
RFlasher

Raisonance Flash Programming Software.



Overview and
Getting Started

November 2005

RAISONANCE

Table of Contents

1. INTRODUCTION	5
2. FIRST STEPS WITH RFLASHER	6
2.1 Hardware configuration	7
2.2 File loading	8
2.3 Memory view	8
2.4 Programmer-specific options	9
2.5 Blank check	9
2.6 Erase	10
2.7 Program	10
2.8 Reset and run the program	10
2.9 Verify	10
2.10 Statistics	10
2.11 Reload memory	10
2.12 Read	11
2.13 Save the memory dump	11
3. AUTO MODE	12
3.1 Hardware configuration and file loading	13
3.2 Auto mode for one device	13
3.3 Auto mode for a batch of the same devices	13
4. USING PROJECTS	15
4.1 Create a new project	16
4.2 Add nodes to the project	17
5. PROGRAMMER OPTIONS	18
5.1 ST7	19
5.1.1 ICC with RLink	19
5.1.1.1 RLink options section	19
5.1.1.2 ICC Mode Entry section	19
5.1.1.3 Option Bytes Options section	20
5.2 STR7	20

5.2.1 JTAG with RLink-STR7 or JTAGJet	20
5.2.1.1 Test connect section	21
5.2.1.2 JTAG Clock section	21
5.2.1.3 Sector Selection section	21
5.2.1.4 JTAG Chain Description section	21
5.2.2 UART bootloader with COM port	22
5.2.2.1 Presentation	22
5.2.2.2 Options	22
5.2.2.3 Operation	22

Glossary

- *Programmer / programming tool*: the piece of hardware used to send some code from a workstation to the target microcontroller, i.e. to write the code into the microcontroller Flash memory.
- *Target*: the microcontroller or embedded device on which the current program is meant to run. The target is connected to the workstation running RFlasher via the programming tool.

1. Introduction

RFlasher is a user-friendly Windows interface that allows you to program object files to a microcontroller's flash memory.

You can connect to microcontrollers in order to manage their non-volatile memory: erase, blank-check, program from the PC memory used by RFlasher, or read-out to the PC memory used by Rflasher.

You can load object files into the PC memory that is used by RFlasher and then view, edit them and save them.

RFlasher also provides a configurable automatic mode for grouping commands (for example, erase and program in one single click) and for mass programming (for example programming multiple devices in sequence).

RFlasher provides a third mode of operation: project mode. The project mode is slightly more complex to use but offers more possibilities such as allowing you to save the configuration for each application in case you have several different applications to handle. It also allows you to create projects from several object files.

RFlasher can be used to program microcontrollers in any of several supported families using the connection protocol specific to each. The list of supported devices is not fixed and will increase as time passes; it is available in the target selection window.

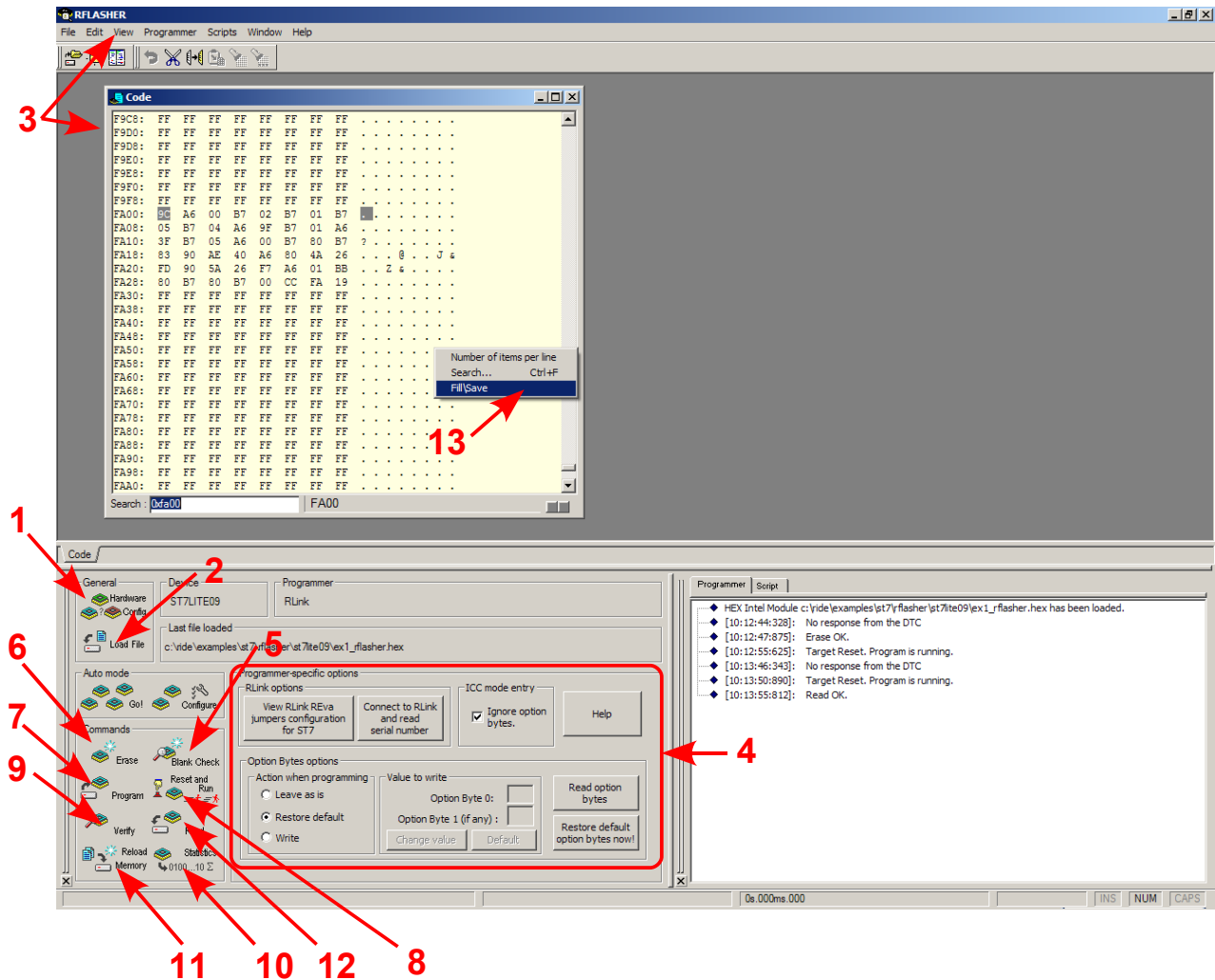
This document provides information about RFlasher features that are common to all supported targets and/or programming tools, general user information for the supported tools and device families. For details about features that are specific to your target and/or programming tool, please refer to the on-line help, after having selected your programming tool.

Note for RIDE users:

RFlasher is simply a special, limited mode of RIDE (Raisonance Integrated Development Environment). Therefore, everything that RFlasher does can also be done with RIDE. Among other things, RIDE can create and open RFlasher projects, program and erase devices, etc. The opposite, however, is not true.

2. First steps with RFlasher

This chapter walks you through the steps for programming your target with RFlasher, the RIDE programming interface. These steps are presented in the order that they are typically used.

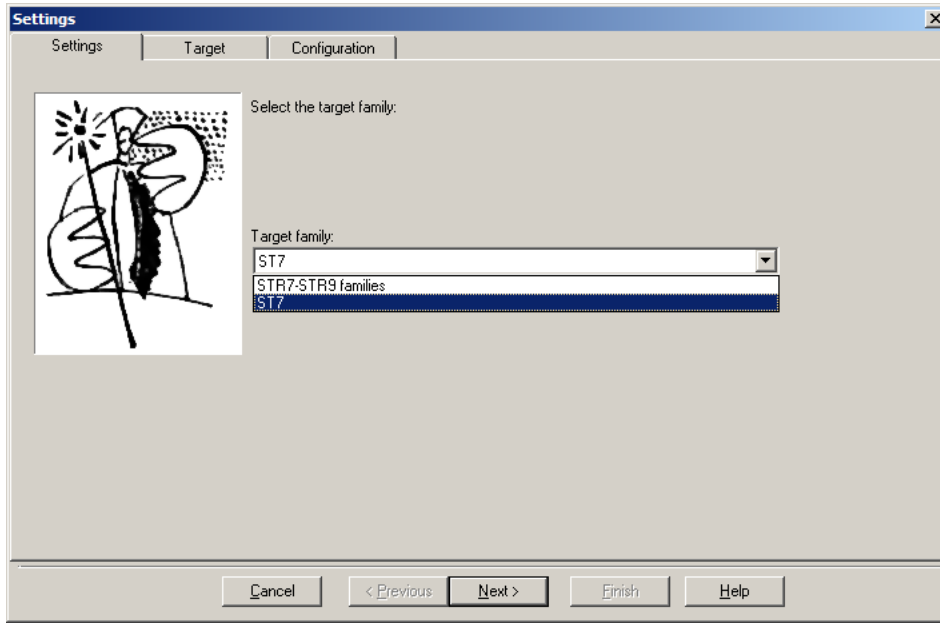


These numbers represent both the order of the steps, and the section number for reference in this document. For example, step n^o4 is described in the section 2.4.

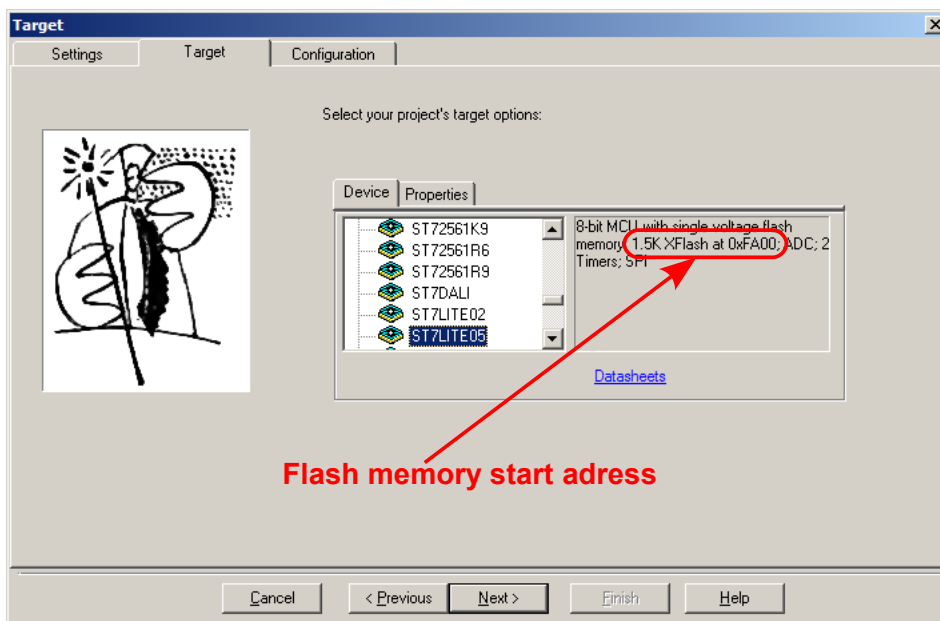
2.1 Hardware configuration



The first time you launch RFlasher, the hardware configuration window is automatically opened; otherwise, click on the "Hardware Config" button. The Settings dialog box appears as shown below:

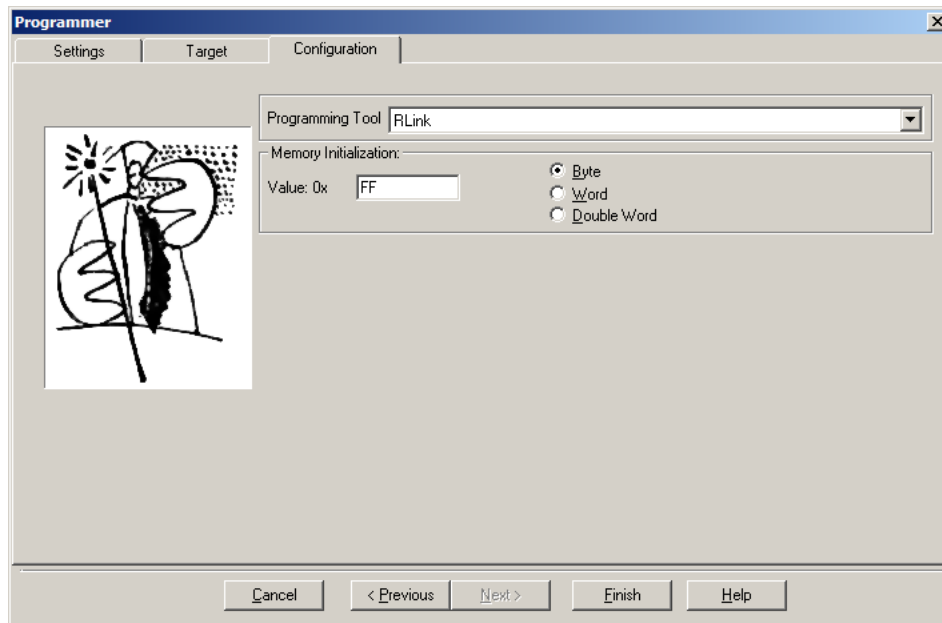


Select your target family, then click "Next" to go to the target tab. In the Target tab, select your target derivative and note the start address for the flash memory shown in the device description, then click "Next".



Important: note the Flash memory start address given in the description field for future reference.

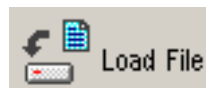
In the Configuration tab, choose your programming tool. You can also enter a memory initialization value (for advanced users).



When your hardware configuration is complete, click on **"Finish"** to close the window and go to the next step.

2.2 File loading

This button allows you to load the content of an object file into the PC memory that is used by RFlasher prior to programming the target device's memory.



You can load a file in any of the formats that Raisonance tools recognise as ready to program (HEX, ELF, BIN, AOF ...).

Select your file, then click **"Open"**.

In the Options dialog box, click on **"OK"** - do not modify the default options.

The name of the file you have loaded now appears in the "Last file loaded" field.

Sample HEX files, are provided in the RIDE install directory. For standard installations of RIDE and RFlasher go to:

```
c:\ride\examples\<<family>\RFlasher\<<derivative>\ex1_RFlasher.hex.
```

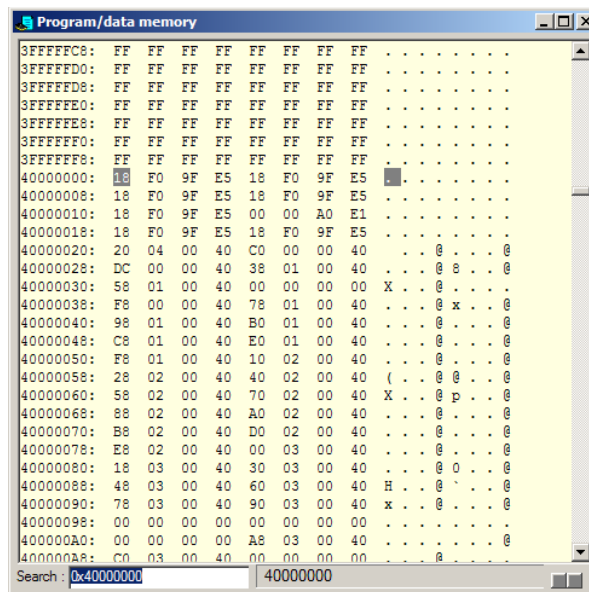
(These examples are designed to run on the REva board with the LED jumpers enabled.)

2.3 Memory view

To view the memory with the loaded file prior to programming, select **"Data Dump... -> Memory view"** in the **View** menu.

In the window that appears, enter the Flash memory start address (noted in step 2.1) in the **"Search"** field. Use hexadecimal format ("**0x**"): for example, type **"0x40000000"**.

This window shows the code loaded in the PC memory that is used by RFlasher prior to programming the target. At this stage, the display is not an image of your device's memory as the device has not yet been programmed. The file has only been loaded in the PC memory used by RFlasher.



```

3FFFFFFC8: FF FF FF FF FF FF FF FF . . . . .
3FFFFFFD0: FF FF FF FF FF FF FF FF . . . . .
3FFFFFFD8: FF FF FF FF FF FF FF FF . . . . .
3FFFFFFE0: FF FF FF FF FF FF FF FF . . . . .
3FFFFFFE8: FF FF FF FF FF FF FF FF . . . . .
3FFFFFFF0: FF FF FF FF FF FF FF FF . . . . .
3FFFFFFF8: FF FF FF FF FF FF FF FF . . . . .
40000000: 18 F0 9F E5 18 F0 9F E5 . . . . .
40000008: 18 F0 9F E5 18 F0 9F E5 . . . . .
40000010: 18 F0 9F E5 00 00 A0 E1 . . . . .
40000018: 18 F0 9F E5 18 F0 9F E5 . . . . .
40000020: 20 04 00 40 C0 00 00 40 . . . . .
40000028: DC 00 00 40 38 01 00 40 . . . . .
40000030: 58 01 00 40 00 00 00 00 X . . . .
40000038: F8 00 00 40 78 01 00 40 . . . . .
40000040: 98 01 00 40 B0 01 00 40 . . . . .
40000048: C8 01 00 40 E0 01 00 40 . . . . .
40000050: F8 01 00 40 10 02 00 40 . . . . .
40000058: 28 02 00 40 40 02 00 40 ( . . . . .
40000060: 58 02 00 40 70 02 00 40 X . . . .
40000068: 88 02 00 40 A0 02 00 40 . . . . .
40000070: B8 02 00 40 D0 02 00 40 . . . . .
40000078: E8 02 00 40 00 03 00 40 . . . . .
40000080: 18 03 00 40 30 03 00 40 . . . . .
40000088: 48 03 00 40 60 03 00 40 H . . . .
40000090: 78 03 00 40 90 03 00 40 x . . . .
40000098: 00 00 00 00 00 00 00 00 . . . . .
400000A0: 00 00 00 00 A8 03 00 40 . . . . .
400000A8: C0 03 00 40 00 00 00 00 . . . . .
Search : 0x40000000 40000000

```

2.4 Programmer-specific options

A part of the main window panel is reserved for programmer-specific options. You can refer to chapter 5 page 18 for your programming tool. There is often a help button in this section that you can click to get further information about programming-tool specific options.

You must make sure that these options are correctly set before attempting **any** communication with the device.

You are about to communicate with the chip for the first time, so please make sure that the Programmer is connected to the PC and the target board, and that the target board is powered. Some programmers provide connection tests in their programmer-specific options. If this is the case with yours, please use them to check the connections and options.

2.5 Blank check

This button allows you to check if the device is blank:



After communication with the target device, a dialog box informs you of the result of the operation.

If the device is blank, you can jump directly to step 2.7; otherwise you must erase your device first.

2.6 Erase

Click on this button to erase your device's Flash memory prior to programming it:



You can do a blank check again after erasing to make sure your device is ready.

2.7 Program

You can program your device by clicking on this button:



The program operation consists of loading the content of the PC memory that is used by RFlasher (initialized with "Load File") into the target device's flash. After communicating with the device, a dialog box informs you of the success of the operation.

2.8 Reset and run the program



Click on this button to make the program run on your device.

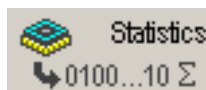
2.9 Verify

Click on the "verify" button to perform a strict verification of the program integrity in the Flash memory by comparing its content with the program loaded in the PC memory, which is visible in the Memory View:



The verification status is given in a dialog box at the end of verification.

2.10 Statistics



Depending on your hardware, you can click on this button to obtain statistics on the target. Usually this includes the CheckSum.

2.11 Reload memory

Close and reopen RFlasher to clear its memory, or click on the "Reload Memory" button, which has the same effect:



If you go to the start address of Flash memory in the memory view, you should not longer see any data.

2.12 Read

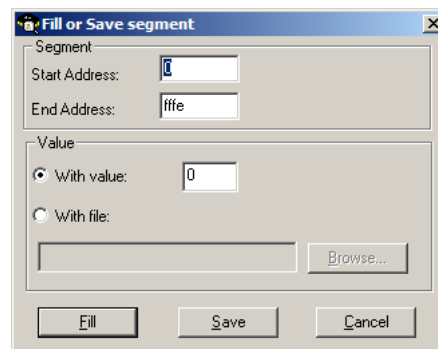
Click on this button to copy the device's Flash memory content to the PC memory used by RFlasher:



This button allows RFlasher to dump (make an exact copy of) the device Flash memory into the RFlasher memory view. To see the Flash memory, go to its start address using the "Search" field.

2.13 Save the memory dump

You can then save the memory dump to a file for future reference. To do this, note the start address and the end address of the memory block you want to save, then right-click on the memory view and choose "Fill/Save". The following dialog box appears.



Enter the start and end addresses, then select "With file" and click on "Browse...". Here you can enter a file name such as "dump.hex", and click on "Open". Back in the "Fill or Save segment" dialog box, click on the "Save" button. The file `dump.hex` is created, containing the memory dump in Intel HEX format.


3. Auto Mode

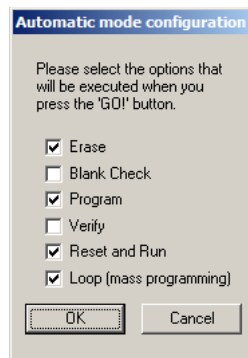
RFlasher provides an Automatic Mode in which you can define a sequence of operations to be executed on a single click. This way, you can save a lot of time when processing a batch of devices for instance.

3.1 Hardware configuration and file loading


Check that your programmer and device are correctly connected, and follow the steps 2.1 to 2.3 beginning on page 7, to prepare the PC memory with the object file that you will load in your device.

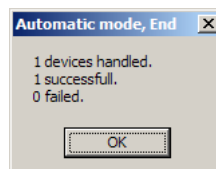
3.2 Auto mode for one device

Click on the "Configure"  button to select the operations to include in your sequence. The following dialog box appears:




Select the options you want, but **uncheck the last one** ("Loop") for now. Click "OK" to validate your choices.

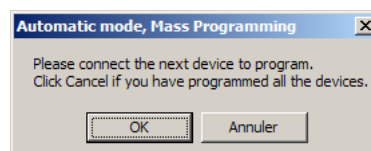
Click on the "Go!"  button to launch the auto mode sequence. The operations are done, then a dialog box indicates the success or failure of the operations:



3.3 Auto mode for a batch of the same devices

Select the "Configure"  button again, and this time check the "Loop" option as if you had several devices to program sequentially with the same program, and click "OK".

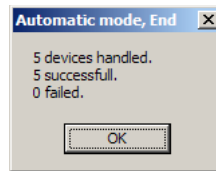
When you select the "Go!" button, it launches the process. At the end of the sequence, a dialog box asks you whether you want to program another device.



Click "OK" to program each device in the batch.

When you have programmed all the devices, click "Cancel" to end the sequence loop.

The dialog box indicates, this time, the statistics of success or failure in programming your devices:



4. Using projects

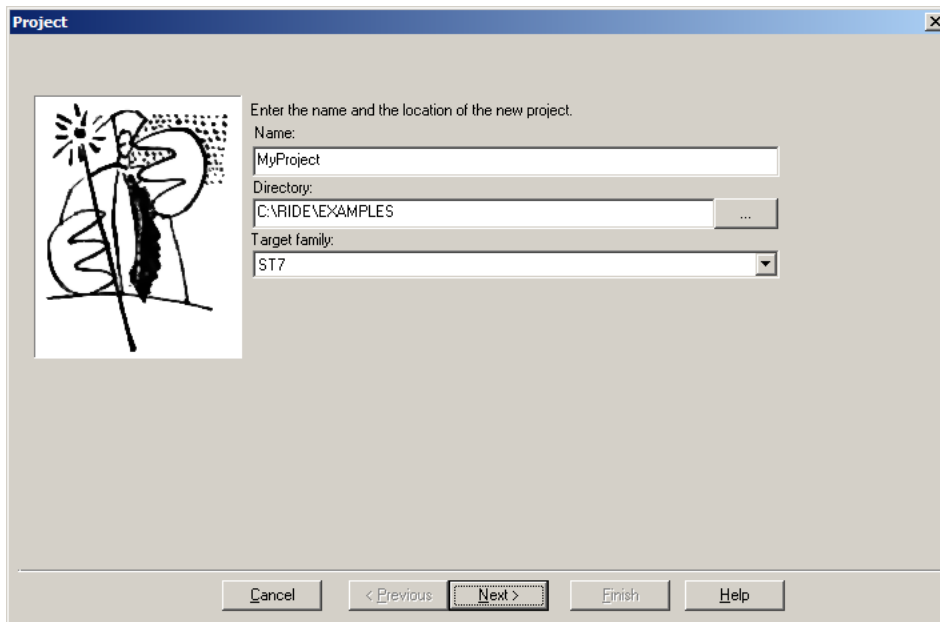
You may need to organize your work when using a programming software: for example to handle different HEX files to load in the same device at different addresses, to insert references to documents, etc.

In this case, you can gather your information and program files in a project for management within Rflasher.

Important remark: one of the main goals of Project management in RFlasher is to provide a way for restoring your configuration: chosen target and associated options, object files to include, etc. If you open a previously created project, all your options will be recovered and your Memory content will be reloaded (i.e. your object files will be mapped in memory).

4.1 Create a new project

In order to create a new project, select **"New -> Project"** from the **"File"** menu. The New Project dialog box appears:

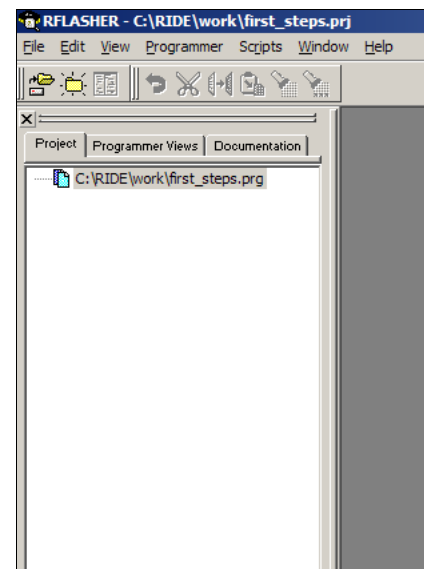


Choose a project name such as **"first_steps"**. Select a working directory (for example **"C:\RIDE\work"**) and a Target family, then click **"Next"**.

The next steps are related to your hardware configuration, please refer to chapter 2, section 2.1, page 7.

After configuring the hardware and clicking on Finish (section 2.1), the project panel should appear on the left side of the main window. It is composed of three tabs, accessible via their titles on top of the active view:

- Project
- Programmer Views
- Documentation



4.2 Add nodes to the project

In order to include programs in your project, follow this procedure:

1. Right-click on the project file (`first_steps.prg`).
2. Select "Insert to project" – a file-selection dialog box opens.
3. Choose the object file to include from anywhere on your disk
(for instance, choose the example object files at
"`c:\ride\examples\<family>\RFlasher\<derivative>\ex1_RFlasher.hex`").
4. Click "Open".

Change the order of objects in the project display by selecting an object and pressing `shift+↑` or `shift+↓` to place your nodes in the expected order.

Repeat these steps for any object file or documentation file you want to include.

Note about object file inclusion:

Including several object files can be useful if you have, for example a HEX file containing the program code and another HEX file containing initialization values to store in Flash memory. In this case, you must ensure (using the Memory View) that your memory blocks don't overlap and that the stored values are in the expected address so that the program can retrieve them.

Note about documentation files:

You can include documentation files in the `PDF`, `HTML` (`.HTM` extension only), `DOC` and `TXT` formats in the Project View (using "Add node"). These files, once included, can also be found under the Documentation View. To open them, double-click on their node and the appropriate application for viewing will be automatically opened.

Note about Memory view:

When using projects, the default application memory content is not "blank" but contains the mapping of the object files included in your project. As a consequence, if you click the "Reload memory" button, the memory will not become "blank" but your object files will be reloaded. This is not the same behavior as when programming without a project. Without a project the default application memory is blank.

5. Programmer options

5.1 ST7

RFlasher supports most ST7 devices from ST.

The supported ST7 devices use the ICC protocol that has been designed for in-circuit programming them. ST7 in-circuit programming tools use this protocol.

Option bytes control a variety of configurable features for ST7 Flash devices such as memory write protection, clock source, watchdog and low voltage detection. RFlasher allows you to read, erase and program the option bytes in the programmer specific panel. Please refer to the device datasheet for more information about these bytes and their effect on the device. (watch out for read and write protection bits)

Warning: For XFlash devices, if write protection is enabled the XFlash memory behaves like ROM memory. You will not be able to debug your application and it is impossible to rewrite the XFlash memory.

Some ST7 devices (ST7Litexxx, ST726x, etc.) feature EEPROM-like memory called XFlash. These devices do not need to be erased before programming them.

The other type of memory, HDFlash, is based on real Flash and needs proper erasing before reprogramming.

Please refer to your device's documentation for information about its Flash memory type.

5.1.1 ICC with RLink

5.1.1.1 RLink options section

This section is used to test the connections between the PC and the RLink.

- **View REva Jumpers:** this opens a window containing pictures showing how to set the jumpers on the RLink attached to the REva board for communicating with ST7 devices through ICC.
- **Connect to RLink and read Serial Number:** this connects to RLink and reads its serial number. It is used to check the connection and the installation of the RLink USB driver. Raisonance support team will need your RLink's serial number if you contact us.

5.1.1.2 ICC Mode Entry section

This section is used to specify the method to be used when entering ICC mode.

- **Ignore Option Bytes:** when checked, RFlasher will first attempt to enter ICC mode using the default option bytes. If unchecked, the first reset will be done using the custom option bytes. If the first method fails, RFlasher will automatically attempt to enter ICC mode using the other method.

You can refer to the ST application note "**AN1813 In-Circuit Programming Considerations**" for more information on the ICC mode entry methods.

5.1.1.3 Option Bytes Options section

This section allows handling the target's option bytes and the configuration RFlasher's behavior regarding option bytes during the program operation.

- **Action When Programming:** these options tell RFlasher what it should do with the option bytes when you launch the "Program" command in the "Commands" section of the panel.
 - If "Leave as is" is checked, then the option bytes will not be modified.
 - If "Restore Default" is checked, then the default value will be programmed in the option bytes. To find out what this value is, check "Write" and then click "Default" in the "Value to write" sub-section.
 - If "Write" is checked, then the value that you entered in the "Value to write" sub-section will be written in the option bytes.
- **Value to write:** this allows the specification of the value to write in the option bytes if "Write" is checked in the "Action when programming" sub-section.
 - Use the edit fields to enter a value manually.
 - Use the "Default" button to have RFlasher write the default value in the edit fields.
 - Use the "Change Value" button to open a configuration window and select the option bytes to write.
- **Read Option Bytes:** this allows the reading of the option bytes from the device.
 - "Restore Default Option Bytes now!" / "Write Option Bytes now!": this allows writing the option bytes to the device without changing the Flash.
The text on the button changes depending on the option selected in the "Action when programming" sub-section.

5.2 STR7

The STR7 family is a family of ARM-core based microcontrollers from ST. RFlasher supports most of them.

They provide four JTAG pins that allow you to program them. Most programming tools use them.

Some STR7 devices (STR73x) also provide a UART bootloader that can be used for programming them.

Most STR7 devices provide several boot modes (FLASH, RAM, BOOT, etc.), usually selected by jumpers on the target board. Make sure that the correct mode is selected before attempting to use RFlasher: you should select FLASH mode for JTAG FLASH programming. You should select BOOT mode for using the bootloader. See the device and the target board's documentation for more information.

5.2.1 JTAG with RLink-STR7 or JTAGJet

5.2.1.1 Test connect section

This section is used to test the connections between the PC, the RLink, and the target device.

- View REva Jumpers: this opens a window containing pictures showing how to set the jumpers on the RLink attached to the REVA board for communicating with STR7 devices through JTAG. Appears with RLink, not JTAGJet.
- Connect to RLink: this connects to RLink and reads its serial number. It is used to check the connection and the installation of the RLink USB driver. Raisonance support team will need your RLink's serial number if you contact us. Appears with RLink, not JTAGJet.
- Connect to Target: this connects to the target via JTAG and checks the target's IdCode. Use this to check that connections, power and the other options in this window are correctly set.
- Help: opens the help window.

5.2.1.2 JTAG Clock section

This section is used to specify the speed of the JTAG clock.

Speed of the JTAG Clock: the value entered in the edit field specifies the maximum allowed speed for the JTAG clock.

- The value is interpreted in KHz.
- The minimum value is 400.
- The default is 4000.
- The maximum is 24000.
- The JTAG clock should always be slower than the target board's clock. You must change this if your target board's clock is slower than 4MHz.
- You can use the "Connect to target" button to check if the selected speed is compatible with the target board's clock speed.

5.2.1.3 Sector Selection section

This section allows the application of a mask on the sectors to be affected by the commands.

BxSy: Each checkbox corresponds to one sector. When the box is checked, the sector will be affected by the commands. Otherwise, the sector will be skipped. **x** is the bank number and **y** is the sector number in this bank.

See the device datasheet for the precise definition of the sectors.

5.2.1.4 JTAG Chain Description section

If your STR7 device is chained with other JTAG devices, describe the JTAG chain in this section.

- **Single Device**: check this box if the STR7 is NOT chained with other JTAG devices. Unchecking this is like setting `DRB=0, IRB=0, DRA=0, IRA=0`. (see below)
- **Chain Parameters**: this is the actual description of the chain. There are four parameters needed to be able to communicate with the STR7 device:
 - **DRB & DRA**: This is the number of devices before and after the target STR7.
 - **IRB & IRA**: This is the sum of the lengths of the IR registers of the devices before and after the target.

Example: if the chain contains two devices before the STR7, the first with `IRlength=4` and the second with `IRlength=6`, then you should configure the parameters like this: `DRB=2, IRB=10, DRA=0, IRA=0`.

5.2.2 UART bootloader with COM port

5.2.2.1 Presentation

The bootloader is a piece of software in the ROM of the STR730 devices.

It allows programming the device's Flash memory using the UART.

It is mapped at address zero and executed at reset if the M0 signal is tied to GND and the M1 signal is tied to VCC.

Most boards provide jumpers for doing this.

This part of RIDE and RFlasher provide an interface to it for easily programming your application in the device.

5.2.2.2 Options

The only options are for the communication:

- **COM port**: you must tell the software which port of your PC is connected to the device.
- **Baudrate**: you must also tell it what speed you want the communication to be performed at. Depending on the target's clock speed, you might want to use a lower speed than the maximum available.

Please refer to the device datasheet for more information.

5.2.2.3 Operation

- **Connection**: when these options are correctly set, just follow these steps:

1. Power the target board.
 2. Select the jumpers for entering bootloader (or "**system**") mode.
 3. Connect the serial cable between the selected port of the PC and the UART0 of the target device.
- **Reset**: then, you must reset the target board before each operation (erase, program, blank-check, etc.).
 - **Command-Line**: the STR730_pgm.exe program installed with RIDE allows performing the same operations from the command-line. Launch it without argument in order to see the help that will tell you how to use it.