

Visual Studio Code for ModusToolbox™ user guide

ModusToolbox[™] tools package version 3.1.0

About this document

Scope and purpose

This document provides information and instructions for using Visual Studio Code (VS Code) with ModusToolbox[™] software.

ModusToolbox[™] software is a set of tools and libraries that support device configuration and application development. These tools enable you to integrate our devices into your existing development methodology.

Document conventions

Convention	Explanation				
Bold	Emphasizes heading levels, column headings, menus and sub-menus				
Italics	Denotes file names and paths.				
Courier New	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets				
File > New	Indicates that a cascading sub-menu opens when you select a menu item				

Reference documents

Refer to the following documents for more information as needed:

- <u>ModusToolbox[™] tools package installation guide</u> –Provides information and instructions about installing the tools package on Windows, Linux, and macOS.
- <u>ModusToolbox[™] tools package user guide</u> –Provides information about all the tools included with ModusToolbox[™] tools package.
- <u>Debugging in Visual Studio Code</u>
- <u>GitHub Marus/cortex-debug: Visual Studio Code extension for enhancing debug capabilities for Cortex-M</u>
 <u>Microcontrollers</u>



Table of contents

Table of contents

Abo	out this document	
Tab	le of contents	2
1	Download/install software	
1.1	ModusToolbox™ tools package	3
1.2	VS Code	3
1.3	J-Link	3
2	Getting Started	4
2.1	Create new application	4
2.2	Export existing application	
2.3	Open workspace in VS Code	8
3	Add/modify application code	10
4	Using ModusToolbox™ tools	11
4.1	ModusToolbox™ Assistant extension	
4.2	Command line	
5	Build the Application	13
6	Program/debug using KitProg3/MiniProg4	14
6.1	Connect the Kit	
6.2	Program	
6.3	Debug	
7	Program/debug using J-Link	16
7.1	Configure J-Link programmer/debugger settings	
7.2	Connect the Kit	
7.3	Program	
7.4	Debug	
8	Multi-core debugging	19
Revi	ision history	21



Download/install software

1 Download/install software

1.1 ModusToolbox[™] tools package

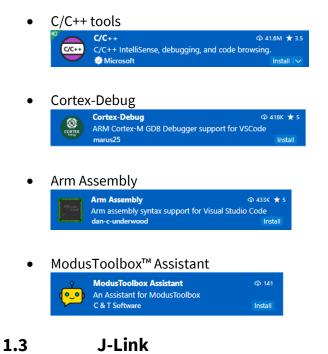
Refer to the instructions in the <u>ModusToolbox[™] tools package installation guide</u> for how to download and install the ModusToolbox[™] tools package.

1.2 VS Code

The ModusToolbox[™] tools package includes various tools to create and manage applications, but it does **not** include VS Code. If you do not already have VS Code installed on your computer, you can download it from the website:

https://code.visualstudio.com/

After opening an application in VS Code, it will recommend several extensions. The C/C++ tools and Cortex-Debug extensions are required for build and debug. Other extensions such as the ModusToolbox Assistant and Arm Assembly improve the development and debug experience.



For J-Link debugging, download and install J-Link software:

https://www.segger.com/downloads/J-Link



2 Getting Started

This section covers the ways to get started using VS Code with ModusToolbox™ software

- Create new application
- Exporting existing application
- Open workspace in VS Code

2.1 Create new application

Creating an application includes several steps, as follows:

2.1.1 Step 1: Open Project Creator tool

The ModusToolbox[™] Project Creator tool is used to create applications based on code examples and template applications. The tool is provided in GUI form and as a command line interface. For more details, refer to the <u>Project Creator user guide</u>. By default, the tool is installed in the following directory:

<user_home>/ModusToolbox/tools_<version>/project-creator

Open the Project Creator the tool as applicable for your operating system. You can launch it from the ModusToolbox[™] Dashboard, and you can launch it from the VS Code ModusToolbox[™] Assistant extension.

Choose Board Support Package (BSP) - Project Creator 2.10	-		Х
<u>Settings</u> <u>H</u> elp			
Source Template			
Enter filter text			
Kit Name MCU/SOC/SIP Connectivity > AIROC™ Bluetooth® BSPs > > AIROC™ Connectivity BSPs > > PMG BSPs > > PSoC™ 4 BSPs > > PSoC™ 6 BSPs > > TRAVEO™ BSPs > > Wireless Charging BSPs > > XMC™ BSPs >			
Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-mw-manifest/raw/v2.X/mtb-mw-manifest.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-mw-manifest/raw/v2.X/mtb-mw-manifest-fv2.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-th-mw-manifest/raw/v2.X/mtb-th-mw-manifest.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-tile mw-manifest/raw/v2.X/mtb-wifi-mw-manifest.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-wifi-mw-manifest/raw/v2.X/mtb-wifi-mw-manifest-fv2.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-wifi-mw-manifest/raw/v2.X/mtb-wifi-mw-manifest-fv2.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-wifi-mw-manifest/raw/v2.X/mtb-wifi-mw-manifest-fv2.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-wifi-mw-manifest/raw/v2.X/mtb-wifi-mw-dependencies-manifest.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-wifi-mw-manifest/raw/v2.X/mtb-wifi-mw-dependencies-manifest.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-staging/mtb-wifi-mw-manifest/raw/v2.X/mtb-wifi-mw-dependencies-manifest.xml' Finished download of file 'https://gitlab.intra.infineon.com/repo-stag	tt >	Clo	 ✓ se



2.1.2 Step 2: Choose Board Support Package (BSP)

When the Project Creator tool opens, expand one of the BSP categories under **Kit Name** and select an appropriate kit; see the description for it on the right. For this example, select the **CY8CKIT-062S2-43012** kit. The following image is an example; the precise list of boards available in this version will reflect the platforms available for development.

Kit Name MCU/SOC/SIP Conner > AIROC [™] Bluetooth ® BSPs The CY8CKIT-062S2-43012 PSoC [™] 652 Wi-Fi BT Pioneer Kit is a enables design and debug of PSoC [™] 6 MCUs. It comes with a Wi-Fi+ Bluetooth Combo Chip), industry-leading CAPSENSE [™] on-board debugger/programmer with KitProg3, microSD card NOR flash, PDM-PCM microphone interface. > PSoC [™] 6 BSPs Sterling CY8CEVAL-062S2-LAI-4373M2 CY8C624ABZI-S2D44 CY8CEVAL-062S2-LAI-4373M2 CY8C624ABZI-S2D44 CY8CEVAL-062S2-LAI-4373M2 CY8C624ABZI-S2D44 CY8CEVAL-062S2-LAI-4373M2 CY8C624ABZI-S2D44 CY8CKIT-062S2-MIR-43439M2 CY8C624ABZI-S2D44 CY8CKIT-062S2-43012 CY8C624ABZI-S2D44 CY8CKIT-062S2-43012 CY8C624ABZI-S2D44 CY8CKIT-062S2-43012 CY8C624ABZI-S2D44 CY8CKIT-062S2-43012 CY8C624ABZI-S2D44 CY8CKIT-062S40 CY8ES CY8CKIT-062S40 CY8CS CY8CKIT-064S0S2-4343W CY80644ABZI-S2D44 CY8CKIT-064S0S2-4343W CY80644ABZI-S2D44 CY8CKIT-064S0S2-4343W CY80644ABZI-S2D44 CY8CKIT-064S0S2-4343W CY80644ABZI-S2D44 CY8CKIT-064S0S2-4343W CY80644ABZI-S2D44 CY8CKIT-064S0S2-4343W CY80644ABZI-S2D44 CY8CK	Murata 1LV Module (CYW430 ' for touch buttons and slider
CY8CPR0T0-062-4343W CY8C624ABZI-S2D44 LBEE5K CY8CPR0T0-062-33-4343W CY8C6245LQI-S3D72 LBEE5K CY8CPR0T0-0632S3-4343W CY8C6245LQI-S3D72 LBEE5K CY8CPR0T0-0648DS1-BLE CYBLC+Id045-02 (CY8C6347BZI-BLD53) CYBLE-Id045-02 (CY8C6347BZI-BLD53) CYBLE-Id045-02 (CY8C6347BZI-BLD53) CY8CPR0T0-0648DS1-BLE CYB06447BZI-BLD53 <none:< td=""> CY8CPR0T0-0648DS3 CYB06447BZI-BLD54 <none:< td=""> CY8CPR0T0-0648DS1-SB CYB06447BZI-D54 <none:< td=""> CY8LE+416045-FVAL CYBLE4-Id045-02 (CY8C6347BZI-BLD53) CYW43 CY8DSYSKIT-01 CY8C624AFNI-S2D43 CYW43 CYW9P62S1-43012EVB-01 WM-BAC-CYW-50 (CY8C6247FDI-D52) WM-BL4 CYW9P62S1-43438EVB-01 AW-CU4Z (CY8C6247FDI-D52) AW-CU4 KIT-BGT60TR13C-EMBEDD CY8C6247FDI-D02 <none:< td=""> PSOC6-GENERIC CY8C6347BZI-BLD53 <none:< td=""></none:<></none:<></none:<></none:<></none:<>	the primary application ndary processor for low-pow SENSE, a PDM-PCM digital ommunication blocks, 7



2.1.3 Step 3: Select application

Click **Next >** to open the Select Application page.

Select Application - Pr	roject Creator 2.10 -	
<u>S</u> ettings <u>H</u> elp		
Application(s) Root Path:	C:/Users/Test/mtw3.1/vscode	Browse
Target IDE:	<none> ~</none>	
Enter filter text	Browse Vew Application Name New BSP Name	
 Pluetooth® Connectivity Getting Started Graphics Machine Learning Manufacturing Peripherals Sensing Voice Wi-Fi 		
Select one or more temp	late applications to proceed with the new project creation process.	
•	< <u>B</u> ack <u>C</u> reate	<u>C</u> lose

This page displays example applications, which demonstrate different features available on the selected BSP. In this case, the CY8CKIT-062S2-43012 provides the PSoC[™] 62 MCU and the AIROC[™] CYW43012 Wi-Fi & Bluetooth[®] combo chip. You can create examples for PSoC[™] 6 MCU resources such as CAPSENSE[™] and QSPI, as well as numerous examples for other capabilities.

Click **Browse...** next to **Application(s) Root Path** to create or specify a folder where the application will be created.

Pull down the Target IDE menu, and select Microsoft Visual Studio Code.

Application(s) Root Path:	C:/Users/Test/mtw3.1/vscode	Browse
Target IDE:	<none> ~</none>	
	<none></none>	
Enter filter text	Eclipse IDE for ModusToolbox™	
	Microsoft Visual Studio Code	
lemplate Application	IAR Embedded Workbench	
> Bluetooth®	ARM MDK (uVision)	



Under the **Template Application** column, expand **Getting Started** and select **Hello World** from the list. This example exercises the PSoC[™] 6 MCU to blink an LED.

Template Application	New Application Name
> Bluetooth®	
 Getting Started 	
Dual-CPU Empty PSoC6 App	
Empty App	
Hello World	Hello_World
Switching Power Modes	
> Graphics	
> Machine Learning	
> Manufacturing	
> Peripherals	
> Sensing	
> Wi-Fi	

Note: The actual application names available might vary.

Type a name for your application or leave the default name. Do not use spaces in the application name. Also, do not use common illegal characters, such as:

* . " ` / \ [] : ; | = ,

2.1.4 Step 4: Create application

Click **Create** to start creating the application. The tool displays various messages.

Select Application - P	roject Creator 2.10				_	
<u>S</u> ettings <u>H</u> elp						
Application(s) Root Path:	C:/Users/Test/mtw3.1/vscode					Browse
Target IDE:	Microsoft Visual Studio Code				~	
Search		Danuara 🖙 🕬		This and a second advanced at the size of a UA	The second section for a single second	
		Browse 💎 🚰		This code example demonstrates simple UA World" message on a terminal and blinks an		a Hello
Template Application Bluetooth® Getting Started 	New Application	on Name		For more details, see the <u>README on GitHut</u>	2.	
Dual-CPU Em Empty App	pty PSoC6 App					
Hello World	Hello_World					
Switching Por Graphics	wer Modes					
 Machine Learning Manufacturing 						
> Peripherals						
> Sensing > Wi-Fi						
	esssemiconductorco_core-make					
Starting: git -C C:/Users/	Test/mtw3.1/vscode/mtb_shared,	/mtb-hal-cat1 cloneorigin		eckout C:/Users/follettcj/.modustoolbox/cach	e/git/	^
Starting: git -C C:/Users/		/mtb-pdl-cat1 cloneorigin	cypressno-ch	eckout C:/Users/follettcj/.modustoolbox/cach	e/git/	
httpsgithub.com_cyp Cloning into 'TARGET_C\	resssemiconductorco_mtb-pdl-ca /8CKIT-062S2-43012'	at1/mtb-pdl-cat1 release-v3.2	2.0			~
					< <u>B</u> ack <u>C</u> reate	<u>C</u> lose
•						



When the process completes, a message states that the application was created. Click **Close** to exit the Project Creator tool.

0 error(s), 0 warning(s)			^
Summary:			
Successfully created and exported "Hello_World" application.			~
	< <u>B</u> ack	<u>C</u> reate	<u>C</u> lose

2.2 Export existing application

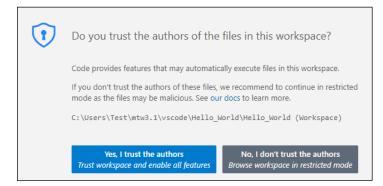
If you have a ModusToolbox[™] application that was created for another IDE or for the command line, you can export that application to be used in VS Code. Open a terminal window in the application directory, and run the command make vscode.

2.3 Open workspace in VS Code

In VS Code, select **File > Open Workspace from File**, navigate to the location of the application that was just created, select the workspace file, and click **Open**.

🗙 Open Workspace from File			×
\leftarrow \rightarrow \checkmark \uparrow \square \lt Users \flat	Test \rightarrow mtw3.1 \rightarrow vscode \rightarrow Hello_World \rightarrow	✓ Ö	earch Hello_World
Organize 🔻 New folder			BII - II ?
💿 My Desktop	^ Name	Date modified	Type Size
💿 My Documents	.vscode	1/23/2023 12:50 PM	File folder
wy bocuments	bsps	1/23/2023 12:49 PM	File folder
💿 My Sync	📙 build	1/23/2023 12:50 PM	File folder
	deps	1/23/2023 12:49 PM	File folder
💻 This PC	images	1/23/2023 12:49 PM	File folder
3D Objects	libs	1/23/2023 12:49 PM	File folder
E Desktop	Hello_World.code-workspace	1/23/2023 12:50 PM	Code Workspace
Documents			
🖊 Downloads			
J Music			
Pictures			
🚆 Videos			
🚔 Windows (C:)			
A	~ <		>
File <u>n</u> ame:	Hello_World.code-workspace	~ Code	Workspace (*.code-works ~
			pen Cancel

Depending on your settings in VS Code, you may see a message about trusting the authors. If so, click **Yes**, **I trust the authors**.





VS Code opens with the Hello_World workspace in the EXPLORER view.

×	<u>F</u> ile <u>E</u>	dit <u>S</u> election	<u>V</u> iew <u>G</u> o	<u>R</u> un	<u>T</u> erminal	<u>H</u> elp	$\leftarrow \rightarrow$	۹) Hello_World (Workspa	ice)	-		×
Ð	EXP	LORER											
	∼ HEI	LO_WORLD (WO	RKSPACE)										
Q	\sim I	lello_World											
Ĩ.	>	.vscode											
പ്പ		bsps											
0		build											
å		deps											
æ		images											
-0		libs											
₿		.cyignore											
		.gitignore											
		Hello_World.co	ode-workspa	ice									
		LICENSE											
		main.c											
		Makefile											
		openocd.tcl											
		README.md											
		ntb_shared											
		cat1cm0p						Sh	now All Commands	Ctrl + Shift + P			
		cmsis							Go to File	Ctri + P			
		core-lib											
		core-make mtb-hal-cat1							Find in Files	Ctrl + Shift + F			
									Start Debugging	F5			
		mtb-pdl-cat1 recipe-make-c	at1a						start bebugging	15			
		retarget-io	atia						Toggle Terminal	Ctri + `			
	1 1	retarget-to											
8													
<u>ና</u> ፖረ	> 00	TLINE											
572	> TIN	ELINE											
⊗ 0	∆ 0											ନ	C
							-				 		



Add/modify application code

3 Add/modify application code

Code example applications work as they are, and there is no need to add or modify code in order to build or program them. However, if you want to update and change the application to do something else, open the appropriate file in the code editor.

```
Double-click the main.c file to open it.
```

<u>File E</u> dit <u>S</u> election <u>V</u> iew <u>G</u> o <u>R</u> un <u>T</u> err	ninal <u>H</u> elp	$\leftarrow \rightarrow$	P Hello_World (Workspace)		-	
EXPLORER	C main.	c ×		3	⊳~	⊜ 🛛
✓ HELLO_WORLD (WORKSPACE) [¹ ₊ E ² ₊ ひ ④	Hello_W	orld > C m				
Y Hello_World	33		e or any product or circuit described in the Soft			
> .vscode	34		horize its products for use in any products where			3.522
> bsps	35		of the Cypress product may reasonably be expecte			
> build	36	* signifi	cant property damage, injury or death ("High Risk	Product"). By	R	Bierr
> deps	37		ng Cypress's product in a High Risk Product, the			
> images	38		system or application assumes all risk of such u	se and in doing		Trans.
> libs	39 40	* so agre	es to indemnify Cypress against all liability.			NATURAL AND
.cvignore	40	1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 - 1999 -	~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~		1	
♦ .gitignore	42	#include	"cvhal.h"		14	-
{} Hello World.code-workspace	43	<pre>#include</pre>			3	
1 LICENSE	44	<pre>#include</pre>	"cy_retarget_io.h"		1	NEP'S PER BARY
C main.c	45				1	
M Makefile	46				1	NAME OF A DESCRIPTION OF A
≡ openocd.tcl	47	·	***************************************	******		SALAR WAS
(i) README.md	48 49	* Macros	*********	*****		19489-94081
✓ mtb_shared	50	/* LED b]	ink timer clock value in Hz */			PROCESSION AND DRAWNING
> cat1cm0p	51		ED BLINK TIMER CLOCK HZ (10000)			An other states and the second
> cmsis	52					And and a second
> core-lib	53		ink timer period value */		1	
> core-make	54	#define L	ED_BLINK_TIMER_PERIOD (9999)		No. of Concession, Name	
> mtb-hal-cat1	55				1	62X 127
> mtb-pdl-cat1	56 57	/******	********	*****	1	BARANK 1550
> recipe-make-cat1a	58	/	n Prototypes			
> retarget-io	59	******	*****	*****		
	60	void time	r_init(void);		1	Pro resultation of the
)	61	static vo	<pre>id isr_timer(void *callback_arg, cyhal_timer_even</pre>	t_t event);		
) outling	62					
> OUTLINE	63	/*******	*********			
> TIMELINE	04	1				

As you type into the file, a dot will appear in the file's tab to indicate changes were made. The file icon will also indicate that there are unsaved changes.



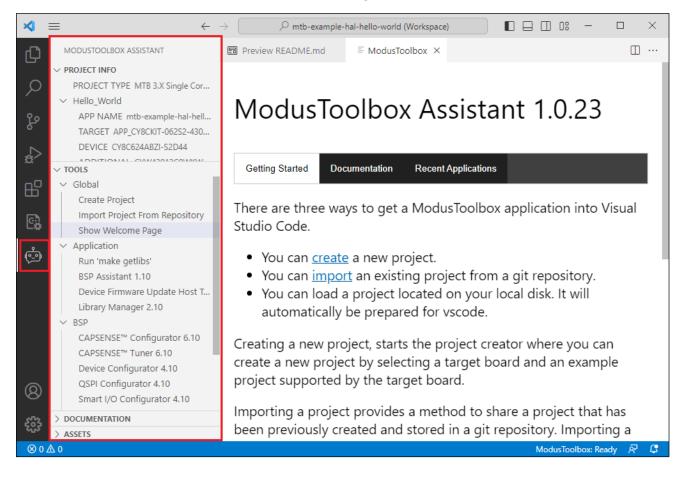


Using ModusToolbox[™] tools

4 Using ModusToolbox[™] tools

4.1 ModusToolbox[™] Assistant extension

The easiest way to open various ModusToolbox[™] tools with VS Code is by installing the ModusToolbox[™] Assistant extension, which provides access to tools, configurators, and documentation.



4.2 Command line

Alternatively, you can open various ModusToolbox[™] tools using make commands in the terminal. Select **Terminal > New Terminal**, then select the main project folder for your application (in this case, Hello_World):

View	iew Select current working directory for new terminal						
	Hello_World C:\Users\Test\mtw3.1\vscode						
KSPAC	mtb_shared C:\Users\Test\mtw3.1\vscode	J					

Note: On Windows, use the modus-shell (Cygwin) terminal.

This section covers a few of the tools you might open more frequently. For a complete list of the tools available, refer to the <u>ModusToolbox[™] tools package user guide</u>.



Using ModusToolbox[™] tools

4.2.1 Library Manager

To add, remove, or modify libraries, open the Library Manager using the following command:

make library-manager

Refer to the <u>ModusToolbox[™] Library Manager user guide</u> for details about that tool.

4.2.2 BSP Assistant

To create or modify a BSP, open the BSP Assistant using the following command:

make bsp-assistant

Refer to the <u>ModusToolbox[™] BSP Assistant user guide</u> for details about that tool.

4.2.3 Device Configrator

To view peripherals, pins, clocks, etc., open the Device Configurator using the following command:

make device-configurator

The Device Configurator provides access to the BSP resources and settings. Each enabled resource contains one or more links to the related API documentation. There are also buttons to open other configurators for CAPSENSE[™], QSPI, Smart I/O, etc. For more information, refer to the <u>Device Configurator user guide</u>, which is also available by selecting **View Help** from the tool's **Help** menu.

C:/Users/Test/mtw3.1/vscode/Hello_World/bsps/T/	ARGET_APP_CY8CKIT-0	62S2-43012/config/design.n	nodus* - Device Configurator 4.10	- 0	×
<u>File Edit View H</u> elp					
CY8C624ABZI-S2D44 LBEE59B1LV/CYW43012C0WF	WBG		Real Time Clock (RTC) - Parameters	6	FΧ
Peripherals Pins Analog-Routing System	Peripheral-Clocks	DMA	Enter filter text	🖉 💆 📃	•
Enter filter text		V 🖻 🖻 🖌 🗎 🛍	Name	Value	^
Resource	Name(s)	Personality ^	✓ Overview		
TCPWM[0] 32-bit Counter 1	tcpwm_0_cnt_1		(?) Configuration Help	Open RTC Documentatio	<u>n</u>
TCPWM[0] 32-bit Counter 2	tcpwm_0_cnt_2		✓ General		_
TCPWM[0] 32-bit Counter 3	tcpwm 0 cnt 3		⑦ Date Format		~
TCPWM[0] 32-bit Counter 4	tcpwm 0 cnt 4		? Enable Daylight Savings (DST)		
TCPWM[0] 32-bit Counter 5	tcpwm 0 cnt 5		✓ Time and Date		_
TCPWM[0] 32-bit Counter 6	tcpwm 0 cnt 6	_	? Seconds	0	-
TCPWM[0] 32-bit Counter 7	tcpwm_0_cnt_7	_	(?) Minutes	0	
 Timer, Counter, and PWM (TCPWM) 1 	cepwin_o_ene_r		? Hours Format	24H	~
✓ System			(?) Hour	12	_
CSD (CapSense, etc.) 0	CYBSP_CSD	CSD-3.0	⑦ Day of the Month	1	
LCD Direct Drive 0	lcd_0_drive_0		⑦ Month	January	~
Multi-Counter Watchdog Timer (MCWDT) 0	srss_0_mcwdt_0		? Year	0	
Multi-Counter Watchdog Timer (MCWDT) 1	srss_0_mcwdt_1		⑦ Day of the Week	SATURDAY	
Real Time Clock (RTC)	srss 0 rtc 0	Real Time Clock-3.0 🗸	✓ Advanced		
<		>	Config in Flash Real Time Clock (RTC) - Parameters	Code Preview	*
			Real time Clock (RIC) - Parameters		
Notice List				6	Ρ×
😢 0 Errors 🔥 0 Warnings 📔 0 Tasks 🚺 4	Infos				
Fix Description			Location		^
The WCO is enabled. Chip startup will be slowe WCO is ready. See the device datasheet for WC consider starting it in main() for faster chip star	O startup timing. If WC				~
Ready					:

Note:

The Device Configurator cannot be used to open Library Configurators, such as Bluetooth®.



Build the Application

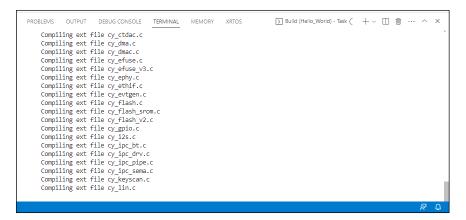
5 Build the Application

Building the application is not specifically required, because building will be performed as part of the programming and debugging process. However, if you are running VS Code without any hardware attached, you may wish to build your application to ensure all the code is correct.

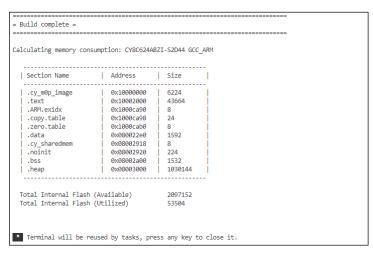
Select Terminal > Run Task. Then select Build Hello_World.

🗙 File Edit Selection View Go Run	Terminal Help	$\leftarrow \rightarrow$	Select the task to run	
	New Terminal Split Terminal	Ctrl+Shift+` Ctrl+Shift+5	Build Hello_World	recently used -🛱 🐯 🗙
✓ HELLO_WORLD (WORKSPACE)			Clean Hello_World	
> Hello_World	Run Task		Rebuild Hello_World	configured
> mtb_shared	Run Build Task	Ctrl+Shift+B	Tool: Library Manager Hello_World	
\$ ↓>	Run Active File Run Selected Text	Ctri+Shift+B	은 grunt 은 gulp 은 jake	contributed
en Ber	Show Running Tasks Restart Running Task Terminate Task		En npm En typescript En cppbuild Show All Tasks	
	Configure Tasks Configure Default Build	Task		

Build information will display in the Terminal.



The build should complete successfully with messages similar to the following:



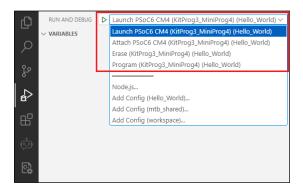


Program/debug using KitProg3/MiniProg4

6

Program/debug using KitProg3/MiniProg4

Most PSoC[™]-based kits use KitProg3/MiniProg4 as the default programmer/debugger. The VS Code GUI shows these configurations by default:



- **Launch:** This builds the entire application on both cores, programs all the device's memories, and then starts a Cortex-M4 debugging session.
- Attach: This starts a Cortex-M4 debugging session attaching to a running PSoC[™] 6 target without programming or reset.
- Erase: This erases all internal memories.
- **Program:** This builds the entire application on both cores, programs all the device's memories, and then runs the program.

6.1 Connect the Kit

Follow the instructions provided with the kit to connect it to the computer with the USB cable.

6.2 Changing programming interface SWD/JTAG

To change the target interface for KitProg3_MiniProg4, edit the *openocd.tcl* file in the project root directory. This file contains the OpenOCD command allowing you to select the debugging interface: "transport select". Set this to either swd or jtag.

EXPLORER	≡ openocd.tcl ×
〜 MTB-EX 📭 📴 ひ 🗿	Hello_World > 🗧 openocd.tcl
 > deps > images > libs ⇒ .cproject ≡ .cyignore ◆ .gitignore ≅ .project € LUCENSE C main.c Makefile (} mtb-example-hal-he 	<pre>source [find interface/kitprog3.cfg] transport select swd source [find target/psoc6_2m.cfg] \$ source [find target/psoc6_2m.cfg] \$ \${TARGET}.cm4 configure -rtos auto -rtos-wipe-on-reset-halt 1 psoc6 sflash_restrictions 1 8 </pre>
■ openocd.tcl	
(i) README.md	
> mtb_shared	

6.3 Program

Select the **Run And Debug** icon in the VS Code Activity Bar, select the **Program (KitProg3_MiniProg4)** Launch Configuration, and click **Start Debugging** icon or press **F5**.





Program/debug using KitProg3/MiniProg4

If needed, VS Code builds the application and messages display in the Terminal. If the build is successful, device programming starts immediately. If there are build errors, then error messages will indicate as such. When programming completes successfully, the LED will start blinking.

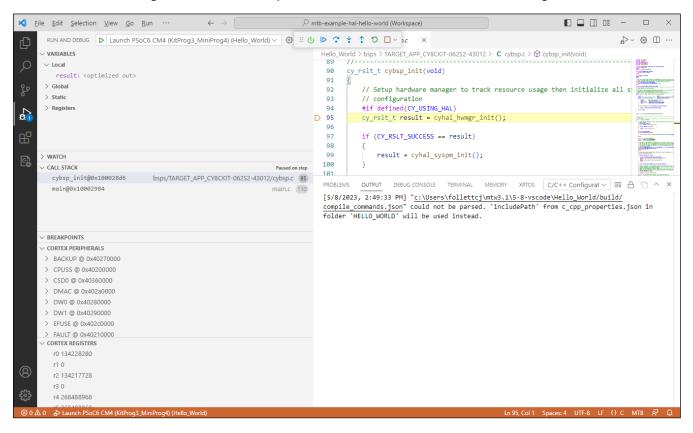


6.4 Debug

Select the **Run And Debug** icon in the VS Code **Activity Bar**, select the **Launch PSoC6 CM4** (KitProg3_Miniprog4) Launch Configuration, and click **Start Debugging** icon or press F5.

RUN AND DEBUG Launch PSoC6 CM4 (KitProg3_MiniProg4) (Hello_World) ~

If needed, VS Code builds the application and messages display in the Console. If the build is successful, VS Code switches to debug mode automatically. If there are build errors, then error messages will indicate as such.





Program/debug using J-Link

7 Program/debug using J-Link

Most PSoC[™]-based BSPs default to using the KitProg3/MiniProg4 programmer/debugger launch configurations. This section covers how to use J-Link.

7.1 Configure J-Link programmer/debugger settings

1. Open your ModusToolbox[™] single-core application's *Makefile*, or multi-core application's *common.mk* file, and enter the following variable:

BSP_PROGRAM_INTERFACE=JLink

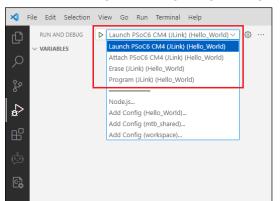
2. If you install J-Link software in a non-default location, or if you intend to use make commands, such as make program and make qprogram, also enter the following variable:

MTB_JLINK_DIR=<path to J-Link>

- *Note:* If you want all projects/applications for the BSP to use the J-Link programmer/debugger, you can specify the BSP_PROGRAM_INTERFACE and MTB_JLINK_DIR variables in the bsp.mk file instead.
- 3. Save the Makefile.
- 4. In a bash Terminal run:

make vscode

When the command completes, J-Link configurations will be shown. These are the same configurations described in <u>Program/debug using KitProg3/MiniProg4</u>, but applicable to J-Link.



5. Open the *settings.json* file and *<app>.code-workpace* file to verify the path to the J-Link GDB server.

For example, the default on Windows is:

"cortex-debug.JLinkGDBServerPath": "C:/Program Files/SEGGER/JLink/JLinkGDBServerCL.exe"

Hello Wo	vrld > .vscode > {} settings.json >
12	//mtD//
13	<pre>"modustoolbox.toolsPath": "C:/Users/follettcj/ModusToolbox/tools_3.1",</pre>
14	<pre>"cortex-debug.armToolchainPath": "\${config:modustoolbox.toolsPath}/gcc/bin",</pre>
15	<pre>"cortex-debug.openocdPath": "\${config:modustoolbox.toolsPath}/openocd/bin/openocd.exe",</pre>
16	"cortex-debug.JLinkGDBServerPath.windows": "C:/Program Files/SEGGER/JLink/JLinkGDBServerCL.exe",
17	"cortex-debug.JLinkGDBServerPath.osx": "/Applications/SEGGER/JLink/JLinkGDBServerCLExe",
18	"cortex-debug.JLinkGDBServerPath.linux": "JLinkGDBServerCLExe"
19	
{} mtb-e	w xample-bal-bello-world-code-workspace
	zi xample-hal-hello-world.code-workspace × vrld > {} mtb-example-hal-hello-world.code-workspace >
Hello_Wo	rld > {} mtb-example-hal-hello-world.code-workspace >
Hello_Wo	<pre>rld > () mtb-example-hal-hello-world.code-workspace > "modustoolbox.toolsPath": "C:/Users/follettcj/ModusToolbox/tools_3.1",</pre>
Hello_Wo 21 22	<pre>wid> () mtb-example-hal-hello-word.code-workspace> "modustoolbox.toolsPath": "C:/Users/follettcj/ModusToolbox/tools_3.1", "cortex-debug.armToolchainPath": "\${config:modustoolbox.toolsPath}/gcc/bin", "cortex-debug.openocdPath": "\${config:modustoolbox.toolsPath}/openocd/bin/openocd.exe",</pre>
Hello_Wo 21 22 23	<pre>whd > () mtb-example-hal-hello-workspace > "modustoolbox.toolsPath": "C:/Users/follettcj/ModusToolbox/tools_3.1", "cortex-debug.armToolchainPath": "\${config:modustoolbox.toolsPath}/gcc/bin",</pre>
Hello_Wo 21 22 23 24	<pre>vid > () mtb-example-hal-hello-word.code-workspace > "modustoolbox.toolsPath": "C:/Users/follettcj/ModusToolbox/tools_3.1", "cortex-debug.armToolchainPath": "\${config:modustoolbox.toolsPath}/gcc/bin", "cortex-debug.openocdPath": "\${config:modustoolbox.toolsPath}/openocd/bin/openocd.exe", "cortex-debug.JLinkGDBServerPath.windows": "C:/Program Files/SEGGER/JLink/JLinkGDBServerCL.exe</pre>



Program/debug using J-Link

7.2 Connect the Kit

Follow the instructions provided with the kit and from SEGGER to connect it to the computer with the J-Link probe.

7.3 Changing programming interface SWD/JTAG

To change the target interface for J-Link, edit the *launch.json* file to specify the applicable "interface": swd or jtag. Do this for all the applicable configurations (e.g., Launch, Attach, Erase, etc.)

EXPLORER ····	{} launch	json ×	
〜 MTB-EX 📭 📴 ひ 🗿	Hello_Wo	rld > .vsc	ode > {} launch.json > [] configurations > {} 0
✓ Hello_World	32	"ve	rsion": "0.2.0".
> .mtbLaunchConfigs	33		nfigurations": [
> .settings	34		{
∨ .vscode	35		"name": "Launch PSoC6 CM4 (JLink)",
<pre>{} c_cpp_properties.js</pre>	36		"type": "cortex-debug",
{} extensions.ison	37		"request": "launch",
 Iaunch.json 	38		<pre>"cwd": "\${workspaceFolder}",</pre>
-	39		"executable": "./build/APP_CY8CKIT-062S2-43012/Debug/mtb-example-hal-h
{} settings.json	40		"servertype": "jlink",
{} tasks.json	41		<pre>device": "CY8C6xxA CM4_sect256KB",</pre>
> bsps	42		"interface": "swd",
> build	43		"preLaunchCommands": [
> deps	44		// Program via the hex file to get complete coverage
	45		"exec-file ./build/APP_CY8CKIT-06252-43012/Debug/mtb-example-hal-
> images	46],

7.4 Program

Select the **Run And Debug** icon in the VS Code Activity Bar, select the **Program (JLink)** Launch Configuration, and click **Start Debugging** icon or press **F5**.



If needed, VS Code builds the application and messages display in the Terminal. If the build is successful, device programming starts immediately. If there are build errors, then error messages will indicate as such. When programming completes successfully, the LED will start blinking.



7.5 Debug

Click the **Run and Debug** icon, select **Launch PSoC6 CM4 (JLink)** config, and click **Start Debugging** icon or press **F5**.





Program/debug using J-Link

<u>File Edit Selection View Go</u> \cdots \leftarrow \rightarrow	🔎 mtb-example-hal-helio-world (Workspace)			
RUN AND DEBUG 🕨 Launch PSoC6 CM4 (/Link) (He 🗸 🕲 …		© □ ·		
∨ VARIABLES	Hello_World > bsps > TARGET_APP_CY8CKIT-062S2-43012 > ℃ cybsp.c > ۞ cybsp_init(void)			
) V Local	87 //	Manager and the second second		
result: <optimized out=""></optimized>	88 // cybsp_init	INSIGN OF ANY		
> Global	89 //			
> Static	01 5	INTERACTORY INCOME		
> Registers	91 { 92 // Setup hardware manager to track resource usage then initialize all			
> WATCH	93 // configuration	Bree Martineer		
CALL STACK Paused on step	94 #if defined(CY_USING_HAL) D 95 cy rslt t result = cyhal hwmgr init();	100 1 1 00 00 00 00 00 00 00 00 00 00 00		
cybsp_init@0x100028d6 bsps/TARGET_APP_CY8CKIT-062	96			
main@0x10002984 main.c (110	<pre>97</pre>			
> BREAKPOINTS	102 #ifdef CY_CFG_PWR_VDDA_MV	Thursday was		
V CORTEX PERIPHERALS	103 if (CY_RSLT_SUCCESS == result)			
> BACKUP @ 0x40270000 > CPUSS @ 0x40200000	PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL MEMORY XRTOS Filter (e.g. text, !exclude)	≣× ^		
> CPUSS @ 0x40200000 > CSD0 @ 0x40360000	A program is being debugged already.			
	Are you sure you want to change the file?			
> DMAC @ 0x402a0000	(y or n) [answered Y; input not from terminal]			
> DW0 @ 0x40280000	2 Resetting target			
V CORTEX REGISTERS				
r0 134228280	Temporary breakpoint 1, main () at main.c:97			
r10	9/ { >			
0 ▲ 0 B Launch PSoC6 CM4 (JLink) (Hello_World)	Ln 95, Col 1 Spaces: 4 UTF-8 LF {} C M	ITB 🔗		



Multi-core debugging

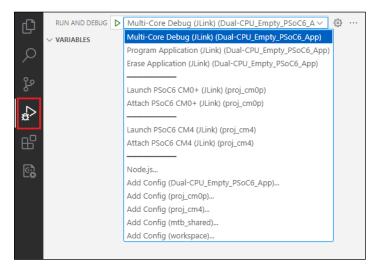
8 Multi-core debugging

Projects created for VS Code also provide debug configurations for multi-core applications. They support these probes:

- KitProg3 onboard programmer
- MiniProg4
- J-Link (See Configure J-Link programmer/debugger settings)

8.1 Configurations

The configurations support debugging one core at a time and multiple cores as well. After the application has opened, there will be several configurations available for use in the **Run and Debug** tab of **Activity Bar** as shown.



These include:

- Multi-Core Debug: programs multiple hex files, launches OpenOCD|J-Link GDB Server and starts multi-core debug session
- Program Application: downloads combined hex file into the flash
- Erase Application: erases all internal memory banks
- Launch <device>: launches debug session on the chosen core
- Attach <device>: attaches to the running core

8.2 Changing programming interface SWD/JTAG

For multi-core debugging, there are more than one separate projects, and each project defines its own debugging interface. If changing from SWD to JTAG, or vice versa, make sure to set the same interface for all projects.

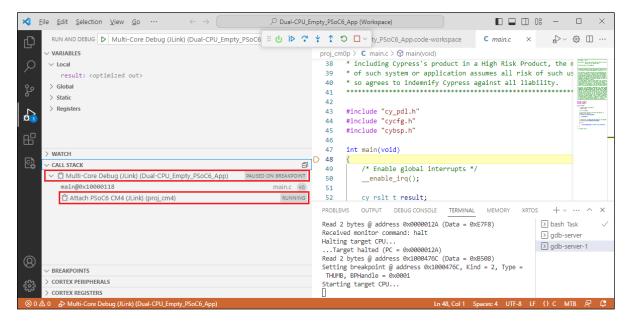
- For **KitProg3_MiniProg4**, set the same interface in all *openocd.tcl* files (see <u>Changing programming</u> <u>interface SWD/JTAG</u> for KitProg3/MiniProg4). There should be one *openocd.tcl* file per project and one for the application located at the application root directory.
- For **J-Link**, set the required interface for "interface" property (see <u>Changing programming interface</u> <u>SWD/JTAG</u> for J-Link). The multi-core debugging configuration also involves launching Attach configurations; define the same interface for each of them.



Multi-core debugging

8.3 Launch the configuration

To launch multi-core debugging, run the **Multi-Core Debug** configuration. You will end up with a debug session containing two debug processes in CALL STACK view.



Once a session has started, the CM0+ core is halted at the beginning of main(), while the CM4 core is spinning in an endless loop in boot code, waiting for start. It will start and halt at main() as soon as the application running on the CM0+ executes the Cy SysEnableCM4() function.

In the CALL STACK view you can observe two debug processes, each of them associated with a specific core. You can switch between the cores by selecting the appropriate process.

Note: There is one limitation for XMC7000 MCUs. Before launching a multi-core debug session, you must program the MCU by launching the **Program Application** configuration.



Multi-core debugging

Revision history

Revision	Date	Description
**	2023-05-16	New document.

Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

Edition 2023-05-16 Published by

Infineon Technologies AG 81726 Munich, Germany

© 2023 Infineon Technologies AG. All Rights Reserved.

Do you have a question about this document? Email: erratum@infineon.com

Document reference 002-37543 Rev. **

Important notice

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie")

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.