

## **Robotics: Science and Systems I**

### **Lab 1: System Overview and Introduction to the $\mu$ ORCboard**

**Distributed: Wednesday 2 February 2010, 4:00pm**

**Checkoffs due: Wednesday 2 February 2010, 5:00pm**

## **Objectives**

Your objectives in this lab are to:

- Gain perspective on the computational architecture of your robot
- Gain experience in reading a circuit board schematic
- Familiarize yourself with multimeters and oscilloscopes
- Learn about pulse width modulation
- Learn to use the class wiki

By the end of this lab, you will have a powered, functioning  $\mu$ ORCboard and will be ready to tackle Lab 2, Chassis Build, Motor Characterization, and Motor Control.

## **1 Lab Materials and Equipment Check**

Pause for a minute to run through the following lists and verify that you have each item nearby (if not, tell an RSS staff member).

Paper handouts (this handout is 1-A; all handouts are also posted on the class web site):

- (1-B) MASLab<sup>TM</sup>  $\mu$ ORCboard Layout (one per table)
- (1-D) Lab 1 Checkoff (one per student)

Lab materials:

- A  $\mu$ ORCboard, battery, battery cable, and optionally, an AC power adapter;
- A multimeter;
- An oscilloscope and probes; and
- A Sun workstation, a netbook, and an Ethernet cable.

## **2 Overview: Robot and Development Environment**

Over the coming term, in addition to designing and building robot hardware, you will develop software to process input from your robot's sensors (e.g. camera and sonar) and generate output to command its actuators (e.g. motors). The block diagram (Figure 1) represents a high-level overview of how you will develop your software, how and where that software will execute and how that software connects to the robot hardware. We'll return to this diagram in future labs.

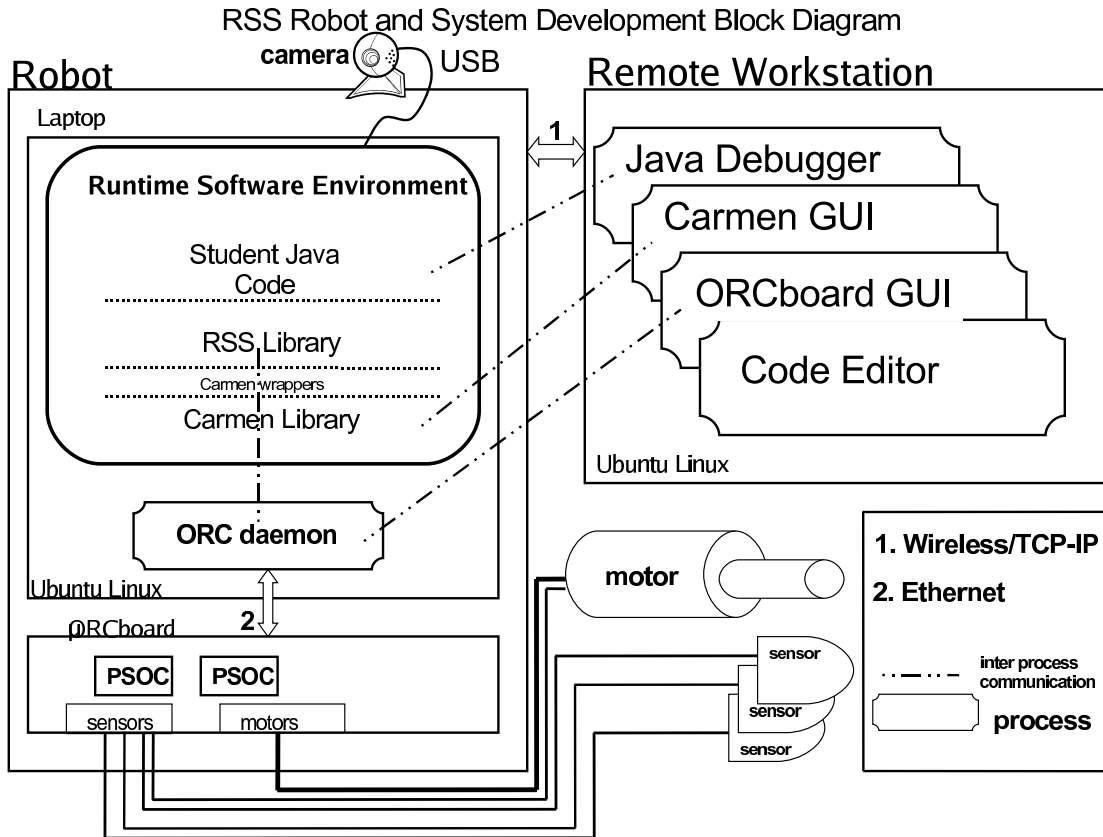


Figure 1: RSS Robot and Development System Block Diagram

The high-level software controlling your robot will run on a netbook that physically resides on the robot. You'll develop your code in Java using either the netbook or your Sun workstation. While Java is a good choice of language when writing high-level algorithms, it is not an ideal choice for interfacing with the physical world. To bridge the gap between your Java code and the robot's motors and sensors, we use the  $\mu$ ORCboard.

The  $\mu$ ORCboard (at the lower left corner of the diagram) supports the lowest level of computation in your robot, and physically connects all of the robot's computations to its sensors and actuators. The heart of the  $\mu$ ORCboard is a 32-bit ARM Cortex-M3 microprocessor that runs code written in C. The board is powered by either a rechargeable lead-acid battery or an AC power supply. When on, the  $\mu$ ORCboard should always be connected to its battery, and optionally to the AC power as well. *Never run the  $\mu$ ORCboard solely off AC power; doing so may damage the board.*

The  $\mu$ ORCboard is the place where the "brain meets the body" of your robot. Any motor or sensor has some set of wires, which are physically attached to appropriate connection points on the  $\mu$ ORCboard, which then makes that device available to the robot software. (One exception is the camera, which is such a high-bandwidth sensor that it needs a dedicated USB connection to the laptop, bypassing the  $\mu$ ORCboard.)

*Deliverables: None, but be prepared for questions from the staff at various checkoff points.*

### 3 Configuring your Netbook and Sun Workstation

Both the netbook and the Sun workstation run Linux, specifically Ubuntu 10.04. Each has an account setup for the `rss-student` user with a default password. As a group, you need to decide on a password to replace the default

password.

1. Decide on a common password for your group. This password should be strong: include numbers and letters, upper- and lower-case, and a symbol or two. Both your Sun station and netbook have network connections to the outside world that can be exploited by hackers. A weak password leaves your computer more likely to be compromised.
2. Log-in to your netbook using the `rss-student` account and the default password provided by the course staff.
3. Open a terminal window (Application → Accessories → Terminal) and run `passwd`. The `passwd` command will ask you to verify your old password and then ask for your new one.
4. Log-out and re-log-in to test your new password.
5. Repeat steps 2–4 for your Sun workstation.

*Deliverables: None.*

## 4 Learning about the $\mu$ ORCboard

*If you know your enemy and know yourself, you need not fear the result of a hundred battles.*  
— Sun Tzu, The Art of War

1. Fill in the blanks in the checkoff sheet with either a short description of the component or a fitting name for that component. Refer to the physical board or online documentation as needed. (Hint: For several of them, the answer is quite simple and literally right under your nose.)

*Deliverables: Fill in the blanks, using the provided vocabulary as a guide. Be prepared for questions from the staff.*

## 5 Powering up the $\mu$ ORCboard and Connecting to a Computer

1. Connect the provided 12V battery to your  $\mu$ ORCboard using the provided fused battery cable. Be sure that the red wire of the battery cabled is connected to the positive terminal of the battery and that the black wire is connected to the negative terminal of the battery. *Reversing the polarity will likely destroy your  $\mu$ ORCboard.*
2. Switch on your  $\mu$ ORCboard. Verify that the power LED turns on and the activity LED flashes rapidly.
3. Connect your  $\mu$ ORCboard to your netbook using an Ethernet cable.
4. From a terminal on your netbook, run the program affectionately known as OrcSpy:

```
orcspy
```

Familiarize yourself with its graphical user interface (GUI), shown in Figure 2.

## 6 Scope and Multimeter Training

We now want to review multimeters and (re)introduce oscilloscopes. Oscilloscopes are a very useful tool for debugging circuits, allowing you to see how the voltage in your circuit changes with time. For this part of the lab, we will investigate how the  $\mu$ ORCboard provides power to drive the motors. You will be responsible for exploring the oscilloscope nearest to you with your team.

The  $\mu$ ORCboard controls motor speed using a technique called pulse-width modulation (PWM). We'll learn much more about this next week, including implementing a PWM controller, but all you need to know about PWM for now is that it uses a time-varying waveform to control the average power received by the motor. Our goal will be to measure the properties of this waveform. First, let's see what we can learn from employing the multimeter included in your box.

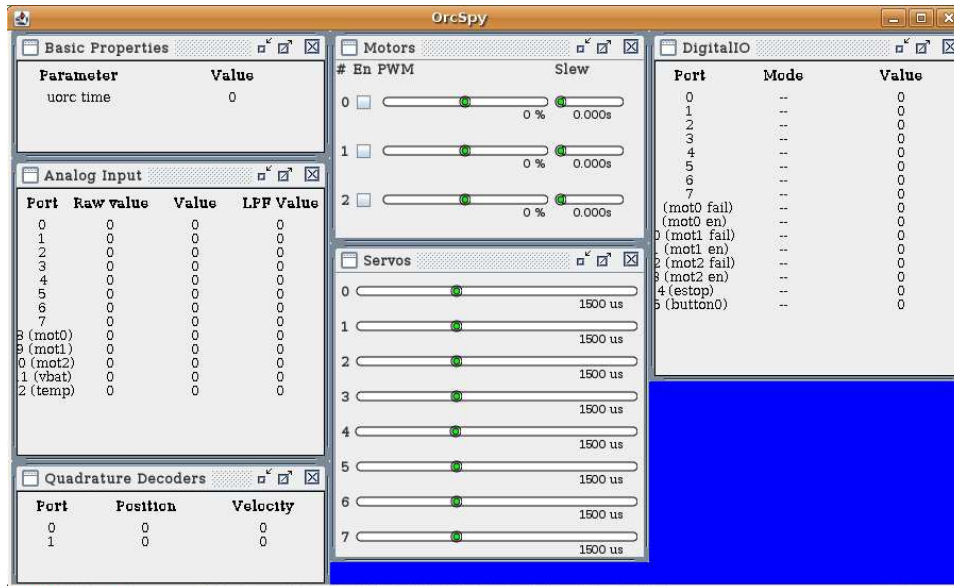


Figure 2: OrcSpy GUI (Graphical User Interface)

## 6.1 Multimeter

- In order to measure the PWM, we need to find where the motor output channels are located. Use the  $\mu$ ORCboard schematic that you completed to locate them on the actual  $\mu$ ORCboard. Once you have found the motor outputs, select the first motor output (MOT0) and hold one of the probes to each of the MOT0 pins on the board. You will need to complete the circuit with the multimeter if you want to get a reading on the PWM - we're imitating a motor with the multimeter.
  - After setting the probes to the motor output, check that your settings on the meter are ready to read the PWM output:
    - Make sure that the two probes are plugged into the lower two ports.
    - Set the dial on the multimeter to the 20V setting in the DC voltage measurement section.
    - When done with any and all measurements on the multimeter, be sure to set the dial back to the OFF position - so as to conserve batteries.
  - With the multimeter properly calibrated for our signal, we can now set out to measure the outputs generated by the  $\mu$ ORCboard. To get the  $\mu$ ORCboard to output a signal, use OrcSpy as in the previous section. Move the motor control slider corresponding to the port to which you connected the oscilloscope and notice how the multimeter reading changes.
  - The PWM signal should be present at the  $\mu$ ORCboard's motor connector, and you should be able to measure a non-zero voltage there using your multimeter. Record the multimeter voltage readings for the following PWM settings: -100%, -80%, -60%, ..., 100%. Fill in your checkoff sheet with the measured values.
  - Note: if you're getting negative voltages for positive PWM inputs and positive voltages for negative PWM inputs, your probes are most likely reversed. Swap the placement of the probe tips and try again.
- That'll do for our work on multimeters. As we move on to oscilloscopes, keep the following questions in mind:
- Is there a trend among the voltages you recorded from the  $\mu$ ORCboard? What are the highest voltages that you read? How does this relate to the power supply that you are using?

- What can the multimeter measurements tell us about PWM? Given what the device shows, how would you define PWM?
- Given the voltage readings, what do you expect the motor to do at each of the PWM settings you measured? (Don't write them all out, but think about how it should behave.)

## 6.2 Oscilloscope

If you have questions about how to use the oscilloscopes in lab, consult the HP 1662AS Logic Analyzer/Oscilloscope Overview handout or the course staff.

1. Boot one of the oscilloscopes. Once there's something displayed on the screen, you should be ready to take some measurements. Take the probe plugged into CH 1 and attach it to the first motor port (MOT0). The probes we use have an alligator clip to attach to ground and a main probe tip that has a removable hook on the end. Clip the ground probe to one of the motor port pins, then hook the main probe tip to the other pin.
2. After attaching the probe to the motor output, we now need to adjust the settings on the scope to handle our signal. The steps for doing this are:
  - Make sure that the signal is visible and centered. using the intensity, focus, and position controls.
  - Change the sec/div and volts/div to fit the entire signal in the window. Recall the voltage strength you detected with the multimeter - make sure that your scope will be able to show you the min and max signal voltages without needing constant adjustment. As for the period of the signal, you'll need to find this empirically once you've started generating PWM signals.
  - Check that the trigger is configured in normal, not automatic mode.
  - If you have any questions about setting up the oscilloscope, ask a staff member.
3. With the scope properly calibrated for our signal, we can again generate a signal with the  $\mu$ ORCboard. Recall that, to get the  $\mu$ ORCboard to output a signal, use OrcSpy as in the previous section. Move the motor control slider for MOT0 and notice how the display changes.
4. If the PWM is displaying correctly, we can measure the properties of the signal. In particular, we want to find:
  - The peak-to-peak period;
  - The duration of the high voltage signal per period;
  - The duration of the low voltage signal per period (note that this is simply the total period minus the duration of the high voltage signal); and
  - The magnitude of the signal ( $\Delta V$  from the high to low).
5. As before, record the measurements for all PWM values of interest (-100%, -80%, -60%, ..., 100%). Fill out the rest of the table on the Quest 1 Log handout with these recordings.

Our tutorial on working with oscilloscopes is now complete. Before contacting a staff member for a checkoff, think over the following questions:

- How does the oscilloscope output change with respect to PWM? Given what it shows, how would you define PWM in your own words?
- How does the display of information vary between the multimeter and the oscilloscope? If there are significant differences, does this mean that one of the devices is lying?
- Given the output of both the oscilloscope and the multimeter, is one of the devices providing a more accurate, more readable, or more comprehensive output?

*Deliverables: Present a staff member with your findings from the multimeter and oscilloscope; explain your measurements and thoughts on the questions above, and s/he will check you off.*

## 7 Creating a Team Wiki Area and Member Pages

*It is the digital projection of your mental self.*

— Adapted from Morpheus, *The Matrix*

The RSS wiki is where you will submit many of your assignments and a good source of information about many aspects of 6.141. The wiki is only accessible with an Athena certificate. When you first access the wiki with your Athena certificate, an account will automatically be created and you will be automatically logged-on. Until you inform the course staff that you have created an account on the wiki, you will be unable to edit it. To prevent your teammates from gaining access to your certificates, you may use Firefox's password manager to protect them. Additionally, so that each team member can log-into the wiki independently, you may create a Firefox profile for each teammate.

1. Configure Firefox to run its profile manager at start-up: *On the Sun station*, right click the Firefox icon in GNOME Panel at the top of the screen, and select Properties in the context menu. In the Launcher Properties dialog that appears, change the text in the Command field to read `firefox --ProfileManager`. Then, click Close.
2. Create Firefox profiles for each group member: Start Firefox by left clicking on the Firefox icon in the GNOME Panel. You'll be presented with the Choose User Profile dialog. Use the Create Profile button to create a profile for every group member. Once each group member has a profile, click Exit.
3. Start Firefox using the link in the panel at the top of the screen and then select one group member's profile in the Choose User Profile dialog. Then, click Start Firefox.
4. Enable the Firefox password manager: In Firefox, select Edit → Preferences and then select the Security icon at the top of the resulting Firefox Preferences dialog. Click to select the 'Use a master password' check-box. Enter a unique and secret password for the active profile. Click OK. Close and Firefox Preferences Dialog.
5. Obtain an Athena certificate: Navigate to <https://ca.mit.edu/ca/> and follow the instructions.
6. Log-into the wiki: Navigate to <https://courses.csail.mit.edu/rss/wiki/>. Ignore the security warning. Then, if asked, provide your password from Step 3 and instruct Firefox to access the site with your newly acquired Athena certificate. Once you have access to the wiki, you'll see that it knows your identity as indicated by the display of your name near the top right of the page.
7. Close Firefox and repeat Steps 2—6 for each teammate. Leave your last teammate logged-into the wiki.
8. Repeat Steps 1—7 on your netbook.
9. Give a staff member a written list your full names as they appeared in the upper right of the wiki when you were each logged-in. The staff will then proceed to give you write access to the wiki so that you can continue with the next step.
10. You will see that each group has a generic team wiki page, each containing a generic page for each team member. It is now your responsibility to populate your team and member wiki pages.
11. We recommend that you start by working on the team wiki page while you're all in the lab. Make sure that the following sections are set up and formatted to your team's tastes:
  - Rename your team to something more epic than "Team N" - you can change this again later, of course, but we prefer awesome names to the placeholders we provide.
  - Upload a fitting picture to represent your team. For example, you could use your team's favorite robot. Add it to your team page. You can use Google Image Search to find a picture; we just want you to have practice uploading files to the wiki.
  - You can leave the team inventory alone for now - you'll place completed assignments, uploaded image links, and your weekly wiki reports into the appropriate pouches later in the class.
  - Fill out the Availability section in a manner that allows for clear, concise viewing of each team member's strict time commitments. A sample format is provided - feel free to use it, embellish it, or completely redesign it; so long as we know when you're free, we'll be happy.

- Lastly, we invite you to fill out a bio for your team wiki page. This is a great place to jot down your collective background in robotics, recall your biggest kills, or remind the Lab Authorities that this class is serious business (not the glorified gaming experience they seem to think it is).

12. Next, we can move on to the key parts of the wiki. Each member of your team is responsible for his/her own member page, and all of them will need to be updated regularly, so save the URLs!

- First off, fill out the General Information section with your name, E-mail address, mobile phone number, specialization (hardware, software, firmware, kitchenware, etc.). If you're feeling creative, add any other key attributes that you feel should be recorded. Please keep the contact information (at least) up-to-date as the year progresses.
- Next, update the Saved Progress section with the hours that you have worked this lab session. All future hours that you put into this class should also be logged here (lectures excluded). We will be using this chart to keep track of how long assignments take you and your team, so be sure to update it regularly.
- The personal inventory is for any uploads that you would like to have access to as you work through the class. This includes the existing pouches for your reports and reflections, as well as additional space for extra supplies you'll need on the journey ahead.
- The skills section is where you'll record your skills at the beginning and end of each lab. Relevant skills will be outlined in each new lab handout. There is a chart for this lab included, so you can practice by incorporating values in for your relative proficiency with each skill **as of the beginning and end of this lab** in each of the following areas, on a scale of 1 to 5 (1=Not at all proficient; 2=slightly proficient; 3=reasonably proficient; 4=very proficient; 5=expert):
- Ignore the Achievement section for now; if you redesign your page, we suggest that you leave space for it somewhere. The staff will populate this section as the semester progresses.

13. That's it! Once your team has completed its team wiki, and each of your team members has created and filled out his or her own member page, you're ready to venture forth.

*Deliverables: Show a staff member your team wiki entries. He or she can then check you off for your work.*

## Deliverables

Turn in a completed Lab 1 checkoff, having analyzed the  $\mu$ ORCboard, worked with the multimeter and oscilloscope, and created all of the wiki pages for your team. This deliverable must be done before the deadline listed at the top of this handout.

## Presentation

Note that you will be giving verbal briefings for all labs in this class, except this one! As you run through the labs, be sure to jot down notes from your checkoff sheet, and record useful data or plots in your wiki, as necessary for your preparations. Your presentations should cover all of the key points in this lab, and must include a speaking role for every member of the team. The CI-M lecture on Wednesday, February 9th, will cover the staff's expectations for these briefings.

## RSS-I Glossary and Acronyms

**Carmen** The Carnegie Mellon Robot Navigation Toolkit is an open source collection of software for mobile robot control implemented in C. Its programs control the movement of the robot and accept inputs from the sensors plus handle more sophisticated tasks like mapping, navigation and localization. Carmen also includes a simulator. An update of Carmen has been developed for RSS and will later be released. For more information see: <http://www-2.cs.cmu.edu/~carmen/>

**Daemon** A daemon (pronounced DEEmuhn) is a continuously-running, simple program designed to handle periodic service requests. A daemon may generate additional requests for other programs or processes.

**Driver** (From whatis.com) A driver is a program that interacts with a particular device or special (frequently optional) kind of software. The driver contains the special knowledge of the device or special software interface that programs using the driver do not.

**Ethernet protocol** A method of connecting computers in a local area network.

**GUI** Graphical User Interface

**IDE** Integrated Development Environment

**$\mu$ ORCboard**  $\mu$ ORC = “micro Our Robot Controller”. The “ $\mu$ ORCboard” robotics interface board was designed and manufactured for the MASLab competition and is an evolved version, (hence the “micro”), of earlier year’s boards. More information is available at [http://www.orcboard.org/wiki/index.php/Main\\_Page](http://www.orcboard.org/wiki/index.php/Main_Page).

**PCB** Printed Circuit Board. See <http://www.lvr.com/pcbs.htm> for more information.

**Serial protocol** A communication standard where information is sent one at a time as opposed to in parallel.

**TCP-IP** (from whatis.com) TCP/IP (Transmission Control Protocol/Internet Protocol) is the basic communication language or protocol of the Internet. Its higher layer, Transmission Control Protocol, manages the assembling of a message or file into smaller packets that are transmitted over the Internet and received by a lower layer that reassembles the packets into the original message, even if they arrive out of order.

**USB** Universal Serial Bus, a protocol supporting connection of external devices (such as digital cameras, scanners, and mice) to personal computers with data transfer rates of up to 12Mbps (million bits per second).