ME218c
Smart
Product
Design

ME 218c Spring 2009 Project
Tele-Operated Water Rugby

Project Preview on May 21, 2009 6-10 PM in SPDL.
Grading Session on May 27, 2009 9 AM-12 PM in Terman Pond.
Project Presentations on May 28, 2009 in Terman Pond starting at 5:00 PM.

Goal:

The goal of this project is to provide a framework in which you can apply your knowledge of microcontrollers and multi-processor communications to a task that will provide an enjoyable experience for the users and the observers.

Purpose:

The underlying purpose of this project is to provide you with an opportunity to gain experience in integrating all that you have learned in the ME218 course sequence, with an emphasis on the new material in ME218c.

The Task:

Design and build a Tele-Operated Water Rugby Player (TOWRP) and a companion Creatively Operated Active Control Helper (COACH, i.e. remote controller). Groups of TOWRPs will operate in Terman Pond and cooperatively strive to move an oversize rugby ball into the opponent's goal.

Specifications

General:

- ☐ Each team will construct a TOWRP and a COACH.
- ☐ The TOWRPs are devices capable of navigating in Terman Pond while it is filled with water, and maneuvering an oversized Rugby ball.
- ☐ The COACHs are the remote controllers for the TOWRPs.
- ☐ TOWRPs and COACHs are interoperable – any COACH is capable of controlling any TOWRP.

Game Play:

- ☐ A game will be a competition between two teams each composed of four COACH/TOWRP pairs and a marauding COACH without a corresponding TOWRP. The makeup of the teams will vary for each game.
- ☐ Each game will begin with the rugby balls and goals in the positions shown in Fig. 1 and the TOWRPs placed in the Start Zone for their team.
- ☐ For a given game, the TOWRPs will begin paired with the COACH produced by that team.
- ☐ The goal of the game is to get one of the rugby balls into the opponent's goal. The first team to place a ball into their opponent's goal wins the game.
- ☐ If the Morale status of a TOWRP reaches a value of -10, the TOWRP will send a Mutiny In Progress message to the COACH that is currently in control.
- ☐ If, within 3 seconds after receiving a Mutiny In Progress message, the operator of the COACH does not successfully perform the actions to put down the mutiny, the mutiny will be successful and the coach will lose control of the TOWRP. At this point, the TOWRP will broadcast an I AM Available message via the RF link and await an Assertion of Command message that can be sent by any COACH not currently controlling another TOWRP.

The Field:

- ☐ The Field is comprised of a portion of Terman Pond, initially measuring approximately 26 ft. wide by 30 ft. long (see Figure 1). One fountain spout will be located at the approximate center of the Field. Every effort will be made to ensure that the fountain will be off during grading sessions and public presentations.
- ☐ The absolute boundaries of the Field are formed by the cement border to the pond.

- ☐ The Start Zone is the region adjacent to the cement border at the centerline of the Field. All TOWRPs will be placed in the Start Zone corresponding to their team at the beginning of each Game.
- ☐ The goals will be floating, un-tethered, at the positions shown in Fig. 1 at the start of the game.

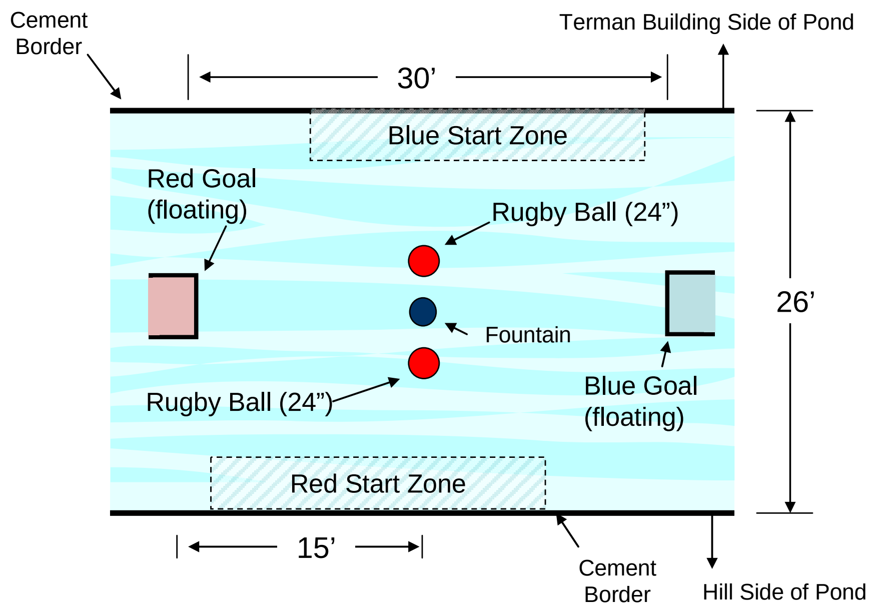

Figure 1: Field layout and dimensions

The TOWRPs:

- ☐ Each TOWRP must be capable of moving under its own power within Terman Pond. Terman Pond will be filled with water (its normal state) at the time of the grading and the public presentation. Every effort will be made to ensure that the fountains are off during the events.
- ☐ TOWRPs must be battery powered and operate without a tether.
- ☐ Control of TOWRP functions must be achieved via a COACH using the class standard RF data link.
- ☐ If a TOWRP receives a Jettison the Crew command from the paired COACH, it will stop accepting commands from the COACH, broadcast an I AM Available message and await an Assertion of Command message directed to it by any COACH not currently controlling another TOWRP.
- ☐ Each TOWRP must be clearly marked with the team's number, using digits that are at least 4" in height.
- ☐ Each TOWRP must carry a highly visible indicator of which game team (red or blue) that it is being controlled by at any given point in a game. This indicator must be under software control and clearly visible in sunlight at a distance of 20'
- ☐ TOWRPs must incorporate an easily accessible switch that disables all propulsion systems.
- ☐ The top view projection of the TOWRP must fit into a rectangle 24"x18". Height is not restricted.
- ☐ If a TOWRP is not actively under control (see Game Play), it must accept the first Assertion of Command message that it receives directed to it and submit control of the TOWRP to the COACH that issued the Assertion of Command.
- ☐ TOWRPs must incorporate a class standard foam bumper around their perimeter, and must be tolerant of moderate bumping from other TOWRPs. The bottom edge of the foam bumper must be at a height between 0" (the water line) and 1.5"inches above the water line of the TOWRP.

- ☐ During the playing of a game, each TOWRP will maintain an "Morale" status number that will be reported back to the COACH over the RF link using the class standard communications protocol. The value of the Morale status will start at zero at the beginning of a game and increment/decrement (capped at +-10) at a rate set by the percentage of maximum speed commanded by the COACH:

0-30% of maximum:	Morale increments at a rate of 1 count per second.
30-60% of maximum:	No increment or decrement.
60-90% of maximum:	Morale decrements at a rate of 1 count per second.
over 90% of maximum:	Morale decrements at a rate of 2 counts per second.

The COACH:

- ☐ Each team will design and construct a COACH (remote controller) that will relay commands from a human to a TOWRP, receive and display status information from the TOWRP.
- ☐ Each COACH must be able to control the function of any TOWRP in the class. Commands will arrive over the class standard RF link.
- ☐ COACHs will provide bi-directional communications between itself and the TOWRP that it is controlling.
- ☐ COACHs must be battery powered, and shall have sufficient battery capacity for at least 8 hours of continuous operation. The report should show documentation and calculations to support meeting this requirement.
- ☐ COACHs must be free standing, and completely borne by the helmsperson.
- ☐ Input to the COACH should involve at least 3 sensing modalities (e.g. position, force, audio, acceleration, etc.). Use of unorthodox methods is encouraged.
- ☐ The actions required by the user of the COACH to issue commands to the TOWRP should be inventive and interesting for the audience to watch. Use of actions that make the operator look and feel foolish are encouraged.
- ☐ Each COACH must, at a minimum, provide speed and steering commands to the TOWRP it is controlling. A single command that has not been anticipated by this specification may be specified by the communications committee. What a given TOWRP does with the extra command is up to each team. However, every TOWRP must accept the extra command without getting confused.
- ☐ The COACH must provide a visible indication to the operator of the TOWRP number it is currently controlling.
- ☐ The COACH must provide a visible indication to the operator of the Morale status of the TOWRP (See Game Play).
- ☐ The COACH must provide a visible indication to the operator of a Mutiny in progress on board the TOWRP (see Game Play).
- ☐ The COACH may issue commands to a TOWRP at a rate no greater than 5 Hz.
- ☐ The COACH will have a switch to indicate the team to which it belongs for a given game.
- ☐ Each COACH will implement a special series of actions necessary to put down a mutiny (see Game Play). These actions must be capable of being preformed in not less than 1 nor more than 2 seconds.
- ☐ The COACH will have a specific action on the part of the operator that will issue a Jettison the Crew command to the TOWRP that it is currently controlling.
- ☐ COACHs should be intuitive to operate, and/or have sufficient visual instructions that a typical spectator (even a non-engineer) would be able to learn its controls within the time span of a single game.

Communications:

- ☐ Communications will take place over an SPDL-supplied 802.15.4 radio (Xbee24) using the Non-Beacon API mode of operation.

- ☐ Once a game begins, communication will take the form of bi-directional communications between a TOWRP and its designated COACH. All commands will be sent from the COACH to the TOWRP and executed by the TOWRP. TOWRP status messages will be sent from the TOWRP to the COACH and displayed there for the operator.
- ☐ Each TOWRP and COACH will be assigned a unique ID that will be used to program the Source Address of each radio.
- ☐ The communications between the COACH and the TOWRP will take place using a standard communications protocol that will allow any COACH to provide control for any TOWRP.
- ☐ The details of the communications protocol will be defined and specified by a Communications Committee, which will consist of one member from each project team. The specification must be in a written form and with sufficient detail that someone sufficiently skilled in ME218 material could implement it.
- ☐ In order to better balance the workload and learning among team members, each of the following tasks must be completed by a different member of the team: serve on the communications committee, implement communications on the TOWRP, implement communications on the COACH.

General Requirements:

- ☐ At a minimum, either the COACH or the TOWRP must contain two actively communicating processors. There is no class imposed upper limit on the number of processors employed.
- ☐ You are limited to an expenditure of **$150.00/ team** for all materials and parts used in the construction of your project. Materials from the lab kit or the Cabinet Of Freedom do not count against the limit. All other items count at their fair market value.
- ☐ A project logbook must be maintained for each group. An on-line blog is appropriate for to meet this requirement as long as it is made available to the teaching staff for review. This book should reflect the current state of the project, planning for the future, results of meetings, designs as they evolve etc. The project logbook will be collected at irregular intervals for evaluation.
- ☐ A report describing the technical details of the system will be required. The report should be of sufficient detail that a person skilled at the level of ME218c could understand, reproduce, and modify the design. The report must be in website format, and be suitable for posting on the SPDL site.
- ☐ TOWRPs based substantially on purchased boat platforms are not allowed.
- ☐ All projects must respect the spirit of the rules. If your team is considering something that may violate them, you must consult a member of the teaching staff.

Safety:

- ☐ Both the TOWRPs and the COACHs should be safe, both to the user and the spectators.
- ☐ Intentionally disabling or damaging other TOWRPs is not allowed. Un-allowed actions include, but are not limited to, the following: ramming at excessive speed (as determined solely at the discretion of the teaching staff), swamping, spraying, and/or sinking.
- ☐ TOWRPs will be exposed frequently to water. Although teams are prohibited from intentionally introducing water onto or into other TOWRPs, mishaps and unintentional water ingress *will* occur. Electronics, actuators and energy storage devices (e.g. batteries) do not typically fare well in the presence of water. Plan on it. Design accordingly.
- ☐ The teaching staff reserves the right to disqualify any device considered unsafe.

Check-Points

Design Review:

During class-time on **05/05/09** we will conduct a design review. Each group should prepare a few slides (scans of sheets of paper are fine) showing your proposed designs for both the TOWRP and the COACH. At this time, initial calculations are required for estimating the mass of your proposed TOWRP as well as any water displacement calculations relevant to your design. You will present these in 556 to the class, members of the teaching staff and coaches who will provide feedback.

First Draft of Communications Standard:

Due by 5:00 pm on **05/05/09**. Ed will meet with the communications committee on the evening of 05/06/09 to provide feedback on the specification.

Communications Standard:

Due by 5:00 pm on **05/08/09**. This is the working draft of the communications standard.

First Check-Point:

On **05/12/09**, you must demonstrate
1) The ability to receive and correctly decode and respond to commands from the COACH (simulated inputs are acceptable at this time).

2) Your team's TOWRP. At this check-point, you will demonstrate that your TOWRP platform has been built, and is capable of bearing the approximate weight of all the necessary components it will carry when complete. It is encouraged, but not required, to demonstrate working propulsion and steering subsystems.

The final working version of the communications standard is due. No further changes are allowed to the standard. This protocol will be evaluated with respect to its completeness and suitability for the proposed system. **Note:** this is a functional evaluation only. The focus should be on demonstrating **functional** hardware and software.

Second Check-Point:

On **05/19/09**, you must demonstrate the ability to communicate all required functionality between your TOWRP and COACH. This will include commands from the COACH to the TOWRP, all status messages from the TOWRP to the COACH.

Project Preview:

At the Project Preview on **05/21/09**, each team must demonstrate (in addition to the 1st & 2nd check-point functionality)
1) The ability to successfully send and execute the drive and steering commands from an operator of the COACH to the TOWRP. COACHs and TOWRPs will be tested independently with TOWRPs and COACHs from other teams.
2) The ability to recognize and Morale status, announce a Mutiny In Progress, send an I AM Available message and accept an Assertion of Command message from another COACH.

Grading Session:

During the Grading Session on **05/27/09**, each team will be required to demonstrate the ability to successfully participate in a game. This will include
1) Navigating a TOWRP from the Start Zone and successfully moving a rugby ball into a goal (in the absence of an opponent).
3) Monitoring, reporting and responding to the Morale status with an I AM Available message.
4) Recognizing and responding to an Assertion of Command message.
5) Recognizing and responding to commands from another COACH after an Assertion of Command message.

Public Presentation:

This will take place on **05/28/09** starting at 5:00 pm in Terman Pond. At this event, members of the public will be allowed to act as operators of the COACHs.

Report:

Draft due on **06/01/09** by 4:00 pm. The final version (with revisions incorporated) is due by 5:00 pm on **06/05/09**.

Evaluation

Performance Testing Procedures:

One or more of the team members will demonstrate the TOWRP and COACH during the first & second check points and project preview. Members of the teaching team will operate the TOWRP and COACH during the grading session.

Grading Criteria:

- ☐ **Concept (15%)** This will be based on the technical merit of the design and coding for the machine. Included in this grade will be evaluation of the appropriateness of the solution, as well as innovative hardware, software and use of physical principles in the solution.
- ☐ **Implementation (15%)** This will be based on the prototype displayed at the evaluation session. Included in this grade will be evaluation of the physical appearance of the prototype and quality of construction. We will not presume to judge true aesthetics, but will concentrate on craftsmanship and finished appearance.
- ☐ **First Check Point (10%)** Based on the results of the performance demonstrated on 05/12/09**.**
- ☐ **Second Check Point (10%)** Based on the results of the performance demonstrated on 05/19/09**.**
- ☐ **Preliminary Performance (10%)** Based on the results of the performance demonstrated during the Project Preview.
- ☐ **Performance (15%)** Based on the results of the performance testing during the Grading Session.
- ☐ **Report (10%)** This will be based on an evaluation of the report. It will be judged on clarity of explanations, completeness and appropriateness of the documentation.
- ☐ **Report Review (5%)** These points will be awarded based on the thoroughness of your review of your partner team's report. Read the explanations, do they make sense? Review the circuits, do they look like they should work?
- ☐ **Log Book (5%)** This will be evaluated by the evidence of consistent maintenance as well as the quality and relevance of the material in the log book.
- ☐ **Housekeeping (5%)** Based on the timely return of SPDL components, cleanliness of group workstations as well as the overall cleanliness of the lab. No grades will be recorded for teams who have not returned all loaned materials.

Gems of Wisdom from Prior Generations

- Get the radio working with the ‘E128 first.
- Do not continue working until the wee hours of the morning unless you absolutely have to because errors propagate when tired. A fresh look at things in the morning will save you a lot of pain at night. Sleep is not a crutch, it is a necessity.
- Put some time into your first prototype. You might be surprised how many things you throw together for testing purposes make it into your final project.
- Label or color-code your connectors so that it’s easy to plug them into the right place. Connectors that can only be hooked up one way (such as Molex) prevent undesirable incidents like reversing the voltage and ground connections and frying components in the process.
- When building networks, add nodes one at a time to better track down “bad nodes”.
- Debugging LEDs are useful for getting feedback on the operational state of PICs.
- A “power central” board is a good thing to have, particularly if you’re dealing with multiple supply voltages. This makes the circuitry cleaner, and can save you from supplying your PIC with 37 volts.

- Think twice before planning to provide PWM for motors with PICs (at least the one with 20 pins). You will need to take care of output compares and timers also.
- You will need to leave some pins on PICs (especially those with only 20 pins) open for debugging.
- Don’t hesitate to add another PIC and SPI communication. It’s really easy.
- Using shift registers for debugging can also be a helpful trick to obtain more information, but it is not good for timing issues.
- Try working during the day (seriously!). Debugging is way easier with a clear head.

- The radio boards runs on 3.3V not 5V. The iButton reader runs on 5V not 3.3V. Design your circuits accordingly and be ready to convert between the two voltages.
- Jameco is NOT OPEN of weekends. Don’t postpone your trip until Saturday – you will be sorely disappointed.
- Don’t be dead set on a theme at the beginning of the project. Let the project theme develop as you move through the project. You’ll be surprised how many great ideas pop up as you go along.
- Hot glue down all soldered wire connections. You’ll lose a lot of time tracking down an error that may end up being a loose/broken wire.
- Write all functions as non-blocking code – no matter where they fit into the flow of the program.
- Just because two points on a circuit look like ground when probed doesn’t mean they are connected.

- Test circuit as it will be implemented in final form, as well as fully integrated.
- Test in environment in which hardware will be used (radios outside, with appropriate distant in between).
- Testing our radio pair in the presence of other active radio pairs revealed problems that didn’t exist when we test alone.
- It’s easy to make a design with bad ergonomics which make it impossible for the user to perform the task. Prototype/try out the user scenario yourself as early as possible.
- Keep circuit diagrams up to date as you make them.
- Use lab notebook so that all information is at one place and teammates can have easy access to it.
- Take a lot of picture as you go.
- Remember to HAVE FUN.

- If you are having intermittent problems (e.g. it works only some of the time) check your connections – especially those connecting your various circuits to a common ground.
- Modularize as much as possible – test all of the components separately before integrating
- Build and test all of your circuits and sensors on breadboard before you make them hard mounted on perfboard.
- When moving your circuits from breadboard to perfboard, rather than dismantling your breadboards, leave your working breadboards intact and buy new components and build entirely new circuits on the perfboard. That way, if something goes wrong once everything is built, you will always have a backup copy of your circuits on the breadboard that you know worked before you integrated everything.
- Isolate your circuits onto individual perf-boards (rather than having a giant perf-board with all of your circuits). Makes it much easier to take them out to debug them.
- A nice pair of wire snips (flush cutters) and wire strippers makes wirewrapping and circuit building in general much easier.
- Do your circuit calculations to make sure you have enough/not too much voltage/current/power
- Have plenty of spare parts ready to go in case something blows at the last minute
- Always take the time to test on the actual competition field at the actual competition location
- Make sure at least two people of the group understand or at least have an idea of each component – mechanical, electrical, software. Doesn’t have to be the same two people, but it insures that if someone’s missing, that the group isn’t stuck.
- Be friendly with other teams – you never know when you’re going to need help.

- Test your wireless communication outside and at range!
- Test your components for interoperability with everyone else’s before game day.
- Source an off-the-shelf housing/controller and gut it. That way you can focus on the electronics and not the mechanical design.
- If you can avoid having to spend time building something by buying an equivalent part, do it!
- Remember your banksel commands and save yourself hours of debugging.
- Make things accessible (i.e. batteries, boards, DIP sockets, etc.) so you don’t have to unscrew things when you need to test, power cycle, or reset things.
- Learn to use assembler macros. They clean up your code visually.

- Don’t use macros when you can use a function-type subroutine!
- Size does matter. The bigger or larger the motions involved in your controller, the more entertaining it will be to watch.
- Do not bury your wireless antenna in a box. Try to keep it out in the open.
- Buy a good pair of wire strippers, preferably ones that can strip 30AWG wire. Your fingers will thank you.
- Try to avoid having to scramble together parts or code for a check-off. This means keeping on top of the project schedule. If you don’t, you will end up throwing away a lot of hours on setups that will not make it to your final design.
- PICs are notoriously difficult to debug. Either source an MPLAB ICD2 (or equivalent), or finish your circuitry early, before writing the bulk of your code. You will find that changing even small things in assembler can cost you hours.

- If a change causes things not to work the first thing you should check is if the code is in the correct bank. It is always a good idea to use a bank select command at the start of every routine rather than assume you'll know where it is.
- Build and test the code in small manageable pieces. If a lot of changes are made at once and the new program doesn't work, it is very hard to isolate the problem without a lot of work.
- Use the debugger. Running routines through the debugger to see what will happen will save lots of time and effort. Getting a routine to work in the debugger usually allows you to assume problems that come up in actual testing are hardware rather than software related.
- When you're tired and everything starts to fail don't forget to check the batteries.
- If you're tired and everything starts to fail and it's not the battery consider going home and looking at it again the next morning rather than changing a lot of code. Often it is some small little change you overlooked and are too tired to notice.
- Make sure all data tables are in the correct location.
- Make use of calls and macros whenever possible to keep the code clean. This also makes repetitive actions easier to code and change.
- Make use of #defines for labeling pins and value as much as possible. This makes it very easy to see what pins are connected to what and allows for the easiest changes. Rather than searching for a specific port and pin throughout the code you only have to change one #define value.

- Don't believe anyone that tells you that the 218C project is less time consuming than the 218b project. It's not.
- Pick your battles early. Learning to program the PIC's and the Zigbees is a lot of work on its own. Trying to add other challenges can be tough.
- Move to solder boards or wire wrap boards as soon as you can. If you are developing simple hardware that you understand well, don't be afraid to solder it on a board. Troubleshooting bad connections on a breadboard is a waste of your time.
- Allocate your pins and subsystems early. A spreadsheet that shows all of your pins is very handy.
- Practice on the course as soon as possible to test your operable range.

- PICs are apparently not designed to be inserted backwards. We recommend against doing this.
- Don’t spend more than a few hours debugging SPI code before debugging all of the related hardware.
- Start by making a schedule for the project and include any outside events like vacations, graduations, etc. to avoid surprises later on.
- Black objects left in the sun tend to melt any hot glue that is exposed. This is detrimental to the project’s structural integrity. It is therefore wise to a) avoid hotglue or more realistically, b) avoid leaving black hotglued objects in the full sun for extended periods of time.

- Although checkpoints are important, the key is to continue working on the final product, so at all times try to write code/build hardware that you will be able to use in the final product. Try to minimize writing special “check-off code” and building “check-off hardware” that you won’t use later.
- Thinking very carefully about your electrical design/layout will save you lots of soldering time. By designing carefully, you’ll optimize locations of every components, and you’ll end up making a lot less solder joints/connectors/electrical boards, fewer corrections.
- Use Debugging Leds if using PICs. Reserve a few outputs so you can toggle the bits and see if you get into loops or states. This was really helpful when we were trying to figure out what was wrong with our code. Also, since we already used the SSP and Asynchronous communications outputs, we could not use printfs to the terminal.
- Check your #defines and labels. With PIC programming, you tend to have a lot of GOTOs and CALLs which means you need a lot of labels. Try to have a good system for labeling things and creating your constants and variables. We used CAP_UNDERSCORE for # defines and FirstCapitalLetter with no spaces for variables. Where we went wrong was creating “FORWARD” and “FORWARD_CMD” which we misinterpreted and messed us up for a long time.
- Talk to other people about the communication protocols and how they implement their code. It’s hard to figure out the datasheets by yourself with no help from anyone.
- Debug code extensively prior to integration with other software/hardware elements.
- Utilize 7-segment display or LCD display for real-time debugging.

- Modularize mechanical systems so that simpler parts can be made earlier and used from the beginning of development.
- Develop a clear understanding of the communications protocol from the beginning of code development.
- Communicate well within your team so that some tasks are not overlooked, while others are duplicated.
- Learn some of the common problems with writing PIC code (such as dividing your long code into small sections using the “Maincode code” command).
- Bathe as frequently as possible; encourage others to do so as well.