

The software requirements and overview have been dealt with elsewhere in this document. The present section addresses the design and implementation of the software that forms the X system.

0.1 Software Design Process

The software was designed in a modular top down manner for both the robot and the controller. All code for the robot and controller were designed, tested and implemented independent of each other with the exception of the Serial Communications module which was identical on both sides. Furthermore each module designed to operate as independently as possible from the others for easier integration.

0.1.1 Software Development Environment

All software was created and compiled using MPLAB X IDE v3. Software was then run and debugged using the ICD 3 first on the PIC18F452 development board and then on the PIC18F4520 minimal board.

0.1.2 Software Implementation Stages and Test Plans

Each software was first implemented, debugged and tested on the PIC18F452 development board until it worked.

The modules were then implemented on the PIC18F4520 minimal board and integrated in the following order:

1. **LCD and Menu Nav** on the Controller.
2. **The buttons and Joystick** was integrated with the Menu Nav of the controller.
3. **PWM motors** operation and control on the jousting robot.
4. **Serial Communications** was then integrated to both the controller and jousting robot and tested. Once this was working the Controller was fully integrated.
5. **Manual Mode** was then integrated on to the robot and tested with the controller via Serial Communications.
6. **IR Sensors** was then integrated to the robot along with Full Auto and Assisted mode.
7. After this was all integrated further tests were conducted for operation under Manual, User Assist and Auto modes for the jousting robot using the joystick and button controller inputs.

0.2 Software Quality Assurance

Each module had its own .c file and its own header file. All interactions between different modules were outlined in each header file. A code header block was used for each file indicating the following in order:

- The module this file applies to.
- Date created and by whom.
- Date Last edited by whom and details of edit.
- File Description
- File Dependencies
- Current issues with file

Another method used to ensure the quality of software was to implement all interrupts in the same file Interrupt_Definition_Robot calling functions from the respective modules.

0.3 Software Design Description

0.3.1 Architecture

The following state transition diagrams show how the run modes are changed both for the controller and robot. Once a change is made on the controller the same change is made on the robot. Refer to Serial communications module for more information on how these changes occur.

0.3.2 Software Interface

0.3.3 Software Components

Commander:

- Menu Navigation.
- Menu Navigation submodule LCD.
- Serial Communications.
- Analog to Digital conversion.

Robot:

- PWM motors.
- IR Sensors.
- Serial Communications.
- Analog to Digital Conversion.

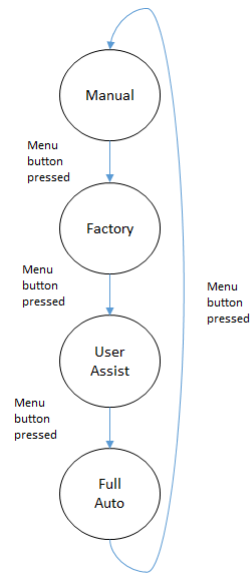


Figure 1: Controller State Transition Diagram

0.4 Preconditions for Software

0.4.1 Preconditions for System Startup

0.4.2 Preconditions for System Shutdown

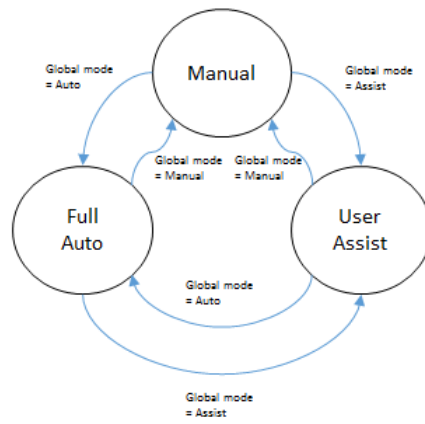


Figure 2: Robot State Transition Diagram