

CG2271 Real Time Operating Systems

Lab 4 Timer and PWM Output

1. Introduction

In this lab we will implement a system that reads input from two analog ports, and control the brightness of an LED through one of the PWM outputs. We will use a photoresistor to control a buzzer, and we will add in a potentiometer to control the brightness of an LED.

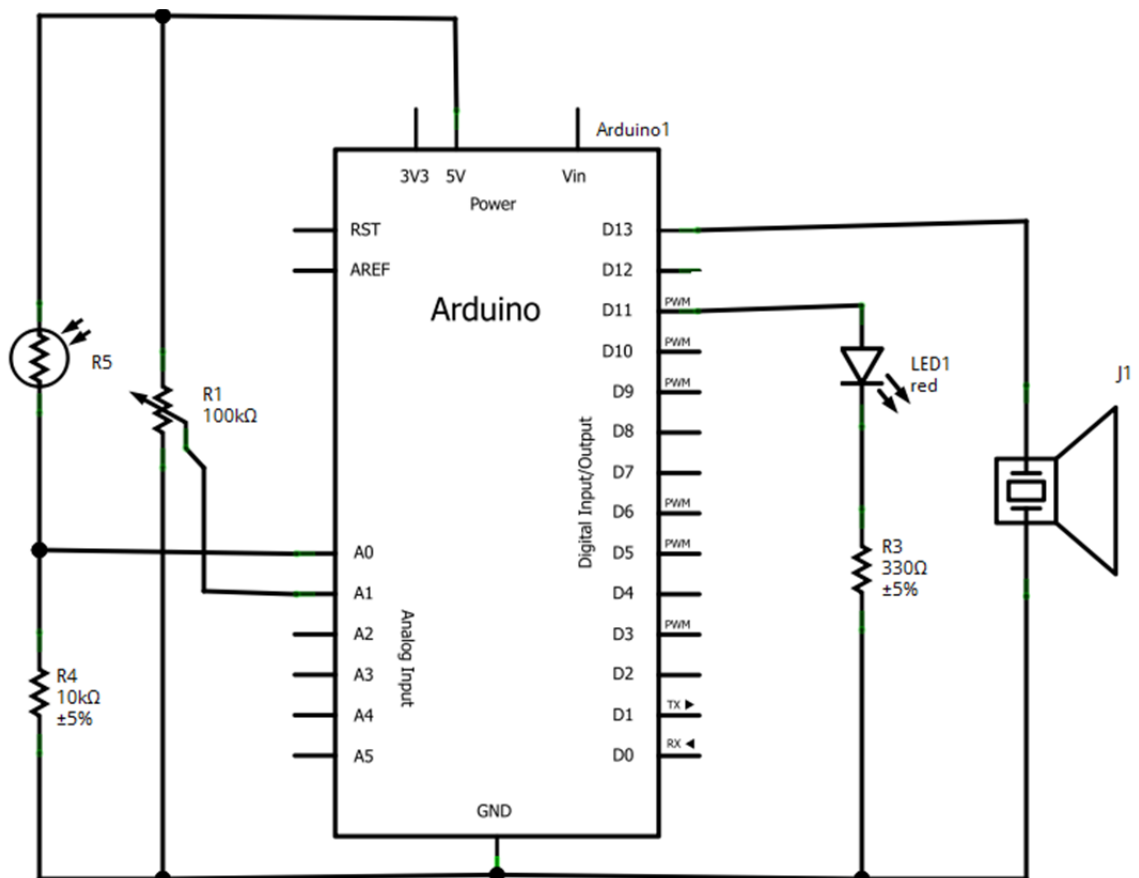
2. Submission Instructions

You will submit your answer book and demo your assignment either at the end of the lab session in the week of 26 September 2011, or the following week on 3 October.



Please note that you must have your answer book duly filled out and printed before you can do the demo.

3. Questions

Assemble the circuit shown below. It is essentially the same circuit as in Lab 3, but with a potentiometer and LED connected to analog input A1 and digital pin 11 respectively.



Parts List:

Parts Label	Description	Quantity
R1	Potentiometer 	1
R4	10K resistor (brown, black, orange)	2
R3	330 ohm resistor (orange, orange, brown)	1
R5	photoresistor 	1
J1	Piezoelectric buzzer	1
-	Arduino Uno	1
-	Breadboard	1
	Wires	As needed

Create a project called lab4part1 and answer the following questions. You should implement your program in lab4part1.c.

Question 1

The LED is connected to Arduino digital pin 11. Which timer output compare pin (e.g. OC0A) does this correspond to?

Question 2

We want to set up pin 11 to supply a 125 Hz phase-correct PWM signal. What is a suitable prescaler value? Fill in the register bit values for the relevant registers as shown in your answer book. Example answers for OC0A are shown below (If your answer to Q1 was OC0A, then x=0 and y=A. If it was OC2B, then x=2, y=B, etc. The “?” in the table entry means that the original bit value should stay untouched.)

Prescaler=2

x=0 y=A

TCCRxA

PinNum	7	6	5	4	3	2	1	0
PinName	COM0A1	COM0A0	COM0B1	COM0B0	-	-	WGM01	WGM00
Value	1	0	?	?	0	0	0	1

TCCRxB (Initial value)

PinNum	7	6	5	4	3	2	1	0
PinName	FOC0A	FOC0B	-	-	WGM02	CS02	CS01	CS00
Value	?	?	0	0	0	0	0	0

Note that the prescaler in TCCR0B must be initially 0. Otherwise the PWM will start converting immediately.

Question 3

Based on your answer to question 2, create a function called “setupPWM” with the following prototype:

```
void setupPWM();
```

Cut and paste your code into the answer book.

Question 4

Which is the best interrupt vector (e.g. TIMER0_OVF_vect or ADC_vect) to use for the PWM? Assume that the duty-cycle value is in a global variable called “pwm_val”, and create an ISR for the PWM signal that reloads pwm_val to generate the necessary PWM wave. Cut and paste your code into the answer book.

Question 5

Now create a function called “startPWM” that, as its name implies, starts generating the PWM wave, using the prototype below.

```
void startPWM();
```

Cut and paste the code into your answer book.

Question 6

We will now read in the analog inputs. Write the following functions and cut and paste them into your answer book, with explanations for each of the major lines of code.

```
void setupADC(); // Set up ADC using
                // interrupt programming.
void startADC(int chan); // Starts ADC conversion for channel
                        // chan
```

Use a sampling rate of approximately 256 kHz. The startADC function takes one argument which is the ADC channel to read. Note that the ADC can only convert one channel at a time. Also write the ISR for the ADC and cut and paste the code into your answer book. Write the output of the ADC to a variable called *adc_val*. Cut and paste the ISR code as well into your answer book.

Question 7

Now write the complete program to control the brightness of the LED using the potentiometer (pot), and the pitch of the buzzer using the photoresistor. The LED should be (almost) fully off at one end of the pot's range and (almost) fully bright at the other. Similarly the buzzer should sound at 200 Hz when it is fully dark and 500 Hz when it is fully bright.

Some notes:

The ADC can only convert one channel at a time. So you need to alternate between channels 0 and 1.

The ADC output ranges from 0 to 1023, but due to physical anomalies in the photoresistor and potentiometer, the actual ADC reading will range from perhaps 100 to 900 instead of 0 to 1023. You need to know the actual range to accurately control the LED and buzzer. Since you have no way of measuring the actual ADC inputs, get creative. ☺

You may find it difficult to control the brightness of the LED and the pitch of the buzzer. Do your best. ☺

See Lab 3 if you cannot remember how to control the buzzer or how to remap ADC values to the correct range.

Create a new project called lab4part2, and copy all of your code from lab4part1 into lab4part2.c.

Question 8

Now modify your code so that:

- You no longer continuously alternate between A0 and A1.
- Instead you program timer 0 to trigger every 10 ms (approximately), and each time the timer triggers, you convert either A0 or A1. I.e. the first time the timer triggers, convert A0. The second time A1, the third time A0, etc.

Timer 0 must trigger every 10 ms (or as close as possible). If you set it to trigger at any other rate you will not get any credit for this part. You must also not affect OC0A or OC0B.

Explain the changes in detail in your answer booking, cutting and pasting code as necessary to help you explain, including how you chose the prescaler, and the register set-up.