CENG 412 Embedded Systems
Lab 10 Pulse Width Modulation Module of the HC9S12

Name _____

Signature Part A _____ **Part B** _____ **Part C** _____

Introduction

In this lab you will use the Pulse Width Modulation function of the HCS12 to generate square wave at various frequencies and duty cycles.

Procedure A - Assembler Code Implementation

Create an Absolute Assembly project in CodeWarrior to generate a square wave at 40 khz and a duty cycle of 40% using the Pulse Width Modulation function of the HCS12. Use the Lab 10 Work Sheet to configure the program code.

> **Note: To simplify the programming with Assembler projects use a new directive (include mc9s12dp256.inc) to define all the register addresses for the assembler. The directive is placed at the start of the code.**

Requirements:

- Select Clock A, select a pre scale value of 8 for Clock A
- Select PWM channel 4, High Polarity and left aligned output

Identify the pin on the Demo board header for PWM ch 4. Pin No = _____

Download and run the program. The square wave should be present on the correct pin on the Demo board header. Verify using the scope that you have the correct waveform.
By changing the Polarity bit edit the code to get a 40 kHz square wave with a 60% duty cycle.

Procedure B - C Code Implementation

Create a C project in CodeWarrior to generate a square wave at 2 kHz and a duty cycle of 30 % using the Pulse Width Modulation function of the HCS12. Use the Lab 10 Work Sheet to configure the program code.

Requirements:

- Select Clock SB
- Select a pre scale value of 8 for Clock B and a pre scale value of 10 for Clock SB
- Select PWM channel 2
- Select a High Polarity and left aligned output
-

Identify the pin on the Demo board header for PWM ch 2. Pin No = _____

Make the project and download the .s19 file in AsmIDE. Execute the program. The

square wave should be present on the correct pin on the Demo board header.
Verify using the scope that you have the correct waveform.
Demonstrate your programs and have your lab sheet signed.

Part C – Demonstration Code

The code in this program integrates several of the built-in functions of the HC9s12 microcontroller. Create a CodeWarrior project. Copy and paste the code into main.asm. Assemble the code. Connect a DC motor to either one of the Motor Driver ports. Connect an oscilloscope to monitor either PWM channel 3 (Motor Driver 2) or 4 (Motor Driver 1) – you will need to determine the pin numbers for these channels. Open AsmIDE, download the code and execute the program. Adjust the Demo Board potentiometer and observe what happens on the LEDs and the motor.

Examine the code carefully and answer the following questions.

What HC9S12 built-in functions are being used? List them all.

_____

_____

_____

Is the A/D converter in Single Scan or Continuous Scan mode?

Is the A/D converter reading a Single Channel or Multiple Channels?

What is the calculated frequency of ClockA? _____

What is the calculated frequency of the PWM channel 3/4 signal? _____

What is the measured frequency of the PWM channel 3/4 signal? _____

Why are both Clocks A and B being used?

In what subroutine in the program is the Duty Cycle of the output square wave being set?

The Motor Driver outputs are referred to as Driver1 and Driver2 in the HC9S12 documentation. What are the Demo Board pin numbers for the input signal to the Motor driver circuits?

Driver1 pin no _____ Driver2 pin no _____

What is the purpose of the Motor Driver circuit?

Locate and draw the complete circuit diagram for Motor Driver1.

```
; CENG 412
;
; PWM/Demo Board Code Demonstration
; The analog signal from the Demo Board pot is converted to
; digital by the HCS12 A/D converter.The A/D output value
; sets the duty cycle of the PWM module to control the speed
; of a DC motor connected to both of the Demo Board motor driver ports

  include mc9s12dp256.inc

;ATD Variables
admask2:  equ %10000000 ;AFFC,ADPU=1 - Enable Analog to Digital
admask3:  equ %01000000 ;FRZ1,FRZ0=0  8 conversions
admask4:  equ %10000101 ;SMP1,SMP0 = Select 8 bit conversion pre-scale=5
admask5:  equ %10110000 ;S8CM = 1, SCAN = 1, MULT = 1
SCFflag:  equ %10000000 ;SCF - Sequence Complete flag

           ORG $2000
start:      jmp Entry

advalue: ds 2

Entry:
        jsr  ATDInit
        jsr  LEDsInit
        jsr  PotPWMInit
again:  jsr  ATD
        jsr  PotPWM
        bra  again
        swi             ;code never reaches here

;*****************************************************************
;Initialize Analog To Digital
ATDInit:
        movb #$F0,ATD0DIEN      ;Make bits 0-3 analog, 4-7 digital i/p(push buttons)
        movb #admask2,ATD0CTL2  ;enable ATD
        jsr  wait20us           ;allow ATD to stabilize
        movb #admask3,ATD0CTL3 ;8 conversions,FIFO mode 0
        movb #admask4,ATD0CTL4 ;8-bit convert,select Sample rate
        movb #admask5,ATD0CTL5 ;right just,unsigned,continuous, multi channel,start 0
        rts
;*****************************************************************
```

```
; Read AtoD converter and store values - does 8 conversions but only some are used
ATD:
        brclr ATD0STAT0,SCFflag,* ;Loop here until SCF of ATD is set ;ATD0STAT0
        ldd ATD0DR2H              ;potentiometer input
        std advalue
        rts

;*****************************************************************
;
PotPWMInit:
        movb #%00011000, PWMPOL   ;High during duty cycle
        movb #%00000000, PWMCLK   ;Clock A & Clock B
        movb #%01110111, PWMPRCLK ;Clock A = Bus Clock / 128, Clock B = Bus Clock / 128
        movb #%00000000, PWMCAE   ;All channels operate in Left Aligned Output Mode
        movb #%00001100, PWMCTL   ;No concatenation of channels
        movb #%11111111, PWMPER3
        movb #%11111111, PWMPER4
        movb #%00011000, PWME      ;Enable Port P channels 3 and 4 as PWM
        rts

;*****************************************************************
;
PotPWM:
        ldd  advalue
        stab PTH                  ;Turn LEDs on
        stab PWMDTY3
        stab PWMDTY4
        rts

;*****************************************************************
;
PotPWMDisabled:
        movb #%00000000,PWME      ;All channels disabled
        clr PTH                   ;Turn LEDs off
        rts

;*****************************************************************
;
LEDsInit:
        movb #$FF,DDRH     ;Bar LEDS, Make Port H = o/p
        rts

;*****************************************************************
;
; 20us delay - uses Output Compare Timer function

C0F:       EQU %00000001

wait20us:
        movb #$90,TSCR1  ;enable TCNT and fast time flag clear
        movb #0,TSCR2    ;set TCNT pre-scale to 1
        bset TIOS,$01    ;enable OC0
        ldd  TCNT        ;start OC0
        addd #480        ;480 cycles at 24MHz =  20us
        std  TC0
        brclr TFLG1,C0F,* ;wait
        rts

        end
```

Lab Exercise
Written by David Lloyd
Computer Engineering Program
Humber College