Faculty of Science, Engineering and Technology

Computer Systems - Week 7

Overview

In this laboratory you will begin programming the Raspberry Pi in ARM Assembly .

Purpose: Learn the basics for bare-metal ARM assembly programming with your Raspberry Pi, and get familiar with the GPIO registers

Time: This lab is due by the start of your week 8 lab.

Assessment: This lab is worth 1% of your assessment for this unit.

Bring to Lab

- your Raspberry Pi (either model 2B, 3B, 4B)
- an SD card and SD card reader
- 4 lead wires, two LEDs, two resistors and and breadboard (you could also do this lab with one LED if needed).

Resources:

- Lectures 7.x videos and slides - week 7 (on Canvas)
- FASMARM (the compiler!):
  - https://arm.flatassembler.net/
  - For Linux, see instructions at: https://github.com/FelipMarti/COS10004-RPi
- RPiGPIO.xls (download task resources from Canvas)
- Felip's COS10004 GitHub for RPi resources (SD card files and information:
  - https://github.com/FelipMarti/COS10004-RPi
- SD Card formatting:
  - https://www.raspberrypi-spy.co.uk/2015/03/how-to-format-pi-sd-cards-using-sd-formatter/
  - For Linux, see instructions at: https://github.com/FelipMarti/COS10004-RPi
- COS10004 Video tutorials:
  - Wiring up a simple LED circuit
  - The GPIO registers and header pins

Submission Details

You must submit to Canvas a PDF document containing all required work as described below.

SWINBURNE UNIVERSITY OF TECHNOLOGY

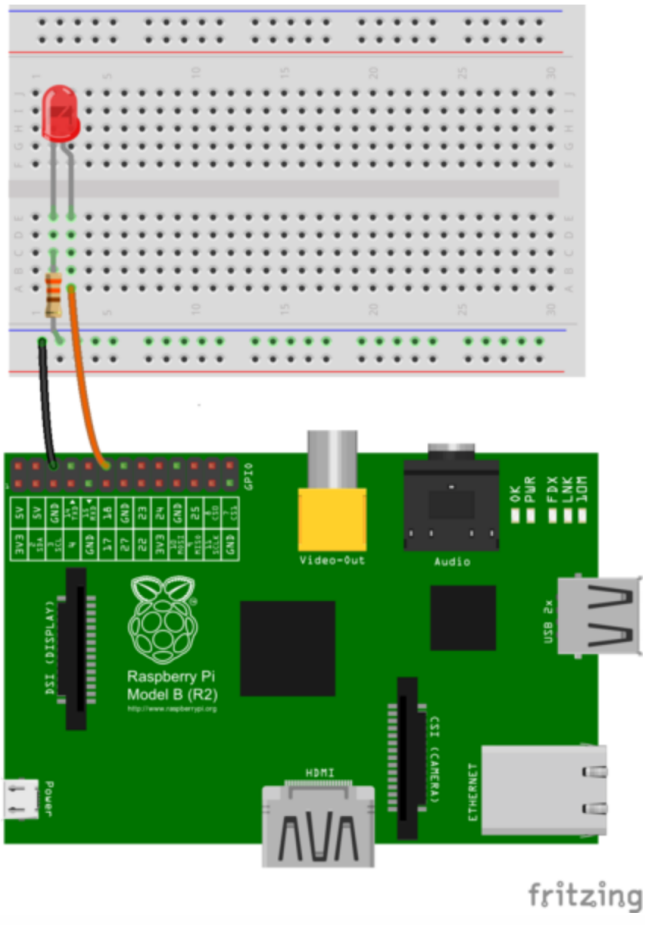
Instructions

Setting up

1. The Raspberry Pi reads all program code and data from a microSD card. The SD card needs to formatted correctly, as a single partition FAT32 filesystem. If you are unsure how to do this, see the resources above where you can find instructions. If things don't work, incorrect formatting is often the reason why!
2. Before you can start executing your programs on your Pi, you will need to ensure your micro-sd card has the necessary files copied to it so it is bootable for the RPi. To do this, simply copy the Resources.zip file from the Lab 7 page on Canvas and extract the contents of the folder from the zip file. Within this folder you will see sub-folders for either 2B-3B, 3B+, or 4B. Choose the one that matches your Pi model, and copy to your SD card
   - **DO NOT create a folder on your card, just copy the files to the root directory.**
3. You now have a bootable microSD card.
4. The files you have copied contain an executable file that we will test, however first we need to wire up a simple circuit to see the output.
5. Wire up a circuit exactly as shown below (your breadboard maybe slightly different, and your Raspberry Pi will have more pins, but the same principles apply). Note the exact header pins to which the lead wires are connected. It is crucial that you have these correct. Ask your lab demonstrator for help if you need.

The lead wires are connected to the third and sixth pins of the top row of pins. If unsure, watch the video linked in the task resources above

SD card slot is here →



Note the Pi pictured here will not look exactly like yours

6. Once you have connected your circuit, insert your prepared microSD card into your Pi, and power it up. There is a slot for the SD card on the side opposite to the USB and ethernet ports.
7. After a few seconds (it could be as long as 10 seconds depending on the Pi model) you should see your LED turn on. If this is not the case, double check all the above steps, and if problems persist, ask your demonstrator for help. If it is working - great ! We know everything is working.

Lets get coding

8. We are now going to explore the GPIO registers of the Raspberry Pi, and how we can program specific GPIO pins to send ON and OFF signals using ARM assembly. Review the relevant slides in this weeks lecture that show how this is done for the wired LED configuration we tested above.
9. We are going to copy this code into the FASMARM editor. To do this, launch the FASMARM application (linked in resources above and in lecture slides)
10. Once FASMARM is launched, type out the code from the lecture slide, paying careful attention to the details.
    10.1. Remember you will need to use the right BASE address for your RPi:
    - **RPI 4B:**

      ```
      BASE = $FE000000
      ```

    - **RPi 3B/3B+/2B(v1.2)**

      ```
      BASE = $3F000000
      ```

    - **Earlier models**

      ```
      BASE = $0x20000000
      ```

11. Save the file as “lab07_GPIO18.ASM”
12. Once you are comfortable, select “compile” from the Run menu. If everything is OK, then no messages will be given. If syntax errors are reported, take a careful look at what you have typed, and make sure it matches. Ask your lab demonstrator or friends if you need help.
13. This will create a file “lab07_GPIO18.img” in the same folder as the source file (if it is lab07_GPIO18.bin instead - no problem! - keep going).
14. Rename this file “kernel.img”, and copy it to the root directory of your microSD card (and yes! Replace the existing kernel.img file on the card). The kernel.img is an expected binary file of machine instructions read in almost immediately upon booting up.
15. Repeat step 6 and hopefully you will see your LED turn on (it won’t flash this time though -- you are not a Jedi yet!).

It is important you understand what is going on in this code so lets take a closer look.

16. **In your submission document**, provide the lines of code from your lab07_GPIO18.ASM file that achieve the following (all of these will be more than 1 line):
    16.1. Establish the base address of the GPIO registers ?
    16.2. Program GPIO18 for writing ?
    16.3. Set GPIO18 to ON ?
    16.4. Stop the instruction pointer (program counter) from continuing beyond the executable program code ?

Provide your answers to each of these questions in your submission document

There is only one thing better than turning an LED ON … turning two LEDs on! I know this because I've done it .. and you can too!

17. Wire up a second LED, this time using the GPIO pin 23 to control the LED. Refer to the diagram on the right to find where GPIO pin 23 is on the header, and where another available ground (GND) pin is.

    NOTE: If you only have the one LED to play with, then just move the lead wires to these new pins.

18. You will need to modify your lab07_GPIO18.ASM file to turn on the second LED. Make a copy and name it lab07_GPIO18_23.ASM.
19. You are going to add code snippets to do the following:
    19.1. program GPIO23 for writing
    19.2. set GPIO23 to ON

The code for this will follow the exact same pattern as for GPIO18, however, the numbers (bit shifts, and byte offsets) will be different. To find out what these are you will first need to refer to the GPIO register diagram linked off Canvas (and in the task resources on Canvas).

We covered this lookup table in Lecture 7.5, however you might want a refresher. If so, watch the short video tutorial linked at the top of the lab sheet.

If you are still stuck, ask your lab demonstrator - this can be quite tricky at first, so give it time to understand.

20. If you are comfortable in your understanding of the GPIO registers then you should be able to answer the following questions:

20.1. What number bit is set (within the associated 32 bit block) to enable GPIO23 for writing ?

20.2. What is the byte offset from GPIO_BASE that this 32 bit block must be written to in memory ?

20.3. What number bit is set to set GPIO23 to ON (again within the 32 bit block associated with that GPIO pin)?

20.4. What is the byte offset from GPIO_BASE that this 32 bit block must be written to memory ?

Provide your answers to each of these questions in your submission document

21. Now go ahead and add the assembly code needed to turn on the second LED. When you have it working, demonstrate it to your lab demonstrator (or provide a video link)

Include the full source code of your solution (with comments) into your submission document

22. Consider how you would program one of the GPIO pins used above to turn off the LED it controls. Choose one of the LEDs to turn off, and answer the following:

22.1. Which exact snippet of code will need to change compared to turning the LED on ?

22.2. Provide the alternative code to turn the LED off (again you will need to refer to the GPIO register diagram). No need to demonstrate this working. We'll deal with flashing LEDs next week.

Include this answer in your final submission document.

When complete:

- Submit your answers (screen shots, etc) in a single document using **Canvas**
- Show your lab demonstrator your working circuits in class (you must do this to get the 1%). Your lab demonstrator may request you to resubmit if issues exist.