Bachelor PO - SmartUniversity

Application Example



Hochschule für Angewandte Wissenschaften Hamburg

Important Links

- Code: https://github.com/RIOT-OS/RIOT
- Wiki: https://github.com/RIOT-OS/RIOT/wiki
- Mailing List: <u>devel@riot-os.org</u>
- IRC: <u>irc.freenode.org</u> #riot-os



First Steps

- Get a GitHub account: https://github.com
- Fork the RIOT repository: https://help.github.com/articles/fork-a-repo/
- RIOT & Git Cheatsheet
 https://github.com/RIOT-OS/RIOT/wiki/Git-cheatsheet



Development Environment

- 1. Use a PC and in the Lab with a prepared VM
- 2. Or setup your own Linux develop environment (recommended!)
 - Tools: git, gcc toolchain for arm, OpenOCD
 - Wiki entries that can help you:
 - https://github.com/RIOT-OS/RIOT/wiki/Gettingstarted-with-STM32F
 %5B0%7C3%7C4%5Ddiscovery-boards
 - https://github.com/RIOT-OS/RIOT/wiki/OpenOCD



Lab VMs

- Start the VM located on the USB drive
 - 1. Open Oracle VM VirtualBox
 - 2. Click on *Machine/Add* and select the image
 - 3. E:\PO_SMD \LUbuntu_RIOT_PO_2015\LUbuntu_RIOT_P O_2015.vbox
 - 4. Start the VM
- User: RIOT-OS-Devel, PW: riot

Get the Code

- Open a terminal
- Get the code and switch folders

```
$ git clone https://github.com/USER_NAME/RIOT.git
$ cd RIOT
```

Create a branch and check if it worked

```
$ git checkout -b YOUR_BRANCH_NAME
$ git branch
```



Your First RIOT Program

Go to RIOT/examples/ and create a new folder

```
$ cd examples
$ mkdir my_first_project
$ cd my_first_project
```

Copy Makefile and main.c from the hello-world example

```
$ cp ../hello_world/Makefile .
$ cp ../hello_world/main.c .
```



Adjust main.c

New includes

```
#include "board.h"
#include "xtimer.h"
```

A code snippet for the LEDs

```
for(int i = 0; i < 10; i++) {
    puts("LED on");
    LED_RED_ON;
    xtimer_sleep(1);
    puts("LED off");
    LED_RED_OFF;
    xtimer_sleep(1);
}</pre>
```



Adjust the Makefile

Rename the application

```
APPLICATION = my_first_project
```

Include the xtimer module

```
USEMODULE += xtimer
```



Prepare the Compiler

 Add the bin folder of the arm compiler to your PATH

```
$ export PATH=$PATH:/*path*/*to*/gcc-arm-none-
eabi-*version*/bin
```

Check if it worked (prints compiler version)

```
$ arm-none-eabi-gcc -v
```



Compile Your Code

- Build application for your target device
 - Atmel SAM R21 Explained Pro: samr21-xpro
 - Physic phyWAVE KW22: pba-d-01-kw2x
 - \$ BOARD=*board_name* make
- Flash your binary
 - \$ BOARD=*board_name* make flash
- Open a serial connection
 - \$ sudo BOARD=*board_name* make term



Flashing from the VM

- Windows driver should be installed automatically
- Connect the USB device to the VM
 - Click Geräte/USB/*device* EDBG CMSIS-DAP
 - Or a similar name ...
- Now, flashing should work from your VM



Exceptions for Atmel

- Use a Windows terminal emulation for the serial output
 - Disconnect device from the VM
 - Find the connected COM port (Systemsteuerung/Geräte-Manager)
 - Start Tera Term (on the USB drive E:\PO_SMD)
- Tera Term is preconfigured
- Adapt the COM port (Setup/Serial port/Port)



Application Output

- Press the reset button on the board
- You should see
 - A << Hello World >> message
 - A blinking LED
- Congratulations! You just build your first RIOT application:-)



Track Changes with GIT

Look at your changed files

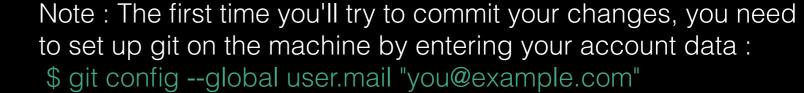
```
$ git status
```

Add and commit files

```
$ git add main.c Makefile
$ git commit
```

Push changes to GitHub

```
$ git push
```



\$ git config --global user.name "Your Name"





www.riot-os.org