## MB2K2 Firmware Build and Flashing Guide

## <u>Introduction</u>

This guide covers the installation of the Xmos xTIMEcomposer toolchain which is used to flash and reprogram the system together with a walkthrough on building the firmware and flashing this to the hardware.

## **Prerequisites**

In order to talk to the XU216's JTAG interface for debugging and programming the on board flash memory it will be necessary to obtain an Xmos 'XA-XTAG' USB debug adaptor. These are really available from component distributors such as Digi-Key for about \$20. Octopart has a list of distributors and stock <u>here</u>. The XTAG plugs vertically into the 20 pin header on the MB2K2 PCB.



Step by step quide

It's recommended to use Windows 10 for firmware development on the MB2K2.

1) Download and install version 14.4.1 of the Xmos xTIMEcomposer toolchain from <u>here</u>. the 14.4.1). You will need to register for a free account, make a note of the username/password as these will be used later.

Run the installer and except all the default options. At the end of the process there should be icons on the desktop for the xTIMEcomposer command prompt, the IDE and a link to documentation.



2) Go to the HackaDay page for the MB2K2 project <u>here</u> and follow the link for the MB2K2\_V0.91-release. Download the latest version of the distribution using the 'clone or download' button or if preferred clone the repository instead.

	Fi	nd file	Clone or dov	vnload 🗸			
	Clone with HTTPS ⑦ Use Git or checkout with 9		a the web LIRI				
ıbl	https://github.com/rumballd/MB2K2_V0.90-						
	Open in Desktop Download ZI						
			29 da	ys ago			

Unzip the dowloaded file to a convenient location, the contents of the unzipped folder should be something like this :-

docs	04/06/2020 16:25	File folder	
📕 firmware	04/06/2020 16:25	File folder	
📕 hardware	04/06/2020 16:25	File folder	
MB2K2 source	04/06/2020 16:25	File folder	
gitattributes	04/06/2020 16:25	GITATTRIBUTES File	1 KB
🧧 cern_ohl_p_v2_howto	04/06/2020 16:25	PDF File	54 KB
🧰 cern_ohl_p_v2-2	04/06/2020 16:25	PDF File	109 KB
license and changes	04/06/2020 16:25	Text Document	1 KB
🧰 read me first	04/06/2020 16:25	PDF File	28 KB
💕 README	04/06/2020 16:25	MD File	2 KB

Note that there is documentation for xTIME composer in the 'firmware/data sheets & app notes etc' folder in the documentation.

3) Run the xTIME Composer Studio (Community\_14.4.1) application (IDE). Note that the IDE needs a 32 bit Java Runtime Environment (JRE) to be installed. Once the IDE has started accept the default workspace location and sign in using the credentials for your Xmos account. Then close the 'XMOS Welcome' tab and select the 'edit' perspective. At this point the window should look something like this :-

📲 O 🔄 🧱 🌍 就 XXMOS Edit - xTM XT XXMOS Edit - xTIMEcomposer	4E-	© ti ✔ □ 40 17:13 50 - 0 ×
File Edit Navigate Search Project Run Window Help		
⊡©]% - dd 0° 10 - \$≠ • O • ∮ •	8~2~2~2~4~	
© Project Diplere 1         C 0         V * • • •           Image: Second S	***	Nondre 11 ° D Ac adres 11 rote enhalte.
Content from https://www.wice.com Usage Generation Gen		
<ul> <li>S/PDIF library (lib_spdif)</li> <li>Sample Rate Conversion Library (lib_src)</li> <li>Board/Device Support</li> <li>A-Series Support Library (lib_a_series_support)</li> </ul>	Console 31 [2] Proteins .i.k Task Verver No consoles to display at this time.	
A Sinte Support Tabley (Inc., Series, Support Tables)     Of The series (Series, Series, Series)     Of The series (Series, Series)     Of The series (Series, Series)     Of The series (Series)     Of The series     Of The series (Series)     Of The series	Length,	

4) Import the MB2K2 project. Select 'import' from the file menu, then expand the 'general' folder icon and select 'Existing Projects into Workspace' then hit 'next'.

XT Import	_		×
Select Create new projects from an archive file or directory.			Ľ
Select an import source:			
type filter text			
<ul> <li>▷ General</li> <li>Q. Archive File</li> <li>▷ Existing Projects into Workspace</li> <li>○ File System</li> <li>□ Preferences</li> <li>&gt; ▷ C/C++</li> <li>&gt; ゐ Git</li> <li>&gt; ▷ Install</li> <li>&gt; È Run/Debug</li> </ul>			~
(?) < Back Next > Finit	sh	Can	icel

Select the 'archive file' option and browse to the project archive in the 'firmware' folder in the MB2K2 distribution package and select 'open'. There will now be a list of project modules looking something like this :-

xT Import					×
Import Projects				1	
Select a directory to sea	irch for existing I	Eclipse projects.			-
O Select root directory:			~	Brows	se
Select archive file:	C:\Users\david	\Desktop\MB2K2_V	0.90-release-mas $  imes $	Brow	se
Projects:					
V lib_gpio (lib_gpio				Select	All
✓ lib_logging (lib_lo ✓ lib_trycatch (lib_tr				Desele	ct All
lib_uart (lib_uart)	,,			Refre	sh
✓ lib_usb (lib_usb) ✓ lib_xassert (lib_xa)	scort)				
✓ lib_xcore_c (lib_xa					
MB2K2 (MB2K2)					
Options					
<ul> <li>Search for nested pro</li> <li>Copy projects into w</li> </ul>	·				
Working sets	onspace				
Add project to work	ting sets				
Working sets:	-		~	Select.	

Hit 'finish' and the project will be imported into the workspace. The 'project explorer' tab should now look like this :-

Project Explorer
> 🐸 lib_gpio [lib_gpio - 1.1.0]
> 🐸 lib_logging [lib_logging - 2.1.0]
> 😂 lib_trycatch [lib_trycatch - 1.0.0]
> 🞏 lib_uart
> 🖆 lib_usb [lib_usb - 3.1.2]
> 😤 lib_xassert [lib_xassert - 3.0.0]
> 😂 lib_xcore_c [lib_xcore_c - 2.0.0]
> 🖆 MB2K2 [1.0.2]

The source file structure consists of a number of Xmos lib projects together with the MB2K2 source folder :-

✓ <sup>™</sup> > MB2K2 [MB2K2 V1\_PCB] > Installed Targets > 🐰 Binaries > 前 Includes > 📴 > bin boot image and build tools
drivers for USB CDC class for Windows7
MON09 source code and build tools
promdisk image DSK (used by xFLASH) > 📴 boot > 🔄 > drivers > 📴 > mon09 > 📴 promdisk 🗸 🚮 > src serial port source
6809 source
uPD7220 source
6821 and PS/2
PROMdisk source
decode/RAM handler source
RAMdisk source
RTC and I2C interface source
XUD USB interface source > 📴 ACIA > 🔐 CPU > 🔐 GDC > 🔐 PIA 🔉 📴 promdisk > 📴 RAM > 📴 ramdisk > 🗁 RTC > 🔄 > USB > 💀 main.xc 🗞 Makefile > mb2k2\_v1.xn
- hardware description file

The use of the IDE for editing the code is outside the scope of this document but full details can be found in the 'xTIMEcomposer' User Guide which can be found in the 'firmware/data sheets & app notes' folder in the distribution.

5) Clean the project. Select 'Clean...' from the 'Project' menu. Select 'clean all projects' and unselect 'Start a build immediately' the hit 'OK'. Only for the first time after the project is imported, clean the project a second time.

Clean all projects	○ Clean projects <u>s</u> elected below	
😂 lib_gpio		
12 lib_logging		
≌lib_trycatch		
😂 lib_uart		
😂 lib_usb		
😂 lib_xassert		
😂 lib_xcore_c		
i≌MB2K2		
Start a <u>b</u> uild immediately		
Build the entire workspace		
Build only the selected projects		

6) Build the project. Expand the 'MB2K2' and then 'src' folders and select 'main.xc'. The build icon (hammer) in the toolbar will then become solid and a build configuration can be selected from the list.

Edit Navigate Search	Project Run
a   🐔 🛨 🗟	👩 🚳 🕻
1 debug_ftdi	
2 debug_usb	1.1.0]
3 release_ftdi	ging - 2.1.0
4 release_usb	
→ 🔑 lib_uart	
🏂 🔷 😂 lib_usb (lib_usb	- 3.1.2]
💼 🕓 🕾 lib vassart flib v	10.001 2.001

There are two classes of build configurations in the project, 'usb' which uses the USB connection for serial connections and 'ftdi' which replaces the USB interface with a pair of buffered UARTs. Each is further split into 'debug' which has optimisation set to -00 and adds support for the use of debugPrintf() and 'release' which has optimisation set to -03 and removes all debugging support.

The release configs should be used when not debugging as the compiled code runs 3-5 times faster than the debug configs!

For now, use the 'release\_usb' configuration. Click on the pulldown arrow next to this and select 'release\_usb' and the build will start.

Progress in the build will be shown and status messages/warnings/errors will appear in the Console tab at the bottom of the screen. With a fresh install there should be no errors and the warning messages can be safely ignored.

When the build is complete then a summary of the resource usage in the two XU216 tiles will be shown something like this :-

Creating mb2k2 release us	o.xe								
Constraint check for tile	[0]:								
Cores available:	8,	used:	7.	OKAY					
Timers available:	10,	used:	7.	OKAY					
Chanends available:	32,	used:	16 .	OKAY					
Memory available:	262144,	used:	234116	OKAY					
(Stack: 5908, Code: 7)	1420, Dat	ta: 156788	3)						
Constraints checks PASSED									
Constraint check for tile	[1]:								
Cores available:	8,	used:	7.	OKAY					
Timers available:	10,	used:	9.	OKAY					
Chanends available:	32,	used:	25 .	OKAY					
Memory available:	262144,	used:	259872	OKAY					
(Stack: 35636, Code: 3	15256, Da	ata: 20898	30)						
Constraints checks PASSED									
xmap: Warning: More than	6 cores 1	used on a	tile. Ensure	e this is	not th	e case	on ti	le running.	XUD.
xmap: Warning: More than	6 cores 1	used on a	tile. Ensure	e this is	not th	e case	on ti	le running.	XUD.
Build Complete									

The map warnings can be ignored but note that the memory for the two tiles is quite full. This should be borne in mind when modifying the firmware. (see the Firmware notes in section 3 of the User Notes).

7) At this point the compiled binary may be loaded onto the hardware or placed into the flash memory and these operations will use the XA-XTAG adaptor to communicate with the XU216 via it's JTAG interface. If not there already the XTAG should be plugged into the MB2K2 and it's microUSB cable connected to the host computer. Note that it's safe to connect and disconnect the XTAG when the MB2K2 is powered on providing that the usual precautions are taken to avoid static discharges.

## Debugging and flashing the firmware

Window	Help		
• *	• 🚺 •	<b>7</b> -	Ŷ

The 'Debug' toolbar icon (on the left hand side above) will load the compiled firmware onto the device's RAM via the XTAG adaptor. Note that as this uses the RAM of the XU216 the firmware will be erased when power is removed. A configuration needs to be set up the first time debug is used. Hit the dropdown beside the debug button and choose 'Debug Configurations...', double click on 'xCORE Application' and name the new config. Many of the settings will be pre filled but it may be necessary in the 'main' tab to browse and select the project and application. Check that the XTAG is selected and showing that it is connected to the target device and lastly hit 'apply' to save the changes.

🗎 🗶 🗎 🔿 👻	Name: MB2K2 release_usb					
pe filter text	Main Main Sir	mulator 👿 XSco	pe 🕸 Debu	gger 🤟 Source	e) 🗖 Common	
<ul> <li>ARM Application</li> <li>Launch Group</li> </ul>	Project:					
<ul> <li>Launch droup</li> <li>xCORE Application</li> <li>MB2K2 release_usb</li> </ul>	MB2K2					Browse.
	Build configuration: Use Active					
	,	Sel	lect configur	ation using 'C/	C++ Application'	
	C/C++ Application:					
	bin/release_usb/mb2k2_releas	e_usb.xe				
				Variables	Search Project	Browse.
	Run JTAG I/O server     Run UART output server     Run XScope output server     Enable GPROF collection     Other xgdb options:     Display run command in new	w console				

Now hit 'Debug', the console will show the code being loaded via the XTAG and the IDE will switch to the debug perspective. The use of the IDE for debugging is outside the scope of this document but full details can be found in the 'xTIMEcomposer' User Guide which can be found in the 'firmware/ data sheets & app notes' folder in the distribution.

In the same manner as debug, it is necessary to create a flash configuration before this function can be used. Hit the dropdown beside the 'flash' button and choose 'Flash Configurations...' then double click on 'xCORE Application' and name the new config. Unlike the debug config there are many settings to be specified for flashing. In the 'main' tab browse and select the project and application if necessary and check that the XTAG is selected and showing that it is connected to the target device. The 'main tag of the config should look something like this :-

Create. manage, and run configurations         Image: Image and run configurations         Image: Image and run configuration         Image: Im	XT Flash Configurations			×
Walk (************************************	Create, manage, and run con	figurations		4
Walk (************************************				$\mathcal{V}$
type filter text            Main	🖹 🗎 🗶 🖻 🎲 🖛	Name: MB2K2 flash		
▶ New_configuration       Project:         MB2K2       Browse         Build configuration:       Use Active         ☑ Select configuration using 'C/C++ Application'       C/C++ Application:         bin/velease_usb/mb2k2_release_usb.xe       Variables         Device options:       Target: XMOS XTAG-3 connected to O(0) [ZByhc2hJ] ∨         Refresh list       Display run command in new console				
MB2K2     Browse       Build configuration:     Select configuration using 'C/C++ Application'       C/C++ Application:     Din/velease_usb/mb2k2_release_usb.xe       Variables     Search Project       Device options:     Target:       Target:     XMOS XTAG-3 connected to O(0) [ZByhc2hJ] v       Refresh list     Display run command in new console		Project:		
Select configuration using 'C/C++ Application' C/C++ Application: bin\release_usb\mb2k2_release_usb.xe Variables Search Project Browse Device options: Target: XMOS XTAG-3 connected to O(0) [ZByhc2hJ] ∨ Refresh list Display run command in new console	<ul> <li>New_conliguration</li> </ul>	MB2K2		Browse
C/C++ Application: bin\release_usb\mb2k2_release_usb.xe Variables Search Project Browse Device options: Target: XMOS XTAG-3 connected to O(0) [ZByhc2hJ] v Refresh list Display run command in new console		Build configuration:	Use Active	~
bin\release_usb\mb2k2_release_usb.xe         Variables       Search Project         Device options:         Target:       XMOS XTAG-3 connected to O(0) [ZByhc2hJ] ~         Refresh list         Display run command in new console			Select configuration using 'C/C++ Application'	
Variables       Search Project       Browse         Device options:       Target: XMOS XTAG-3 connected to O(0) [ZByhc2h] ~       Refresh list         Display run command in new console       Display run command in new console       Refresh list				
Device options: Target: XMOS XTAG-3 connected to O(0) [ZByhc2h] ~ Refresh list Display run command in new console		bin\release_usb\mb2k2_release_us		
Target:       XMOS XTAG-3 connected to O[0] [ZByhc2hJ]       Refresh list         Display run command in new console       Display run command in new console			Variables Search Project	Browse
Display run command in new console				
Filter matched 2 of 2 items Apply Revert		Display run command in new co	nsole	
Filter matched 2 of 2 items Apply Revert				
Filter matched 2 of 2 items Apply Revert				
Filter matched 2 of 2 items Apply Revert				
Filter matched 2 of 2 items Apply Revert				
Filter matched 2 of 2 items Apply Revert				
Filter matched 2 of 2 items Apply Revert				
Filter matched 2 of 2 items Apply Revert				
Filter matched 2 of 2 items Apply Revert				
	Filter matched 2 of 2 items		Apply	Revert
Plash Close	?		Flash	Close

in the 'XFlash options' tab select the 'boot partition size' option and set the value to '0x100000' (1MB). In 'other Flash options' set '-verbose --no-compression --data promdisk/mb2k2pd.dsk'

XT Flash Configurations				×
Create, manage, and run con	figurations			
-	-			7
[] 🗎 🗶 📄 🔆 ▼				
type filter text	Name: MB2K2 flash			
✓ ► xCORE Application	Main 🖻 XFlash Options			
New_configuration	Output to file (disables device programming):		Browse	
	Encrypt using AES key file:		Browse	
	SPI spec file:		Browse	
	Boot partition size (bytes):	0x100000		
	Other XFlash options:	>romdisk/mb2k2pd.dsk		
Filter matched 2 of 2 items			Apply	Revert
_				
?			Flash	Close

Lastly hit 'apply' to save the changes and then 'flash'. The process will take a few mins to complete displaying progress via a scrolling list of

<

block addresses as each is written and verified. At the end, the MB2K2 should restart and the MON09 or FLEX prompts should appear.