Programming Embedded Systems 2018 – Exercise 3

Alexander Tokos, 38636

2.10.2018

Task

The task for this exercise was to implement a traffic light system using a multi-state machine. The transition between states should be based on either input (switch) or timed. For this exercise the main loop should be empty and the low power mode used.

Equipment used

- Texas Instruments LaunchPad MSP430g2553 microcontroller
- Laptop

Preformed work

Starting off I realized that almost all functions in the led.c file from the last exercise were not suitable for this exercise so I deleted them all except for the init function. I then created a function for selecting a led and turning it on or off depending on the arguments given.

Next I defined a enum containing all the possible states, which were the following:

- The initial state when the program is started (Both leds on)
- Red (Only red led on)
- Transition from red to green (Both leds on)
- Green (Only green led on)
- Transition from green to red (Both leds on)

Next I created a traffic.c and corresponding header file. Using the hints from the lab introduction I created 3 global variables in the traffic file: one variable to keep track of the state, one to keep track of the time spent in the current state and one to keep track of when the button was pressed (in which case the state should transition to green). The duration of each state was also defined in this file.

Next I created a traffic update function and replaced the Led_Update with it in the ISR, now each time the timer creates an interrupt the Traffic_Update() function is called. In the update function the majority of the new code was written. Every time the update function is called the time in duration variable counts up one step. After that I made a switch statement to determine which state the lights are in currently, each state has an own case. In each case ther is an if-statement checking if the time in duration has been fulfilled, if it has the state is changed to the next one and the time variable reset. Now the states transitions work correctly.

Next task was to make the switch work. I used the existing code from the previous exercise for the switch but moved the if-statement for checking button presses from the main loop to the update function. When a button is pressed the gotoGreen variable is set to TRUE. Next the Red case in the switch statement was modified so that if gotoGreen is true, the timer variable is reset, gotoGreen is set to false and the state changes to red-green which leads to green.

Lastly I downloaded the uart files and included them in traffic.c and main.c. The functions worked out of the box, which was nice. The real problem was finding the device on my computer, after I found it I used putty to read from that serial port and confirmed that the uart was working. At this stage I realized that I had forgot to put the low power mode in the main loop.

Results

The traffic light system now works as intended and the uart sends data to the serial port, except for one small problem. When the button is pressed, the state changes to green via the red-green transition state, after the green duration has been fulfilled it goes to the green red transition. Sometimes the state changes to red-green right after green-red, entirely skipping the red state. My best guess is that if the button is not pressed fast enough the gotoGreen variable is set to TRUE again after the state has changed from red, thus skipping the red state the next time it reaches it.