24/01/2012 Giovanni Salvador

DC10444

# AN00153: Interfacing FlashRunner with Toshiba TLCS-870/C Devices

FlashRunner is a Universal In-System Programmer, which uses the principles of In-Circuit Programming to program Toshiba TLCS-870/C Series microcontrollers. This Application Note describes how to properly set up and use FlashRunner to program TLCS-870/C Flash devices.

This Application Note assumes that you are familiar with both FlashRunner and the main features of the TLCS-870/C Series. Full documentation about these topics is available in the FlashRunner user's manual and in device-specific datasheets.

## 1. Introduction

In-system programming of TLCS-870/C microcontrollers is performed through UART standard protocol.

In order to use FlashRunner to perform in-system programming, you need to implement the appropriate in-circuit programming hardware interface on your application board.

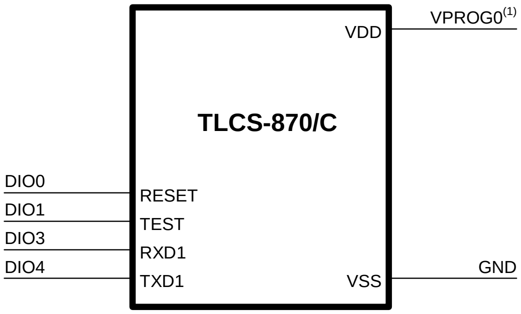
Thanks to its in-system programming capabilities, FlashRunner allows you to program or update the content of the Flash memory when the chip is already plugged on the application board.

## 2. Hardware Configuration

The microcontroller's lines needed to program a TLCS-870/C device are the following:

- **RESET:** Reset signal.
- **TEST:** Test pin for out-going test.
- **RXD1:** UART data input 1.
- **TXD1:** UART data output 1.

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



[1] Connect this line if you want FlashRunner to automatically power the target device

# 3. 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 following parameters must be correctly specified through the relative **`TCSETPAR`** commands (although the order with which these parameters are set is not important):

- VDD voltage;
- VDD_AUX voltage;
- Power Up time;
- Power Down time;
- Reset Up time;
- Reset Down time;
- Internal CPU frequency;
- Clock oscillator frequency driven by FlashRunner;
- Communication Baudrate;
- Password Flash Address;
- Password size Flash Address;
- Password.

## TPSETPAR 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 TEST, RESET, TXD1 and RXD1 signals. Additionally, the specified voltage is routed to the VPROG0 line of the FlashRunner’s “ISP” connector, which can be used as a supply voltage for the target board.

## TPSETPAR VDD_AUX

Command syntax:

```
TCSETPAR VDD_AUX <voltage mV>
```

Parameters:

**`voltage mV`**: Auxiliary supply voltage, expressed in millivolts, in the range 3000-14500mV.

Description:

This command is used to generate an optional, auxiliary voltage level for user purposes. The specified voltage is routed to the VPROG1 line of the FlashRunner “ISP” connector.

A value of 0 drives the VPROG1 line to GND. If the **`TCSETPAR VDD_AUX`** is not sent, the VPROG1 line is driven to HiZ.

## TPSETPAR 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 programming 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 programming mode.

## TPSETPAR 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.

## TPSETPAR 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 isn’t long enough, FlashRunner could not be able to enter the UART programming mode.

## TPSETPAR RSTDOWN

Command syntax:

```
TCSETPAR RSTDOWN <time µs>
```

Parameters:

**time µs**: Reset falling time, expressed in microseconds.

Description:

The Reset falling 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.

## TCSETPAR FOSC

Command syntax:

```
TCSETPAR FOSC <frequency Hz>
```

Parameters:

**`Frequency Hz:`** External quartz or oscillator frequency, expressed in Hz.

Description:

This frequency depends by the value of the external clock source, by the package and by the settings of the internal clock generator.

## TCSETPAR CLKOUT

Command syntax:

```
TCSETPAR CLKOUT 25000000 | 12500000 | 6250000 | 0
```

Command options:

Frequency of a clock signal to be generated at the CLOCKOUT pin of the FlashRunner “ISP” connector, expressed in Hertz.

Description:

Generates an auxiliary clock signal at the CLOCKOUT pin of the FlashRunner “ISP” connector. This signal can be used to enter the target device’s UART mode when the target device does not have an external clock. 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 isn’t greater than the maximum allowed frequency for your device. If the target device has an internal frequency divider, the actual device’s frequency will be a fraction of the CLKOUT frequency.

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

## TCSETPAR BAUDRATE

Command syntax:

```
TCSETPAR BAUDRATE <baudrate bit/s>
```

Parameters:

**`Baudrate bit/s:`** Communication Baudrate, expressed in bit/s.

Description:

This command sets the baudrate used during communication; it can assume fixed values: 9600, 19200, 38400, 57600, 62500, 76800. All these values are expressed in bit/s.

## TCSETPAR PWD_CNT_ADDR

Command syntax:

```
TCSETPAR PWD_CNT_ADDR <address>
```

Parameters:

**`Address:`** Flash address where password size is stored.

Description:

This command sets the Flash address where the password size is stored. The maximum password size is of 128 bytes.

## TCSETPAR PWD_FLASH_ADDR

Command syntax:

```
TCSETPAR PWD_FLASH_ADDR <address>
```

Parameters:

**`Address:`** Flash address where password data are stored.

Description:

This command sets the Flash address where the password is stored.

## TCSETPAR PWD

Command syntax:

```
TCSETPAR PWD <size> <data 1> <data 2> <data 3> <data 4> <data 5> <data 6> <data 7> <data 8>
```

Parameters:

**`Size:`** Password size. It indicates the number of data following in this TCSETPAR command.

**`Data x:`** Sets the password data.

Description:

This command specifies the size of the password and the data that form the password.

# 4. 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.

Toshiba TLCS-870/C -specific target programming commands are the following:

- **TPCMD MASSERASE**;
- **TPCMD BLANKCHECK**;
- **TPCMD PROGRAM**;
- **TPCMD VERIFY**;
- **TPCMD SECURE**;
- **TPCMD RUN**.

## TPCMD MASSERASE

Command syntax:

```
TPCMD MASSERASE F
```

Command options:

**F:** Specifies Flash (**F**) memory.

Description:

It erases Flash memory.

## TPCMD BLANKCHECK

Command syntax:

```
TPCMD BLANKCHECK F <tgt start addr> <len>
```

Command parameters and options:

**F**: Specifies Flash (**F**) memory.

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

**len**: Number of locations to be blankchecked.

Description:

It blankchecks Flash memory. Blankchecks **len** locations starting from the address specified by **tgt start address**. The start address is not related to PDI memory map but is the logical address specified in each single device datasheet under “Flash Memory” chapter.

The blankcheck command can be executed only with checksum method.

## TPCMD PROGRAM

Command syntax:

```
TPCMD PROGRAM F <src offset> <tgt start addr> <len>
```

Command parameters and options:

**F**: Specifies Flash (**F**) memory.

**src offset**: Offset from the beginning of the source memory.

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

**len**: Number of locations to be programmed.

Description:

It programs **len** locations of Flash memory starting from the **tgt start addr** address.

Flash memory is organized as 32-bit locations: **len** specifies the number of locations to be programmed. **tgt start addr** is not related to PDI memory map but is the logical address specified in each single device datasheet under “Flash Memory” chapter.

## TPCMD VERIFY

Command syntax:

```
TPCMD VERIFY F S <src offset> <tgt start addr> <len>
```

Command parameters and options:

**F**: Specifies Flash (**F**) memory.

**S** Specifies CRC (**S**).

**src offset**: Offset from the beginning of the source memory.

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

**len**: Number of locations to be verified

Description:

It verifies **len** locations of Flash memory starting from the **tgt start addr** address. The CRC parameter print back the CRC of the entire Flash memory calculated from target microcontroller.

Flash memory is organized as 32-bit locations: `len` specifies the number of locations to be verified.

The verify command can be executed only with checksum method.

## TPCMD SECURE

Command syntax:

```
TPCMD SECURE
```

Command parameters:

None.

Description:

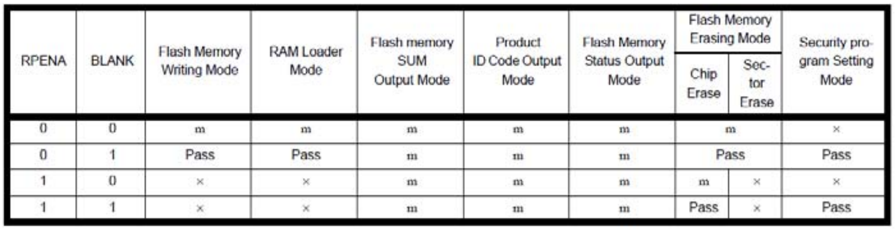
It secures the Flash memory of the device. An unsecure command doesn't exist. If an unsecure operation is necessary, the erase command must be used. Using the secure command, the micro will be set in "Security Program Enable". The RPENA bit is set to 1. Flash Memory writing mode and Sector erase cannot be executed. It is possible to use the Chip erase command if the BLANK bit is set to 0; if the BLANK bit is set to 1 the Chip erase command with password can be executed.

All the possible combinations of conditions are shown in the following table, where

m = the command can be executed

Pass = the command can be executed with a Password

x = the command can not be executed



| RPENA | BLANK | Flash Memory Writing Mode | RAM Loader Mode | Flash memory SUM Output Mode | Product ID Code Output Mode | Flash Memory Status Output Mode | Flash Memory Erasing Mode | | Security program Setting Mode |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | Chip Erase | Sector Erase | |
| 0 | 0 | m | m | m | m | m | m | | × |
| 0 | 1 | Pass | Pass | m | m | m | Pass | | Pass |
| 1 | 0 | × | × | m | m | m | m | × | × |
| 1 | 1 | × | × | m | m | m | Pass | × | Pass |

## TPCMD RUN

Command syntax:

```
TPCMD RUN
```

Command parameters:

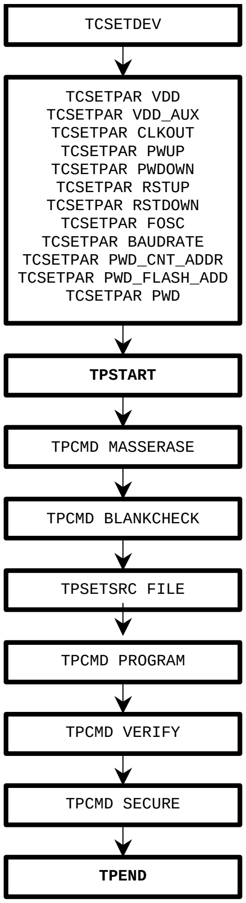
None.

Description:

It 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 below shows a typical programming flow for a Toshiba TLCS-870/C TMP86FS49BFG device.

```
;
; FLASHRUNNER SCRIPT EXAMPLE FOR TOSHIBA TMP86FS49BFG
;
; Use this example as a starting point for your specific programming needs
;
;
; ---------------------
; HARDWARE CONNECTIONS
; ---------------------
; DIO0 (RESET)
; DIO1 (TEST)
; DIO3 (RxD1)
; DIO4 (TxD1)
;

; Turns off logging
#LOG_OFF
; Halt on errors
#HALT_ON FAIL

; Sets device
TCSETDEV TOSHIBA TMP86FS49BFG TSHB_A

;----------
; SETTINGS
;----------
; Target voltage, mV (change as needed)
TCSETPAR VDD 5000

; VPROG1 voltage, mV (from 3000 to 14500, 0 to disable)(change as needed)
TCSETPAR VDD_AUX 0

; Clock oscillator frequency driven by FlashRunner, Hz
; Possible frequencies are: 25000000, 12500000, 6250000, 0 (DISABLED)
TCSETPAR CLKOUT 0

; VDD Rise-Time, ms (change as needed)
TCSETPAR PWUP 10
```

```
; VDD Fall-Time, ms (change as needed)
TCSETPAR PWDOWN 10

; RESET Rise-Time, us (change as needed)
TCSETPAR RSTUP 100

; RESET Fall-Time, us (change as needed)
TCSETPAR RSTDOWN 100

; Specifies external quartz or oscillator frequency, Hz
TCSETPAR FOSC 10000000

; Set the communication baudrate, bit/s
; The following baudrate values are allowed according to the FOSC value
; FOSC = 2000000,  baudrate values available: 9600
; FOSC = 4000000,  baudrate values available: 9600, 19200, 31250
; FOSC = 4190000,  baudrate values available: 9600, 19200, 31250
; FOSC = 4915200,  baudrate values available: 9600, 19200, 38400
; FOSC = 5000000,  baudrate values available: 9600, 19200, 38400
; FOSC = 6000000,  baudrate values available: 9600
; FOSC = 6144000,  baudrate values available: 9600
; FOSC = 7372800,  baudrate values available: 9600, 19200, 57600
; FOSC = 8000000,  baudrate values available: 9600, 19200, 31250, 38400, 62500
; FOSC = 9830400,  baudrate values available: 9600, 19200, 38400, 76800
; FOSC = 10000000, baudrate values available: 9600, 19200, 38400, 76800
; FOSC = 12000000, baudrate values available: 9600, 19200, 31250, 57600
; FOSC = 12288000, baudrate values available: 9600, 19200, 31250, 57600
; FOSC = 12500000, baudrate values available: 9600, 19200, 31250, 57600, 62500
; FOSC = 14745600, baudrate values available: 9600, 19200, 38400, 57600
; FOSC = 16000000, baudrate values available: 9600, 19200, 31250, 38400, 62500, 76800
TCSETPAR BAUDRATE 76800

;---------------------------
; PASSWORD SETTINGS
;---------------------------
; Use this settings only if you need to protect or unprotect the device
; Set the Flash address where Password size is stored
; TCSETPAR PWD_CNT_ADDR $1400

; Set the Flash address where Password data are stored
; TCSETPAR PWD_FLASH_ADDR $1800

; Set Password size and data
; TCSETPAR PWD 8 $01 $02 $03 $04 $05 $06 $07 $08
```

```
;---------------------------
; START PROGRAMMING SESSION
;---------------------------
TPSTART

; Mass erases Flash Memory
TPCMD MASSERASE F

; Blank checks Flash memory, checksum method(address and length are expressed in bytes)
TPCMD BLANKCHECK F $1000 61440

; Image file to be programmed (must be placed in the \BINARIES directory)
TPSETSRC FILE FLASH.FRB

; Programs Flash memory (address and length are expressed in bytes)
TPCMD PROGRAM F $1000 $1000 61440

; Verifies Flash memory, checksum method (address and length are expressed in bytes)
TPCMD VERIFY F S $1000 $1000 61440

; Protect the device using the specified Password settings
; TPCMD SECURE

; Ends programming block
TPEND
```

The FlashRunner’s system software setup will install script examples specific for each device of the TLCS-870/C Series on your PC.

# 7. Programming Times

The following table shows programming times for selected Toshiba TLCS-870/C TMP86FS49BFG devices.

| Device | Mem. Size | Conditions | Operations | Time |
|---|---|---|---|---|
| TLCS-870/C TMP86FS49BFG | 60KB Flash | VDD = 5V, UART, BAUDRATE = 76800 | Erase + Blank Check + Program + Verify | 12,19 s |

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

## 8. References

FlashRunner user’s manual

Microcontroller-specific datasheet