Shyamal Anadkat, Sneha Patri, James Papa ECE 315
4/5/17

Lab 2: Motor Control Using PWM

Lab Goals

- The goals of this lab were to configure GPIO pins as PWM peripherals, to configure a PWM peripheral, and to develop a driver that enables software to control the direction and speed of the robot.
- Ultimately the goal was to program the robot to move forward, backward, and at different speeds.
- Lastly, demonstrate that the robot moves forward 2 seconds, backwards 2 seconds, turn left for 5 seconds and then turn right for 5 seconds.

Steps taken to accomplish goals

GPIO Configuration

- First, we configured the GPIO pins as PWM peripherals.
- Specifically, the PB4, PB5, PE4, and PE5 GPIO pins were configured as PWM peripherals in the alternate function register. The port control register was also configured for each pin.
- Lastly, PF2 was configured as a digital input, and PF3 was configured as a digital output.

PWM Configuration

- Next, we created the pwmConfig() function which configures the PWM peripheral.
- Firstly, we initialized the RCGCPWM register
- Then set the PR mask to base and wait till the PRPWM is polled.
- Then configure the RCC to clear bits 19-17 and set the 20th bit
- Then have switch statements for the different generators and set the enable accordingly

Driver Development

- Lastly, we developed the functions that control the motors. Specifically we implemented the functions leftForward, rightForward, leftReverse, rightReverse, turnLeft, and turnRight.
- All six of these functions call pwmConfig() with the different configurations and PWM signals.
- For the turnleft and turnright functions we called right forward and left reverse and left forward and right reverse respectively.

To demo the project, we needed to have the robot drive in specific directions for a given amount of time. This was accomplished with our interrupts we created last lab.

Problems Faced

- The main difficulties we had were with configuring the GPIO pins and PWM peripheral. It was difficult to verify if we were using the right macros.
- The steps to complete the GPIO configuration are detailed in the ready reference below. Since the data sheet's PWM configuration instructions gave a general setup, it was difficult to identify exactly what to do.
- Lastly, it was difficult for us to figure out how each of the motor functions (leftForward, rightForward, etc.) should call pwmConfig.
- Getting the duty cycle right was crucial for calling the pwmConfig functions in drv8833.t

Ready Reference

GPIO Config:

The six pins needed to control the DRV883 are: PB4, PB5, PE4, PE5, PF2, PF3. 1) Enable gpio ports B, E, F.

Ex: gpio_enable_port(GPIOB_BASE);

2) Configure PB4 & PB5 as a digital enable

Ex: gpio_config_digital_enable(GPIOB_BASE, PB4 | PB5);

3) Configure PE4 & PE5 as a digital enable
4) Configure PB4 & PB5 as an output

Ex: gpio_config_enable_output(GPIOB_BASE, PB4 | PB5);

5) Configure PE4 & PE5 as an output
6) Configure PB4 & PB5 as an alternate function register

Ex: gpio_config_alternate_function(GPIOB_BASE, PB4 | PB5);

7) Configure PE4 & PE5 as an alternate function register
8) Configure the port control register for each pin given the mapping

Ex: gpio_config_port_control(GPIOB_BASE, GPIO_PCTL_PB4_M0PWM2 | GPIO_PCTL_PB5_M0PWM3);

gpio_config_port_control(GPIOE_BASE, GPIO_PCTL_PE4_M1PWM2 | GPIO_PCTL_PE5_M1PWM3);

9) Configure PF2 as a digital input, PF3 and a digital output

Ex: gpio_config_digital_enable(GPIOF_BASE, PF2 | PF3);

gpio_config_enable_input(GPIOF_BASE, PF2);

gpio_config_enable_output(GPIOF_BASE, PF3);

10) Set PF3 to one and use the following command Ex:

GPIOF->DATA |= PF3;

SYSCTL->RCGCPWM |= SYSCTL_RCGCPWM_R0 | SYSCTL_RCGCPWM_R1;