Massachusetts Institute of Technology

Robotics: Science and Systems I

Lab 1: System Overview and Introduction to the μORCboard

Distributed: February 4, 2015, 3:30pm
Checkoffs due: February 9, 2015 3:00pm

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

By the end of this lab, you will have a powered, functioning μORCboard 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 website):

- (1-B) Scope Diagram (one per table)
- (1-C) HP 1662AS Logic Analyzer/Oscilloscope Overview (one per table)
- (1-D) Lab 1 Checkoff (one per student)

Lab materials:

- A μORCboard, battery, battery cable, and an AC power adapter;
- A multimeter;
- An logic analyzer/oscilloscope and probes; and
- A 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 will return to this diagram in future labs.

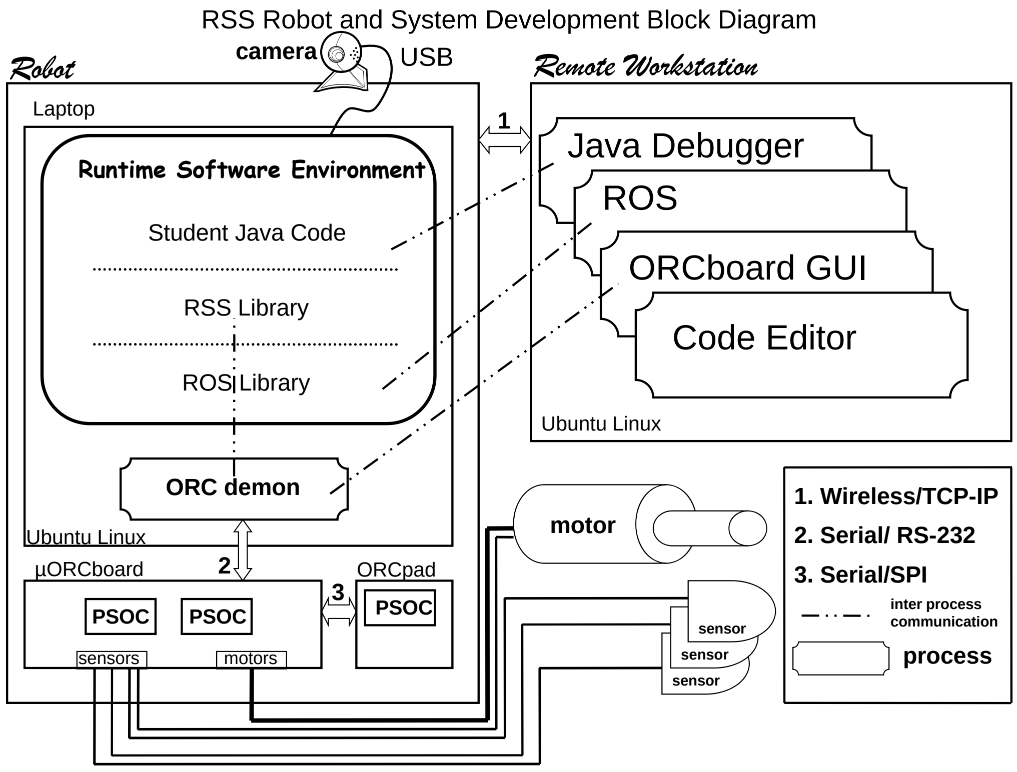


Figure 1: RSS Robot and Development System Block Diagram

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 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 μORCboard.

The μORCboard (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 μORCboard 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 μORCboard should always be connected to its battery, and optionally to the AC power as well. *Never run the μORCboard solely off AC power; doing so may damage the board.*

The μORCboard 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 μORCboard, 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 μORCboard.)

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

3 Boot your Netbook and Workstation

Both the netbook and the workstation run Linux, specifically Ubuntu 11. Each has an account setup for the `rss-student` user with a default password.

1. Log-in to your netbook using the `rss-student` account and the default password provided by the course staff.

Deliverables: None.

4 Learning about the μORCboard

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.)

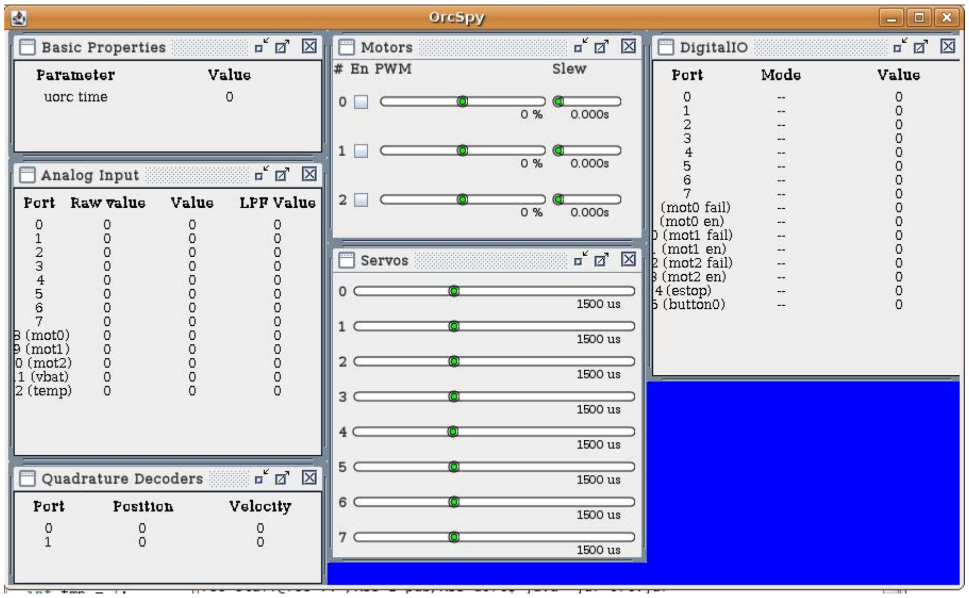
Deliverables: Fill in the blanks, using the provided vocabulary as a guide. Be prepared for questions from the staff.

5 Powering Up the μORCboard and Connecting It to a Computer

1. Connect the provided 12V battery to your μORCboard using the provided fused battery cable. Be sure that the red wire of the battery cable 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 μORCboard.*
2. Switch on your μORCboard. Verify that the power LED turns on and the activity LED flashes rapidly.
3. Connect your μORCboard 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.



Parameter	Value
uorc time	0

Port	Raw value	Value	LPF Value
0	0	0	0
1	0	0	0
2	0	0	0
3	0	0	0
4	0	0	0
5	0	0	0
6	0	0	0
7	0	0	0
8 (mot0)	0	0	0
9 (mot1)	0	0	0
0 (mot2)	0	0	0
1 (vbat)	0	0	0
2 (temp)	0	0	0

Port	Position	Velocity
0	0	0
1	0	0

#	En	PWM	Slew
0	☐	0 %	0.000s
1	☐	0 %	0.000s
2	☐	0 %	0.000s

0	1500 us
1	1500 us
2	1500 us
3	1500 us
4	1500 us
5	1500 us
6	1500 us
7	1500 us

Port	Mode	Value
0	--	0
1	--	0
2	--	0
3	--	0
4	--	0
5	--	0
6	--	0
7	--	0
(mot0 fail)	--	0
(mot0 en)	--	0
(mot1 fail)	--	0
(mot1 en)	--	0
2 (mot2 fail)	--	0
3 (mot2 en)	--	0
4 (estop)	--	0
5 (button0)	--	0

Figure 2: OrcSpy GUI (Graphical User Interface

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 μORCboard provides power to drive the motors. You will be responsible for exploring the oscilloscope nearest to you with your team.

The μORCboard controls motor speed using a technique called pulse-width modulation (PWM). We will 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.

6.1 Multimeter

1. In order to measure the PWM, we need to find where the motor output channels are located. Use the μORCboard schematic that you completed to locate them on the actual μORCboard. 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 are imitating a motor with the multimeter.
2. 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 battery life.
3. With the multimeter properly calibrated for our signal, we can now set out to measure the outputs generated by the μORCboard. To get the μORCboard 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.
4. The PWM signal should be present at the μORCboard'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.
5. 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 μORCboard? 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 μORCboard. Recall that, to get the μORCboard 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 (ΔV 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.

Deliverables

Turn in a completed Lab 1 checkoff, having analyzed the μORCboard and worked with the multimeter and oscilloscope. 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 (to be set up in the next lab), 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 next scheduled Forum will cover the staff's expectations for these briefings.

RSS-I Glossary and Acronyms

ROS ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. `http://www.ros.org/wiki/`

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

μORCboard μORC = "micro Our Robot Controller". The "μORCboard" 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`.

PCB Printed Circuit Board. See `http://en.wikipedia.org/wiki/Printed_circuit_board` 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).