

10Base-T1S Hands-On Training

Lab Manual

Goal of this Hands-On Class

- Present a demonstration of an example of Single Pair Ethernet communication over 10BASE-T1S
- Use LAN8670 RMII evaluation board with SAME54 Curiosity Ultra
- Based on Application Note AN4131



AN4131

MPLAB® Harmony v3 LAN867x Driver Example

1.0 INTRODUCTION

The LAN867x is a high-performance 10BASE-T1G single-pair Ethernet PHY transceiver that is targeted for 10 Mbit/s half-duplex networking over a single pair of conductors.

This document guides you in creating a sample TCP/IP Client node (bare-metal or FreeRTOS™ based), using the LAN867x PHY. It describes how to configure the PHY in either Physical Layer Collision Avoidance (PLCA) or Carrier-Sense Multiple Access/Collision Detection (CSMA/CD) mode.

The description in this document is based on an ATSAME54P20A running on a SAM E54 Curiosity Ultra Development Board [3]. However, it can also be applied to other infrastructures; for example, to an ATSAME70Q21B running on a SAM E70 Xplained Ultra Evaluation Kit [4].

1.1 Audience

This document is written for developers who want to create a sample TCP/IP Client node, using the LAN867x PHY. Developers should be familiar with the infrastructure of MPLAB Code Configurator (MCC) and its plug-ins [1].

1.2 References

The following sources should be referenced when using this application note.

[1] MPLAB Code Configurator

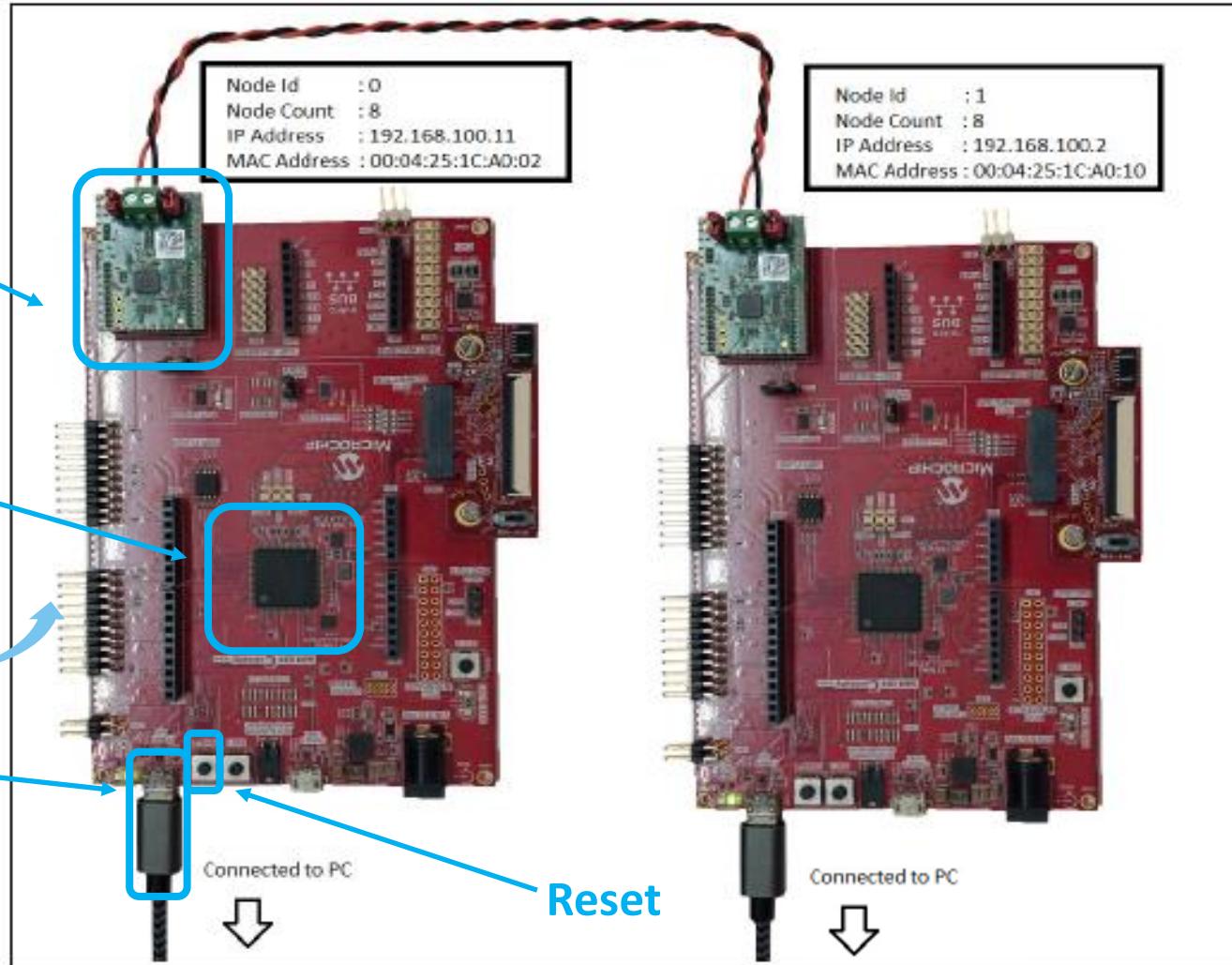
- Link to AN4131 is given [here](#) and is also available on [LAN8670 Product Page](#)

LAN8670 RMII evaluation board with SAME54 Curiosity Ultra

Ethernet Interface
with RMII mounted
with EVB-LAN8670-
RMII to 10BASE-T1S
interface card

ATSAME54P20A
On board EDBG
debugger

USB Debugger
Interface



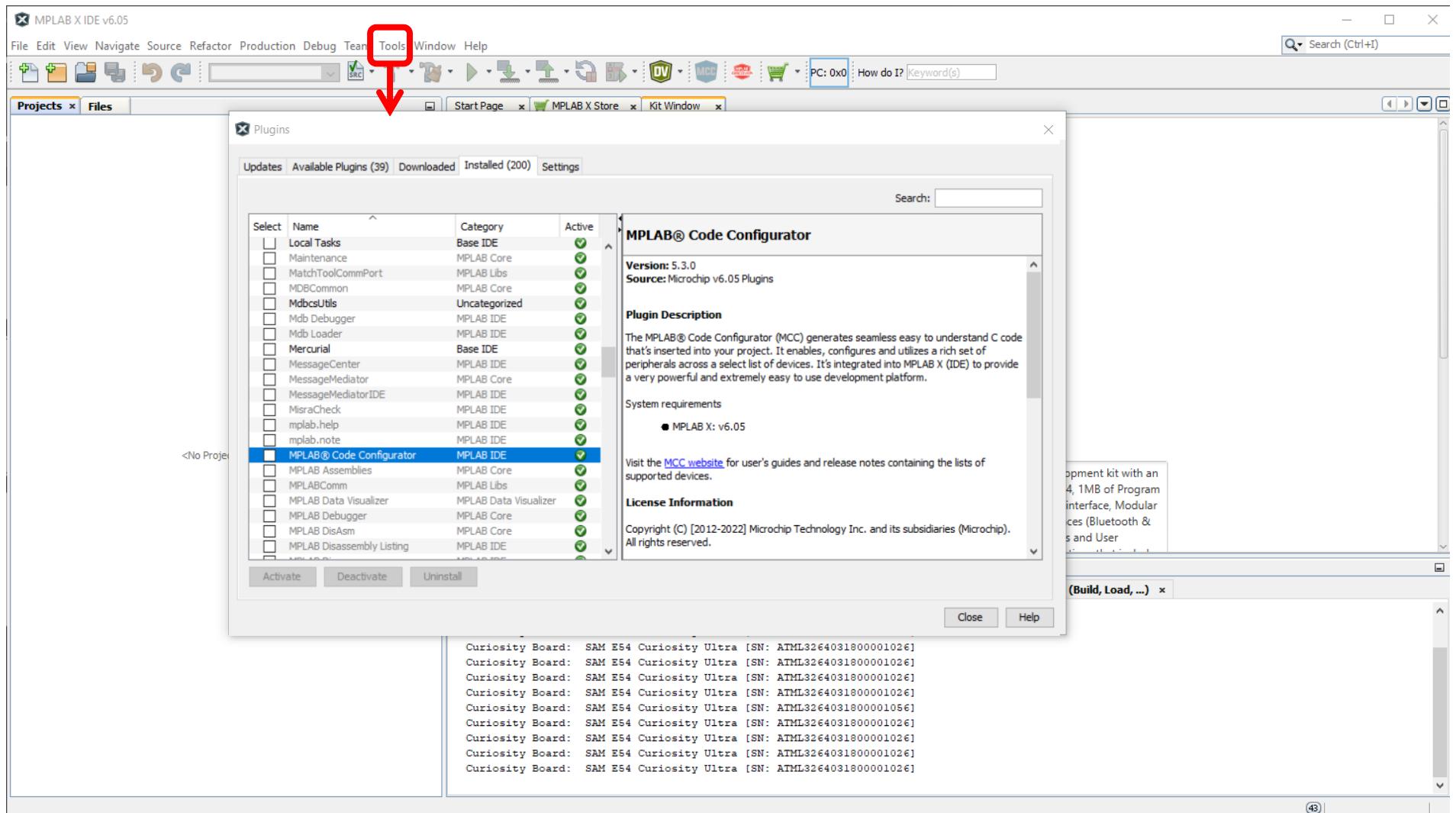
Project Setup

Step-by-Step Guide to Creating the Project

Step 1

Having opened MPLAB X v6.05 (See [Useful Links and References](#)) , in Tools > Plugins, check that you have MPLAB Code Configurator installed.

If it is not there, look in Available Plugins and install it from there.

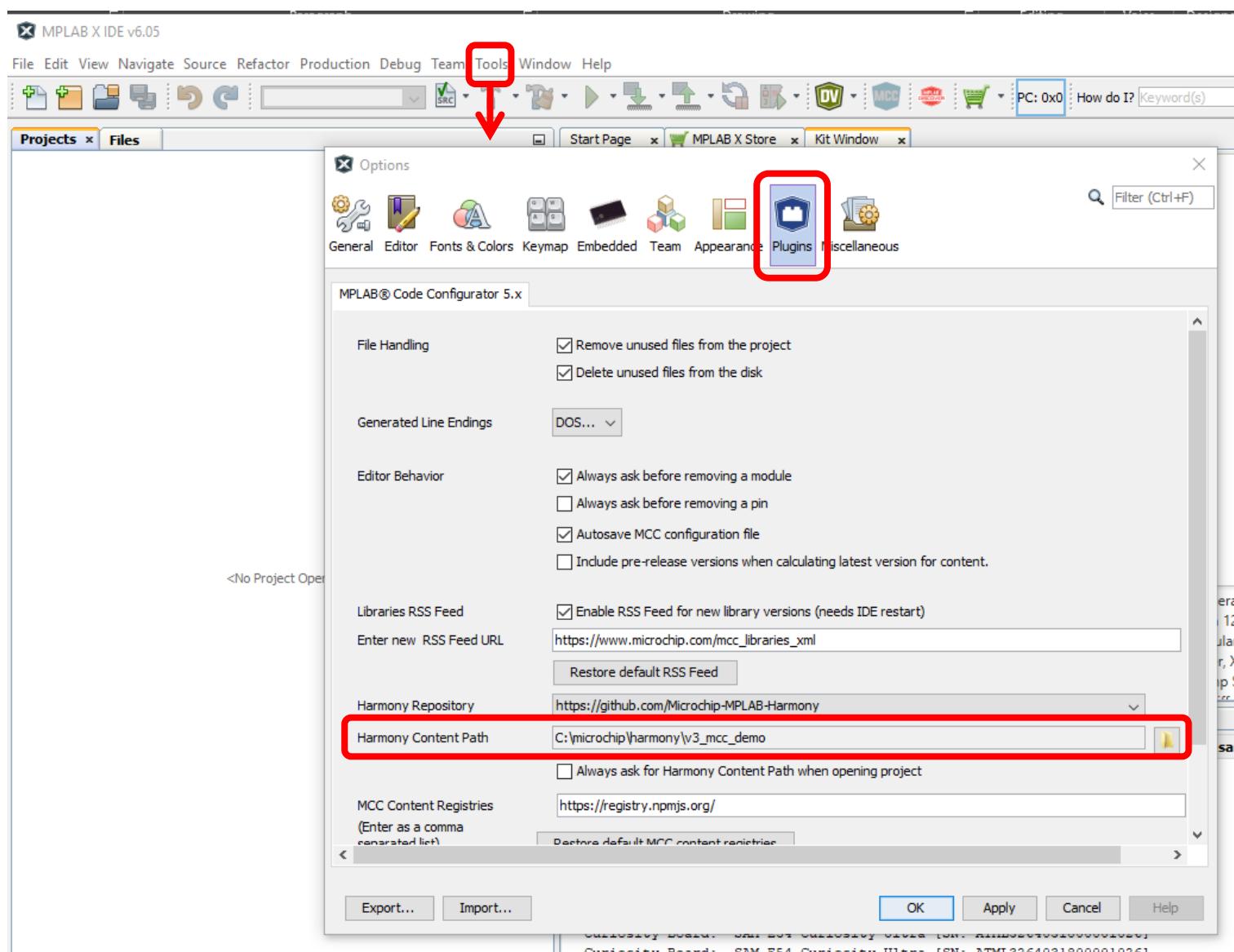


Step 2

In Tools > Options, go to the Plugins tab.

If you have not used Harmony in the past: In your Folders on your machine you need to create an empty folder where all of the harmony libraries will be downloaded to. Having created this folder, put the folder location in the **Harmony Content Path** in as shown.

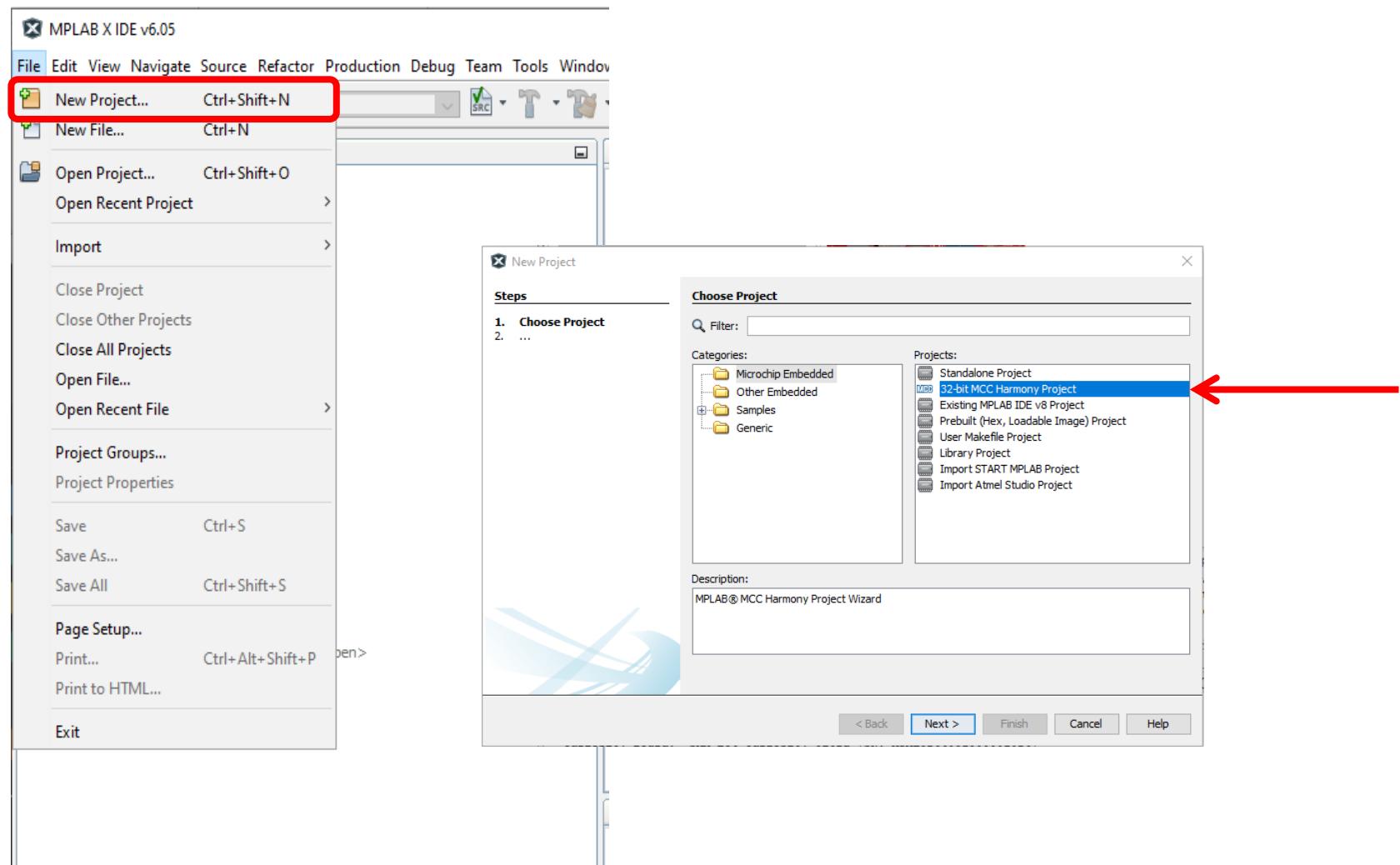
If you are a Harmony user: if you have used Harmony before, you will have already created a repository for all of the libraries, and you can use this same folder for the purposes of this class.



Step 3

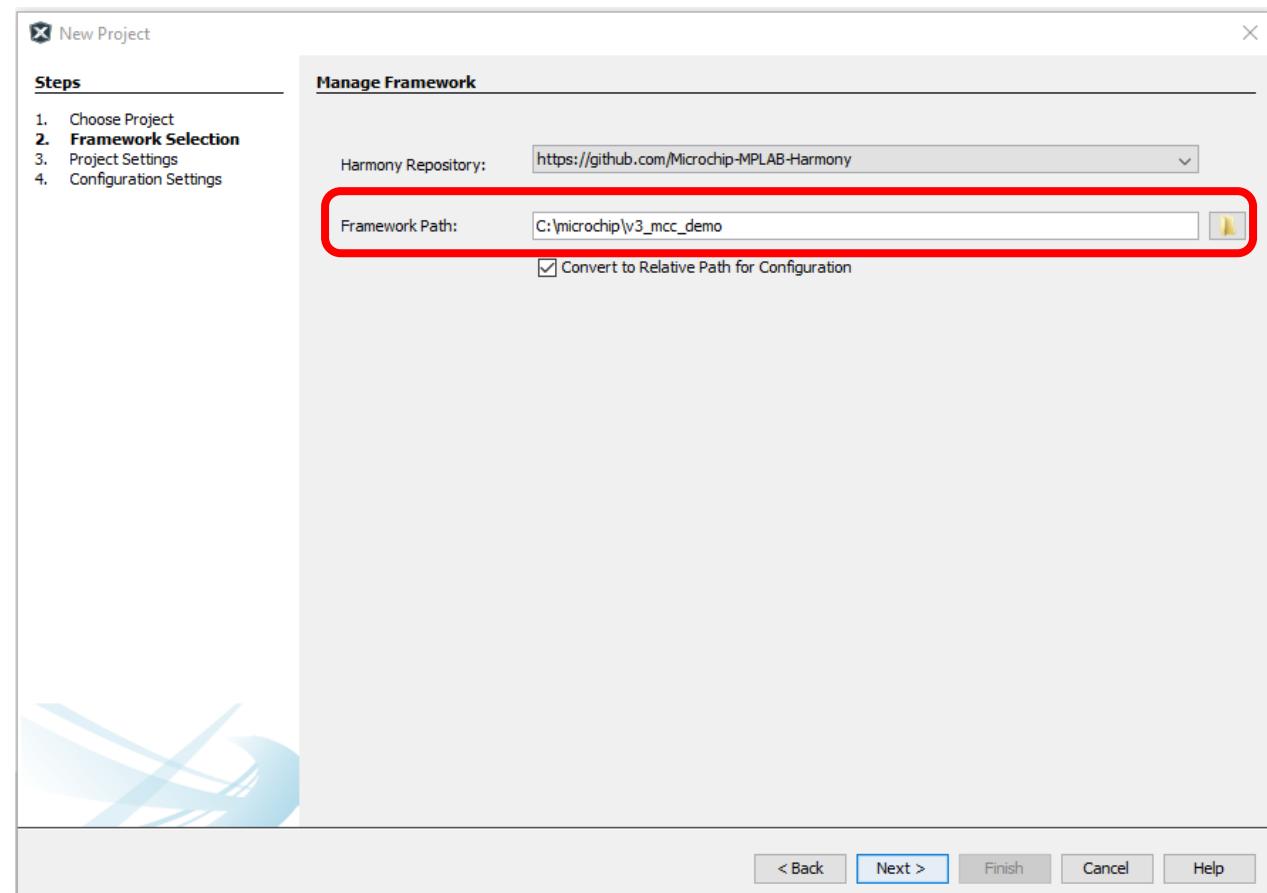
In MPLAB X 6.05, go **File**
> New Project...

Select **32-bit MCC
Harmony Project** and
click **Next>**



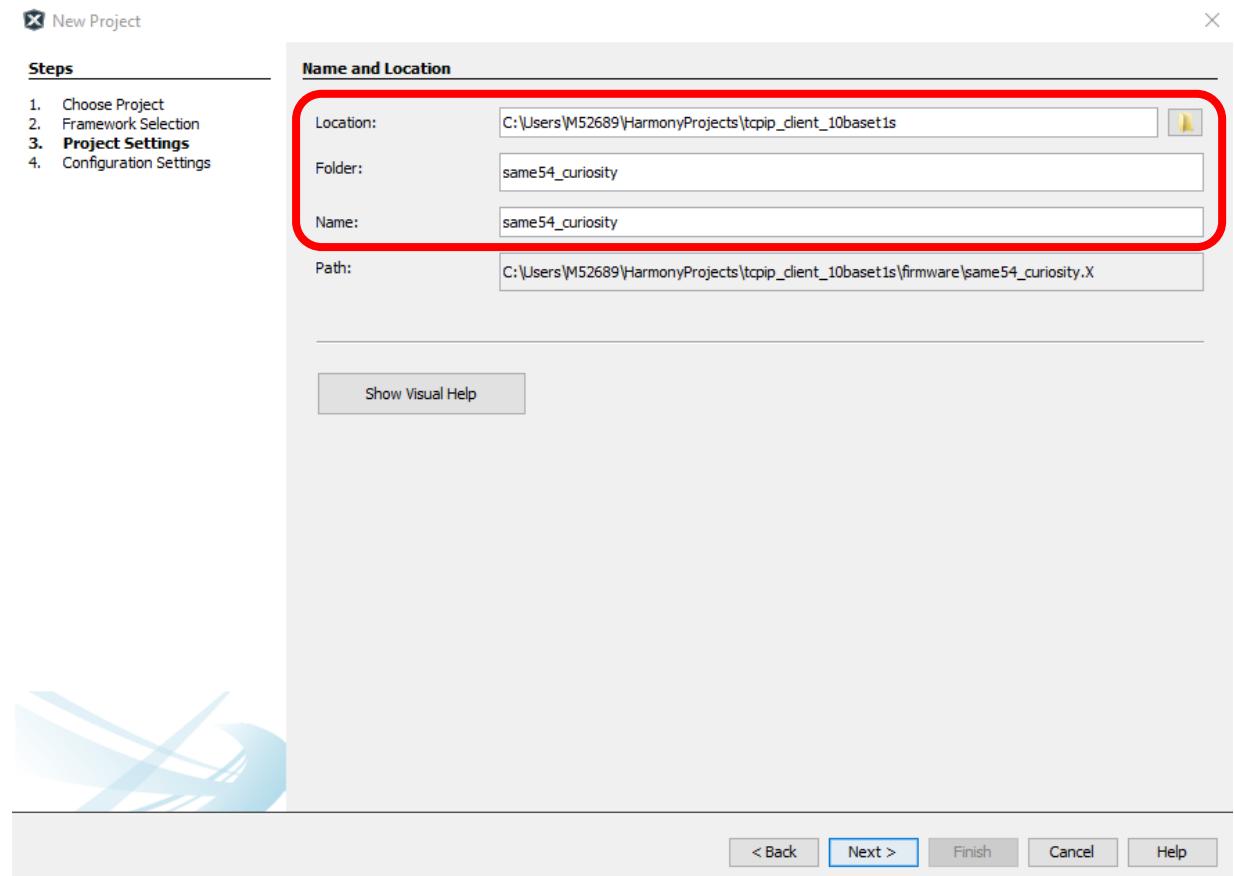
Step 4

The Framework Path needs to be the same as you specified in the Tools>Options>Plugins **Harmony Content Path** in the Step 2.



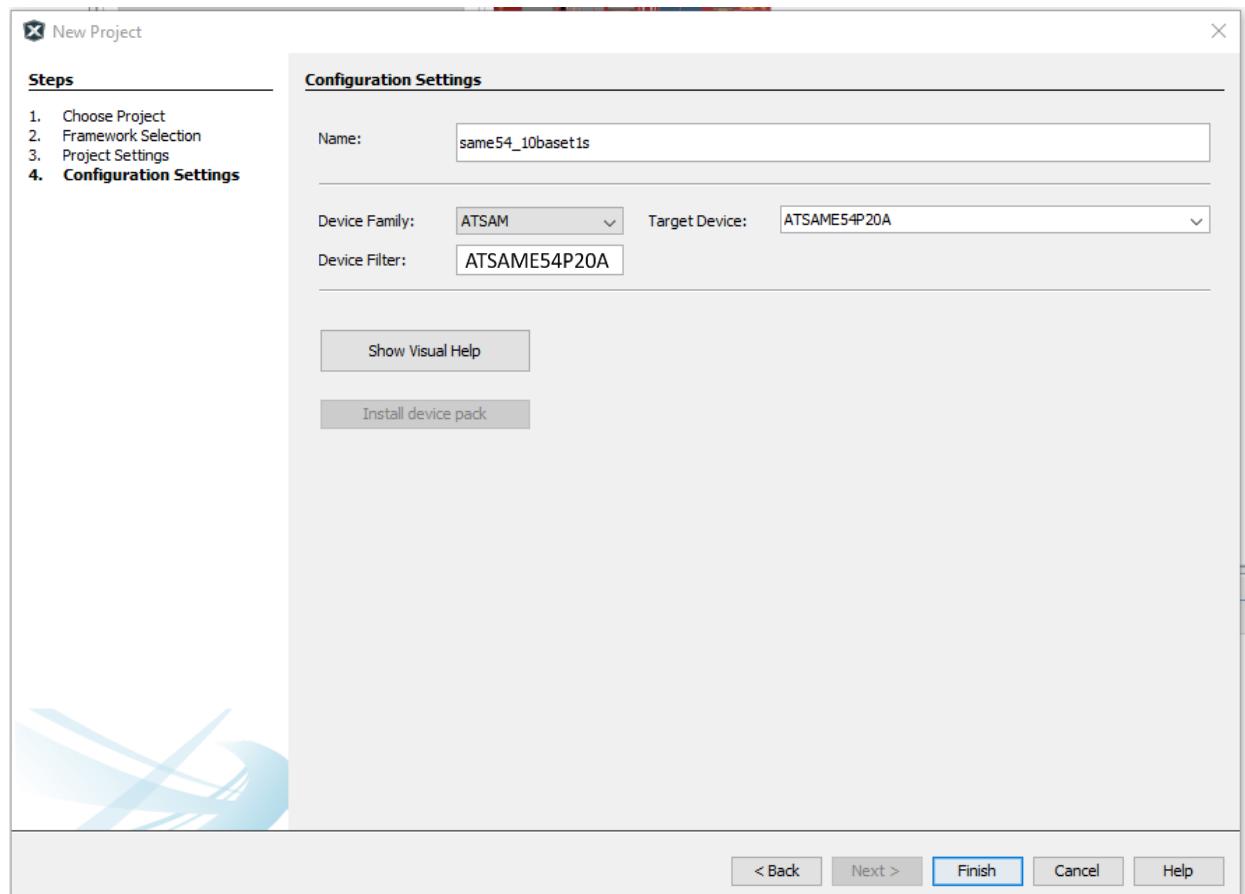
Step 5

Specify the **Location**, **Folder** and **Name** of the project (Tip – Show Visual Help is really useful to understand the differences between these.). Click **Next >**.



Step 6

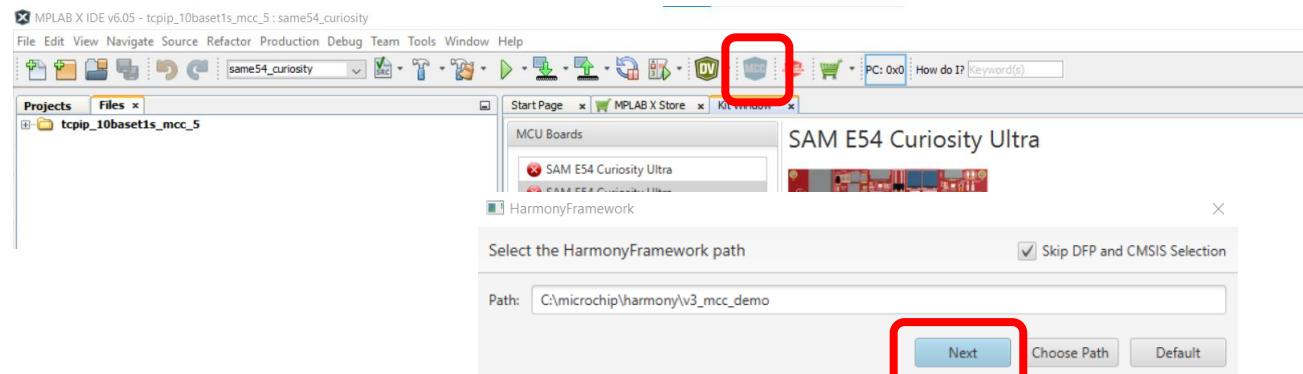
Choose a **Configuration Name** and select the **Device Family** as ATSAM, and **Target Device** as ATSAME54P20A. (Tip: copy/paste ATSAME54P20A into the Device Filter and this will find the device quickly) Click **Finish**.



Step 7

MCC should now open automatically. If it doesn't, use the blue MCC button to open MCC

- This button can also be used to close MCC if it is open
- Check the Framework path is correct and click Next

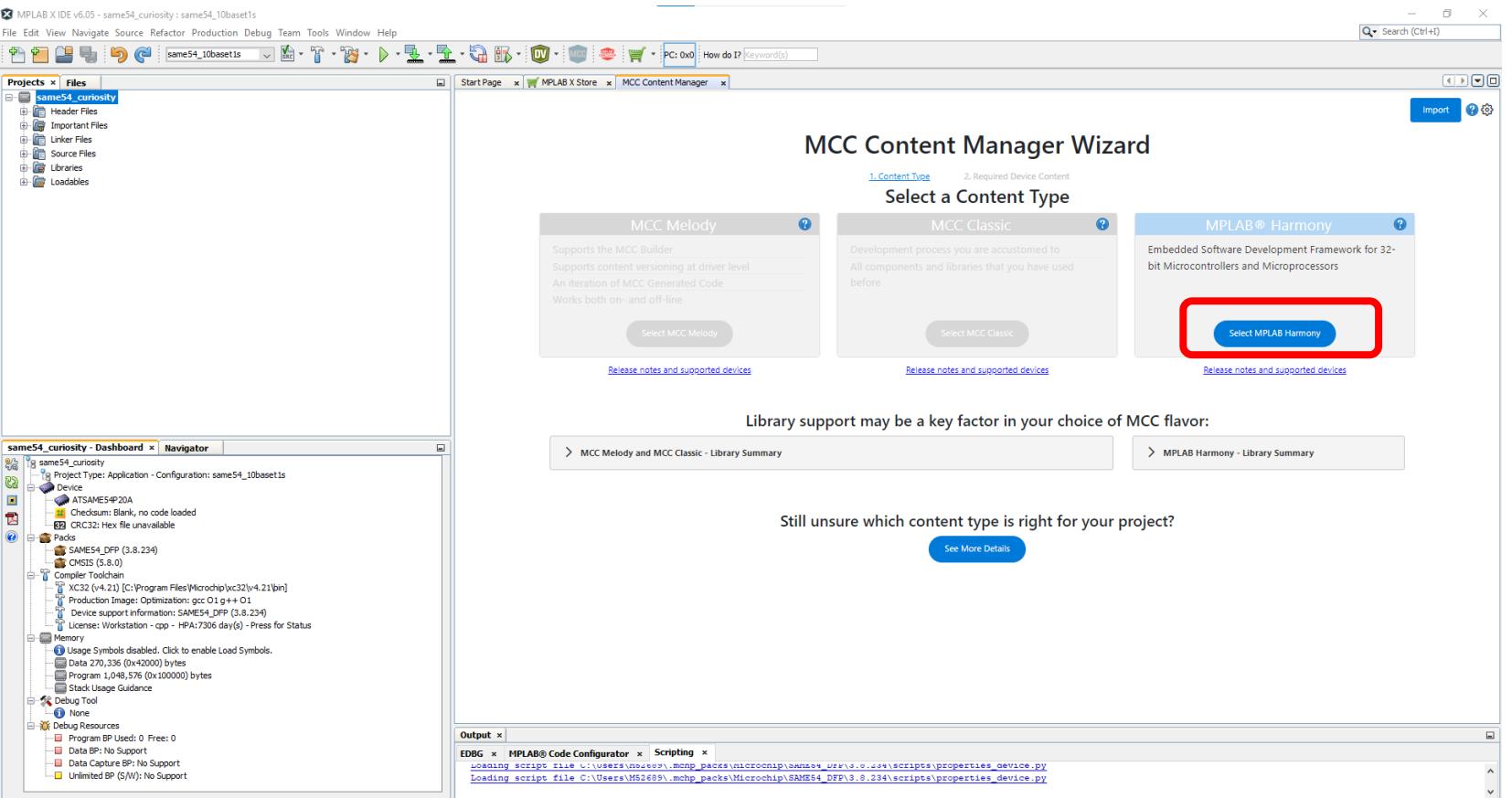


Step 8

You have now created your project and opened MCC. The next step is to download the necessary Harmony Content libraries for the demo to run.

You can now see your project in the Projects pane on the left, and on the right the tab for the MCC Content Manager.

Click on **Select MPLAB Harmony** for the Content Type.



Step 9

Focussing on the MCC Content Manager Wizard, you can see that there are two types of Content, Required Content and Optional Content. Required Content is necessary for any Harmony project, but everything else is application dependent.

Click on the “down arrow” to see the names of each package.

The screenshot shows the MCC Content Manager Wizard interface. At the top, tabs for 'Content Type' (selected), 'Required Device Content', and 'Finish' are visible. Below the tabs, the title 'MCC Content Manager Wizard' is displayed. The 'Required Content' section is shown with a table:

Component ↑	Version	Update progress	Description
Harmony 3 - Chip Support Package			
csp	3.16.0		
dev_packs	3.16.0		
Harmony 3 - Reference Apps			
quick_docs	1.5.0		
Harmony 3 - Harmony Services			
harmony-services	v1.2.1		

Below this, the 'Optional Content' section is shown with a table:

Component ↑	Version	Description
Harmony 3 - Wireless BLE solutions		
wireless_apps_pic32cxbz2_wbz45	1.2.0	
wireless_ble	1.0.0	
wireless_pic32cxbz_wbz	1.1.0	
wireless_system_pic32cxbz_wbz	1.2.0	

Step 10

From the list of Optional Packages, you will need:

net
net_10base_t1s
bsp
core
crypto
wolfssl
CMSIS-FreeRTOS

Note: The versions shown here are not necessarily the most up-to-date ones, but please choose the most up-to-date versions available for each pack.

Scroll back up to the top and click Finish.

The packages will start downloading from Github into the Harmony Content Folder.

Use the Update progress column to view the installation progress.

Harmony 3 - Networking Stack and Solutions	
<input type="checkbox"/> ethercat	3.2.0
<input checked="" type="checkbox"/> net	3.9.2
<input checked="" type="checkbox"/> net_10base_t1s	1.2.2
<input type="checkbox"/> net_apps_pic32mx	3.8.0
<input type="checkbox"/> net_apps_pic32mz	3.8.0
<input type="checkbox"/> net_apps_sam_9x60	3.9.0
<input type="checkbox"/> net_apps_sam_9x7	3.0.0
<input type="checkbox"/> net_apps_sam_a5d2	3.9.0
<input type="checkbox"/> net_apps_sam_a7g5	3.0.0
<input type="checkbox"/> net_apps_sam_e5x	3.8.0
<input type="checkbox"/> net_apps_sam_e70_v71	3.8.0
<input type="checkbox"/> net_apps_sam_rh71	3.8.0
Harmony 3 - USB solutions	
<input type="checkbox"/> usb	3.10.0
<input type="checkbox"/> usb_apps_device	3.4.1
<input type="checkbox"/> usb_apps_dual_role	3.3.1
<input type="checkbox"/> usb_apps_host	3.4.1
<input type="checkbox"/> usb_apps_multi_controller	3.3.1
Harmony 3 - Core	
<input checked="" type="checkbox"/> bsp	3.15.0
<input checked="" type="checkbox"/> core	3.12.0
<input type="checkbox"/> core_apps_cec173x	3.0.0
<input type="checkbox"/> core_apps_pic32cm_jh00_jh01	3.3.0
<input type="checkbox"/> core_apps_pic32cm_le_ls	3.3.0
<input type="checkbox"/> core_apps_pic32cm_mc00	3.3.0
<input type="checkbox"/> core_apps_pic32cx_mt	3.0.0

arm CMSIS	
<input type="checkbox"/> CMSIS_5	5.9.0
<input type="checkbox"/> CMSIS-DSP	v1.14.4
<input checked="" type="checkbox"/> CMSIS-FreeRTOS	v10.5.1
<input type="checkbox"/> CMSIS-NN	23.02
<input type="checkbox"/> CMSIS-View	1.1.0
Harmony 3 - Azure RTOS solutions	
<input type="checkbox"/> azure_rtos	1.0.0
<input type="checkbox"/> filex	v6.2.1_rel
<input type="checkbox"/> netxduo	v6.2.1_rel
<input type="checkbox"/> threadx	v6.2.1_rel
Harmony 3 - Amazon FreeRTOS solutions	
<input type="checkbox"/> amazon-freertos	202107.00
Harmony 3 - Micrium uCOS III port	
<input type="checkbox"/> micrium_uicos3	3.1.0
Harmony 3 - TensorFlow Lite for Microcontroller (TFLM) Solutions	
<input type="checkbox"/> tflite-micro	v1.0.0
<input type="checkbox"/> tflite-micro-apps	1.0.2
Harmony 3 - Reference Apps	
<input type="checkbox"/> reference_apps	1.5.0
Harmony 3 - Cryptography solutions	
<input checked="" type="checkbox"/> crypto	3.8.0
<input type="checkbox"/> crypto_apps_encrypt_decrypt	3.7.1
<input type="checkbox"/> crypto_apps_large_hash	3.7.1
<input type="checkbox"/> crypto_apps_speed_test	3.8.0

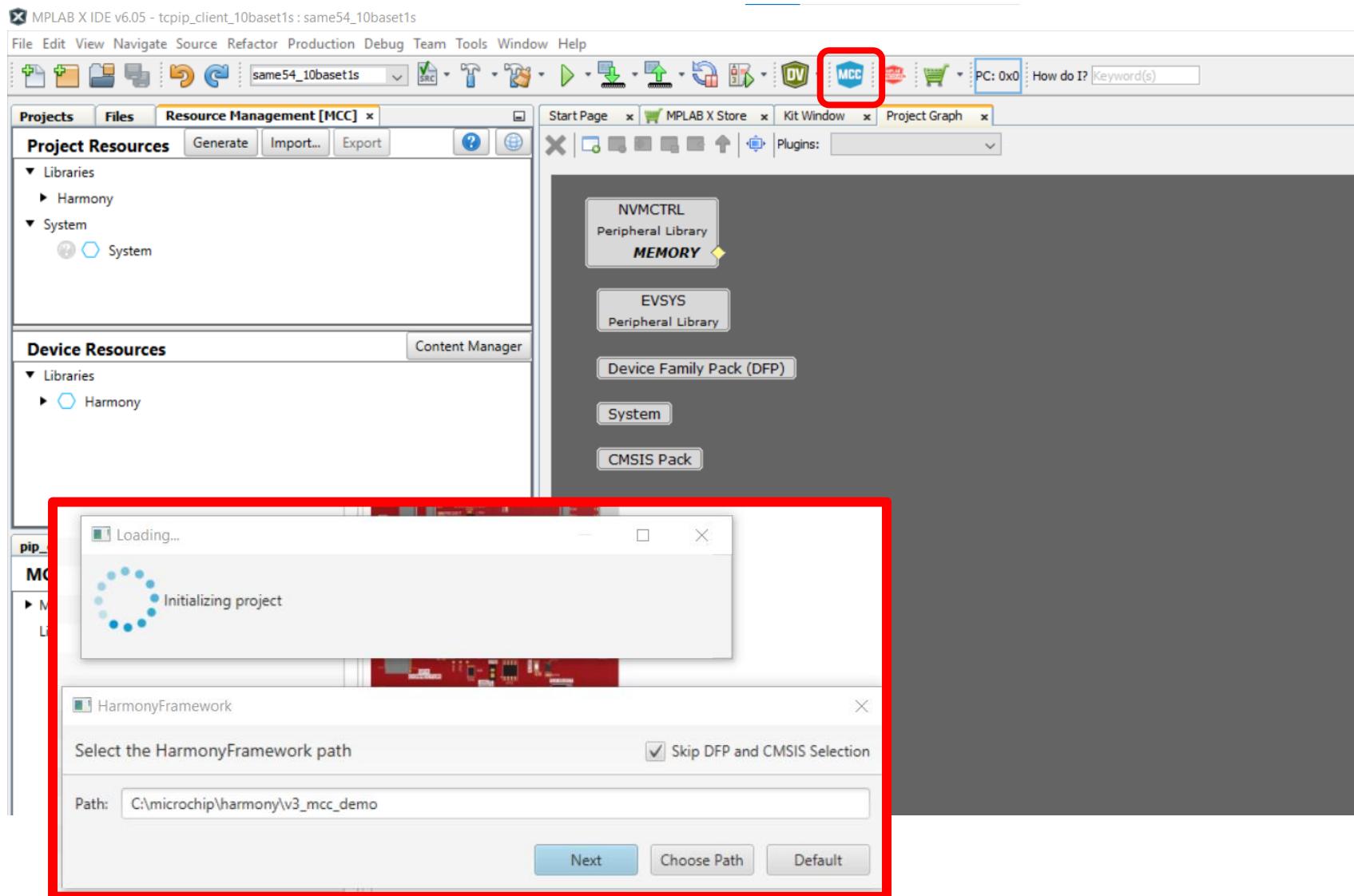
<input type="checkbox"/> aws_cloud	3.2.0
Harmony 3 - WolfSSL solutions	
<input type="checkbox"/> wolfMQTT	v1.11.1
<input type="checkbox"/> wolfssh	v1.4.1
<input checked="" type="checkbox"/> wolfssl	v5.4.0
Harmony 3 - Soteria secureboot solution library	
<input type="checkbox"/> cec173x_soteria_lib	3.2.0
Harmony 3 - zlib data-compression library	

Note : it is helpful to have a fast internet connection for this step!

Step 11

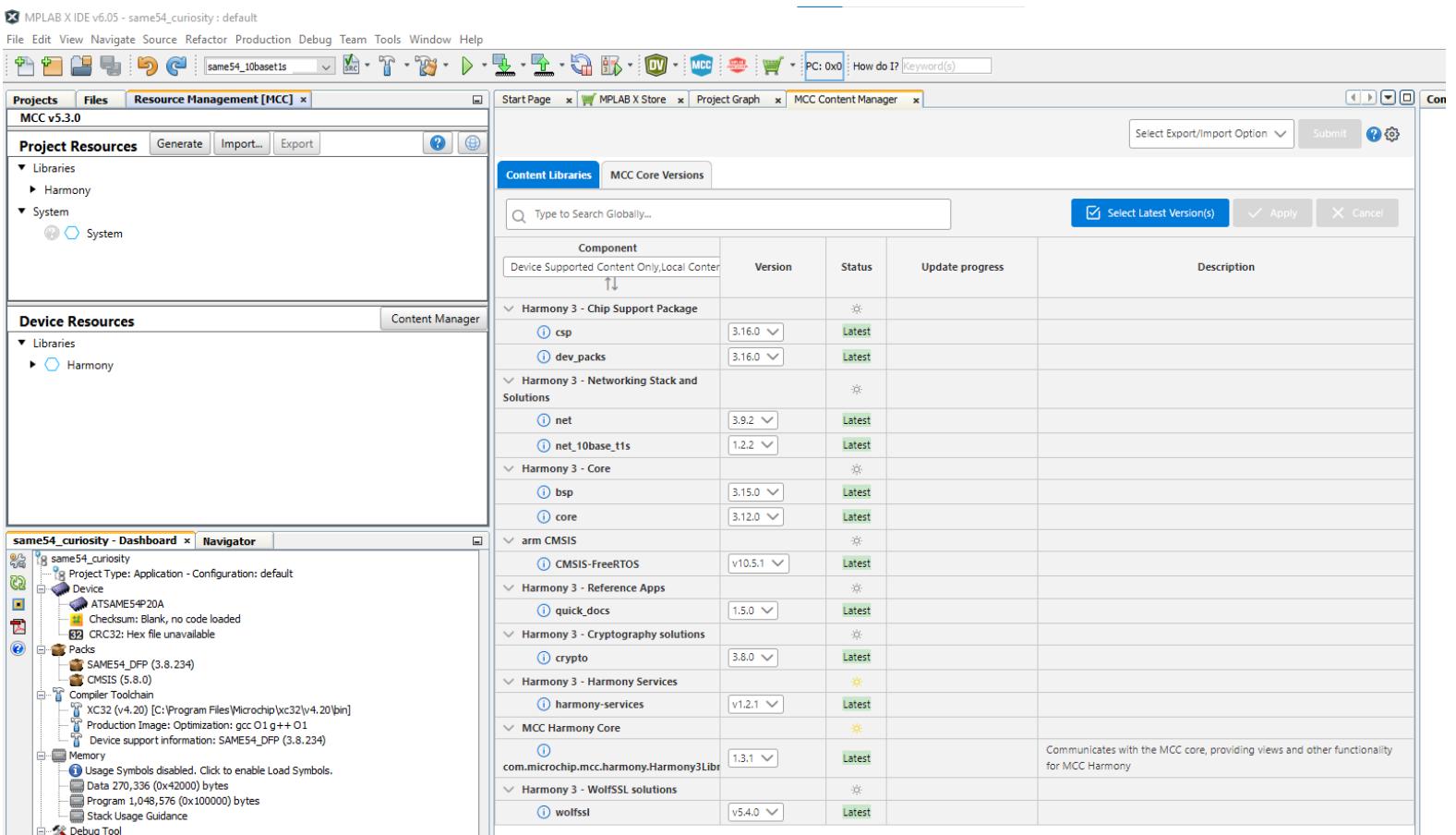
Once the libraries have downloaded, the project should open in MCC automatically. If it doesn't, then you can open MCC by clicking on the MCC button at the top of the MPLAB X window.

If the Harmony Framework path box appears, make sure that it has the same path that you specified originally for the project and for the Plugin Options in the earlier steps.



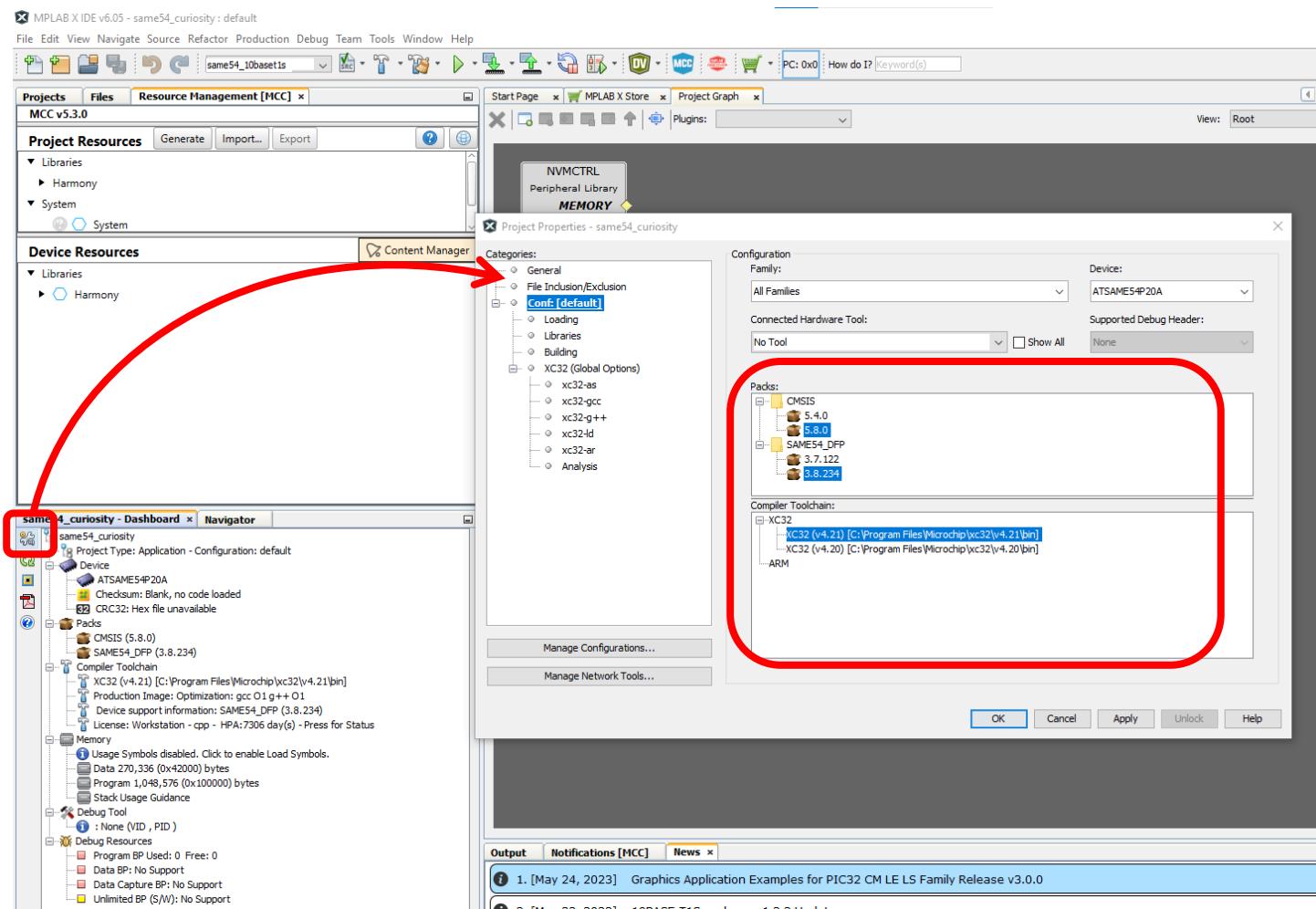
Step 12

Open the Content Manager and view the packages that have been downloaded (tip: use the down arrows beside the package headings to view all of the packages)



Step 13

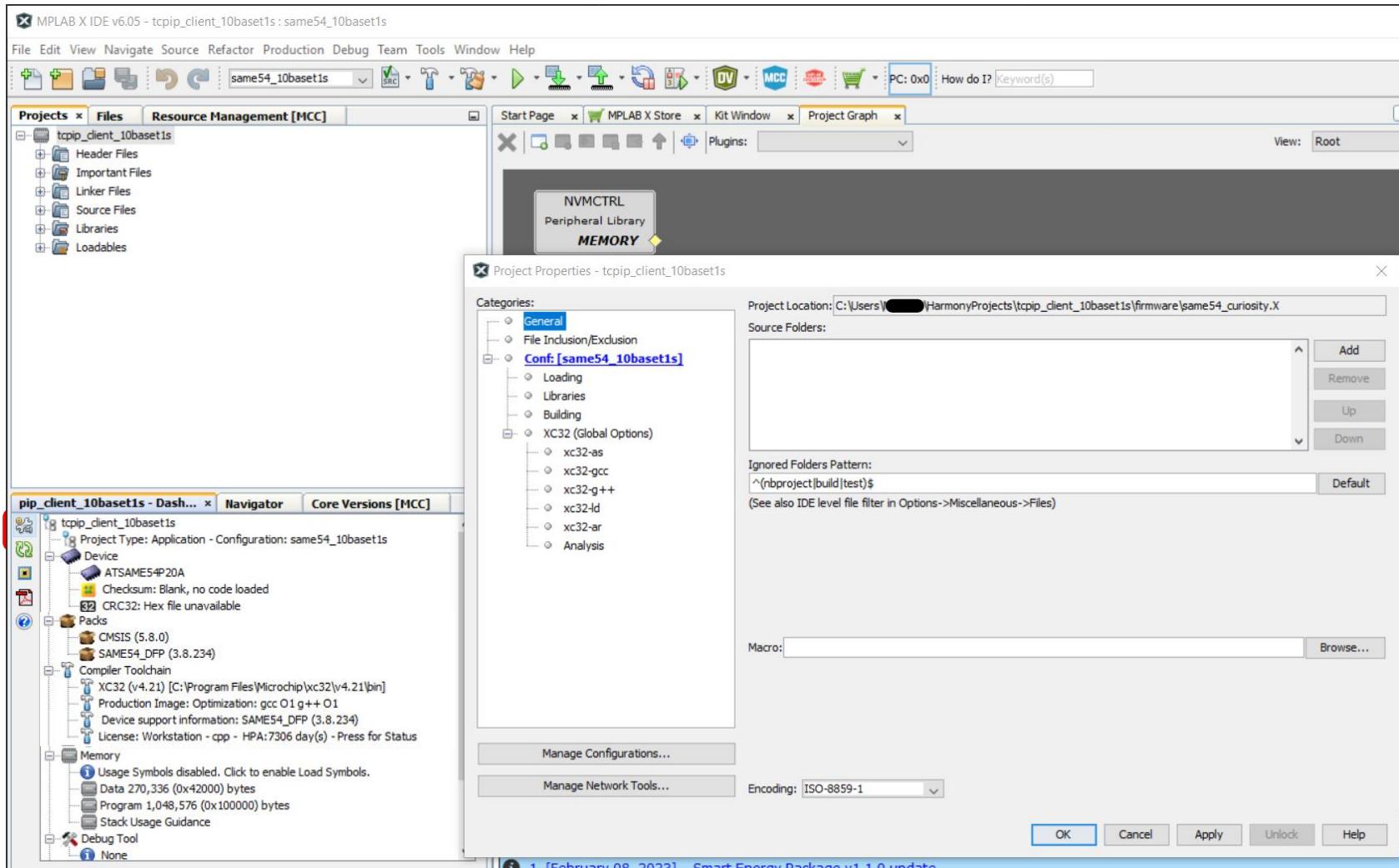
Check that you have the most up-to-date Device Pack and that your XC32 compiler is included in the project. To do this, open the Project Properties and view the Packs and Compiler Toolchain.



Step 14

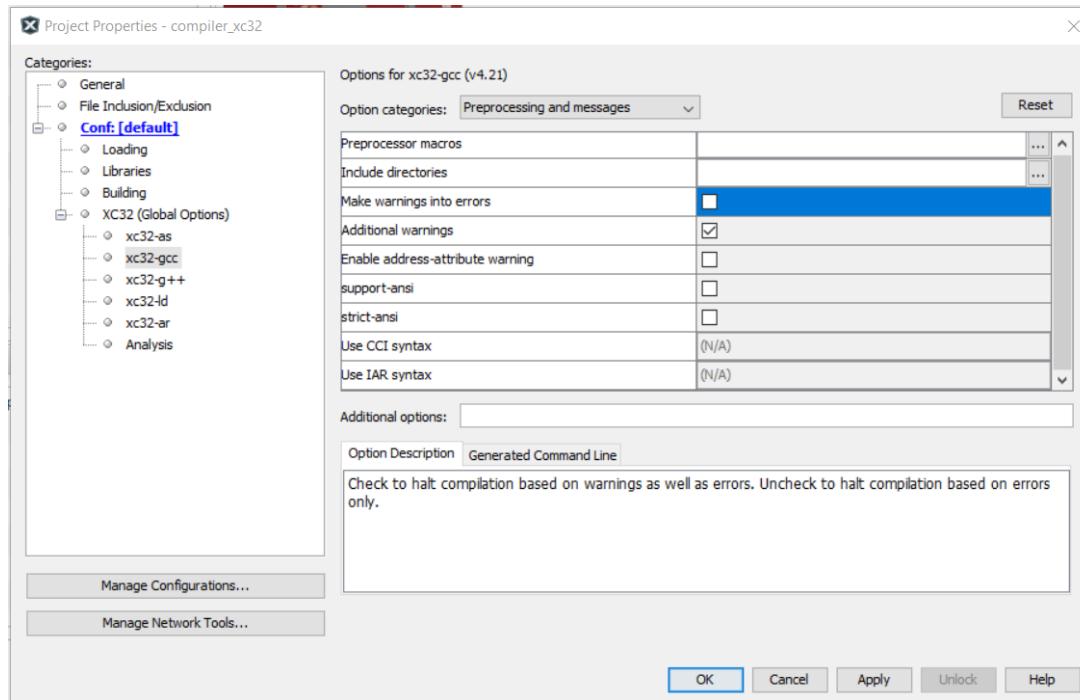
Now you are all set!
Take note of the location
of your project, you can
do this by opening the
Project Properties and
going to the General
category, and viewing
the Project Location.

You will be able to close
the project now, and re-
open it during the
10BaseT1S class.

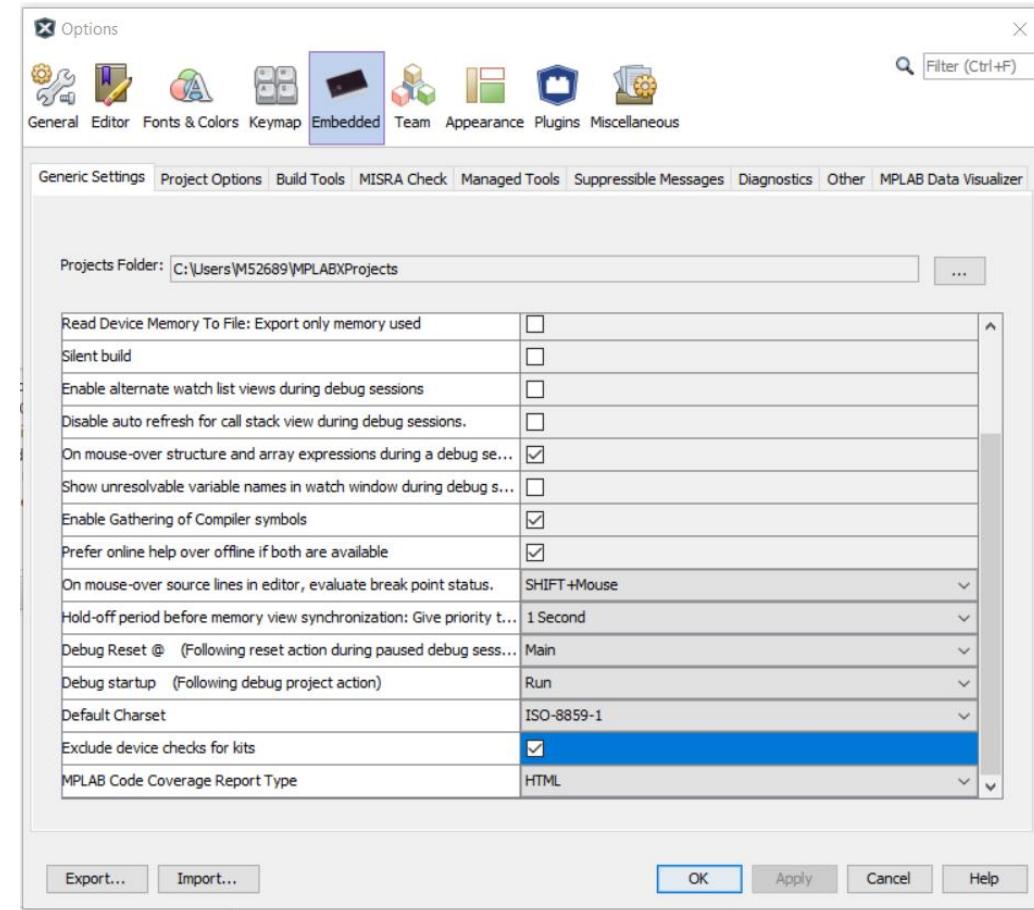


Additional settings that could cause problems!

In Project Properties, uncheck “Make warnings into errors”



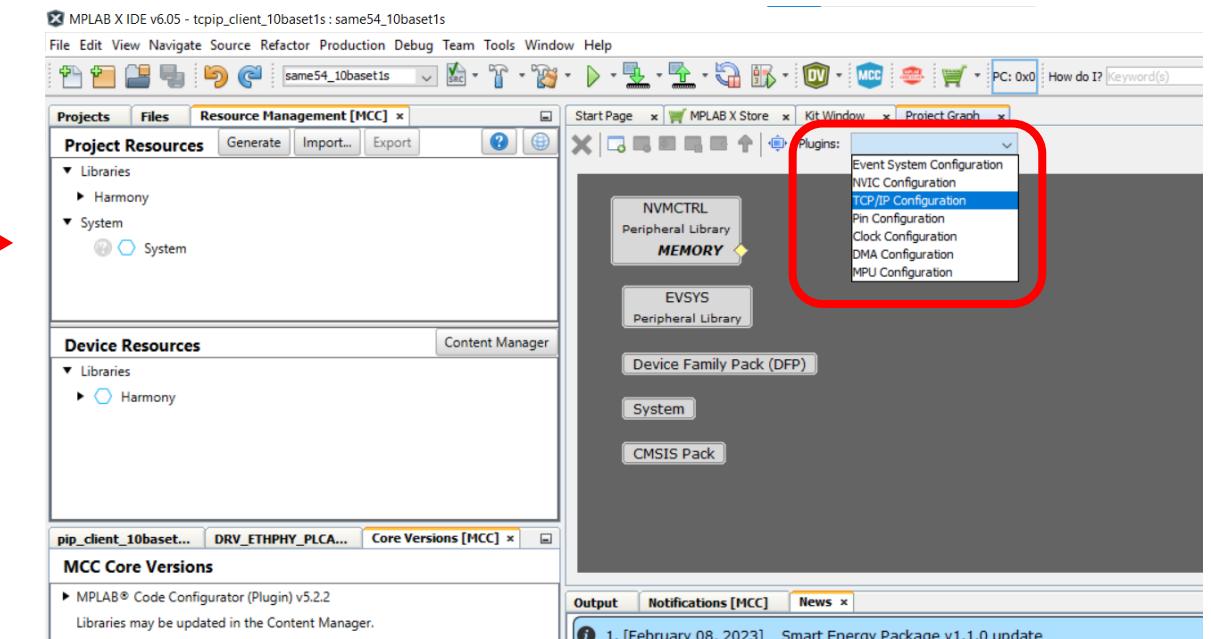
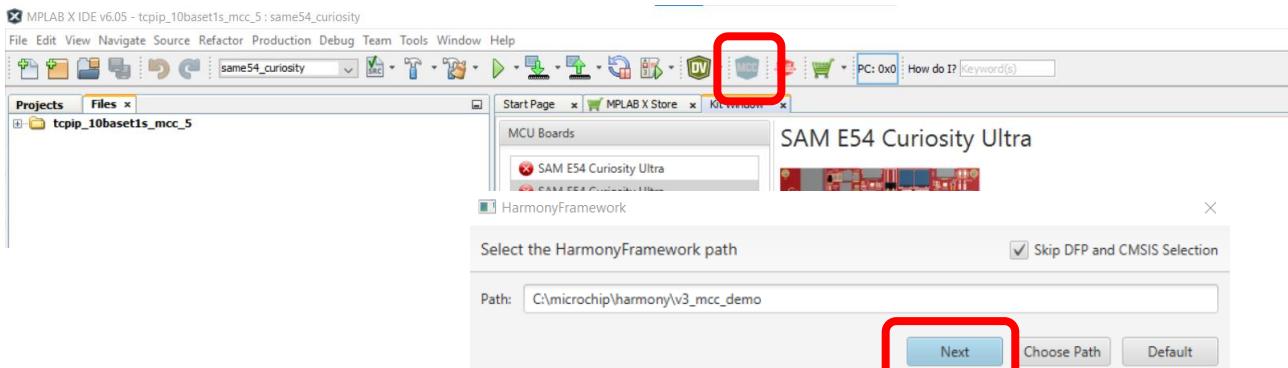
In Options > Embedded, check “Exclude device checks for kits”



Configuring the Project in MCC

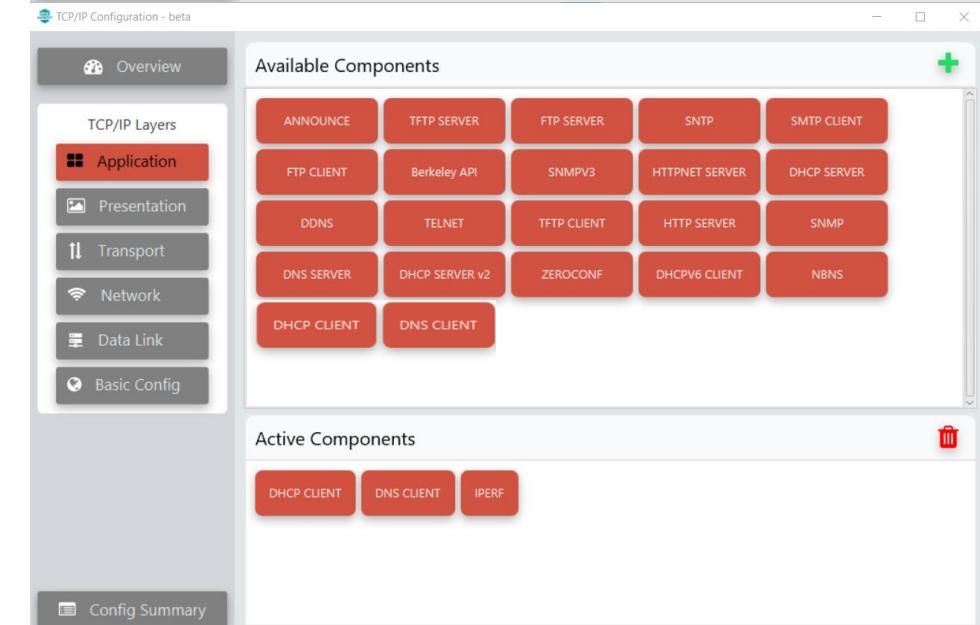
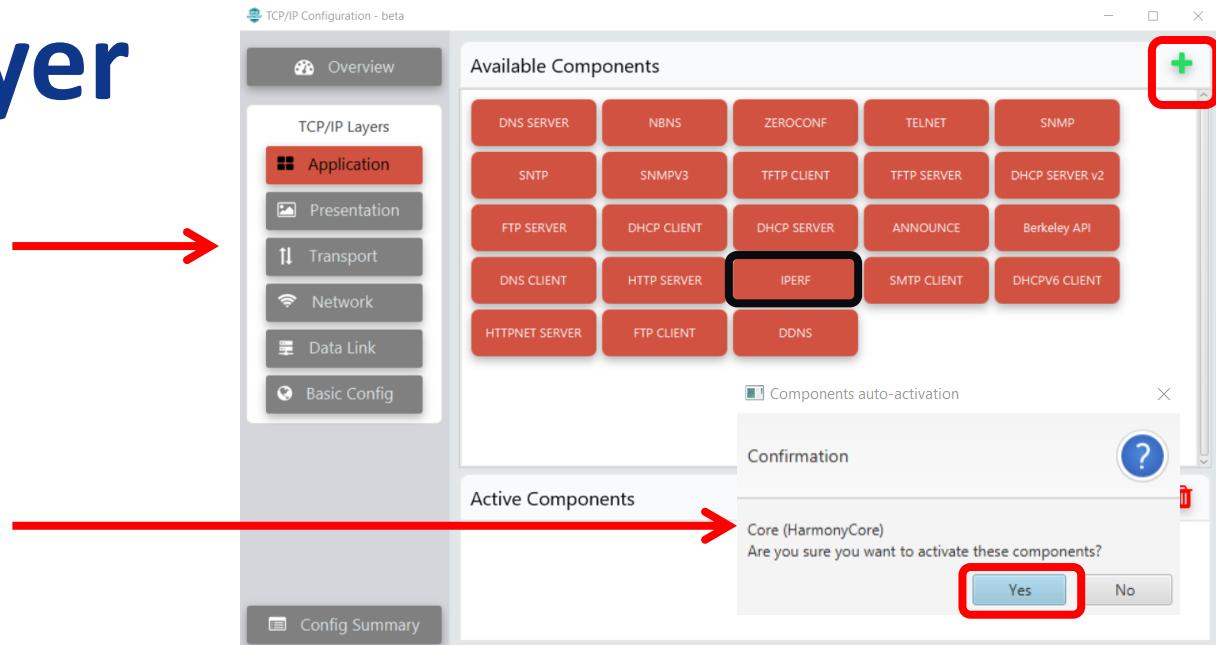
Open MCC and the TCP/IP Configuration Tool

- To open (or close) MCC, click on the blue MCC button
- Make sure to check that the Harmony Framework Path is correct in the popup box, as per Step 11



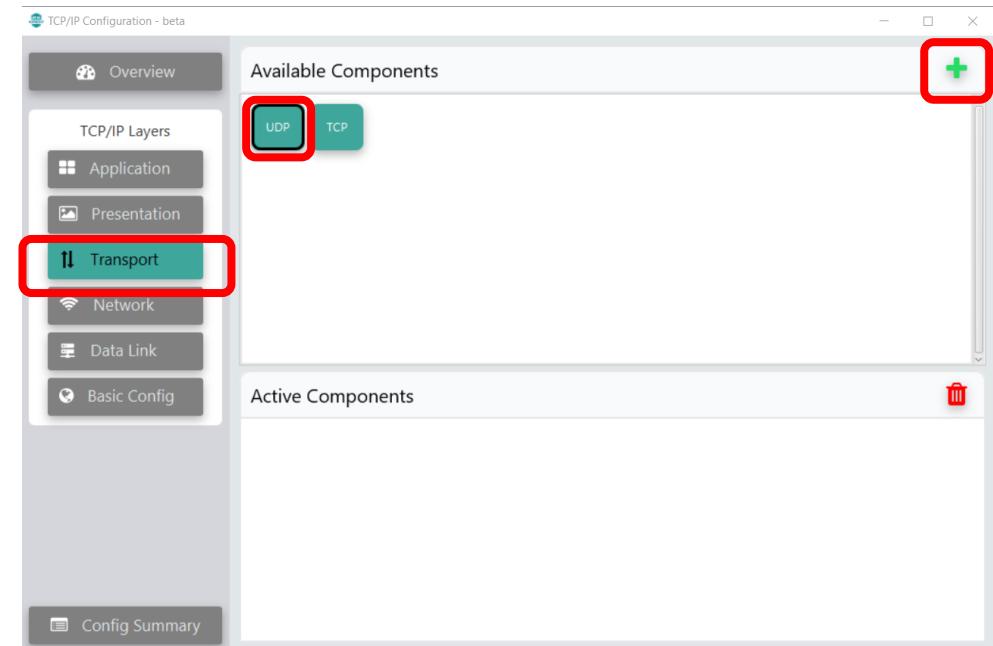
TCP/IP: Application Layer

- In the TCP/IP Configuration tool, in the Application Layer, Select IPERF and click +
- Click Yes to activate the Harmony Core component, and No for FreeRTOS (not needed for this demo)
- Also add DNS CLIENT and DHCP CLIENT

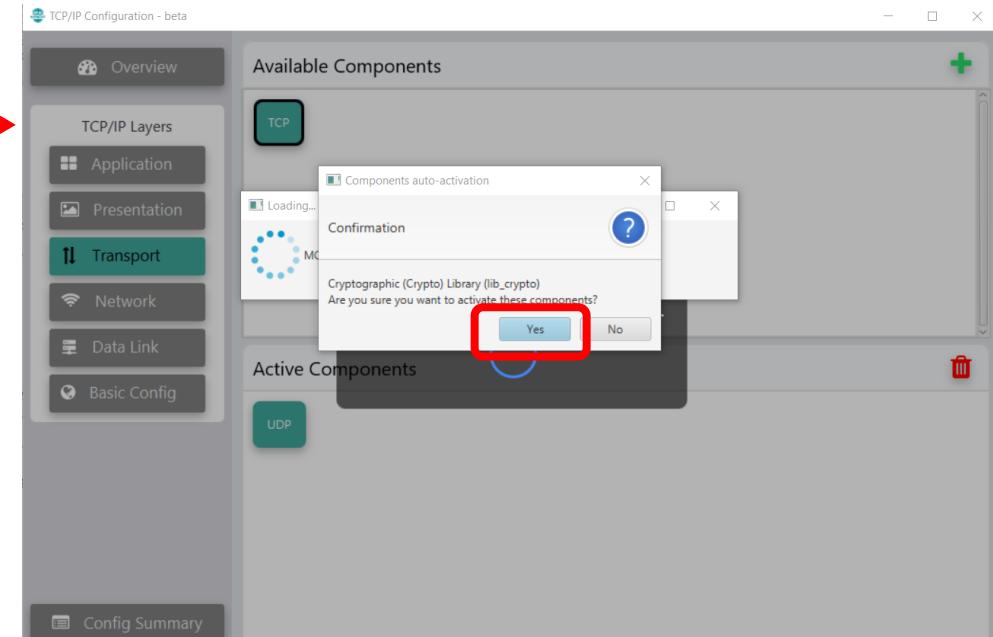
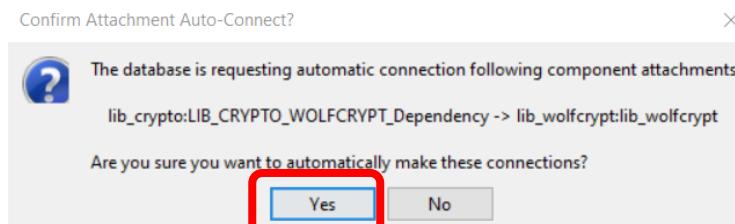


TCP/IP: Transport Layer

- In the Transport Layer, add the UDP Component in the same way



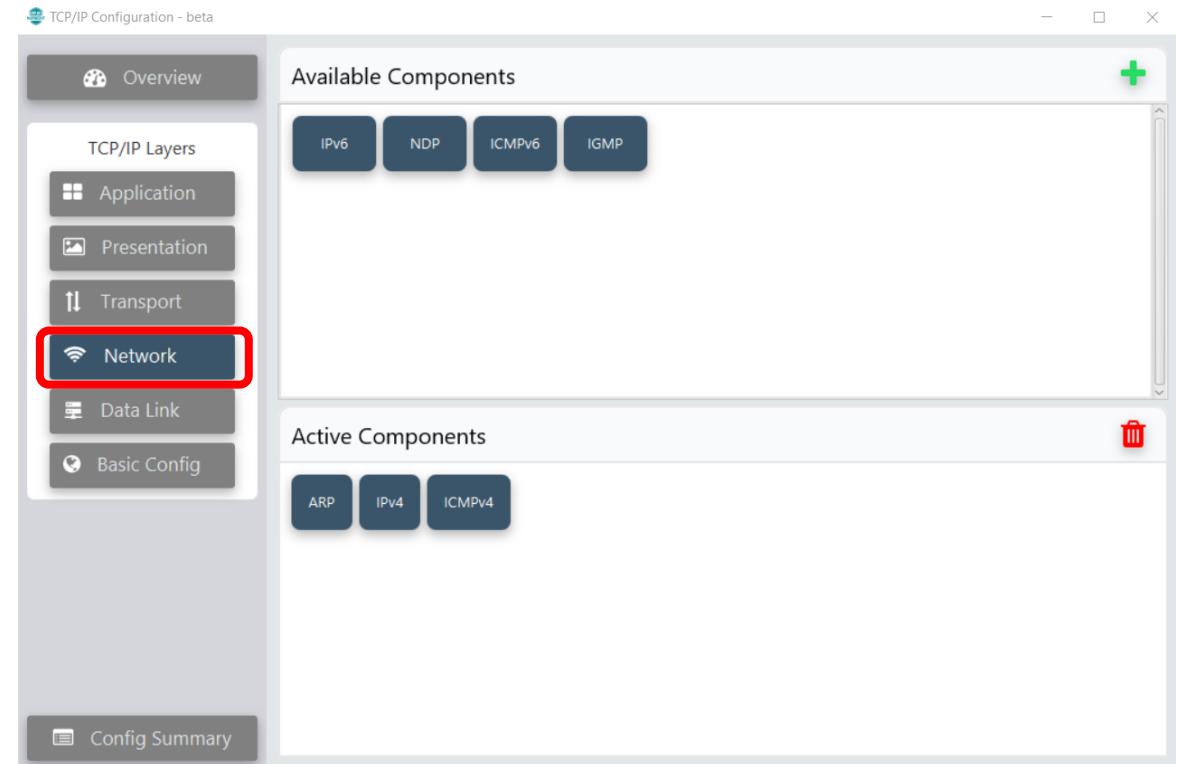
- Add the TCP Component, which needs the Crypto library and the WolfSSL libraries, click Yes to add these



Note: this pop-up sometimes is hidden behind the various MPLABX windows, you may not see it until a later point, just make sure to click Yes when it appears

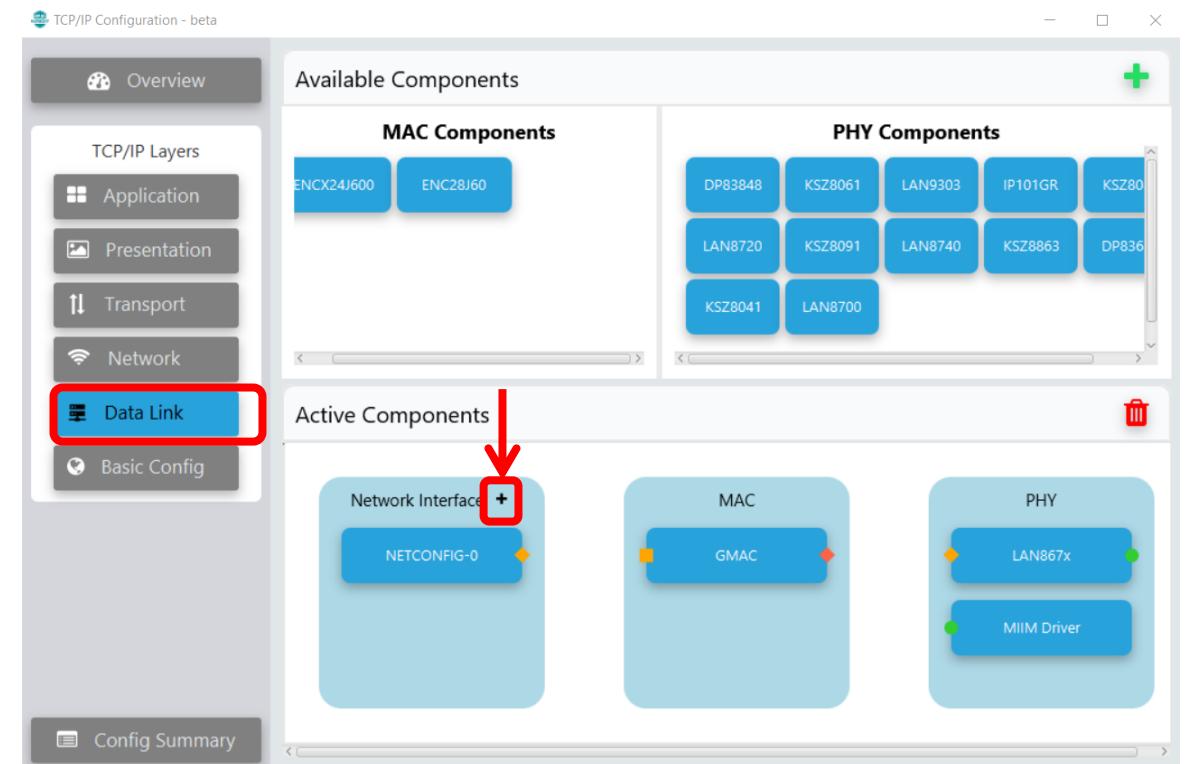
TCP/IP: Network Layer

- In the Network Layer, add the ARP, IPv4 and ICMPv4 components



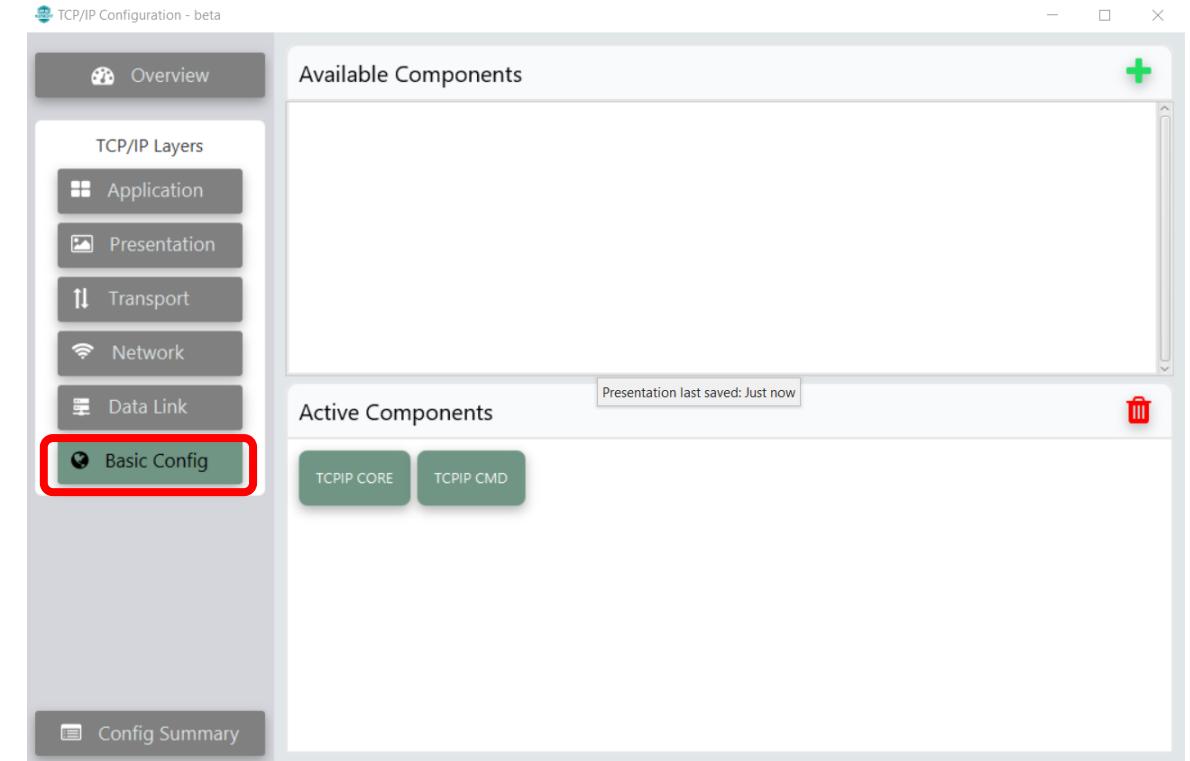
TCP/IP: Data Link Layer

- In the Data Link Layer, add the GMAC component, the LAN867x and MIIM Driver components
- Click the “+” beside Network Interface to add the NETCONFIG-0 component



TCP/IP: Basic Config

- In Basic Config, the TCPIP Core is there already
- Add the TCPIP CMD component



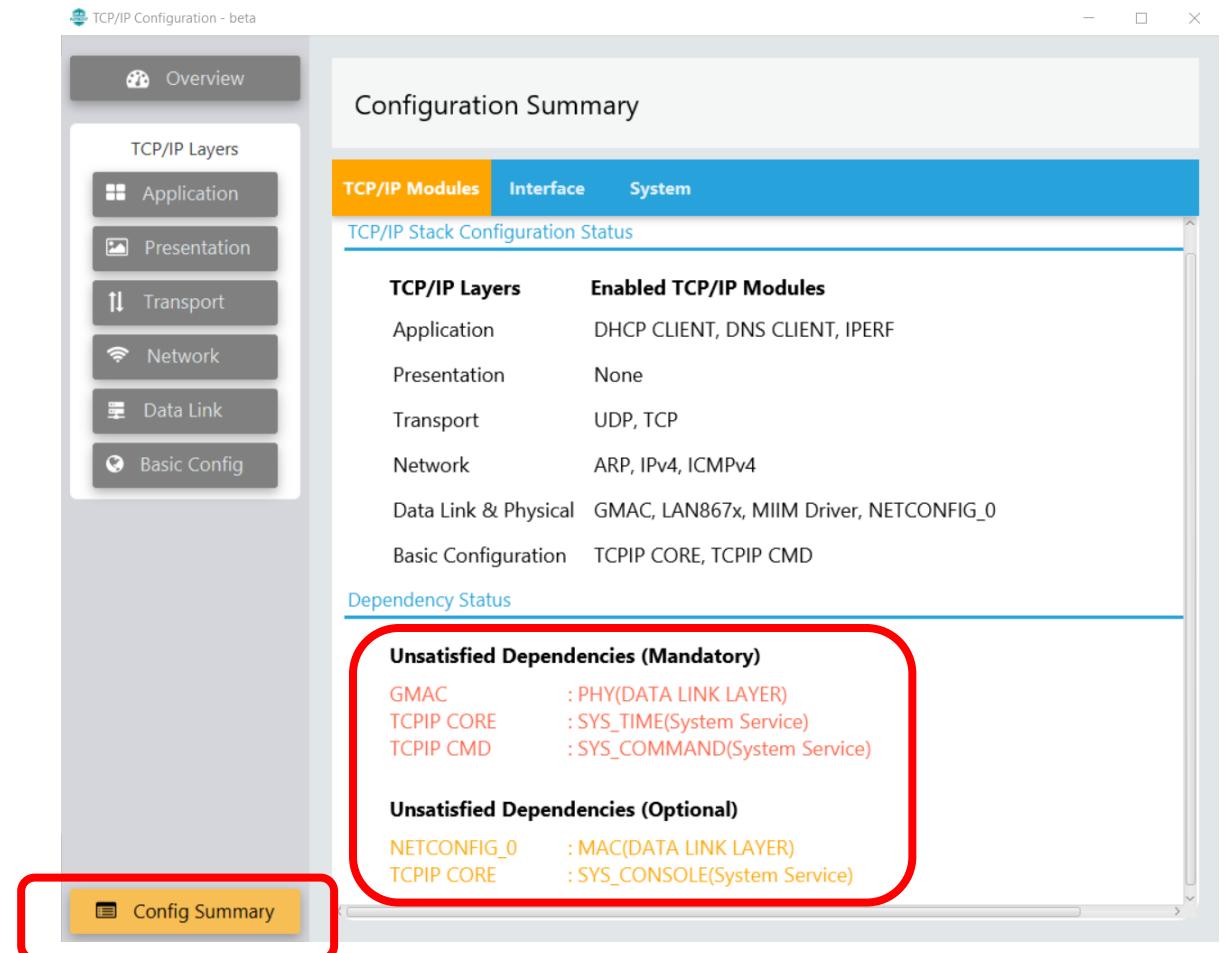
Overview

- The Overview of the TCP/IP Configurator should now be populated with the necessary components for the stack



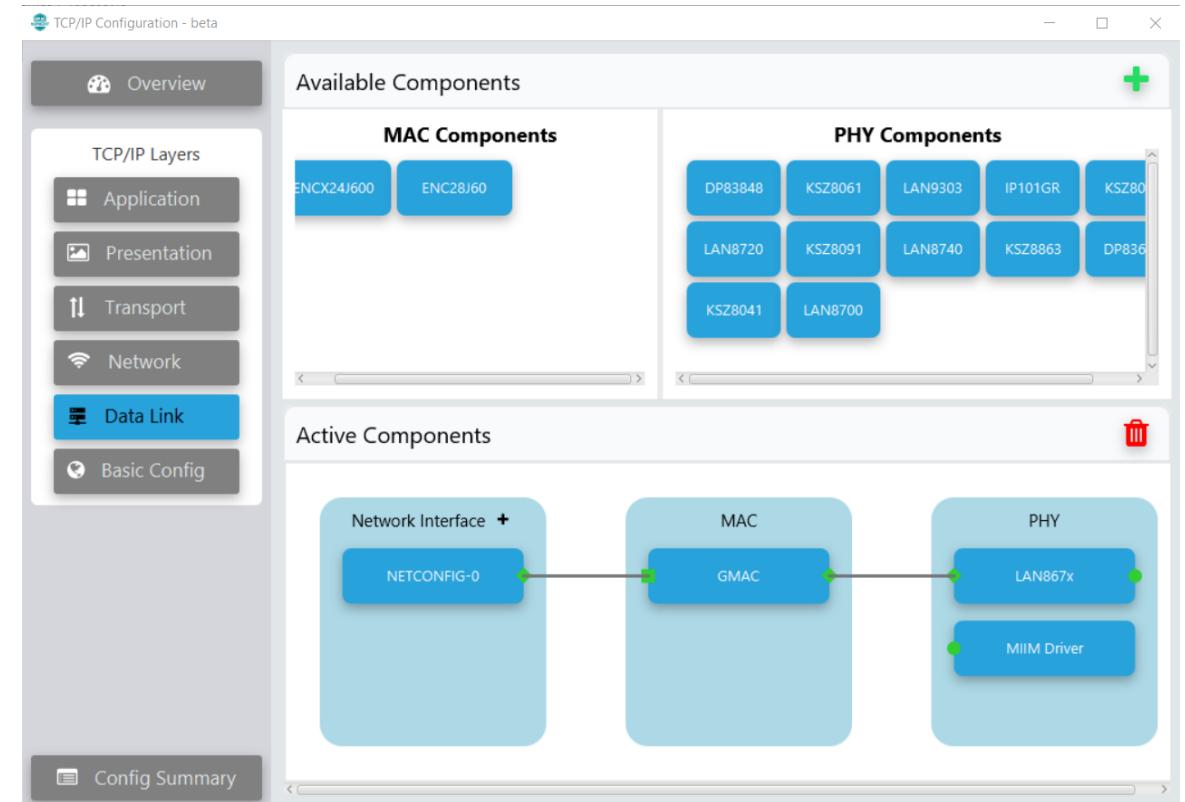
Config Summary

- Check the Config Summary to see what has been included
- Note the Unsatisfied Dependencies, these are elements of the project that the components require in order to work, and they must be satisfied.



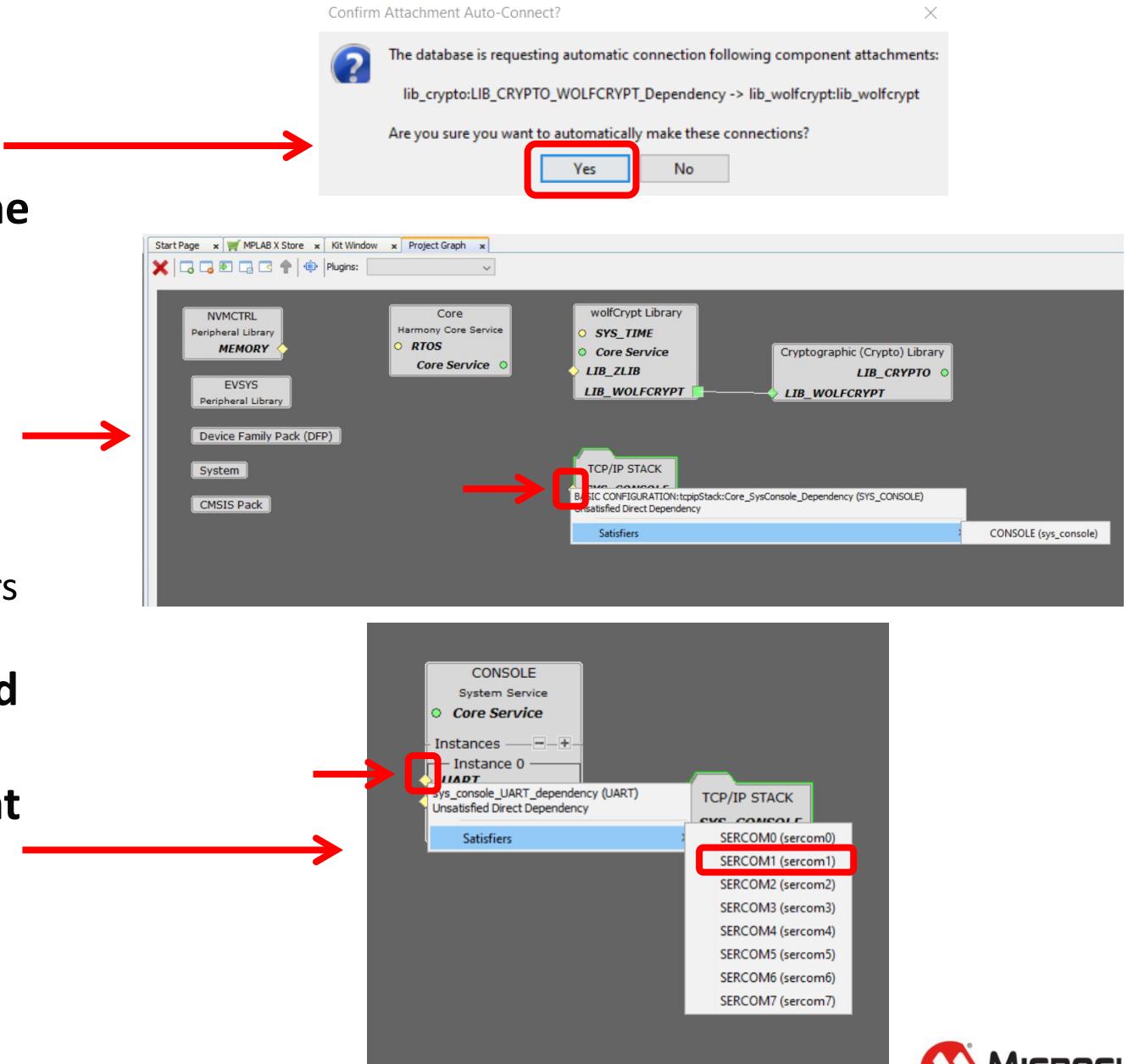
TCP/IP Data Link Dependencies

- In the Data Link Layer, connect NETCONFIG-0 to GMAC
- Connect GMAC to LAN867x PHY driver
 - To do these connections, simply use the mouse to “draw” a line between the red/orange dots on the components



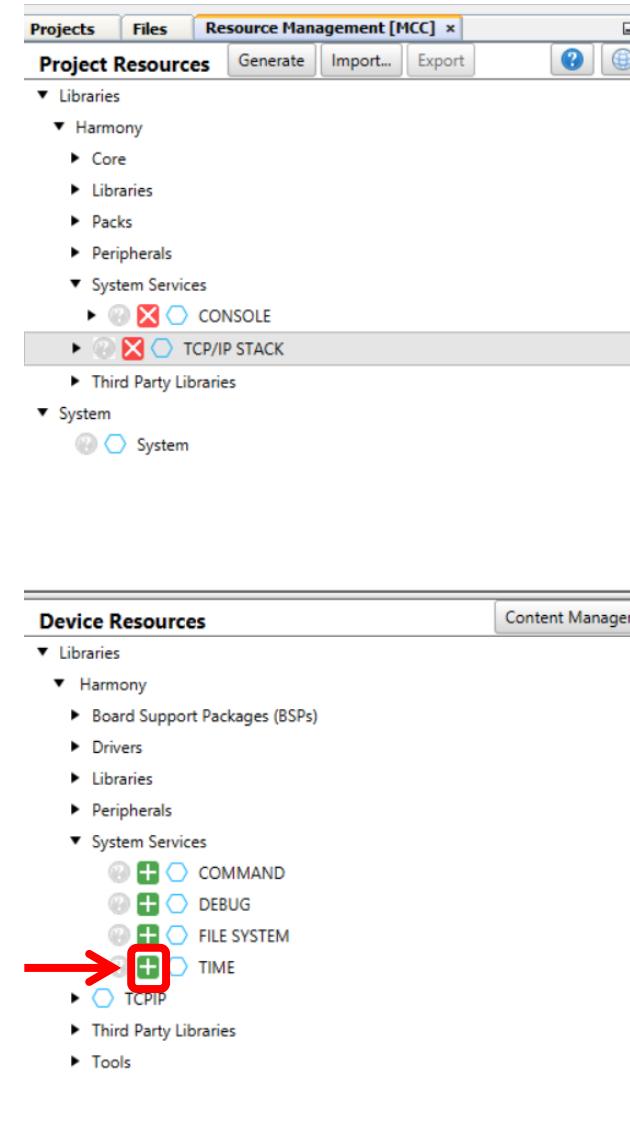
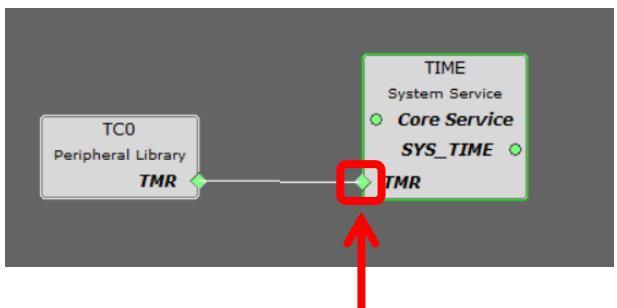
System Console

- Return the to Project Graph, you should see the TCP/IP Stack and the WolfCrypt components. (You may need to accept the Wolfcrypt dependency confirmation popup at this point if you have not already)
- Use the Project Graph to add the System Console to the project
 - Do this by right-clicking on the yellow diamond "SYS_CONSOLE" and select Satisfiers > CONSOLE (sys_console)
- The System Service Console is now added to the project graph
- This Console needs a UART instance, right click on the yellow diamond labelled "UART" and select SERCOM1



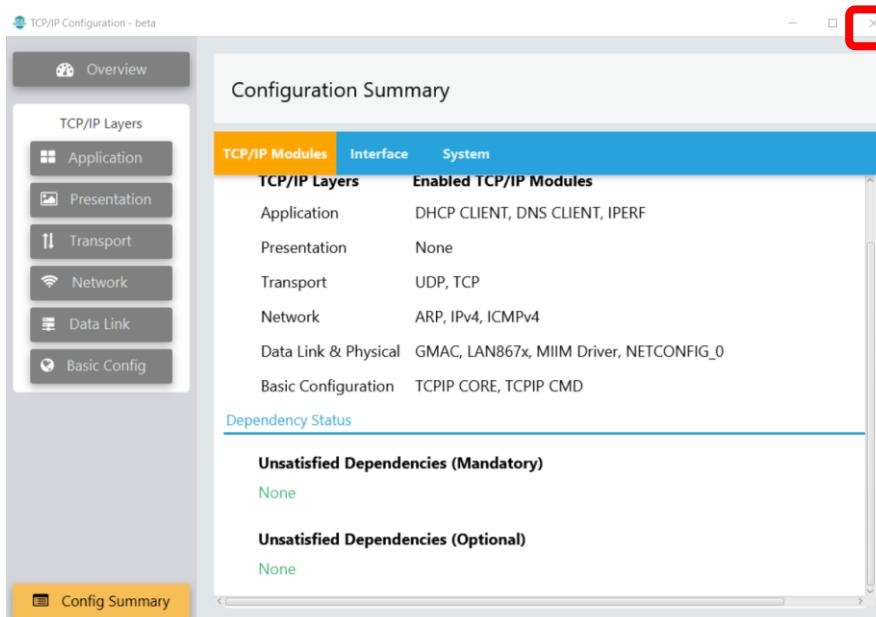
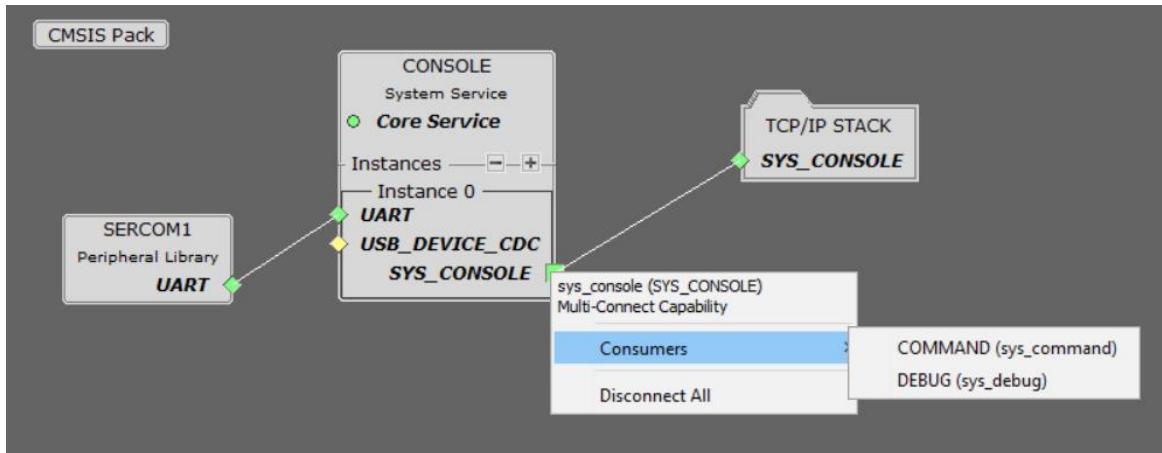
TIME System Service

- Add the TIME System Service from the Device Resources window, which is located in Libraries > Harmony > System Services
- Do this by clicking on the “+” beside the TIME system service
- Right click on the yellow diamond labelled “TMR” and select TC0 as the timer for the project (the diamond will turn green when TC0 is added)



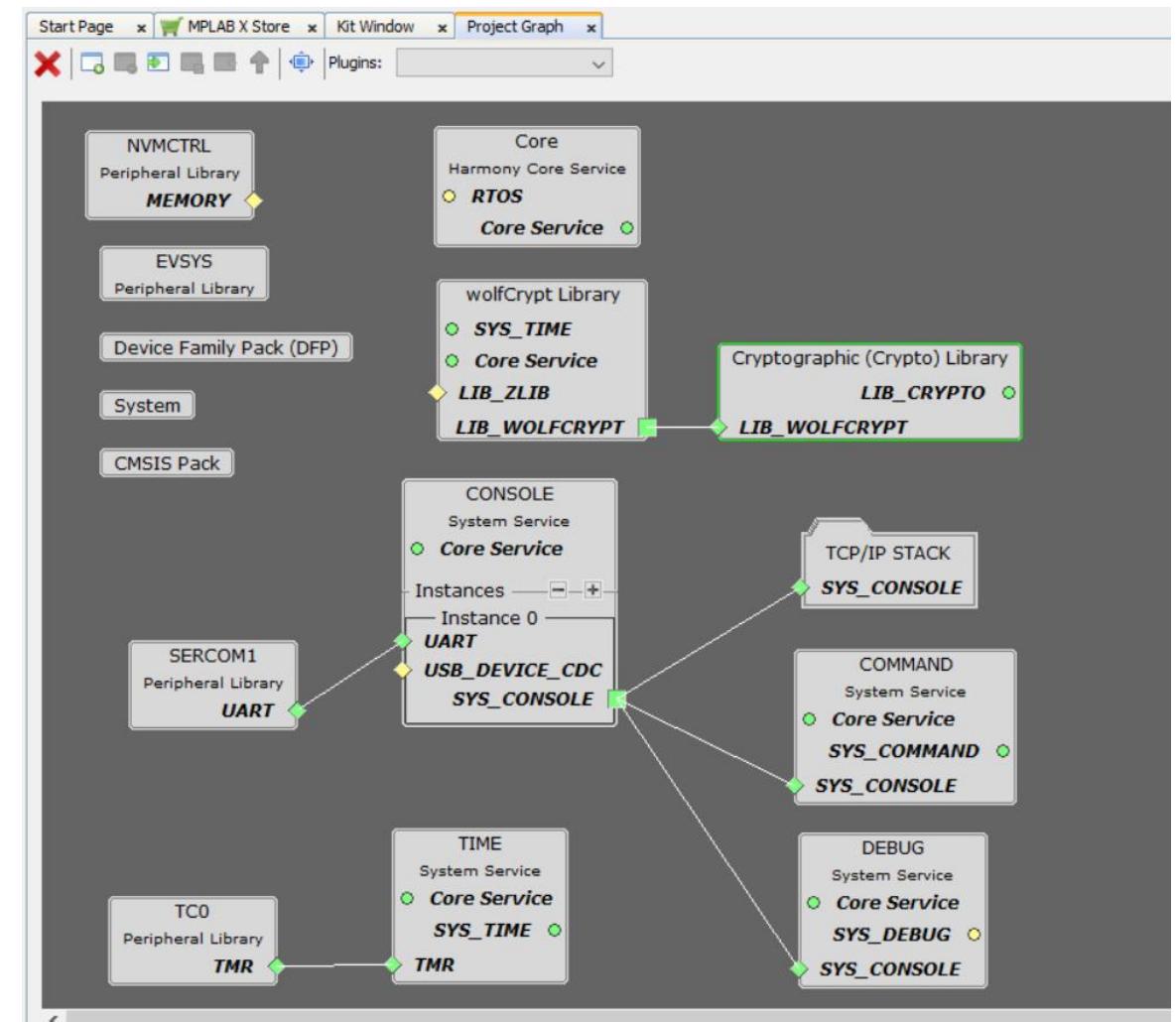
COMMAND and DEBUG

- Add the COMMAND (sys_command) and DEBUG (sys_debug) as Consumers for the Console System Service, along with the TCP/IP Stack.
- Do this by right clicking on the green square labelled “SYS_CONSOLE” and selecting these components, one at a time
- The TCP/IP Config Summary should now have no unsatisfied dependencies
- Close the TCP/IP Configuration Tool at this point



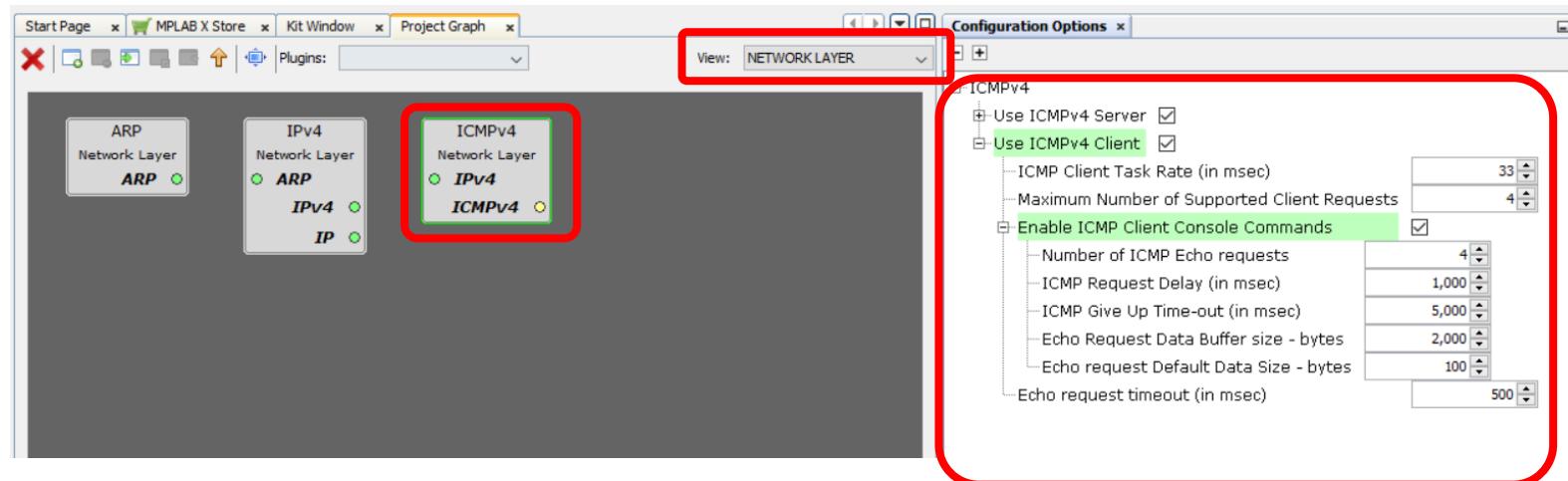
Project Graph

- The project graph should now contain all of the necessary components



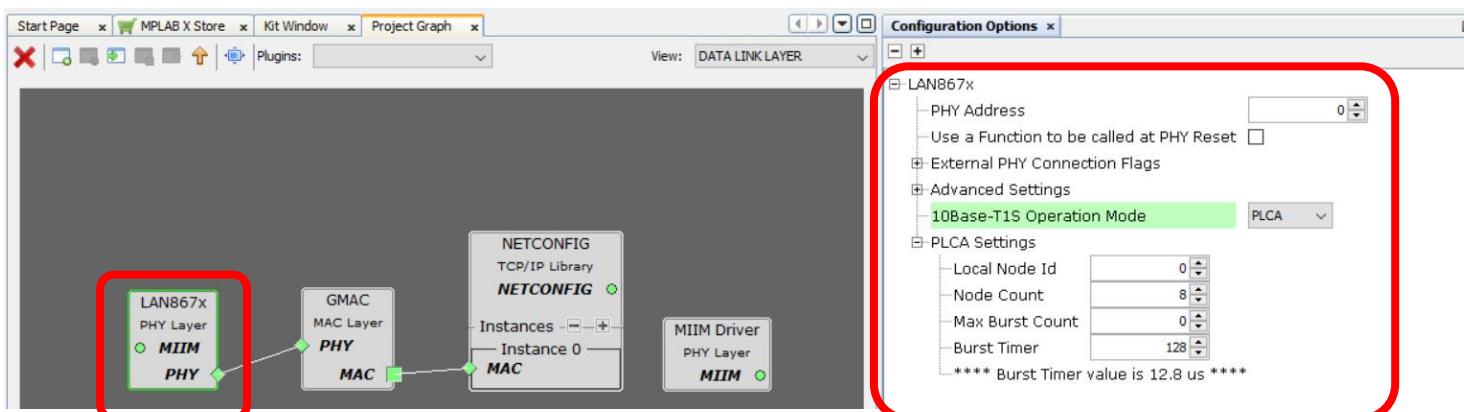
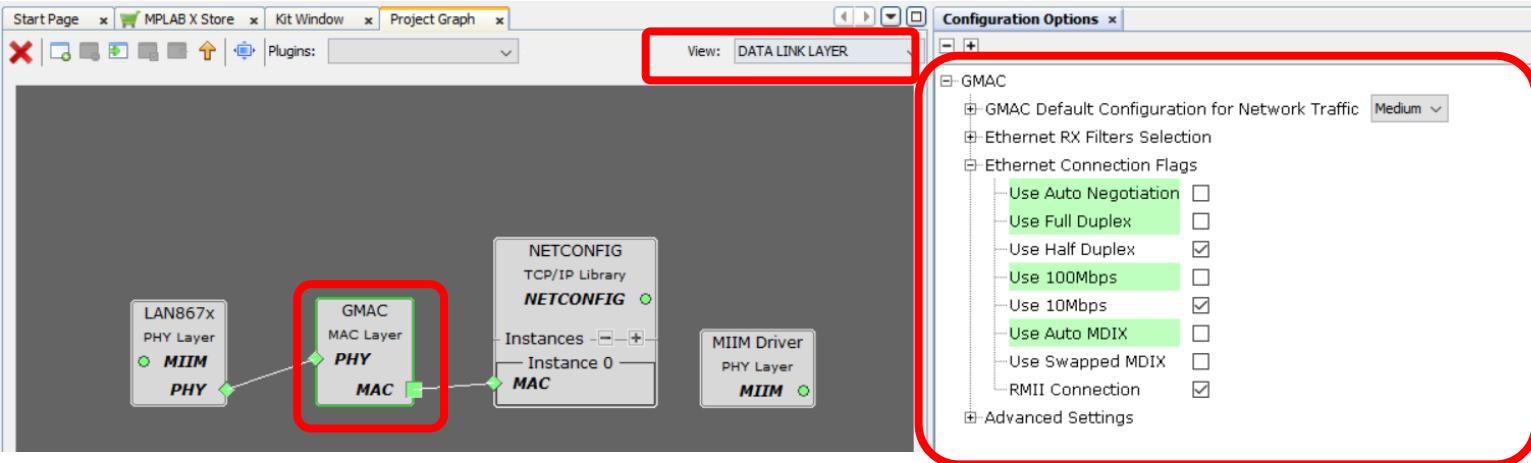
Network Layer Settings

- In the Network Layer, Select to Use ICMPv4 Client, and Enable ICMP Client Console Commands



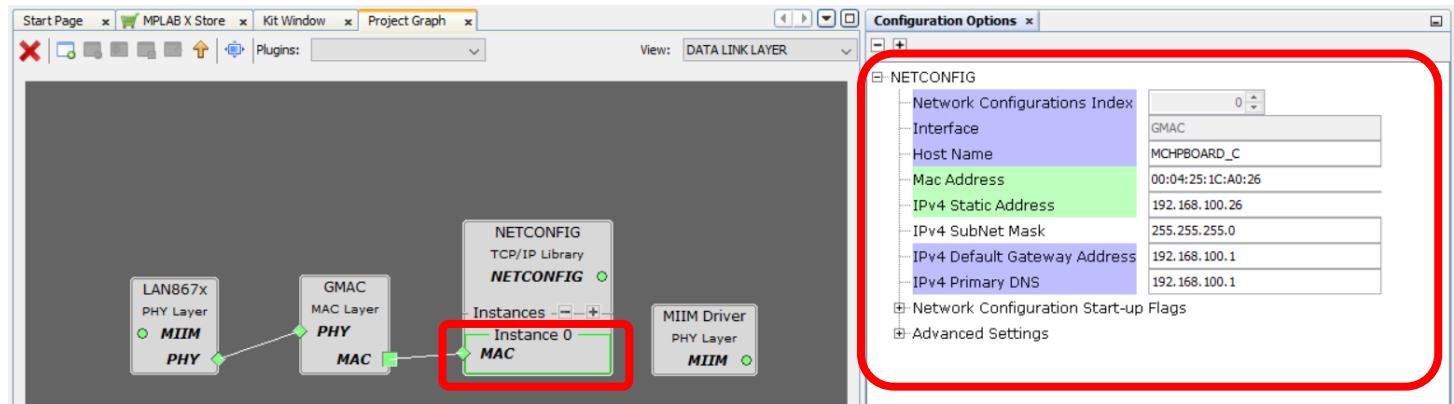
Data Link Layer Settings

- In the Data Link Layer, click on the GMAC component and set the Ethernet Connection Flags as shown
- Click on the LAN867x PHY Layer component and select the Advanced Settings to use PLCA, leave the other settings as is



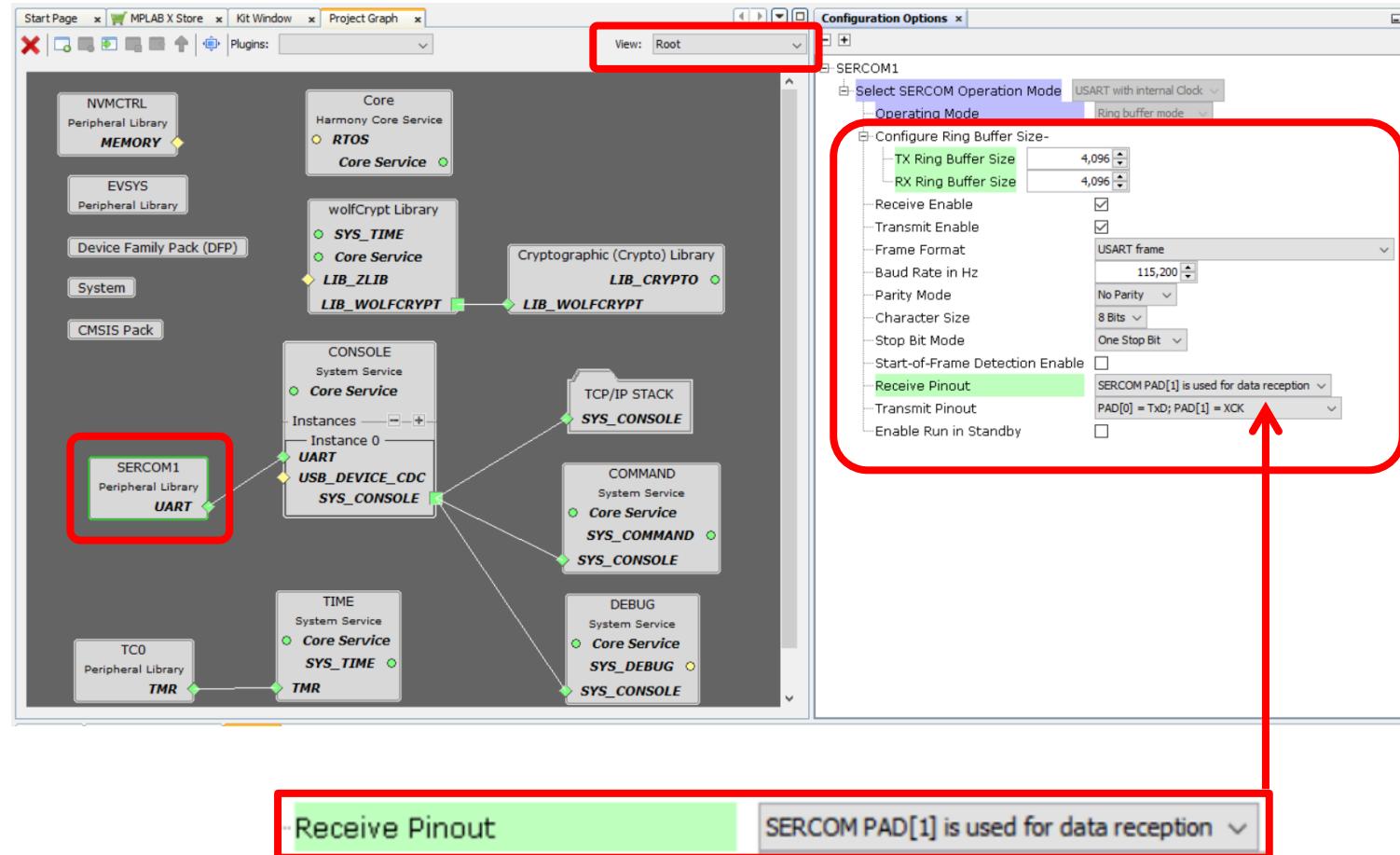
Network Configuration

- Observe the network configuration settings in the NETCONFIG component, change the MAC address and IP Address.
- Each of the two boards will need distinct MAC addresses
- They will also need distinct IP Addresses which will need to be on the same network subnet with distinct node numbers, here “26” is used



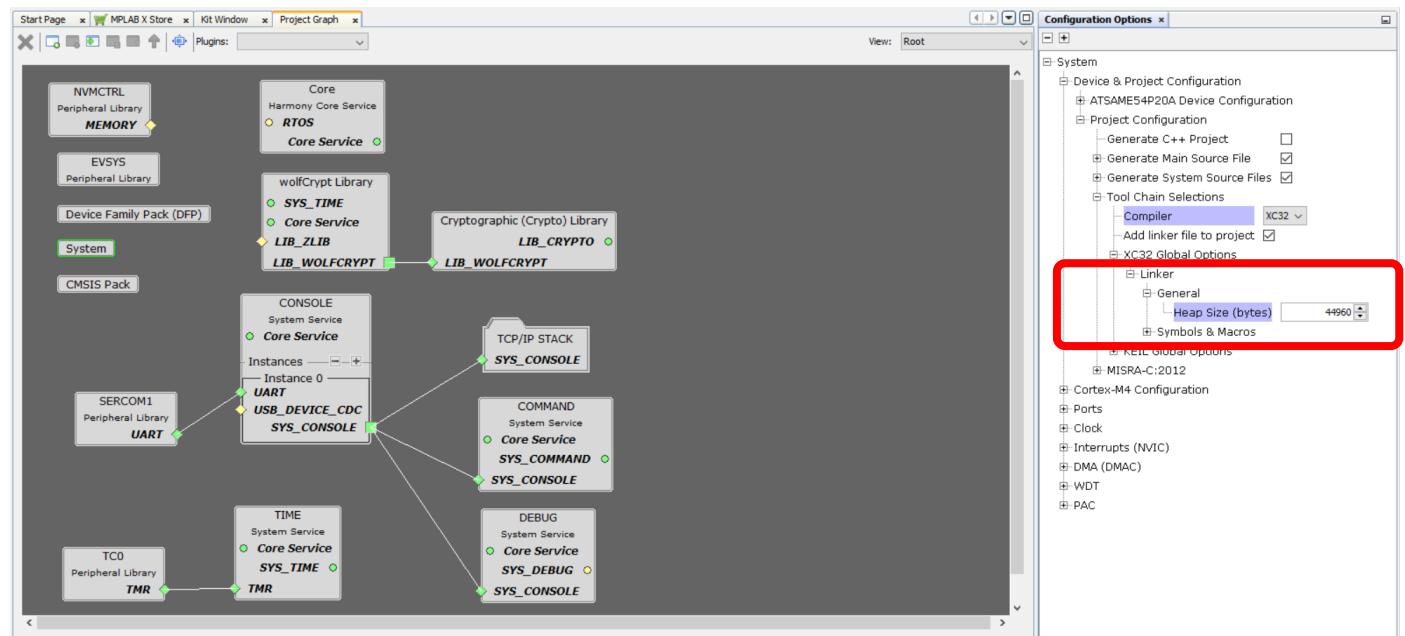
SERCOM Configuration

- In the Root layer of the Project Graph, configure SERCOM1 to increase the TX and RX Ring Buffer Size to 4096
- Set the Receive Pinout to: “SERCOM PAD[1] is used for data reception”



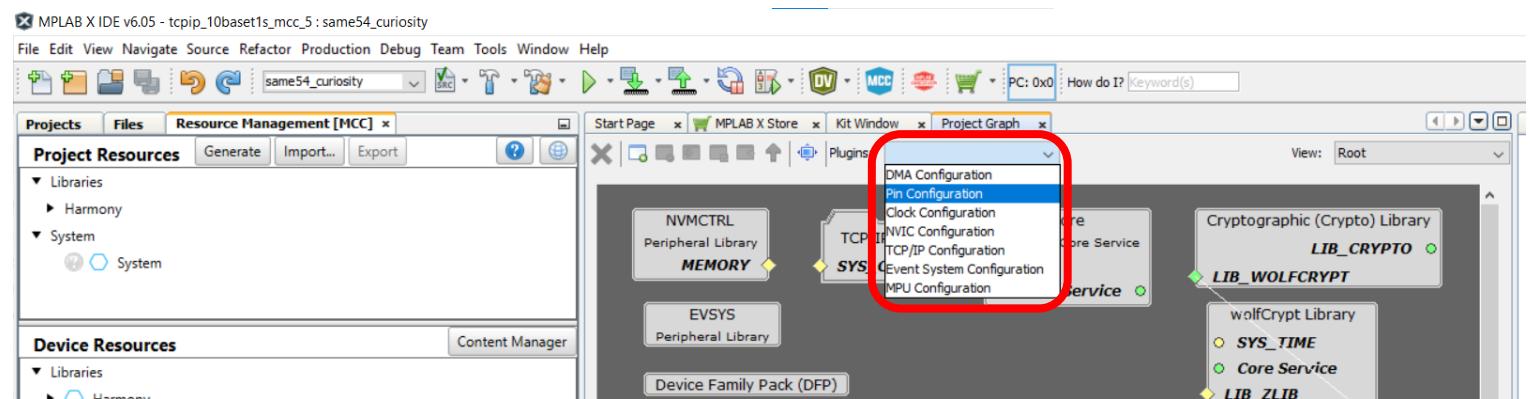
Heap Size

- Select the System component, and change the Heap Size to 44960 bytes as shown



Pin Configuration Tool

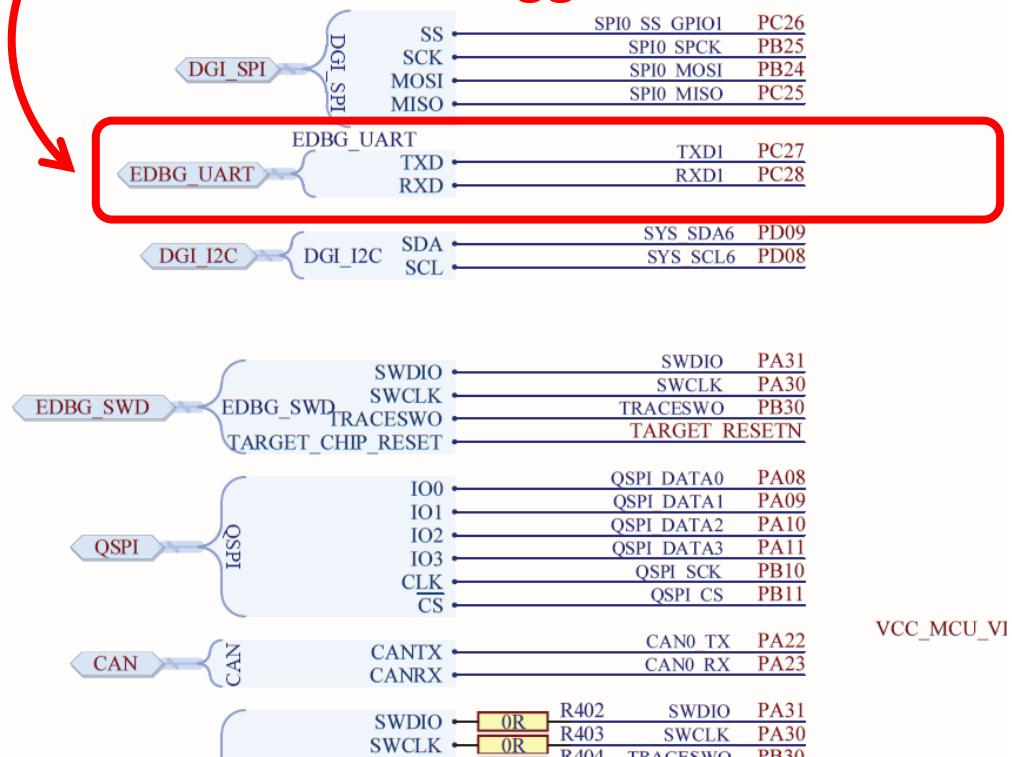
- Open the Pin Configuration Tool in the Plugins dropdown menu



Check the schematic to see the pin connections for the ethernet and UART pins (below are snapshots from Evaluation Board Schematic)

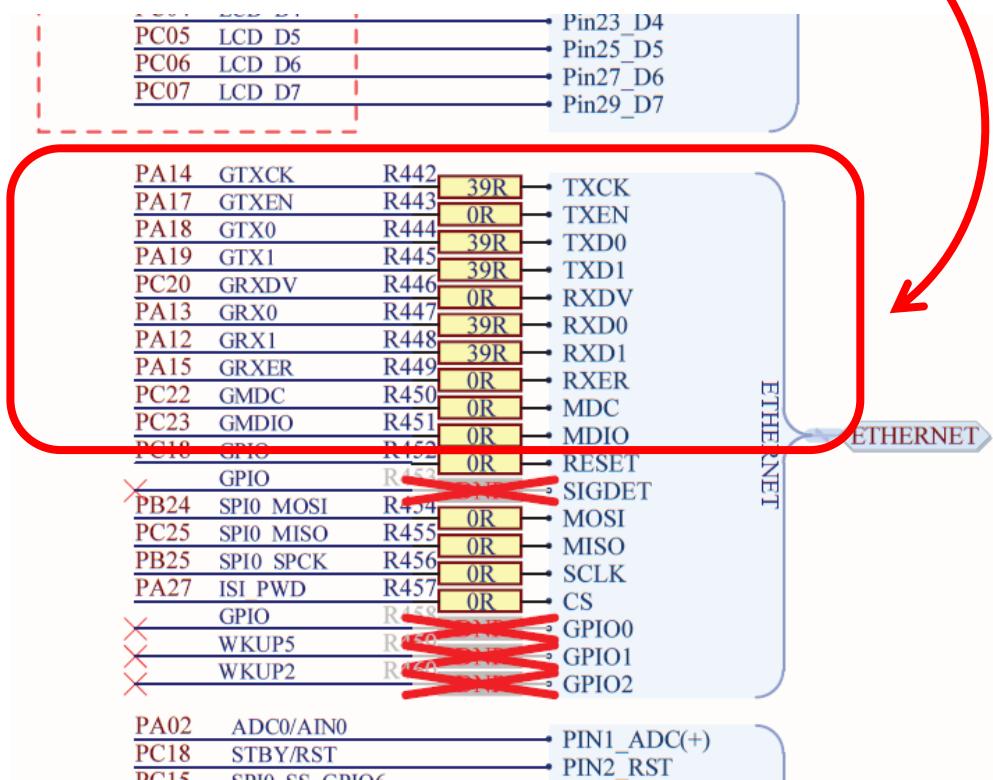
UART Pin Configuration

UART TX and RX lines via
Embedded Debugger



GMAC Pin Configuration

RMII Ethernet Signals
and Configuration



Pin Configuration Tool

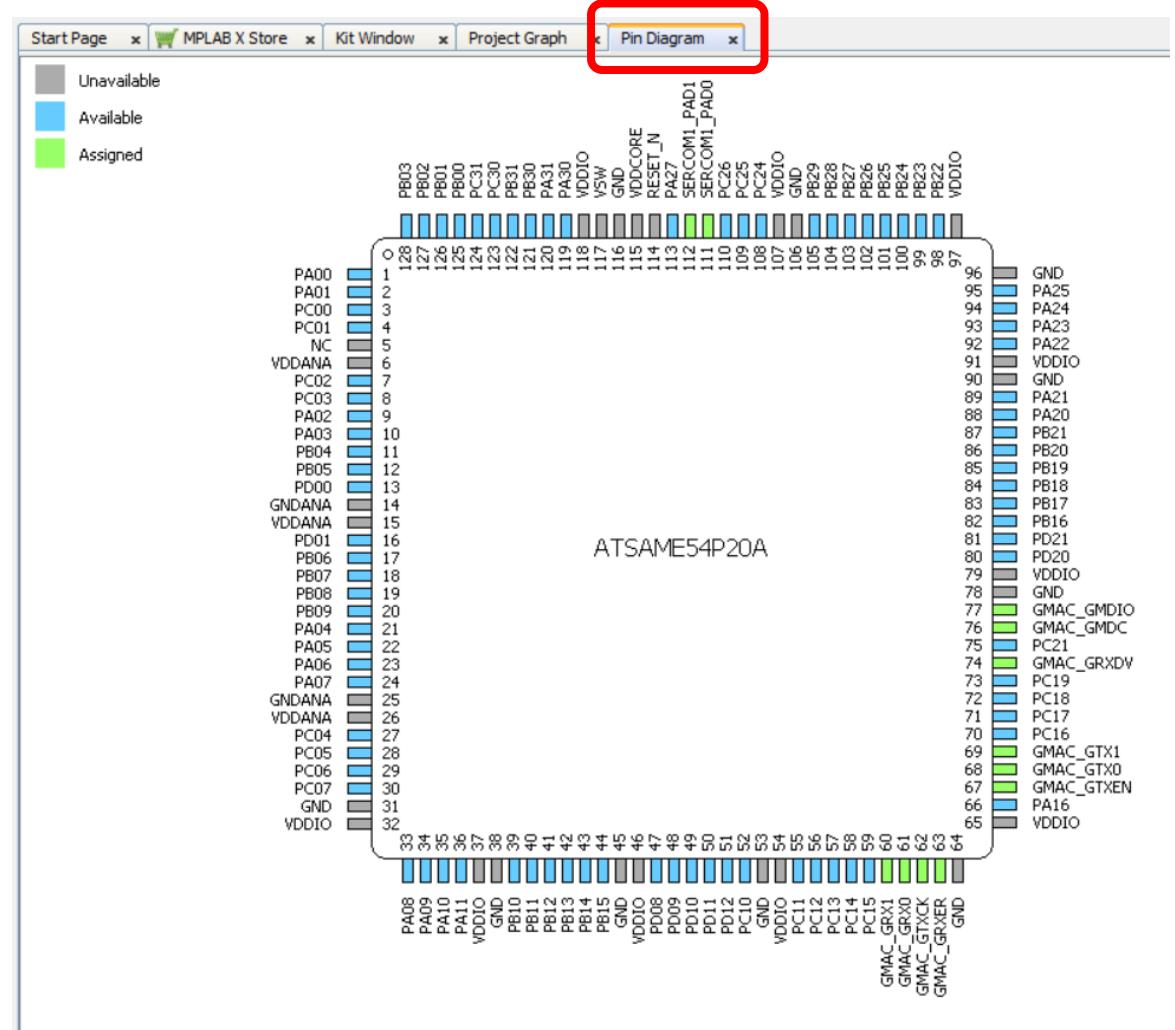
- Having checked the schematic to see the connections on the board, use the Pin Configuration Tool to set up the pins.
- The Pin Settings tab is the easiest way to do this. Select the values shown for PA12, PA13, PA14, PA15, PA17, PA18, PA19, PC20, PC22 and PC23, to match with the board connections.

Pin Number	Pin ID	Custom Name	Function	Mode	Direction	Latch	Pull Up	Pull Down	Drive Strength
57	PC13		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
58	PC14		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
59	PC15		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
60	PA12	GMAC_GRX1	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
61	PA13	GMAC_GRX0	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
62	PA14	GMAC_GTXCK	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
63	PA15	GMAC_GRXER	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
64	GND		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
65	VDDIO		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
66	PA16		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
67	PA17	GMAC_GTXEN	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
68	PA18	GMAC_GTX0	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
69	PA19	GMAC_GTX1	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
70	PC16		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
71	PC17		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
72	PC18		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
73	PC19		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
74	PC20	GMAC_GRXDV	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
75	PC21		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
76	PC22		GMAC_GMDC	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
77	PC23		GMAC_GMDIO	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
78	GND		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
79	VDDIO		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
80	PD20		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
81	PD21		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
82	PB16		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
83	PB17		Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL

Pin	Pad	Available	Digital	High Impedance	Low				Normal
110	PC26	Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
111	PC27	SERCOM1_PAD0	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
112	PC28	SERCOM1_PAD1	Digital	High Impedance	n/a	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL
113	PA27	Available	Digital	High Impedance	Low	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NORMAL

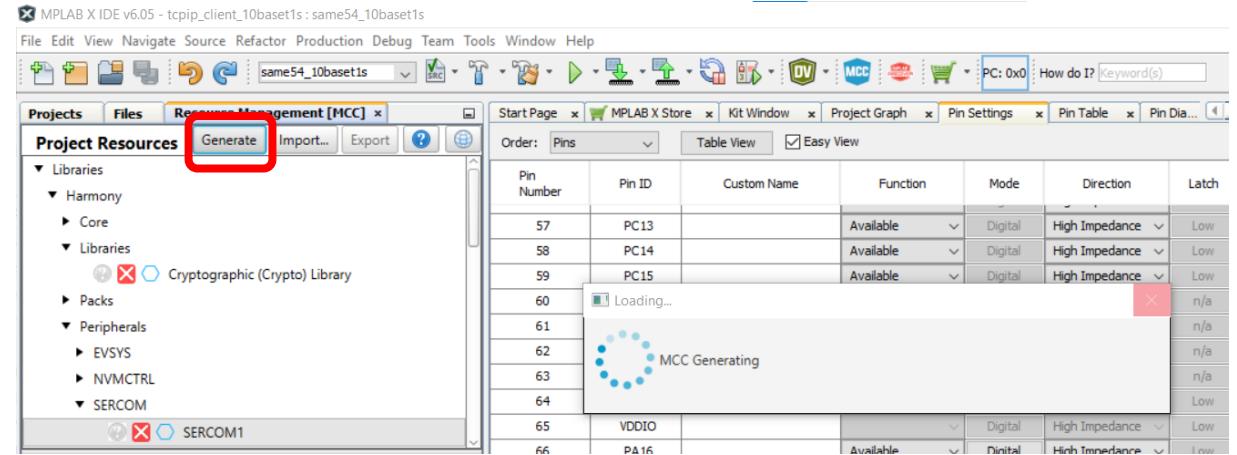
Pin Diagram

- The Pin Diagram should now look as shown – the pins that are green are the ones that have been configured using the Pin Settings



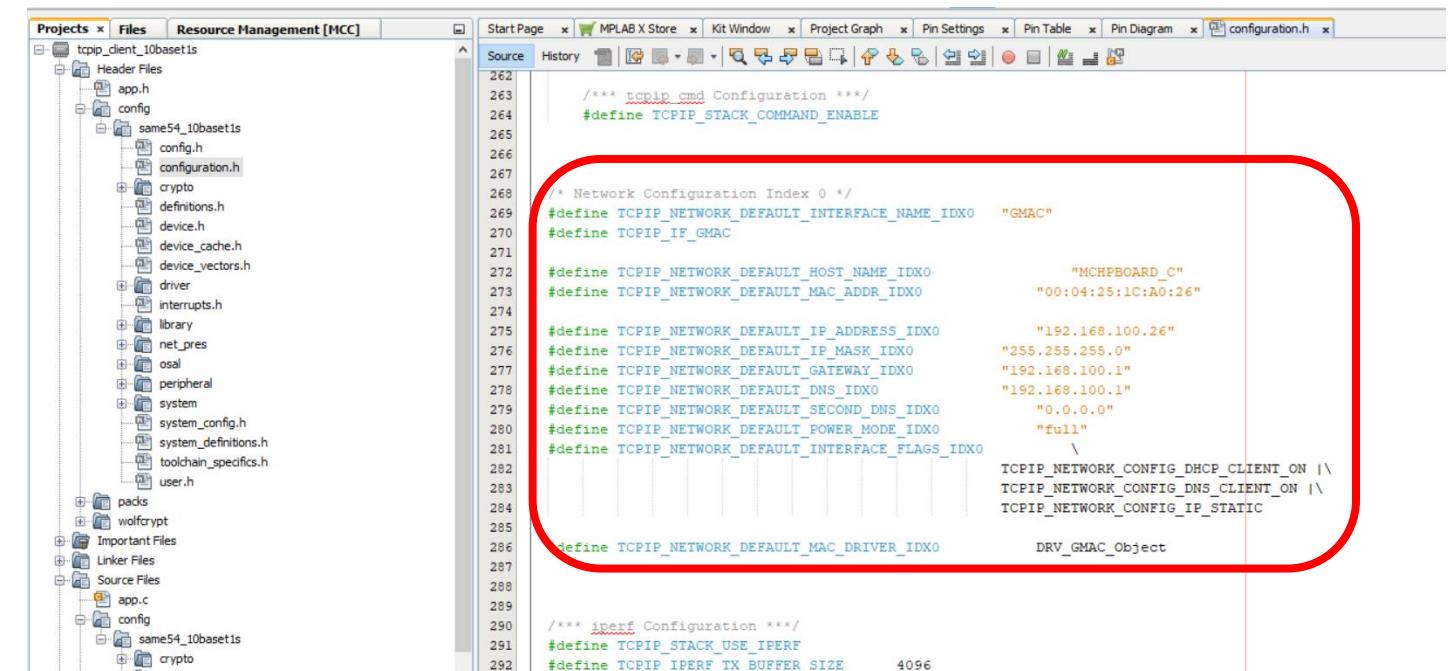
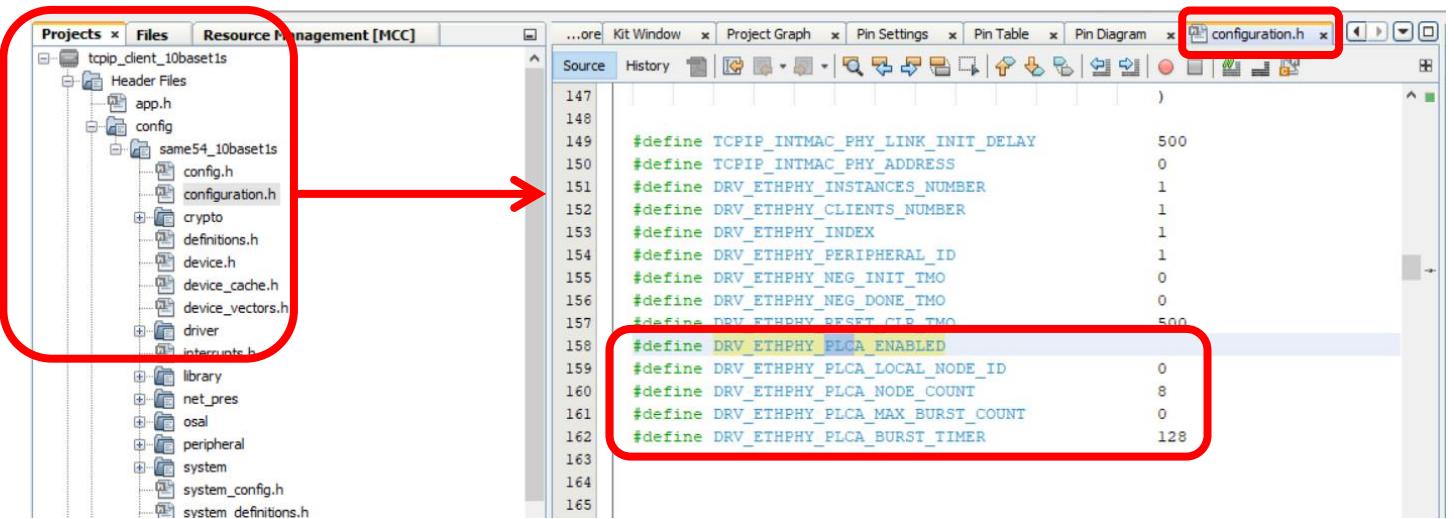
Generate the Code

- In Resource Management[MCC], use the Generate button to create the code for the project



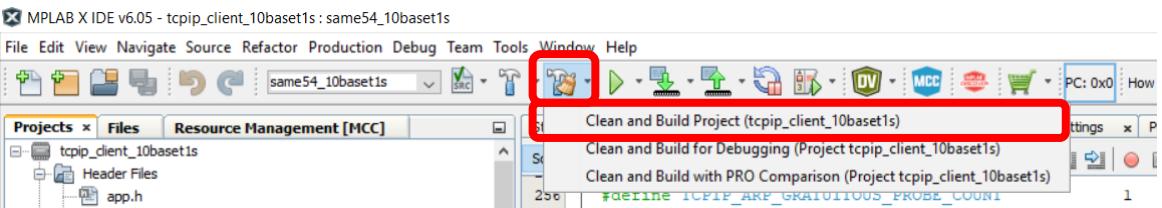
Configuration.h

- The Project is now populated
- Check the configuration.h Header File
 - The PLCA settings and the Network Configuration settings can be seen in this file
 - Check that the PLCA settings are correct
 - Check that the IP Address and MAC Address are correct
 - These can be changed in this file if necessary
 - But any changes in this file will not be reflected back into the MCC Project Graph settings



Build the Project and Program the Device

- Clean and Build the Project



- Check the output of the Build to make sure it was successful, similar to what is shown.



```
tcpip_client_10base1s (Clean, Build, ...)

Info: Loading file: ../../src/config/same54_10base1s\ATSAME54P20A.ld
"C:\Program Files\Microchip\xc32\v4.21\bin"\xc32-gcc.exe
make[2]: Leaving directory 'C:/Users/M52689/HarmonyProjects/tcpip_client_10base1s/firmware/same54_curiosity.X'
make[1]: Leaving directory 'C:/Users/M52689/HarmonyProjects/tcpip_client_10base1s/firmware/same54_curiosity.X'

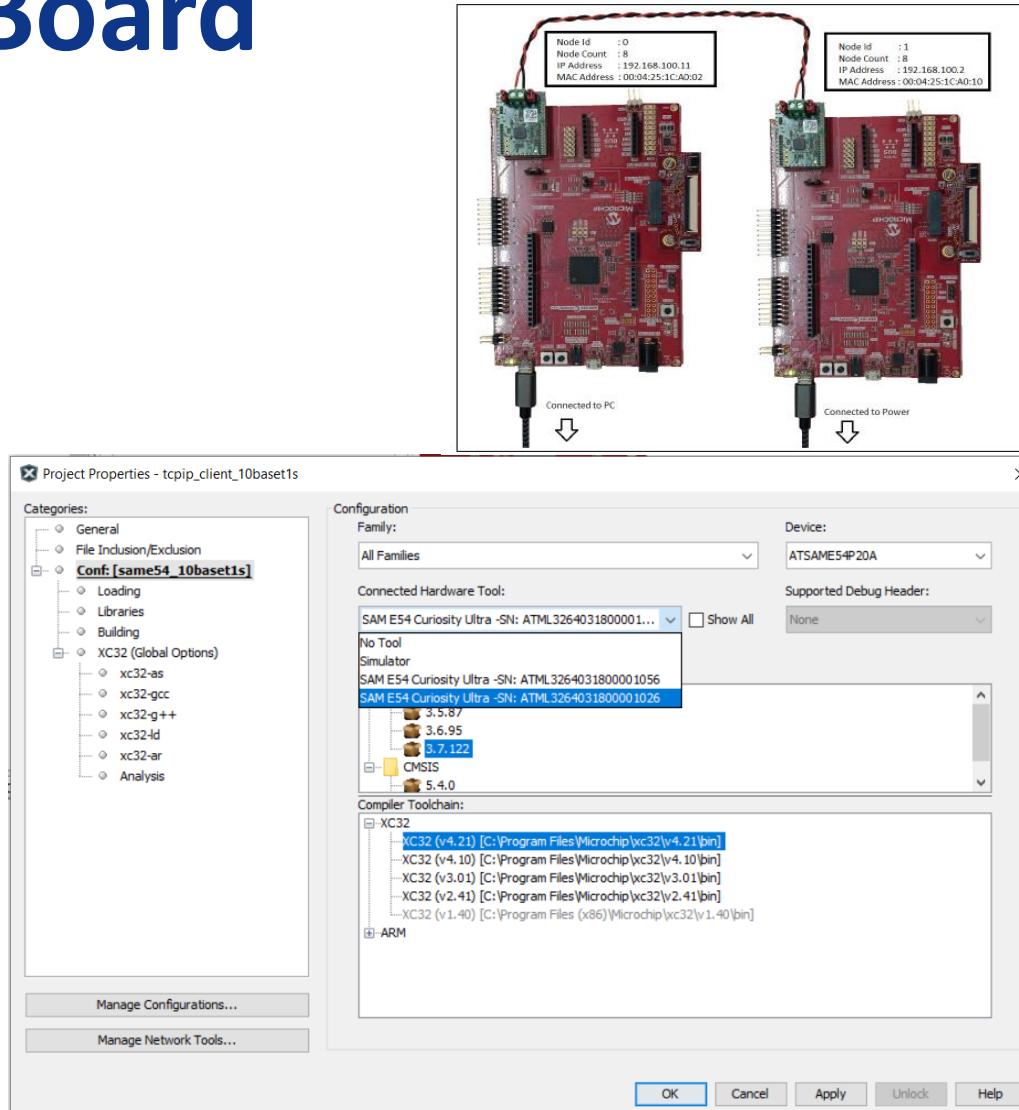
BUILD SUCCESSFUL (total time: 2m 19s)
Loading code from C:/Users/M52689/HarmonyProjects/tcpip_client_10base1s/firmware/same54_curiosity.X/dist/same54_10base1s/production/same54_curiosity.X.production.hex...
Program loaded with pack, SAME54_DFP, 3.7.122, Microchip
Loading completed
```

The "Output" window displays a log of the build process. It starts with the command "tcpip_client_10base1s (Clean, Build, ...)". It then shows the loading of the linker script "ATSAME54P20A.ld" from the "config" directory. The "make" command is run, with "make[2]" and "make[1]" indicating directory levels. The build is successful, taking 2 minutes and 19 seconds. The log then shows the loading of the generated hex file "same54_curiosity.X.production.hex" from the "dist" directory. Finally, it indicates that the program has been loaded with the "SAME54_DFP" pack version 3.7.122 and that the loading is completed.

Program the Node 0 Board

- Connect the boards to the PC

- In the Project Properties, check the Connected Hardware Tool list dropdown menu and select which board to program
- This board will be the PLCA Node 0 board



Program the Node 0 Board

- Use Make and Program Device to program the board.
- Check the output to make sure there were no errors



```
Output x Notifications [MCC] News
Kits x EDBG x MPLAB® Code Configurator x Scripting x EDBG-same54_curiosity x Configuration Loading Message x Configuration Loading Error x tcpip_client_10base1s (Build, Load, ...)

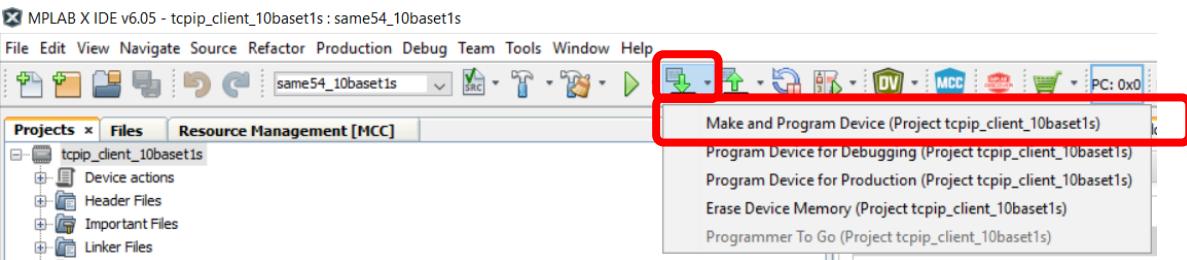
Currently loaded versions:
Application version.....3.37.438 (0x03.0x25.0x01b6)
Tool pack version .....1.4.384
Target voltage detected
Target device ATSAME54P20A found.
Device Revision Id = 0x3
Device Id = 0x61040000

Calculating memory ranges for operation...

Erasing...

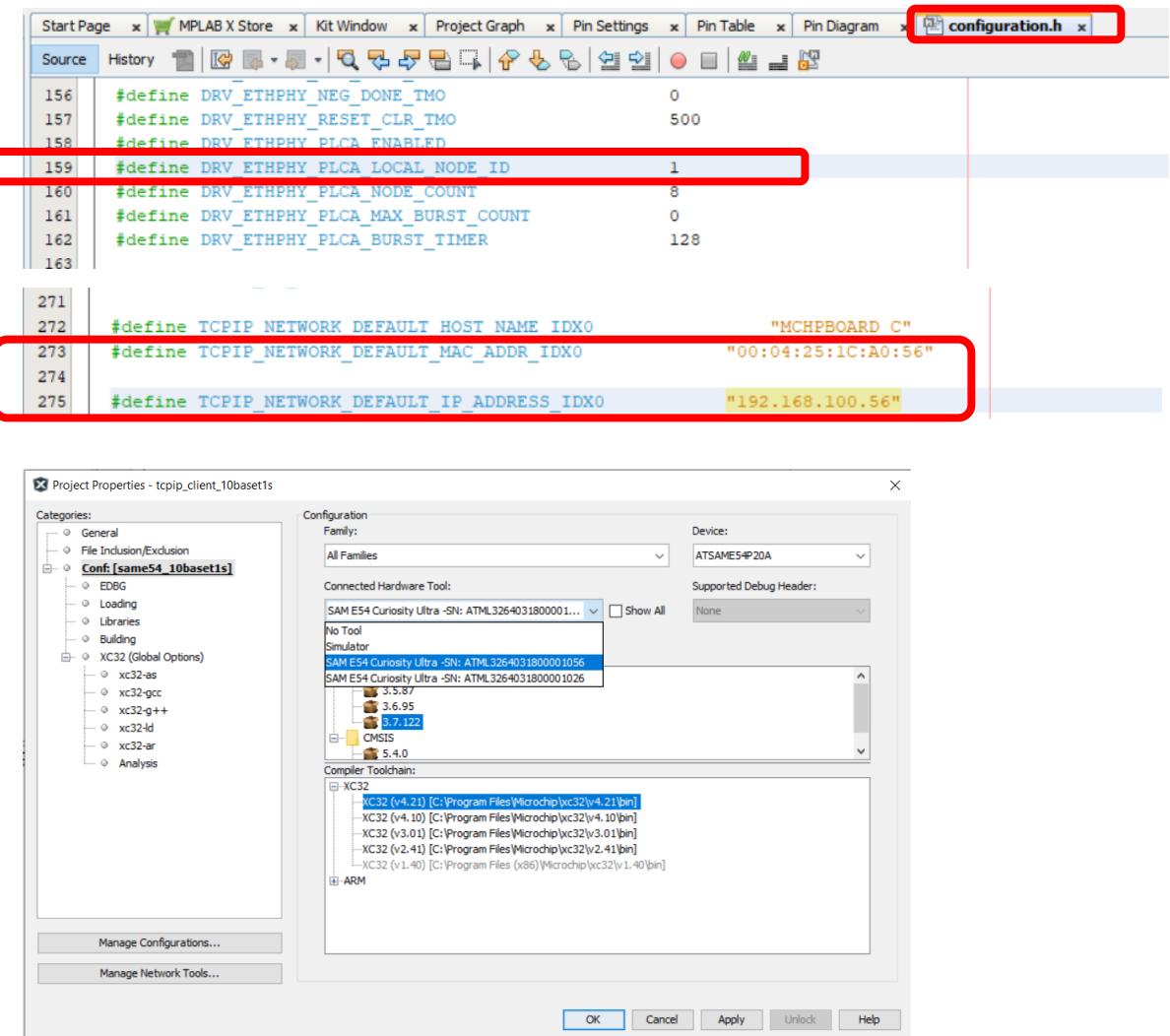
The following memory area(s) will be programmed:
program memory: start address = 0x0, end address = 0xfffff
configuration memory

Due to the large memory ranges on this device, only the areas of memory that have been loaded with code (via the build process or loading a hex file) will be read by default. If you wish to read
Programming complete
```



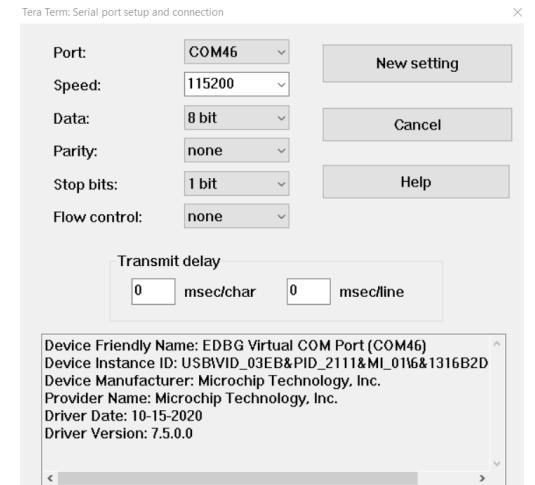
Program the Node 1 Board

- In configuration.h, change the Node ID to 1 and change the MAC and IP Address so that they are distinct from the Node 0 Board
- In Project Properties, change the Connected Hardware Tool to the other board
- Repeat the “Make and Build Main Project” and “Make and Program Device” steps
- The boards are now programmed



Test

- Open a serial connection to both boards using the settings shown
- Reset both boards using the Reset button on the board
 - You should see the TCP/IP Stack Initialization messages to show the stack is working
- Use the “netinfo” command to show the Network Configuration
 - Tip – wait a few moments after resetting the boards to allow the link to establish



COM47 - Tera Term VT

File Edit Setup Control Window Help

```
TCP/IP Stack: Initialization Started
TCP/IP Stack: Initialization Ended - success
>netinfo
----- Interface <eth0/GMAC> -----
Host Name: MCHPBOARD_C - NBNS disabled
IPv4 Address: 192.168.100.26
Mask: 255.255.255.0
Gateway: 192.168.100.1
DNS1: 192.168.100.1
DNS2: 0.0.0.0
MAC Address: 00:04:25:1c:a0:26
default IP address is ON
dhcp is enabled
Link is UP
Status: Ready
>
```

COM46 - Tera Term VT

File Edit Setup Control Window Help

```
TCP/IP Stack: Initialization Started
TCP/IP Stack: Initialization Ended - success
>netinfo
----- Interface <eth0/GMAC> -----
Host Name: MCHPBOARD_C - NBNS disabled
IPv4 Address: 192.168.100.56
Mask: 255.255.255.0
Gateway: 192.168.100.1
DNS1: 192.168.100.1
DNS2: 0.0.0.0
MAC Address: 00:04:25:1c:a0:56
default IP address is ON
dhcp is enabled
Link is UP
Status: Ready
>
```

ROCHIP

Ping test

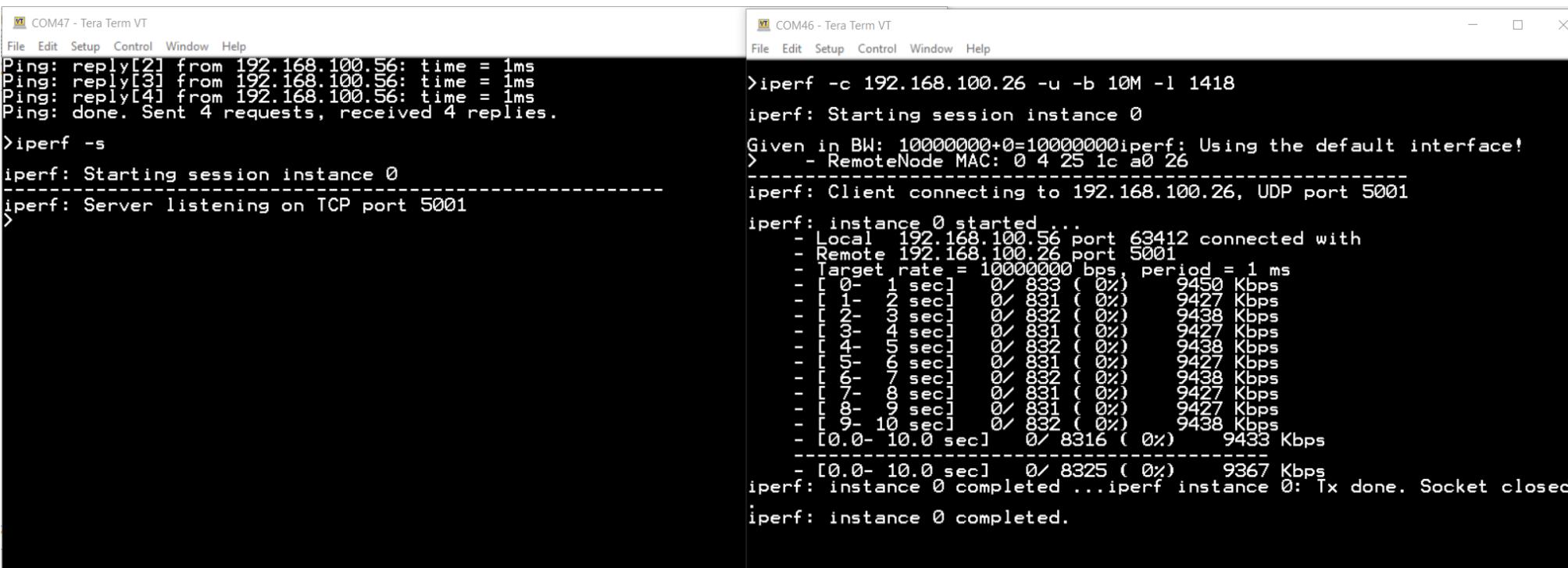
- Use the “ping” command to check the link

```
COM47 - Tera Term VT
File Edit Setup Control Window Help
TCP/IP Stack: Initialization Started
TCP/IP Stack: Initialization Ended - success
>
>netinfo
----- Interface <eth0/GMAC> -----
Host Name: MCHPBOARD_C - NBNS disabled
IPv4 Address: 192.168.100.26
Mask: 255.255.255.0
Gateway: 192.168.100.1
DNS1: 192.168.100.1
DNS2: 0.0.0.0
MAC Address: 00:04:25:1c:a0:26
default IP address is ON
dhcp is enabled
Link is UP
Status: Ready
>ping 192.168.100.56
>Ping: reply[1] from 192.168.100.56: time = 1ms
Ping: reply[2] from 192.168.100.56: time = 1ms
Ping: reply[3] from 192.168.100.56: time = 1ms
Ping: reply[4] from 192.168.100.56: time = 1ms
Ping: done. Sent 4 requests, received 4 replies.
>
```

```
COM46 - Tera Term VT
File Edit Setup Control Window Help
TCP/IP Stack: Initialization Started
TCP/IP Stack: Initialization Ended - success
>
>netinfo
----- Interface <eth0/GMAC> -----
Host Name: MCHPBOARD_C - NBNS disabled
IPv4 Address: 192.168.100.56
Mask: 255.255.255.0
Gateway: 192.168.100.1
DNS1: 192.168.100.1
DNS2: 0.0.0.0
MAC Address: 00:04:25:1c:a0:56
default IP address is ON
dhcp is enabled
Link is UP
Status: Ready
>ping 192.168.100.26
>Ping: reply[1] from 192.168.100.26: time = 1ms
Ping: reply[2] from 192.168.100.26: time = 1ms
Ping: reply[3] from 192.168.100.26: time = 1ms
Ping: reply[4] from 192.168.100.26: time = 1ms
Ping: done. Sent 4 requests, received 4 replies.
>
```

iperf

- On one terminal type “iperf -s” to initiate the iperf server on that node
- On the other terminal type “iperf -c 192.168.100.xx -u -b 10M -l 1418” to test the speed of the link
- Try other ethernet frame length options other than 1418, such as 700 and 300 to show the reduction of the bandwidth when the length of the frame is limited in this way



The image shows two terminal windows side-by-side. The left window, titled 'COM47 - Tera Term VT', displays the output of an iperf server. It shows four ping replies from 192.168.100.56 with a time of 1ms each, followed by the command '>iperf -s' and its response: 'iperf: Starting session instance 0'. It also shows 'iperf: Server listening on TCP port 5001' and a final '>'. The right window, titled 'COM46 - Tera Term VT', displays the output of an iperf client. It starts with the command '>iperf -c 192.168.100.26 -u -b 10M -l 1418', followed by 'iperf: Starting session instance 0'. It then shows 'Given in BW: 10000000+0=10000000 iperf: Using the default interface!' and 'iperf: - RemoteNode MAC: 0 4 25 1c a0 26'. A dashed line follows, and then it shows 'iperf: Client connecting to 192.168.100.26, UDP port 5001'. Below this, it shows the performance data for 10 seconds, starting with '- Local 192.168.100.56 port 63412 connected with'. The data includes target rate, period, and throughput for each second. The last few lines of the client output are: '- [0.0- 10.0 sec] 0/ 8316 (0%) 9433 Kbps', '- [0.0- 10.0 sec] 0/ 8325 (0%) 9367 Kbps', 'iperf: instance 0 completed ... iperf instance 0: Tx done. Socket closed', and 'iperf: instance 0 completed.'

Useful Links and References

For more information...

- **For installing the necessary software, see:**
 - [MPLAB® X IDE Essentials - 01: Installation and Ecosystem – YouTube](#)
 - Make sure to download the latest XC32 Compiler v4.21 as well as MPLAB X v6.05 and MCC Plugin v5.2.2 as shown in this video
 - We won't need the Data Visualizer for this class, you can skip that part
- **For more about creating a project:**
 - [Creating a MPLAB® Harmony Project in MCC – YouTube](#)
- **Further reading:**
 - [Learn MPLAB X Integrated Development Environment \(IDE\) | Microchip Technology](#)

Demo Project

- There is a demo project checked into the Github Harmony repository that you will have downloaded already in the net_10base_t1s library as part of the Optional Packages list during the project setup
- Use File > Open Project and navigate to this library, and choose from the freertos or non-freertos versions
 - Note: in order to generate and compile these projects, you will still need to have downloaded the necessary packages as described in the project setup

