SofTec Microsystems
AN00137: Interfacing FlashRunner with Freescale S08 Devices

by Paolo Dal Ben (paolo.dalben@softecmicro.com)

FlashRunner is a Universal In-System Programmer which uses the principles of In-System Programming (ISP) to program Freescale S08 microcontrollers. This application note describes how to properly set up and use FlashRunner to program S08 Flash devices.

This Application Note assumes that you are familiar with FlashRunner and with the main features of the S08 family. Full documentation about these topics is available in the FlashRunner user's manual and in device-specific datasheets.

1. Introduction

S08 family devices can be divided into four subfamilies based on the internal clock source:

- ICG: Internal Clock Generator (e.g. MC9S08AW, MC9S08GB, MC9S08GT, MC9S08LC);
- ICS: Internal Clock Source (e.g. MC9S08EL, MC9S08QD, MC9S08QG, MC9S08SG);
- MCG: Multi-purpose Clock Generator (e.g. MC9S08DN, MC9S08DV, MC9S08DZ, MC9S08EN);
- No internal clock source (e.g. MC9S08RC, MC9S08RD, MC9S08RE, MC9S08RG).

Revision 1.0 – April 2007

SofTec Microsystems is a Freescale Alliance Member

freescale™ Alliance Member

DC01306

SofTec Microsystems

E-mail (general information): info@softecmicro.com

E-mail (technical support): support@softecmicro.com

Web: http://www.softecmicro.com

It is important to know the internal clock source of your target device in order to properly create a FlashRunner script. For more information, please refer the specific device datasheet.

In-system programming of S08 microcontrollers is performed by entering the device's active background mode, which allows the programming of the MCU memory through a single-wire interface. In-system communication with S08 devices is performed through a so-called BDM interface.

To use FlashRunner to perform in-system programming, you need to implement the BDM hardware interface on your application board.

2. Hardware Configuration

The microcontroller lines needed to implement the BDM interface and program an S08 device are the following:

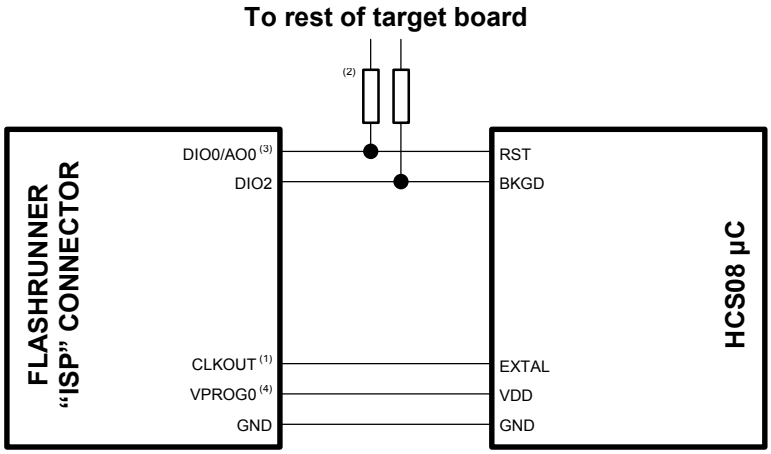
- **BKGD:** Single-wire background interface pin. The BKGD pin is used for bidirectional communication of active background mode commands and data between FlashRunner and the target MCU.

  This pin is also used to request a timed sync response pulse to allow FlashRunner to determine the correct clock frequency for background debug serial communications.
- **VDD:** Device power supply voltage. The range of operating voltage is typically 2.7–5.0V, depending on the MCU. This line must be connected to FlashRunner when you use **`SW_RST`** or **`PWUP_RST`** BDM entry modes (see later).
- **RST:** Device reset input/output pin. This line must be connected to FlashRunner when you use **`HW_RST`** BDM entry mode.
- **GND:** Device power supply ground.

If you want FlashRunner to clock your target device, you must also connect FlashRunner's CLKOUT pin to the EXTAL pin of your target device.

Freescale has defined a standard 6-pin connector (BDM) that allows an interface pod to be connected to any target S08 Family MCU.

The lines mentioned above must be connected to the FlashRunner's "ISP" connector according to the following diagram:



Notes

(1) Connect this line if you want the target device to be clocked by FlashRunner. Please note that the voltage level of FlashRunner CLKOUT signal is that specified by the VDD parameter (see later). This could be an issue for certain devices for which the clock signal has a voltage level different than the supply voltage's. In this case you cannot use FlashRunner to clock the target device.

(2) If the lines needed to enter monitor mode are used for other purposes in the application, series resistors should be implemented to avoid a conflict in case the rest of the target board forces the signal level. If these lines are used as outputs, these resistors are not necessary.

(3) It is not necessary to connect this line if you choose to enter BDM mode by software reset or power up reset.

(4) It is not necessary to connect this line if you choose to enter BDM mode by hardware reset and FlashRunner doesn't power the target board.

3. S08-Specific TCSETPAR Programming Commands

Overview

`TCSETPAR` commands set device-specific and programming algorithm-specific parameters. These commands must be sent after the **`TCSETDEV`** command and before a **`TPSTART`** / **`TPEND`** command block.

All of the FlashRunner programming capabilities rely on the background monitor mode of the target device. In order to enter this special mode (which establishes a communication channel between the target device and FlashRunner) and configure it properly, all of the following parameters must be correctly specified through the relative **`TCSETPAR`** commands (although the order with which these parameters are set is not important):

- $V_{DD;}$
- Power up time;
- Power down time;
- Reset up time;
- Reset down time;
- BDM entry mode;

- FLL frequency (only for devices with an ICG module).
- CLKOUT frequency (only when using FlashRunner to clock the target device).

TCSETPAR VDD

Command syntax:

`TCSETPAR VDD <voltage mV>`

Parameters:

`voltage mV`: Target device supply voltage, expressed in millivolts.

Description:

This command is used to properly generate the voltage level of the ISP signals. Additionally, the specified voltage is routed to the VPROG0 line of the FlashRunner “ISP” connector, which can be used as a supply voltage for the target board.

TCSETPAR PWUP

Command syntax:

`TCSETPAR PWUP <time ms>`

Parameters:

`time ms`: Power rising time, expressed in milliseconds.

Description:

This command is necessary because, to enter the monitor mode, FlashRunner must properly drive the V_{DD} line during the power-on reset.

The V_{DD} rising time (PWUP) is expressed in milliseconds and depends on the features of your target board. Make sure to choose a value large enough to ensure that the V_{DD} signal reaches the high logic level within the specified time. Note that, if the V_{DD} line has a high load, a longer time is required for the V_{DD} signal to reach the high logic level. If PWUP is not long enough, FlashRunner could not be able to enter the monitor mode.

TCSETPAR PWDOWN

Command syntax:

`TCSETPAR PWDOWN <time ms>`

Parameters:

`time ms`: Power falling time, expressed in milliseconds.

Description:

The V_{DD} falling time (PWDOWN) is expressed in milliseconds and depends on the features of your target board. Make sure to choose a value large enough to ensure that the V_{DD} signal reaches the low logic level within the specified time. Note that, if the V_{DD} line has a high load, a longer time is required for the V_{DD} signal to reach the low logic level.

TCSETPAR RSTUP

Command syntax:

TCSETPAR RSTUP <time µs>

Parameters:

time µs: Reset rising time, expressed in microseconds.

Description:

The Reset rising time (RSTUP) is expressed in microseconds and depends on the features of your target board. Make sure to choose a value large enough to ensure that the Reset signal reaches the high logic level within the specified time. Note that, if the Reset line has a high load, a longer time is required for the Reset signal to reach the high logic level. If RSTUP is not long enough, FlashRunner could not be able to enter the background monitor mode.

TCSETPAR RSTDOWN

Command syntax:

TCSETPAR RSTDOWN <time µs>

Parameters:

time µs: Reset falling time, expressed in microseconds.

Description:

The Reset rising time (RSTDOWN) is expressed in microseconds and depends on the features of your target board. Make sure to choose a value large enough to ensure that the Reset signal reaches the low logic level within the specified time. Note that, if the Reset line has a high load, a longer time is required for the Reset signal to reach the low logic level. If RSTDOWN is not long enough, FlashRunner could not be able to enter the background monitor mode.

TCSETPAR BDM_ENTRY_MODE

Command syntax:

TCSETPAR BDM_ENTRY_MODE HW_RST|SW_RST|PWUP_RST

Command options:

HW_RST: enters background monitor mode by hardware reset.

SW_RST: enters background monitor mode by software reset.

PWUP_RST: enters background monitor mode by power-up reset.

Description:

Specifies the BDM entry mode.

Choose **HW_RST** to enter background mode by hardware reset. In this case, in order to enter background mode, FlashRunner pulls the Reset line low. The target device's RST signal must be connected to FlashRunner's DIO0/AO0 signal. It is not necessary to connect the VDD signal. Note that you cannot enter BDM mode by hardware reset for devices such as MC9S08QD4 where the RST signal can only be used to reset into user mode.

Choose **PWUP_RST** to enter background mode by power-up reset. In this case, in order to enter background mode, FlashRunner powers off and on the target device. The target device's VDD signal must be connected to FlashRunner's VPROG0 signal. It is not necessary to connect the RST signal.

Choose **SW_RST** to enter background mode by software reset. The target device's VDD signal must be connected to FlashRunner's VPROG0 signal. It is not necessary to connect the RST signal.

TCSETPAR FLL_OSC

Command syntax:

TCSETPAR FLL_OSC <frequency Hz>

Parameters:

frequency Hz: FLL frequency ($f_{ICGDCLK}$), expressed in Hertz.

Description:

This command is needed only for target devices featuring an ICG module. It specifies the $f_{ICGDCLK}$ value to be used during the programming process. This value is twice the bus frequency.

Please note that programming time can be reduced by driving the internal FLL circuitry to the maximum allowed frequency (typically 40 MHz). Make sure that the value you set for the **frequency Hz** parameter is included in the DCO clock frequency range, specified in the specific device datasheet. Setting a frequency of 0 for the **frequency Hz** parameter will disable the FLL.

TCSETPAR CLKOUT

Command syntax:

CSETPAR CLKOUT 25000000|12500000|6250000|0

Command options:

Frequency of the clock signal to be generated at the CLKOUT pin of the FlashRunner “ISP” connector, expressed in Hertz. The available clock frequency values are 25MHz, 12.5MHz and 6.25MHz.

Description:

Generates an auxiliary clock signal at the CLKOUT pin of the FlashRunner “ISP” connector. This signal can be used as an auxiliary clock source, and is particularly useful when the target microcontroller requires an external clock which is not otherwise available on the target board.

Furthermore, this signal can be used to speed up programming (when you want to use a clock faster than that provided by your target board).

Make sure that the clock frequency you select is not greater than the maximum allowed frequency for your device. Note that, since all devices feature an internal frequency divisor, the actual bus frequency will be a fraction of the CLKOUT frequency.

If you specify `0` as the CLKOUT frequency, no clock signal is generated.

This command is mandatory only if you use FlashRunner’s CLKOUT pin as clock source.

4. S08-Specific TPCMD Programming Commands

Overview

`TPCMD` commands perform a programming operation (i.e. mass erase, program, verify, etc.) These command must be sent within a **`TPSTART`** / **`TPEND`** command block.

The S08-specific target programming commands are the following:

- **`TPCMD IS_DEVICE_SECURED`**;
- **`TPCMD BLANKCHECK`**;
- **`TPCMD MASSERASE`**;
- **`TPCMD TRIM`**;
- **`TPCMD PROGRAM`**;
- **`TPCMD VERIFY`**;
- **`TPCMD RUN`**.

TPCMD IS_DEVICE_SECURED

Command syntax:

`TPCMD IS_DEVICE_SECURED`

Command parameters:

None.

Description:

Returns whether the device is secured (**1**) or not (**0**).

TPCMD BLANKCHECK

Command syntax:

TPCMD BLANKCHECK F|E <tgt start addr> <len>

Command options and parameters:

F|E: Memory type to be blankchecked. **F** stands for Flash memory and **E** stands for EEPROM memory.

tgt start address: Device memory location from where the blankcheck operation will start.

len: Number of bytes to be blankchecked.

Description:

Blankchecks Flash or EEPROM memory. Blankchecks **len** bytes starting from the address specified by **tgt start address**.

Please note that, if the Flash memory is divided into two or more zones, you will have to send this command for each zone.

TPCMD MASSERASE

Command syntax:

TPCMD MASSERASE F|E

Command options:

F|E: Memory type to be mass erased. **F** stands for Flash memory and **E** stands for EEPROM memory.

Description:

Mass erases Flash or EEPROM memory.

Please note that, after a device is mass erased, the trimming value is lost. Thus, if your application uses the device's internal clock, it is suggested that, after a mass erase command, you trim the device's internal clock.

TPCMD PROGRAM

Command syntax:

TPCMD PROGRAM F|E <src offset> <tgt start addr> <len>

Command options and parameters:

F|E: Memory type to be programmed. **F** stands for Flash memory and **E** stands for EEPROM memory.

src offset: Offset from the beginning of the .FRB source file.

tgt start addr: Device memory location from where the program operation will start.

len: Number of bytes to be programmed.

Description:

Programs **len** bytes in the Flash or EEPROM memory starting from the **tgt start addr** address. If the memory range includes the trimming location specified by the **TPCMD TRIM** command, this location will be programmed with the calculated trimming value (see the description of the **TPCMD TRIM** command for more info about this topic).

Please note that, if the Flash memory is divided into two or more zones, you will have to send this command for each zone.

TPCMD TRIM

Command syntax:

TPCMD TRIM <frequency Hz> <addr> <tolerance_pct>

Command parameters:

frequency Hz: Value of the desired frequency, expressed in Hertz.

addr: Address of the memory location where the trimming value is to be programmed.

tolerance_pct: Precision of trimming value calculation, expressed as a percentage (1 to 100).

Description:

This command trims (calibrates) the device's internal oscillator, if present. It calculates the trimming value for the frequency specified by the **frequency Hz** parameter (to a precision specified by the **tolerance_pct** parameter), and prepares to program it at the **addr** address.

For devices featuring the ICG module, the frequency to be trimmed is the internal reference frequency. For these devices, Freescale suggests to use a **frequency Hz** value of 243000.

For devices features the ICS or MCG module, FlashRunner considers the DCO output frequency (which is twice the bus frequency) as the reference frequency. Make sure that the value you specify for the **frequency Hz** parameter is included in the DCO output frequency range. Alternatively, you can find a suitable value for the **frequency Hz** parameter in the script examples included in the CD-ROM you received along with FlashRunner.

Please note that this command does not program the calculated value in the Flash memory: the actual programming will occur during the next **TPCMD PROGRAM** command involving the trimming location.

TPCMD VERIFY

Command syntax:

TPCMD VERIFY F|E R <src offset> <tgt start addr> <len>

Command options and parameters:

F|E: Memory type to be verified. **F** stands for Flash memory and **E** stands for EEPROM memory.

R: Specifies the verifying method. **R** (the only available option) stands for a complete read-out of all written data.

src offset: Offset from the beginning of the .FRB source file.

tgt start addr: Device memory location from which the verify operation will start.

len: Number of bytes to be verified.

Description:

Verifies that the contents of the target device memory correspond to that of the source image file.

Please note that, if the Flash memory is divided into two or more zones, you will have to send this command for each zone.

Please also note that, if the program operation has secured the device memory, the verify operation should be performed within the same **TPSTART** / **TPEND** command block.

TPCMD RUN

Command syntax:

TPCMD RUN

Command parameters:

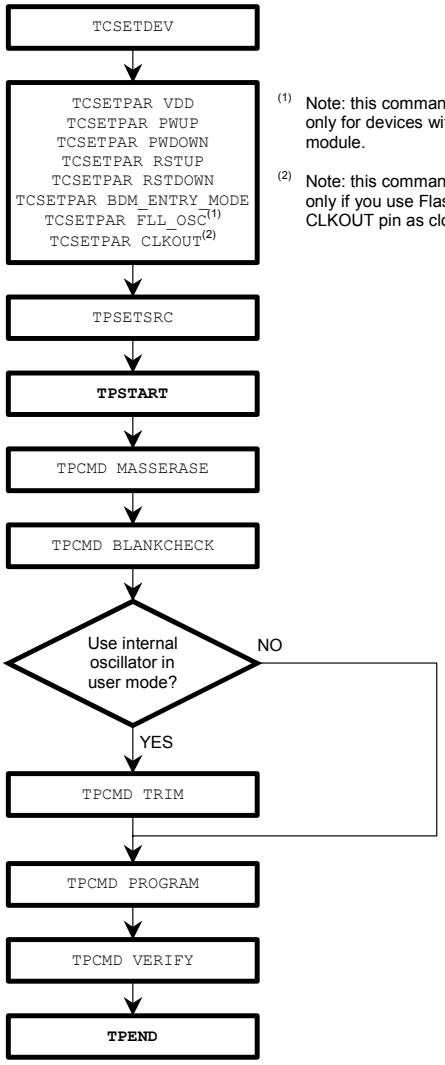
None.

Description:

Runs the target application.

5. Typical Programming Flow

The following flow chart illustrates typical steps to help you write your own script file.



6. Script Example

The example provided below will help you understand how the commands discussed above should be used for a typical S08 device, in this case the MC9S08AW60.

```
;
; FLASHRUNNER SCRIPT EXAMPLE FOR FREESCALE MC9S08AW60
;
; Use this example as a starting point for your specific programming needs
;
; ----------
;
; Hardware connections
;
; DIO0/AO0    (RESET)
; DIO2        (BKGD)
; CLKOUT      (CLOCK - optional)
;
; Turns off logging
#LOG_OFF
; Halt on errors
#HALT_ON FAIL
; Sets device
TCSETDEV FREESCALE MC9S08AW60 HCS08
; Target voltage, mV (change as needed)
TCSETPAR VDD 5000
; Clock oscillator frequency, Hz
; The possible frequencies are: 25000000, 12500000, 6250000, 0 (DISABLED)
TCSETPAR CLKOUT 0
; Reset rising time, us
TCSETPAR RSTUP 100
; Reset falling time, us
TCSETPAR RSTDOWN 100
; Power rising time, ms
TCSETPAR PWUP 10
; Power falling time, ms
TCSETPAR PWDOWN 10
; Sets the BDM entry mode
TCSETPAR BDM_ENTRY_MODE HW_RST
; Speeds up programming by driving the internal FLL circuitry to the maximum bus frequency.
; The maximum bus frequency depends on the supply voltage.
; (Change the FLL frequency as needed. 0 disables the FLL.)
TCSETPAR FLL_OSC 40000000
; Starts programming block
TPSTART
; Image file to be programmed (must be placed in the \BINARIES directory)
TPSETSRC FILE TEST.FRB
; Mass erases Flash memory
TPCMD MASSERASE F
; Trims internal reference oscillator (change frequency, trimming location and tolerance as needed)
; For this device the typical internal reference frequency is 243000 Hz.
; You can calibrate the internal reference from 182250 Hz to 303750 Hz.
TPCMD TRIM 243000 $FFBE 1
; Blank checks Flash memory (change address and length as needed)
TPCMD BLANKCHECK F $870 3984
TPCMD BLANKCHECK F $1860 59296
; Programs Flash memory (change addresses and length as needed)
TPCMD PROGRAM F $870 $870 3984
TPCMD PROGRAM F $1860 $1860 59296
; Verifies Flash memory, read-out method (change addresses and length as needed)
TPCMD VERIFY F R $870 $870 3984
TPCMD VERIFY F R $1860 $1860 59296
; Ends programming block
TPEND
```

The FlashRunner's system software setup will install script examples specific for each device of the S08 family in your PC.

7. Programming Times

The following table shows programming times for several devices representative of the S08 family.

Device	Memory Size	Bus Freq.	Operations	Time
MC9S08QD4	4 KB	8 MHz	Program	1.18 s
MC9S08QD4	4 KB	8 MHz	Verify	0.85 s
MC9S08QD4	4 KB	8 MHz	Erase + Program + Verify	1.97 s
MC9S08SH8	8 KB	16 MHz	Program	1.30 s
MC9S08SH8	8 KB	16 MHz	Verify	0.88 s
MC9S08SH8	8 KB	16 MHz	Erase + Program + Verify	2.15 s
MC9S08GT32	32 KB	20 MHz	Program	2.95 s
MC9S08GT32	32 KB	20 MHz	Verify	2.00 s
MC9S08GT32	32 KB	20 MHz	Erase + Program + Verify	4.35 s
MC9S08GB60	60 KB	8 MHz	Program	5.22 s
MC9S08GB60	60 KB	8 MHz	Verify	3.55 s
MC9S08GB60	60 KB	8 MHz	Erase + Program + Verify	8.75 s
MC9S08QD4	4 KB	8 MHz	Program	1.18 s
MC9S08QD4	4 KB	8 MHz	Verify	0.85 s
MC9S08QD4	4 KB	8 MHz	Erase + Program + Verify	1.97 s

Programming times depend on Programming Algorithm version, target board connections, target microcontroller mask, and other conditions. Programming times for your actual system may therefore be different than the ones listed here. SofTec Microsystems reserves the right to modify Programming Algorithms at any time.

8. References

FlashRunner user’s manual

Microcontroller-specific datasheets