Bootloader utility for MC56F83xxx

The MC56F83xxx is the latest DSC family, it integrates new features such as CAN-FD, USB, enhanced DMA, can be used in motor control, switch mode power supply applications.
The MC56F83xxx runs code in on-chip flash, so the Bootloader's main task is to provision the internal flash memory with an embedded applicable firmware image during manufacturing, or at any time during the life of the device. The Bootloader does the provisioning by acting as a slave device, and listening to various peripheral ports where a master can start communication.

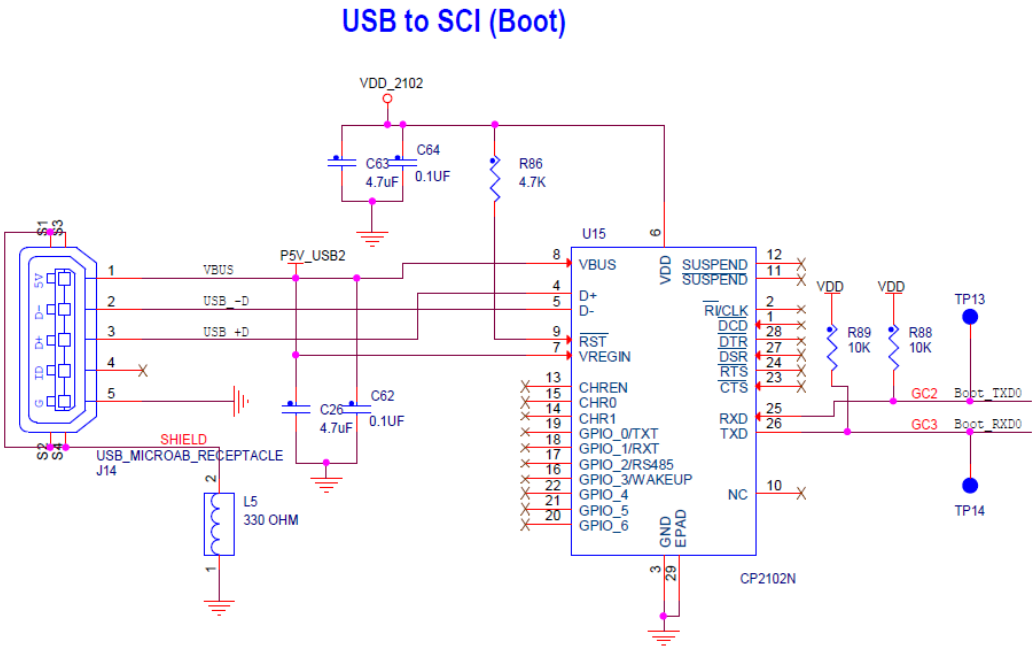
1)Bootloader hardware connection

The MC56F83xxx has on-chip ROM bootloader, which is same as in Kinetis and LPC family, because the bootloader code is saved in ROM, the bootloader can not be modified. It uses the following fixed peripherals to download application code. If customer designs target board, the following pins must be used if customer wants to use the on-chip bootloader.

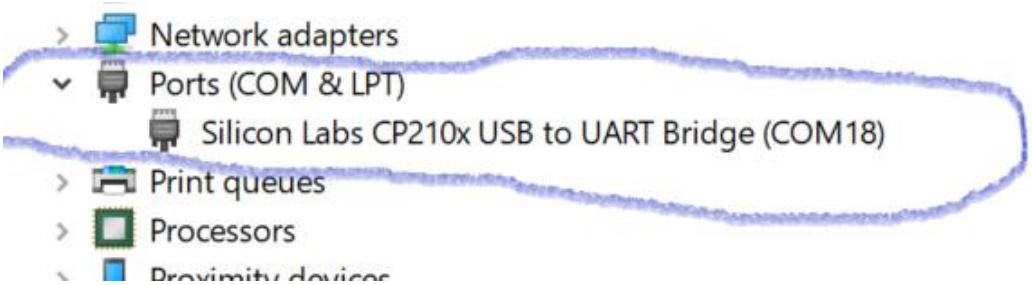
Table 6-1. Bootloader Peripheral Pinmux

Peripheral	Instance	Alt Mode	Pins
UART/SCI	0	0	GPIOC2, QSCI0_TXD
		2	GPIOC3, QSCI0_RXD
I2C	0	0	GPIOC14, I2C0_SDA
			GPIOC15, I2C0_SCL
CAN	0	0	GPIO11, CAN Tx
			GPIO12, CAN Rx

On the MC56F83000-EVK board, the QSCI0 signals GPIOC2(QSCI0_TXD) and GPIOC3(QSCI0_RXD) are connected to the CP2102, in order to download application code via Bootloader, user has to connect J14 USB port to PC.



After the J14 USB is connected, the USB port is enumerated as virtual serial port as the following Fig in Computer Device Manager.



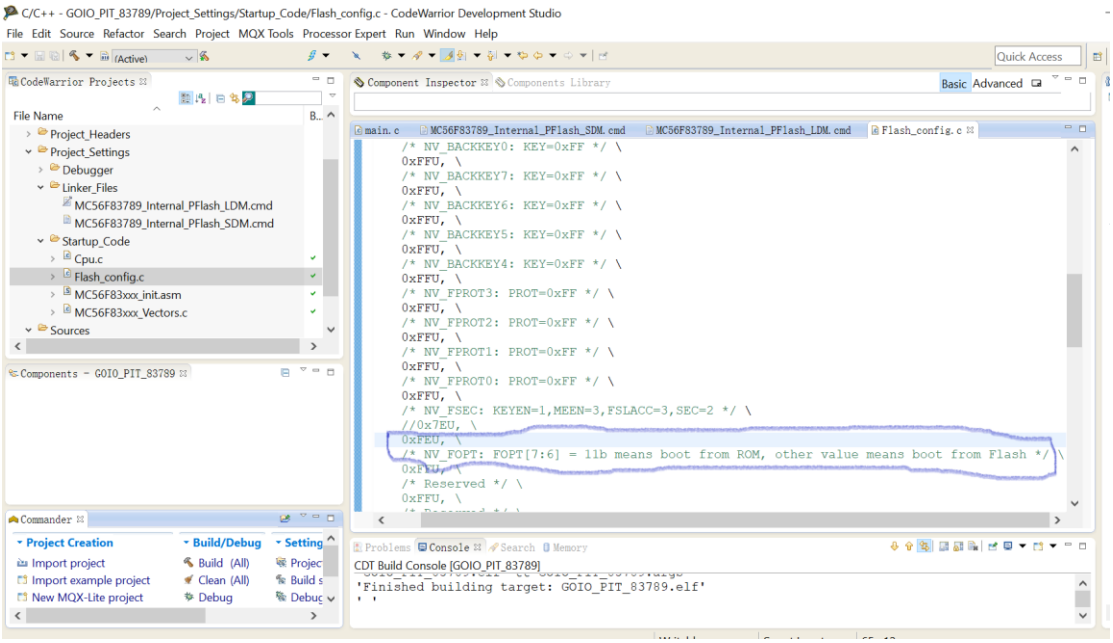
2)BCA and Flash configuration field configuration

There is a special space in on-chip flash called BCA, located from P:0x3C0, it provides all of the parameters needed to configure the Bootloader operation. For uninitialized flash, the Bootloader uses a predefined default configuration. A host application can use the Bootloader to program the BCA for use during subsequent initializations of the bootloader.

Flash configuration Field description

There is a special space called Flash configuration field located at P:0x400~0x40F, the Bit7 and 6 of FOPT location (p:0x40D) has to be set so that the bootloader can be executed at start-up after Reset. For detailed inf, pls refer to section 20.3.1 Flash configuration field description

For setting up the BCA configuration and Flash configuration field, customer just needs to modify the Flash_config.c which is created by CodeWarrior for mcu tools automatically.

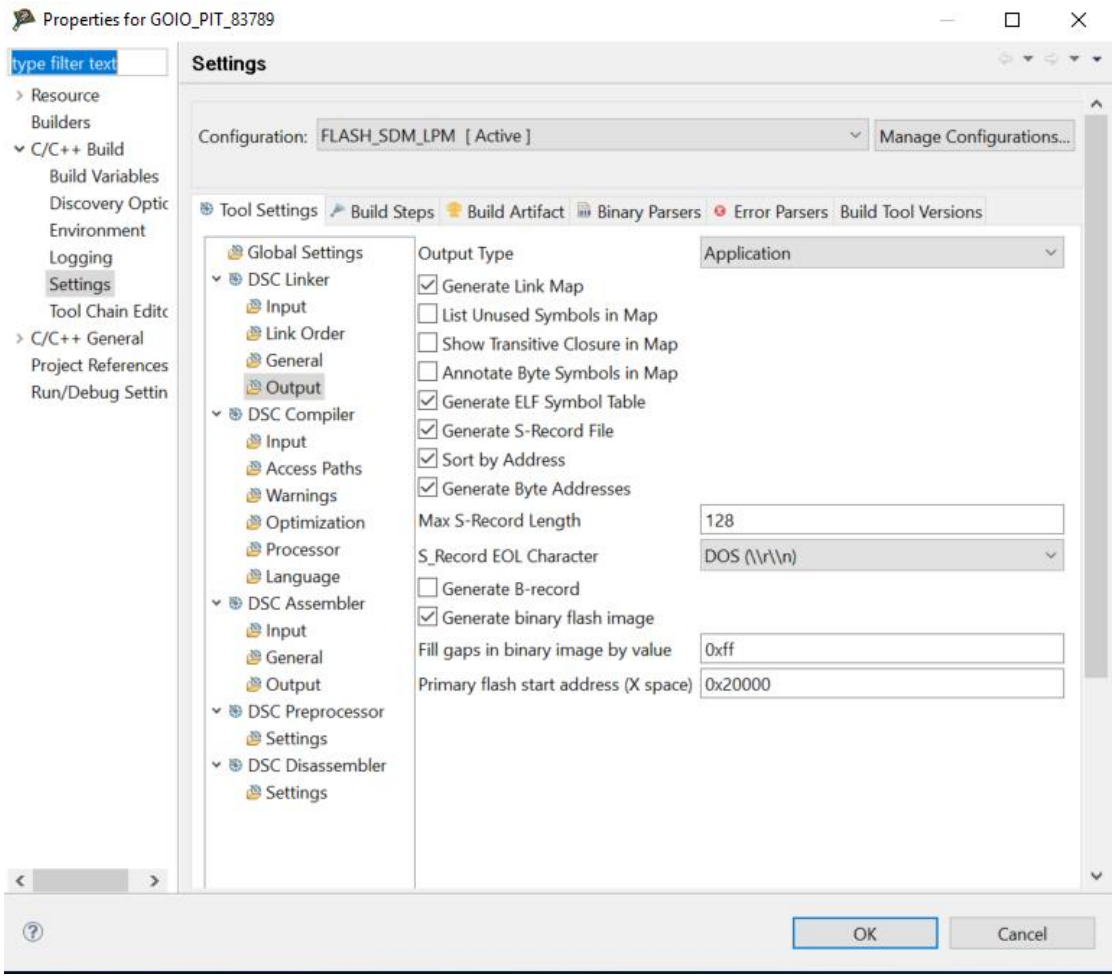


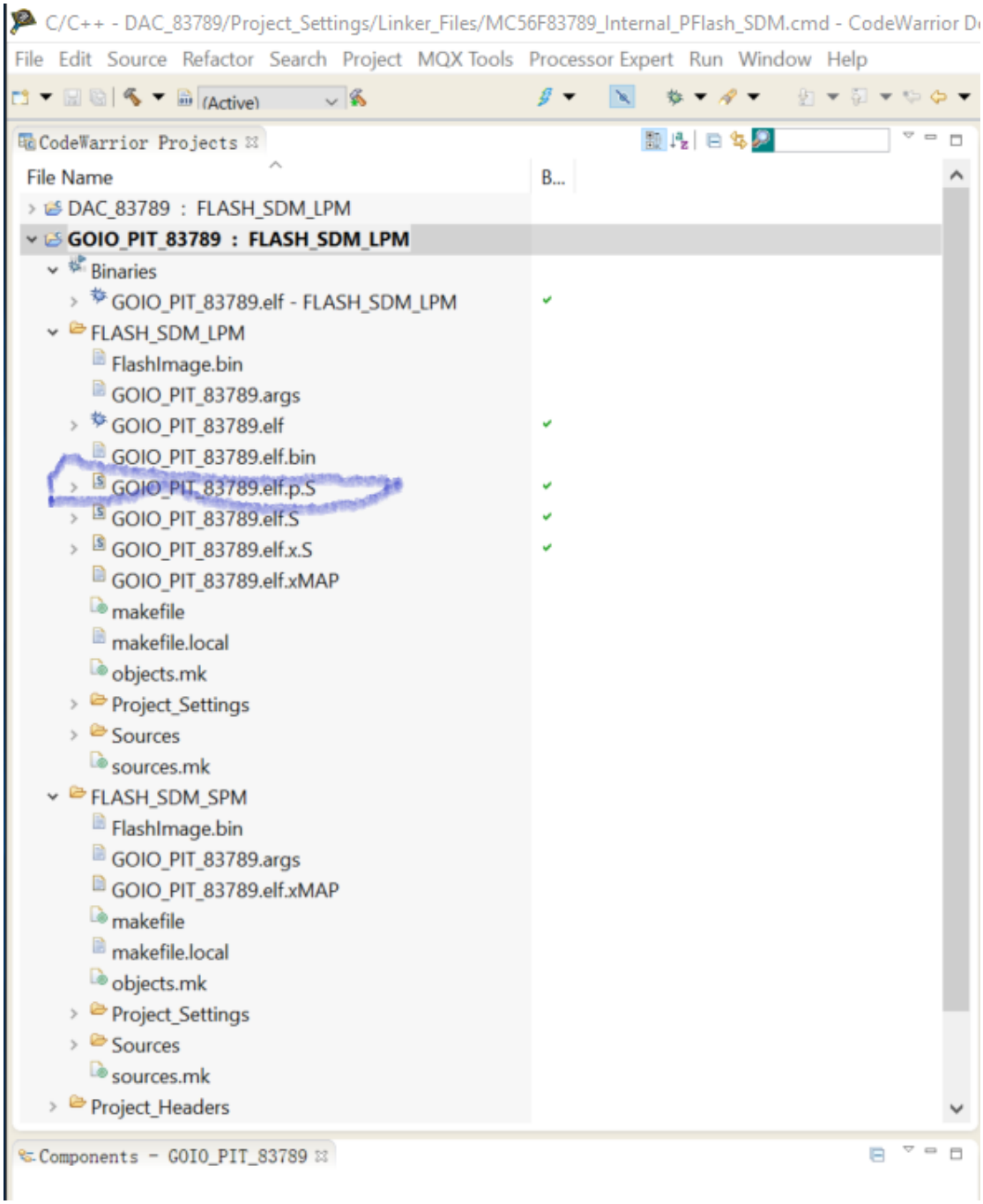
3)**software configuration for generating S-Record file.**

Generating S-Record file based on CodeWarrior for mcu ver11.x

The bootloader can download application code to flash with the S-Record format file. As following fig, the S-Record file can be generated by the CoideWarrior for MCU tools so that the blhost can download the S-Record file to flash.

Note that the boes of Generate S-Record File/Sort by Address/Generate Byte Address must be checked. The Max S-Record length is set up as 128 and DOS mode has to be selected as the following fig.

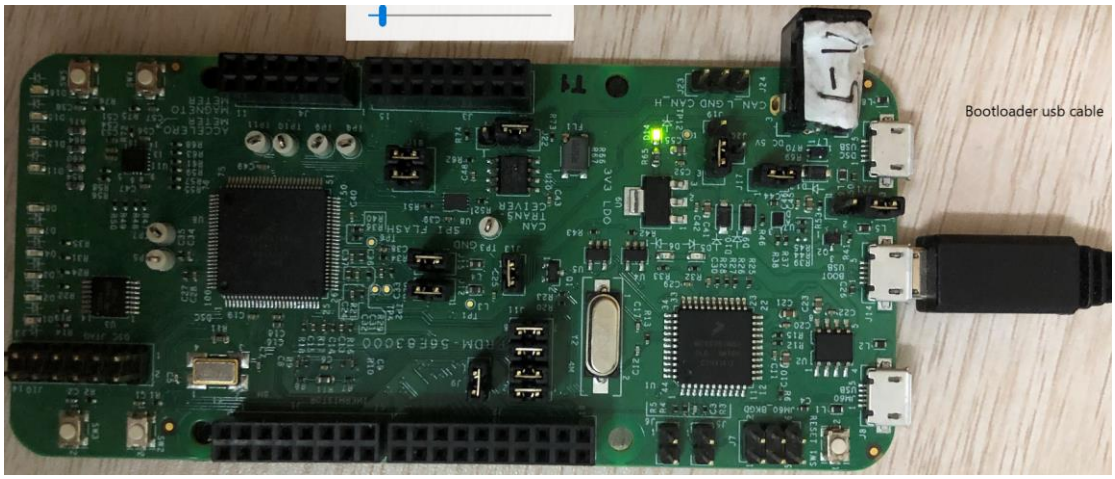

Properties for GOIO_PIT_83789
type filter text
Resource
Builders
C/C++ Build
Build Variables
Discovery Optic
Environment
Logging
Settings
Tool Chain Editc
C/C++ General
Project References
Run/Debug Settin
Settings
Configuration: FLASH_SDM_LPM [Active]
Manage Configurations...
Tool Settings
Build Steps
Build Artifact
Binary Parsers
Error Parsers
Build Tool Versions
Global Settings
DSC Linker
Input
Link Order
General
Output
DSC Compiler
Input
Access Paths
Warnings
Optimization
Processor
Language
DSC Assembler
Input
General
Output
DSC Preprocessor
Settings
DSC Disassembler
Settings
Output Type
Application
Generate Link Map
List Unused Symbols in Map
Show Transitive Closure in Map
Annotate Byte Symbols in Map
Generate ELF Symbol Table
Generate S-Record File
Sort by Address
Generate Byte Addresses
Max S-Record Length
128
S_Record EOL Character
DOS (\\r\\n)
Generate B-record
Generate binary flash image
Fill gaps in binary image by value
0xff
Primary flash start address (X space)
0x20000
OK
Cancel



Note that the *.elf.p.S file can only be downloaded by blhost tools instead of the *.elf.S. Restriction: ROM bootloader puts its global variable into RAM from x:0x0000 and has no protection for them. If we use ".S" or ".x.S" files, it may corrupt ROM bootloader's variables. So let's use ".p.S" file with ROM.

4)**Bootloader hardware connection**

It is easy to use bootloader, just connect the J14 USB slot to PC and jump the pin2&3 of J19 so that the board is powered by the J14 usb cable.



5)**Blhost commands**

Blhost -p com19 get-property 1

Blhost -p com19 flash-erase-all-unsecure //mass erase and set the DSC in unsecure mode by setting the P:0x40C byte in BYTE address mode

Blhost -p com19 flash-image *.s19 erase

```
C:\DriveF\New Tools\mcu-boot\bin\Tools\blhost\win>blhost -p com32 get-property 1
Ping responded in 1 attempt(s)
Inject command 'get-property'
Response status = 0 (0x0) Success.
Response word 1 = 1258423808 (0x4b020600)
Current Version = K2.6.0
```

```
C:\DriveF\New Tools\mcu-boot\bin\Tools\blhost\win>blhost -p com32 flash-erase-all-unsecure
Error: Initial ping failure: No response received for ping command.
C:\DriveF\New Tools\mcu-boot\bin\Tools\blhost\win>blhost -p com32 flash-erase-all-unsecure
Ping responded in 1 attempt(s)
Inject command 'flash-erase-all-unsecure'
Successful generic response to command 'flash-erase-all-unsecure'
Response status = 0 (0x0) Success.
```

```
C:\DriveF\New Tools\mcu-boot\bin\Tools\blhost\win>blhost -p com32 flash-image T83789P.S19 erase
Ping responded in 1 attempt(s)
Inject command 'flash-image'
Successful generic response to command 'flash-erase-region'
Successful generic response to command 'flash-erase-region'
Successful generic response to command 'flash-erase-region'
Wrote 448 bytes to address 0
Successful generic response to command 'write-memory'
(1/3)100% Completed!
Successful generic response to command 'write-memory'
Wrote 76 bytes to address 0x3c0
Successful generic response to command 'write-memory'
(2/3)100% Completed!
Successful generic response to command 'write-memory'
Wrote 888 bytes to address 0x410
Successful generic response to command 'write-memory'
(3/3)100% Completed!
Successful generic response to command 'write-memory'
Response status = 0 (0x0) Success.
```

Refer to BLHOST guide for more details: https://www.nxp.com/docs/en/user-guide/MCUBLHOSTUG.pdf