Meera Krishnamoorthy
TA: Fabio Arai

Remote Unit Functional Specification

Overall Description

The Remote Unit consists of the keypad and display through which the user interacts with a three wheeled motor unit. The keypad and display allow the user to control the movements of the RoboTrike (the wheeled motor unit), as well as the behavior of the laser on top of the RoboTrike. The unit that the user interacts with is the keypad. The user inputs what they want the RoboTrike to do by pressing buttons on a keypad. The keypad will then display (briefly) a string outlining what command it is going to execute. The unit then sends a serial command to the motor unit telling the motor unit what command to execute in language that the motor unit will understand. After the motor unit executes this command, it will serially send back a status update which will be more permanently displayed on the display (it contains the updated angle and speed of the RoboTrike). If a serial/parsing error occurs on the motor side, or a serial error occurs on the remote side, this will also be displayed on the display in the form of an error message.

Shared Variables/Structures

The shared variables used in this system and their descriptions are listed below. These variables are all set by the user.

- EventQueue: This structure holds all the events that happen (keypad press event and serial data transfer/ error events). It is dequeued in the main loop of the Remote Unit.
    - If a key press event occurs, it will call a function to send a command to the motor unit/ display a string.
    - If a received data event (from the serial port) occurs, it will display the received data
    - If an error event (from the serial port) occurs, it will display that an error has happened
- Received Data String Buffer: This structure holds a string of received data. Received data is sent in strings, but is received in characters. This buffer holds the characters sent to the remote unit until a carriage return (symbolizing the end of the received data string). Then, the string will be displayed.
- Critical Flag: This flag is set when the Event Queue is full. It indicates that the system needs to be restarted.
- Error Flag: This flag is set when a serial error occurs. It indicates no data should be received on the remote side until the error flag is cleared (this would be after a string is displayed, on the remote side).

Input/ Output Devices

Keyboard

The keypad is programmed to have 16 keys, which are discussed in more detail in the user interface portion of this functional specification, but in general, the user will press buttons on the keypad in order to change the motion of the RoboTrike or to modify behavior of the laser. This change comes about because the keypad sends commands over the serial interface to the motor unit, which responds by changing the motor/ laser behavior.

Display

The display consists 8 digits. We have chosen the display to have 14 segments with right hand decimal point per digit. More details about the display are discussed in more detail in the user interface portion of this functional specification, but in general, the display displays the motion of the RoboTrike, as well as errors that are occurring. The display receives information from the motor input via the serial interface.

Serial Port

A serial interface is a method of communication between two systems in which data is transmitted between the systems as a series of voltage pulses down a wire. A “1” is represented by a high logical voltage, and a “0” is represented by a low logical voltage. The user and motor unit interact through the serial interface using a defined protocol.

The inputs are fed to the three wheeled motor unit via a standard serial port, which uses a 16C450 UART. This serial port connects the display/keypad to the three wheeled motor unit, and sends data one bit at time from the display/keypad to the three wheeled motor unit.

The serial interface is described as being a standard serial interface. There is transmission and reception of data occurring gat the same time in this interface, at the same baud rate. A baud is described as being a unit of transmission speed equal to the number of times a signal changes state per second (one baud is equivalent to one bit per second).

The serial interface not only sends data, but also must ensure that the data is being sent slowly enough so that it can all be acted upon by the Robotrike and displayed appropriately. The slowness of data sending can be controlled by flow control, or “handshaking”, which is a method utilized to prevent a very fast flow of bytes. When this occurs, the bytes can overrun a terminal, computer, modem, or other device. In the case of the Robotrike, the three wheeled motor unit would be overrun. Handshaking is not implemented in the serial interface of the Robotrike.

The flow control used in the Robotrike is considered “hardware flow control”, which uses dedicated signal wires like RTS/CTS. RTS/CTS uses the pins RTS (which stands for ready to send), and CTS (which stands for clear to send) on the serial port. When RTS is asserted (a positive voltage is sent through it) at the receiver, it means: keep sending data. If RTS is negative (the voltage is negative), it means: stop sending data. The RTS signal controls the CTS signal. When the three wheeled motor unit is ready for more input, RTS becomes positive again. Thus, the RTS signal is an input, and the CTS signal is an output.

User Interface

The RoboTrike moves manually via the keypads (using keys like Left/ Right/ Forward/ Reverse) and sends those commands via serial interface to the motor unit. The motor unit sends back status information which is displayed on the display, along with current information about RoboTrike movement.

There are, as mentioned before, 16 buttons on the keypad. The keypad is modeled thusly.

Stop RoboTrike	Set speed of RoboTrike to half of its maximum speed	Set speed of RoboTrike to a random value close to its Maximum speed	Set speed of RoboTrike to its maximum speed
Increase speed by 1000 units	Increase speed by 10000 units	Decrease speed by 1000 units	Decrease speed by 10000 units
Turn RoboTrike to the right	Turn RoboTrike to the left	Keep RoboTrike moving forward	Turn RoboTrike backwards

Turn RoboTrike 45 degrees to the right	Turn RoboTrike 45 degrees to the left	Fire laser	Turn laser off

These buttons, when pressed, do two things. They send a command over the serial port to the motor unit indicating the action they would like perform. There is a certain type of command that the motor side can compute:

- Changes in absolute speed correspond to the character “S” followed immediately by a “+” and a number or just a number. That number indicates the new speed. This number should be followed by a carriage return to signify the end of a command.
- Changes in relative speed correspond to the character “V” followed immediately by a “+” and a number, or just a number, or a “-“ and then a number. That number indicates the absolute value of the change in speed from the current speed to the new speed. A “-“ indicates the number should be subtracted from the current speed to obtain the new speed, while a “+” or no sign indicates the amount should be added to the current speed to obtain the new speed. This number should be followed by a carriage return to signify the end of a command.
- Changes in direction correspond to the character “D” followed immediately by a “+” and a number, or just a number, or a “-“ and then a number. That number indicates the absolute value of the change in direction from the current direction to the new direction. A “-“ indicates the number should be subtracted from the current direction to obtain the new direction, while a “+” or no sign indicates the amount should be added to the current direction to obtain the new direction. This number should be followed by a carriage return
- Firing the laser corresponds to just the character “F” followed by a carriage return.
- Turning off the laser corresponds to just the character “O” followed by a carriage return.

(Note: Spaces in between the numbers/carriage return are acceptable)

There is a table that links key presses with the appropriate command strings. When the appropriate key is pressed, a command is sent over the serial port to the motor side for the motors to perform the command. The motor unit reacts to this information by moving the motor or activating the laser.

The pressed buttons also result in strings displayed on the keys that indicate the action that was just performed. There is a table that links key presses with the appropriate display string, so those strings are displayed after a command is sent serially to the remote side. These strings, however, do not show up on the display because they are over written by the motor unit status updates. The motor unit sends status updates to the display. The status updates that the display shows are the current direction and speed of the RoboTrike, and are updated each time a serial command is executed.

The display also shows any errors that are occurring on the serial interface (from the RoboTrike to the remote unit and vice versa), like framing, parity, break, overrun, buffer overflow, etc; as well as parsing errors that the motor unit might encounter. This is discussed more in detail in the error handling section of this functional specification.

Algorithms

One algorithm used on the remote side is multiplexing the LEDs. To make it appear that all 8 digits are on, at the same time, I turn on each digit for a short amount of time and display each digit very rapidly. This requires a timer interrupt (which I implement using Timer 0). In the event handler for display, I call the routine that does the multiplexing routine (keeps track of the digit just outputted and outputs the next digit), so each digit is displayed on a timer interrupt. This allows it to seem as though all digits are on at the same time when they are technically not.

Error Handling

We handle serial errors that could occur when receiving and sending to the remote side using a function that reads in the Line Status Register of the 16C450, a register that contains the type of serial error that occurred. This function then looks up in a table an error message that corresponds to the type of serial error that occurred, and displays that this error has happened.

Limitations

One limitation to this approach is that the string displayed to represent the status update of the motor displayed very quickly after the key press string (that indicates what command was sent to the motor) is displayed. AS a result, the key press string is not displayed for every long, and is therefore not seen.