ECE 4534 Fall 2016 Patterson Sections

Project Constraints v1.0

Overall

1. The project must use two rovers performing distinct tasks you propose that are either cooperative or competitive.
2. Both rovers must operate autonomously rather than simply follow motion commands from another source.
3. You are responsible for carefully selecting and evaluating the type, number, diversity, and positioning of sensors that suit your chosen application.

Software

1. PIC32 code must use Harmony with FreeRTOS.
2. All tasks and ISRs must communicate with FreeRTOS-controlled queues rather than global variables.
3. Each task should perform a specific function such as particular computation, communication, or control operation. Hierarchical decomposition should be used so that every C function can be entirely viewed on a single screen with normal font size. This precludes cutting and pasting code.
4. Code development should include the study and selection of appropriate algorithms. For example, control algorithms should explicitly compute an error between the current and desired states (e.g. location, proximity, angle, ...) and apply a proportional or proportional-integral-derivative feedback mechanism.
5. It should be easy to make changes to any system parameter or constant value. This implies that code should assign meaningful symbolic names to all constant values rather than repeatedly hardcode a value.
6. Excluding libraries, the amount of code developed by each team member is expected to be on the order of 500 lines. Strive for concise and elegant code.
7. You are responsible for carefully selecting and evaluating sensor data filtering, conditioning, and fusion algorithms.

Communication

1. The Raspberry Pi serves as the wireless hub for all communication between PIC32 boards, even if the PIC32 boards reside on the same rover. The PIC32 boards use a WiFly shield with a UART physical interface, and the Raspberry Pi uses a WiPi dongle. The Raspberry Pi may provide a user interface through a connection to the Internet or a laptop.
2. All wireless communication must use packets with self-identifying fields (also called key-value pairs) including the sender, receiver, whether this is a request or response, sequence numbers, and data.

3. Rather than ad hoc formats, all wireless communication must use an efficient serialization library such as MessagePack.
4. The Raspberry Pi cannot be used for computation, control or sensing, but can be used as a database server if memory requirements exceed what is available on a PIC32 board.

Individual Responsibilities

1. Each team member is responsible for all of the code residing in a distinct PIC32 board that either resides on a rover or is in a fixed location connected to sensors and/or actuators. Two PIC32 boards can be mounted on a single rover.
2. You must partition the system so that each team member implements distinct functionality on their PIC32 board even if code could be combined in fewer than four PIC32 boards.
3. All team members must demonstrate comprehensive and automated unit tests to verify that their board correctly performs all the required functionality without using other PIC32 boards. The Raspberry Pi and/or a laptop could be used to emulate the other PIC32 boards. Milestone demonstrations cannot stop the system (e.g. by using a debugger breakpoint).
4. Unit tests should display communication packets in a human-readable form.
5. Unit tests should include showing that sensible things happen when a PIC32 encounters:
    a. invalid requests and responses,
    b. dropped, incomplete, or garbled packets.