



Please note that Cypress is an Infineon Technologies Company.

The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content

The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers

Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.



THIS SPEC IS OBSOLETE

Spec No: 002-09180

Spec Title: FM3/FM4/FM0+ Family Universal Programmer User Manual

Replaced by: None



OBSSO

FM3/FM4/FM0+ Family

Universal Programmer User Manual

Doc. No. 002-09180 Rev. *A

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone (USA): 800.858.1810
Phone (Intl): +1 408.943.2600
www.cypress.com

© Cypress Semiconductor Corporation, 2013-2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC (“Cypress”). This document, including any software or firmware included or referenced in this document (“Software”), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress’s patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage (“Unintended Uses”). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

Contents



1. Introduction	4
1.1 About Off-line Programmer	4
1.2 About FM MCU	4
1.3 About Programmer Features	5
2. Component	6
2.1 PGM Board	6
2.2 PC Configuration Tool	9
3. Operation Process	13
3.1 SWD Programming Process	13
3.2 UART Programming Process	21
4. Program.ini Introduction	26
4.1 MCU Section	28
4.2 MCU Flash Section	28
4.3 External Flash Section	28
5. Maintenance	29
6. Troubleshooting	Error! Bookmark not defined.
6.1 Connection Error	Error! Bookmark not defined.
6.2 Operation Error	Error! Bookmark not defined.
6.3 Operation Warning	Error! Bookmark not defined.
7. Additional Information	30
7.1 Reference Documents	Error! Bookmark not defined.
Revision History	31
Document Revision History	31

1. Introduction



This user manual describes how to use the FM MCU Universal Programmer (named as PGM hereinafter) to serve as an off-line programmer for FM series MCU and on-board external flash memory (Quad SPI Flash, Hyper Flash, and NAND Flash).

Target products

This user manual describes how to use the programmer in the following products:

FM3 MCU: TYPE0 – 12

FM4 MCU: TYPE1 – 6

FM0+ MCU: TYPE1 – 3

1.1 About Off-line Programmer

The off-line programmer enables users to update the MCU's program memory or on-board external flash memory without removing the mounted MCU chip or flash memory chip from the actual end product.

1.2 About FM MCU

FM microcontrollers incorporate the latest ARM® Cortex® standard cores (M3, M4 and M0+), offering customers the optimal product for a wide range of industrial and consumer applications. The scalable platform ranges from low-pin-count, low-power microcontrollers to high-performance products with a rich set of peripherals.

- Outstanding performance
- Functional safety
- High-performance flash memory
- Advanced peripherals

Now there are 13 different types of MCUs in FM3 family; for FM4, the type number is 6, and FM0+ is 3. For all these different types of MCUs, there are 3 types of flash structure: Main, Dual, and Main + Work. This enables the customer to choose the appropriate MCU for the application according to the flash structure.

1.3 About Programmer Features

The main features of this programmer are as follows:

1. Support all types of FM3 MCU (0 - 12), Type 1-6 of FM4 and Type 1-3 of FM0+ (please check the type information in the datasheet of the MCU).
2. Program interface: UART or SWD.
3. Storage media: SD Card or USB-Disk (not supported yet).
4. One key operation.
5. Power supply optional: USB/DC In/Battery.
6. Status display by 3 LEDs.
7. Programming file operation is controlled by a configuration file which is output by the PC.

2. Component

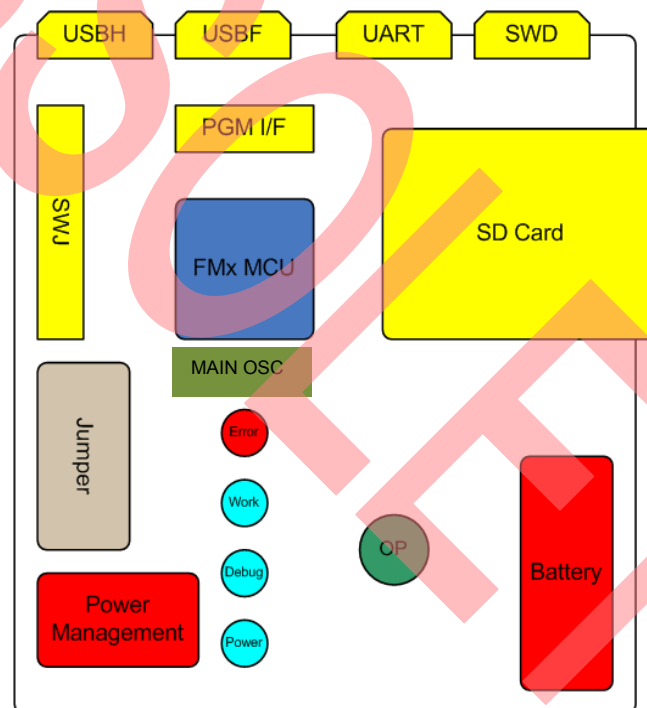
The whole PGM includes: PGM Board and PC Configuration Tool.

2.1 PGM Board

The board provides the hardware method to operate the target board.

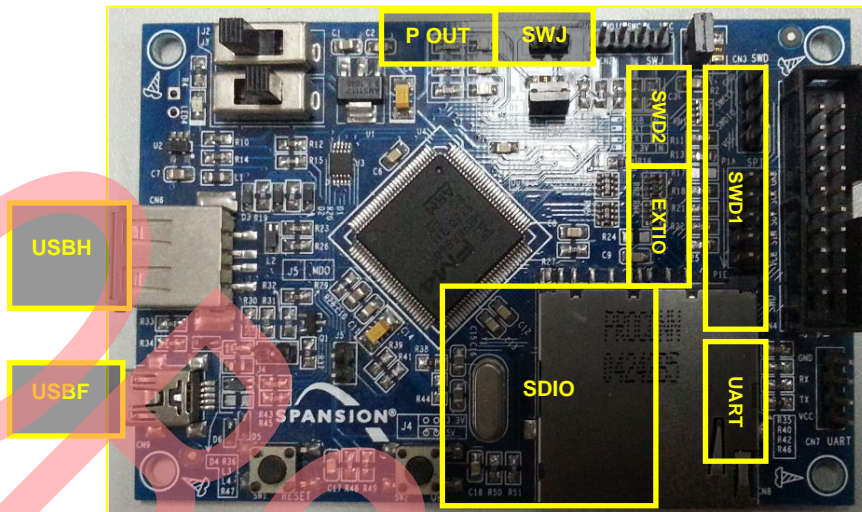
- The architecture of the whole system

Figure 2-1. PGM Architecture



■ Introduction to the interfaces

Figure 2-2. PGM Interface



UART: 4-pin interface. This interface can be used to serve as the communication interface to program the target board by asynchronous protocol.

SWD1: 20-pin interface. This interface can be used to serve as the communication interface to program the target board by SWD protocol.

SWD2: 6-pin interface. The function is the same as SWD1.

USBH: USB host socket. USB-Disk with the target files can be connected with the PGM through this interface (not supply u disk).

USBF: USB device socket. Used for power supply or served as the debug interface for CMSIS-DAP (not supported yet).

SWJ: System debug interface, used to debug the PGM.

SDIF: SD card socket, SD card with the target files can be connected with the PGM through this interface; supports SDSC and SDHC cards.

Power output: Provide DC power output.

■ **Jumper table**

Table 2-1. Jumper Table

Jumper	Function	Setting
J1	JTAG power output	Open: No power output from CN4-19 Close: Power output from CN4-19
J2	Battery/USB input selection	Right: Battery input Left: USB input
J3	Ext power voltage selection	Right: Direct external power input Left: Adjust the external input power to 3.3V
J4	USB input voltage adjusting	Open: VCC MCU is 3.3V Close: VCC MCU is 5V
J5	Mode pin	Open: Normal run mode Close: Boot loader code run mode
J7	Function pin	Open: UART program mode Close: SWD program mode

Note: Please check the [Figure 2-2](#) for recognizing the 'Left' and 'Right'.

■ **Power supply**

The PGM can be powered by:

JATG: SWJ

USB: J2 → Right

Battery: J2 → Left

■ **Voltage selection**

Use 3.3 V

J2 → Right - USB (5 V) / Left - Battery (4.5 V) input

J3 → Left

J4 → Close

J5 → Open

Use 5 V

USB (5 V) input

J2 → Left

J3 → Right

J4 → Open

J5 → Open

2.2 PC Configuration Tool

The tool controls the operation on the target board. This tool can be run on Win 7 (32-bit) and Win 7 (64-bit). By using this tool, you can set the target board and get the input files for the PGM.

- Overview

Figure 2-3. MCU Page of PGM CFG Tool

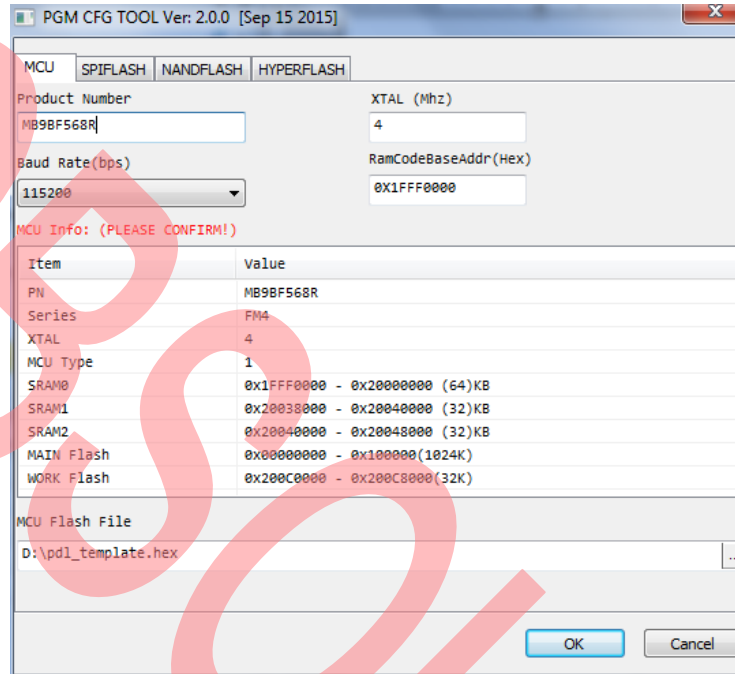


Figure 2-4. External SPI Flash Page of PGM CFG Tool

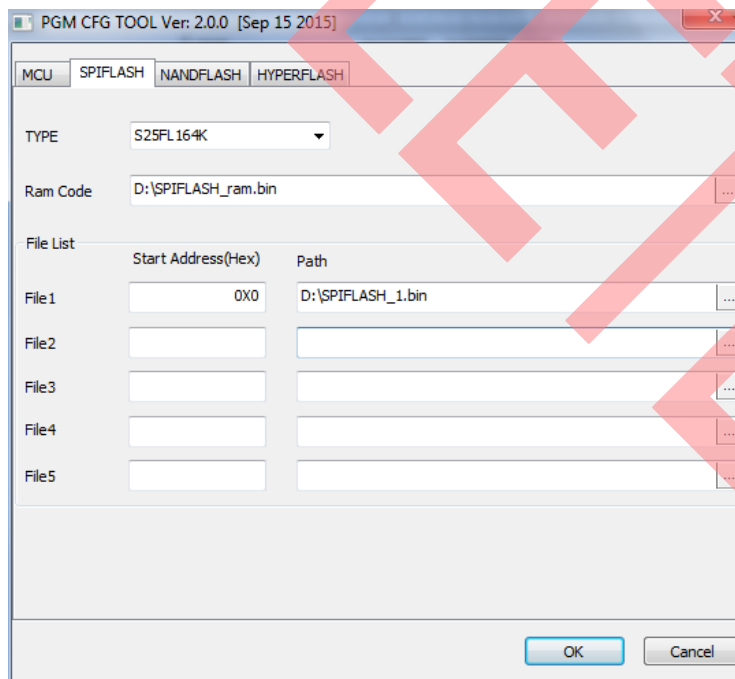


Figure 2-5. External NAND Flash Page of PGM CFG Tool

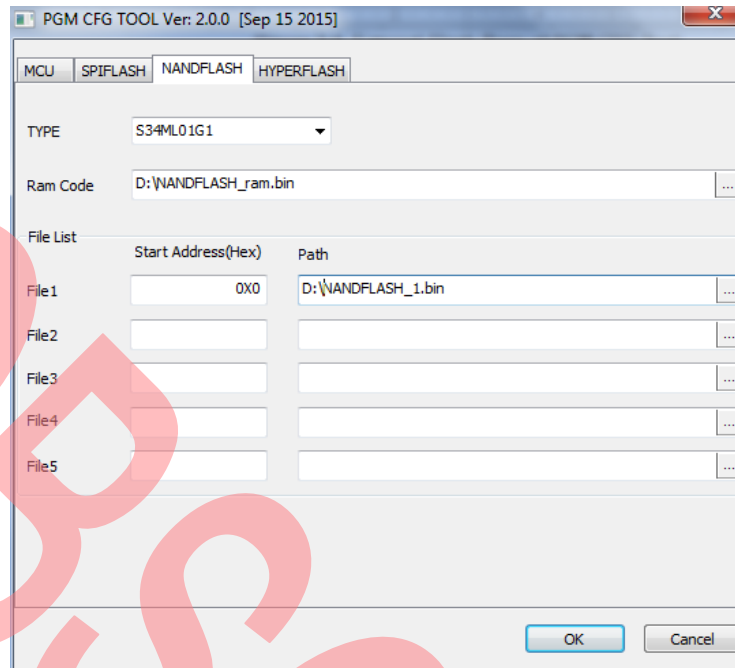
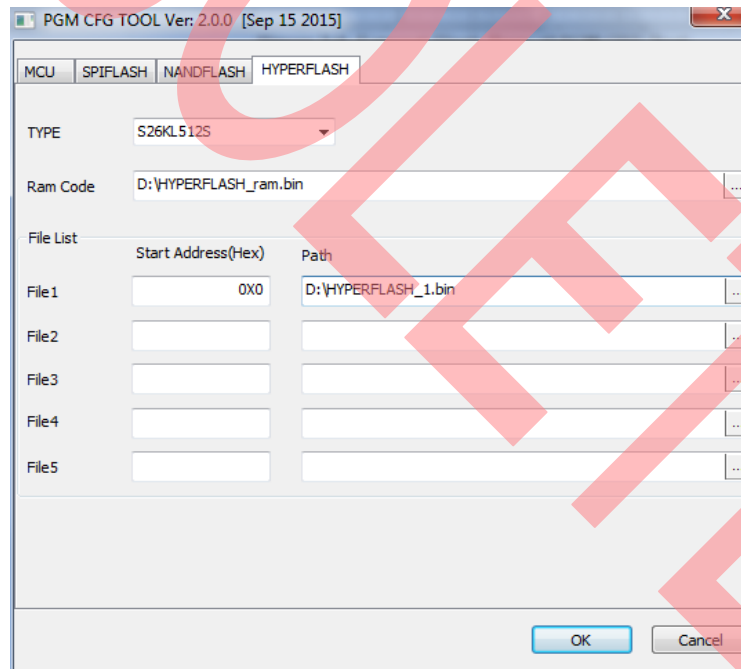


Figure 2-6. External Hyper Flash Page of PGM CFG Tool



Notes:

- The external flash includes SPI flash, NAND flash and hyper flash, so you can see SPIFLASH page, NANDFLASH page and HYPERFLASH page in the tool GUI.
- The programmer supports programming up to five files.
- The RAM code base address and the external flash data file start address must be the hex value.

■ **Introduction to MCU Page of Each Item**

XTAL: Input the external oscillator number of the target board.

Baud Rate: Set the UART baud rate between the PGM and the target board.

RamCodeBaseAddr: Input the run address of the external flash RAM code.

MCU Info: Display the detailed information of the selected MCU.

MCU Flash File: Select the internal flash programming hex file.

Product Number: The part number of the dedicated Product Number MCU

Note: The 'RamCodeBaseAddr' is the MCU base ram address, and it will be automatically set after the 'Product Number' is set.

Table 2-2. Product Number Naming Rule

Type	Naming rule	Sample
FM3	MB9XFxyzY X: A/B x: function description y: family z: flash size Y: J/K/L/M/N/R/S/T, pin number	MB9BF506R
FM4	MB9XFxyzY Same as FM3	MB9BF568R
	S6E2XxzY X: C/D/G/H x: function description z: flash size Y: B/C/D/E/F/G/H/J/K/L, pin number	S6E2CCAJ
FM0+	S6E1XxzY X: A/B/C x: function description z: flash size Y: B/C/D/E/F/G/H/J/K/L, pin number	S6E1A12C

■ **Introduction to External Flash Page of Each Item**

TYPE: Select the external flash memory type.

Ram Code: Select the external flash ram code bin file.

File1 Start Address: Set the first external flash programming file start address.

File1 Path: Select the external flash first programming file.

File2 Start Address: Set the second external flash programming file start address.

File2 Path: Select the external flash second programming file.

File3 Start Address: Set the third external flash programming file start address.

File3 Path: Select the external flash third programming file.

File4 Start Address: Set the fourth external flash programming file start address.

File4 Path: Select the external flash fourth programming file.

File5 Start Address: Set the fifth external flash programming file start address.

File5 Path: Select the external flash fifth programming file.

Cancel: Close the tool.

Ok: Generate.

■ **Output Files**

The generated output files are located in 'PGM_OUTPUT' folder. The output files are listed in the following table:

Table 2-3. Output Files

File Name	Type(MCU/Board)	Function	Sample
program.ini	MCU Type: All MCU flash and external flash	Serve as the configuration file to guide the PGM to program the target board	program.ini
x_MAIN.bin	MCU Type: All MCU flash (except type 6, 8, 9, 12 of FM3)	The bin file stored at the main flash	MB9BF568R_MAIN.bin
x_MAIN2.bin	MCU Type: Type 3, 5 of FM4 MCU flash	The bin file stored at the second main flash area	S6E2CCA_MAIN2.bin
x_DUAL0/1.bin	MCU Type: Type 6, 8, 9, 12 of FM3 MCU flash	The bin file stored at the dual flash is named as x_DUAL0.bin The bin file stored at the dual flash is named as x_DUAL1.bin	MB9AFB44NA_DUAL0.bin MB9AFB44NA_DUAL1.bin
x_WORK.bin	MCU Type: Type 4, 5 of FM3 MCU flash Type 1, 2, 6 of FM4 MCU flash	The bin file stored at the work flash	MB9BF568R_WORK.bin
SPIFLASH_ram.bin	Board Type: SK-FM4-216-ETHERNET	The bin file run at the target MCU ram area	SPIFLASH_ram.bin
SPIFLASH_y.bin		The flash file stored at the SPI flash memory	SPIFLASH_1.bin
NANDFLASH_ram.bin	Board Type: SK-FM4-U120-9B560	The bin file run at the target MCU ram area	NANDFLASH_ram.bin
NANDFLASH_y.bin		The flash file stored at the NAND flash memory	NANDFLASH_1.bin
HYPERFLASH_ram.bin	Board Type: SK-FM4-176L-S6E2DH	The bin file run at the target MCU ram area	HYPERFLASH_ram.bin
HYPERFLASH_y.bin		The flash file stored at the Hyper flash memory	HYPERFLASH_1.bin

x: MCU product number; y: the external flash data file number (1 - 5).

3. Operation Process



3.1 SWD Programming Process

You can program the internal flash and external flash memory of the target board through SWD communication interface. The operation steps are as follows:

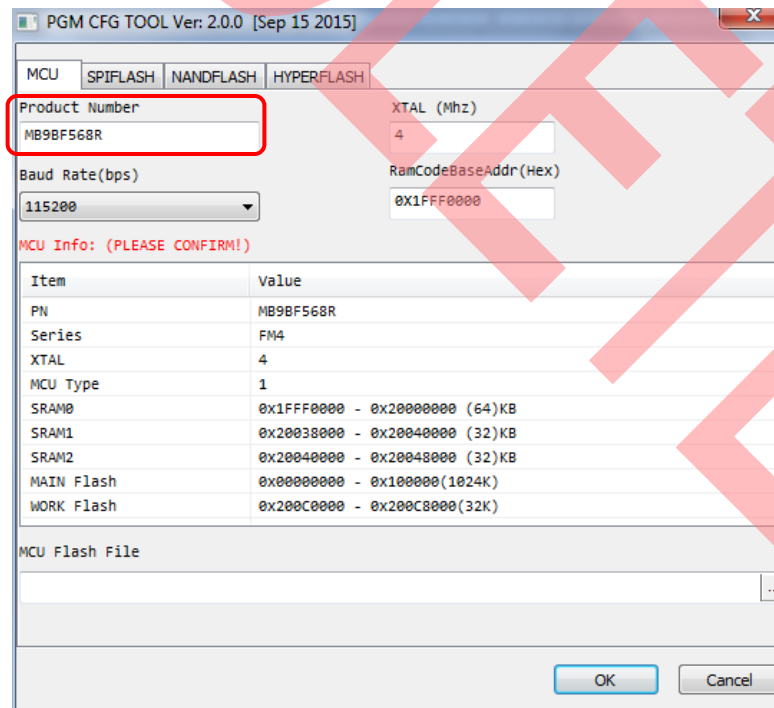
3.1.1 Prepare the Files

- If you program the internal flash of the target board, you can get the hex (Intel mode) file through IAR or KEIL, for example, the file name is 'pdl_template.hex'.
- If you program the external flash memory of the target board, you need the external flash RAM Code bin file and the flash data files of the external flash memory. The external flash RAM Code bin file is provided by the Cypress.

3.1.2 Convert and Get the Target Board Files

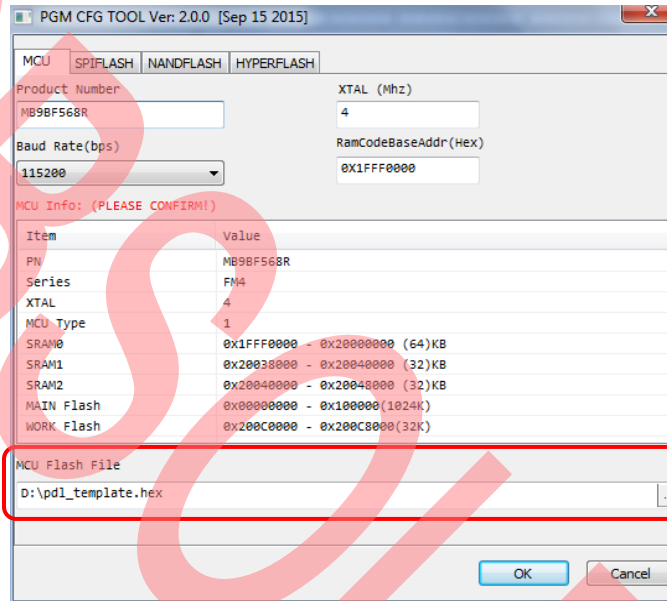
1. Run the 'hex2bin_Demo.exe'
2. Input the 'Product Number' of target board

Figure 3-1. Product Number



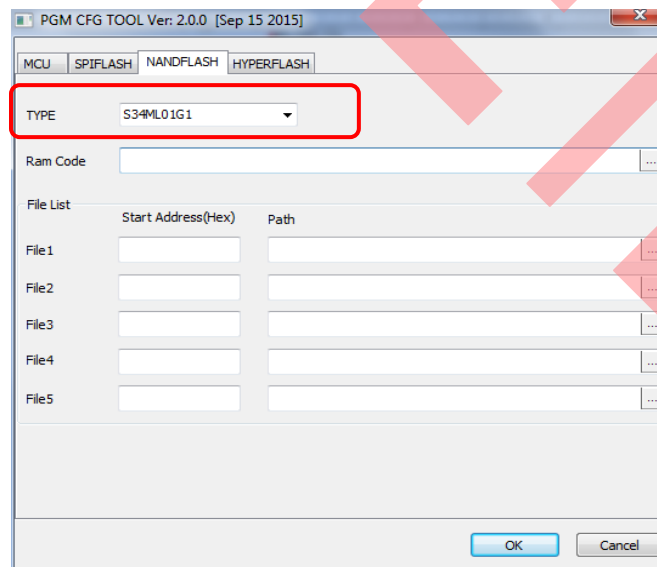
- a. If you input the proper product number, the detailed information of the MCU will be displayed below the 'MCU Info:' as show in [Figure 3-1](#).
 - b. If no detailed information is displayed at that place, please make sure the part number of the target MCU is correct. Refer to [Table 2-2](#) of the product number naming rule.
3. Do not set the 'XTAL'
 4. Do not select the 'Baud Rate'
 5. If you program the internal flash memory of the target board, you need select the internal flash file, as show in [Figure 3-2](#). If you don't program the internal flash memory, please skip this step.

Figure 3-2. Internal Flash File Selection



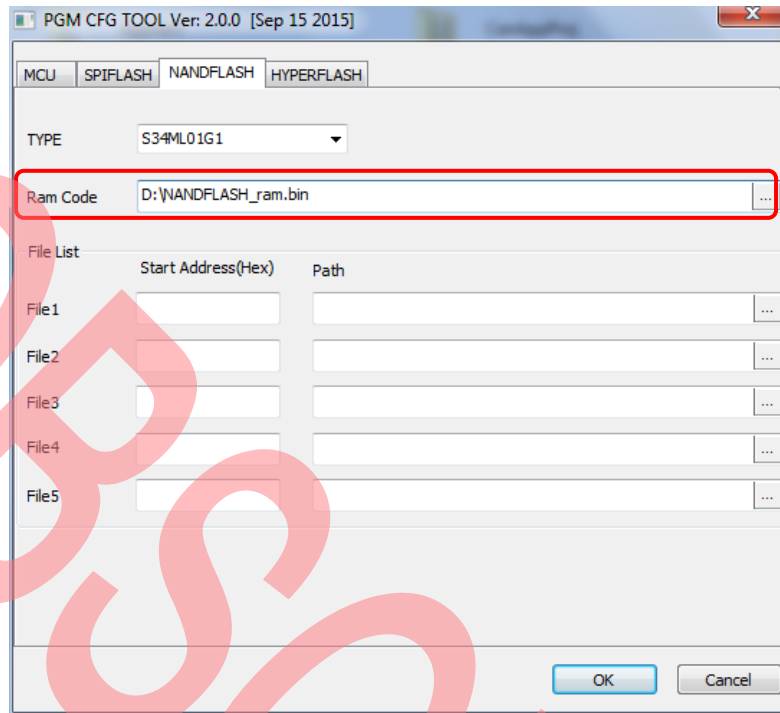
6. If you do not program the external flash memory of the target board, please skip to step 10. If you program the external flash memory, you need to open the page for setting external flash and select the external flash type, as show in [Figure 3-3](#).

Figure 3-3. External Flash File Selection



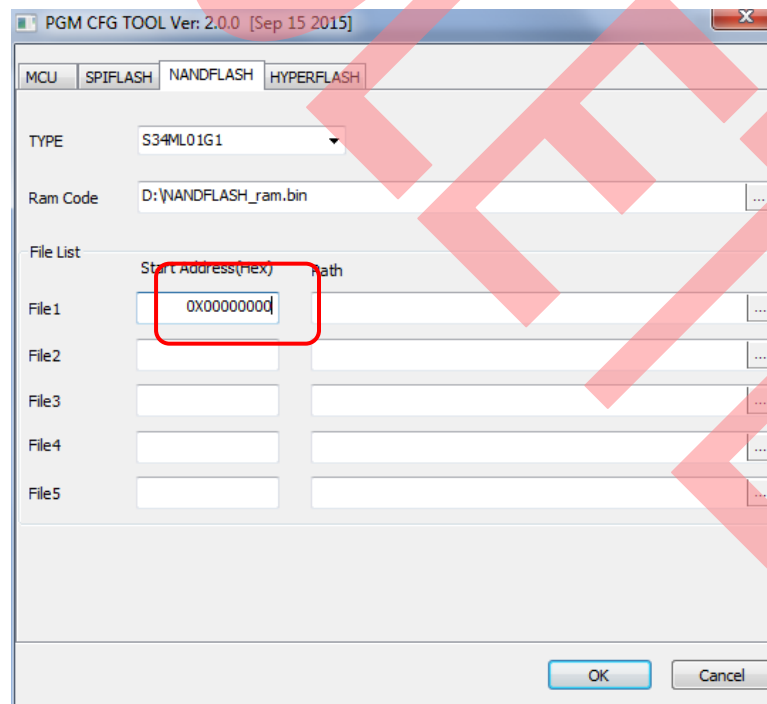
7. Select the path of external flash RAM code.

Figure 3-4. Select External RAM Code Path



8. Input the start address of external flash operation.

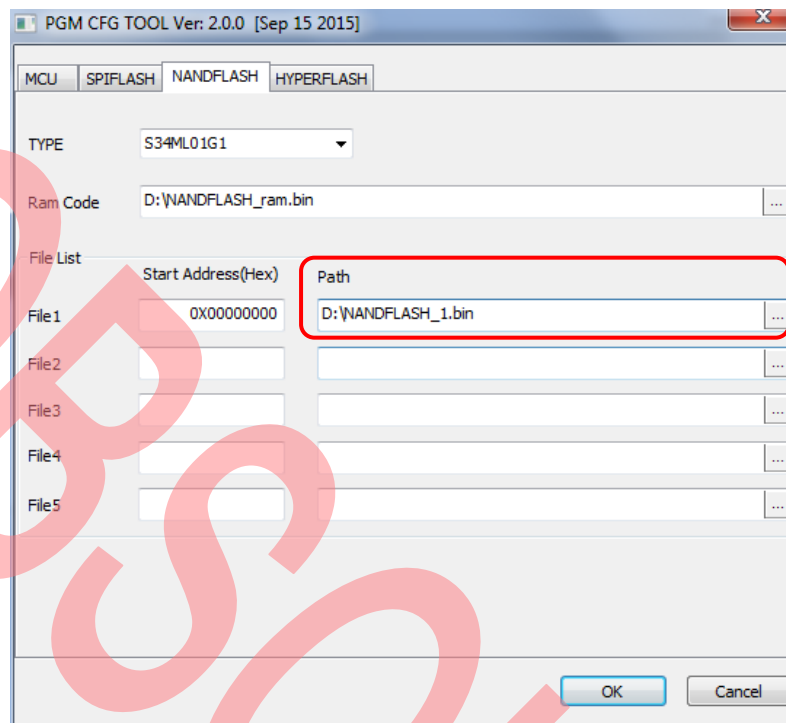
Figure 3-5. Input External Flash Address Info



Note: The value you entered must be in hexadecimal format.

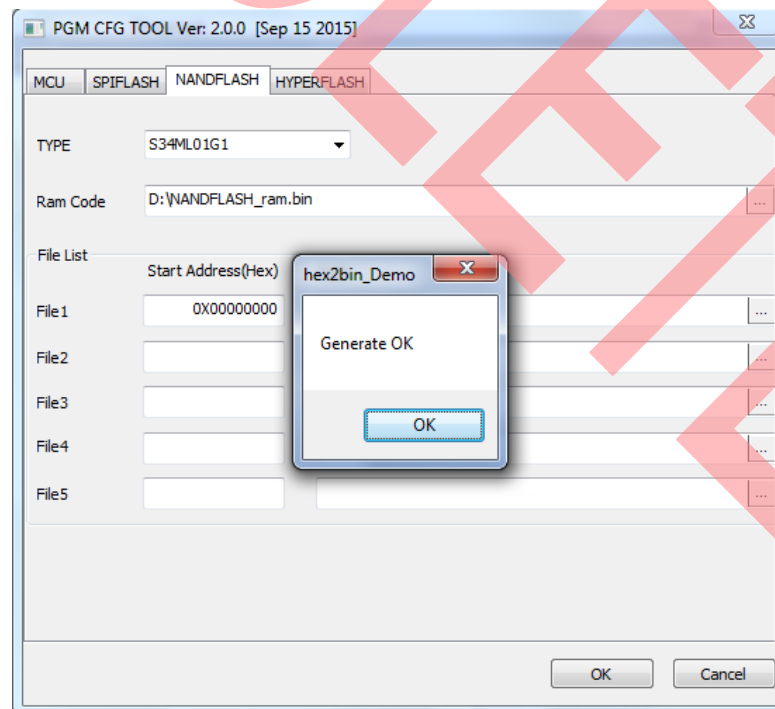
9. Select external flash data file, as shown in Figure 3-6. If you need to select other files, please repeat steps 8 and 9.

Figure 3-6. Select External Flash File Path








10. Press **OK** to generate data.

Figure 3-7. Generation Result



11. Check the output

Figure 3-8. Convert Result

Name	Type	Size
 MB9BF568R_MAIN.bin	BIN File	1,024 KB
 MB9BF568R_WORK.bin	BIN File	32 KB
 NANDFLASH_1.bin	BIN File	1,322 KB
 NANDFLASH_ram.bin	BIN File	10 KB
 program.ini	Configuration settings	1 KB

- a. If you program the internal and external flash memory of the target board, the 'PGM_OUTPUT' folder includes 'MB9BF568R_MAIN.bin', 'MB9BF568R_WORK.bin', 'NANDFLASH_1.bin', 'NANDFLASH_ram.bin' and 'program.ini'.
- b. If you only program the internal flash memory of the target board, the 'PGM_OUTPUT' folder includes 'MB9BF568R_MAIN.bin', 'MB9BF568R_WORK.bin' and 'program.ini'.

12. Copy these files to the SD card

3.1.3 Hardware Connection and Setting

1. PGM Board Power and jumper setting

USB (5V) power input

J1: Close

J2: Left

J3: Left

J4: Open

J5: Open

J7: Close

Battery (4.5V) power input

J1: Close

J2: Right

J3: Left

J4: Open

J5: Open

J7: Close

2. Target Board Mode Setting

Set the target board MCU to serial programming mode.

MD0: High

MD1: Low

3. Communication line (SWD) connection

SWD1 (pin number):

Pin 2 (Universal PGM) ↔ GND (Target Board)

Pin 7 (Universal PGM) ↔ SWDIO (Target Board)

Pin 9 (Universal PGM) ↔ SWDCLK (Target Board)

Pin 19 (Universal PGM) ↔ VCC (Target Board)

SWD 2 (pin function):

GND (Universal PGM) ↔ GND (Target Board)

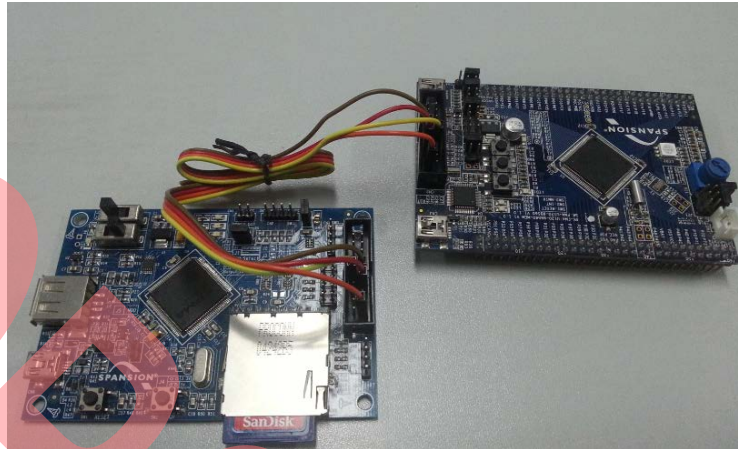
TMS (Universal PGM) ↔ SWDIO (Target Board)

TCK (Universal PGM) ↔ SWDCLK (Target Board)

VCC (Universal PGM) ↔ VCC (Target Board)

The connection is shown in the following figure:

Figure 3-9. SWD Line Connection



The user can also use the standard 20-pin cable as shown in the following figure:

Figure 3-10. 20-pin Cable



4. SD Card insertion

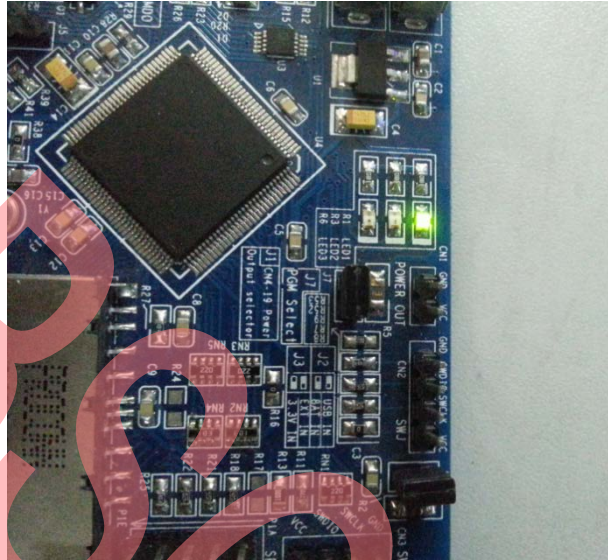
Insert the SD card into the SD socket correctly.

3.1.4 Operation

- SD card detection

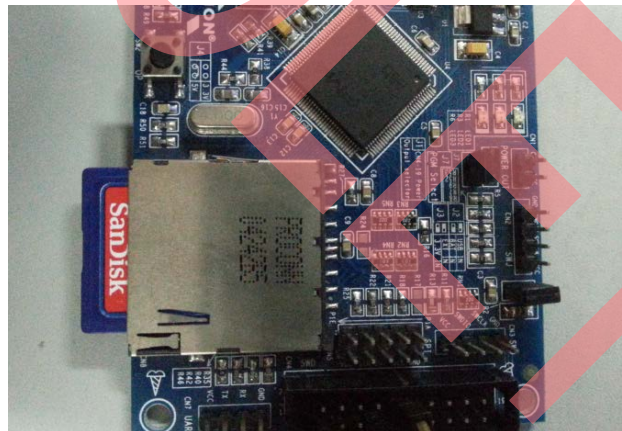
If no SD card is inserted or the target file format is incorrect, the LED1 will be on as shown in the following figure:

Figure 3-11. File Abnormal Error



If the SD card is inserted and the target file is correct, the LED1 is turned off as shown in the figure below:

Figure 3-12. Check File Correct

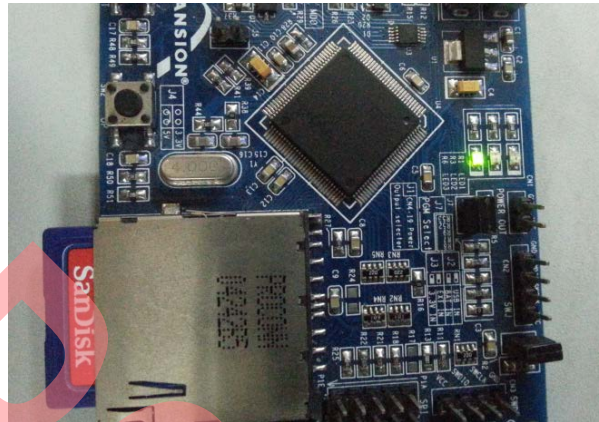


If the LED status is shown as above, the user can press the **Op** key to start the programming of the target board.

3.1.5 Result and Status Check

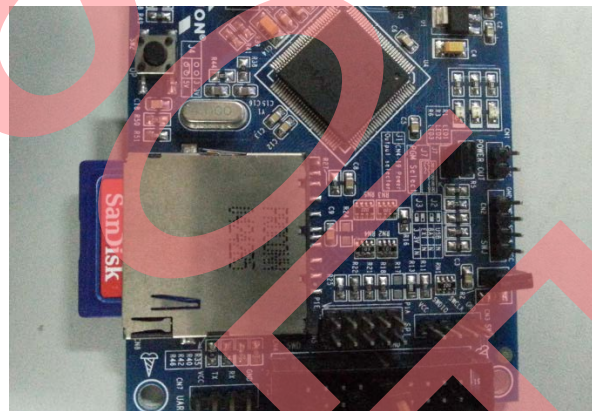
During the programming process, the status of the LED3 is as shown in the following figure (LED3 on):

Figure 3-13. Programming



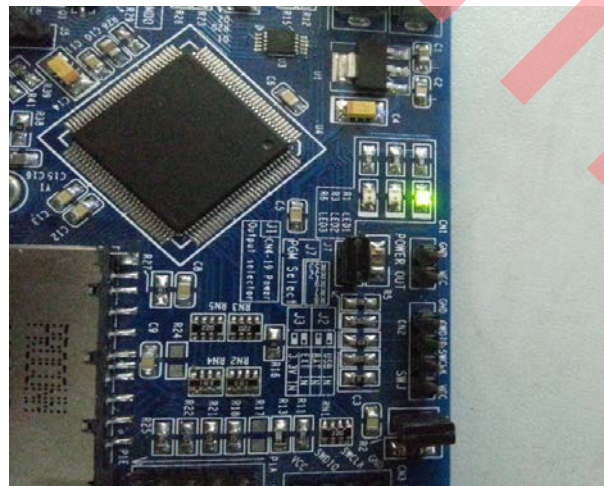
If the program is successful, the LED status is as shown in the following figure (all off):

Figure 3-14. Program OK



Otherwise, the LED status is as shown in the following figure (LED1 on):

Figure 3-15. Program Error



3.2 UART Programming Process

The user can program the internal flash memory of target board through UART communication. The detailed operation process is as follows:

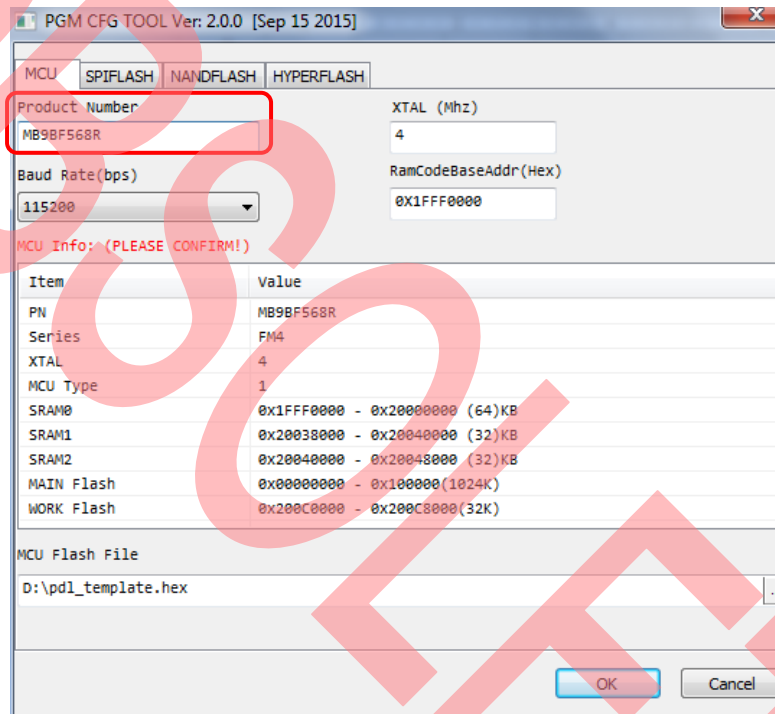
3.2.1 Prepare the hex File

You can get the hex (Intel mode) file through IAR or KEIL, for example, the file name is 'pd1_template.hex'.

3.2.2 Convert and Get the Target Board Files

1. Run the 'hex2bin_Demo.exe'
2. Input the 'Product Number'

Figure 3-16. Product Number



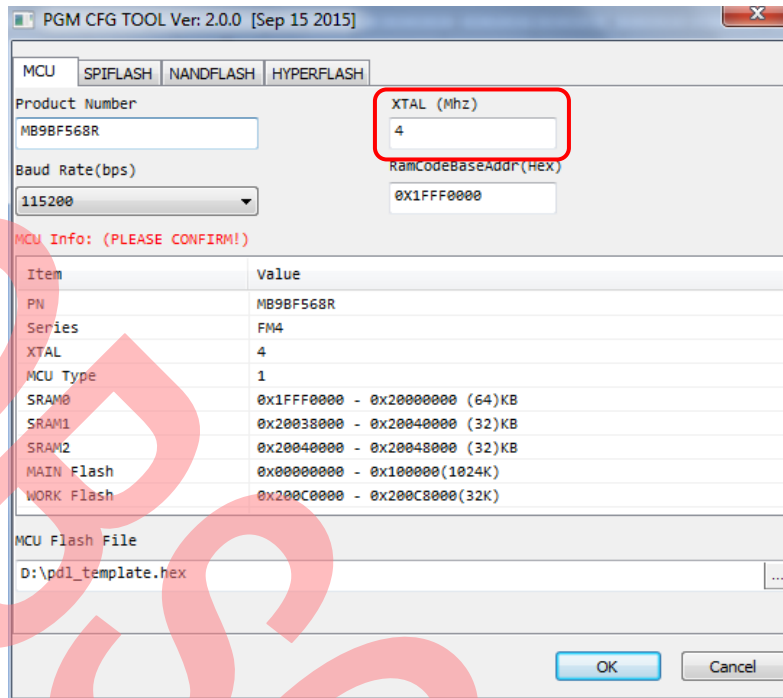
- a. If the user inputs the whole product number properly, the detailed information of the MCU will be displayed below the 'MCU Info:'
- b. If no detailed information is displayed at that place, please input the correct part number of the target MCU. Refer to [Table 2-2](#) of the product number naming rule.

Note: The UART programming does not support external flash RAM code.

3. Set the 'XTAL'

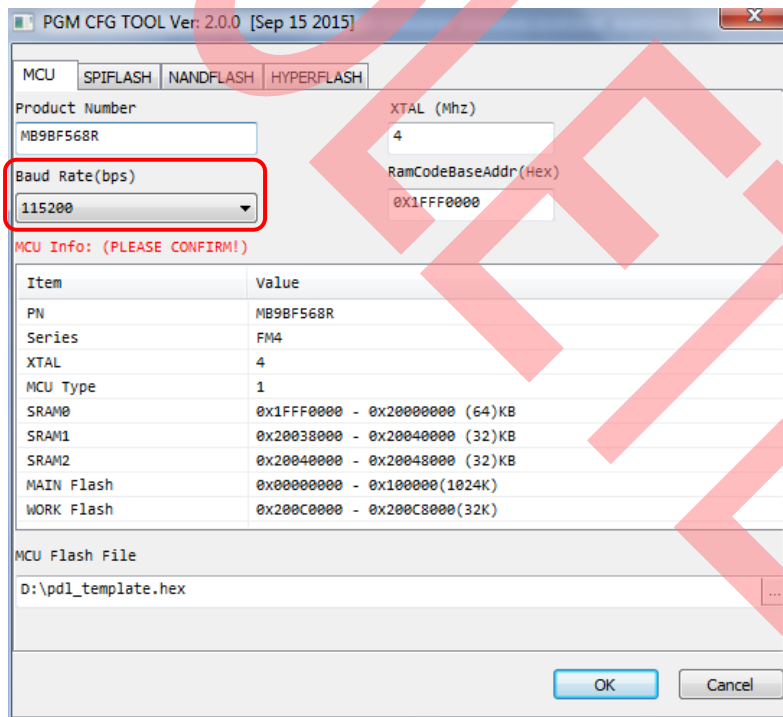
Set the external oscillator number (MHz) in the 'XTAL'

Figure 3-17. Ext. Osc. Setting



4. Select the 'Baud Rate'

Figure 3-18. Baud Rate Selection



Select the baud rate according to the maximum MCU working frequency.

Table 3-1. Baud Rate Setting (Recommendation)

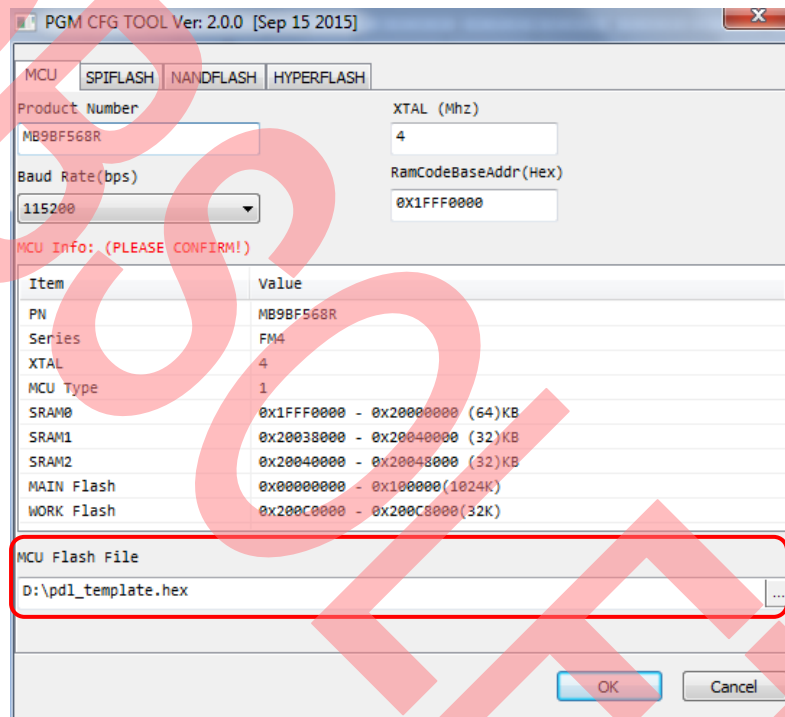
Maximum Freq.	Baud Rate
>= 72MHz	All
>= 40MHz, < 72MHz	256000, 115200
< 40MHz	115200

Note: If the programming failed, please try a lower baud rate.

5. Select the file

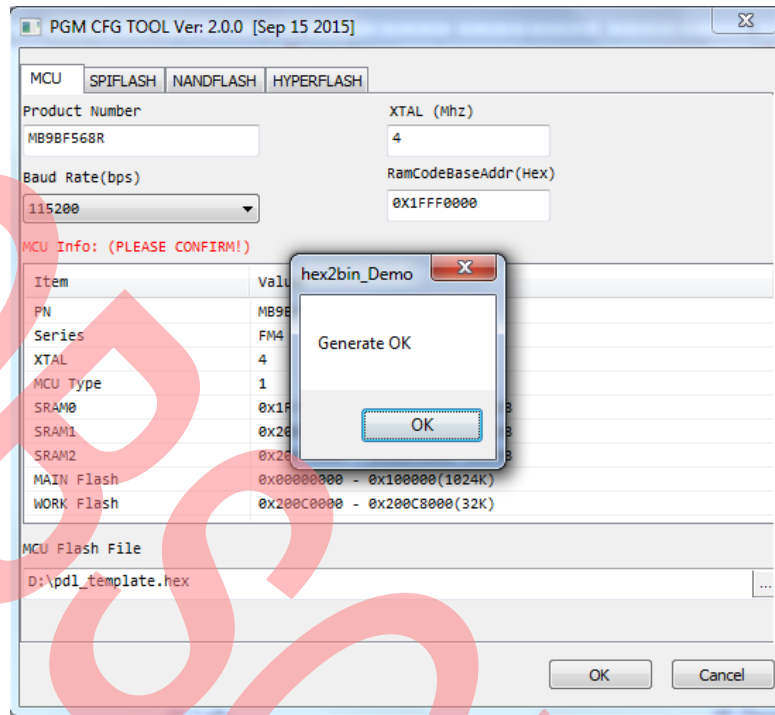
Select the internal flash programming hex file, as show in [Figure 3-19](#).

Figure 3-19. File Selection



6. Generate

Figure 3-20. Generation Result



7. Check the output

'MB9BF568R_MAIN.bin', 'MB9BF568R_WORK.bin' and 'program.ini' are output in the 'PGM_OUTPUT' folder of the tool.

8. Copy these three files into the SD card

3.2.3 Hardware Connection and Setting

1. Power and jumper setting

USB (5V) power input

J1: Close

J2: Left

J3: Left

J4: Open

J5: Open

J7: Open

Battery (4.5V) power input

J1: Close

J2: Right

J3: Left

J4: Open

J5: Open

J7: Open

2. Target Board Mode Setting

Set the target board MCU to serial programming mode.

MD0: High

MD1: Low

3. Communication line (UART) connection

UART (TTL):

VCC (UART) (Universal PGM) \leftrightarrow VCC (Target Board)

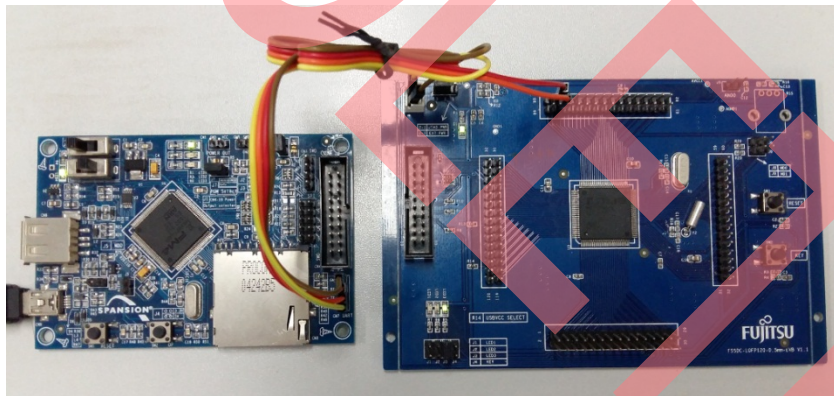
TX (UART) (Universal PGM) \leftrightarrow RX (Target Board)

RX (UART) (Universal PGM) \leftrightarrow TX (Target Board)

GND (UART) (Universal PGM) \leftrightarrow GND (Target Board)

The connection is as shown in the following figure:

Figure 3-21. Line Connection



4. SD Card insertion

Insert the SD card into the SD socket correctly.

3.2.4 Operation

Set P60 of the target board to low if the MCU has the USB function.

The operation is same as that of SWD. See “3.1.4 Operation” for more information.

3.2.5 Result and Status Check

The operation is same as that of SWD. See “3.1.5 Result and Status Check” for more information.

4. Program.ini Introduction



In this part, the detailed meaning of each line of the 'program.ini' will be introduced. Before you read this chapter, you must know the INI file format; if you do not have the knowledge of INI file format, you can learn the knowledge from the [Wikipedia](#) web. Please check the following example:

Figure 4-1. Program.ini Content

```
; Target MCU Configuration information section
[MCU]
Name=MB9BF568R           ; MCU name
Series=FM4               ; MCU Series: FM4/FM3/FM0+
Type=1                   ; MCU type
Xtal=4                   ; Ext. OSC
BaudRate=115200         ; UART baud rate
RamCodeBaseAddr=0x1FFF0000 ; MCU RAM base address

; Target MCU Internal Flash section
[MCUFLASH]
MainStartAddr=0x00000000 ; Main flash operation start address
MainFileName=MB9BF568R_MAIN.bin ; Main flash data file name
MainFileChecksum=0x00000047 ; Main flash data file checksum
SecondStartAddr=0x200C0000 ; Second flash operation start address
SecondFileName=MB9BF568R_WORK.bin ; Second flash data file name
SecondFileChecksum=0x00000000 ; Second flash data file checksum

; Quad SPI Flash section
[SPIFLASH]
Type=S25FL164K           ; SPI flash name
RAMCodeFileName=SPIFLASH_ram.bin ; SPI flash RAM code file name
FilesNumber=5            ; Number of the flash data files
File1StartAddr=0x00000000 ; Start address of flash data file1
File1Name=SPIFLASH_1.bin ; Flash data file1 name
File1Size=0x00001400     ; Flash data file1 size
File1Checksum=0x48C792FD ; Checksum of flash data file1
File2StartAddr=0x00002000 ; Start address of flash data file2
File2Name=SPIFLASH_2.bin ; Flash data file2 name
File2Size=0x00003000     ; Flash data file2 size
File2Checksum=0x5A320355 ; Checksum of flash data file2
File3StartAddr=0x00005000 ; Start address of flash data file3
File3Name=SPIFLASH_3.bin ; Flash data file3 name
File3Size=0x00100000     ; Flash data file3 size
File3Checksum=0xADA56B80 ; Checksum of flash data file3
File4StartAddr=0x00200000 ; Start address of flash data file4
File4Name=SPIFLASH_4.bin ; Flash data file4 name
File4Size=0x00100000     ; Flash data file4 size
File4Checksum=0xB562A680 ; Checksum of flash data file4
File5StartAddr=0x00300000 ; Start address of flash data file5
File5Name=SPIFLASH_5.bin ; Flash data file5 name
File5Size=0x00300200     ; Flash data file5 size
File5Checksum=0xE3038849 ; Checksum of flash data file5

; NAND Flash section
[NANDFLASH]
Type=S34ML01G1           ; NAND flash name
RAMCodeFileName=NANDFLASH_ram.bin ; NAND flash RAM code file name
FilesNumber=5            ; Number of the flash data files
File1StartAddr=0x00000000 ; Start address of flash data file1
File1Name=NANDFLASH_1.bin ; Flash data file1 name
File1Size=0x00020400     ; Flash data file1 size
File1Checksum=0x3E8A539C ; Checksum of flash data file1
File2StartAddr=0x00030000 ; Start address of flash data file2
File2Name=NANDFLASH_2.bin ; Flash data file2 name
File2Size=0x000A0000     ; Flash data file2 size
File2Checksum=0x8C0D38D0 ; Checksum of flash data file2
File3StartAddr=0x00100000 ; Start address of flash data file3
```

```

File3Name=NANDFLASH_3.bin           ; Flash data file3 name
File3Size=0x00100000                ; Flash data file3 size
File3Checksum=0x0F7ECD80            ; Checksum of flash data file3
File4StartAddr=0x00200000           ; Start address of flash data file4
File4Name=NANDFLASH_4.bin           ; Flash data file4 name
File4Size=0x00100000                ; Flash data file4 size
File4Checksum=0x9B787A80            ; Checksum of flash data file4
File5StartAddr=0x00300000           ; Start address of flash data file5
File5Name=NANDFLASH_5.bin           ; Flash data file5 name
File5Size=0x00900200                ; Flash data file5 size
File5Checksum=0x23DF9331            ; Checksum of flash data file5

; Hyper Flash section
[HYPERFLASH]
Type=S26KL512S                       ; Hyper flash name
RAMCodeFileName=HYPERFLASH_ram.bin   ; Hyper flash RAM code file name
FilesNumber=5                          ; Number of the flash data files
File1StartAddr=0x00000000             ; Start address of flash data file1
File1Name=HYPERFLASH_1.bin            ; Flash data file1 name
File1Size=0x00020400                  ; Flash data file1 size
File1Checksum=0x3E8A539C              ; Checksum of flash data file1
File2StartAddr=0x00002000             ; Start address of flash data file2
File2Name=HYPERFLASH_2.bin            ; Flash data file2 name
File2Size=0x000A0000                  ; Flash data file2 size
File2Checksum=0x8C0D38D0              ; Checksum of flash data file2
File3StartAddr=0x00005000             ; Start address of flash data file3
File3Name=HYPERFLASH_3.bin            ; Flash data file3 name
File3Size=0x00100000                  ; Flash data file3 size
File3Checksum=0x0F7ECD80              ; Checksum of flash data file3
File4StartAddr=0x00200000             ; Start address of flash data file4
File4Name=HYPERFLASH_4.bin            ; Flash data file4 name
File4Size=0x00100000                  ; Flash data file4 size
File4Checksum=0x9B787A80              ; Checksum of flash data file4
File5StartAddr=0x00300000             ; Start address of flash data file5
File5Name=HYPERFLASH_5.bin            ; Flash data file5 name
File5Size=0x00900200                  ; Flash data file5 size
File5Checksum=0x23DF9331              ; Checksum of flash data file5

```

Notes:

- The program.ini file includes section and keys, every key has a name and a value; Sections include 'MCU', 'MCUFLASH', 'SPIFLASH', 'HYPERFLASH', 'NANDFLASH'; the program.ini file must include 'MCU' section.
- If only program the MCU internal flash, the file only includes 'MCU' and 'MCUFLASH' section.
- If only program the external flash, the file do not include MCU flash section.
- If program the MCU internal flash and external flash, the program.ini file must include MCU flash section and one or several external flash sections.
- Maximum support 5 external flash data files.

4.1 MCU Section

- Name: Please refer to [Table 2-2](#) for product number naming rule, check the detailed information in the corresponding datasheet.
- Series: FM3/FM4/FM0+
- Type: MCU internal flash type
 - For FM3, it is 0 – 12
 - For FM4, it is 1 - 6
 - For FM0+, it is 1 – 3
- Xtal: Set the MHz number of external oscillator of the main board
- BaudRate: Set the communication baud rate of the UART if UART is used to program
- RamCodeBaseAddr: Set the RAM code address of the external flash programming

4.2 MCU Flash Section

- MainStartAddr: The file address of the file stored at the main flash
- MainFileName: The file name stored at the main flash
- MainFileChecksum: The checksum of the file stored at the main flash
- SecondStartAddr: The file address of the file stored at the MCU second flash memory area.
- SecondFileName: The file name stored at the MCU second flash memory area
- SecondFileChecksum: The checksum of the file stored at the MCU second flash memory area

4.3 External Flash Section

- Type: The name of the external flash memory
- RAMCodeFileName: The external flash programming RAM code file name
- FilesNumber: The number of the external flash data files
- FilexStartAddr: The flash data file start address of the file stored at the external flash memory (x: file number; value: 1 - 5)
- FilexName: The flash data file name stored at the external flash memory (x: file number, value: 1 - 5)
- FilexSize: The flash data file size of the file stored at the external flash memory (x: file number; value: 1 - 5)
- FilexChecksum: The checksum of the file stored at the external flash memory (x: file number; value: 1 - 5)

5. Maintenance



The code of the programmer can be updated by SD card. The operation method is as follows:

(Due to the source code will not be provided to the users, these steps should be performed by internal engineers)

1. Build '(FWSC)FMx_Universal_PGM' project.
2. Make sure the bin file name is 'FM4APP.BIN'
3. Copy the bin file to the root directory of the SD Card
4. Insert the card into the slot and power on
5. The 'Work' LED (LED3) is on while updating the code to the programmer
6. If the 'Work' LED (LED3) is off, all LEDs flash 2 times, the update is finished
7. While using the SD card to program the target board, please make sure the 'FM4APP.BIN' is not in the card to prevent auto-update of the programmer

The error code of the operation process can be logged in the SD card. If users encounter any problems during the operation, solutions/countermeasures are provided:

1. Provide the debug version of the programmer with error code log function
2. You can update the debug version code into the programmer through the above method
3. You can operate the programmer to program the target board
4. If any error occurs, it will be logged in the SD card
5. You can send the error code (error.log file and system.log file) back to Cypress for analysis

Notes:

- In this application, SD Card with standard size is recommended for stable and better performance.
- When the error logs are processed, the programmer will write data into the SD card. Ensure no other important data is saved in the SD card.

6. Additional Information



For more information on Cypress FMx Family MCU, visit our website:

<http://www.cypress.com/products/32-bit-arm-cortex-microcontroller-mcu-families>

Please contact your local support team for any technical question.

Revision History



Document Revision History

Document Title: FM3/FM4/FM0+ Family Universal Programmer User Manual				
Document Number: 002-09180				
Revision	ECN	Issue Date	Origin of Change	Description of Change
**	-	10/31/2014	CPQI	Initial release
		07/21/2015	HUAL	Updates
*A	5622849	02/22/2017	HUAL	Migrated Spansion User Manual "AN706-00094-2v0-E" to Cypress format. Document Obsoleted.