Meera Krishnamoorthy
Section: Wednesday, 5 PM
TA: Fabio Arai

Motor Unit Functional Specification

Overall Description

The motor unit involves the RoboTrike and the serial interface and their communication to the user. The RoboTrike is a three wheeled robot with holonomic motion that is controlled by user input via a keypad. When a key is pressed, in the remote unit, a string is sent to the motor side with a command outlining a change that the user wants to make in either motor speed, direction, or laser status. This unit makes that change through a parser that employs a state machine. Once the parsing is complete, the RoboTrike communicates with the user through the serial interface by sending the user status updates if the parsing of the command was successful, or error messages if the parsing was unsuccessful.

Shared Variables

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 (serial data transfer/ error events). It is dequeued in the main loop of the Motor Unit.
    - If a received data event (from the serial port) occurs, it will parse the received data, and execute a motor command if the parsing is successful. Then it will send a status message over the serial port to the remote side to display. If the parsing is not successful, an error message will be sent to the serial port for the remote side to display.
    - If an error event (from the serial port) occurs, the main loop will send an error message to the remote side to display.
- 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

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.

Motors and Laser

The motors and laser are controlled by an 8255 parallel port, The 8255 is a chip that was developed and manufactured by Intel. The motors are connected to ports on the 8255, which latch the data written to them and hold that data on the output lines.

A description of all motors involved in the Robotrike is below:

- three DC motors
    - The DC motors are used to rotate the wheels of the RoboTrike in either direction. The goal is for the wheels to be able to rotate in all directions, so that the RoboTrike can move at every angle.
    - These motors are connected to port B of an 8225.
    - Each motor may be run clockwise or counterclockwise, as determined by one bit of Port B for each motor.
    - The DC motors work to move the RoboTrike using PWM, or pulse width modification, is a method of getting analog results from digital means.
- the laser
    - The laser is on the turret, and can be "fired".
    - It is controlled via a single bit of parallel output of an 8255, and is connected to port B. The laser is powered when port B is activated.

The motor unit consists of the laser and motors together. The motor unit of the Robotrike is controlled by the serial interface, and should output its current status (its speed, angle, turret position, etc.) to the serial interface whenever that status changes. The motor unit receives inputs from a keypad, and outputs its status to the display.

User Interface

The RoboTrike has no user interface. All user interface is in the remote unit.

Algorithms

The main algorithm implemented on the motor side is the algorithm for calculating the motor motion. After the motor speed and direction are set, the total power given to each motor (based on the fact that the first motor is only powered in the horizontal direction, and the second and third motors are powered in both the horizontal and vertical directions, but more in the vertical direction). Then, this power value is compared to a counter which starts at 0 and is incremented each time the event handler is called. If the motor's pulse width (power) is greater than that of the counter, the motor is on. If it's not, the motor is off. This function also determines the direction of motion – if the pulse width is negative, the motor is travelling backwards, and if it is positive, the pulse width is moving forwards. An

overall signal is sent to the motors telling each motor whether its on or off, and if its on, what direction its travelling in.

Another algorithm used on the motor unit side is implemented in the parser. The parser reads in a character sent to it. It starts of in its initial state (INIT). There is a proper format for sending commands to the motor side (outlined in the remote unit functional specification). If this proper format is followed, the parser will enter subsequent states until it enters the state in which it executes a motor command (changing the speed, direction or laser status of the RoboTrike).

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 sends that message over the serial port to the remote side to display.

We handle parsing errors by sending an error message indicating that a parsing error has occurred to the remote side to display.

Limitations

One of the limitations known to us is the fact that a regular robotic car would generally include some feedback so that it could determine its position. There is no such feedback for the RoboTrike, so it has to use a holonomic motion approach to its movement. The way that the RoboTrike uses this movement is outlined in the “Algorithms” section of this functional specification.