XeThru Module Firmware Programming

How to Program the MCU on XeThru sensor and development kit.

XeThru Application Note **by Novelda AS**

Rev. B - November 26. 2018

Summary

This application note shows how to connect XeThru module and Atmel-ICE for firmware programming.

1 Introduction

This application provides an introduction on how to connect and program an XeThru module with an Atmel-ICE debugger.

2 Procedure

2.1 Erasing the Microcontroller Content

> **WARNING**
>
> This step will erase all content on the on-board microcontroller (MCU). It will no longer be possible to use the module as a XeThru sensor, e.g. X4M200/X4M300.

The X4M200 and X4M300 share same hardware design X4M02. X4M03 and X4M06 share same MCU board XTMCU02. Module's MCU content has to be manually erased the first time before flashing custom firmware. The erase pin is located at test point TP304 for X4M02 (Fig. 1) and TP202 for XTMCU02 (Fig. 2). The MCU is erased by following steps:

1. Disconnect module power
2. Pull the erase input high (3V)
3. Power the module via USB or connector P1
4. Hold erase input high for at least 5 seconds

The TP104 will always output 3V when the module is powered on, and can be used to pull up erasing test point. Another option is connecting erasing test point directly to a power supply. After this is done, the LED is blue for about 5 seconds before it is turned off. MCU is now erased and is ready to be programmed.

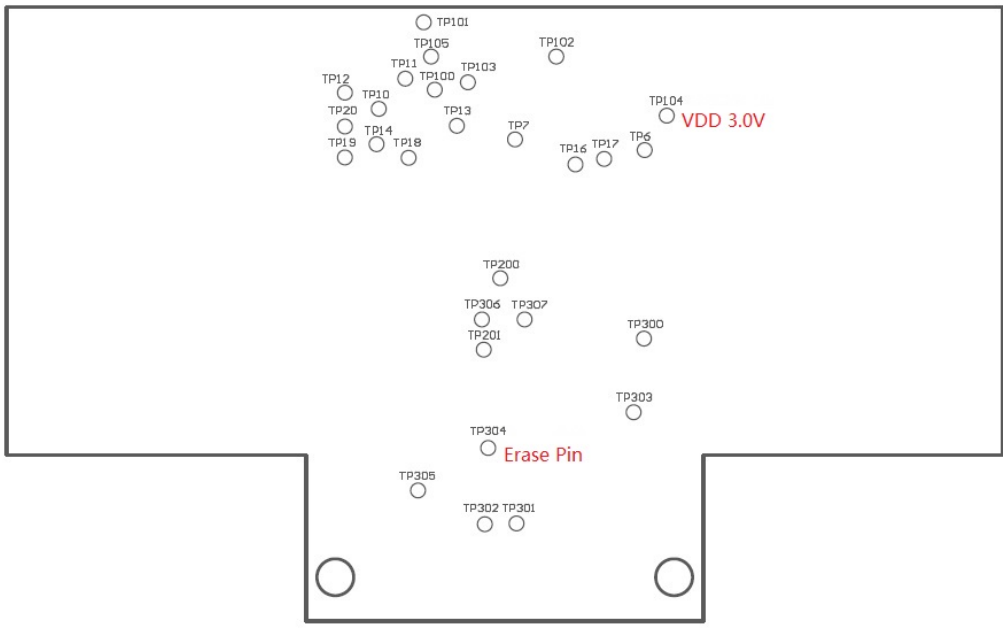

Fig. 1. X4M02 test points.

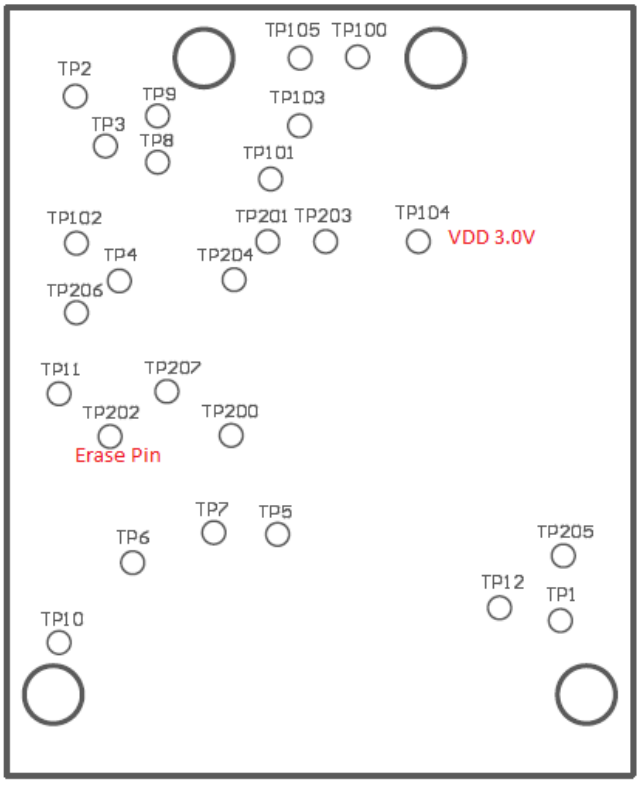

Fig. 2. XTMCU02 test points.

2.2 Hardware connection

Atmel-ICE is used as a debugger and programming tool for Atmel MCU. It supports both SAM and AVR devices. X4M02 and XTMCU02 are connected through the SAM port. The pin definitions can be found in Atmel-ICE documentation [1]. Table 1 shows how to connect an Atmel-ICE to the X4M02 pin header P1. Developer have to make custome cable for connection between X4M02 and Ateml-ICE SAM port. Please refer to the X4M02 hardware documentation [2] for details.

Tab. 2. X4M02 Pins

Pin on Atmel ICE	Pin on X4M02	Name	Comment
1	TP104	VCC	Used as voltage reference by Atmel-ICE. As an alternative, use a 3V external power supply
2	10	SWDIO	
3+5+9	2	GND	Used as GND reference by Atmel-ICE. As an alternative, use GND on the external power supply
4	9	SWDCLK	
6	8	SWO	
10 (or 0)	7	nRESET	

XTMCU02 has a 50mil 10-pin connector with the 10-pin Cortex Debug Connector pinout. It can be directly connected to an Atmel-ICE using the cable included with the Atmel-ICE. Take care to connect it to the connector labeled "SAM" on the Atmel-ICE. Connect the USB cable from the Atmel-ICE to the PC.

Note that the X4M02/XTMCU02 must be powered separately to be able to program it.

2.3 Programming X4M02/XTMCU02

The X4M02/XTMCU02 MCU can be programmed as usual now. Follow instructions on how to flash X4M03 XeThru Embedded Platform (XEP) [3]. Developer can use the same way to program X4M02/XTMCU02 with their custom firmware.

2.3.1 Programming the X4 module with normal firmware

To program the X4M02/XTMCU02 a programmer or debugger supporting the Microchip SAMS70-series (e.g. Atmel-ICE, Power Debugger, Segger J-Link) is required. In addition, a software supporting the device and programming tool is required (e.g. Atmel Studio 7 on Windows or OpenOCD for multiple platforms).

Programming using Atmel Studio
1. Open Atmel Studio 7
2. Open the Programming dialog, either from the top menu Tools->Device Programming, or by pressing Ctrl + Shift + P
3. Select the tool (e.g. Atmel-ICE), device (ATSAMS70Q20 or ATSAMS70Q21) and Interface (SWD)
4. Click Apply. If you are asked to upgrade the tool, click Upgrade and wait for the upgrade to finish before clicking Close, and then click Apply again.
5. Test the connection by clicking the button labeled Read under Device Signature. It should state a 32-bit hexadecimal value. If it displays an error, review the error and fix the hardware connections or settings.
6. On the left hand side, find the tab Memories and click it
7. Under Flash, click the browse button (...) and navigate to the hex file you wish to program
8. Click Program
9. Go to GPNVM Bits and verify that BOOT_MODE is checked. Otherwise check BOOT_MODE and click Program.
10. X4M02/XTMCU02 should now be properly programmed

2.3.2 Programming the X4 module with OpenBootloader firmware

Follow the guide above for general programming of the X4M03.
Use the file called xep_x4m0x_s70_with_bldr.hex
After finishing step 10 in the above guide:
11. Go to the Lock bits tab
12. Type 0xFFFFFFFF into LOCKBIT_WORD0 and press Program
13. X4M02/XTMCU02should now be properly programmed with OpenBootloader and XEP

2.3.3 Using OpenBootloader to program the X4 module

OpenBootloader comes bundled with a couple of scripts; add_checksum.py and upgrade_firmware.py.

Calculating checksum for custom hex image
1. Build XEP and locate the generated *_4bl.hex file
2. Run add_checksum.py, and specify the input file by -i and output file by -o (e.g. python add_checksum.py -i ..\bin\xep_x4m0x_s70_4bl.hex -o xep_x4m0x_s70_4bl_checksum.hex)

Uploading firmware image
1. Locate the checksummed hex file to upload (e.g. xep_x4m0x_s70_4bl_checksum.hex)
2. Run upgrade_firmware.py, and specify the checksummed hex file (e.g. python upgrade_firmware.py -f xep_x4m0x_s70_4bl_checksum.hex)
3. upgrade_firmware.py should automatically detect the correct COM port when using USB. If using UART, or the autodetect fails, the COM port could be specified by using -d (e.g. python upgrade_firmware.py -f xep_x4m0x_s70_4bl_checksum.hex -d COM18).

3 Reference

[1]	Atmel, "Atmel ICE user guide", http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-ICE_UserGuide.pdf
[2]	Novelda, "X4M02 datasheet", https://www.xethru.com/community/resources/x4m02-radar-sensor-datasheet.115/
[3]	Novelda, "XeThru Embedded Platform", https://www.xethru.com/community/resources/xep-binary.88/

4 Document History

Rev.	Release Date	Change Description
A	2018-04-20	Initial release
B	2018-11-26	Add XTMCU02 board

5 Disclaimer