

Exercise 2: Eclipse / ARM GCC Toolchain MCUXpresso SDK and Config Tools for LPCXpresso824MAX board under Windows

Revision History

Revision Number	Date	Author	Description
0.1	04Oct2014	JW	First draft.
0.2	26Nov2014	JW	Generalized Eclipse installation. Added SEGGER J-Link GDB support.
0.3	04Feb2015	JW	Updated GNU tools section. Updated Eclipse section. Fixed table of contents links.
0.4	10Aug2016	JW	Updated SEGGER J-Link support to version 6.00e
0.5	13Nov2022	JW	Updated to latest Eclipse (Embedded) Updated link to ARM GNU Tool chain (GCC compiler/linker) Added NXP example project (LPC824) Added SDK for MCUXpresso Config tool

Audience

This document is intended for the ARM Cortex-M beginner and seasoned developers interested in evaluating ARM Cortex-m platforms.

Acknowledgements

The authors have been supported in this work by NXP Semiconductors. Select images in this text were reprinted with the permission of NXP Semiconductors.

Table of Contents

Revision History 2

Audience 2

Acknowledgements..... 2

1 NXP SDK for LPCXpresso 824MAX board 4

 1.1 Importing SDK 12

 1.2 Configuration 13

 1.2.1 Clock Tree Configuration..... 13

 1.2.2 Pin Configuration 17

 1.2.3 Peripheral Configuration..... 19

 1.2.4 Code Generation 20

2 Hardware Requirements..... 21

 2.1 LPC824Max modifications..... 22

3 Importing project into Eclipse..... 24

 3.1 Cross Assembler FLAGS..... 27

 3.2 C Cross Compiler FLAGS 28

 3.3 Cross Linker FLAGS..... 29

 3.4 Include Paths..... 30



1 NXP SDK for LPCXpresso 824MAX board

This exercise picks up on the Eclipse/ARM free tool chain set up document. The focus is on how to generate a GCC NXP Xpresso SDK for the LPCXpresso 824MAX board and import it into the Eclipse IDE for Embedded C/C++ Developers. Using this code generation tool aids in navigating the MCU platforms user manual, provides access to the complete register set and more important provides a graphical representation of the MCU clock tree!

NOTE: NXP SDK generator will deliver the GCC sources as a CMake based project. For this exercise we will not use CMAKE scripts and batch files. We will import the sources as an Eclipse *C Managed Build* project. We will explore CMake managed projects on our next document Exercise 3 *How to import an Empty or Existing CMake Project into the Eclipse IDE*.

1. Go to NXP.com, if you do not have a login account create one
2. Login to your account, from the NPX home page navigate to:
PRODUCTS → ARM MCUs → General Purpose MCUs → LPC800 Arm Cortex-M0+
 This will take you to the LPC800 Series landing page:

The screenshot shows the NXP website's product selector for the LPC800 series. The sidebar on the left lists various microcontroller families, with 'LPC800 Arm Cortex-M0+' circled in green. The main content area features a 'PRODUCT SELECTOR' button and a table titled 'LPC800 Series Product' with columns for Family, Core, Memory, Differentiated Features, and Package Options. The 'LPC82x MCU Family' row in the table is also circled in green.

Family	Core	Memory	Differentiated Features	Package Options
LPC8N04 MCU	8 MHz Cortex-M0+ core	32 KB Flash 8 KB SRAM 4 KB EEPROM	Up to 12 GPIO NFC/RFID ISO 14443 type A interface Temperature sensor with ±1.5 °C accuracy -40 °C to +85 °C	HVQFN24
LPC80x MCU Family	15 MHz Cortex-M0+ core	Up to 32 KB EEPROM Flash Up to 4 KB SRAM	Up to 30 GPIO 12-bit ADC, 10-bit DAC, Comparator -40 °C to +105 °C	TSSOP20 TSSOP29 TSSOP24 HVQFN32
LPC81x MCU Family	30 MHz Cortex-M0+ core	Up to 16 KB Flash Up to 4 KB SRAM	Up to 18 GPIO SCTimer/PWM Comparator -40 °C to 105 °C	TSSOP16 TSSOP29 SOP20 XSOP16
LPC82x MCU Family	30 MHz Cortex-M0+ core	Up to 32 KB Flash Up to 8 KB SRAM	Up to 29 GPIO SCTimer/PWM 12-bit ADC, Comparator -40 °C to +105 °C	TSSOP20 HVQFN33
LPC83x MCU Family	30 MHz Cortex-M0+ core	Up to 32 KB Flash Up to 4 KB SRAM	Up to 29 GPIO SCTimer/PWM 12-bit ADC -40 °C to +85 °C	TSSOP20 HVQFN33
LPC84x MCU Family	30 MHz Cortex-M0+ core	Up to 64 KB Flash Up to 16 KB SRAM	Up to 54 GPIO SCTimer/PWM Fast Initialization Memory (FIM) 12-bit ADC, Dual 10-bit DAC, Comparator 9 Button Matrix Capacitive Touch -40 °C to +105 °C	HVQFN33 HVQFN48 LQFP48 LQFP64
LPC86x MCU Family	60 MHz Cortex-M0+ core	Up to 64 KB Flash Up to 8 KB SRAM	Up to 54 GPIO I3C Flex Timer 12-bit ADC, Comparator -40 °C to +105 °C	HVQFN33 HVQFN48 LQFP64

Figure 1 © 2022 NXP B.V.

3. From the landing page select LPC82x MCU Family, scroll down to Software and select the *MCUXpresso Software Development Kit (SDK)* **DOWNLOAD OPTIONS** button:

The screenshot displays the 'Software' section of the NXP website. On the left, there are filters for 'NXP (8)' and 'Partners (23)'. Below these are sections for 'Embedded Software' and 'Development Software'. The main content area shows a search bar and a list of software files. The first item is 'MCUXpresso Software Development Kit (SDK)' under the category 'BSP, DRIVERS AND MIDDLEWARE', which is marked as 'FEATURED'. A green circle highlights the 'DOWNLOAD OPTIONS' button for this item. Other items include 'MCUXpresso Config Tools - Pins, Clocks, Peripherals' and 'MCUXpresso Integrated Development Environment (IDE)'. A banner indicates 'Additional software available. View our featured partner solutions.' Below this, there are more items like 'LPC824 Example Code Bundle Keil' and 'LPCOpen Software Development Platform LPC8XX'.

Figure 2 © 2022 NXP B.V.

4. From the MCUXpresso Software Development Kit (SDK) landing page select the MCUXpresso SDK – SDK Builder **DOWNLOAD** button. Once at the SDK Builder page click the *Select Development Board* button:

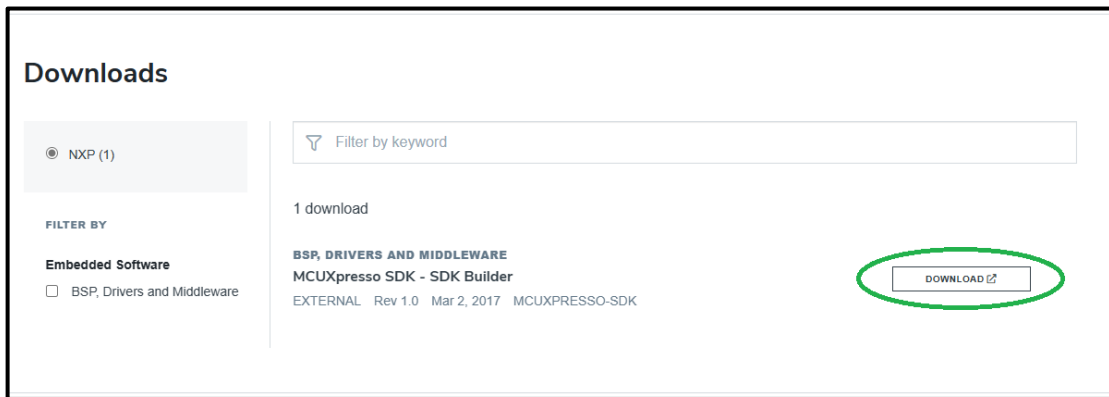


Figure 3 © 2022 NXP B.V.

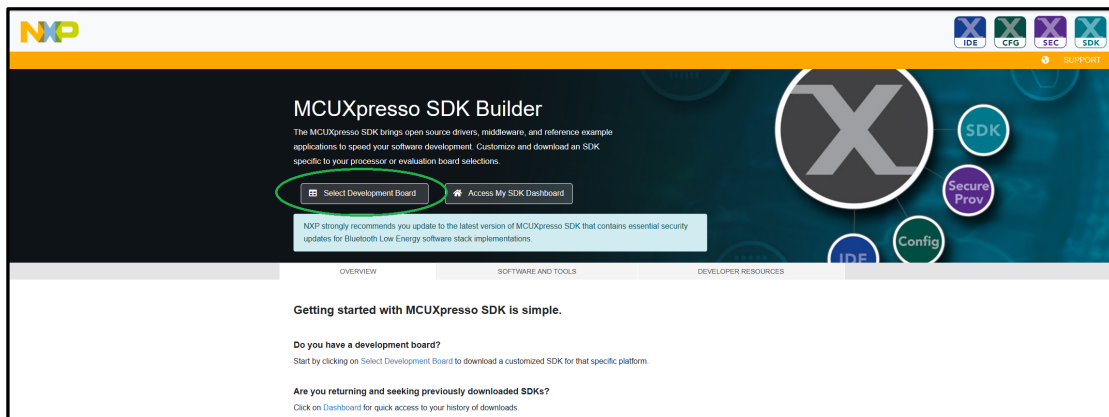


Figure 4 © 2022 NXP B.V.

5. In the *Search for Hardware* field enter **LPCXpresso824MAX** select the board then select the **Build MCUXpresso SDK** button [vX.XX.X] (Located on the right side of the landing page):

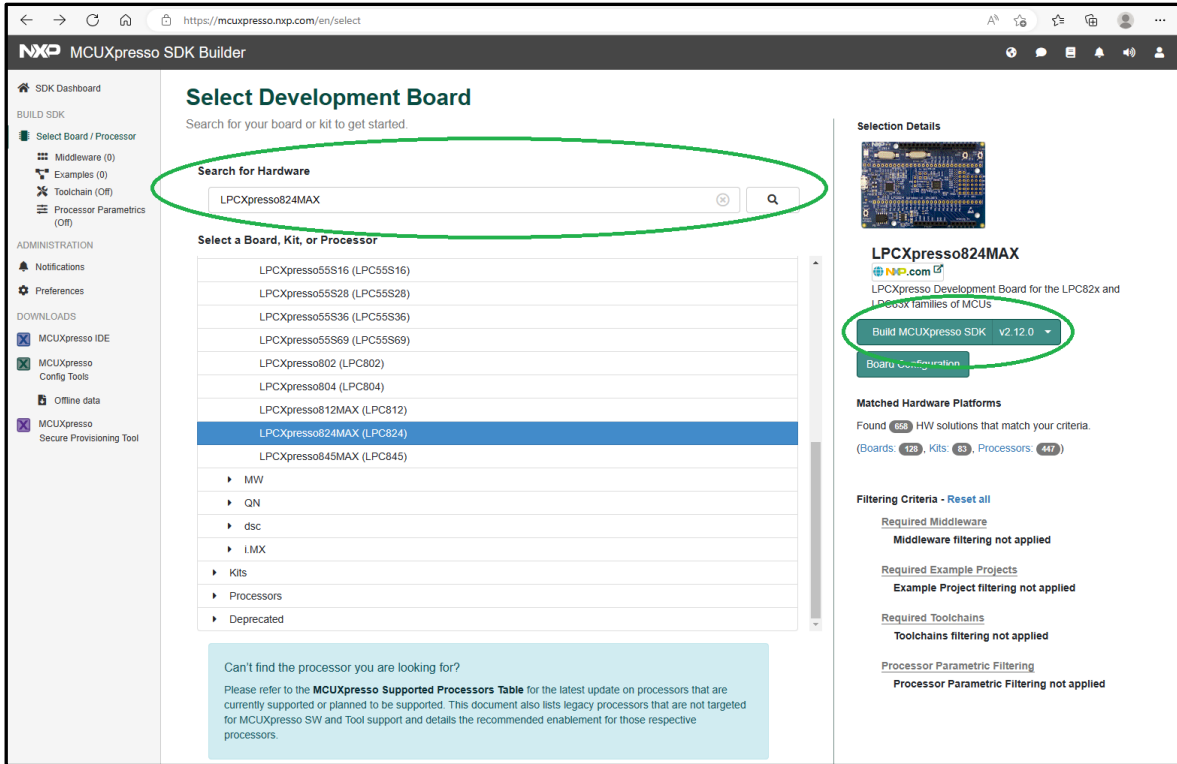


Figure 5 © 2022 NXP B.V.

- With this exercise the GNU GCC tools are used, ensure the Toolchain/IDE the GCC ICON is highlighted. Select the **DOWNLOAD SDK** button, this will generate the SDK and once complete will take the developer their SDK Dashboard page, select the download ICON for the LPCXpresso824MAX (GCC ARM Embedded toolchain). Select the *Download SDK* icon in the **SDK_2.xx.x_LPCXpresso824MAX** Archive:

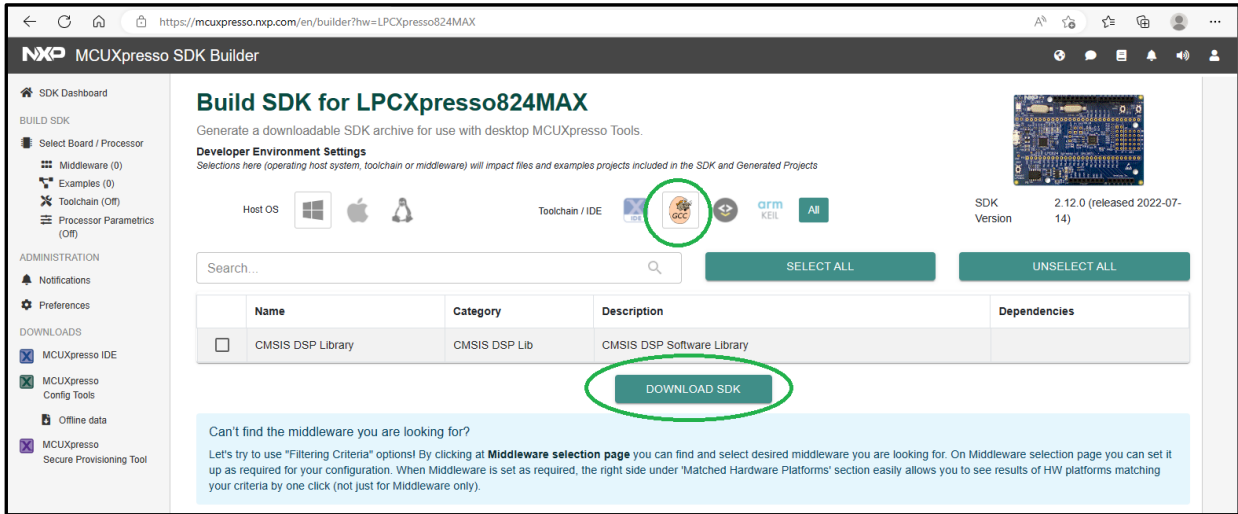


Figure 6 © 2022 NXP B.V.

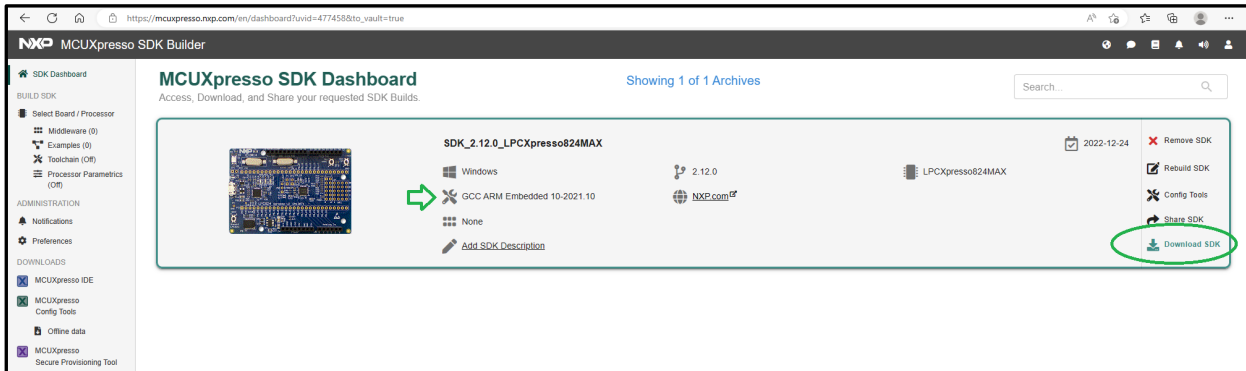


Figure 7 © 2022 NXP B.V.

7. From the **Download** dialog select *Download SDK Archive including documentation* [Agree to the *Software Terms and Conditions*]. Download the archive to a location to be accessed later:

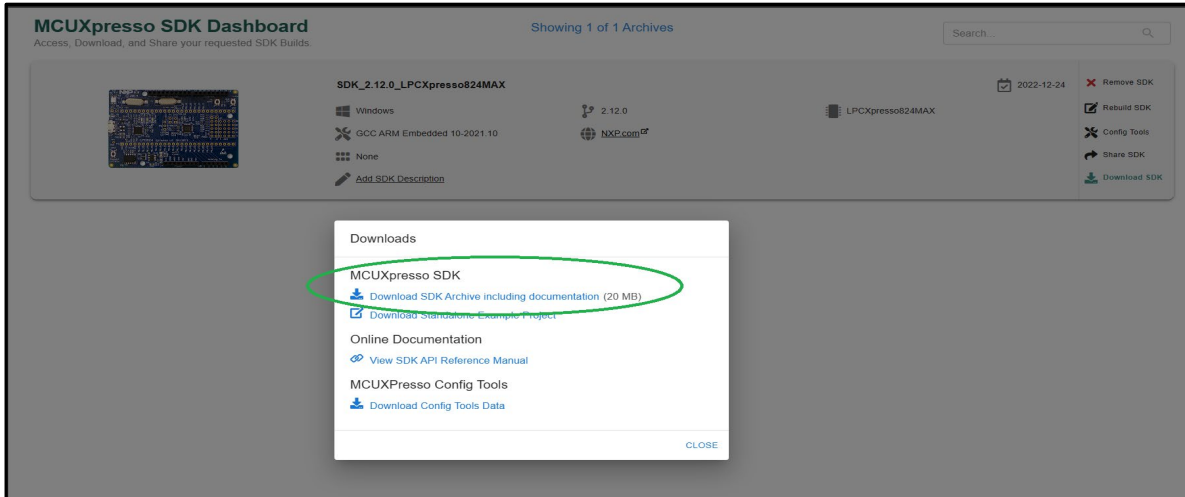



Figure 8 © 2022 NXP B.V.

8. Return to the MCUXpresso page, <https://mcuxpresso.nxp.com/en/welcome> and select the CFG icon , this will take you to the MCUXpresso Config tools landing page. The Config tool is a GUI that use the SDK to easily configure and generate C sources for the LPC824.
Note: Bookmark the MCUXpresso welcome page as this page provides quick button access to all NXP MCUXpresso tools.

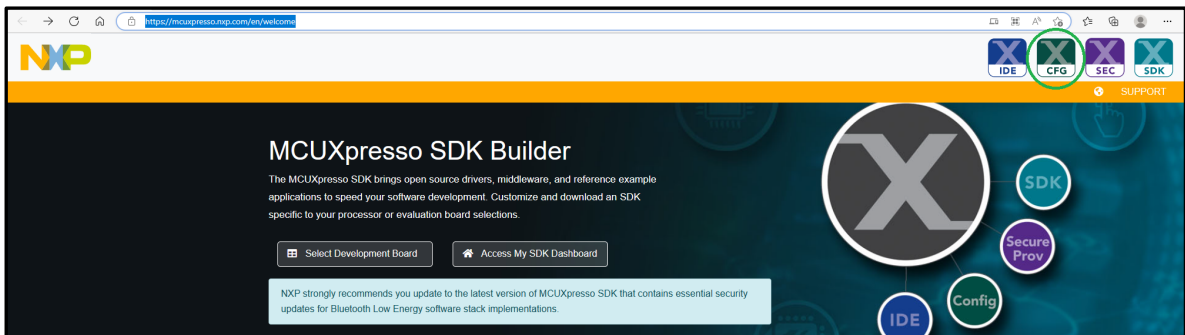


Figure 9 © 2022 NXP B.V.

- Select the **DOWNLOADS** button, once at the Downloads section select the **CODE GENERATION TOOLS** <MCUXpresso Config Tools, Windows Installer [FEATURED]> to begin the download:

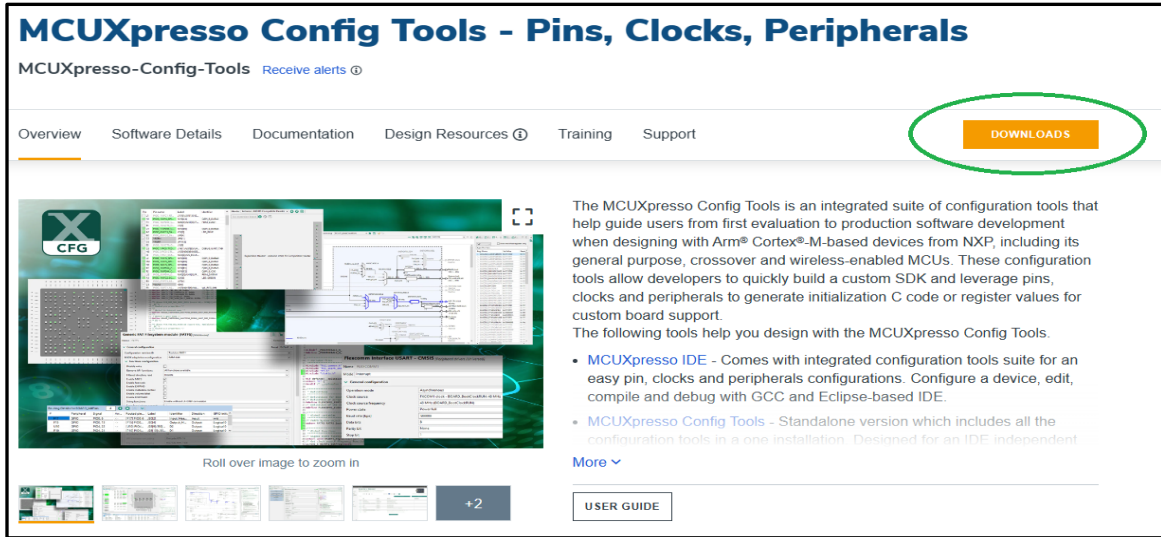


Figure 10 © 2022 NXP B.V.

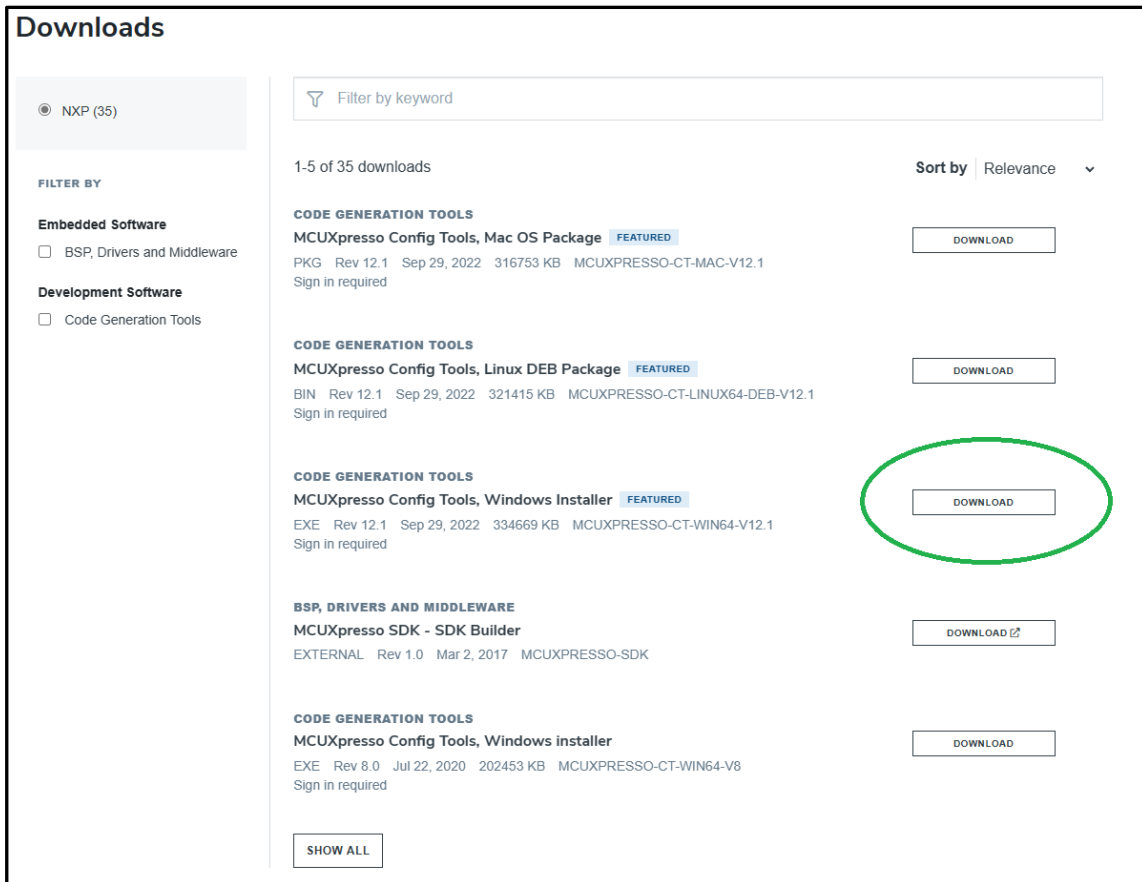


Figure 11 © 2022 NXP B.V.

10. Please download the *MCUXpresso Config Tools User's Guide* for detailed instructions on installation and use:

The screenshot shows the website for 'MCUXpresso Config Tools - Pins, Clocks, Peripherals'. The page has a navigation bar with links for Overview, Software Details, Documentation, Design Resources, Training, and Support, along with a 'DOWNLOADS' button. The main content area features a large image of the software interface with a 'CFG' logo. To the right of the image is a descriptive paragraph and a list of two tools: 'MCUXpresso IDE' and 'MCUXpresso Config Tools - Standalone version'. Below the text is a 'More' dropdown menu, and a 'USER GUIDE' button is highlighted with a green circle.

MCUXpresso Config Tools - Pins, Clocks, Peripherals

MCUXpresso-Config-Tools [Receive alerts](#)

Overview Software Details Documentation Design Resources Training Support [DOWNLOADS](#)

The MCUXpresso Config Tools is an integrated suite of configuration tools that help guide users from first evaluation to production software development when designing with Arm® Cortex®-M-based devices from NXP, including its general purpose, crossover and wireless-enabled MCUs. These configuration tools allow developers to quickly build a custom SDK and leverage pins, clocks and peripherals to generate initialization C code or register values for custom board support.

The following tools help you design with the MCUXpresso Config Tools.

- **MCUXpresso IDE** - Comes with integrated configuration tools suite for an easy pin, clocks and peripherals configurations. Configure a device, edit, compile and debug with GCC and Eclipse-based IDE.
- **MCUXpresso Config Tools - Standalone version** which includes all the configuration tools in a one installation. Designed for an IDE independent.

[More](#) ▾

[USER GUIDE](#)

Figure 12 © 2022 NXP B.V.

1.1 Importing SDK

After successful installation of the MCUXpresso Config tools, unpack the SDK for LPCXpresso824MAX board downloaded in section 2 (our path C:\NXP_DEMO) then launch the MCUXpresso Config tools. By default, the *Start development* dialog will appear. From this dialog select the *Create a new configuration and project based on an SDK example or a "hello world" project* radio button and select the Next button:



Figure 13 © 2022 NXP B.V.

The next dialog will allow us to select project paths and options. For this exercise we will work in the C:\NXP_DEMO folder, this is where the SKD archive was unpacked. In the *SDK Path* group use the Browse... button to navigate to the SDK folder. Once the path is set the *Toolchain* group should have only one option "GCC ARM Emedded XX-XXXX.XX" if not use the dropdown to select the correct toolchain. If the GCC ARM toolchain is not available go to 2 and start again... In the *Action* group select the *Clone selected example for board kit* radio button and in the *SDK Example* group choose the *led_blinky* select C:\NXP_DEMO as the working directory and final select the Finish button:

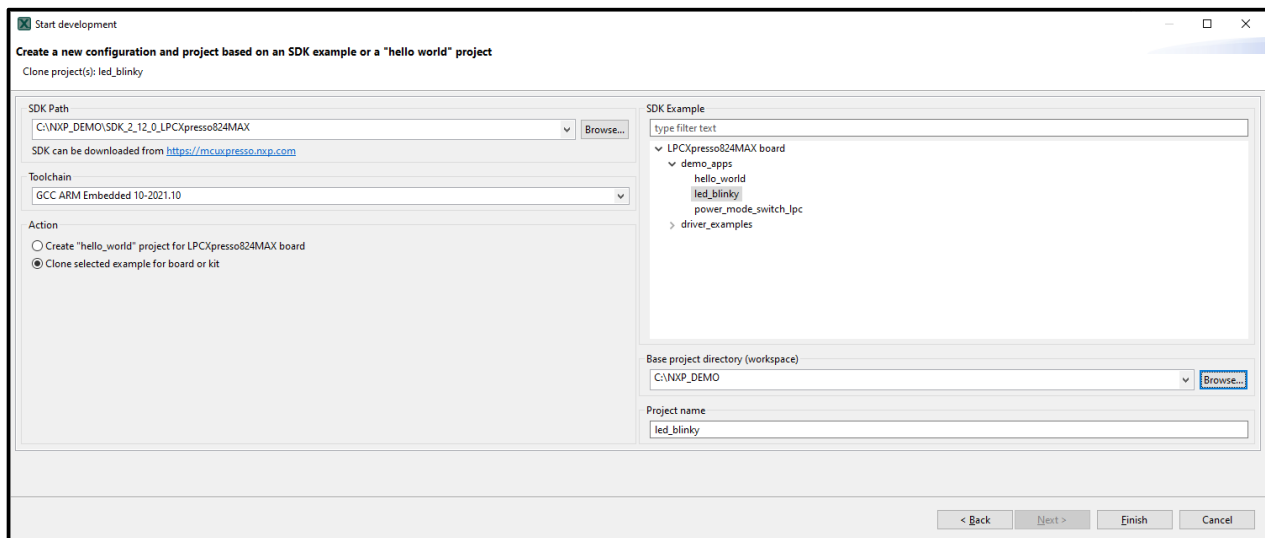


Figure 14 © 2022 NXP B.V.

1.2 Configuration

The *Config Tools Overview* dialog will appear once the base code configuration has been generated. We will use this dialog to initialize the system clocks, required peripherals and pin assignment.

1.2.1 Clock Tree Configuration

By default, the clock tool is disabled, enable the clock tool by selecting the slider button then click on the Clocks icon:

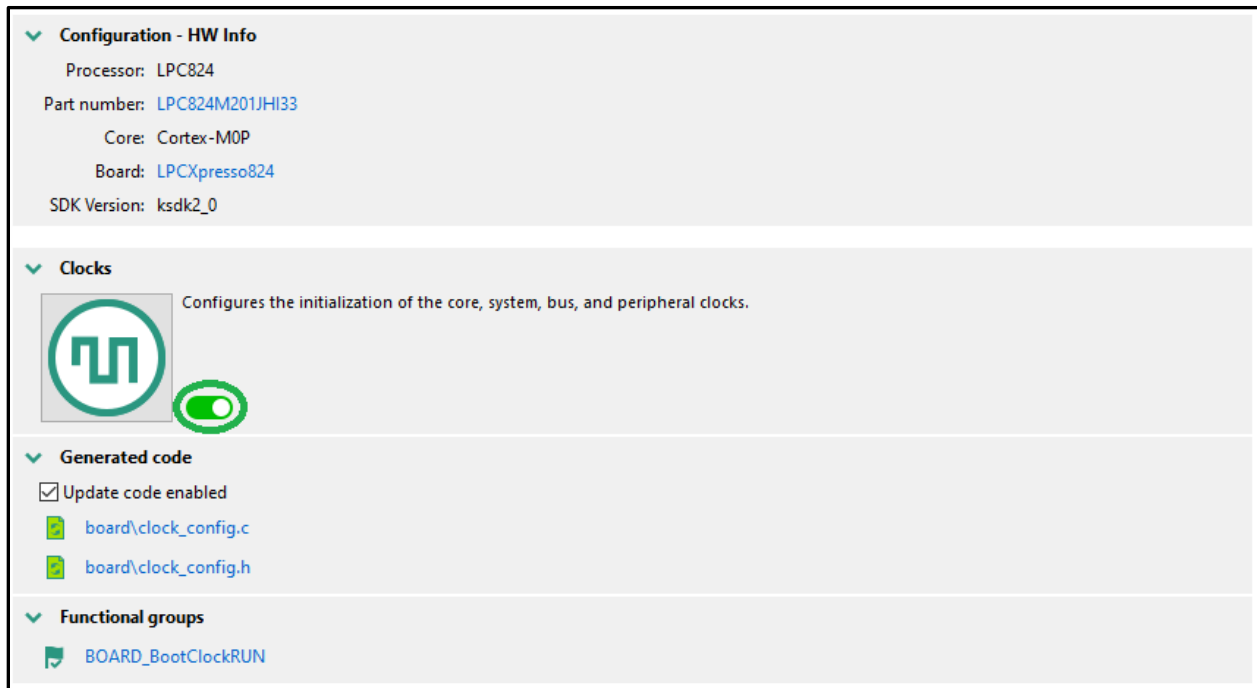


Figure 15 © 2022 NXP B.V.

This will bring up the *Clocks Diagram* tab, a GUI that can be used to initialize all system clocks. Using this GUI we will initialize the system clock to use the 12 MHz crystal oscillator enable the PLL to boost the clock to 60 MHz then divide it by two (2) resulting in a system clock of 30 MHz (maximum for the LPC824). We will then direct the **main_clk** to a CLKOUT PIN, divide by $60 \text{ MHz} / 250 = 240 \text{ kHz}$ to verify the system clock.

- 1) Connect the XTALIN and XTALOUT PINs; right click on the **sys_osc** block and select *connected* from the drop down:

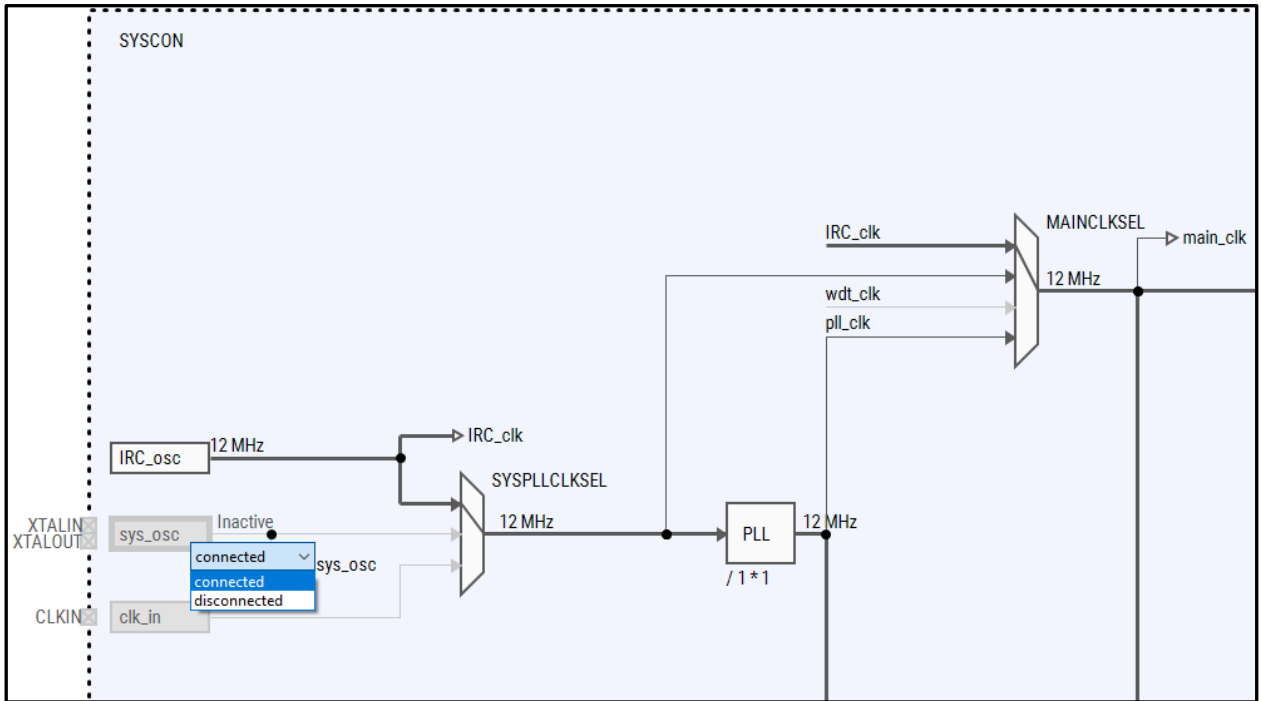


Figure 16 © 2022 NXP B.V.

- 2) Right click on the **SYSPLLCLKSEL** MUX and from the list box select *Crystal Oscillator (SYSOSC)*:

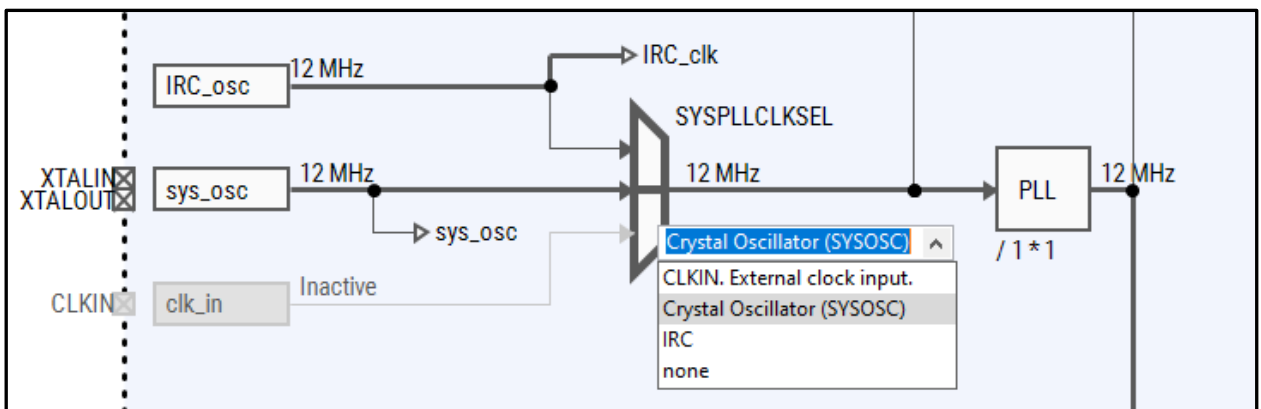


Figure 17 © 2022 NXP B.V.

- For the next setting we will use the *Details* tab (located to the right of the clock tree diagram). Double click on the **PLL** block, set **M_MULT** to * 5 (boosting the oscillator clock to 60 MHz):

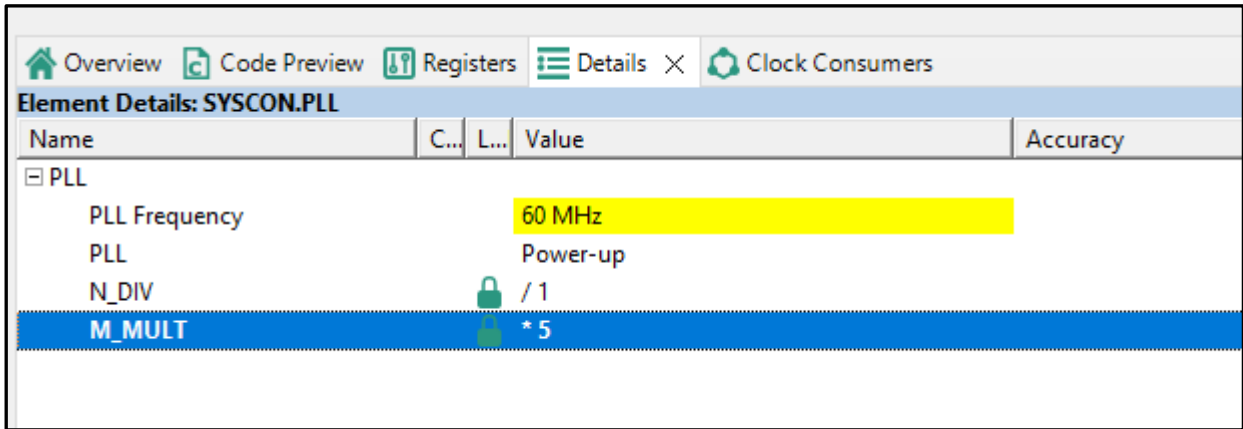


Figure 18 © 2022 NXP B.V.

- Next double click on the **MAINCLKSEL** MUX and from the list box select **pll_clk**:

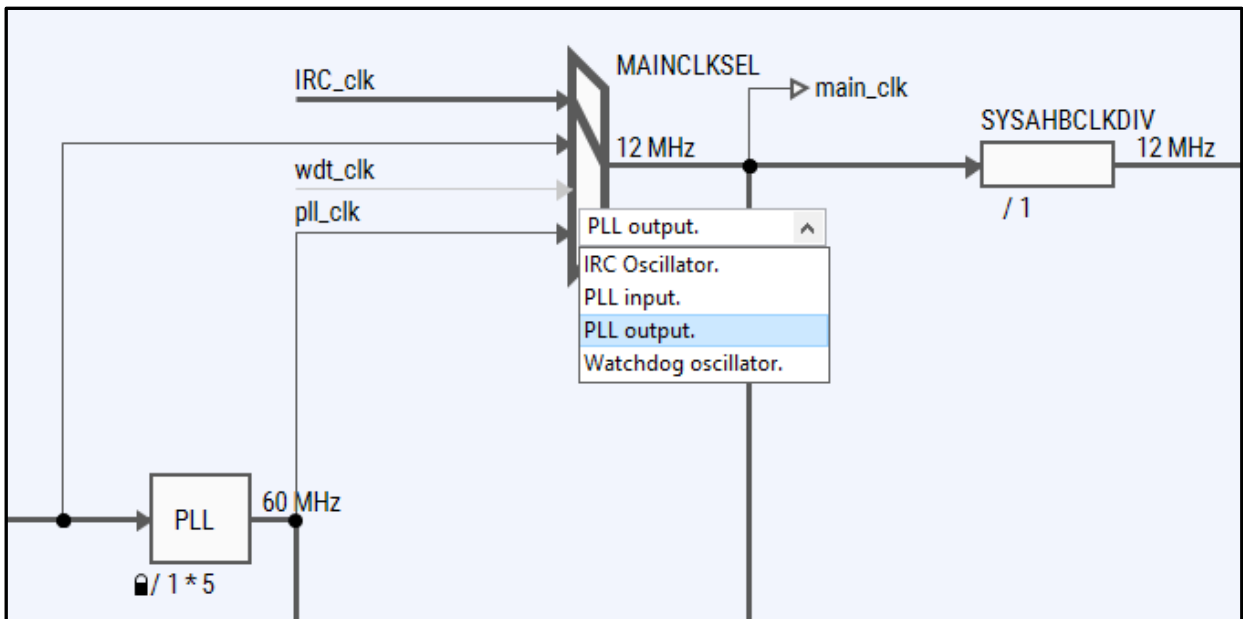


Figure 19 © 2022 NXP B.V.

- 5) Once this select has been made the tool will automatically set the **SYSAHBCLKDIV** to two (2) producing the desired 30 MHz system clock:

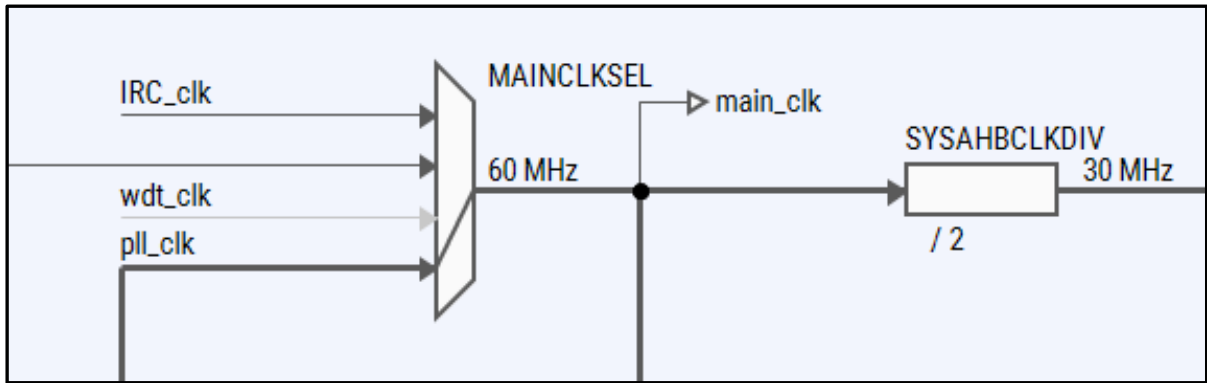


Figure 20 © 2022 NXP B.V.

- 6) From the **CLKOUTSEL** MUX select **main_clk** and set the CLKOUTDIV block and divide by 250) to produce a 240 kHz clock signal at this pin:

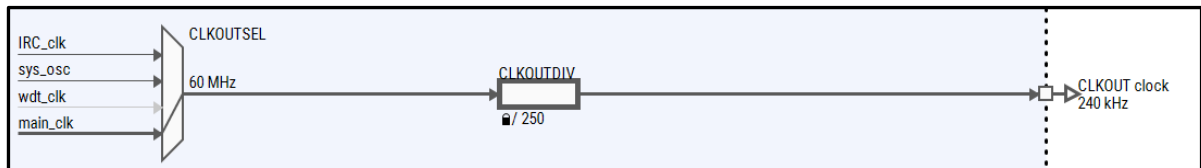


Figure 21 © 2022 NXP B.V.

- 7) The final clock to configure is the UART; in the **UARTCLKDIV** block set the divide value to 30 resulting in a 2 MHz reference for the all UARTS:

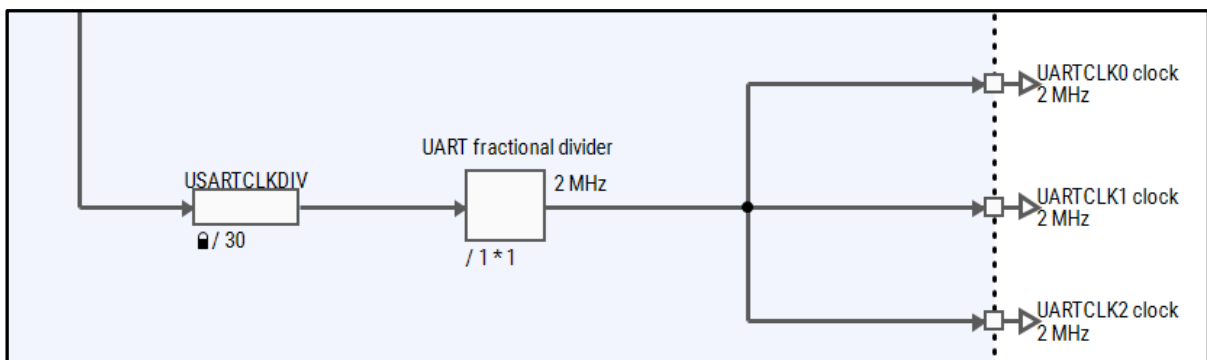


Figure 22 © 2022 NXP B.V.

1.2.2 Pin Configuration

Once the clock trees have been configured, we need to set the PIN MUX (IOCON) to route the XTAL, CLKOUT and UART pins. In the upper right corner of the Config Tools select the PIN configuration icon:

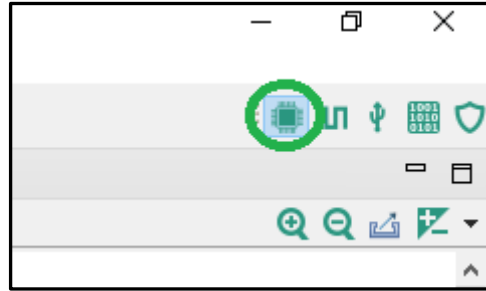


Figure 23 © 2022 NXP B.V.

- 1) By default, PIO0_12 will be configured as GPIO (for the red led “blinky”). In the identifier column of the PIN table select the RED_LED from the list box:

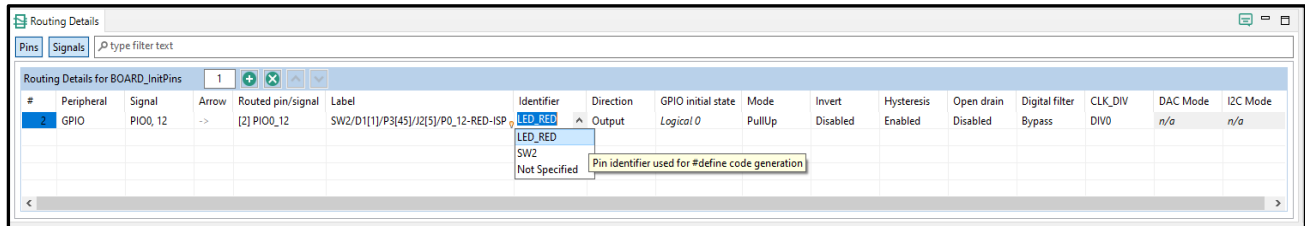


Figure 24 © 2022 NXP B.V.

2) Next, we will configure the balance of pins. From the Peripheral Signals tab select the **SWD**, **SYSCON** and **UART0** check boxes. From the SWD enable the SWCLK [PIN6/PIO0_3] and SWDIO [PIN7/PIO0_2] to dedicate these pins to the debug interface. From the SYSCON enable the XTALIN [PIN18/PIO0_8], XTALOUT [PIN17/PIO0_9] for the crystal oscillator connection. From the UART0 select TXD [PIN30/PIO0_19] and RXD [PIN29/PIO0_20] and route the CLKOUT to [PIN24/PIO0_0]:

The screenshot shows the Pin Manager software interface for the LPC824M201JHI33 - HVQFN 32 package. The peripheral selection tree on the left is expanded to show the following configurations:

- SWD**
 - SWCLK - [6] SWCLK/PIO0_3 / routable to 1 pin, 1 signal
 - SWDIO - [7] SWDIO/PIO0_2 / routable to 1 pin, 1 signal
- SYSCON**
 - CLKIN - routable to 1 pin, 1 signal
 - CLKOUT -> [24] PIO0_0/ACMP_11 / routable to 29 pins, 1 signal
 - RESETN - routable to 1 pin, 1 signal
 - XTALIN <- [18] PIO0_8/XTALIN / routable to 1 pin, 1 signal
 - XTALOUT -> [17] PIO0_9/XTALOUT / routable to 1 pin, 1 signal
- USART0**
 - CTS - routable to 29 pins, 1 signal
 - RTS - routable to 29 pins, 1 signal
 - RXD <- [29] PIO0_20/ADC_6 / routable to 29 pins, 1 signal
 - SCLK - routable to 29 pins, 1 signal
 - TXD -> [30] PIO0_19/ADC_7 / routable to 29 pins, 1 signal

The central pin grid shows the following pins highlighted in green:

- PIO0_17/...
- PIO0_18/...
- PIO0_19
- PIO0_20
- PIO0_21/...
- PIO0_22/...
- PIO0_23/...
- PIO0_14/...
- PIO0_13/...
- PIO0_12
- RESETN/...
- PIO0_4/...
- PIO0_28/...
- SWCLK
- SWDIO
- PIO0_11/...
- PIO0_0
- PIO0_6/...
- PIO0_7/...
- VREFP
- VREFN
- VDD
- XTALIN
- XTALOUT

The routing details table at the bottom provides the following information:

#	Peripheral	Signal	Arrow	Routed pin/signal	Label	Identifier	Direction	GPIO initial state	Mode	Invert	Hysteresis	Open drain	Digital filter	CLK_DIV	DAC Mode	I2C Mode
2	GPIO	PIO0_12	->	[2] PIO0_12	SW2/D1[1]/P3[45]/J2[5]/P0_12-RED-ISP	LED_RED	Output	Logical 0	PullUp	Disabled	Enabled	Disabled	Bypass	DIV0	n/a	n/a
6	SWD	SWCLK	-	[6] SWCLK	P5[4]/U2[16]/TARGET_SWCLK	DEBUG_SWD_SWCLK	Not Specified	n/a	PullUp	Disabled	Disabled	Disabled	Bypass	DIV0	n/a	n/a
7	SWD	SWDIO	-	[7] SWDIO	P5[2]/U2[17]/TARGET_SWDIO	DEBUG_SWD_SWDIO	Not Specified	n/a	PullUp	Disabled	Disabled	Disabled	Bypass	DIV0	n/a	n/a
18	SYSCON	XTALIN	<-	[18] XTALIN	P3[13]/U2[27]/P0_8-LINK_nSLEEP	n/a	Input	n/a	PullUp	Disabled	Disabled	Disabled	Bypass	DIV0	n/a	n/a
17	SYSCON	XTALOUT	->	[17] XTALOUT	P3[12]/U2[26]/P0_9-LINK_nWAKE	n/a	Output	n/a	PullUp	Disabled	Disabled	Disabled	Bypass	DIV0	n/a	n/a
24	SYSCON	CLKOUT	->	[24] PIO0_0	P3[10]/J2[8]/P0_0-ISP_U_RXD	n/a	Output	n/a	PullUp	Disabled	Disabled	Disabled	Bypass	DIV0	n/a	n/a
30	USART0	TXD	->	[30] PIO0_19	P3[22]/J2[6]/P0_19	n/a	Output	n/a	PullUp	Disabled	Disabled	Disabled	Bypass	DIV0	n/a	n/a
29	USART0	RXD	<-	[29] PIO0_20	P3[20]/J5[6]/P0_20-ADC6	n/a	Input	n/a	PullUp	Disabled	Disabled	Disabled	Bypass	DIV0	n/a	n/a

Figure 25 © 2022 NXP B.V.

1.2.3 Peripheral Configuration

By default, the SDK provides code for a console, this will be connected to peripheral UART0. Enable peripheral configuration then right click on the Peripherals icon.

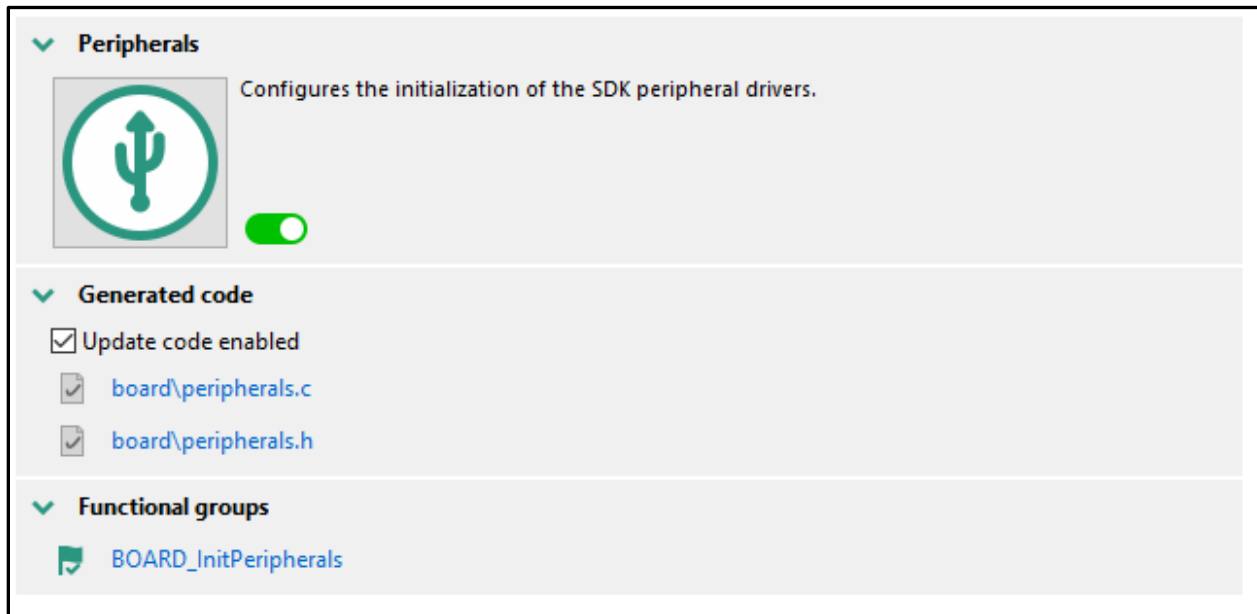


Figure 26 © 2022 NXP B.V.

Once open select UART0 and modify the Baud rate field to 115200.

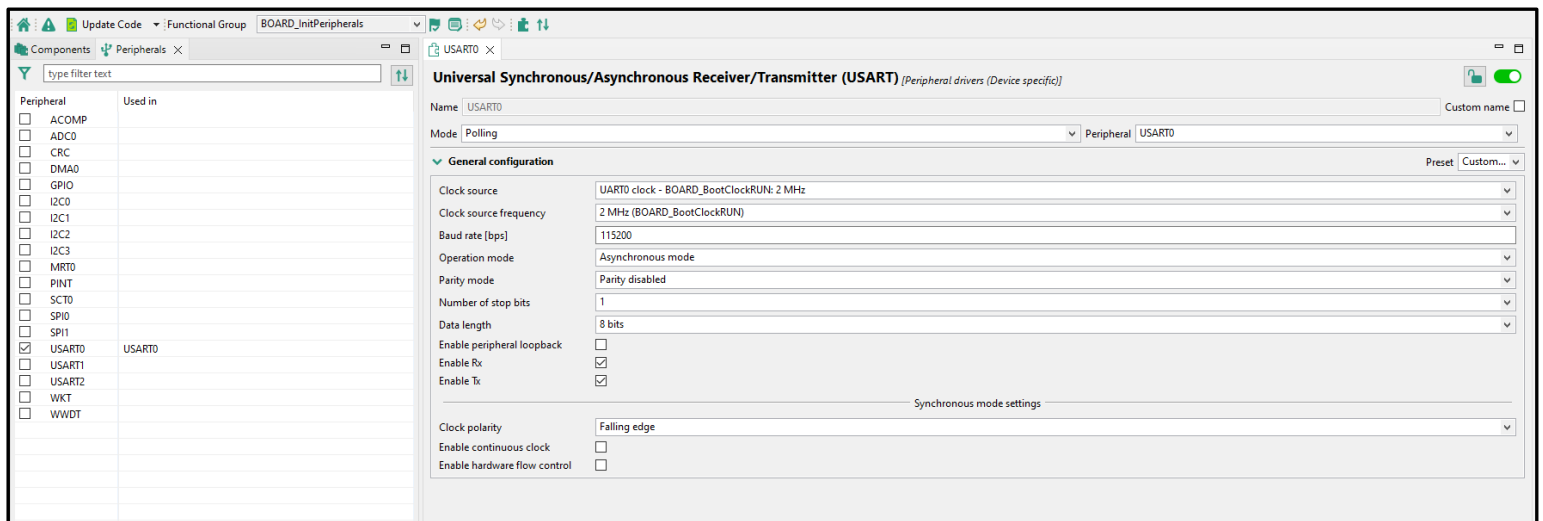
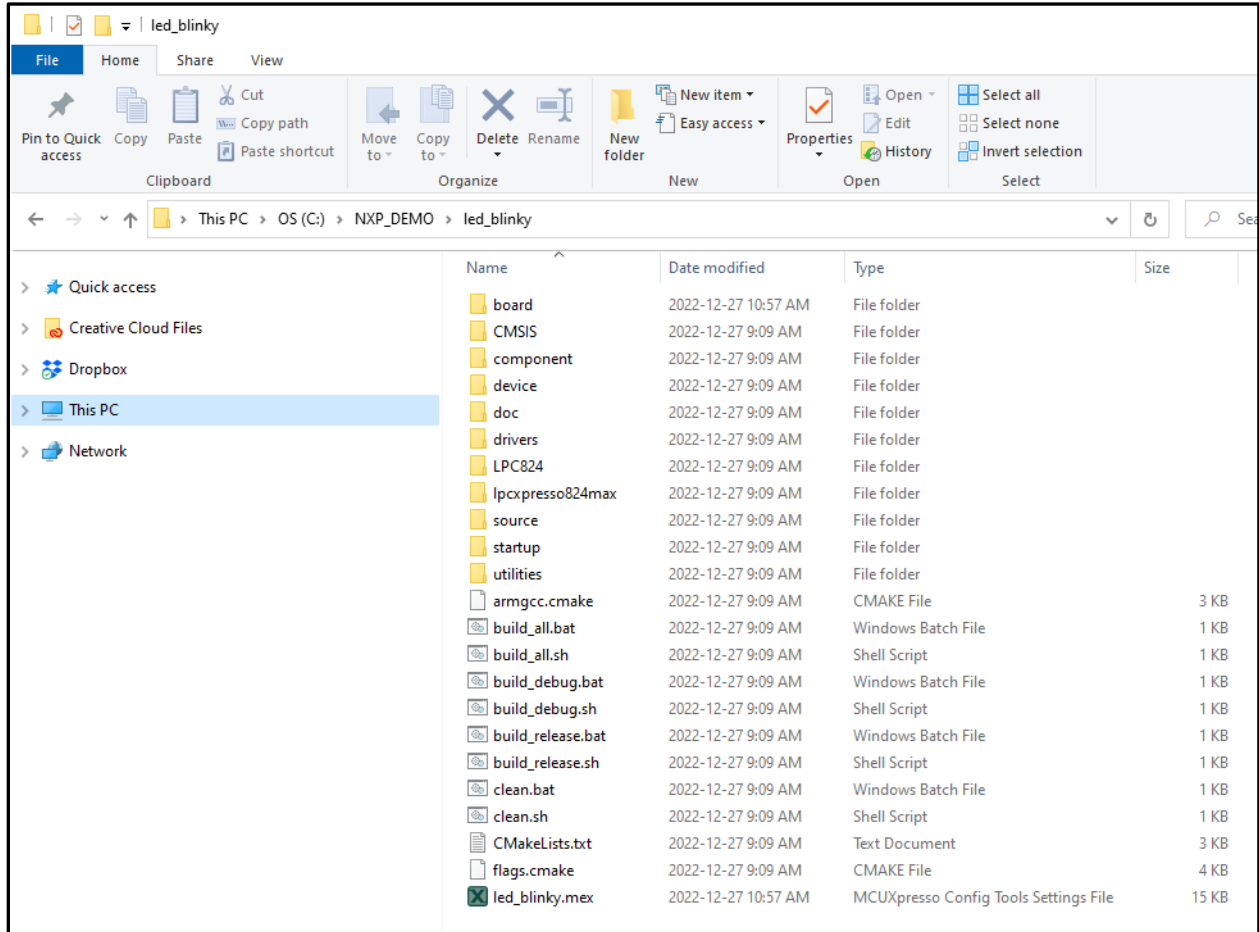


Figure 27 © 2022 NXP B.V.

1.2.4 Code Generation

Finally select the *Update Code* tab to generate the files. Open the led_blinky folder and all C sources will be generated along with CMAKE scripts and batch file. As indicated, we will not be using the CMAKE scripts and batch files for this exercise however they will be used in the next exercise. One additional file is generated, led_blinky.mex, this is the MCUXpresso Config Tools Setting File. This file can be used to launch the Config Tools and load in that last save configuration for the project.



2 Hardware Requirements

For this exercise we use NXP's *LPCXpresso board for LPC824* (single CORE Cortex-M0+) PN: OM13071, an inexpensive and readily available eval board for ARM development. In this section we will describe the hardware required and how to modify the LPCXpresso824MAX board.

- PC running Windows.
- External 5 VDC power supply
- Segger J-LINK debug probe.
- NXP LPCXpresso824MAX Evaluation board.

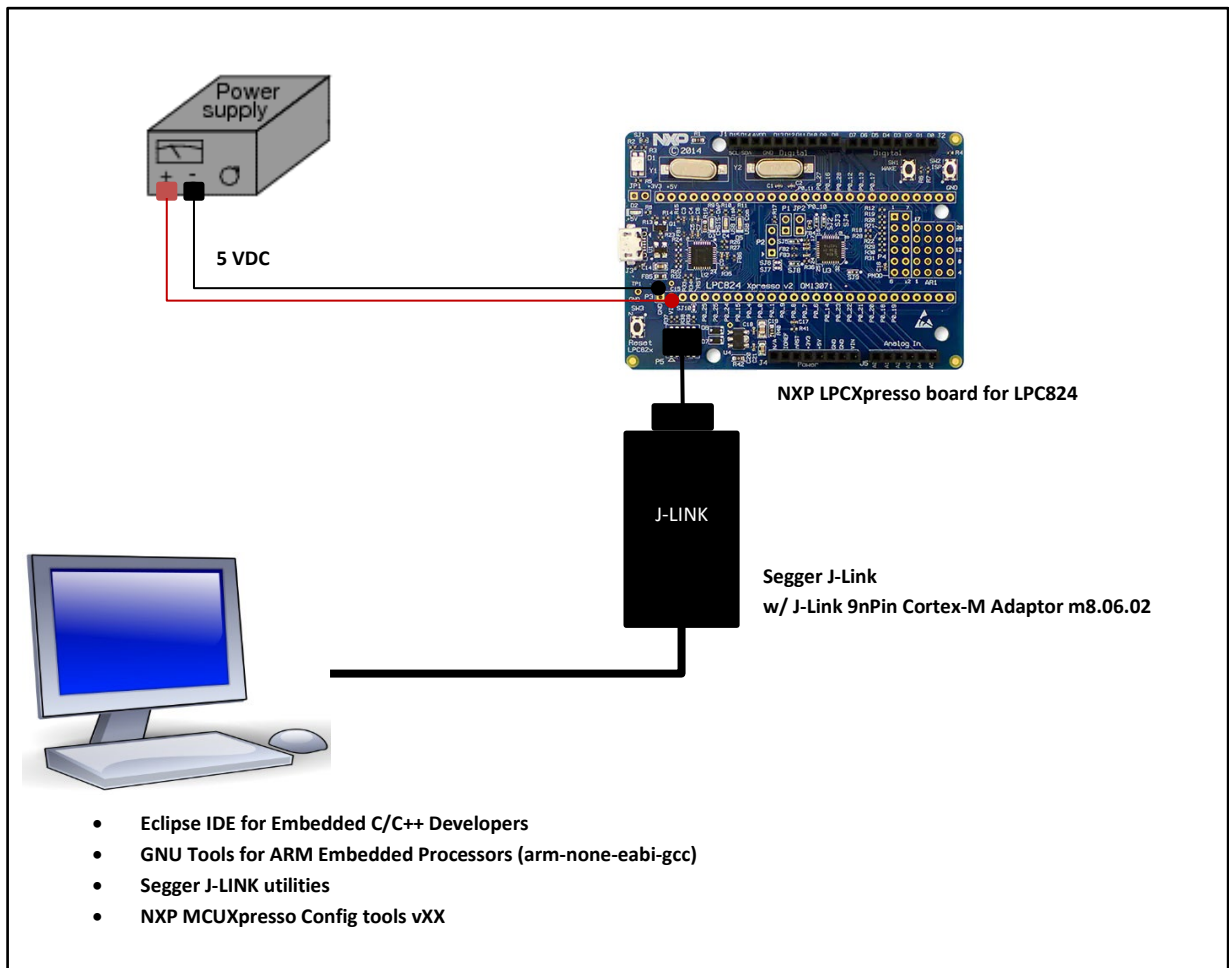


Figure 28 Exercise 2 block diagram

2.1 LPC824Max modifications

This section lists all modification to the LPCXpresso824MAX board.

- 12 MHz oscillator (Y2): This project use Y2 (12 MHz Crystal oscillator) as the reference for the LPC824 system clock. Jumpers SJ3 and SJ4 must be solder bridged between pads 1 and 2 to route crystal to PIN18 (PIO0_8/XLATIN) and PIN17 (PIO0_9/XTALOUT).

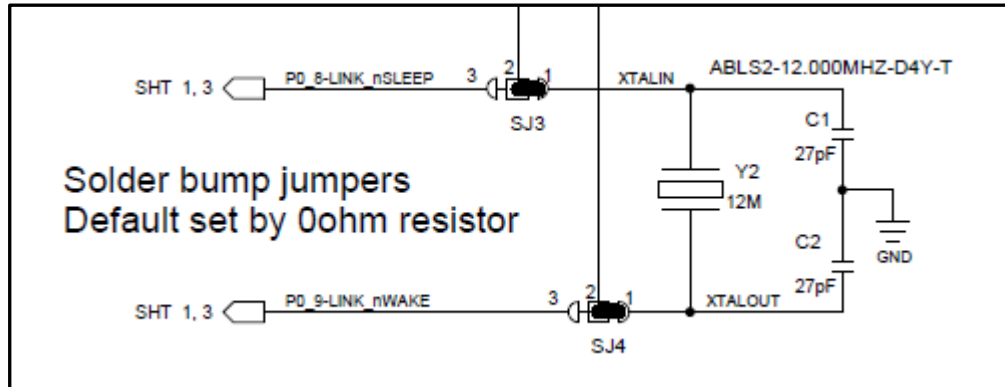


Figure 29 © 2022 NXP B.V.

- JP1 On-chip debug probe disable. This project does NOT use the LPC11U35 debug probe. If not provided, solder in a 2-PIN rectangular header (standard .100 pitch). Insert a jumper shunt to disable the on-chip debug probe.

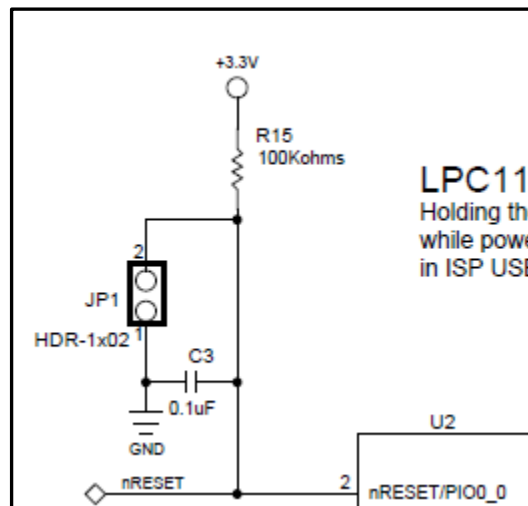


Figure 30 © 2022 NXP B.V.

- Power. There are several ways to power the LPCXpresso824MAX board externally. Our exercise uses P3 PIN1 (GND) and PIN2 (VIN). Solder in a 2-PIN rectangular header on P3.1 and P3.2 and connect an external 5.0 VDC source to P3.1 (GND) and P3.2 (5 VDC). Note: D6 reverse polarity protection is provided.

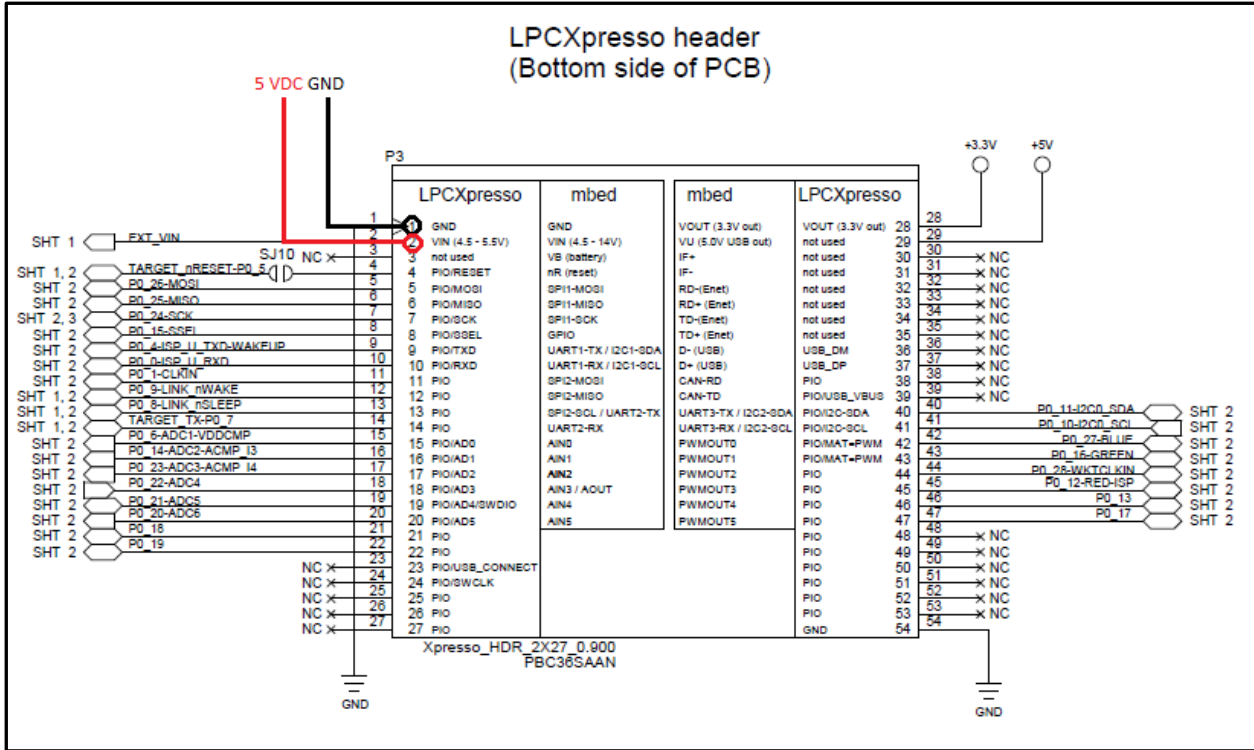
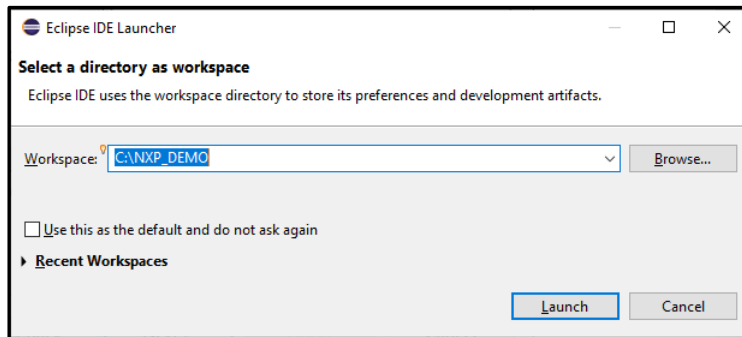


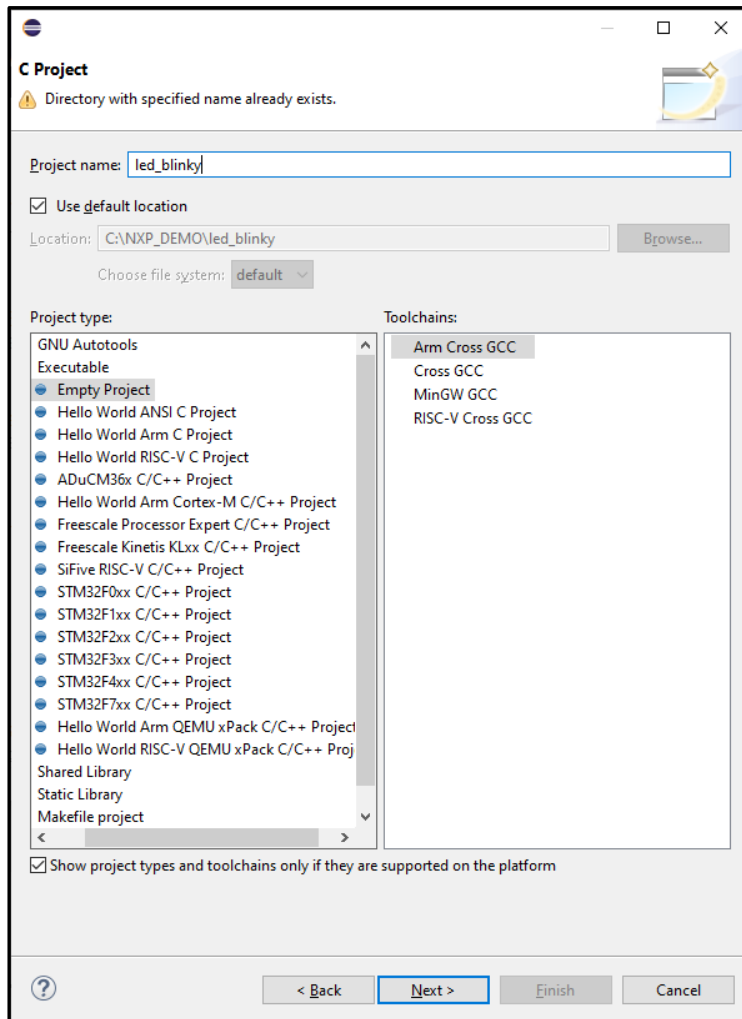
Figure 31 © 2022 NXP B.V.

3 Importing project into Eclipse

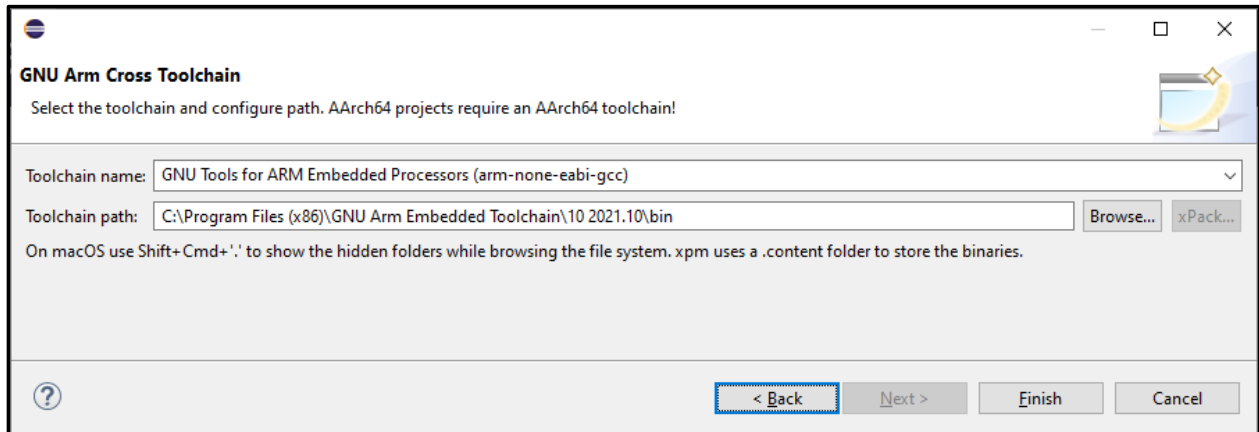
Launch Eclipse, create the workspace under the NXP_DEMO folder:



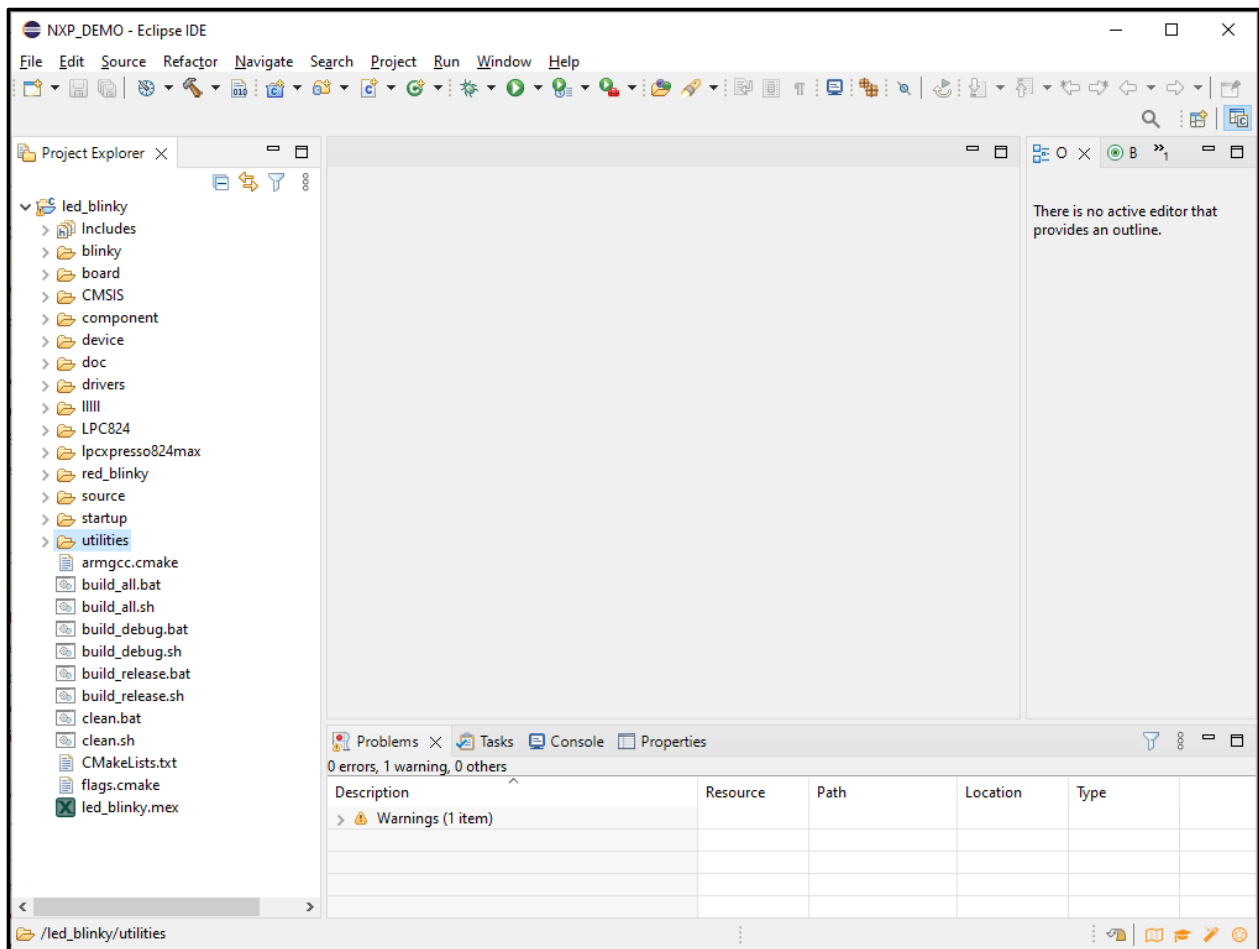
Select the *Create a new Embedded C/C++ project* → *C Managed Build*. For *Project name* select the **led_blinky** folder as all sources were generated under that folder, *Project type*; **EMPTY Project** and *Toolchains*: **ARM Cross GCC**:



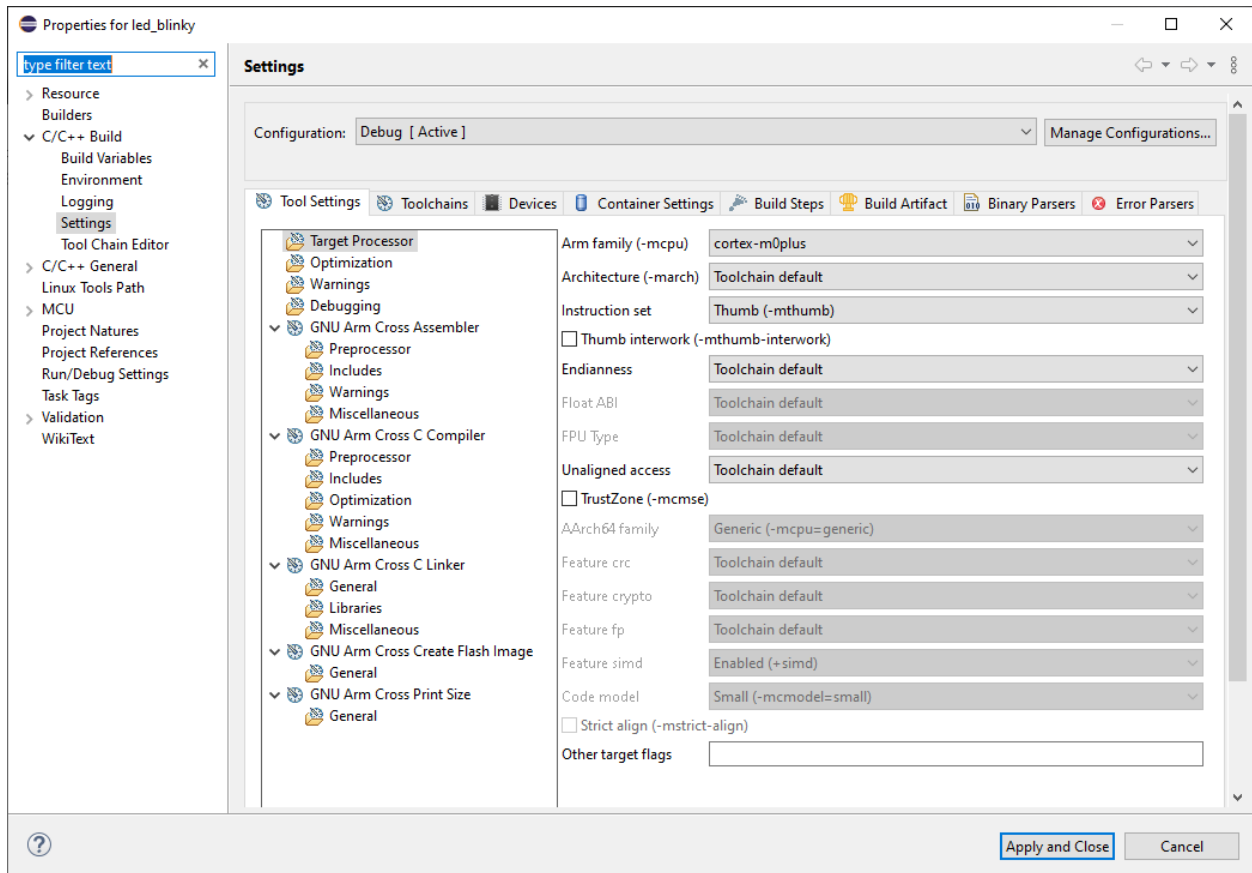
Ensure *Toolchain name* and *Toolchain path* are correct:



The project will import all generated files:



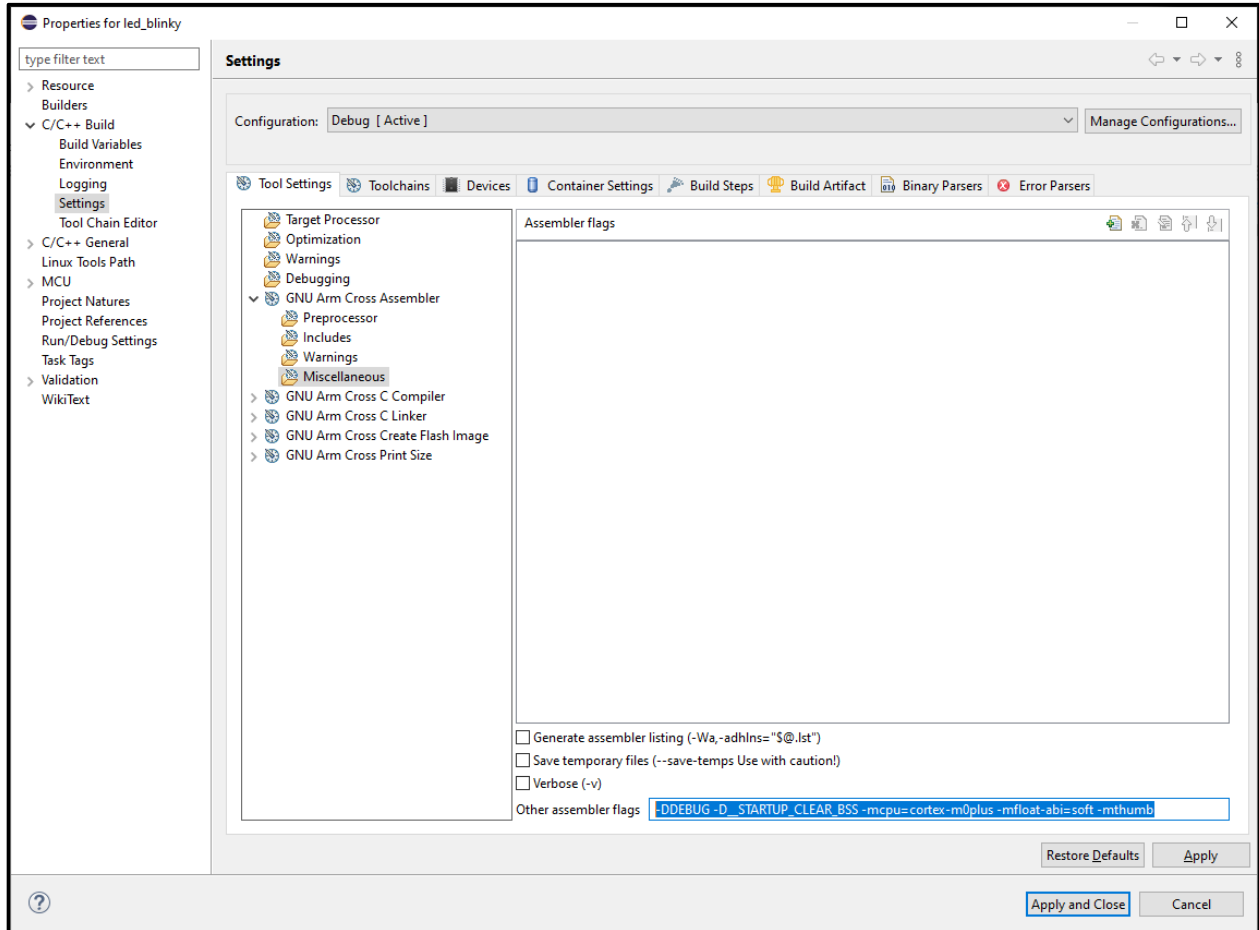
In the project properties ensure Arm family is correct “cortex-m0plus”:



3.1 Cross Assembler FLAGS

Properties→C/C++ Build→Settings→GNU Arm Cross Assembler, Other assembler flags:

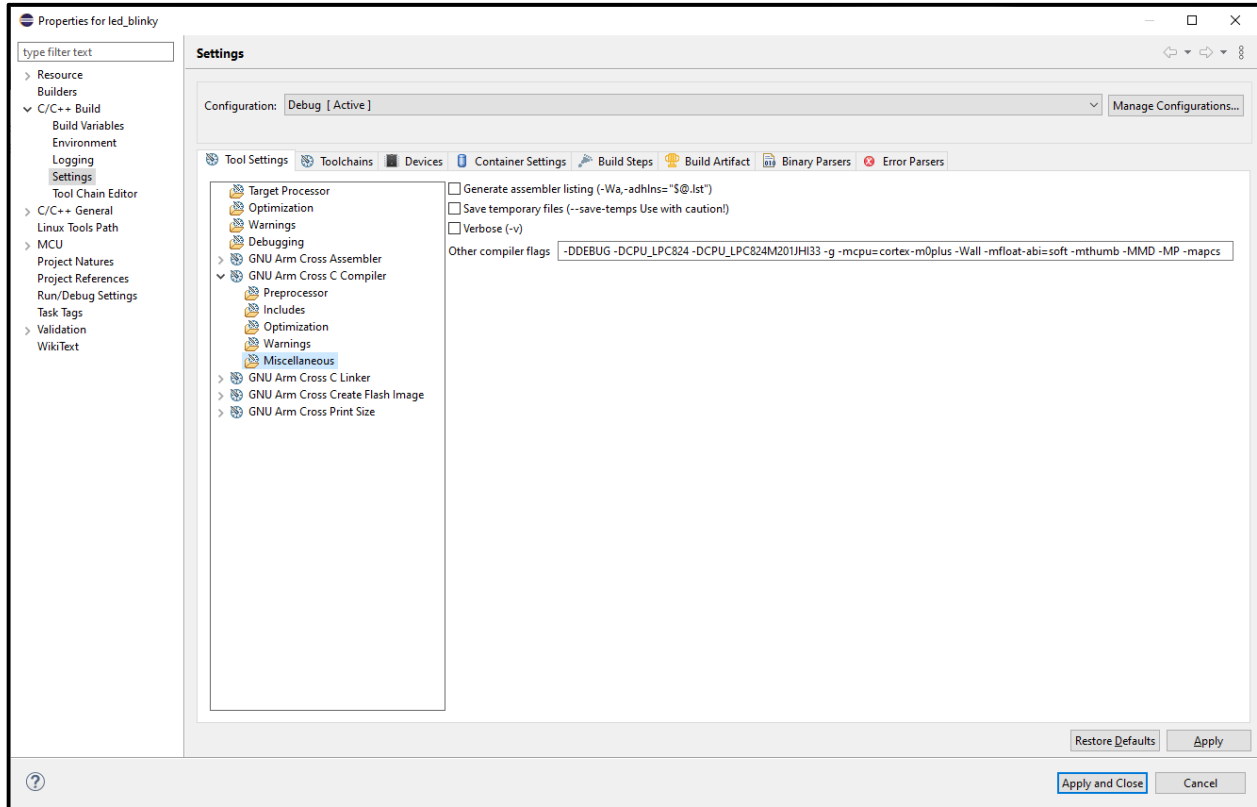
-DDEBUG -D__STARTUP_CLEAR_BSS -mcpu=cortex-m0plus -mfloat-abi=soft -mthumb



3.2 C Cross Compiler FLAGS

Properties→C/C++ Build→Settings→GNU Arm Cross C Compiler, Other compiler flags:

-DDEBUG -DCPU_LPC824 -DCPU_LPC824M201JHI33 -g -mcpu=cortex-m0plus -Wall -mfloat-abi=soft -mthumb -MMD -MP -mapcs



3.3 Cross Linker FLAGS

Properties→C/C++ Build→Settings→GNU Arm Cross C Linker, Other linker flags:

`-g -mcpu=cortex-m0plus -Wall -mfloat-abi=soft -fno-common -ffunction-sections -fdata-sections -ffreestanding -fno-builtin -mthumb -mapcs`

-Xlinker [option]:

`--gc-sections`

`-static`

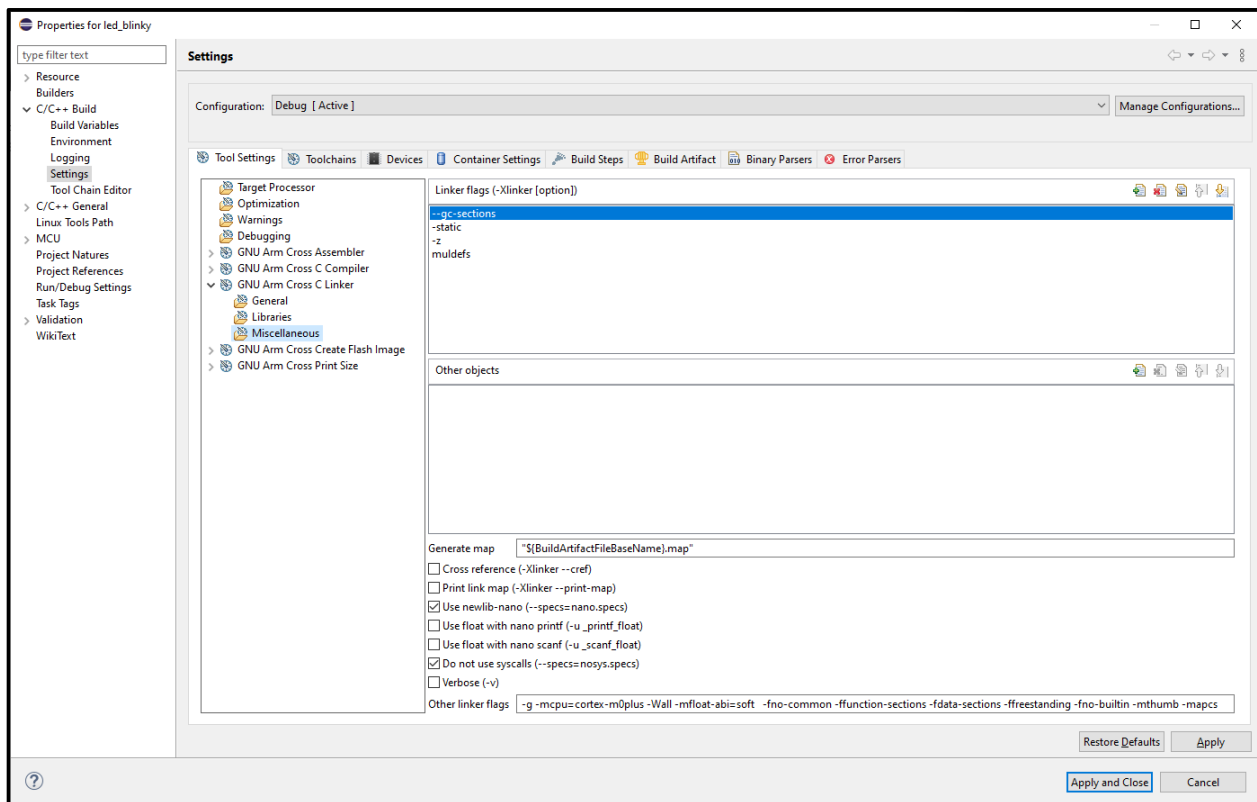
`-z`

`muldefs`

Check Boxes:

Use newlib-nano (`--specs=nano.specs`)

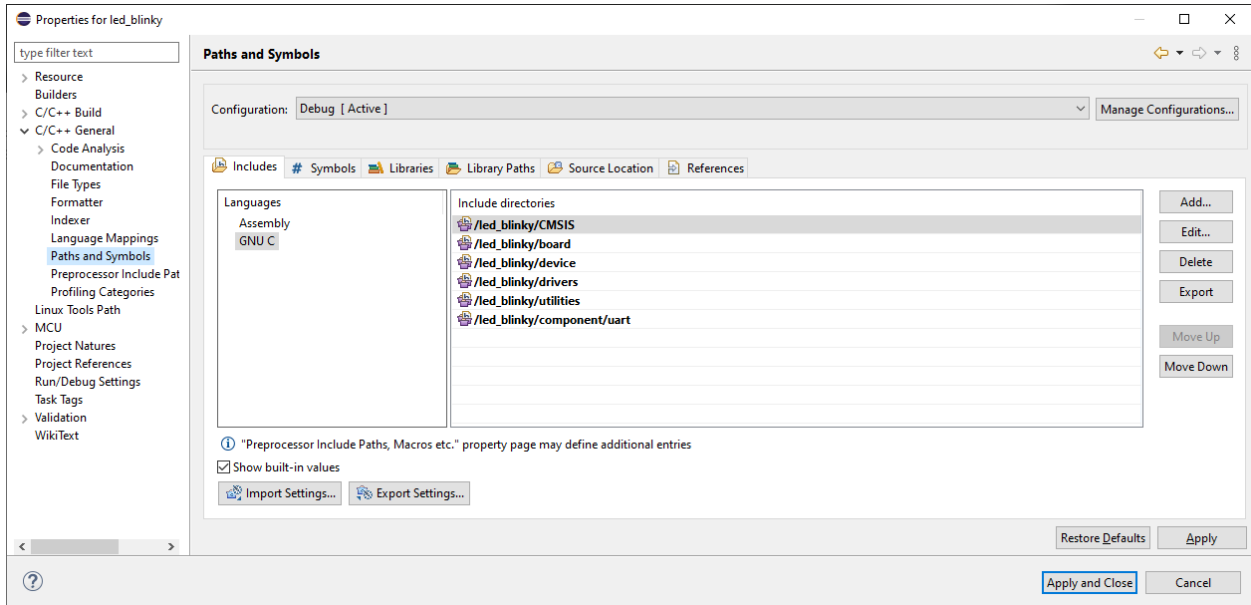
Do not use syscalls (`--specs=nosys.specs`)



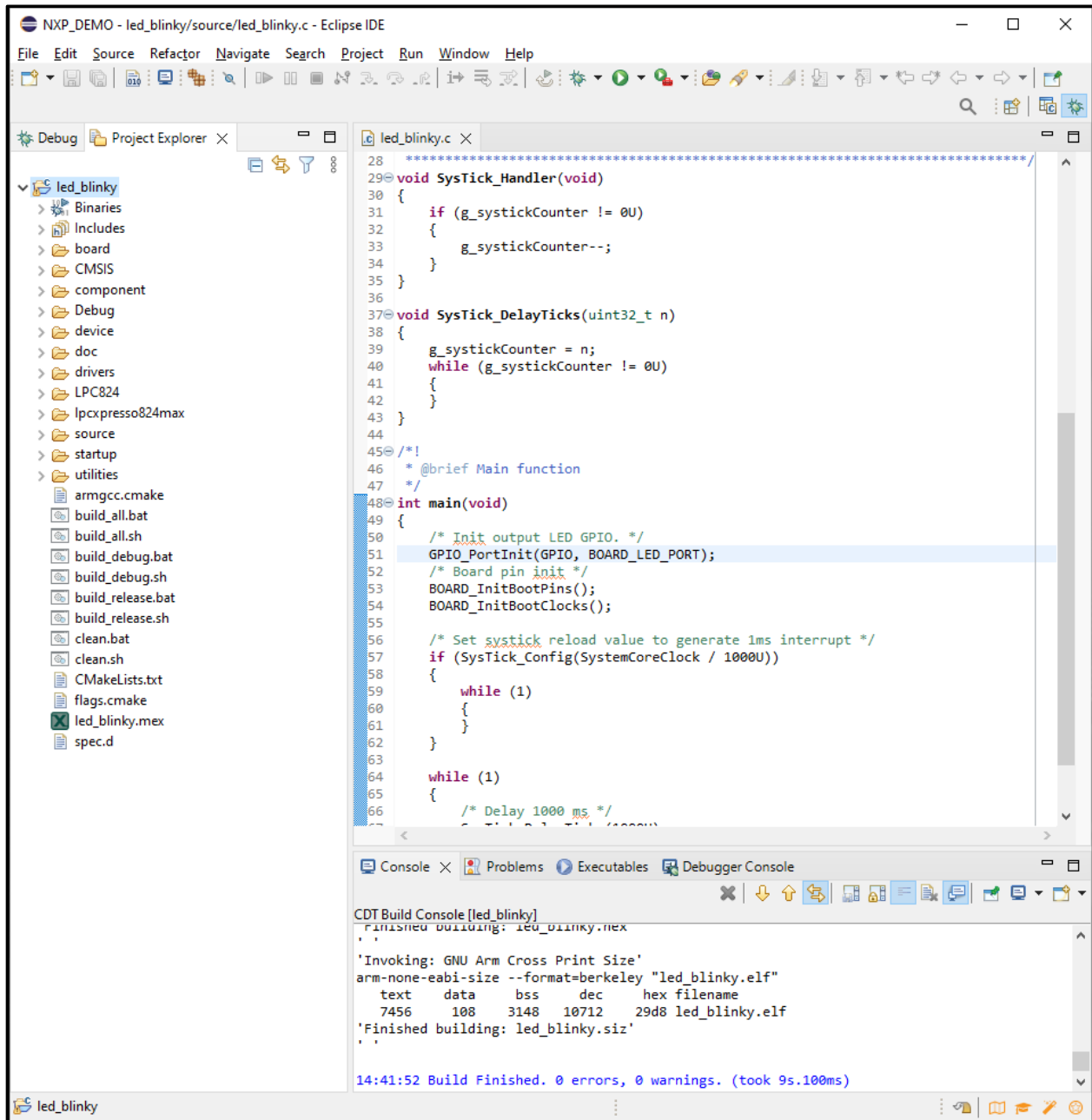
3.4 Include Paths

Properties→C/C++ General→Paths and Symbols→Includes:

- /led_blinky/CMSIS
- /led_blinky/board
- /led_blinky/device
- /led_blinky/drivers
- /led_blinky/utilities
- /led_blinky/component/uart



Finally build and debug:



Debug Configurations

Create, manage, and run configurations

Name: led_blinky Debug

Main Debugger Startup Source Common SVD Path

J-Link GDB Server Setup

Start the J-Link GDB server locally Connect to running target

Executable path: Browse... Variables...

Actual executable: C:/Program Files/SEGGER/JLink/JLinkGDBServerCL.exe

(to change it use the [global](#) or [workspace](#) preferences pages or the [project](#) properties page)

Device name: [Supported device names](#)

Endianness: Little Big

Connection: USB IP (USB serial or IP name/address)

Interface: SWD JTAG

Initial speed: Auto Adaptive Fixed kHz

GDB port:

SWO port: Verify downloads Initialize registers on start

Telnet port: Local host only Silent

Log file: Browse...

Other options:

Allocate console for the GDB server Allocate console for semihosting and SWO

GDB Client Setup

Executable name: Browse... Variables...

Actual executable:

Other options:

Commands:

Remote Target

Host name or IP address:

Port number:

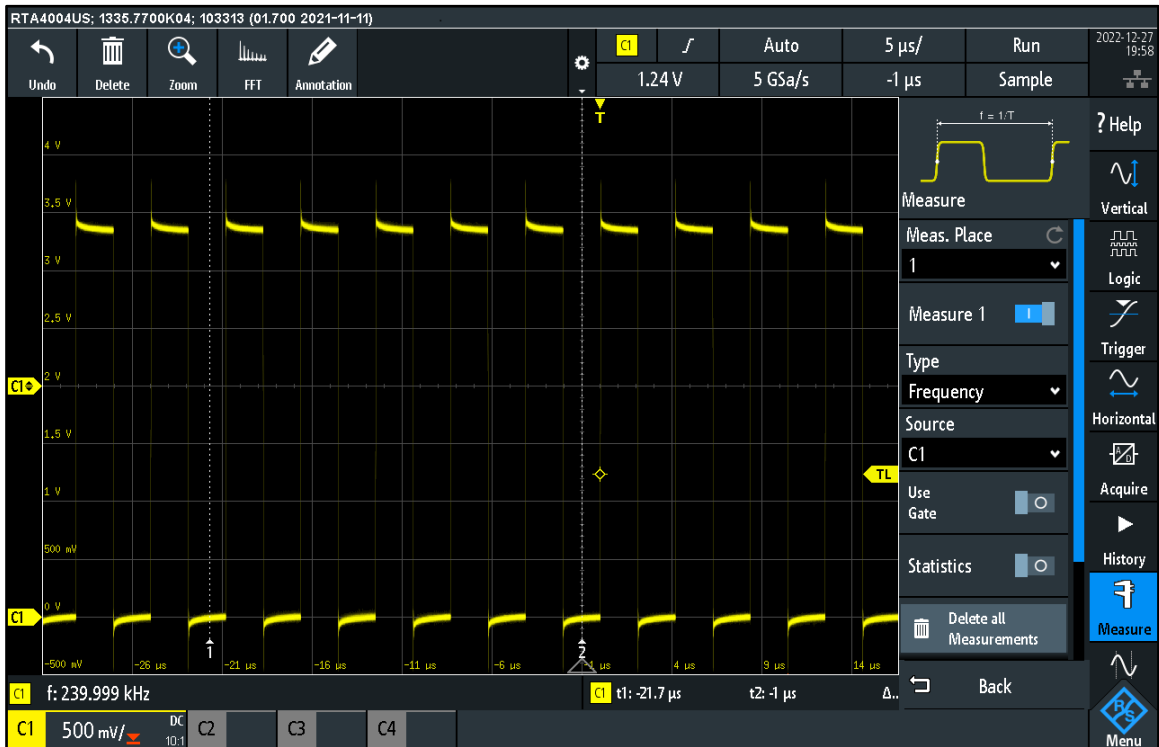
Force thread list update on suspend [Restore defaults](#)

Revert Apply

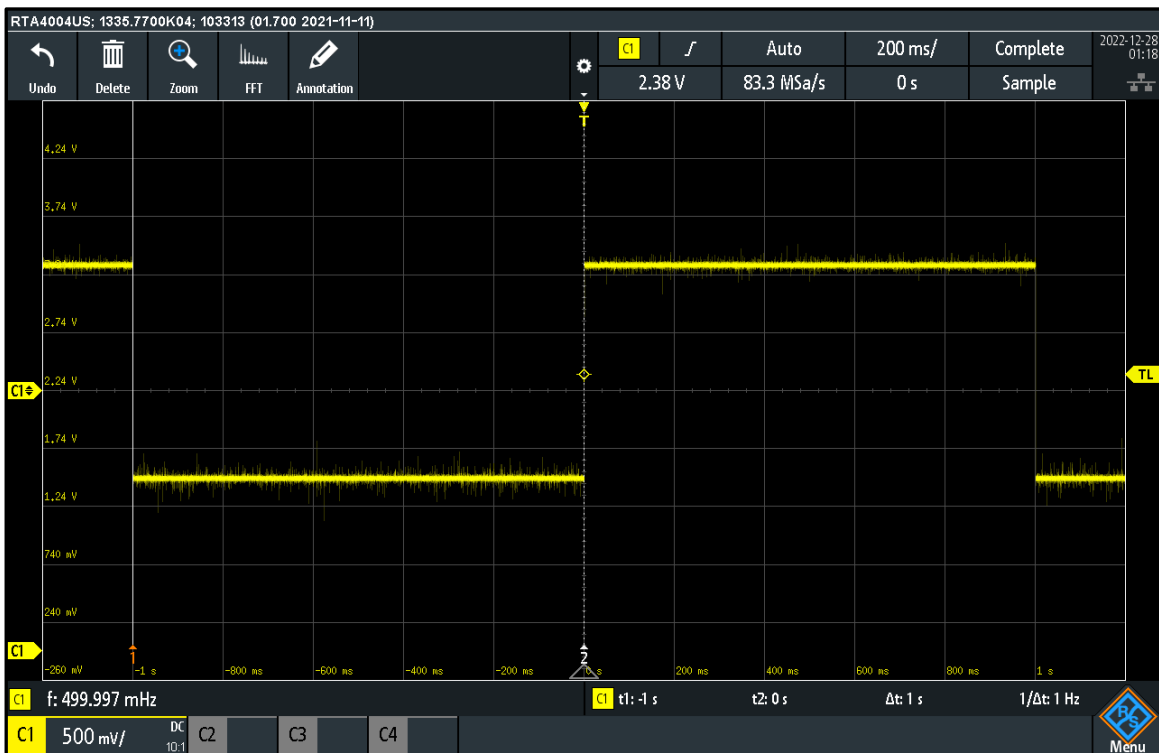
Debug Close

Filter matched 17 of 17 items

Using a scope, probe PIO0_0 P3.10 of the LPCXpresso 284MAX board. If clocks were configured correctly a 240 kHz square wave should be seen at this pin:



Additional verification, the RED LED should toggle every 1000 ms:



The solution is available upon request, reference AN122222-001.

support@wojotech.com