Lab 5 – Prelab

1. The overall purpose of this week's lab is to introduce us to PWM and QEP peripheral modules, and to aid our understanding of its use, through the application of a motor. It is also to learn the principles of operation of the voltage actuator and position sensor used in DC motor control systems. We achieve these tasks by understanding how to convert the oscillator clock frequencies to CPU clock frequencies to PWM clock frequencies. We also learn how to generate and sync desired output waveforms over multiple GPIO pins, and to read as well as write to devices that are operating through PWM.

2.

```
// STEP 1: add DRV8305 Booster Pack to F28379D Launch Pad

//initializing  the PWM registers in setup*************
// STEP 2: set which PWM pins are going to be used:

GpioCtrlRegs.GPAGMUX1.bit.GPIO0 = 0;
GpioCtrlRegs.GPAGMUX1.bit.GPIO1 = 0;

//STEP 10: Set multiplexer pins for QEP
GpioCtrlRegs.GPAGMUX2.bit.GPIO20 = 0;
GpioCtrlRegs.GPAGMUX2.bit.GPIO21 = 0;
GpioCtrlRegs.GPAGMUX2.bit.GPIO23 = 0;

//STEP 3: setting scale factor for PWM clock

ClkCfgRegs.PERCLKDIVSEL.bit.EPWMCLKDIV = 1;    // fclk.pwm = 100MHz

//STEP 4: writing to PWM registers – delay is important

CpuSysRegs.PCLKCR2.bit.EPWM1 = 1;    // module clk enable
asm(" NOP"); asm(" NOP");            // wait

//STEP 11: Setting QEP Clock
CpuSysRegs.PCLKCR4.bit.EQEP1 = 1;
asm(" NOP"); asm(" NOP");

//STEP 5: Set waveform to triangle wave

EPwm1Regs.TBCTL.bit.CTRMODE = 0b10;

//STEP 6: Assign Range and frequency

EPwm1Regs.TBPRD = PWM_MAX_COUNT;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = 0;
```

```
EPwm1Regs.TBCTL.bit.CLKDIV = 0;

//STEP 12: Setting the Max QEP ctr value
EQep1Regs.QPOSMAX = 0xFFFFFFFF;

// Set the output actions************
//STEP 7: Configure green pin logic signals

EPwm1Regs.AQCTLA.bit.CAU = 0b10;
EPwm1Regs.AQCTLA.bit.CAD = 0b01;

//Enable the time base clock********
//STEP 8: Sync time-base clock signals of each active PWM module

CpuSysRegs.PCLKCR0.bit.TBCLKSYNC = 1;

//STEP 13: Enable QEP Ctr subsystem and initialize values

EQep1Regs.QPOSINIT = 0;
EQep1Regs.QEPCTL.bit.QPEN = 1;
EQep1Regs.QEPCTL.bit.SWI = 1;


//STEP 9: MAP AND WRITE DUTY CYCLE*****
EPwm1Regs.CMPA.bit.CMPA = 167;      // D=0.9173 with maxval=2000

interrupt void TimerISR(void)
{
    // Fast – write a motor voltage value to PWM module


   // Fast – Read a motor position value from QEP module

    //STEP 14: QEP ctr value

   QEPctr = EQep1Regs.QPOSCNT; // cast this value into a int32 from UINT32

   // Slow – Compute and store the next motor voltage value
   // Fast – Acknowledge the interrupt
}
```

3. JP1, JP2, JP3 need to be disabled for power isolation, and JP6 needs to be enabled and connected to 5V.

4. EN_GATE is a GPIO signal on the driver chip that enables gate driver and current shunt amplifier.