Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

System Construction Course 2015,

Assignment 1

Felix Friedrich, ETH Zürich, 15.9.2015

Introduction

At the beginning of systems programming stands the communication with hardware. Very often no diagnostic device such as debugger connected via JTAG is accessible. Therefore the first that we need is a sign of life of the hardware we are working with. The simplest you can do is let some lights blink in order to see that your code is actually running. In this exercise we will execute just this: with a very simple program written in assembly language we will let an LED light up. In the second part of the exercise we will learn more about linking and will experience the advantages of high level (systems) programming.

Lessons to Learn

- Learn to know ARM assembly language basics, understand the limits
- Understand GPIO control on BCM 2835/6
- Understand basics of linking and bootloading

1 Sign of Life

In this part of the exercise we illuminate an LED using low level assembler code.

Preparation

We have already prepared the boot media for the Raspberry Pi with an openelec operating system. We use this only for the bootloading process and replace the kernel by our own. The image stems from `http://openelec.tv/get-openelec`.

All other resources can be found in our subversion repository located at
`https://svn.inf.ethz.ch/svn/lecturers/vorlesungen/trunk/syscon/2015/shared`
referred to as `(repo)` in the following. Use your ETH account credentials.

Development environment (A2)

Windows users:

1. Download the development environment from the repository: you find a `zip`-compressed archive in folder `(repo)/a2/windows`. Extract the archive.
2. Copy the `Work` folder located in `(repo)/a2/` to the WinAos directory. Execute `Aos.exe` located in the WinAos directory.

Linux users:

1. Download the development environment from the repository: you find an installer in folder `(repo)/a2/linux`. Install the development environment following the instructions in the readme file.
2. Copy the `Work` folder located in `(repo)/a2/` to some arbitrary place in your file system. Change diectory there and execute `aos`.

Tasks

1. Open file `Assignment1/RPI.MinimalLED.Mos` in the development environment.
2. Connect an LED to ground and pin number 21 on the RPI2 expansion slot.
3. Using ARM assembler language, program the GPIO pin 21 as output pin and set the pin accordingly.
4. Compile and link the kernel. Copy `kernel.img` to the SD card and insert the SD card into the (powered off) RPI.
5. Power on the RPI and check if the LED lights up upon start.

2 Blinkenlights

Introduction

In this part of the exercise, we will use some high level language code in order to let the LED blink.

Tasks

1. Open file `Assignment1/RPI.BlinkLED.Mos` in the development environment.
2. Read the file. On a first sight, the code should light up the LED. But it does not work (try, if you like). Why? Compile and link it. Look at the linked image and understand how things are organized. Understand what is missing in order to run the program and patch the program. Check that the LED lights up when booting.
3. Now, using high-level language features, implement a blinking LED: Compile and link the kernel. Copy `kernel.img` to the SD card and insert the SD card into the (powered off) RPI.
4. Power on the RPI and check if the LED blinks up upon start. Look at the frequency of the LED and try to estimate the rate of executed instructions per second. What is your observation? Try to explain what you find.

   (Of course, if you like, you can now also add other LED colors and play with the hardware.)

Have fun!

Documents

- BCM2835 ARM Peripherals Technical Manual:
  `BCM2835-ARM-Peripherals.pdf` in folder `(repo)/documents/rpi`
- Lecture Slides System Construction Lecture 1 from the course-homepage
  `http://lec.inf.ethz.ch/syscon`
- A2 Quickstart Guide
  `A2QuickStartGuide.pdf` in folder `(repo)/documents/oberon`

More documents that are not strictly required for this exercise can be found in the repository .