Realtime Programming Lab 4

Erik Alm, Mario Rios

May 4, 2016

1 Controlling Motors

All the motor controlling logic was implemented in the driver.c and driver.h files. We define a structure called Motor, that models the hardware electric motor at a low level of abstraction (it stores the speed pin and the direction pins). Moreover, we define an extra structure called Engine, which models at a higher level of abstraction what would be the whole engine in a real car. This structure stores both motors (hardware motors), the gear of the whole system, and the direction the whole system is going to. The directions is specified as an enum, and has four different values (forward, backward, left and right).

2 Implement the Car

Figure 1 describes at a high level of abstraction the threads created by the program. The main communication method between the input and the rest of the threads is done by means of the mqd_t message queue. CMDThread captures the input from the buttons, sends messages to the queue, and threadDispatcher polls the queue, receiving the messages. Depending on the type of message, it will change the values of the parameters of the engine. This global engine variable, will then be accessed by the other threads, (being protected with mutexes) and will perform they're duty according to the state of the engine. In order to guarantee that the kill switch has highest priority, the message is sent to the queue with higher priority than the rest.

Motor One and Motor Two threads, run the same function, but for different hardware motors, sending pwm pulses, with different directions and speed settings depending on the global state of the engine. LedSteering thread, takes care of the led illumination for the steering buttons. Both leds are controlled by the same thread, since the functioning of them is exclusive (when right led is on, left is off, and viceversa). The same applies for the ledsGo&Stop (it controlls the green and red led).

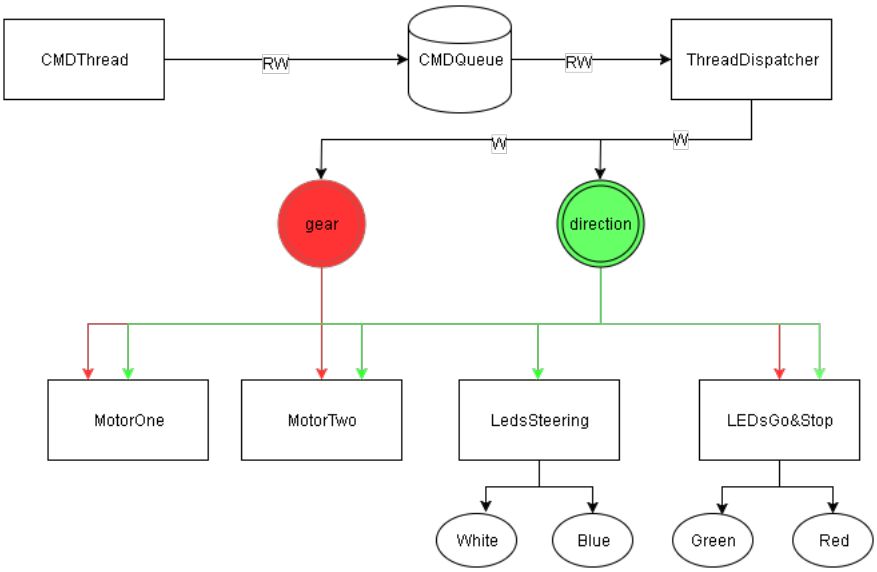

Figure 1: Threads and variables accessed by them in the system