STM32-Primer

Fun, easy introduction kit for STM32 microcontrollers

User Manual

July 2007

RAISONANCE

Table of Contents

1. Presentation 3
2. The STM32 Primer hardware 4
   2.1. List of Contents 4
   2.2. Description 4
   2.3. STM32 microcontroller features 6
   2.4. 3D accelerometer 6
   2.5. Power supply 7
   2.6. For further information… 7
3. Getting started 9
   3.1. Connect battery, charge and power up 9
   3.2. Play 9
   3.3. Install the Ride7 software toolset for ARM 10
   3.4. Modify a STM32 application example 11
4. Managing your CircleOS applications 12
   4.1. CircleOS architecture 12
   4.2. The CircleOS Scheduler 13
      a) Initialization stage 13
      b) Periodic systick interrupt 13
      c) Application scheduler 14
   4.3. CircleOS Memory map 14
   4.4. Managing app. on your STM32-Primer 15
   4.5. Selecting the current application 15
   4.6. Downloading new applications 17
   4.7. Restoring to factory configuration 17
   4.8. Resetting your STM32-Primer 18
      a) Hardware reset: 18
      b) Software reset 18
5. Developing CircleOS applications 19
   5.1. Libraries 19
   5.2. Developing your first CircleOS application 19
   5.3. Debugging your application 20
   5.4. Sharing your application with the Circle community 20
6. Recycling 21

1. Presentation

The STM32 Primer is an innovative, low-cost evaluation and development package that is designed to provide a fun and easy introduction to the features of the STM32 with ARM Cortex™-M3 core.

The Primer's ergonomic design with MEMS-based controls (navigate by tilting the tool left, right, backward or forward) and LCD display provide fun and easy control of the included demonstration firmware that includes graphical user interface and games based on the resources of the STM32 microcontroller.

The included firmware (CircleOS task scheduler, system services and demonstration applications) implements low level functions driving the various STM32 peripherals. In addition, it includes features for dynamic loading and management of new applications. All firmware, demos (C sources and projects) and more future applications are available for free download at the STM32-Primer dedicated site, http://www.stm32circle.com/.

The Primer and Ride7 software toolset provide everything needed for programming the STM32 and debugging applications, including:

- USB host connection for in-circuit programming and debugging
- Ride7 integrated development environment for code editing, device programming and application debugging (debug up to 32K of code, with included version. For information about upgrade to an unlimited version of Ride, visit http://www.stm32circle.com/resources).
- GNU C /C++ compiler (unlimited compiling)

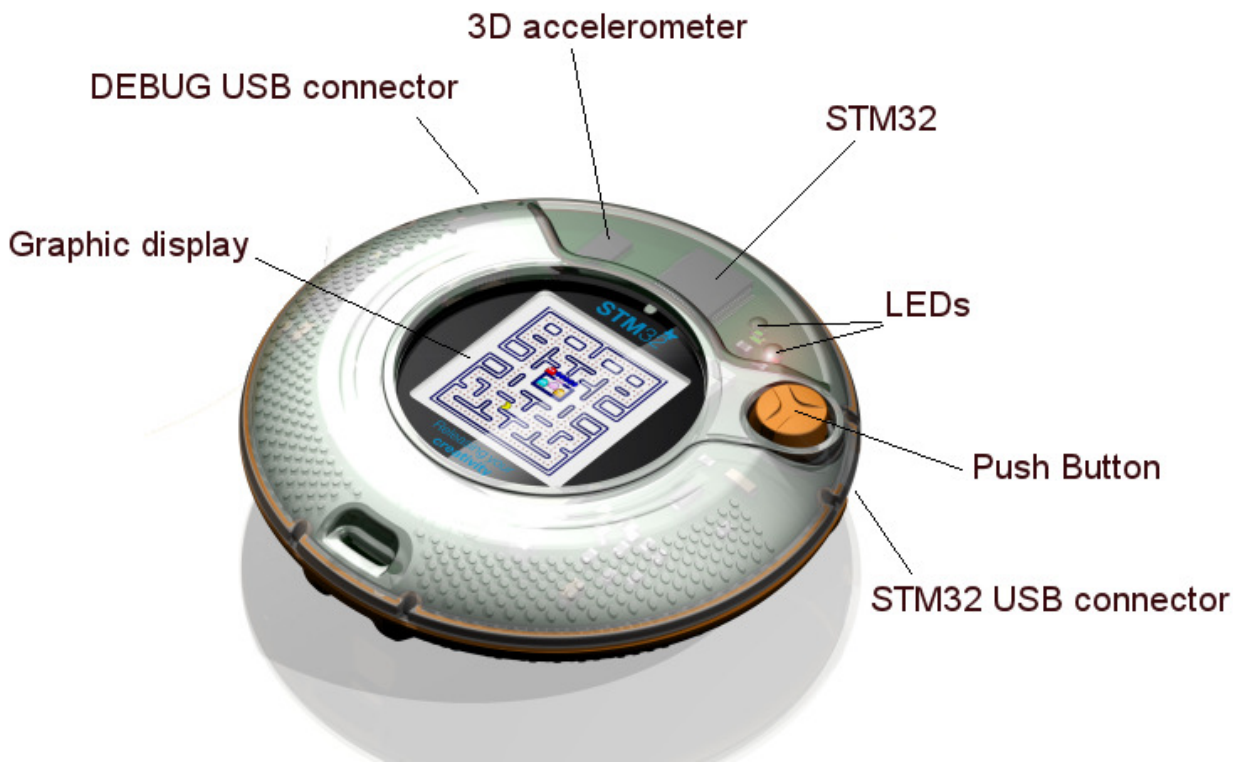
2. The STM32 Primer hardware

2.1. List of Contents

You will find when opening the box:

- The STM32-Primer in its plastic case,
- A USB cable for host PC connection that can be used to program and debug the STM32 microcontroller. Alternatively this cable can be used to communicate with the STM32-Primer if you program a USB client application.
- A CD-ROM containing Ride7 for ARM and all the STM32-Primer documentation.

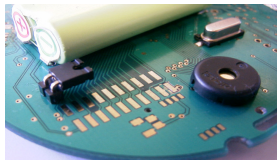
2.2. Description



The STM32-Primer provides the following devices:

- An LCD color monitor (64K colors, 128x128 pixels),
- 2 USB connectors:
  - One marked "Debug" to connect to a PC running Ride for application development,
  - One marked "STM32" that allows the embedded application to communicate with an external USB host.
- One push button to switch on the power supply and to launch menu commands,

- One accelerometer (MEMS) that captures the 3D-position information related to the STM32-Primer, and which is used to navigate through the menus, and to move the pointer.
- Note that 2 footprints on the printed circuit could receive some optional interface components:
- One IrDA transmitter (top side, close to the "Debug USB port") that allows communication between two STM32-Primer,
- One extra connector is linked to some unused I/O pins of the STM32 in order to add extra peripherals.



Opening the plastic case of the STM32-Primer is easy (no screw, no clip). Just separate the two halves of the case and remove the board.

2.3. STM32 microcontroller features

The STM32-Primer is equipped with an STM32F103B6, one of the new ST, Cortex-based, 32-bit microcontrollers. The main characteristics of this device are:

- ARM 32-bit Cortex™-M3 CPU, 72 MHz, 90 DMips with 1.25 DMips/MHz,
- 128KB of Flash program memory, 20KB SRAM,
- Embedded oscillators (for high-speed quartz + RTC),
- SWD and JTAG debug interfaces,
- Fast input/output: up to 80 IOs, ADC, DAC,
- Embedded communication peripherals: USB 2.0, CAN, USART, SPI, I2C, LIN,
- Multiple timers; watchdog, PWM, Systick timer, ...

2.4. 3D accelerometer

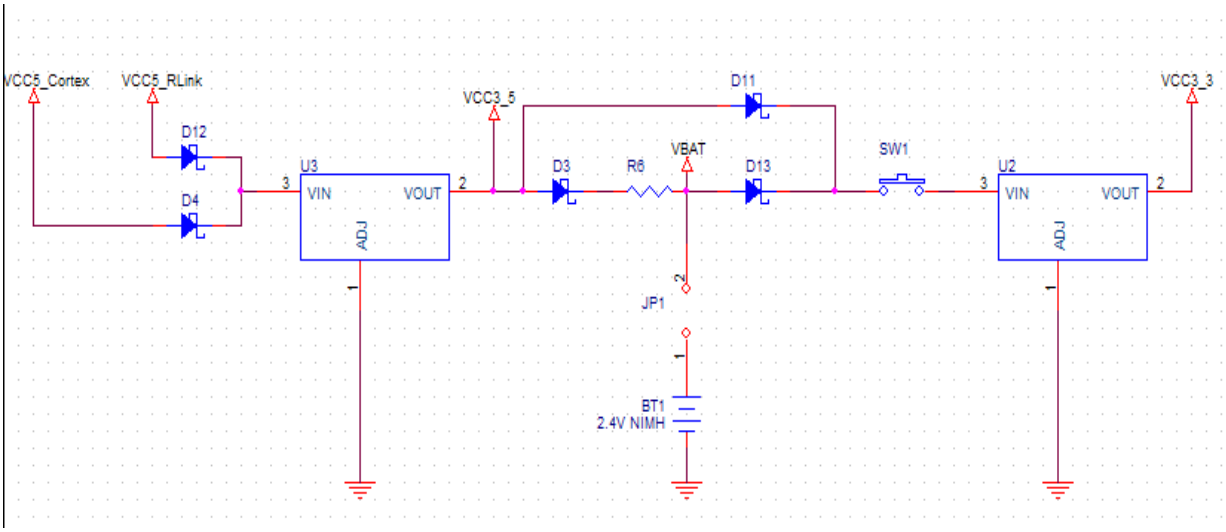
The STM32-Primer is equipped with a MEMS inertial sensor (LIS3LV02DL from STMicroelectronics). This device is used by the STM32-Primer as a human interface device to select commands, in coordination with a graphic pointer. When you start the STM32-Primer for the first time, you will see a small ball moving according to the orientation of the STM32-Primer circuit. The information about the 3D position is provided by the MEMS.

You will find complete documentation of the 3D accelerometer on the companion CD-ROM. Examples are provided that allow evaluation of various functions of the MEMS usage.

2.5. Power supply

The STM32-Primer features a 2.4V rechargeable battery.

The following figure shows the general principle of the power supply schematic (the complete STM32-Primer schematics are available in appendix A):



When one USB connector is linked to a PC host (either the RLink or the STM32), the 5V voltage supplied by the PC is used to recharge the batteries.

When no USB host is connected, the battery is used to supply the power for the STM32-Primer. When the batteries are fully charged, the Primer can be used for about 2 hours.

The duration of the batteries depends on the Primer activity. For instance, it can be extended -or reduced- by changing the setting of the backlight intensity (see the menu “Settings”).

Note: The battery load mechanism of the STM32-Primer is limited. It just applies a voltage to the battery in order to charge it, without any control over the delivered current. This is enough for demonstration and evaluation purposes, but this should not be applied to a project intended for commercialization. Full recharging of the battery takes 16 hours and it is recommended to remove the battery jumper after charging if your Primer will remain connected to your PC for application development.

2.6. For further information…

This document describes the STM32-Primer, basic use and its hardware and firmware features. Further information about tools intended for use with the Primer may be found in:

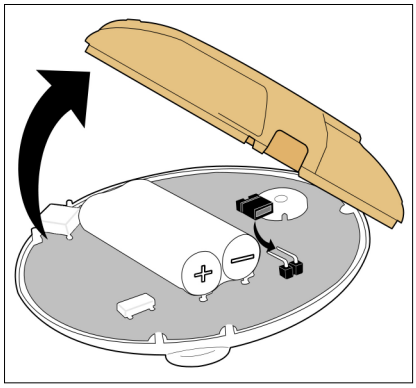
- “LIS3LV02DL MEMS Inertial Sensor Data Sheet”, available from http://www.st.com/.
- “STM32F103 Performance AC Line”, available from http://www.st.com/.
- “STM32F10x Flash programming manual”, available from http://www.st.com/.
- “STM32F10x advanced ARM-based 32-bit MCU reference manual”, available from http://www.st.com/.
- “STM32F10x Datasheet”, available from http://www.st.com/.
- The “Cortex-M3 Technical Reference Manual” document describes the Cortex-M3 Core, and is available directly from http://www.arm.com/.

- "ST7637 Datasheet". This document describes the 65K 132x132 color dot matrix LCD controller/driver, and is available from http://www.sitronix.com.tw/.
- The "PZG15BW-SCLW.pdf" document describes the characteristics for the 128x128 display of the STM32-Primer. This document is available from your Ride7 environment.
- The full GNU Compiler Collection (GCC) manuals are available from the Ride7 environment.

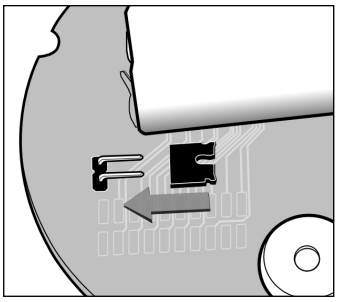
Please visit the Circle community web site at http://www.stm32circle.com/ for more resources.

3. Getting started

3.1. Connect battery, charge and power up



1. Open the Primer's case by pulling the top and bottom halves of the case apart. There are **no** catches or releases that hold the case together.
2. Fit the jumper so that it straddles the two pins to connect the power supply.



3. Close the case.
4. Connect the Primer to the USB port on a PC to charge its battery.
5. After charging the battery, press the push button to activate the Primer.

3.2. Play

1. After the opening screen is displayed, press the push button to call the main menu.
2. Tilt the Primer backward/forward to navigate in the menu, use the push button to select a menu command. Initial menu items include:

Config	-	Configure parameters for the Primer including the pointer, backlight for the LCD display and time
Maze	-	Start the maze game. Note that this menu item depends on the application that is selected in "Applic"
Applic	-	Select an application to run. The application that you choose will then appear in the main menu.
Shutdown	-	Turn off the Primer. To restart the Primer, simply press the push button.

Note: When you receive the Primer, the MEMS based controls are calibrated to a "zero" position that matches a 30° angle from the horizontal (corresponds to the position of a book when reading). To practice controlling the Primer, move the blue dot around the main screen. It takes a little practice.

Preinstalled applications
The Primer includes pre-installed applications:

- **Maze** – A game where the player navigates inside a labyrinth, eating dots to win points while avoiding ghosts
- **Breakout** – A game where the player uses a paddle to bounce a ball against a wall of blocks, destroying the block to win points. Win the game by destroying all the blocks.
- **Configuration** - Configuration and test applications have been pre-installed in order to check your STM32-Primer's capabilities. From the main menu of the STM32-Primer, launch the "Config" command. The following parameters can be set:

- **CPU Freq**. This lets you choose the speed of your STM32-Primer.
- **Backlight**. This lets you tune the backlight intensity. Note that the backlight is the main source of power consumption (with the LCD monitor itself). Reducing the backlight intensity allow to extend the duration when Primer is powered by the battery.
- **Time**. This sets your STM32-Primer time. Note that the RTC clock will remain valid even if your STM32-Primer is shut off.
- **Test**. This performs a quick factory test of the STM32-Primer.

You can add more applications to your STM32-Primer by downloading them on the web site www.stm32circle.com/projects. Refer to the chapter "4.4 Managing applications on your STM32-Primer".

3.3. Install the Ride7 software toolset for ARM

The CD-ROM contains many resources to help you evaluate the STM32 family:

- The RLink driver that needs to be installed in order to program/erase the applications.
- Ride7, the Raisonance IDE to write and debug new applications.
- The complete GNU software toolchain based on the GCC compiler, which is fully integrated into Ride7.
- Some utilities to manage your Circle applications (Circle is the OS embedded on your STM32-Primer; refer to Chapter 4.1 "CircleOS architecture" for details).
- Documentation: Datasheets, user manuals for the different components of the STM32-Primer.

To install:

1. Insert the Raisonance mini CD-ROM on your PC, a menu will appear.
2. Select "Install ARM Toolset" from the menu. It will install all the required resources on your PC, including the RLink driver required for connection on the "Debug" USB port of your Primer.

Note: Install Ride7 before connecting to this USB port.

3.4. Modify an STM32 application example

In this chapter, we will modify an example installed with the “Ride7 Kit ARM”. This example is loaded in Ride7 by default the first time.

The full path is:
“[RIDE7_INSTALL_DIR]\Examples\ARM\Primer\STM32\toggle_with_CircleOS\toggle_with_CircleOS.rprj”

It uses the CircleOS operating system provided as a library (circle.elf) that is described in the next chapter.

Follow these instructions to modify the example:

1. Launch Ride7:

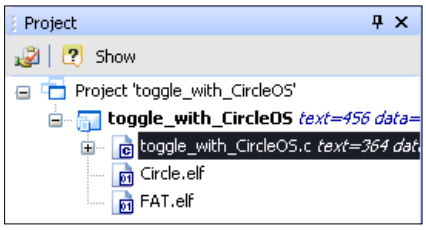
   Start > Programs > Raisonance Tools > Ride7 > Ride7

   Plug the Primer (Debug USB port) into the PC using the USB cable and follow the Windows USB installation.

2. Launch the Debug mode: 
3. Run the program: 

   In the main menu on the Circle choose "toggle". As you can see, the application just toggles the green led to red. Now we will modify the program to make the buzzer beep at each led toggle…

4. Exit the debug mode: 
5. Open the toggle example C file



Just double-click on the "toggle_With_CircleOS.c" in the project window.

6. Add in the code:

```
if(toggle_count >= MAX_TOGGLE_COUNT)
    {
    toggle_count=0;

    CurrentLedMode = (CurrentLedMode == LED_ON) ? LED_OFF : LED_ON;

    LED_Set ( LED_GREEN,CurrentLedMode);
    LED_Set ( LED_RED,!CurrentLedMode);
    BUZZER_SetMode(BUZZER_SHORTBEEP);
    }
```

7. Launch the Debug mode: 
8. Run the program: 

4. Managing your CircleOS applications

4.1. CircleOS architecture

The STM32 Primer is equipped with an STM32F103 that contains 128KB of FLASH and 20KB of RAM.

The STM32-Primer embeds the CircleOS operating system (source files are available on www.stm32circle.com web site). It provides services that will help you develop your STM32-Primer applications, including:

- Application management,
- LCD graphic functions,
- Mems functions,
- LED, Buzzer and Push Button functions,
- Menu functions,
- Scheduler task,
- ...

CircleOS can load several independent applications.

Each application is run by CircleOS when selected, has the full availability of the CPU and can use all the RAM that is not being used by CircleOS (i.e. 16KB in the memory address range from 2000000h to 20003FFFh). It will be scheduled by the CircleOS with full privileges on the device, until it explicitly quits.

4.2. The CircleOS Scheduler

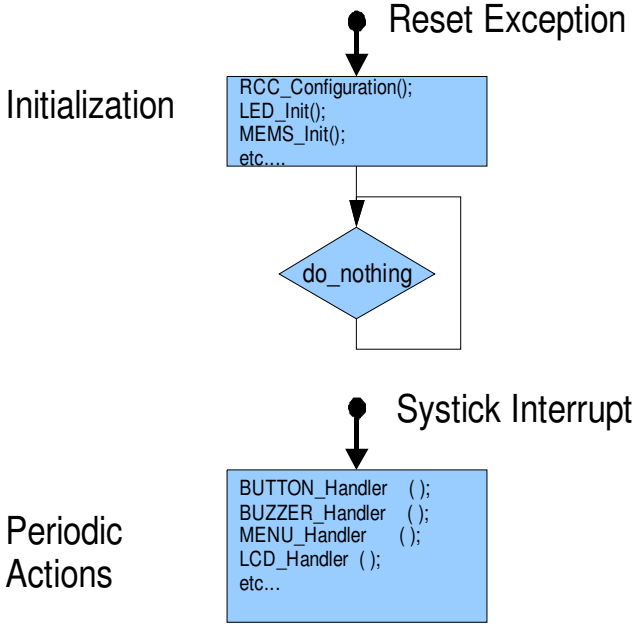
CircleOS acts in several stages: An initialization stage which occurs upon device reset, a periodic systick interrupt, and the scheduling of applications.

a) Initialization stage

During the initialization stage, the hardware configuration is performed, and the periodic system timer (systick) is installed.

The systick period depends on the RCC settings: it can be modified through the menu "Config | CPU Speed" to the following predefined values:

Level	CPU freq	Systick Freq
1	18MHz	0.75 KHz
2	24MHz	1 KHz
3	36MHz	1,5KHz
4	48MHz	2KHz
5	72MHz	3KHz



The (CPU_freq / SysTick = 24000) ratio is applied for all these values.

b) Periodic systick interrupt

The periodic systick makes a call to the CircleOS systick interrupt handler. This interrupt handler will perform a short processing on each of the STM32-Primer components: LEDs, button, buzzer, MEMS, LCD etc…

c) Application scheduler

CircleOS is the base application of your STM32-Primer. It will handle the menu selections and react to user actions.

Once an application is run (usually through a menu selection), CircleOS will first call an initialization routine for the application, then will repeatedly call the application handler at the Systick frequency until it returns a MENU_LEAVE value.

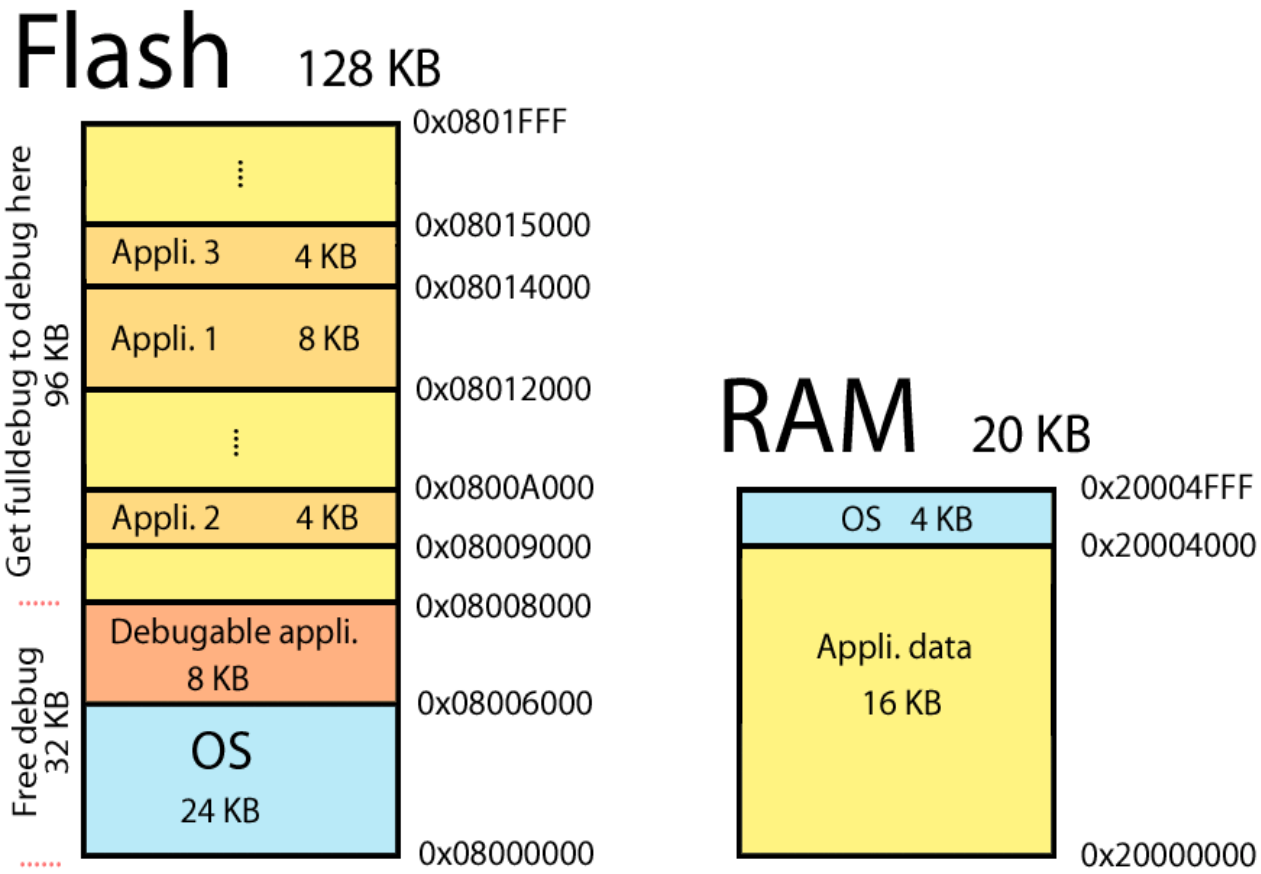
You will find more details about programming CircleOS applications in Chapter 5 “Developing CircleOS applications”.

4.3. CircleOS Memory map

The CircleOS firmware requires 24KB of FLASH and 4KB of RAM (including the stack usage for the applications). The remaining 104KB are available for applications, which can be added or removed at will using a programming tool (see below).

Flash memory can be programmed in 1KB blocks only.

The following figure shows an example of memory mapping:



4.4. Managing applications on your STM32-Primer

The applications can be managed by the "**circle_mgr.exe**" utility. You will find it in the "[RIDE7_INSTALL_DIR]\Bin" directory.

This utility dedicated to the STM32-Primer allows:

- Listing of the currently loaded CircleOS applications.
- Adding of new CircleOS applications.
- Removing of CircleOS applications.
- Checking how much FLASH memory is available.

Refer to the http://www.stm32circle.com/ web site for more information about this utility.

The following commands are available with the "**circle_mgr.exe**" utility:

Command	Syntax	Description
List	L	List the loaded applications. The following information will be output: **circle_mgr.exe L** Reading FAT table... App0: Name=Maze, Addr=0x08006000, Size=8KB App1: Name=Breakout, Addr=0x08008000, Size=4KB Largest free block= 92KB
Add	Afilename	Add a new application (object file). **circle_mgr.exe Ac:\tmp\level.o** Linking file C:\tmp\level.o... Link of C:\tmp\level.o succeeded... Hex file generated... Blank-checking the FLASH area...OK Programming file _tmp_.ld.hex to flash...OK Registering application in FAT... OK
Erase	E* Eappname	**circle_mgr.exe EMaze** /*remove only 'Maze'*/ circle_mgr.exe E* /*remove ALL apps */
Wait	W	When a command list is launched through a batch file, the W command allows you to pause the execution and to check the intermediate results.

Caution: Any hex file can be programmed to the FLASH memory of your STM32-Primer using the "**cortex_pgm.exe**" utility. However, doing this will destroy your CircleOS firmware, and you will have to reinstall it if you wish to use it later (refer to chapter 4.7 "Restoring the factory configuration" for details).

4.5. Selecting the current application

One application is considered to be the “current application”. The ID of the current application is saved in the backup memory. From the main menu, you can launch it directly.

To change the current application, select the “Application” command from the main menu. Then select the application you wish to specify it as the “current application” and push the button. The new “current application” name will now appear in the main menu.

4.6. Downloading new applications

You will find on the http://www.stm32circle.com/projects Circle web site a database where the members can share their applications with the stm32circle community.

An application can include both the source files and the object files, or the object files.

An application is generally made of one object file, but may occasionally have several of them. The linking of the application in such a case can be done either using the **circle_mgr.exe** software (available in the "[RIDE7_INSTALL_DIR]\bin" directory) or within the Ride7 environment. When an application is split into several object files, these object files must be placed in a library in order to pass a unique filename as an argument to **circle_mgr.exe**.

4.7. Restoring the factory configuration

If you have been experimenting with CircleOS applications and have modified your STM32-Primer configuration, you may want to restore the initial (factory) configuration.

In order to do this, please follow these steps:

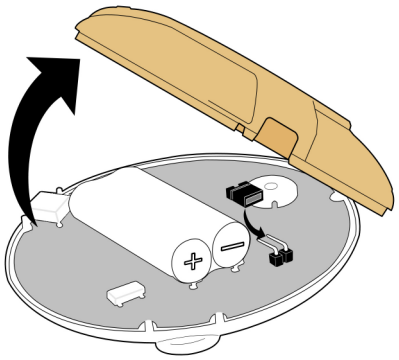
1. Connect your STM32-Primer's debug USB port to your PC.
2. Power-up your STM32-Primer by pressing its button.
3. Open a command prompt from Windows (Navigate to "Start | Programs | Accessories | Command prompt")
4. Change the current directory to the Ride7 installation directory. This can be done with the following command (adapt it to your actual configuration if you did not install Ride7 in its default location):
   ```
   cd "C:\program files\Raisonance\Ride"
   ```
5. Now change the current directory to the STM32-specific library directory. This can be done with the following command:
   ```
   cd lib\ARM\CircleOS
   ```
6. Erase your STM32-Primer, reprogram it with its factory ROM image, which is in the "`circle.hex`" file. Then restart the device. These operations can be done with the following single command:
   ```
   cortex_pgm TSTM32F103RBT6 E Pcircle.hex S
   ```
   - The TSTM32F103RBT6 option selects the appropriate device.
   - The E command erases the ROM.
   - The Pcircle.hex command programs the device with the proper hex file.
   - The S command restarts the device.

   These operations take approximately 30 seconds.

4.8. Resetting your STM32-Primer

In case you need to reset your STM32-Primer, several mechanisms can be used:

a) Hardware reset:



1. Remove any USB cable.
2. Open your STM32-Primer's case to temporarily remove the battery jumper which is on the back side.
3. Replace your battery jumper, close the case of your STM32-Primer and restart it by pressing its button.

b) Software reset

A software reset will erase all the flash memory in your STM32-Primer and all applications and the CircleOS will be suppressed.

1. Connect your STM32-Primer's debug USB port to your PC.
2. Power-up your STM32-Primer by pressing its button.
3. Open a command prompt from Windows (Navigate to "Start | Programs | Accessories | Command prompt")
4. Reset your STM32-Primer using the **cortex_pgm.exe** utility installed in "[RIDE7_INSTALL_DIR]\bin" directory. This must be done with the following command: cortex_pgm S

To reload the initial program, use the following command:

```
Cortex_pgm.exe TSTM32F103RET6 E Pstm32primer1.1.hex S
```

where "`stm32primer1.1.hex`" is the full application, which is available on the web site.

5. Developing CircleOS applications

The full source files of CircleOS are available on the http://www.stm32circle.com/ web site.
Once registered, you will be able to download them, along with many resources for developing your application.
The games originally delivered with the STM32-Primer show working application examples.

5.1. Libraries

Some common services are offered to ease your development of CircleOS applications.

- The STM32 libraries, written by ST, provide access to the embedded peripherals (such as timers, ADC, communication interfaces, thermometer, etc...) of the STM32 microcontroller.
- The low-level CircleOS functions that provide an easy access to the STM32-Primer's on-board peripherals: 3D accelerometer, LCD monitor, button, buzzer, battery, LEDs.
- The graphical functions that provide powerful high-level functionality: Menu management, pointers (linked to the 3D accelerometer), character maps, sound.

The source files of these libraries can be found:

- On the CD or on the ST web site for the STM32 libraries. Specific documentation about the STM32 library is also available.
- On the http://www.stm32circle.com/ web site for the CircleOS libraries (registration required).

Their documentation is accessible from Ride7.

5.2. Developing your first CircleOS application

Creation of a CircleOS application is done automatically in Ride7:

9. Navigate to "Project | New project".
10. Leave the "Type" selection list to "New application".
11. In the "Processor" selection list, select the "STM32F103RBT6_CircleOS" device.
12. Select an application name such as "My CircleOS application".
13. Define the location where your new project will be created.
14. Click the "Finish" button

Your new project will be created, with an application containing a CircleOS application skeleton as well as the Circle.elf and FAT.elf files necessary to connect your application to CircleOS.

1. Open the "Application.c" file.
2. Search for the Application_Name variable in the file.
3. Change the Application_Name value from "My App" to "HELLO".
4. In the Application_Handler function, create a new string as follows:
```
const char msg[] = "Hello, World!";
```
5. Use the DRAW_DisplayString CircleOS service to display the "msg" string variable you just created on the STM32-Primer display:
```
DRAW_DisplayString( 5, 20, msg, sizeof(msg)); // X, Y, string, length
```
6. Connect your STM32-Primer to your PC using the USB cable (be sure to use the "debug" USB port of your STM32-Primer).

7. From Ride, go to “Debug | Start”, which will program your application to your STM32-Primer. This may take about 15 seconds.
8. Go to “Debug | Run”.
9. On your STM32-Primer, select your application name on the main menu.

Your application is now on your STM32-Primer.

For further information about CircleOS application programming and available OS services, please visit http://www.stm32circle.com/.

5.3. Debugging your application

In order to debug your application, you must go to “Project | Properties” in Ride7. In the “Configuration selection box, Circle_Debug and Circle_Release are available. Make sure you select the Circle_Debug configuration (which is the default). Ride7 will take care of all the settings required for switching between debug and release mode through the use of these configurations.

The standard STM32-Primer is limited to debug in the first 32KB only. A software key can be purchased on http://www.stm32circle.com/ to allow debugging in the whole 128KB of memory.

5.4. Sharing your application with the Circle community

Once your application works properly, you can share it with the other members through the http://www.stm32circle.com/ community.

6. Recycling

As part of our continuing efforts to proved the best service to our users an communities, Raisonance is actively defining take-back and recycling programs for EU customers as part of the implementation of the European WEEE directive.

This symbol is a reminder not to dispose of your electronic equipment in standard trash receptacles.

For more information about disposal and recycling of electronic goods, please refer to http://www.raisonance.com/support/weee.php