

Flasher 5 PRO

Tool for serial in circuit
programming of
microcontrollers with on-chip
flash memory

User & Reference Guide

Document: UM05009

Software version: 2.16

Revision: 0

Date: July 11, 2019



A product of SEGGER Microcontroller GmbH

www.segger.com

Disclaimer

Specifications written in this document are believed to be accurate, but are not guaranteed to be entirely free of error. The information in this manual is subject to change for functional or performance improvements without notice. Please make sure your manual is the latest edition. While the information herein is assumed to be accurate, SEGGER Microcontroller GmbH (SEGGER) assumes no responsibility for any errors or omissions. SEGGER makes and you receive no warranties or conditions, express, implied, statutory or in any communication with you. SEGGER specifically disclaims any implied warranty of merchantability or fitness for a particular purpose.

Copyright notice

You may not extract portions of this manual or modify the PDF file in any way without the prior written permission of SEGGER. The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such a license.

© 2010 - 2019 SEGGER Microcontroller GmbH, Monheim am Rhein / Germany

Trademarks

Names mentioned in this manual may be trademarks of their respective companies.

Brand and product names are trademarks or registered trademarks of their respective holders.

Contact address

SEGGER Microcontroller GmbH

Ecolab-Allee 5
40789 Monheim am Rhein
Germany

Tel. +49-2173-99312-0

Fax. +49-2173-99312-28

E-mail: support@segger.com¹

Internet: <http://www.segger.com>

1. By sending us an email your (personal) data will automatically be processed. For further information please refer to our privacy policy which is available at:
<https://www.segger.com/legal/privacy-policy/>.

Manual versions

This manual describes the current software version. If any error occurs, inform us and we will try to assist you as soon as possible.

Contact us for further information on topics or routines not yet specified.

Print date: July 11, 2019

Software	Revision	Date	By	Description
2.16	0	190711	MD	Updated the company name and the copyright information.
2.10	0	141006	MD	Initial version.

About this document

Assumptions

This document assumes that you already have a solid knowledge of the following:

- The software tools used for building your application (assembler, linker, C compiler)
- The C programming language
- The target processor
- DOS command line

If you feel that your knowledge of C is not sufficient, we recommend *The C Programming Language* by Kernighan and Ritchie (ISBN 0-13-1103628), which describes the standard in C-programming and, in newer editions, also covers the ANSI C standard.

How to use this manual

This manual explains all the functions and macros that the product offers. It assumes you have a working knowledge of the C language. Knowledge of assembly programming is not required.

Typographic conventions for syntax

This manual uses the following typographic conventions:

Style	Used for
Body	Body text.
Keyword	Text that you enter at the command-prompt or that appears on the display (that is system functions, file- or pathnames).
Parameter	Parameters in API functions.
Sample	Sample code in program examples.
Sample comment	Comments in program examples.
Reference	Reference to chapters, sections, tables and figures or other documents.
GUIElement	Buttons, dialog boxes, menu names, menu commands.
Emphasis	Very important sections.

Table 1.1: Typographic conventions



SEGGER Microcontroller GmbH develops and distributes software development tools and ANSI C software components (middleware) for embedded systems in several industries such as telecom, medical technology, consumer electronics, automotive industry and industrial automation.

SEGGER's intention is to cut software development time for embedded applications by offering compact flexible and easy to use middleware, allowing developers to concentrate on their application.

Our most popular products are emWin, a universal graphic software package for embedded applications, and embOS, a small yet efficient real-time kernel. emWin, written entirely in ANSI C, can easily be used on any CPU and most any display. It is complemented by the available PC tools: Bitmap Converter, Font Converter, Simulator and Viewer. embOS supports most 8/16/32-bit CPUs. Its small memory footprint makes it suitable for single-chip applications.

Apart from its main focus on software tools, SEGGER develops and produces programming tools for flash micro controllers, as well as J-Link, a JTAG emulator to assist in development, debugging and production, which has rapidly become the industry standard for debug access to ARM cores.

Corporate Office:

<http://www.segger.com>

United States Office:

<http://www.segger-us.com>

EMBEDDED SOFTWARE (Middleware)



emWin

Graphics software and GUI

emWin is designed to provide an efficient, processor- and display controller-independent graphical user interface (GUI) for any application that operates with a graphical display.



embOS

Real Time Operating System

embOS is an RTOS designed to offer the benefits of a complete multitasking system for hard real time applications with minimal resources.



embOS/IP

TCP/IP stack

embOS/IP a high-performance TCP/IP stack that has been optimized for speed, versatility and a small memory footprint.



emFile

File system

emFile is an embedded file system with FAT12, FAT16 and FAT32 support. Various Device drivers, e.g. for NAND and NOR flashes, SD/MMC and Compact-Flash cards, are available.



USB-Stack

USB device/host stack

A USB stack designed to work on any embedded system with a USB controller. Bulk communication and most standard device classes are supported.

SEGGER TOOLS

Flasher

Flash programmer

Flash Programming tool primarily for micro controllers.

J-Link

JTAG emulator for ARM cores

USB driven JTAG interface for ARM cores.

J-Trace

JTAG emulator with trace

USB driven JTAG interface for ARM cores with Trace memory. supporting the ARM ETM (Embedded Trace Macrocell).

J-Link / J-Trace Related Software

Add-on software to be used with SEGGER's industry standard JTAG emulator, this includes flash programming software and flash breakpoints.



Table of Contents

1	Introduction	9
1.1	Flasher overview	10
2	Working with Flasher	13
2.1	Connecting Flasher to the PC.....	14
2.2	Using the Flasher PC software	15
2.3	Using the serial link to program in circuit	25
2.4	Operating Flasher in stand-alone mode.....	26
2.5	Error messages	27
3	Remote control.....	31
3.1	Overview	32
3.2	Handshake control	33
3.3	ASCII command interface	34
3.4	Reply messages from Flasher	37
4	Batch mode.....	39
4.1	Introduction.....	40
4.2	List of commands	41
4.3	Examples	44
5	Hardware	47
5.1	Target interface for M16C/62, M16C26P, M16C/80, M32C and R32C	48
5.2	Serial target interface circuitry	51
5.3	Target interface for other systems	52
5.4	CRC calculation used in Flasher and PC software	53
6	Support	55
6.1	Trouble shooting.....	56
6.2	Known limitations	57
6.3	Contacting support	58
7	Glossary.....	59

Chapter 1

Introduction

This chapter gives a short overview about the features of the Flasher.

1.1 Flasher overview

Flasher is a programming tool for microcontrollers with on-chip flash memory. It is designed for programming flash targets with the Flasher software or in stand-alone mode. Flasher connects to a PC using the USB/Ethernet/RS-232 interface, running Microsoft Windows 2000, Windows XP, Windows 2003, Windows Vista, Windows 7 or Windows 8. In stand-alone mode, Flasher can be driven by the start/stop button, or via the RS-232 interface (handshake control or ASCII interface). Flasher interfaces to target via a 10-pin connector.

1.1.1 Supported microcontrollers

- M16C/1N series
- M16C/26 series
- M16C/24 series
- M16C/26; M16C26A series
- M16C/28; M16C28B series
- M16C/29 series
- M16C/56 series
- M16C/57 series
- M16C/57 series
- M16C/62 series
- M16C/62P series
- M16C/64 series
- M16C/65 series
- M16C/6N series
- M16C/6S series
- M16C/6V series
- M16C/80 series
- M32C/83; M32C/84; M32C/85; M32C/86; M32C/88; M32C/95 series
- R8C/14; R8C/15; R8C/18; R8C/19; R8C/22; R8C/23; R8C/24; R8C/25; R8C/26; R8C/27; R8C/28;
- R8C/29; R8C/2C; R8C/2D, R8C/3x series
- R32C series
- M79 series

For detailed information on supported flash chips please check www.segger.com/flashmcus.htm

1.1.2 Features

- Easy to use Windows software
- Serial in target programming supported
- Programming / Clearing / Verifying / Read back supported; High speed programming
- User or boot area selectable (read only in serial mode)
- 128 MByte flash memory to store target program
- Can be used in a production environment
- PC software for batch mode processing, allowing usage in automated test systems.
- Remote control functions for automated testers
- Flasher 5 PRO can be controlled and accessed via RS-232 or Ethernet.

1.1.3 Working environment

Host System

IBM PC/AT or compatible. CPU: 486 (or better) with at least 8Mb of RAM, running Windows 95 / 98 / 2000 / Vista / Windows 7 or Windows 8. It needs to have an RS-232 or Ethernet interface available for communication with Flasher.

Power supply

Flasher requires a 5V power supply over its USB connector, minimum current consumption is about 80mA. You may use the power supply which comes with the tool or may power it from a USB port from your PC. Flasher is NOT protected against polarity reversion on the supply input. Please avoid excess voltage over 6.0V.

Flasher can be supplied from the target, if the target is operated from 4.5 to 5V.

Installing the Flasher PC-software

The latest PC software FLASHER.EXE is available from the download pages on our website: <http://www.segger.com>.

In order to use the software, simply copy it onto your hard drive and start the executable which will guide you through the installation process.

Chapter 2

Working with Flasher

This chapter describes functionality of Flasher and how to use it.

2.1 Connecting Flasher to the PC

2.1.1 Using the RS-232 interface with a serial COM port

Flasher comes with a standard serial 1 by 1 interface cable with one male and one female 9 pin SUB-D9 connector which can be used to connect the Flasher to the PC. The pin assignment of the 9 pin SUB-D female RS-232 interface connector is as follows:

Pin no.	Signal	Function	Host signal
2	TxD	Serial async. data input	Serial data output (TxD)
3	RxD	Serial async. data output	Serial data input (RxD)
5	GND	Signal ground	Signal ground

Table 2.1: RS-232 signals

Getting started

1. Connect the Flasher to the power supply.
2. Connect the Flasher to a PC using the 1 by 1 RS-232 serial interface cable and run the Flasher PC software FLASHER.EXE
3. Select the chosen COM port of your PC in the communication settings dialog in the Flasher PC software by menu: "Options | Communication".
4. Set up the device via menu "Options | Device" in the PC software.
5. For in-circuit programming: Connect the Flasher to the target system via the standard 10 pin or customized interface cable.

2.1.2 Using the RJ45 Ethernet interface

Flasher comes with an RJ45 Ethernet cable which can be used to communicate via Ethernet. The cable can be plugged in directly into an RJ45 socket on the PC or elsewhere in the LAN using hubs or switches.

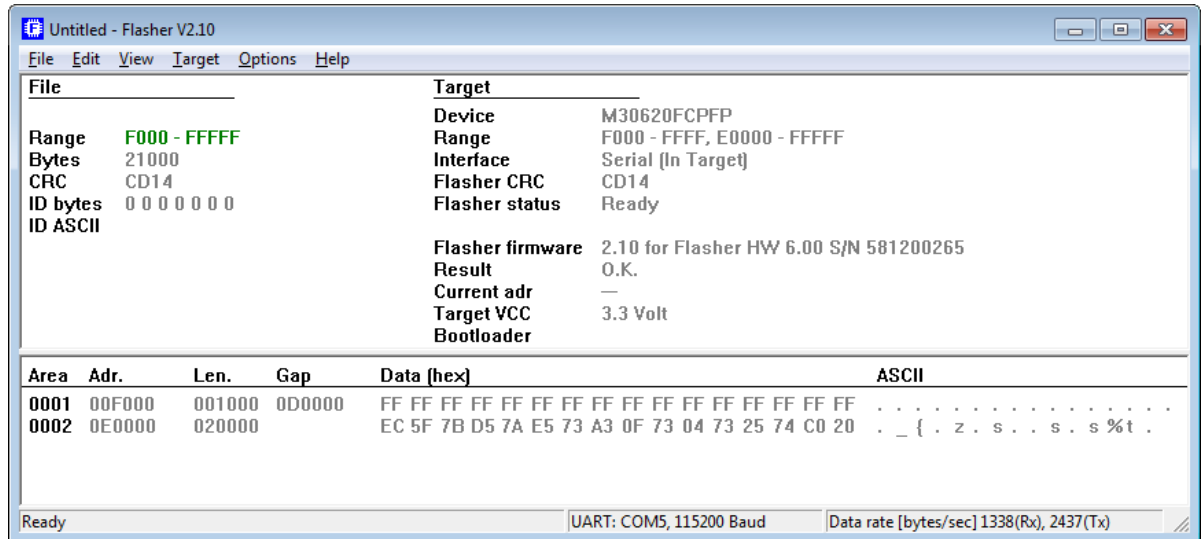
Getting started

1. Connect the FLASHER to the power supply.
2. Connect the FLASHER to a PC or LAN using the RJ45 Ethernet cable and run the Flasher PC software FLASHER.EXE
3. Select the Flasher in the communication settings dialog in the Flasher PC software by menu: "Options | Communication". Details how to select the Flasher with Ethernet connection are described later on in the manual.
4. Set up the device via menu "Options | Device" in the PC software.
5. For in-circuit programming: Connect the Flasher to the target system via the standard 10pin or customized interface cable.

2.2 Using the Flasher PC software

General

Flasher comes with an easy to use Windows software. It allows reading of program files in motorola, intel hex or binary format. The following is a screenshot of FLASHER.EXE with loaded target program.



Setup the communication between PC and Flasher

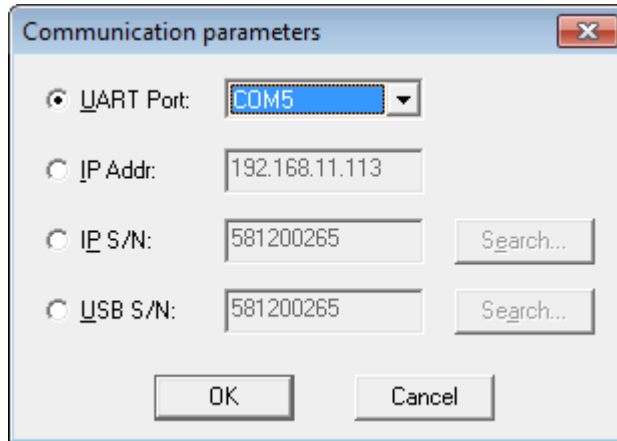
Make sure that the power supply is connected (one of Flasher LEDs should be illuminated) and that the Flasher is connected to your PC with either the 9pin 1 by 1 RS-232 serial interface cable, or the Ethernet cable. Both interfaces can be connected at the same time. It depends on the configuration, which interface is used to communicate. The Ethernet interface is much faster than the serial RS-232 interface, but may require additional configuration of the Flasher.

2.2.1 Setup the serial UART connection for communication

The RS-232 port of the Flasher can be used for communication to PC. You have to select a COM port on your PC and set up the Flasher PC software to use this COM port.

1. Connect the Flasher to a COM port of your PC via the provided serial cable.
2. Chose menu "Options | Communication" to open the setup dialog for the communication parameters.
3. Select the "UART Port" radio button.
4. In the corresponding listbox select the COM port which is connected to your Flasher. You may chose COM1 to COM20. If this does not match any of your COM ports, you may try to assign a port number in the range from 1 to 20 in your Sys-

tem settings to the COM port you want to use.



After the COM port selection, press OK. The Flasher PC software and Flasher should now communicate. The Flasher status and serial number should be visible in the Flasher PC software.

2.2.2 Setup the Ethernet connection for communication

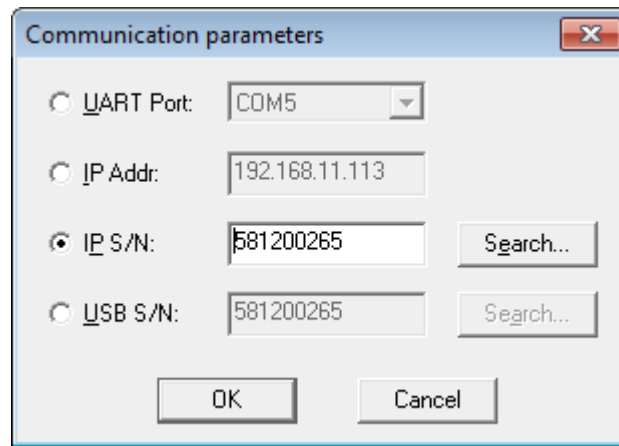
Using the Ethernet for communication has the advantage that download and upload of data between the Flasher and PC is much faster. Since Flasher software version 2.10, communication between Flasher PC software and Flasher via Ethernet is supported.

A Flasher with firmware older than version 2.10 requires a firmware update which can be done via the RS-232 port.

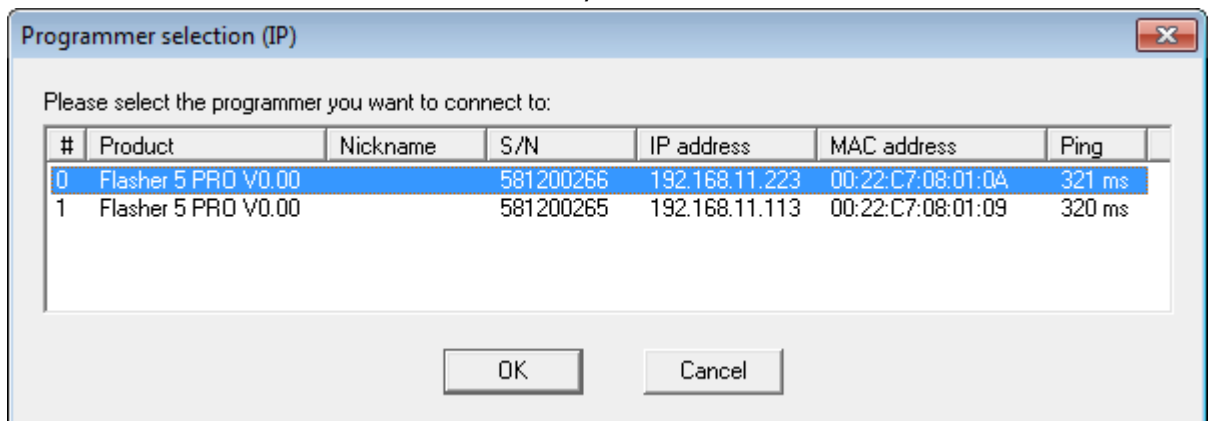
Initially, the Flasher retrieves its IP address from a DHCP server. Therefore, the connection between the Flasher and the PC requires a LAN with an active DHCP server to communicate unless a fixed IP address has been assigned to the Flasher.

When the Flasher and the PC are connected to a hub or switch in the same LAN with DHCP server, the Flasher can be accessed by your PC without further configuration.

1. Connect the Flasher to a hub or switch in your LAN using the RJ45 cable. It may take a few seconds until the Flasher gets its IP address from your DHCP server.
2. In the Flasher PC software chose menu "Options | Communication" to open the setup dialog for communication parameters.
3. Depending on the desired connection mode select "IP Addr" or "IP S/N" radio button. If "IP Addr" radio button is selected the Flasher PC software connects to Flasher using the IP address of Flasher. The IP address can be specified on the edit field on the right in the "dotted" format. If the "IP S/N" radio button is selected the Flasher PC software uses the serial number of the Flasher to connect to it. The serial number can be found on the label at the bottom of the Flasher. The edit field on the right can be used to specify the serial number of the Flasher to connect to.
4. Click "OK" button to confirm your settings.



You may click the "Search..." button to scan the network for your connected Flasher. A list with Flasher which are found in your network is shown:



Select one Flasher in the list and click OK (or double-click on one Flasher in the list). All available data of the selected Flasher is automatically read into the communication parameters dialog. At least the serial number and the IP address are available. When used in a LAN with DHCP server, it is a good choice to address and identify the Flasher by its serial number.

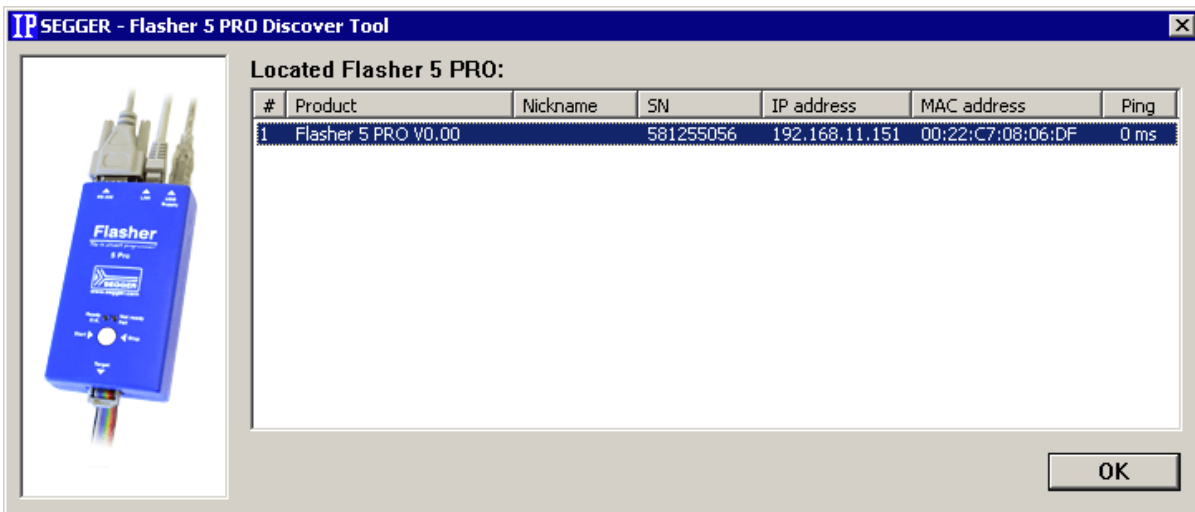
2.2.3 Setup the Network configuration for Flasher

It may be useful to assign a fixed IP address and a nickname to the Flasher, which can be done by the network configuration for the Flasher.

Fixed IP addresses have to be used when the Flasher should be driven in a network without DHCP server, for example, if it shall be directly connected to a PC or Laptop without DHCP server. Note that with version 2.10 of the Flasher PC software, this configuration settings can not be set from the Flasher PC software. Network configuration for the Flasher from the Flasher PC software will be implemented in later versions of the Flasher PC software.

To configure the network settings of Flasher version 2.10, you have to use the built in web-server of Flasher 5 PRO. For first time setup, connect the Flasher to a LAN with DHCP server. To start the web server, you have to examine the IP address of the connected Flasher. To get the IP address, you may use the Flasher PC software and

retrieve the IP address from the programmer selection list as described above, or you may start the 5PRO_UDPDiscover.exe tool which is located in the program folder of the Flasher PC software.



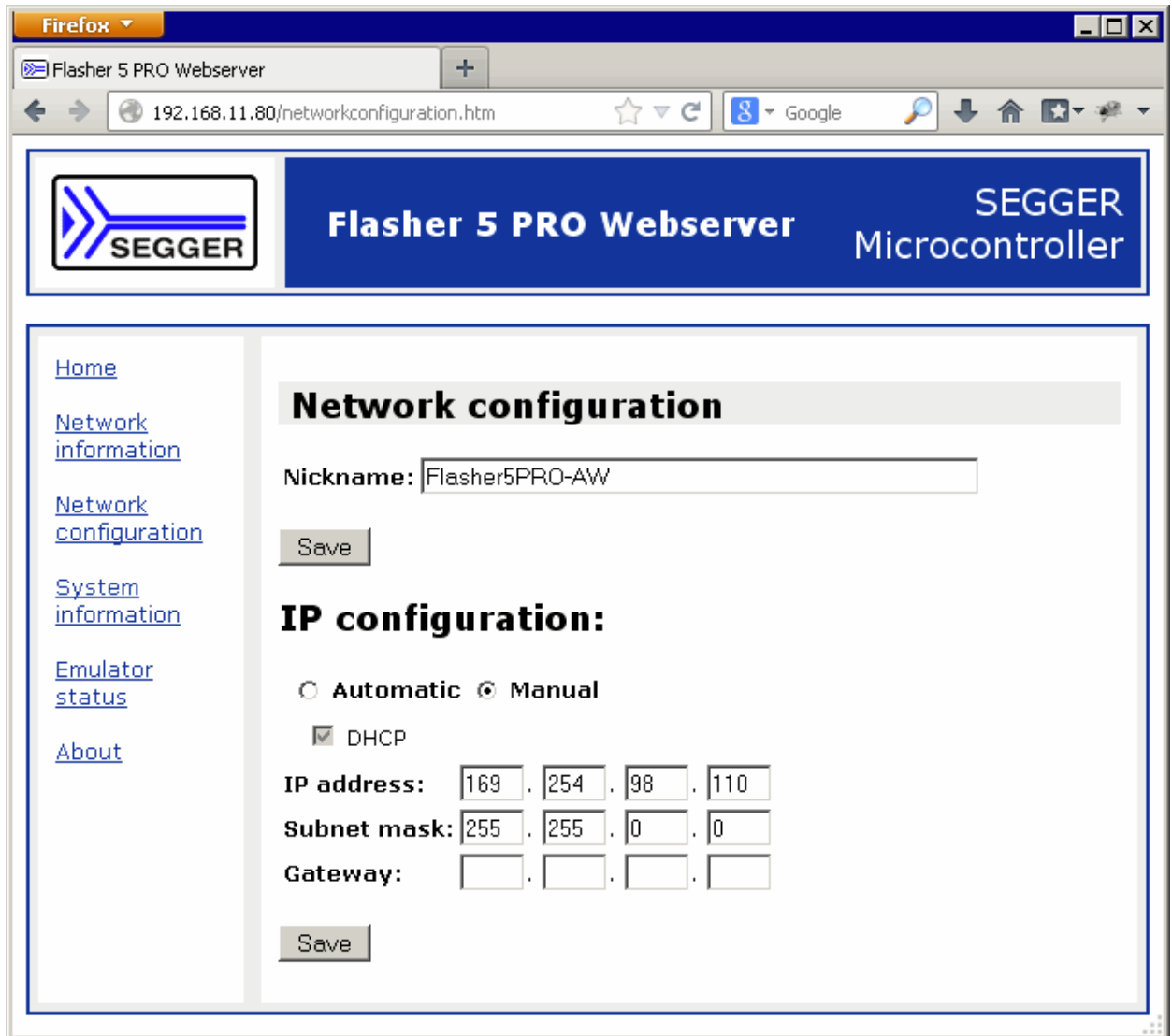
After examining the IP address of the Flasher to configure, start a web browser and enter the IP address in the address bar of your browser. Using the example above, you have to type 192.168.11.151 in the address bar of your browser.

The browser will connect to the Flasher and show the main web-page. In the web server window, chose the function "Network configuration" by a click of the link shown on the left side.

When you want to assign a nickname and a fixed IP address, first enter the nickname and press the "Save" button assigned to the nickname. Then enter the IP address and the subnet mask and press the "Save" button assigned to the IP address settings. There is no need to enter an IP address for a gateway.

When you modify the IP address and save the settings, you will lose the connection. It is a good idea to write down the address you assigned. Be careful to assign an address which you can reach with your PC or laptop.

The screenshot below shows a sample configuration to assign a fixed IP address and a nickname to the Flasher:



The sample above shows a configuration which was used to setup Flasher for a direct connection between a laptop and Flasher. The laptop had the IP address 169.254.98.109, the IP address for Flasher has to differ from the IP address of the PC it should be used with, but has to be in the same subnet.

2.2.4 Setup the USB connection for communication

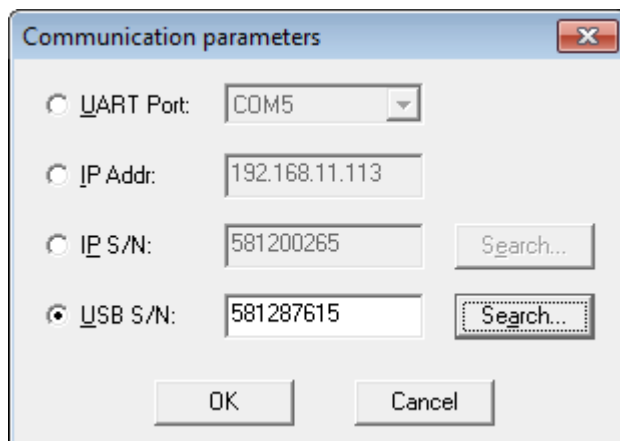
Since the software version 2.10 Flasher supports communication via USB. Communication via USB has the advantage of being faster than communication via UART.

A Flasher with firmware older than version 2.10 requires a firmware update which can be done via the RS-232 port.

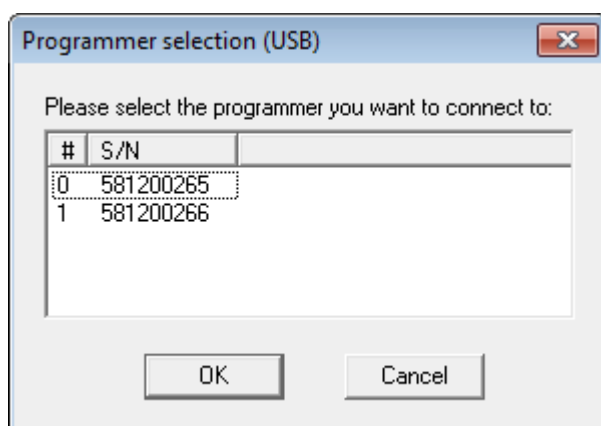
The following steps must be followed in order to connect the Flasher to PC via USB:

1. Connect the Flasher to an USB port of your PC or to a USB hub connected to your PC using the provided USB cable.
2. Chose menu "Options | Communication" to open the setup dialog for the communication parameters.
3. Select the "USB S/N" radio button.
4. In the corresponding edit field specify the serial number of the Flasher you want to connect to.

5. Click "OK" button to confirm your settings.



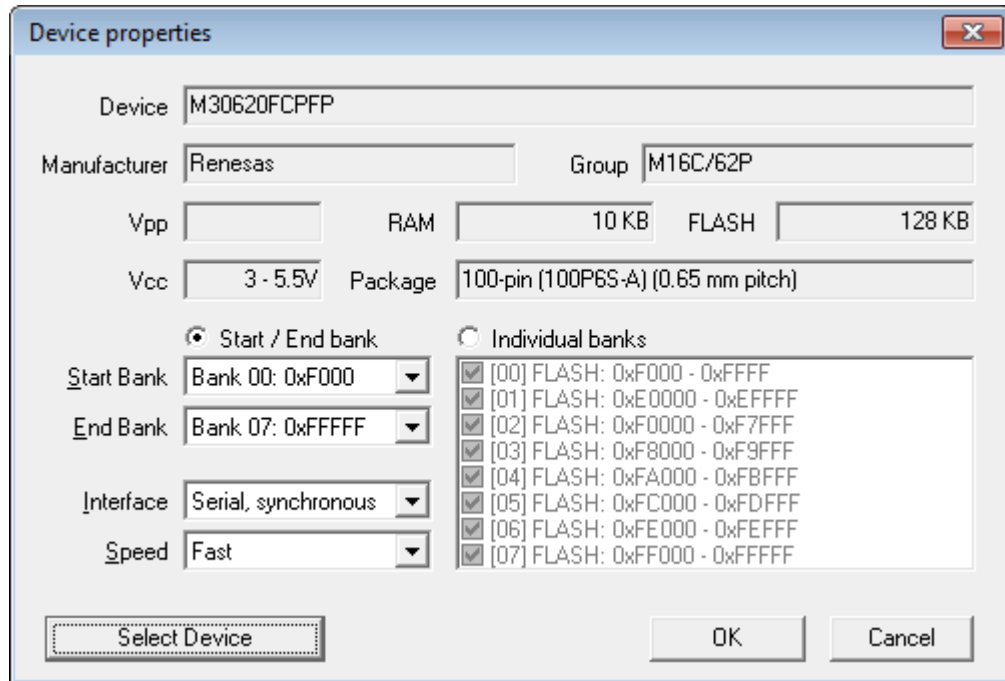
You may click the "Search..." button to scan the USB ports for your connected Flasher. A list with Flasher devices which are found is shown:



Select one Flasher in the list and click OK (or double-click on one Flasher in the list). The serial number of the selected Flasher is automatically read into the communication parameters dialog.

2.2.5 First time setup of Flasher

When using Flasher for the first time, please select the menu point "Options | Device". You will see the following dialog box:



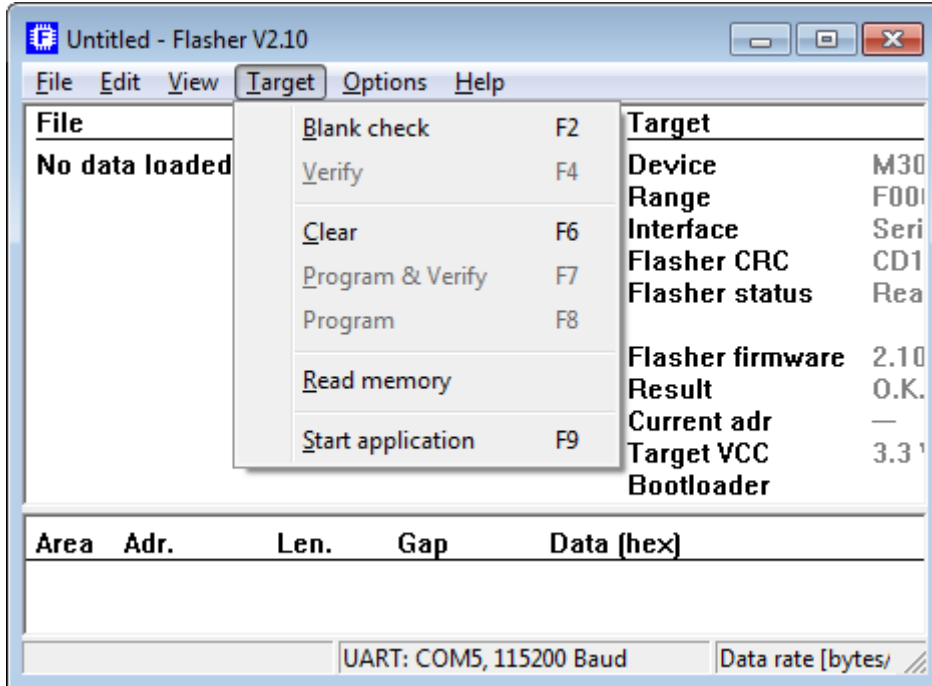
The device properties dialog allows selection of the chip area you would like to access, the sectors of the on chip-flash and the interface you would like to use. The serial interface requires a cable to connect Flasher to your target. For targets running at low frequencies, it may be necessary to set the "Speed" option to "Slow".

To select an other device, press the "Select Device" button. The device selection dialog will open. You can select the group in a dropdown list and the specific device from the list below.

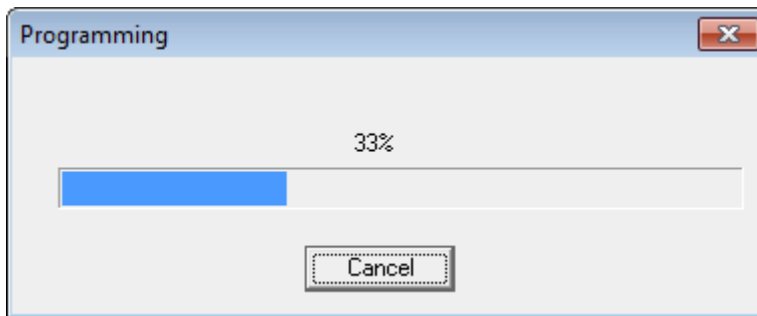
Now you should be able to blank check, clear, program, verify or read the target chip in serial mode (if your target is properly connected to Flasher). The first time you program or verify, the PC downloads your target program to Flasher, where it is stored in the on board RAM chip for programming or verification. The PC Software stores all setup information in an FLASHER.INI; when you start the program the next time, it will start with the same settings.

2.2.6 Program, clear, verify and blank check

Select one of the commands in the "Target" menu to start the operation. Note that some of the menu points may be grayed if you have no connection to the target or no file loaded.



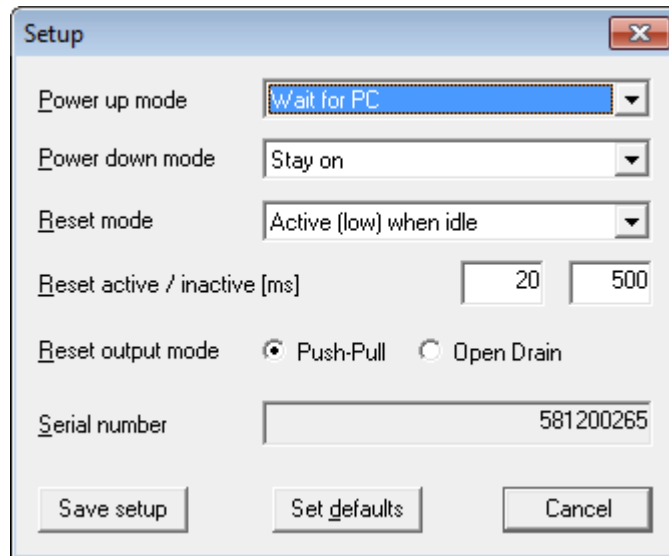
A modal dialog box will indicate the status and progress of the operation; the operation can be stopped any time by hitting the "Cancel" button.



2.2.7 Setup

The operating mode of Flasher may be changed using the "Setup" dialog from the "Options" menu. Power up mode, Power down mode and Reset mode should not be changed for normal operation. You may change the reset active and reset inactive

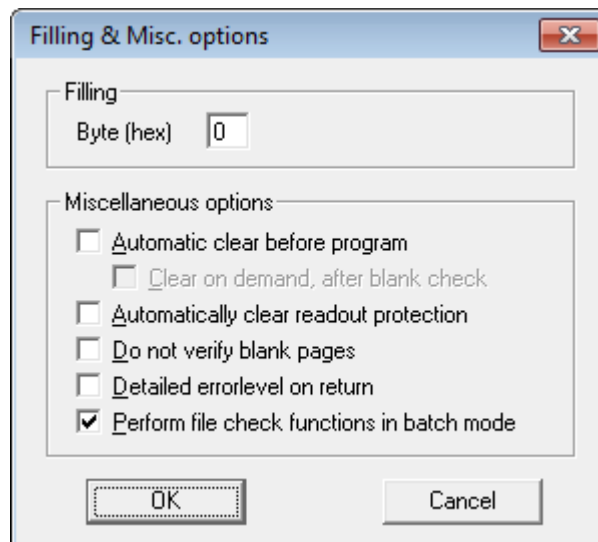
time, if required by your target hardware. You may select reset output mode as Push-Pull output or Open Drain. All setup settings are stored permanently in Flasher after pressing "Save setup" button.



2.2.8 Additional options

The "Filling & Misc." settings from the "Options" menu may be altered if required. Normally there is no need to change any of these settings. Improper setting of fill byte may lock your target CPU! When programming blank (virgin) CPUs the option "Automatic clear before program" is not required, so this feature can be disabled to speed up programming procedure.

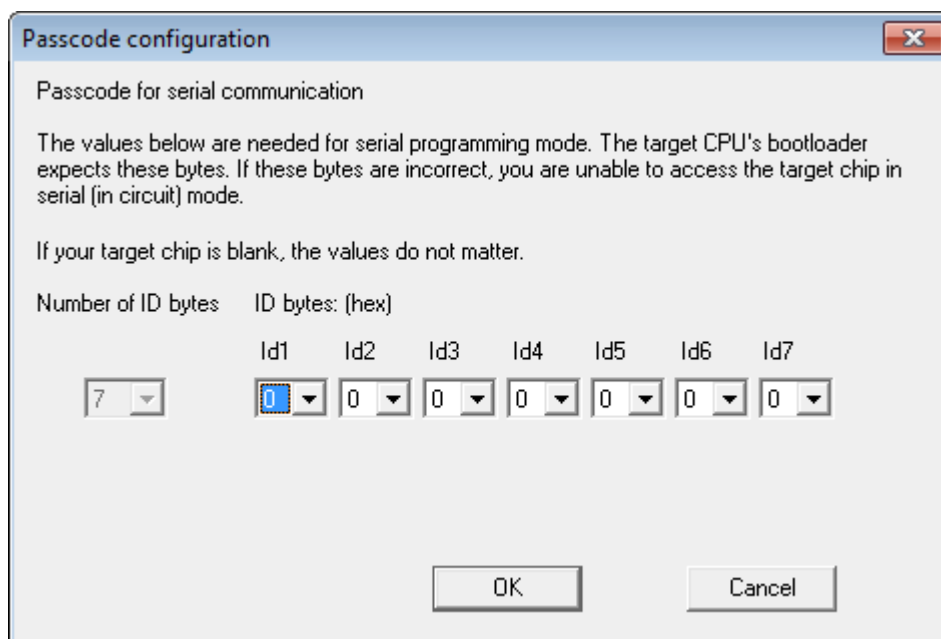
"Detailed errorlevel on return" option may be used to return a detailed error code to the calling program when Flasher is used in batch mode.



2.2.9 ID check

When programming Renesas CPUs in serial mode (in target), an identification of up to 15 bytes has to be supplied. If the target MCUs user program area is blank, this ID-value does not matter. However, after programming, these values need to be set correctly, because otherwise Flasher will be unable to communicate with the target CPU. These ID-values can be set using the menu point "Options | Passcode". With a standard program, these values should be 0, as the high bytes of the interrupt vectors which are used to store the values are usually 0.

For more detailed information, please consult the Renesas users manual. The menu point "Edit | Copy passcode into loaded file" can be used to copy these ID bytes into your application program.

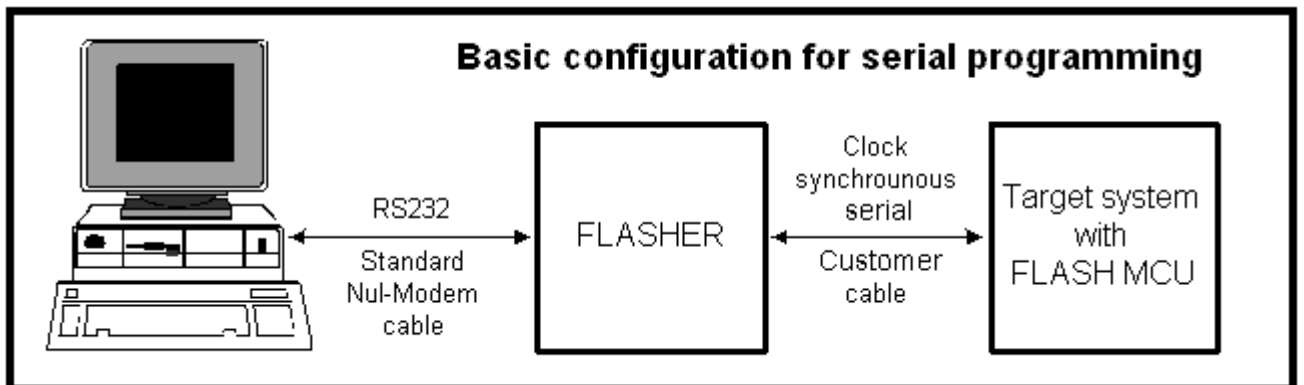


Problems with ID check

You should act carefully when programming ID bytes. If you do not know the ID-value programmed into a target chip, there is no way to erase, read or reprogram the chip in-circuit later. We recommend not to use this feature during the development process.

2.3 Using the serial link to program in circuit

Flasher can be used for in circuit programming of supported CPUs, which incorporate built in firmware for serial update of user flash. The target system has to be designed to support this mode of operation. Refer to target specific connection diagrams or Users manuals of your target CPU.



2.4 Operating Flasher in stand-alone mode

After downloading the target program, all settings are stored in Flasher on board flash memory and remain valid until new settings or data are sent to Flasher. Any number of microcontrollers may now be programmed by Flasher (one at a time) without the need of a host PC, by simply pressing the start button. Flasher will use the settings which have been made in the PC software. This includes the selection of target address range as well as any options. Whether the target CPU will be erased before programming depends on setting of option "Automatic clear before program".

Progress and result of the operation is indicated by Flasher LEDs as follows:

LED status	Meaning
GREEN, flashing	Erasing / Programming / Verifying operation in progress.
GREEN	Programming operation successful.
RED	Programming operation failed

Table 2.2: LED status

2.5 Error messages

The following error messages can occur during operation of Flasher (shown in red on your PC) or returned as errorlevel when operated from batch file and "Detailed error-level on return" option is set.

Code	Error message	Meaning/remedy
1	Erase failed	Erase operation has failed.
2	Write failed	Write operation has failed.
3	Verify failed	Verification failed. Loaded Program and contents of the flash-memory are not identical.
4	Blank check failed	Chip is not blank.
5	Flash write/erase timed out	Could not write into flash memory, the max. waiting time has been exceeded.
6	Can not write into this memory area	In serial mode, the boot area of Renesas CPUs can be read out, but can not be written to.
7	Canceled	Last operation has been canceled by user.
21	Version read failure: Rx line problem ?	In serial mode, Flasher reads out the version of Renesas target CPUs boot-loader. If this is not the right format (VER?.), a read line failure is most likely.
30	No ID	Renesas target CPU has no valid ID (This error should not occur).
31	ID mismatch	Renesas bootloader (target CPU) requires the correct ID. Without it, you will be unable to access, read out, clear, or program the contents of the flash memory. The requested ID depends on ID previously written into the target CPU. For more information, please refer to the section "ID code check function" of the CPUs users manual.
32	ID mismatch	Same as Error 31
40	Target chip says "BUSY"	Renesas target chip has BUSY signal high (active). Most likely CPU did not enter bootmode. Check all signals to target. Refer to CPUs users manual about conditions to enter bootmode.
41	Target chip says BUSY	Renesas target chip keep its Busy signal set (active) during communication. This inhibits communication between FLASHER and target. Some possible reasons: <ol style="list-style-type: none"> 1. RESET is not released 2. Target did not enter the serial I/O mode because the signals applied when RESET is released are not correct (EPM, CE, CNVSS) 3. No clock (check with oscilloscope)

Table 2.3: List of error messages

Code	Error message	Meaning/remedy
43	Timeout of target	Target command Clear, Program or Verify could not be finished within expected time. Please report this error.
44	Timeout during async. data reception	Target communication does not work. This error may be reported with R8C targets, or other targets connected via asynchronous interface. Try another baud rate.
55	CRC check in programmer failed.	Integrity of target data held in Flasher internal memory is lost. Download new data to Flasher.
56	Internal Vcc drop during operation	A voltage drop on Flasher internal supply was detected. Internal data may be damaged.
58	DAC for Vpp not calibrated	Flasher lost calibration data for Vpp generation circuit. Please return Flasher to factory, as calibration can not be done by user.
60	Unsupported interface, check device setting	Flasher is requested to access a target CPU that is not supported by Flasher. Normally this fault should not occur. Ensure that Flasher PC software version fits to Flasher firmware version
61	Unsupported command	Flasher received a command that is not supported. Ensure that Flasher PC software version fits to Flasher firmware version.
62	No target data, please download	Flasher never received valid data for target, therefore target can not be programmed. Download target data.
63	FLASHER lost Setup	Setup data for Flasher invalid. Please run "Option Setup. and Option Device".
202	Blank check failed	Target CPU is not blank.
203	Verify failed	Verification failed. Loaded Program and contents of target CPUs flash memory are not identical.
204	Clear target failed	Erase operation has failed.
205	Target auto function failed	Any error occurred during "Auto" function (Clear / Program / Verify)
206	Programming target failed	An error occurred during programming of target. If target was not blank before, ensure that target is erased before programming. Retry.
207	Reading target failed	An error occurred during reading of target. Retry operation. Check connection to target.
210	Could not open source file	The source file given as parameter could not be opened. Check file name or path.
211	Invalid Parameter.	Invalid command line parameter found.
212	Invalid parameter count.	Check parameter. Refer to command description.
213	Invalid command syntax.	Check parameter. Refer to command description.

Table 2.3: List of error messages

Code	Error message	Meaning/remedy
214	Firmware mismatch	PC software version differs from firm-ware version of Flasher. Download new firmware via Options menu. Otherwise proper function can not be guaranteed.
215	The selected target chip is not supported by the connected programmer.	Check and setup device settings.
216	Session mismatch. Resetting connection	It seemed, that Flasher was disconnected during communication between Flasher and PC. Try an other COM port.
217	Flasher refuses connection..	Did you select a chip which is not supported? Check device settings.
218	None of the selected com ports is available.	PC software could not open selected COM-Port. Select an other COM port.
219	Error in HEX file	The HEX file for target could not be read. Check file format.
220	Invalid file extension, use .MOT, .HEX or .BIN	You tried to open or save a Hex file with unsupported file extension.
221	Invalid file name	You tried to Open or merge a file that could not be found. Check file name or path.
222	Error in Protocol Manager	PC software internal error. Restart program.
223	VCCS > VCCSmax (Target supply voltage too high...)	Supply voltage of target exceeds max value defined for selected device. Check device settings, connection to Flasher and target supply. If everything is correct, Flashers voltage measurement circuitry seems to be damaged and has to be repaired.
224	VCCS < VCCSmin (Target supply voltage too low...)	Supply voltage of target is lower than min. value defined for selected device. Check device settings, connection to Flasher and target supply. If everything is correct, Flashers voltage measurement circuitry seems to be damaged and has to be repaired.
225	Flasher Input voltage too low	Flashers supply voltage is lower than required to generate programming voltage of connected target CPU. Check supply voltage
226	Assertion	PC software internal error. Restart and try again.
227	Invalid default program settings found. Please setup device.	The .ini file contains invalid data about Flasher PC software. Setup device and other options.
228	Flasher not ready !	Setup data could not be sent to Flasher. Is a Flasher connected to your PC? Check COM port.
229	Invalid commandline option	Your command line parameter contains a command which is not supported.
230	Flasher defect !!! Target interface does not work.	Target interface of Flasher not found during initialization. Flasher has to be repaired.

Table 2.3: List of error messages

Chapter 3

Remote control

This chapter describes how to control Flasher via RS-232 interface.

3.1 Overview

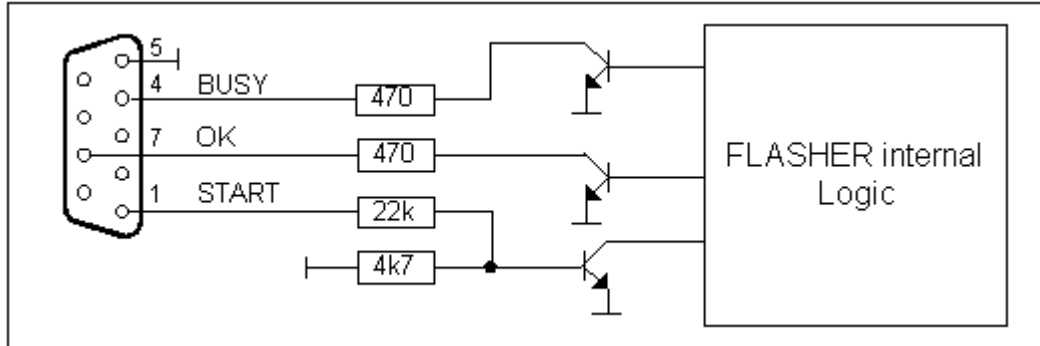
There are 3 ways to control Flasher operation:

- Manual: Programming operation starts when pressing the button. The LEDs serve as visible indication.
- Via Handshake lines: 3 lines on the serial interface are used. 1 line is an input and can be used to start operation, 2 lines are outputs and serve as Busy and status output.
- Terminal communication via RS-232.

3.2 Handshake control

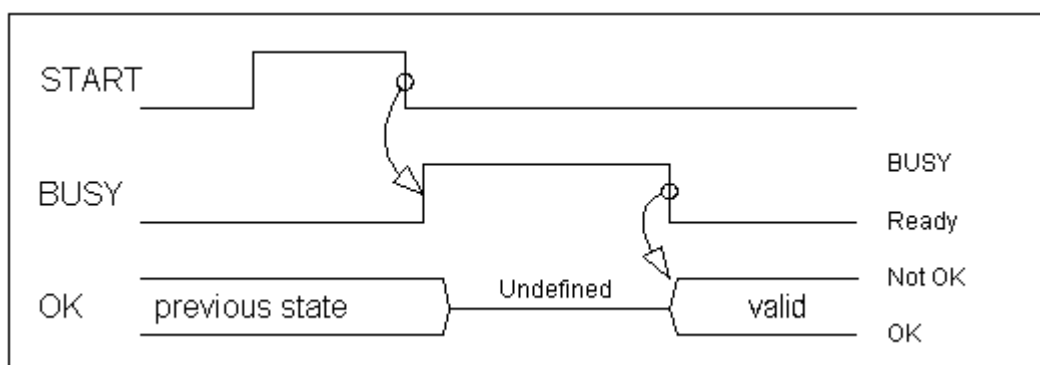
FLASHER 5 can be remote controlled by automated testers without the need of a connection to PC and Flashers PC software. Therefore Flasher is equipped with additional hardware control functions, which are connected to the SUBD9 male connector, normally used as RS-232 interface to PC.

The following diagram shows the internal remote control circuitry of Flasher:



Pin No.	Function	Description
1	START	A positive pulse of any voltage between 5 and 30V with duration of min. 30 ms starts the "Auto" function (Clear / Program / Verify) on falling edge of pulse. Whether Clear is executed depends on Options Filling & Misc. Automatic clear before program.
4	BUSY	As soon as Auto-Function is started, BUSY becomes active, which means that transistor is switched OFF.
7	OK	This output reflects result of last action. It is valid after BUSY turned back to passive state. The output transistor is switched ON to reflect OK state.
5	GND	Common signal ground

Table 3.1: Pins for handshake control



3.3 ASCII command interface

3.3.1 Introduction

Flasher supports a simple ASCII command interface via RS-232. Once set up using Flasher PC software, Flasher can be driven by any application or just a simple terminal using ASCII commands.

Every known command is acknowledged by Flasher and then executed. After command execution, Flasher sends an ASCII reply message. If an unknown command is received, Flasher responds with #NACK.

General command and reply message format

Any ASCII command has to start with the start delimiter #. Any ASCII command has to end with simple carriage return (ASCII code 13). Commands can be sent upper or lower case. Reply messages from Flasher are sent back using the same format. All characters are upper case in reply messages.

Communication port settings

Flasher is driven via RS-232 serial port with the following interface settings:

- 8 data bits
- ODD parity
- 1 stop bit
- 115200 baud

3.3.2 Commands to Flasher

The following commands are supported by current version of Flasher firmware:

#AUTO

The Auto command behaves exactly as the start button or external remote control input.

- Flasher starts clearing target, if "Automatic clear before program" option was set.
- Flasher programs target CPU.
- Flasher verifies target CPU.
- Flasher verifies its internal CRC to validate the programming operation.

Finally, Flasher responds with:

- #OK if no error occurred
- #ERRxxx if any error occurred during operation. xxx represents the error code, normally replied to Flasher PC software. The #ERRxxx message may be followed by an additional error text.

During execution of the AUTO command, Flasher automatically sends "status" messages via RS-232 to reflect the state of execution. Typically during execution of #AUTO command, Flasher will reply the following sequence of messages:

```
#ACK
#STATUS:START CMD
#STATUS:CONNECTING
#STATUS:CLEARING
#STATUS:WRITING
#STATUS:VERIFYING
#STATUS:VERIFY FLASHER CRC
#STATUS:READY
#OK
```

#AUTO NOINFO

This command may be used instead of AUTO, if no status messages from Flasher should be sent during execution. The command ends with #OK or #ERRxxx

#BLANK

This command can be sent to perform a blank check over all selected target Flash sectors.

Flasher will reply the following sequence of messages:

```
#ACK
#STATUS:START CMD
#STATUS:CONNECTING
#STATUS:RESET
#STATUS:CONNECTING
#STATUS:BLANK CHECK
#STATUS:READY
#OK
```

in case of blank check failure, Flasher will respond with:

```
#ERR004:BLANK CHECK FAILED.
```

#CLEAR

This command can be sent to clear all selected target Flash sectors.

Flasher will reply the following sequence of messages:

```
#ACK
#STATUS:START CMD
#STATUS:CONNECTING
#STATUS:RESET
#STATUS:CONNECTING
#STATUS:CLEARING
#STATUS:READY
#OK
```

in case of an erase failure, Flasher will respond with:

```
#ERR001:ERASE FAILED.
```

or an other error message if any communication error occurred.

#START

This command can be sent to release Flashers target interface. All signals from Flasher to target will be set into high-Z mode, reset of target will be released. May be used to start target application program.

Flasher will reply the following sequence of messages:

```
#ACK
#STATUS:START CMD
#STATUS:READY
#OK
```

#CRC

This command is useful to identify or verify Flasher internal data which should be programmed to target CPU.

Flasher responds with the CRC of its current memory content:

- #CRC:xxxx if data is present. xxxx is a 4 digit HEX number.
- #CRC:---- if no valid data is present in Flasher memory

#STATUS

This command can be sent any time, even during other command execution. Flasher responds with its current state. All defined state messages are described under *Reply messages from Flasher* on page 37 later on in this manual.

#PROGRAM

This command can be used instead of #AUTO to program a target without final verification. Whether the target is cleared before programming depends on options which were set during configuration using the Flasher PC software.

#VERIFY

This command can be used to verify the target Flash content against the data stored in Flasher.

#RESULT

This command can be sent any time, even during other command execution. Flasher responds with the last result of a previous executed command, or the result of a running command.

#CANCEL

This command can be sent to abort a running program. It may take a while until the current program is actually canceled. Flasher will respond with:

#ERR007:CANCELLED.

3.4 Reply messages from Flasher

The reply messages from Flasher follow the same data format as commands. Any reply message starts with ASCII start delimiter #. Any reply message ends with simple carriage return (ASCCI code 13).

The following reply messages from Flasher are defined:

#ACK

Flasher replies with #ACK message on reception of any defined command before the command itself is executed.

#NACK

Flasher replies with #NACK, if an undefined command was received.

#OK

Flasher replies with #OK, if a command other then #CRC, #STATUS or #RESULT was executed and ended with no error.

#CRC:xxxx

Flasher replies with #CRC: after calculating the CRC of its internal data memory. The CRC xxxx is sent back as four digit HEX number.

If Flasher does not contain valid data, Flasher replies with #CRC:----.

#STATUS:

Flasher replies with its current state. The following status messages are currently defined:

Message	Description
#STATUS:READY	Flasher is ready to receive a new command.
#STATUS:INITIALIZING	Flasher performs self check and internal init.
#STATUS:CLEARING	Flasher is clearing target CPU.
#STATUS:WRITING	Flasher is writing (programming) target CPU.
#STATUS:READING	Flasher reads target CPU.
#STATUS:VERIFYING	Flasher verifies target CPU.
#STATUS:BLANK CHECK	Flasher performs blank check of target CPU.
#STATUS:IDLE	Flasher is currently idle, but current command is not completed yet.
#STATUS:START CMD	Flasher starts execution of a received command.
#STATUS:ERASE-VERIFY	Flasher performs an Erase Verify operation. Required for some CPUs only.
#STATUS:VERIFY FLASHER CRC	After programming and verifying target, Flasher verifies its internal data memory by recalculating and verifying the CRC over all data.
#STATUS:CONNECTING	Flasher initializes connection to target CPU.

Table 3.2: Status messages

#ERRxxx

If any command other then #CRC, #STATUS or #RESULT was terminated with an error, Flasher cancels the command and replies with an error message instead of #OK message. The three digit error number xxx is the same error code number which would normally be sent back to Flasher PC software. The error code numbers are described in the Flasher user manual.

Some error codes may be followed by colon and an additional error text.

For example:

#ERR007:CANCELED.

Chapter 4

Batch mode

This chapter explains how to operate the Flasher using command line options.

4.1 Introduction

Flasher.exe supports command line options to enable automated programming of targets. This chapter describes the supported commands and their respective parameters. Flasher PC software version 1.72b or above replaces FlasherPro.exe, which is not delivered anymore.

How to start Flasher PC software in batch mode

To use Flasher in batch mode, just call Flasher.exe directly from DOS-Box or start any batch file which calls Flasher.exe, To start any action, parameter may be passed as command line options.

General rules

- The first parameter specified must be the file to load, if download is required.
- The return code is 0 if all operations have been executed successfully, !=0 otherwise.
- All commands are identical to the corresponding commands in the menu bar.
- All commands are processed from left to right.
- If "-exit" is specified as the last command, the PC software will terminate as soon as any error occurs or after all commands have been executed.
- If one parameter contains a space use quotation marks for this parameter.
- If an option takes an argument no space characters is allowed between the command name and the parameter.

4.2 List of commands

The following commands are currently supported as parameter when Flasher is called in batch mode:

Command	Description
-download	Downloads the loaded hex file into Flashers memory without starting any additional action.
-checkblank	Checks if target is blank
-verify	Verifies loaded data with contents of target
-clear	Clears target area, selected memory banks
-clearchip	Clears entire chip. Clearing the entire chip might not be implemented for all devices. Then the command behaves the same as the clear command.
-programverify	Programs & verifies target
-program	Programs loaded data into target
-readback	Reads target area into PC
-start	Starts application program (serial mode only, only Flasher MV3)
-com<PORT>	Sets COM-port of PC (1..4).
-saveas <FILENAME.EXT> [, FIRST-LAST]	Saves the file currently in PC memory. The extension needs to be "MOT", "HEX" or "BIN" and determines the file type. The optional range is used for files in BIN-format.
-merge<FILENAME.EXT>	Merge specified file to current data
-delrange<FIRST-LAST>	Deletes the specified range of data
-relocate<OFFSET>	Relocates current data by offset
-selbanks<START,END>	Sets the numbers of start and end bank. The numbers have to be the same as in the selection box shown under "Options Device".
-selmultiple <BANK1, BANK2 ? BANKn>	Selects individual banks. The numbers have to be the same as in the selection box shown under "Options Device".
-seldevice<DEVICENAME>	Selects the desired device. The name of the device has to be exactly the same as in the selection box shown under "Options Device".
-password <LENGTH, ID1, ID2, . . . , IDn>	Sets the ID-bytes
-setFILL<xx>	Sets the fill byte, the parameter is a two digit hex number.
-setIfSync	Selects the serial synchronous interface for the target.
-setIfAsync	Selects the serial asynchronous interface for the target communication. This command is available for Flasher 5 PRO only.

Table 4.1: List of commands

Command	Description
-setIfSpeed<Speed>	<p>Sets the target interface communication speed. Valid speed settings are:</p> <ul style="list-style-type: none"> • 500000Baud • 250000Baud • 125000Baud • 55555Baud • 38400Baud • 19200Baud • 9600Baud • Fast • Medium • Slow • VerySlow. <p>The parameter used should be a valid parameter as shown in the device selection dialog for the selected device. The command and parameter have to be passed as one word without any spaces.</p> <p>For example: -setIfSpeed500000Baud</p> <p>Passing wrong settings not allowed for the selected device, may result in unexpected interface speed settings.</p>
-openfile<FILENAME.EXT>	opens a HEX file
-setACBP	Sets option 'Automatic clear before program'.
-setCOD	Sets option 'Clear on demand'.
-setDE	Sets option 'Detailed errorlevel on return'.
-setDNVBP	Sets option 'Do not verify blank pages'.
-clearACBP	Clears option 'Automatic clear before program'.
-clearCOD	Clears option 'Clear on demand'.
-clearDE	Clears option 'Detailed errorlevel on return'.
-clearDNVBP	Clears option 'Do not verify blank pages'.
-index<X>	Allows selection of an individual setup data set for Flasher. This is useful when multiple Flasher should be handled concurrently with one PC. After setting up all options together with the index command, further calls of Flasher with the same index command will restore all previous settings for that index.
-ip<SN IPADDR>	Selects Ethernet as communication interface. The Flasher to connect to can be specified either as serial number or IP address in "dotted" format.
-usb<SN>	Select USB as communication interface. SN is the serial number of the Flasher to connect to.
-exit	Finish execution after performing all commands
-help	Displays available commands.
-?	The same as -help

Table 4.1: List of commands

Note: To open a HEX file, place the file name as first parameter just behind the call of Flasher.exe, the -openfile command is not required when only one file should be opened.

Return values

The following return values are sent as errorlevel unless "Options | Filling & misc. | Detailed errorlevel on return" is selected:

Value	Meaning
2	Target not blank
3	Verify error, Contents of target data not identical to FLASHERs internal data.
4	Erase error. Target CPU could not be erased.
5	Error during Program & Verify function
6	Error during programming of target CPU
7	Error during target readback.
8	Error during "Start application".
9	Timeout error.
10	HEX file could not be opened.
>10	Corresponds to error number which would normally shown on PC screen when program was used in normal mode.

Table 4.2: List of return values

When "Detailed errorlevel on return" is set as option, the returned value equals to error codes described under *Error messages* on page 27.

4.3 Examples

4.3.1 Program and verify

In the following example the software:

- reads the file TEST.MOT,
- tells the Flasher to program and verify,
- exits.

```
Flasher.exe test.mot -programverify -exit
```

4.3.2 Programming using specified com port

In the following example the software

- reads the file TEST.MOT,
- sets the COM-port of the PC to Port 2,
- tells the Flasher to program,
- exits.

```
Flasher.exe test.mot -com2 -program -exit
```

4.3.3 Usage in production environment

This example shows a batch file that executes a download to the Flasher once. After the first call of Flasher the batch file runs in a loop that programs the last downloaded file continuously after pressing a key. After every execution of Flasher the return code is evaluated.

Note: Evaluating the return code only works under Windows NT or Windows 2000. If you use Windows 9x Flasher will be started in a new task and you can not evaluate the return code from the DOS-box from which you started Flasher, because your DOS-Box will not halt execution until Flasher finished!

```
@echo off
rem The first call of Flasher loads the the HEX file
Flasher test.mot -program -verify -exit
goto checkerror
rem Loop for repeated programming without download
:loop
pause
Flasher -program -verify -exit
rem Check errorlevel if succeed
:checkerror
if errorlevel 1 goto ERROR_%errorlevel%
echo Operation successfully finished
goto loop
rem handle errors
:ERROR_1
echo Undefined error
goto end
:ERROR_2
echo ERROR_TARGET_CHECKBLANK
goto end
:ERROR_3
echo ERROR_TARGET_VERIFY
goto end
:ERROR_4
echo ERROR_TARGET_CLEAR
goto end
```

```

:ERROR_5
echo ERROR_TARGET_AUTO
goto end
:ERROR_6
echo ERROR_TARGET_PROGRAM
goto end
:ERROR_7
echo ERROR_TARGET_READBACK
goto end
:ERROR_8
echo ERROR_TARGET_STARTAPPLICATION
goto end
:ERROR_9
echo ERROR_TARGET_TIMEOUT
goto end
:ERROR_10
echo ERROR_OPENDOCUMENTFILE
:end
echo Operation canceled

```

4.3.4 Relocate, delete, merge and save

The following example shows a batch file, where the software

- reads file A.HEX,
- deletes the range 0xFC0000..0xFFFFF,
- moves the remaining bytes 0x40000 bytes up,
- saves the memory contents as B.MOT,
- reads the file A.HEX again,
- deletes the range 0xC00000..0xCFFFFF,
- moves the remaining bytes 0x340000 bytes down,
- saves the memory contents as C.MOT,
- merges the files B.MOT and C.MOT
- and save the new file as RESULT.BIN.

```

Flasher A.HEX -delrangeFC0000-FFFFFF -relocate040000 -saveasB.MOT -exit
Flasher A.HEX -delrangeC00000-CFFFFF -relocate340000 -saveasC.MOT -exit
Flasher B.MOT -mergeC.MOT -saveasRESULT.BIN,C00000-CFFFFF -exit

```

4.3.5 Selecting a device and setting ID bytes

The following example shows a batch file, where the software

- selects a device,
- sets the ID bytes to FF,FF,FF,FF,FF,FF,FF
- and reads the contents of the device.

```

Flasher -seldeviceM30201F6xx -password7,FF,FF,FF,FF,FF,FF,FF -readback

```


Chapter 5

Hardware

This chapter gives an overview about Flasher specific hardware details, such as the pinout of the target interface.

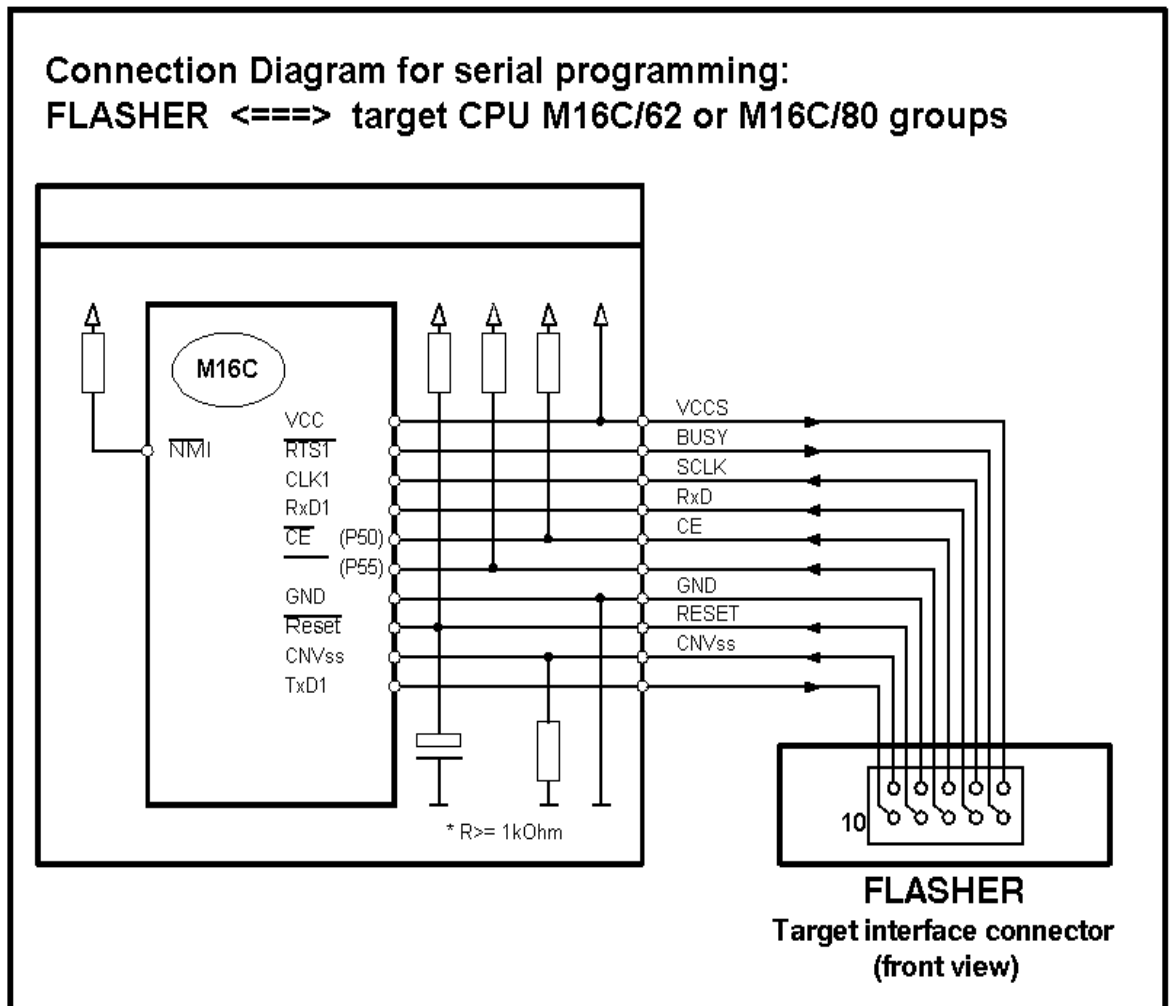
5.1 Target interface for M16C/62, M16C26P, M16C/80, M32C and R32C

The clocked synchronous interface from Flasher to the target system is built with a 10 pin dual in line pin connector, (pin1 is marked at the connector) at the front of Flasher. The function depends on selected target.

Pin No.	Signal	Function for M16C/62, M16C/80	Specification / remarks
1	VCCS	Positive supply voltage of target	Input 3.0..5.5V to supply the interface
2	BUSY	Target CPU Busy signal output	Flasher Input with Pull-Up to internal 3.3V
3	SCLK	Target CPU Serial clock (input)	Flasher Output, CMOS driver via 220 Ohms
4	RxD	Target CPU Serial data input	Flasher Output, CMOS driver via 220 Ohms
5	CE	Chip enable signal of target CPU	Flasher Input / Output
6	EPM	EPM signal of target CPU	Flasher Input / Output
7	GND	Common signal ground	---
8	RESET	RESET signal of target system	Flasher Output, CMOS driver via 220 Ohms
9	CNVss	Target CPU CNVss signal	Flasher analog Output
10	TxD	Target CPU Serial data output	Flasher Input / Output

Table 5.1: Target interface pins

If the RESET of the target system is driven by a reset circuitry with active high driver, the RESET output of Flasher must not be connected directly to the CPU reset of the target. For M16C/62 or M16C/80 targets you do not have to connect RESET to Flasher; you can always manually reset your target system after connecting Flasher.



Caution:

Before connecting the target with Flasher, ensure that there is NO difference in the ground potential between Flasher and target.

When the Flasher is connected to a PC via RS232C cable, ensure that the PC and the target operate on the same ground potential. Connect the ground lines from PC and your target before connecting the Flasher to the target. If a ground potential difference between Flasher and target exists, the Flasher may be damaged.

5.1.1 Serial programming, technical details for M16C/6x, M16C/80, R32C

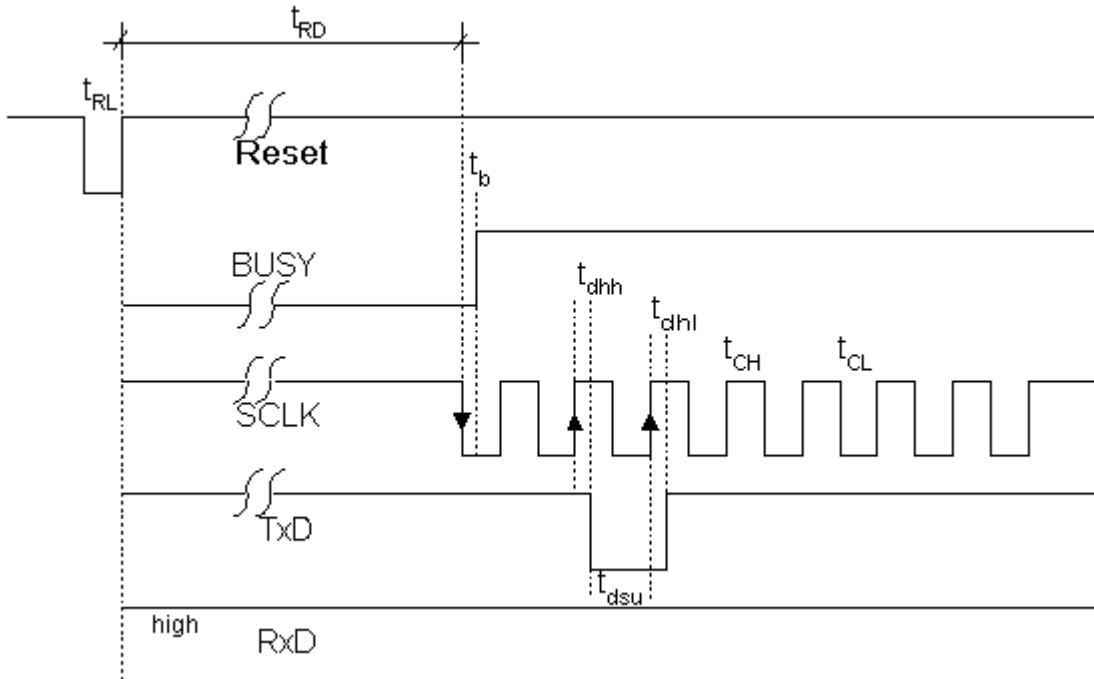
Serial programming uses a clock synchronous interface. 8 bits of data (1 byte) is transferred at a time. The commands which are used are described in the Renesas manuals. In general, the sequence is as follows:

Flasher resets the target system by pulling the /Reset line low for a period of time which is set as Reset active time in "Options | Setup" dialog of PC software. Flasher waits for the Reset inactive time, nominal 500 ms, (t_{RD}) in order to allow the target system to recover from reset. This time can also be set in "Options | Setup" dialog via PC software.

Flasher checks the BUSY line. If it is active (high level) Flasher stops with the error message 40: Target chip says "BUSY" because it can not communicate with the target system.

Flasher outputs one clock (clock changes from high to low and back). BUSY should now be active (high). If it is not active, Flasher stops with error message 41: Target chip: Busy does not react. Flasher outputs 7 more data bits (7 clock cycles) and waits for BUSY to go low.

More data bytes are output (or read) the same way.



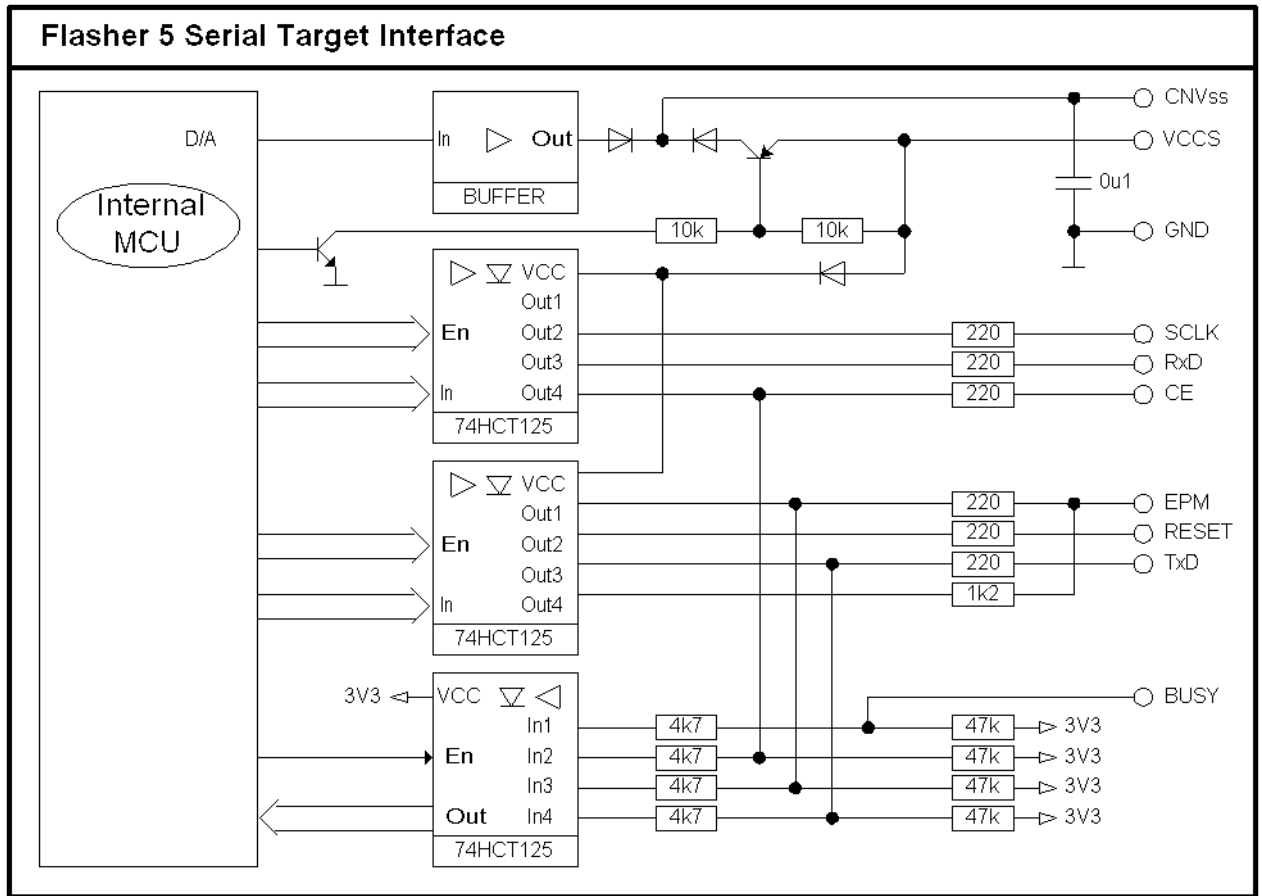
t_{RL} : nominal 20 ms

t_{RD} : nominal 500 ms

The reset active time (t_{RL}) and reset inactive (e.g. reset delay) time (t_{RD}) can be set in "Options | Setup" dialog if required.

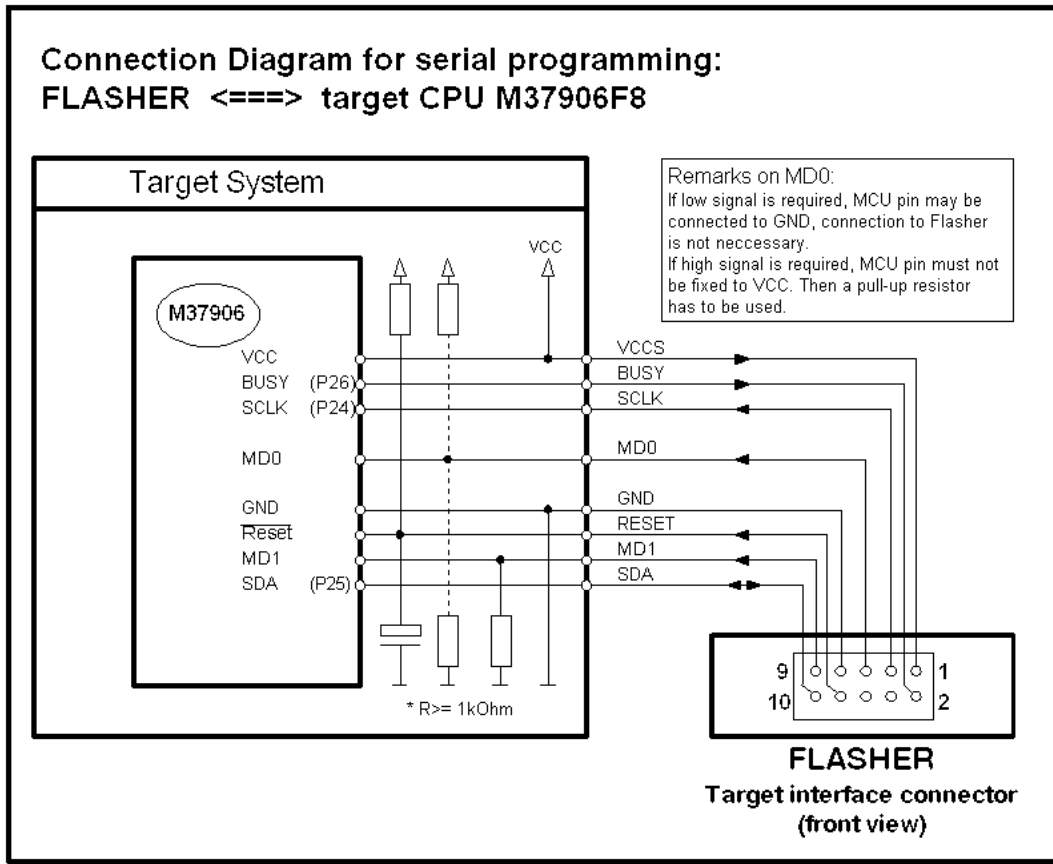
5.2 Serial target interface circuitry

The following schematic shows the target interface of Flasher

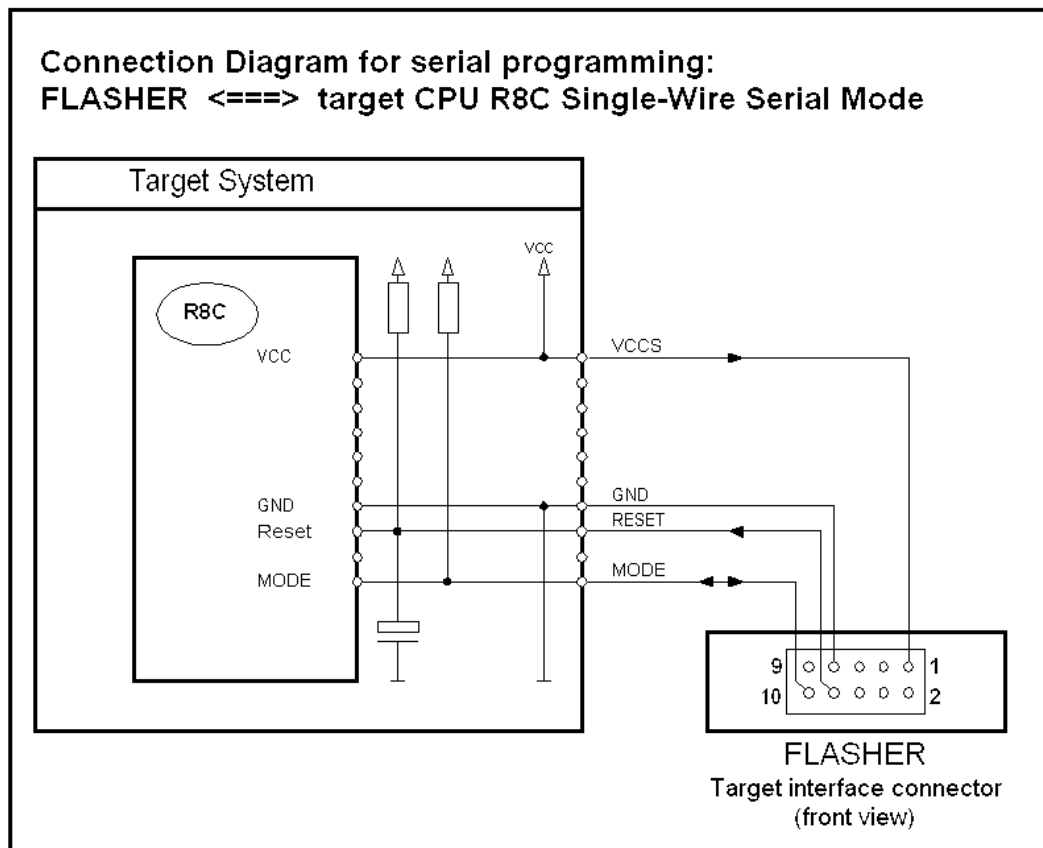


5.3 Target interface for other systems

Target system interface for M37906F8:



Target system interface for R8C:



5.4 CRC calculation used in Flasher and PC software

The Flasher PC software and Flasher calculate a CRC over all data downloaded to the Flasher. The CRC is used to verify correct data transfer as well as an integrity check of target-data stored in the on board non-volatile memory.

The CRC is calculated over all bytes of all selected flash sectors of the target device. The calculation is compatible to the algorithm used in the CRC generator circuit inside a Renesas M16C/62P device. The Renesas application note REU05B0007-0100Z describes how the CRC is calculated.

To calculate the CRC of a target application you have to perform the following steps:

1. Create a buffer, large enough to hold all bytes of all selected flash banks of the target.
2. Fill the whole buffer with the fill-byte which is defined as fill byte in the Flasher Filling & misc. options.
3. Parse your hex-file and fill all defined bytes into the buffer. Bytes not defined in the hex-file were set to the fill byte in step 2.
4. Calculate the CRC over the whole buffer following the algorithm from the application note, or the following sample code.

Alternatively, the CRC over all bytes may be calculated without a buffer, when a function delivers all bytes of all selected flash banks, including fill bytes, in ascending address order from a stream.

Example code for CRC calculation

```
//
// CRC_Calc1_M16() calculates the CRC for one byte.
// Before calling the function the first time, set the CRC to 0x0000
//
void CRC_Calc1_M16(unsigned int* pCRC, unsigned char Data) {
    unsigned int crc;
    unsigned int x16;
    int i;

    crc = *pCRC;
    for (i = 0; i < 8; i++) {
        if ((crc & 0x0001) ^ (Data & 0x01)) {
            x16 = 0x8408;
        } else {
            x16 = 0x0000;
        }
        crc = crc >> 1;
        crc ^= x16;
        Data = Data >> 1;
    }
    *pCRC = crc;
}

//
// CRC_Calc() calculates the CRC over NumBytes bytes in a buffer
//
unsigned int CRC_Calc(unsigned char* pBuffer, unsigned long NumBytes) {
    unsigned int CRC_M16;
    unsigned long i;

    //
    // Initialize CRC and calculate the CRC over all bytes
    //
    CRC_M16 = 0x0000;
    for (i = 0; i < NumBytes; i++) {
        CRC_Calc1_M16(&CRC_M16, ((char) pBuffer[i]));
    }
    return CRC_M16;
}

//
// The following example shows how to calculate the CRC over all bytes
// in a buffer as described above using the CRC_Calc() function
//
unsigned long NumBytes;
unsigned long i;
unsigned char* pBuffer;
unsigned char Fillbyte;
unsigned int CRC;
```

```
NumBytes = SUM_OF_ALL_BYTES; // Number of bytes of all selected flash sectors
//
// Create a buffer for all bytes
//
pBuffer = (unsigned char*) malloc(NumBytes);
//
// Initialize the buffer, fill up with the fill bytes
//
if (pBuffer != NULL) {
    Fillbyte = FLASHER_FILL_BYTE; // The fill byte set in Flasher options
    memset(pBuffer, Fillbyte, NumBytes);
    //
    // Fill the buffer with data. This has to be done by a function that
    // parses the Hexfile and addresses the buffer according the address
    // offset which depends on the selected flash sectors
    //
    ParseFile(pBuffer, NumBytes);
    //
    // Initialize CRC and calculate CRC over all bytes in the buffer
    //
    CRC = CRC_Calc(pBuffer, NumBytes);
}
```

Remarks

The first byte in the buffer has to be the first byte of the first selected flash sector, regardless the address of the sector. For example, if the first selected sector has address 0x3000, the first byte in the buffer (offset 0) is the byte at address 0x3000 in the target device.

If the hex file addresses only some of the bytes in a flash sector, all the other bytes have to be filled up with the fill byte. For example, if the selected flash sector has 4096 bytes, starting from address 0x3000 and the hex-file only contains data for the first 16 bytes, the whole area from 0x3010 to 0x3FFF has to be filled up with the fill byte.

All selected sectors are stored in ascending address order without any gap, regardless the start address of the flash sectors. For example, if the first sector starts at address 0x3000 and has a total size of 0x1000, the second selected sector starts at 0x8000 and has a total size of 0x1000, the buffer to hold the data needs a size of 0x2000 bytes.

The first byte from the first sector is stored at offset 0 in the buffer, the first byte of sector 2 is stored at offset 0x1000.

Chapter 6

Support

This chapter contains troubleshooting tips together with solutions for common problems which might occur when using Flasher. There are several steps you can take before contacting support. Performing these steps can solve many problems and often eliminates the need for assistance.

6.1 Trouble shooting

Proper operation of Flasher in serial mode depends on your target system. If you have any trouble operating Flasher in serial mode, please:

1. Check your target hardware.
2. Check the connecting cable.
3. Use an oscilloscope to check the state of all signals on the target connector, especially to check if the target CPU is RESET properly and the target CPUs BUSY signal works properly.

6.2 Known limitations

Older versions of the Renesas M16C/62 bootloader sometimes do not start after RESET. If you experience problems in communicating with the target system, power down the target system, power it up and try again.

6.3 Contacting support

For support questions, please consult our website at www.segger.com. If this does not answer your questions, please send an email to support@segger.com containing the following information:

- A detailed description of the problem
- Flasher serial number
- Information about your target hardware (processor, board, etc.).

Chapter 7

Glossary

This chapter explains important terms used throughout this manual.

DHCP

Acronym for Dynamic Host Configuration Protocol. A communication protocol for the distribution of network configuration such as IP address.

Ethernet

A standard for data transfer on local area networks.

Host

A computer which provides data and other services to another computer. Especially, a computer which communicates with the Flasher.

In-Circuit programming

A method of writing the program memory of a microcontroller while soldered on a target board. The alternative is to first program the microcontroller and then to solder it.

IP

Acronym for Internet Protocol. A communication protocol to exchange data on a network.

IP address

Unique number which identifies a network device connected via IP.

LAN

Acronym for Local Area Network. A network of computers which covers a small area.

On-chip flash memory

Internal non-volatile program memory of a microcontroller which can be programmed using the Flasher.

RS232

A standard for a serial interface which can be used to transfer data between electronic devices.

USB

Acronym for Universal Serial Bus. A standard for a high-speed serial interface which can be used to transfer data between electronic devices.

Index

B		V	
Blank check	22	Verify	22
C		W	
Clear	22	Web server	17
COM port	15		
D			
DHCP server	17		
E			
Ethernet	14		
I			
ID bytes	24		
ID check	24		
Installing the Flasher PC-software	11		
IP address	16		
L			
LED status	26		
O			
On-chip flash memory	10		
P			
Program	22		
R			
RS	14		
S			
Serial number	16		
Stand-alone mode	26		
Supported microcontrollers	10		
U			
USB	19		

