CMPE 443 PRINCIPLES OF EMBEDDED SYSTEMS DESIGN

PRELAB #6

"ADC and Timer Capture"

Motivation

Timers have a wide range of applications in embedded systems as counting time, capturing inputs, generating signals. In LPC4088, there are 4 Timer peripherals which are Timer0, Timer1, Timer2 and Timer3. You will use one of these timers for capturing the length of the signal. Also, we will use Analog-to-digital Conversion (ADC) So, in this experiment you will learn:

- Using the timer counter and capture register.
- the process of enabling interrupts
- configuring the ADC port of a microcontroller
- reading data from an ADC pin

1) Problem Description

In this prelab, you will use the 3 LEDs as outputs. Also you will get input data from ADC and Timer peripherals. LEDs will be the same as the Quickstart Board LEDs. On and Off states of the LEDs will change according to ADC data value and the RPM (Revolutions per Minute[1]) of the motor. Firstly, the start state of the LEDs is the Off state and when the ADC data is more then or equal to **0x100**, you will turn on the LED1 and other LEDs' on/off condition will be based on the RPM of the motor, when the ADC data is less then **0x100**, you will turn off all the LEDs.

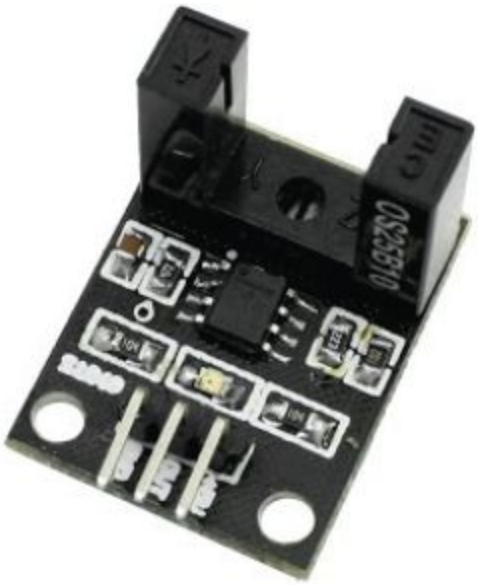
When LED1 is turned on:

- If the motor RPM is more than 50, turn on LED2 and LED3.
- If the motor RPM is between 25 RPM and 50 RPM, turn on LED2 and turn off LED3.
- If the motor RPM is less than 25 RPM, turn off LED2 and LED3.

[1] "Revolutions per minute - Wikipedia." https://en.wikipedia.org/wiki/Revolutions_per_minute. Accessed 15 Jan. 2021.

2) Speed Sensor

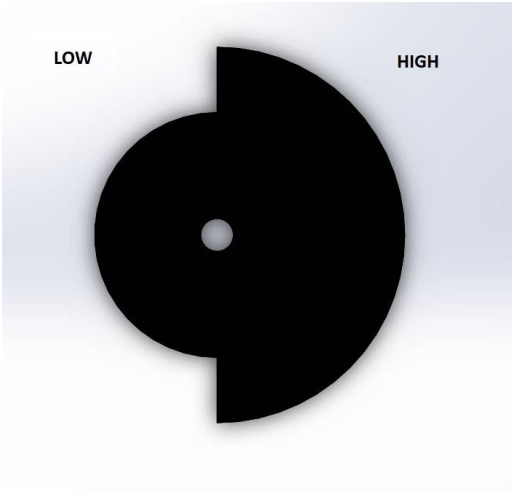
In order to detect the speed of the motor, you will use the Speed Sensor.



You can detect the speed of the motor by observing the frequency of HIGH and LOW values. If the speed of the motor increases, the sensor gives HIGH - LOW values more frequently. So,

- If the speed of the motor increases, the duration of the HIGH and LOW signal values will decrease.
- If the speed of the motor decreases, the duration of the HIGH and LOW signal values will increase.

Your motor is a very basic motor. So you can assume that when you detect HIGH - LOW and HIGH signal, your motor completed its rotation.



3) Input Provider

We will not upgrade the LPC4088 Visualiser, so you need to give input from different software. QEMU now works with TCP protocol, so you can use any programming language. We are giving you this Jupyterlab project as an example but you can use any tool you want:

https://drive.google.com/file/d/1K10Kaq5SnVhIhGdtbMVVdVcm3k6CexEz/view?usp=sharing

When you run the QEMU, QEMU opens TCP socket at: 127.0.0.1:6942. The communication message structure is written on the given code. The important things:

- You can change the ADC, when you use *QEMU_ADC = ord('0');*
- You can change the channel of ADC, *QEMU_ADC_CHANNEL = 2;* variable (ex *QEMU_ADC_CHANNEL* =1; for channel 1)
- You can set the value ADC data, when you use *QEMU_ADC_DATA = 0x100;*

- You can change the Timer, when you use *QEMU_TIMER = ord('3');*
- You can change the TIMER CAPTURE register, by using *QEMU_TIMER_CAPTURE_PIN = 1;* 0 for CR0 and 1 is for CR1.
- You can change motor RPM via *motorRPM = 30;*

4) ADC

You will use the **P0.25** pin for the ADC.

- Change the function value of pin to ADC. (Answer in Moodle) _____
- Change the mode value of pin to mode which should be selected if Analog mode is used. (Answer in Moodle) _____
- Change Analog/Digital mode of pin to Analog. (Answer in Moodle)_____

A/D Converter will work slowly in this experiment and A/D Converter Clock should be 1MHz. (PCLK = 60 MHz)

- Calculate the CLKDIV of ADC for 1 MHz. (Answer in Moodle) _____ *0,5 pts*

In order to use ADC:

- Turn on ADC via PCONP
- You should configure the CR register (Select corresponding AD pins to be sampled and converted. Also, set the CLKDIV, make the A/D converter operational and start conversion) (There is a bug so you should give CR value with = sign) (For example: ADC->CR = (1 << Channel) | (ADC_CLKDIV << 8) | (Operation << 21) | (Start << 24);)
- Enable interrupt for ADC Channel so completion of a conversion on ADC channel will generate an interrupt.
- Enable ADC_IRQn (Interrupt Request).

You can test your code via Input Provider. When an interrupt occurs, IRQHandler of the ADC will be called.

There is no interrupt clear flag for the ADC. So when you read the ADC data from the register, it will automatically clear the interrupt.

5) Timer Capture Interrupt

You will use the **P0.24** pin for Timer Capture.

You can use the your own code from Self Study #7 and #8. However in order to use Timer Capture Interrupt, you should add new configurations and you should delete the match related configurations.

- Turn on Timer via PCONP
- Change the mode of Timer to Timer Mode.
- Disable Timer Counter and Prescale Counter for Timer.
- Reset Timer Counter and Prescale Counter for Timer.
- Change PR Register value for 1 microsecond incrementing
- Configure CCR register for capturing rising edge and falling edge.
- Also configure CCR register for interrupting on capture events.
- Remove the reset on counters of Timer.
- Enable Timer Counter and Prescale Counter for counting.

Also, you should enable TIMER_IRQn from NVIC_EnableIRQ.

When an interrupt occurs, you should clear the interrupt flag. So in the handler method, you should clear the interrupt flag.

6) Changing LED via Sensors

In the update method, change the on or off state of LEDs as described in the problem definition. You will use interrupt in this experiment.