Week 1 Exercises for Robot Project

You will be using a robot based on the DE2 board for your final design project. Previous 2031 semesters have implemented parts of the FPGA design which provide sensing and control functions in the robot, and you will have a starting point that is based on much of that previous work.

One of the most useful previous projects is a robot self-test program. Part of these initial exercises will help you familiarize yourself with the robot self-test, because there may be times later in the semester when you need to make sure that the robot has no hardware problems. The self-test program will be sufficient to prove whether or not there are any malfunctions.

The later parts of these exercises will introduce the Quartus project that is a starting point for your project, and give you some experience controlling the robot.

Preparation

1. Have one person from your team use a Buzzcard to check out a robot and a USB cable from the UTAs, and gather around a workstation, making room for the robot where all of you can see it.
2. Download the *DE2Bot User's Manual* from Canvas (in the Project module) so that you can refer to it while completing these exercises.
3. Turn off (down) all of the slide switches of the DE2 board on the robot. Look under the robot to see the slide switch that turns it on. Once you learn its location, you will typically just reach under the robot to turn it on, but make note of the two things in the vicinity:
    - The charge port.
    - An empty square hole. This is normal (it is the location of the old Amigobot's data port).
4. Go ahead and turn the robot on. It should beep, and the DE2 board should come on. Always use the power switch under the robot to turn it off, not the DE2 board's power button.

Explore the Robot Self-tests

The robot self-test allows you to test most of the robot's hardware, including the motors, sonar, and wheel encoders. If you ever suspect that the robot is malfunctioning, you should run a self-test.

1. Work through the steps in the robot manual's Section 2, "Self-test Operation," through all of the "Automated Self-test" section. Repeat the automated self-test as necessary, with different team members, until everyone understands the procedure. Have one team member demonstrate the self-test to a TA for a check-off. ✓
2. Enter "manual test mode" as described in the robot manual, and enter the sonar test. Enable sonar 0. Use a ruler or other object to measure a distance 3-5 ft away (for example if you don't have a ruler, a standard sheet of paper is 11" tall, or the carpet squares in the lab are 2' square). Place a flat object such as a book at the measured distance and record the displayed distance reading, converting from the displayed hexadecimal to decimal. How accurate is the reading? Record the error on the check-off sheet and request a check-off from a TA. ✓

Explore the Quartus Files

You are given a Quartus project with a complete SCOMP and a full complement of SCOMP peripherals to interface with the robot.

1. Recall (and review if necessary) section 3.1 in the manual ("Changes to SCOMP"). It is important to know the differences between your lab 8 SCOMP and this new version.
2. Download the file DE2Bot_v3_Fall18.zip from the Project Resources on Canvas, and extract it to a directory **on the computer** (ideally C:\users\[your username]).
3. Open the DE2Bot project file and open the top-level design file, DE2bot.bdf. Note the many changes to the Simple Computer system since lab 8. In particular, many new I/O peripherals have been added to control and interface with the robot.

Run Example Assembly Code

1. Open and examine the assembly file that SCOMP will run, SimpleScanDemo.asm (noting that SimpleScanDemo.mif is the file referenced in SCOMP's altsyncram). Note that this program fulfils the requirements listed in the robot manual section "Good Practices for Robot Programming." Some tips:
    - Some things should *always* be done at the beginning of the code, like stopping the motors and checking the battery voltage.
    - Other things, like waiting for the user to push a pushbutton, are not required, but usually make using the robot easier. Note that toggling SW17 to enable the motors is enforced by hardware, but that safety mechanism can be monitored from the code (as is done in this example code).
2. Assemble the software with SCASM, compile the Quartus project, and download it to the robot.
3. Read the assembly code, understand what it should do, and confirm that the robot does those things. Demonstrate the behavior to a TA for a check-off. ✓

Modify Example Assembly Code

As you write assembly code in the following section, **revisit the robot manual** for information on interfacing with the robot devices and features. Don't just guess at what the various robot I/O devices do and how to

interact with them. Section 3.3 in the manual provides a list of the IO devices, and section 3.4 explains most of the devices in more detail.

Modify the existing code or write your own to do the following:

1. After facing the nearest sonar reading, move in that direction until within 10" of the reading.
2. Stop and beep.

Note that there are several ways you can accomplish this. For example, you could use the front sonar sensors to stop short of the object, or you could use the recorded value to determine an appropriate movement distance.

Demonstrate your program for a check-off. ✓
Make sure that someone keeps your group's check-off sheet safe; your GTA will record it next lab meeting.

Begin Project Work

There are no more explicit guided steps or check-offs for the rest of the semester, which means it is up to you to work diligently, especially in this first week when the project deadline seems far away. Use any remaining time this week to plan the project, experiment with the robot, and work on the proposal. You already have the project description and timeline, so today you could:

- Determine your team's makeup. Find out between yourselves if anyone excels at things like assembly code, algorithms, documentation, presentation design, data parsing and interpretation, project management, software or hardware debugging, source control, etc., and use each team member's strengths by allowing them to lead main areas of project work, calling on other members for input when needed.
- Lay out a timeline for early project tasks – e.g. sense objects, move to a particular location, etc. Determine what can be done outside of lab and what needs to be done in lab. Use time out of lab as much as possible so that you can use your time with the robot to its full potential. Begin assembling your goals into a Gantt chart.
- Display a sonar sensor's reading to a display (perhaps by using the troubleshooting mode of the self-test) and move objects in front of it to get an idea of what the sensor data looks like. Try different objects at different angles, distances, and offsets from the center of the beam. Try sensing walls, furniture, and the baffle, identifying situations that might be troublesome during the demo.
- Use the provided serial communication code to transfer scanned data to MATLAB and take a look at it.
- Investigate the new SCOMP instructions, such as ILOAD and ISTORE. Consider if you want to add any new instructions.
- Delegate upcoming tasks and record them in a consensus form. For example: one or two team members might take the lead on creating the proposal. They will need input from the whole team as you all decide the basic goals of your project, and they might (for example) ask someone focusing on the assembly code to create some slides related to that. At the end, everyone on the team should proofread the presentation and you should all practice it together.

These check-offs must be completed **Friday, November 2** by close of lab (check the lab schedule for open lab times). If you finish outside of lab, return this sheet to your GTA at the start of your next lab section.

Team Name _____

Team Members _____

_____

Checkoff	**TA initials**	**Date/time**
1. Run through entire automated self-test		
2. Difference between sonar reading and reality (include units): Reading:_____ Expected:_____ Error:_____		
3. Show robot running the program provided with the project files.		
4. Show robot facing and moving toward the nearest object.		