1.h"
#include "mcc_generated_files/tmr1.h"
#include "Comp4p4z.h"
```

Add call of newly added controller function to ADC Interrupt Service Routine

```
void AN13_Int(void)
{
    adc_vin = ADCBUF12;
    adc_vbuck = ADCBUF13;
    Comp4p4z_Update(&Comp4p4z);
}
```

Adding DCLD Controller Initialization Routine

```
void CONTROL_Initialize(void)
{
    // initialize compensator
    Comp4p4z_Initialize(&Comp4p4z);
    Comp4p4z.Ports.Source.ptrAddress = (unsigned*)&adc_vbuck;
    Comp4p4z.Ports.Target.ptrAddress = (unsigned*)&PG1DC;
    Comp4p4z.Ports.ptrControlReference = (unsigned*)&ref_vbuck;
    Comp4p4z.Limits.MinOutput = 0;
    Comp4p4z.Limits.MaxOutput = (int)(0.8*PG1PER); // 80% DC
}
```

Add call of newly added controller initialization function to main(void)

File main.c

```
/**
  Section: Included Files
*/

[…]
```

Add code before main(void)

```
int main(void)
{
    // initialize the device
    SYSTEM_Initialize();
    CONTROL_Initialize();

    ADC1_Setchannel_AN13InterruptHandler(&AN13_Int);
    TMR1_SetInterruptHandler(&TMR1_Int);

    while (1)
    {
        // Add your application code
    }
    return 1;
}
```

You are all set to program the device now…

May the Power be with you