Application Note

MSPM0 Design Flow Guide

Eason Zhou

ABSTRACT

The application note provides clear steps when developing MSPM0 MCUs. It can help to find the related materials following the project progress. All of the instructions are in steps; tips will be given at the same time.

Table of Contents

List of Figures

List of Tables

Trademarks

1 Overview

The document is written in a flow in order to provide clear steps to every engineer in a project when developing MSPM0 MCUs. A list of the related documents are provided for further reference. Appendix A uses an example: an LED light shows the basic usage method of software Code Composer Studio™ (CCS) , guides you on how to generate a PCB footprint files and how to use MSP-GANG (Mass production tool).

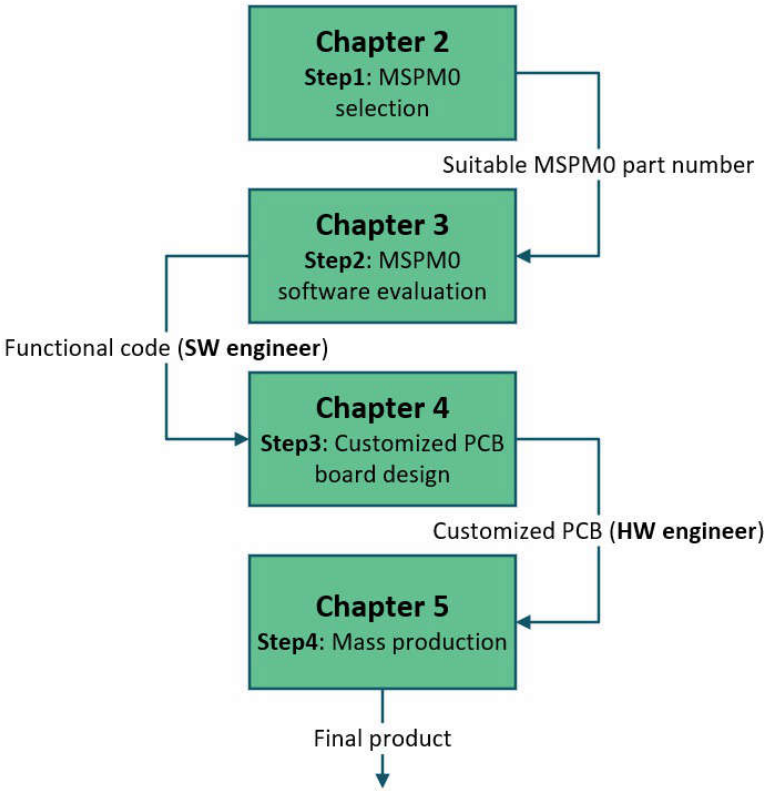

Figure 1-1. MSPM0 Design Flow

2 Step1: MSPM0 Selection

This step shows you how to find a suitable MSPM0 orderable number.

Here is the link for the MSPM0 device list. After you go to this page, you can use the filter on the left to do Initial screening based on your MCU peripheral requirement or you can directly go to the device page through the left search textbox.

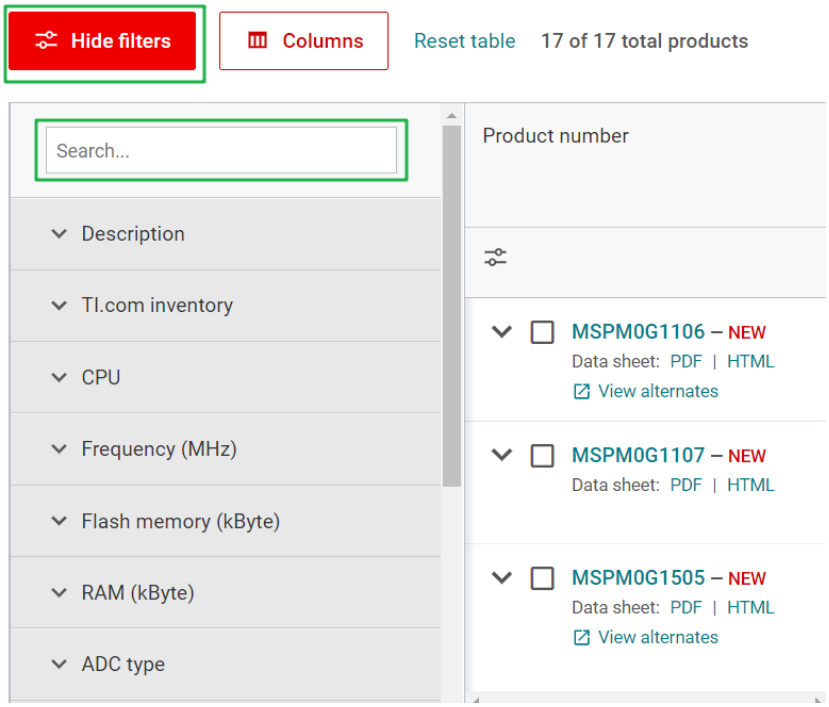

Figure 2-1. MSPM0 Device List

After you go to the **device page**, you can find more spec or functions details for this product. The key documents are the Data Sheet, Technical Reference Manual (TRM) and Errata. The device-specific data sheet introduces the parameters and functional data information of dedicated MSPM0. The device-specific TRM introduces the application method and characteristics of a series MSPM0. The device-specific errata introduces corrigendum description of MSPM0 related series or versions.

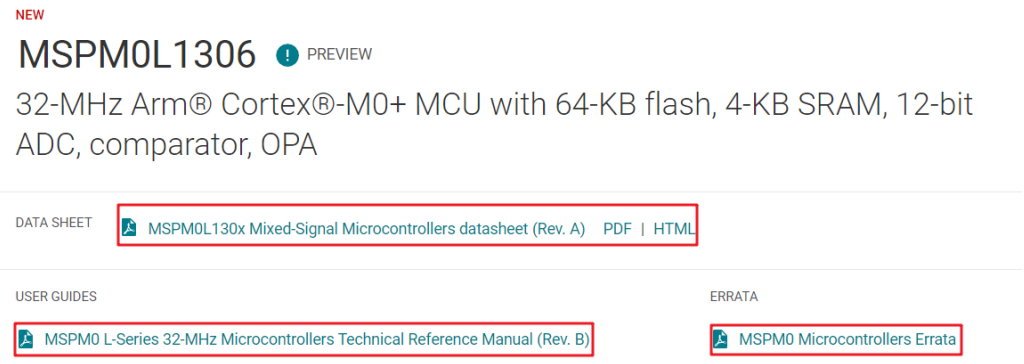

Figure 2-2. MSPM0 Important Document List

In the *Device Comparison* table in the device-specific data sheet, you can easily do a further screening by a quick compare between different part numbers.

5 Device Comparison

Table 5-1. Device Comparison

DEVICE NAME (1) (2)	FLASH / SRAM (KB)	QUAL (3)	ADC CH.	COMP	OPA	GPAMP	UART/I2C/SPI	TIMG	GPIOs	5-V TOL. IO	PACKAGE [BODY SIZE] (4)
MSPM0L1306xRHB	64 / 4	T/S	10	1	2	1	2 / 2 / 1	4	28	2	32 VQFN [5 mm × 5 mm] (5)
MSM0L1305xRHB	32 / 4										
MSM0L1304xRHB	16 / 2										
MSPM0L1306xDGS28	64 / 4	T/S	10	1	2	1	2 / 2 / 1	4	24	2	28 VSSOP [7.1 mm × 3 mm]
MSPM0L1305xDGS28	32 / 4										
MSPM0L1304xDGS28	16 / 2										
MSPM0L1346xDGS28	64 / 4	T	9						22		
MSPM0L1345xDGS28	32 / 4										

Figure 2-3. Device Comparison Table

For the orderable part number and its reference price, see the **Ordering & quality** part of the **device page**.

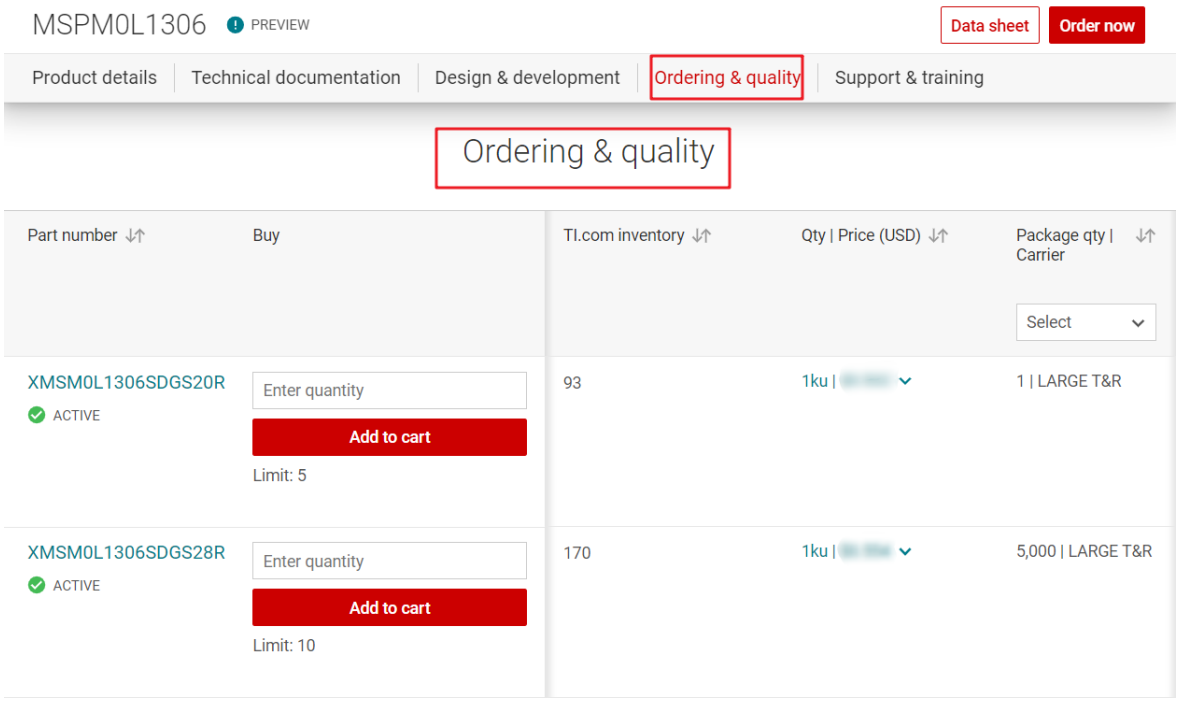

Figure 2-4. Ordering and Quality Part View

3 Step2: MSPM0 Software Evaluation

This step shows you how to setup a software evaluation environment for MSPM0. To get a step-by-step instruction based on CCS and launchpad, see Section A.1.

3.1 Hardware Setup

3.1.1 Debugger Selection

Here is a feature summery of different debugger examples supporting MSPM0. For XDS110 and XDS100 On-Board, they are TI's own debuggers. They will support more functions, like EnergyTrace™ (an energy measurement technology) and BSL than general debuggers. For XDS110 On-Board, it is the cheapest debugger solution, integrated into TI Launchpad. For more information, please refer to the next chapter Section 3.1.2.

Table 3-1. MSPM0 Debugger Compare

Features	XDS110	XDS110 On-Board	J-Link
cJTAG (SBW)	√	√	√
BSL tool	√	√	
Backchannel UART	√	√	
Power supply	1.8~3.6 V	3.3/5 V	5 V
IDE: CCS	√	√	√
IDE: 3**rd** party	IAR/Keil	IAR/Keil	IAR/Keil

3.1.2 Launchpad Introduction

It is suggested to start MSPM0 development with a launchpad first. Figure 3-1 shows an overview on the launchpad. It contains the MCU and a XDS110 debugger. You can also use other debuggers like J-Link to debug the MCU after removing the jumpers.

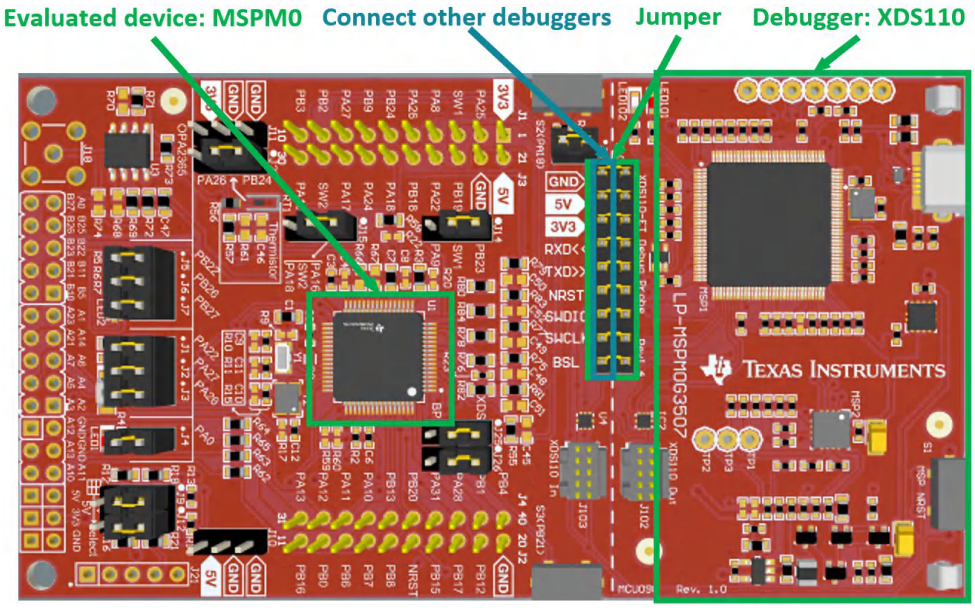

Figure 3-1. MSPM0G3507 Launchpad

Here is the orderable launchpad for a sub family MSPM0 evaluation. You can also find the device-specific EVM User's Guide with **schematic** under these links:

- LP-MSPM0L1306: Landing page, *MSPM0L1306 LaunchPad Development Kit User's Guide*
- LP-MSPM0G3507: Landing page, *MSPM0G3507 LaunchPad Development Kit User's Guide*

3.2 MSPM0 Software Setup

3.2.1 Install IDE

Table 3-2 shows a quick compare for different IDEs supporting MSPM0, which are CCS, IAR and Keil. CCS is TI's own IDE. It is suggested for customers to use CCSTUDIO, which is more mature than CCSTUDIO-THEIA.

Table 3-2. MSPM0 Related IDE Compare

IDEs	CCSTUDIO / CCSTUDIO-THEIA	IAR	Keil
License	**Free**	Charged	Charged
Compiler	TI Arm® Clang / GCC	IAR C/C++ Compiler for Arm	Arm Compiler Version 6
XDS110	Supported	Prepare to support	Prepare to support
J-Link	Supported	Supported	Supported

Here are the related guides for different IDEs. It shows how to do setup and build projects on the related IDE.

- CCS quick start guide
- CCS IDE guide for MPSM0
- IAR quick start guide
- IAR IDE guide for MSPM0
- Keil quick start guide
- Keil IDE guide for MSPM0

For quick start guides, they are under the sub document "MSPM0 SDK Quick Start Guides", which will show how to setup the environment and run the first example on the related IDE.

For IDE guides, they will show the related development knowledge beyond the quick start guides. It is suggested that you to read the quick start guide first and then go through IDE guide for high level MSPM0 development knowledge.

3.2.2 Install SDK

Figure 3-2 shows the structure of the MSPM0 SDK (offline) version, which is shown in the folder type.

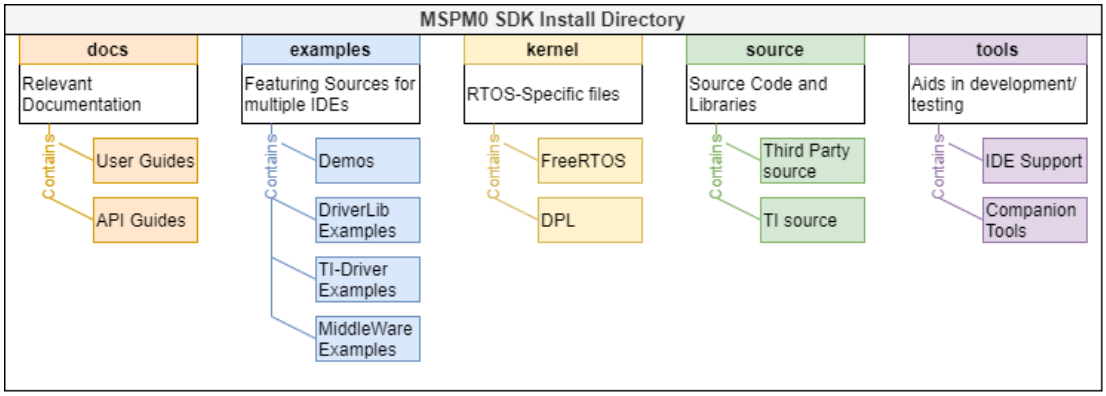

Figure 3-2. SDK Structure

Most folders are example and doc folders. This section focuses on introducing them, as shown in Figure 3-3.

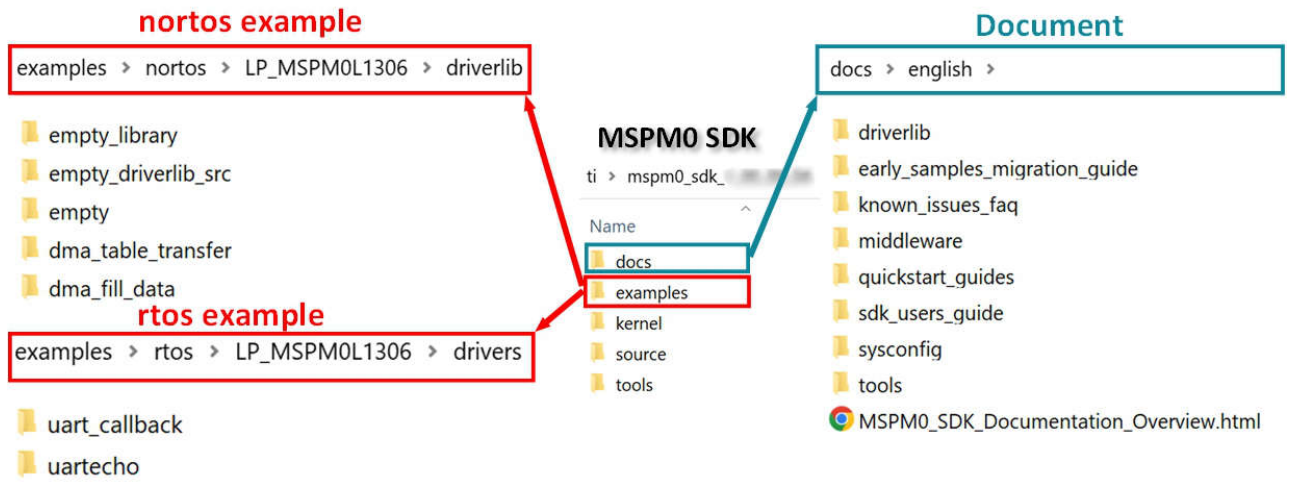

Figure 3-3. MSPM0 SDK

TI makes a LaunchPad for one MSPM0 sub family, with a superset MSPM0 on board. For this MSPM0 sub family, the same example code is reused under the address "mspm0_sdk_x_x_x_x \ examples \ nortos \ LP_MSPM0xxxx" for the nortos example and "mspm0_sdk_x_x_x_x \ examples \ rtos \ LP_MSPM0xxxx" for rtos example. Table 3-3 shows all the example code type.

Table 3-3. MSPM0 Example Coverage

Supported by SDK	Platform 1		Platform 2	Platform 3
IDE	CCS		Keil	IAR
Compilers	TI Arm-Clang	GNU Arm (GCC)	Arm/Keil Compiler	IAR Arm compiler
RTOS	FreeRTOS			
Code examples	Driverlib (Low level driverlib)/TI Dirvers(High level driverlib)			

In the rtos example level, there are two folders. The most important folder is drivers that demos the peripheral control based on TI-Driver.

In the nortos example level, there are many folders. The most important folder is driverlib folder, which contains the peripheral example code based on Driverlib, as shown in Figure 3-3. For demo folder, it contains the example codes that use above one MCU peripherals. For other folders, they are all belong to the middle ware and contains solution for typical application. In nortos examples, you can also find four empty examples for users to build their own project.

Table 3-4. Empty Project Description

Example	Type	Launguage	Use Sysconfig	Library Files in Project
empty	Project	C++	Yes	No
empty_cpp	Project	C	Yes	No
empty_library	Static library (.lib file in Debug folder after debugging)	C	No	No
empty_driverlib_src (**Suggested**)	Project	C	Yes	Yes

The structure and important documents for the doc folders are shown in Figure 3-4. It is strongly suggested that you read these documents first when encountering any questions about SDK .

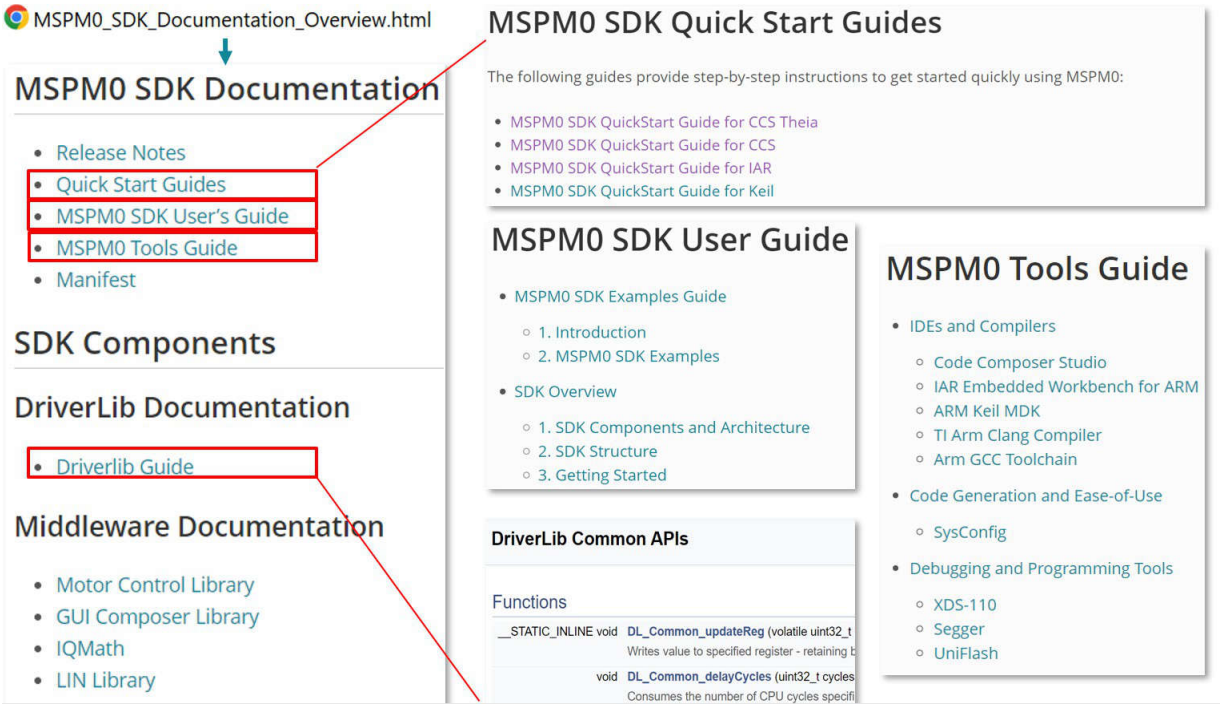

Figure 3-4. Document Overview

Here are the online links for the most important documents:

- MSPM0 SDK User Guide
- MSPM0 Tools Guide
- Driverlib API Guide

3.3 MCU Function Evaluation

After the hardware, IDE and SDK is prepared. MSPM0 evaluation can started.

3.3.1 Fix Pin Functions and Generate Setting Code With Sysconfig

Sysconfig is a tool that can help to generate the peripheral setting code shown in Figure 3-5. Here is the use flow:

- Add the wanted peripherals in “peripheral usage”
- Set its parameters in “Peripheral setting”, paired with the device-specific TRM.
- After debugging, it can generate C code directly.

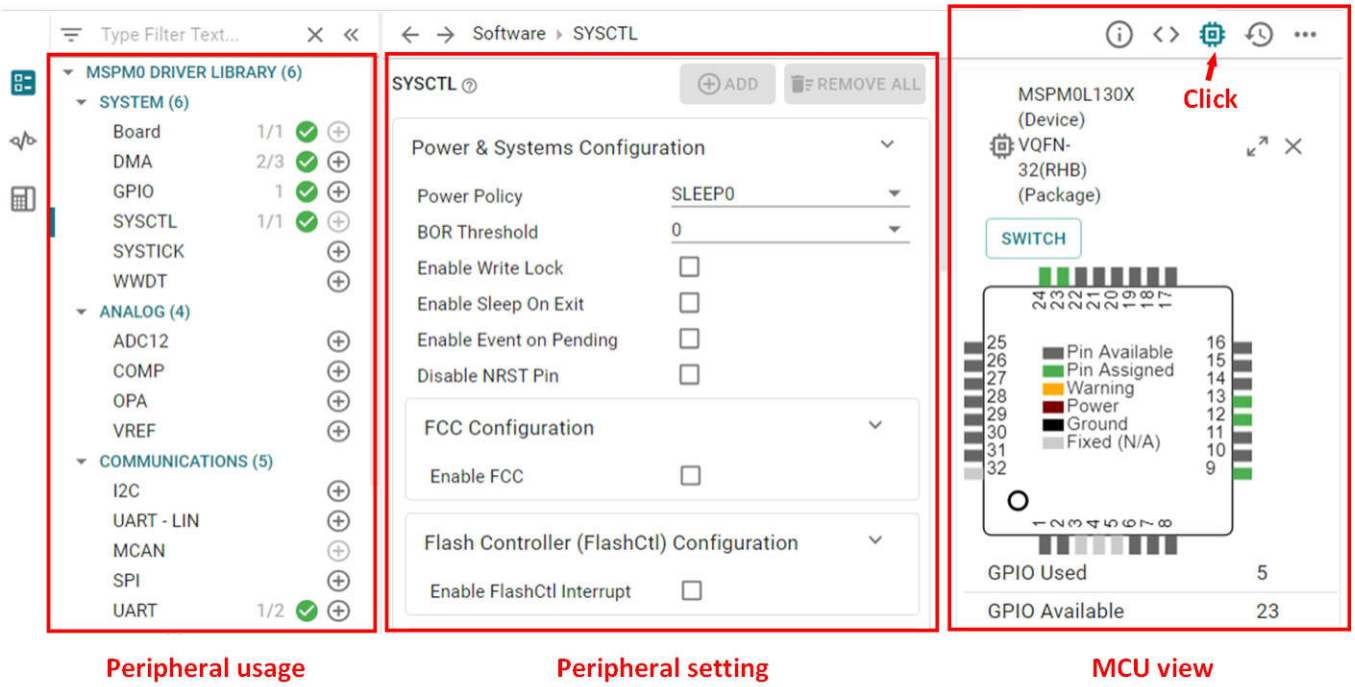

Figure 3-5. Sysconfig Overview

Another advantage of sysconfig is that it can help check the peripheral resource and the function usage conflicts. Using the MCU view, paired with the device-specific data sheet, it can easily help to fix the MCU pin functions.

For more details to use Sysconfig under different IDE, see SysConfig Guide for MSPM0.

3.3.2 Coding and Porting

This section provides a description of the MSPM0 example code based on CCS. The most important files are in red.

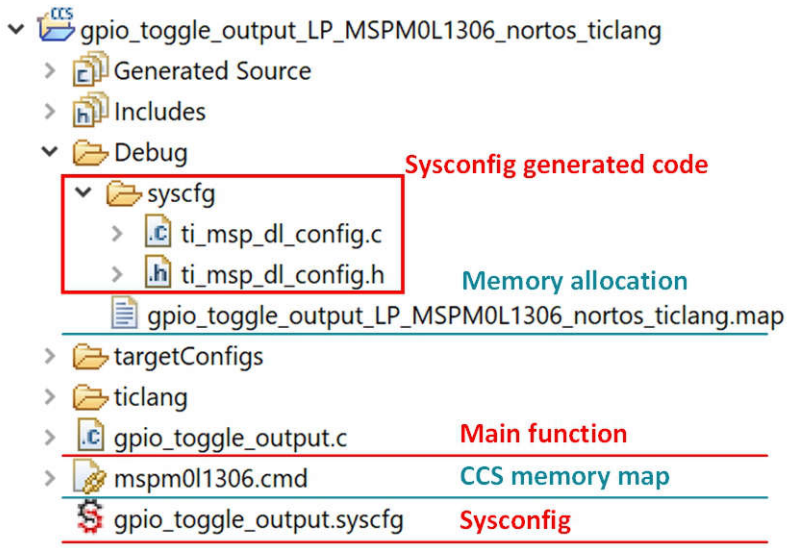

Figure 3-6. CCS Project Overview

Note that syconfig only generates the peripheral setting code. For peripheral usage, see the rtos/nortos code example in SDK, TRM, and MSPM0 driverlib API guide.

Here are the common questions about the first use of CCS:

- **Cannot set break point**: Change optimization level under this address (Your project > Properities > Build > Arm Complier > Optimization level)
- **Output Hex / Bin files**: See the Section 5.1.
- **Cannot find the .c / .h files in a folder**: Add the folder into the compile search path (Your project > Properties > Build > Arm Complier > Include Options)
- **Cannot important a project or its name becomes gary**: Delete the project with the same name under the workspace.
- **Enable Non-main change**: Change the erase type under this address (Your project > Properties > Debug > MSPM0 Flash Setting > Erase MAIN and NONMAIN)

When you finish the basic code development and want to move the project from LaunchPad to the dedicated MSPM0 under the same sub family, you need to click the "Switch" button in MCU view, as shown in Figure 3-5. Then, you need to copy the related linker file or start up file into the project from SDK, as shown in Figure 3-7.

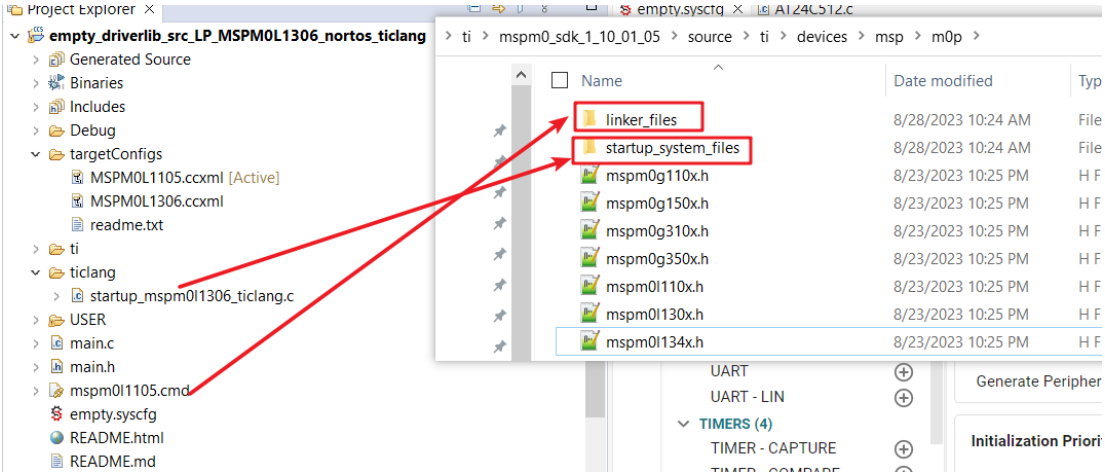

Figure 3-7. Head File and Start File Porting

For other questions, you can check whether the MSPM0 materials summery (*MSPM0 MCUs Quick Reference Guide*) can provide some help. You can also get support from online technical support forum: E2E. TI engineers provide response in 24 hours on this online support platform.

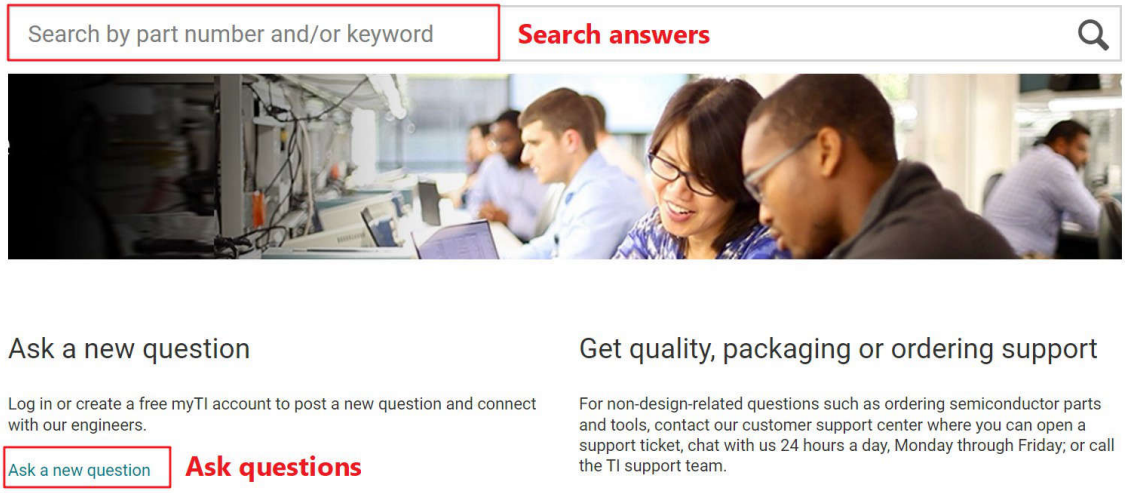

Figure 3-8. E2E Online

4 Step3: Customized PCB Board Design

4.1 Get MSPM0 Package

The easy way to get the MSPM0 package is to use the Ultra Librarian tool on TI.com, as shown in Figure 4-1. For the detailed instruction, see Appendix A.

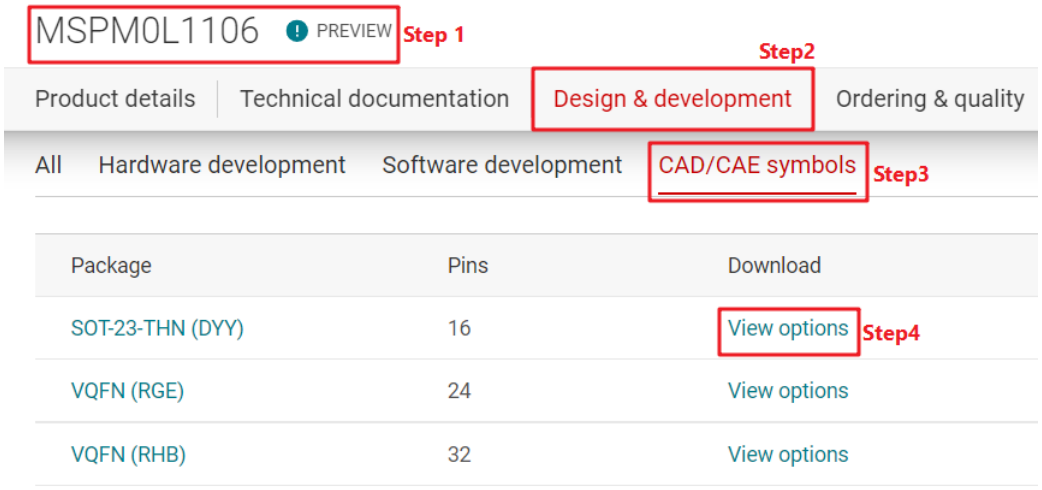

Figure 4-1. Ultra Librarian Tool Entrance

4.2 Fix Pin Functions

It is suggested to use the MCU view of sysconfig to help you fix the pin function with software engineer.

Figure 4-2. Sysconfig MCU View

4.3 Schematic and PCB Generation

Figure 4-3 shows the minimum requirement (Power + Reset + Vcore) with suggested values for MSPM0 hardware setup.

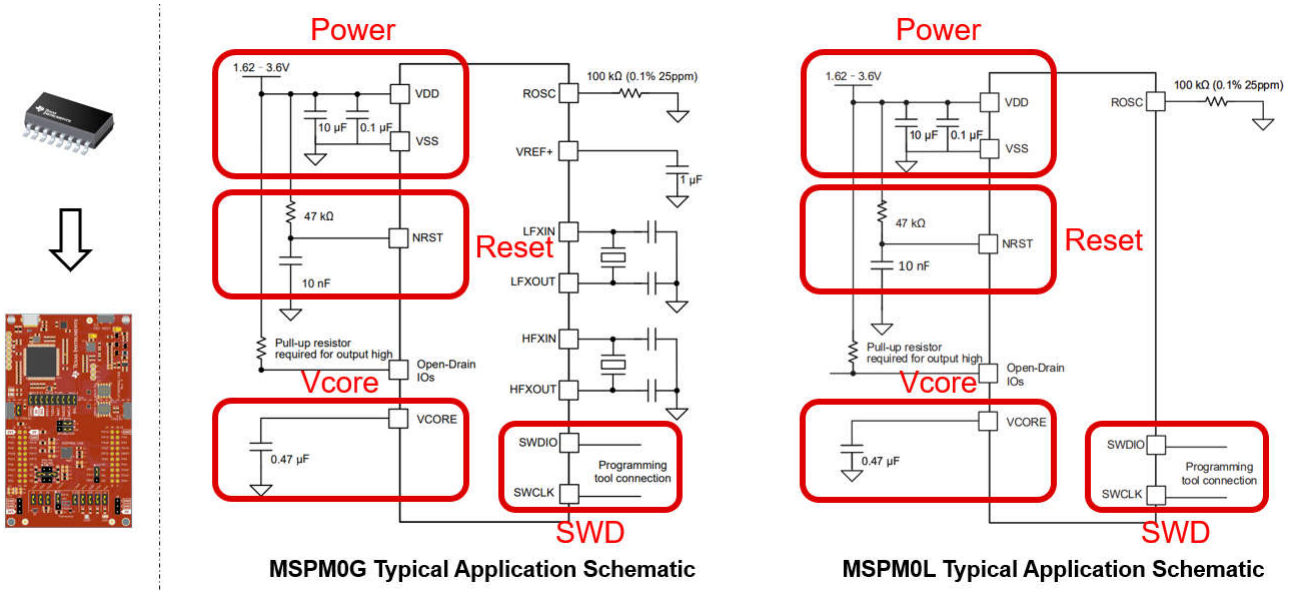

Figure 4-3. MSPM0 Minimum System

Figure 4-4 shows other attentions when designing the schematic in Unable to auto-generate link text.

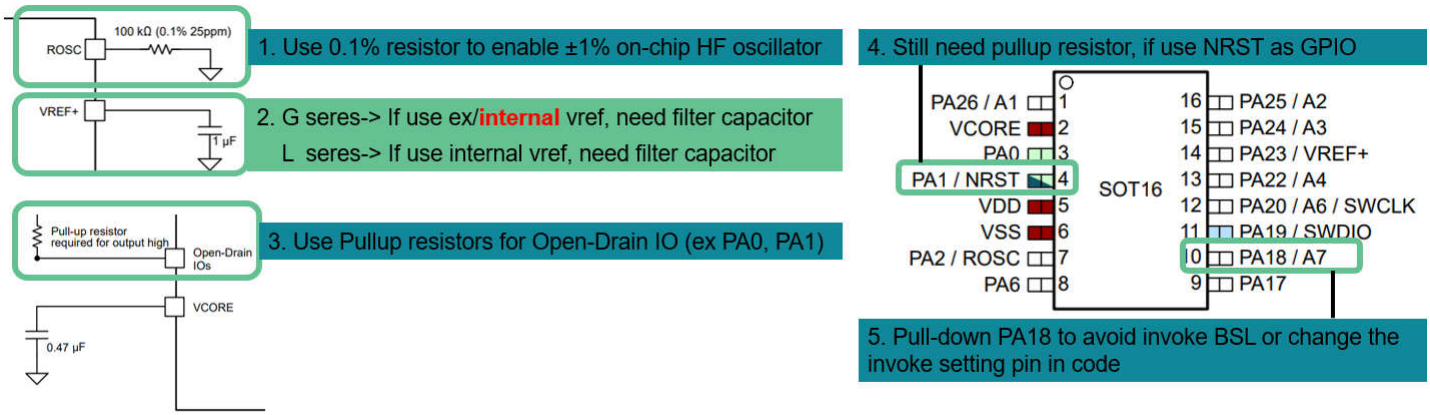

Figure 4-4. MSPM0 Schematic Other Attentions

For further schematic or PCB design reference, see:

- *MSPM0 L-Series MCUs Hardware Development Guide*
- *MSPM0 G-Series MCUs Hardware Development Guide*
- Device-specific MSPM0's Launchpad EVM user's guide
- Device-specific MSPM0's Data Sheet

5 Step 4: Mass Production

5.1 Generate Production Files

Figure 5-1 illustrates the steps to generate production files (.bin, .txt, .TI_TXT) based on CCS.

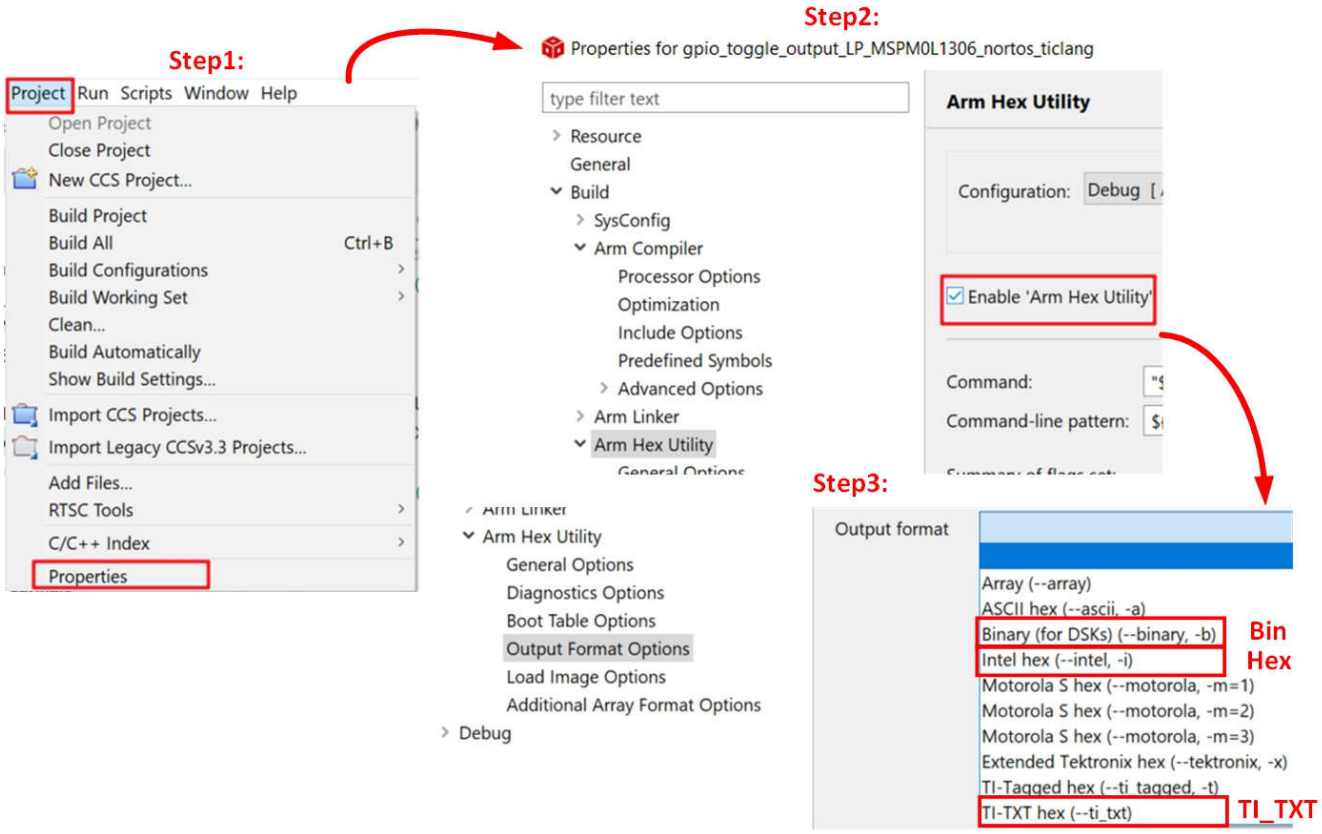

Figure 5-1. Create Program Files

5.2 Program Software and Tools Selection

Figure 5-2 shows an overview of the program software and tools. The available interface is JTAG (SWD) and Bootloader (BSL).

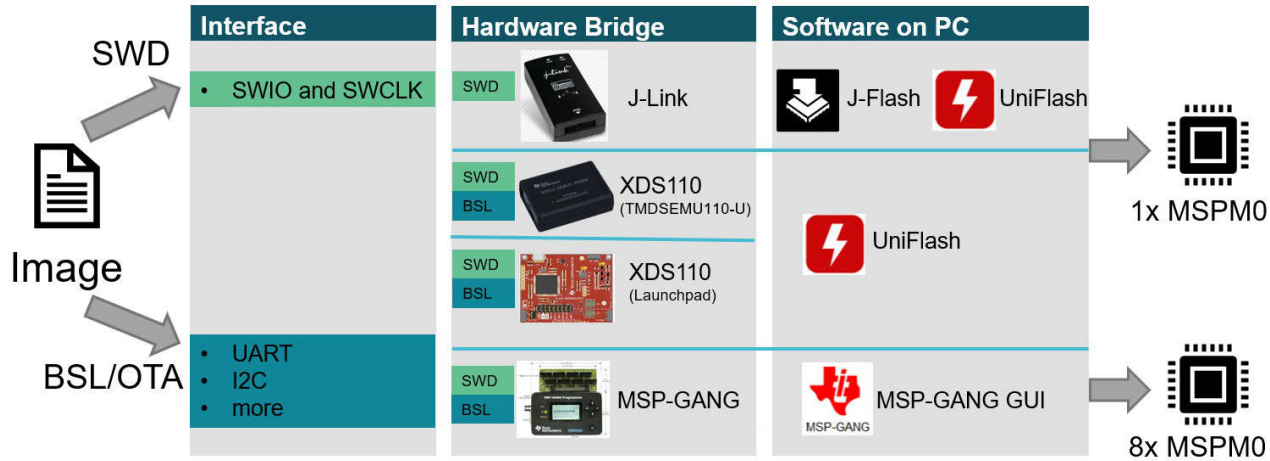

Figure 5-2. Program Software and Tool

J-Link only supports SWD. You can pair J-Flash or UniFlash with it to download the code.

TI uses a hardware bridge for standalone XDS110. Figure 5-3 shows the pin that is used. It only supports UniFlash.

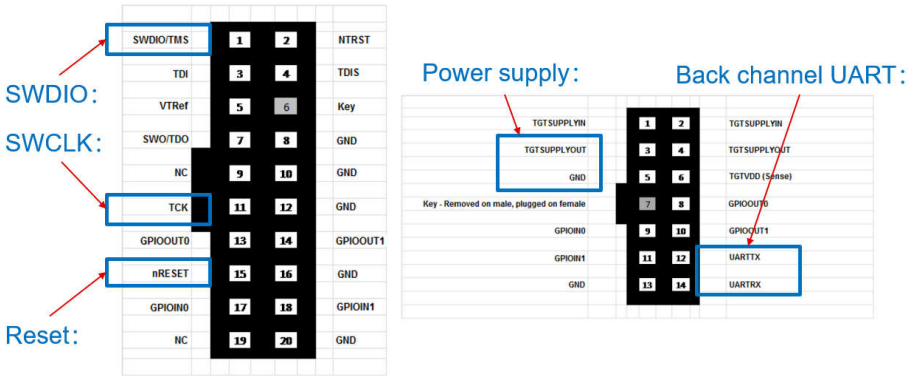

Figure 5-3. Pin Connection of Standalone XDS110

XDS110 on LauchPad keeps the basic programming functions compared to the standalone XDS110, as shown in Figure 5-4. That means you can also use it for mass production.

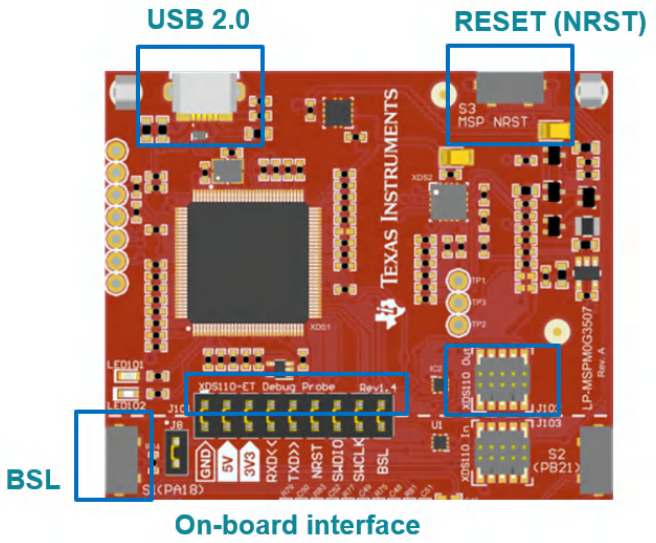

Figure 5-4. XDS110 on Board

MSP-GANG is a production solution, combining a GUI and a hardware bridge. It can program up to eight MSP devices at one time without PC GUI control. For more information on how to use this, see Section A.3. For more information, see the MSP-GANG product page.

Uniflash is TI's own program tool. It is a standalone tool used to program on-chip Flash memory on TI MCUs. Here are the references:

- Uniflash chapter in MSPM0 Tools Guide

A Appendix

A.1 Light a LED and CCS Quick Introduction

This sections shows how to light an LED based on CCS at beginning. A short description to CCS is also provided, in order to let you ramp up with the tool as soon as possible.

A.1.A Install CCS and SDK

Here are the important steps and tips for CCS installation.

1. Download CCS (**above 12.2 version**) and stat installation, and keep pressing next.

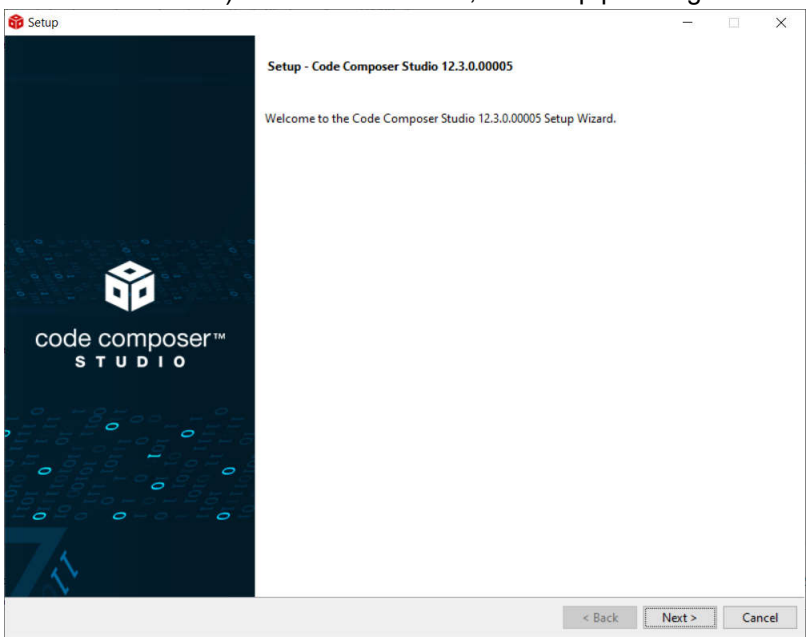

Figure A-1. CCS Installation

2. Select MSPM0 support component.

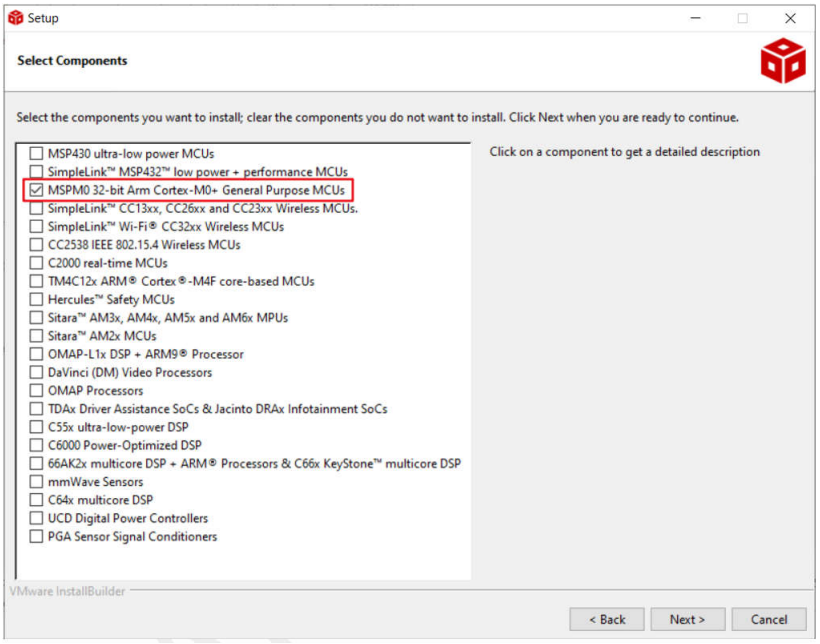

Figure A-2. MSPM0 Support Selection

3. Select J-link if needed.

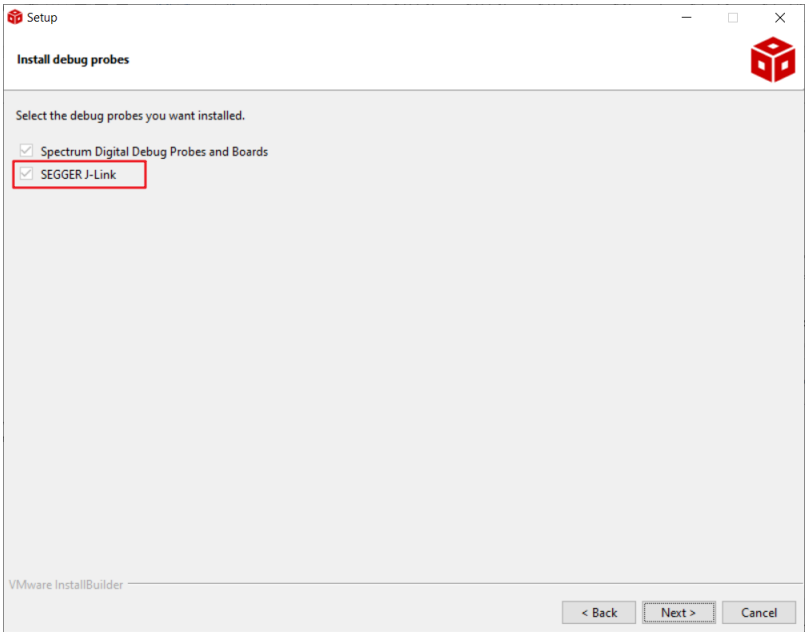

Figure A-3. J-Link Selection

4. Install MSPM0 SDK.

A.1.B Hardware Setup

Get a launchpad and plug in the computer.

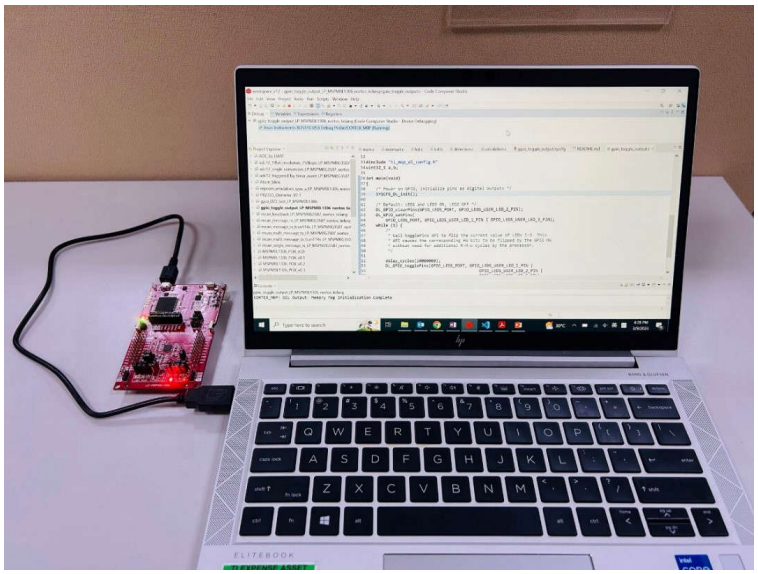

Figure A-4. Hardware Setup

A.1.B Code Import

1. Open CCS. The workspace means the address where to copy your imported project.

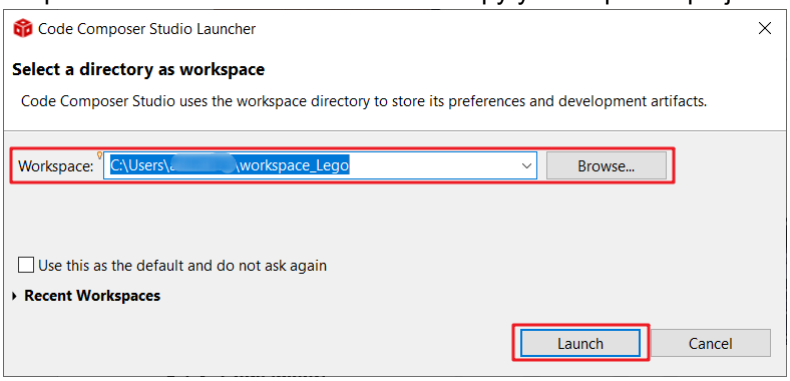

Figure A-5. Choose CCS Workspace

2. Import the general-purpose input/output (GPIO) toggle project with the TI-Clang compiler.

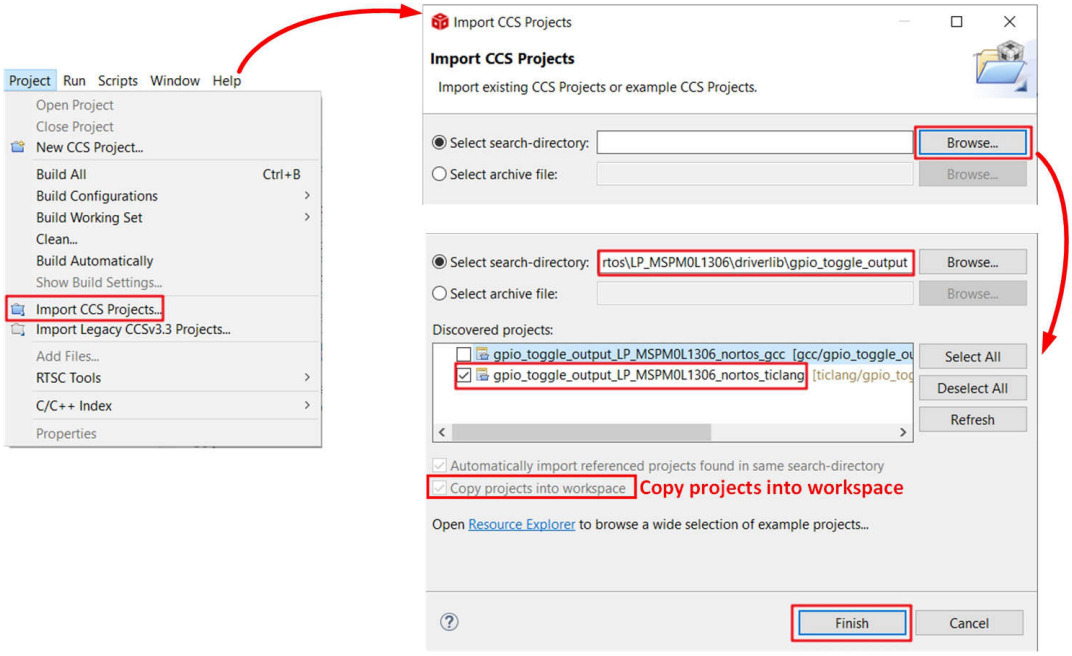

Figure A-6. Import Project

3. If it cannot be imported, delate the same name project under workspace.

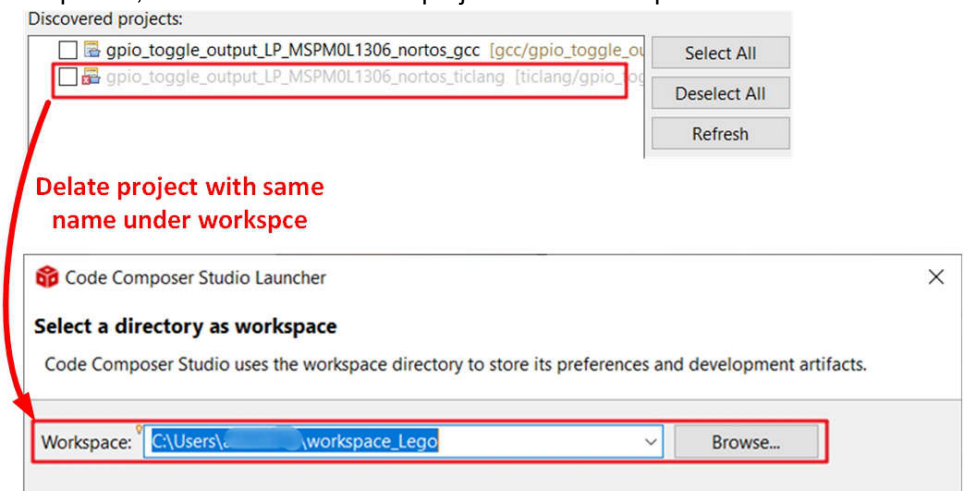

Figure A-7. Remove Duplicated Project

A.1.B Debug and CCS Quick Introduction

1. Start debug, then you can see GPIO toggle on the LP.

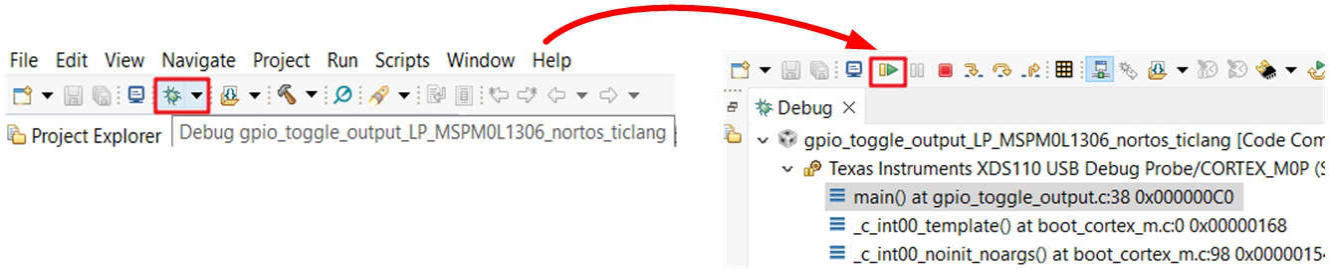

Figure A-8. Debug Code

2. Here we give a quick introduction to CCS functions.
   a. Project properties common used settings:

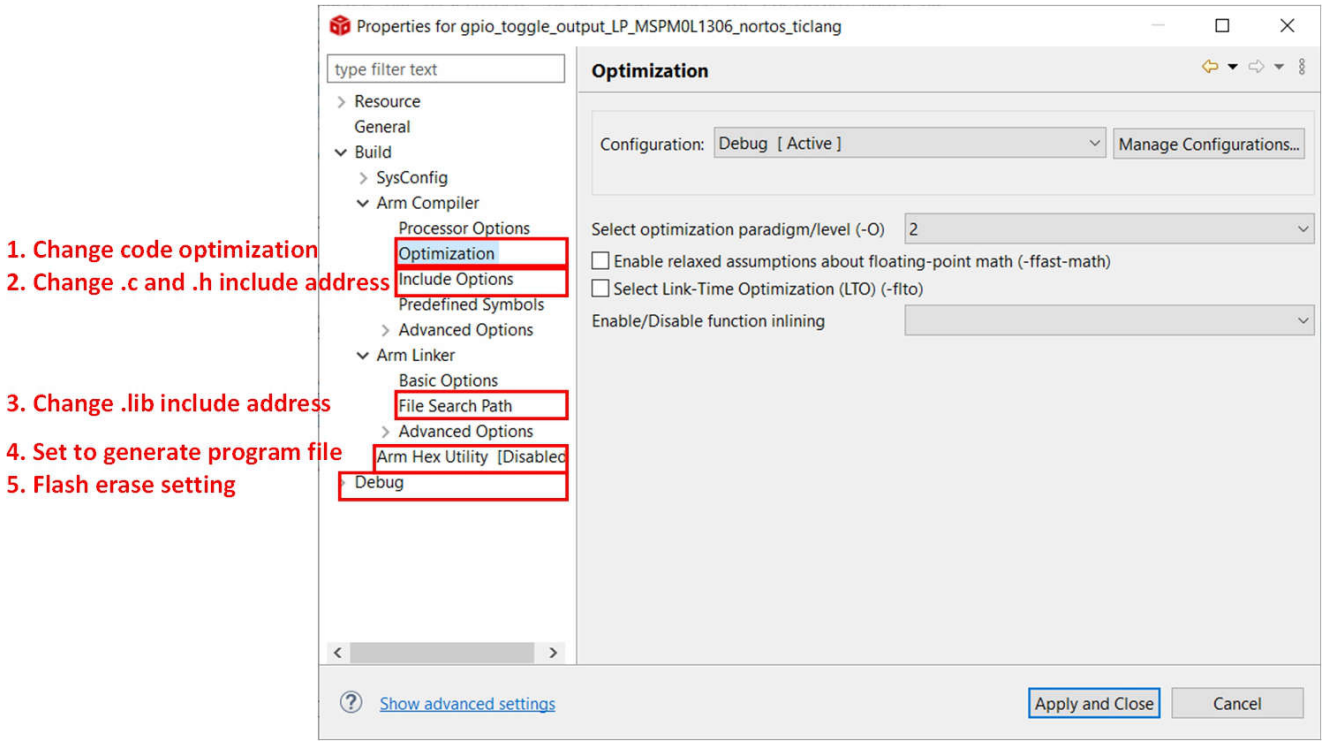

Figure A-9. Common Used Project Settings

   b. Debug common used functions.

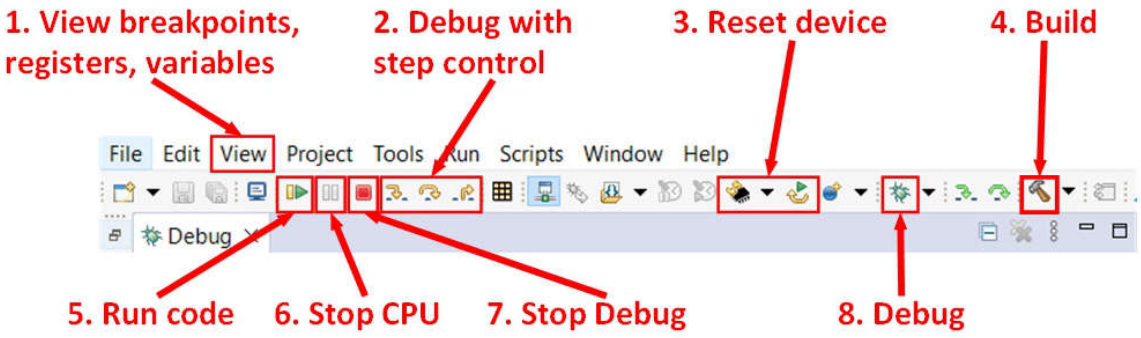

Figure A-10. Common Used Debug Functions

A.2 Steps to Generate PCB Library

1. Go to the entrance of Ultra Librarian tool under the MSPM0 device page using the steps shown in Figure A-11.

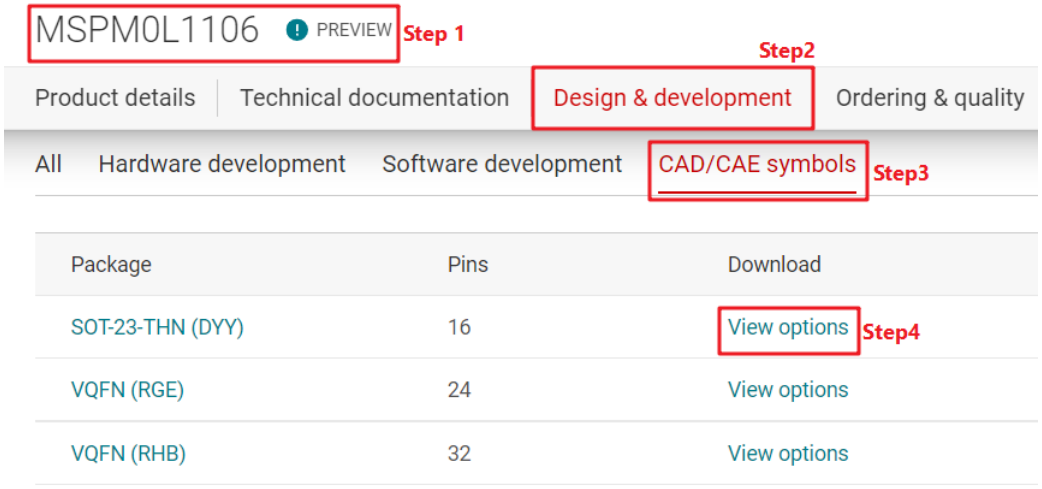

Figure A-11. Ultra Librarian Tool Entrance

2. Select your wanted CAD format and Pin ordering, then you can get the Altium design lib file.

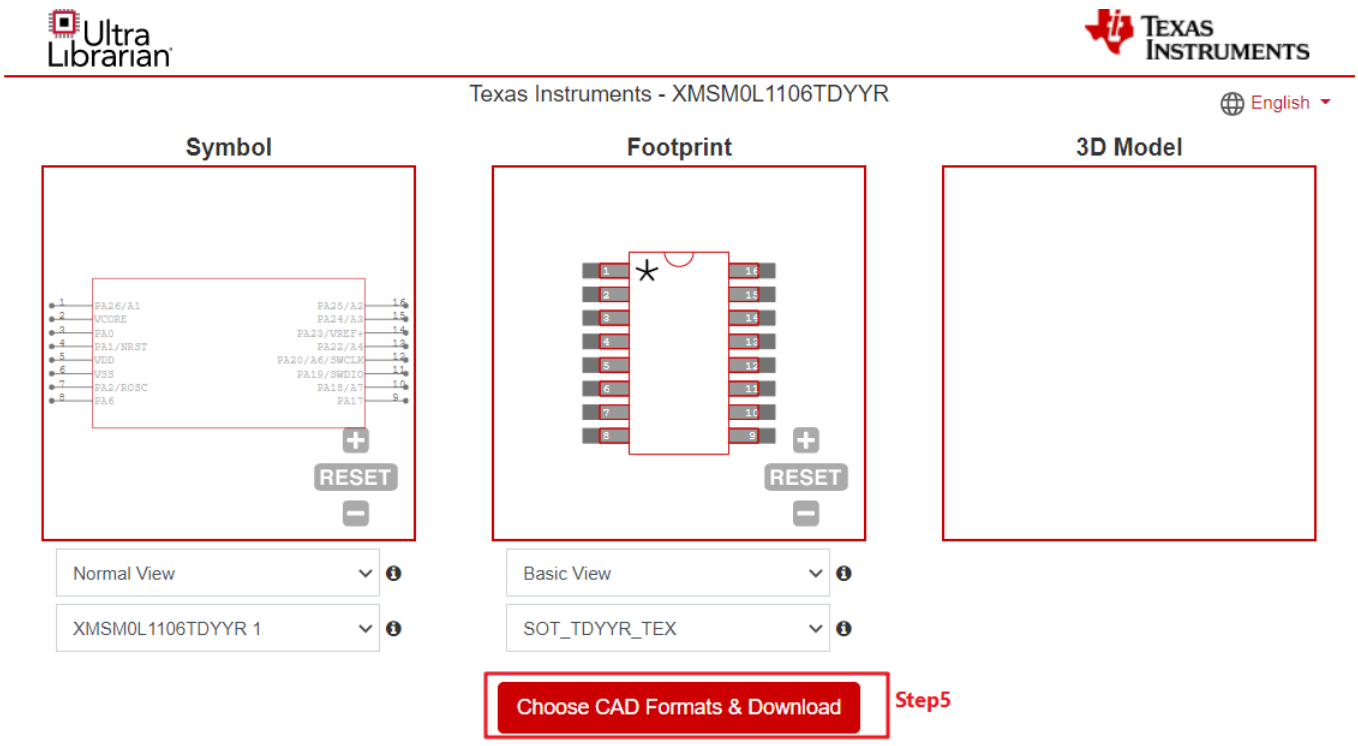

Figure A-12. Ultra Librarian Tool Device Selection

3. Here, Altium Designer lib is used as an example.

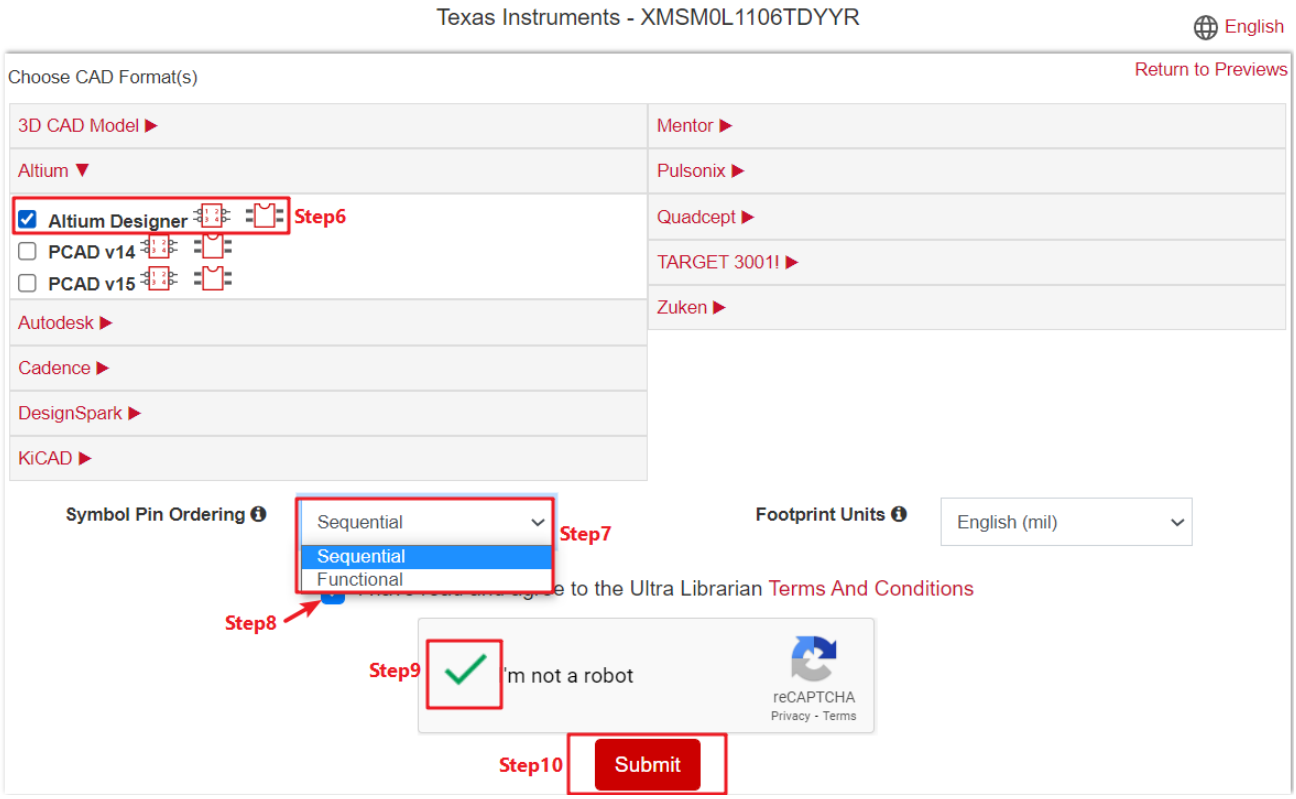

Figure A-13. Ultra Librarian Tool CAD Download

4. Run Altium Designer script as shown in Figure A-14.

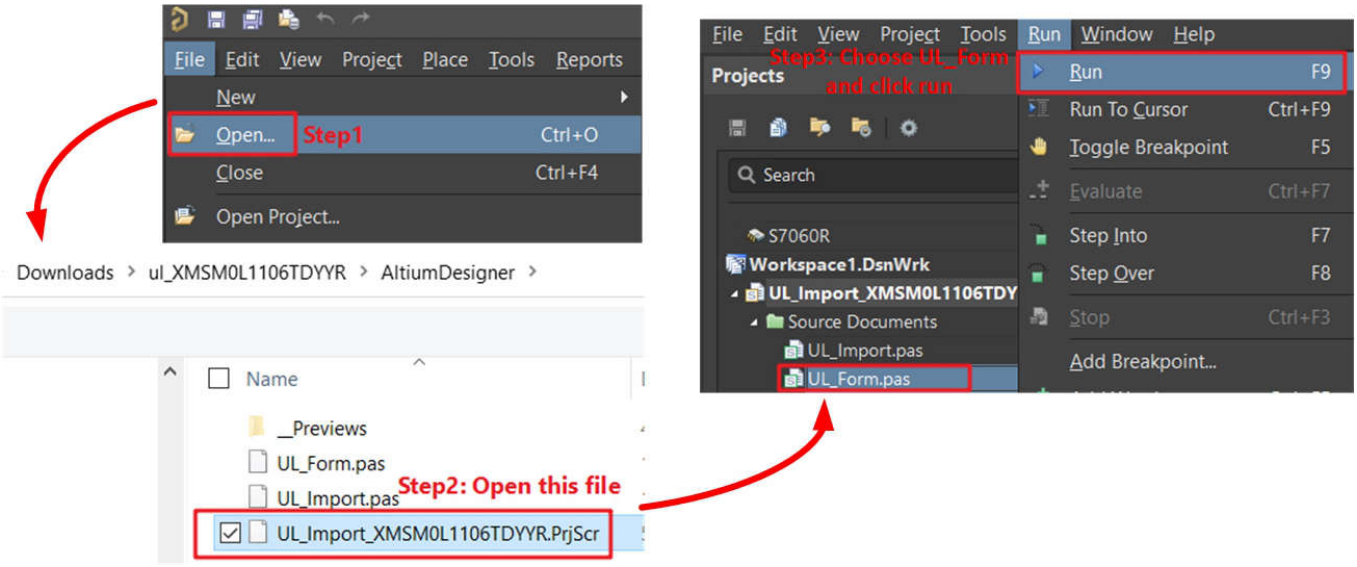

Figure A-14. Run Altium Designer Script

5. Generate PCB library and schematic library as shown in Figure A-15.

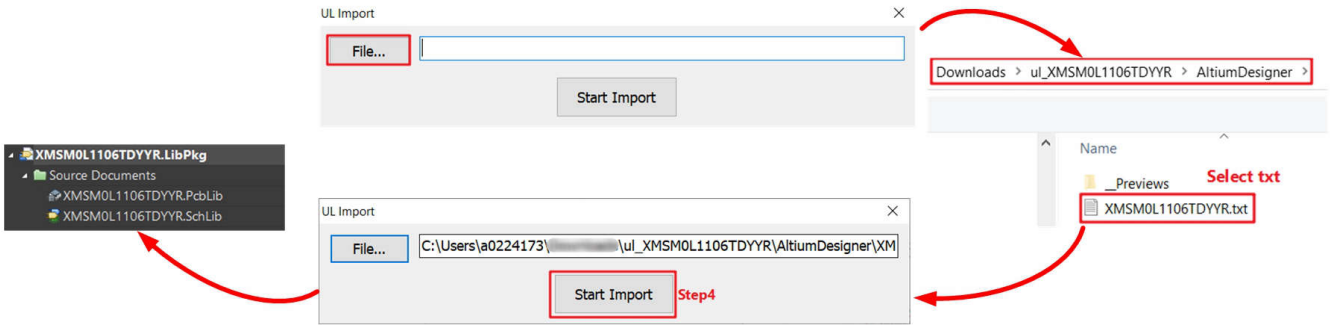

Figure A-15. Generate Library

6. Choose suitable footprint under PCB Library.

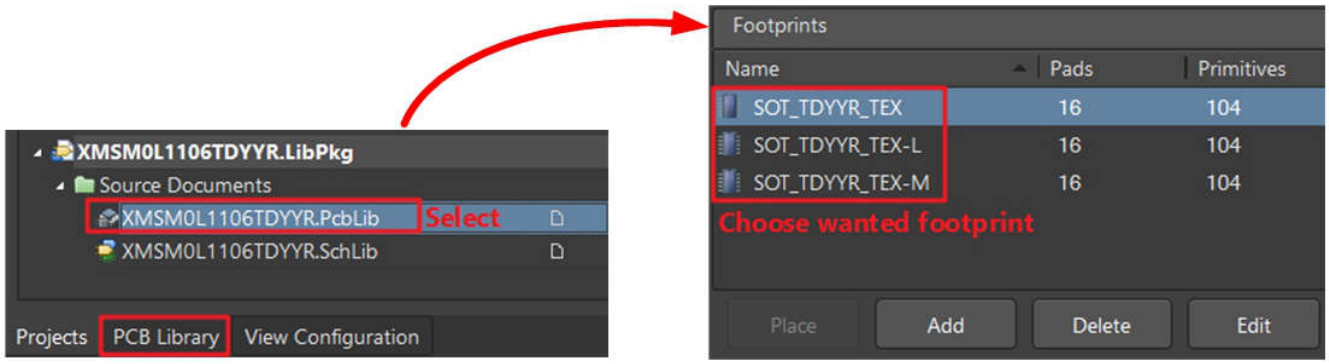

Figure A-16. Select Footprint

7. Import PCB library and schematic library.

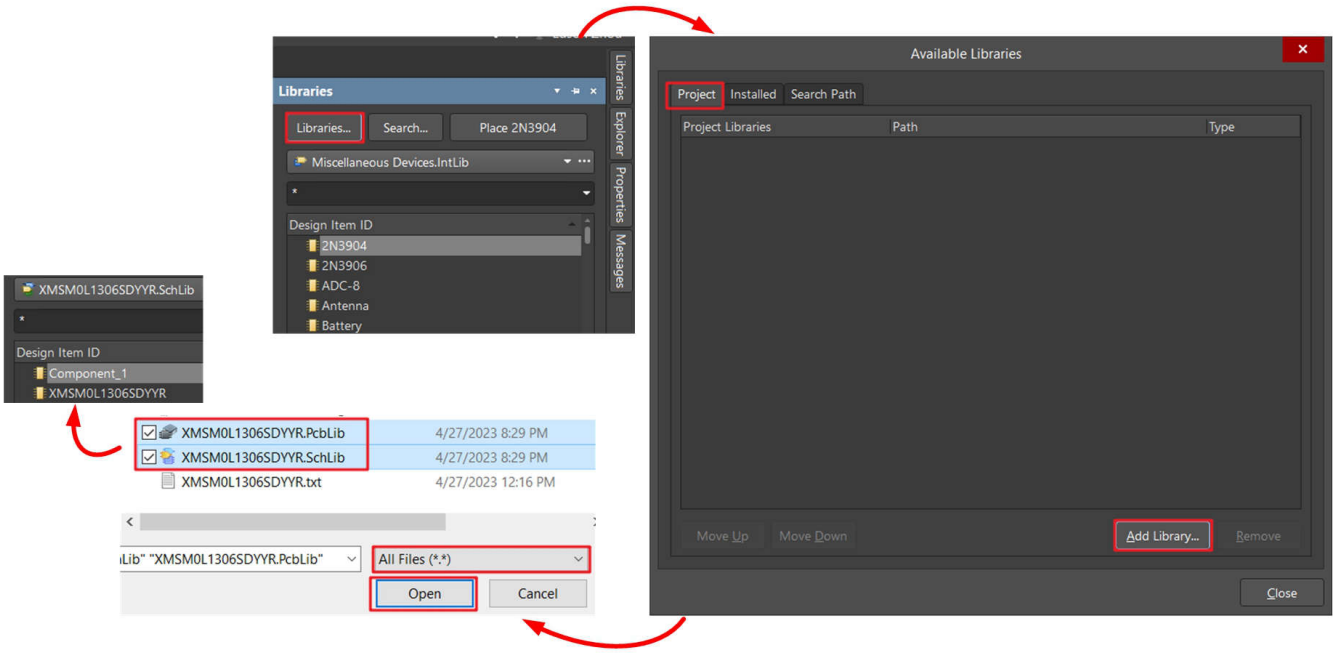

Figure A-17. Import Library

A.3 MSP-GANG Quick Introduction

This section shows how to use MSP-GANG to the offline program MSPM0. The beginning shows how to use MSP-GANG with its GUI to program a MSPM0.

1. Finish the pin connection used for SW, as shown in Figure A-18.

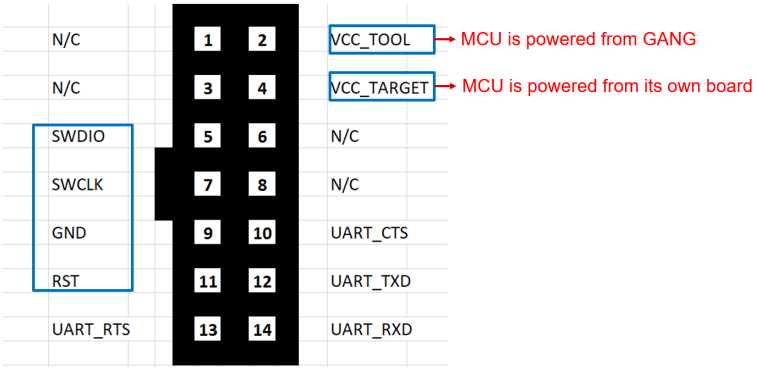

Figure A-18. MSP-GANG Pin Assignment

2. After you finish the hardware setup, you can follow the programming steps. For Step 2, see Section 5.1 to generate the code file. For Step 4, the enabled target is related to the hardware port used, which is signed in number near the port.

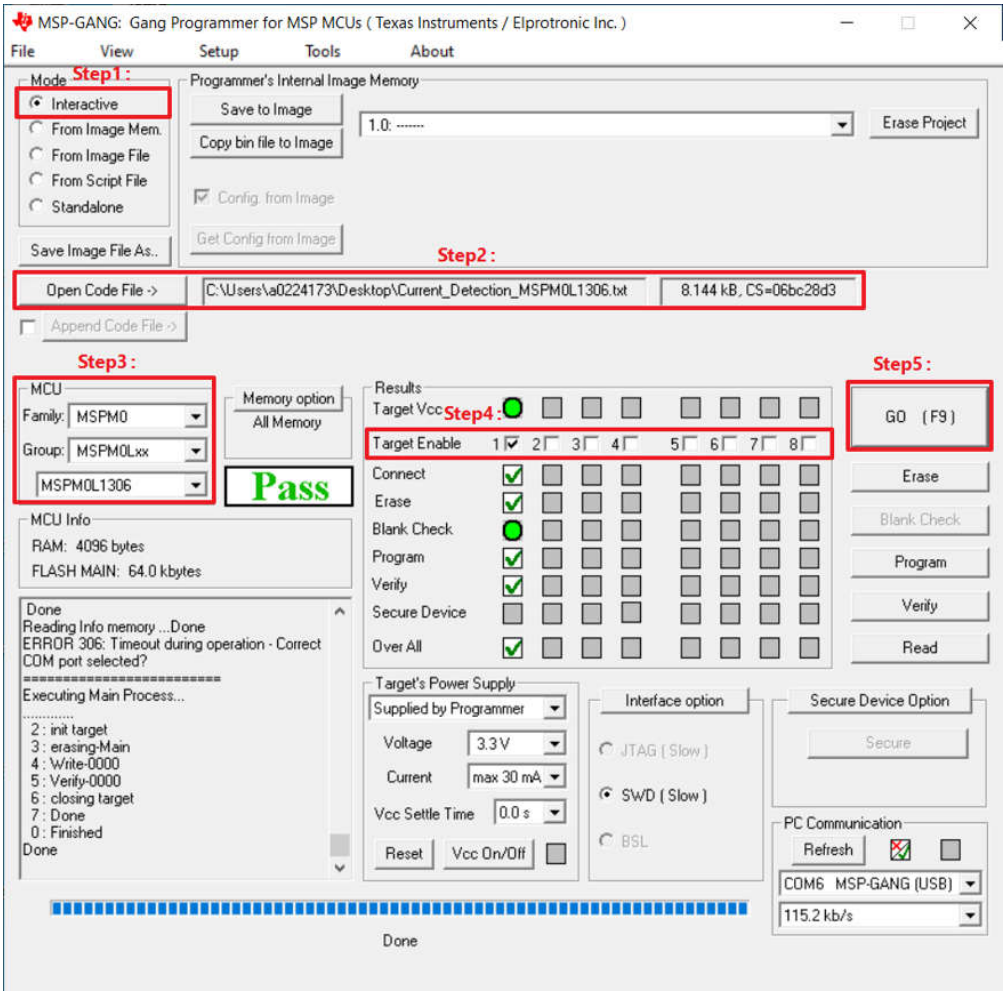

Figure A-19. Download Code Using MSP-GANG With GUI

3. If you need to change the code file in the non-main (SWD and BSL configure flash area), you need to enable this function first.

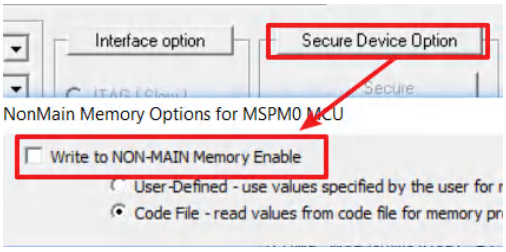

Figure A-20. Enable Non-main Programming

4. Save the code file and settings into MSP-GANG. You can give a project name to this image. Then click "Save to Image" as shown in Figure A-21.

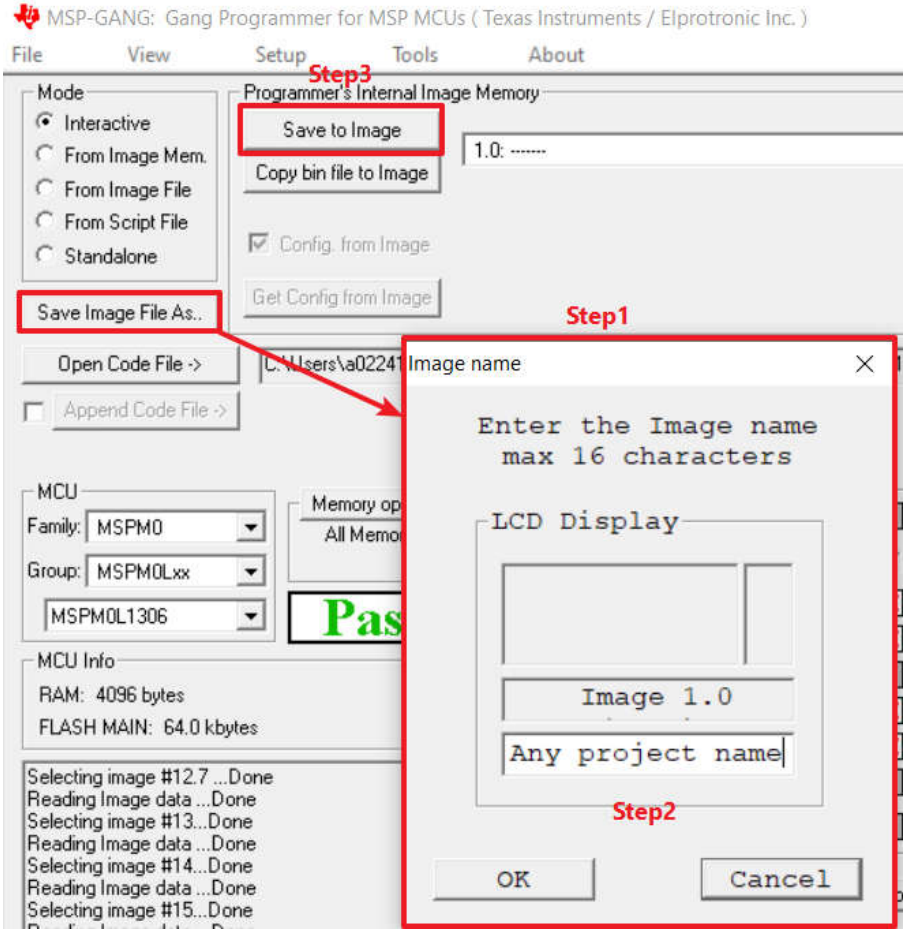

Figure A-21. Generate and Save Image

5. You can change the mode to standalone or directly close the GUI.

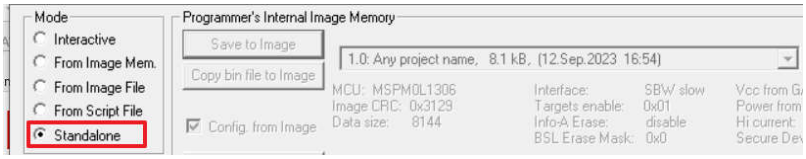

Figure A-22. Change Mode

6. If you only save one image in the MSP-GANG, you can click "Go" to do the programming. If you save more than one image, you need to switch to the correct image first.

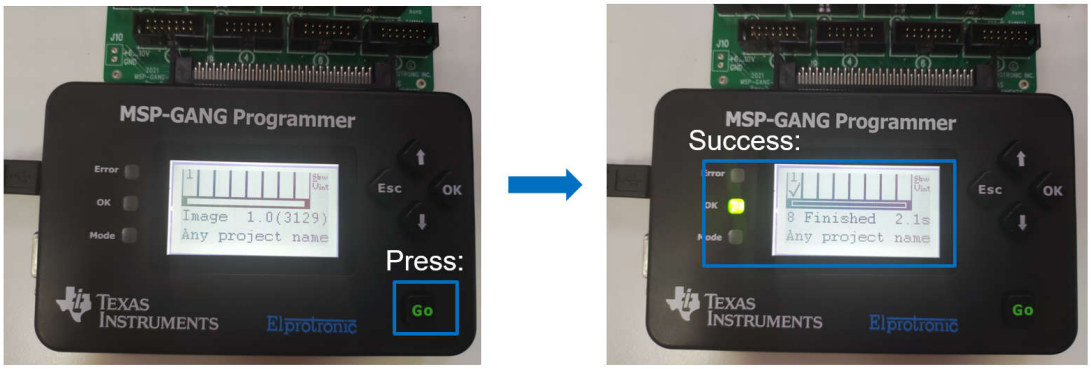

Figure A-23. Offline Porgramming

Revision History

NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

Changes from Revision A (June 2023) to Revision B (September 2023)	Page
• Updated the numbering format for tables, figures and cross-references throughout the document	3
• Changed Section 3.2.1	7
• Changed Section 3.3.2	10
• Changed Section 5.2	14
• Added Section A.3	23

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265