Andrew Covarrubias (axc554)
Jonathan Monreal (jem177)

Design Questions

Question 1:
What is the control signal frequency specification of the motor driver used on the daughter board?

Based on the motor driver manual, the typical signal frequency is about 30 kHz with a max of 100 kHz.

Question 2:
What is the chosen output frequency of your PWM generator?

We decided to choose a frequency of 24.5 kHz for our PWM generator.

Question 3:
How frequently does your design determine the motor's rate of rotation (that is, the frequency that the encoder change counter is saved and reset)?

Since we decided to use a 21-bit clock for the reset, the rate of rotation can be determined to be about 23.8419 Hz.

Question 4:
Assuming a theoretical maximum motor speed of 6000 rpm, what is the maximum motor speed your design will calculate with the provided 192 count/turn encoder attached to the motor?

If the motor has max speed of 6000 rpm with an encoder reading 192 times per turn, we get about 19200 encoder ticks per second. The maximum speed we could calculate would be based on the 8 bit counter and give a max speed of 75 kHz.

Question 5:
Calculate approximately how long it takes to change the speed goal from 0 to full speed in one direction if the button is held constantly.

Using the time each speed is taken we can calculate how long it would take to go full speed. It takes about .0419 seconds for each change in speed and in order to go full speed it would take about 10.7374 seconds of holding down the button.

Question 6:
At maximum gain, what is the measured pulse width of your motor controller implementation when the goal is minimum and maximum for the free-running, enabled motor?
At max gain the pulse width seems to be about 20 microseconds and at minimum it seems to be about 5 microseconds.

Response Requirments

Requirement 1:
Briefly discuss the use of each counter used in your implementation and identify which module it is in.

All of the counters are in their own modules.

button_clock_generator is used to determine the rate at which holding the button will speed up the motor

pwm_clock_generator is used to determine the waveform and how it is affected by the varying speed of the motor

srst_clock_generator is used to determine when the register is filled so that copied data can be reset

up_down_counter determines if the direction of the motor based on if it is 1 or -1

Requirement 2:
Describe the functionality of each module used in the design. Your signal names should make sense relative to their function and correspond to the signal names used in the block diagram.

adder - used to determine the difference between goal speed and measured speed

multiplier - used to determine the product of the difference between the goal speed and measured speed, along with the gain constant

button_clock_generator - generates the clock for the goal speed buttons

duty_cycle_calculator - implements the adder and multiplier module in order to determine the duty cycle

flipflops - used to create a positive edge detector for the motor decoder

goal - creates a goal speed from the switch inputs

motor_decoder - uses the 192 count motor encoder to take in the speed from the motor

pwm_clock_generator - generates a 24.5 kHz clock for the PWM

pwm_gen - generates a pwm based on 8-bit two's compliment duty cycle

srst_clock_generator - generates a clock for the reset and enable function

up_down_counter - implements a counter used in determining the direction of the motor and implements a reset

velocity_register - stores the velocity of the mode, features an enable

Requirement 3:
Summarize how each group member contributed to the completion of this lab.

Jon - main programmer for this lab

Andrew - wrote the report and assisted Jon in debugging and writing the code