MS101 Makerspace
2023-24/II Spring
Expt 3: DC Power Supply (Ver_Jan22)

Objectives:

i) To measure and compare the ripple voltages of half and full wave (bridge) rectifiers.
ii) To compare the output voltage variations and output resistances of the bridge rectifier based DC power supply with that of an IC regulator based DC power supply.

List of components:

Step-down transformer (230 V/0-12 V), IN4007 diodes, electrolytic capacitors, resistors, and LM7805 IC. (Optional Part C.2 – LM741 Op amp and resistors.)

Important note:
This experiment is a very basic one, but you need to be extremely careful with your wiring. Incorrect connections of the diodes and/or the electrolytic capacitors can result in explosion and fire. You might severely damage your bread board too. Please note the following:

i) **Electrolytic capacitors used in circuits of Fig.1.3 and Fig. 2.3 have polarity, i.e. the capacitors have '+' and '-' terminals. If the electrolytic capacitor is connected with the wrong polarity, it may explode and may cause fire/injury.**

ii) **Similarly, the diodes have anode and cathode terminals, which need to be connected correctly. Wrong connections can result in the diode or the transformer burning. Once again careless wiring of the diodes may result in injury.**

iii) **In order to avoid the above dangerous scenarios, you must get your circuit checked by your TA before switching ON power to the transformer. Also, while making changes, such as changing the C value from 100 µF to 1000 µF, be sure to first switch OFF power to the transformer, and then connect the capacitor once again observing the correct polarity. Get your connection ok-ed by your TA and then only switch ON the power. Similarly, it is best to switch OFF power to the transformer while changing the R_L values.**

Part A – Unregulated DC Power Supply using Half-wave Rectifier Circuit and a Capacitor Filter

1.1 Half-wave Rectifier

The circuit diagram of a half-wave rectifier circuit is shown below. We will be using a 230 V/0-12 V step-down transformer and an IN4007 diode (peak forward current: 1A, peak-inverse voltage: 1000 V) to realize the circuit. As shown in the diode image, the cathode lead is identified by the band close to it.

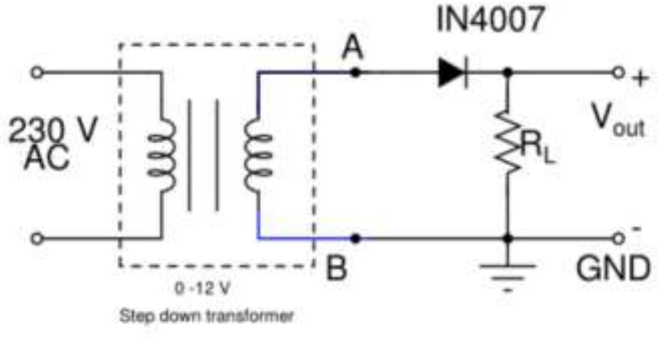


Fig 1.1 Half-wave rectifier circuit

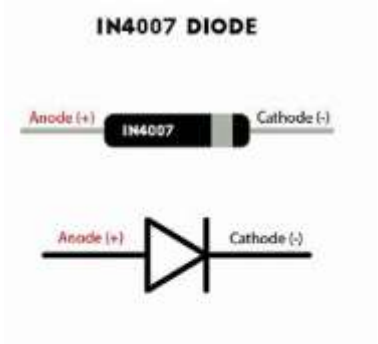


Fig. 1.2 IN4007 diode image

Experiment:

i) Wire the half-wave rectifier circuit neatly and carefully on the breadboard. $R_L = 1\ \text{k}\Omega$. **Take extra care in wiring the diode and the load resistance correctly.**

ii) Connect CH-1 of the DSO to the transformer output (the voltage V_{AB}) and CH-2 to the rectifier output (V_{out}).

Note:

a) Take extra care to avoid shock. The primary side of the transformer has already been insulated and connected to a plug. You should use only the two leads of the transformer secondary for circuit connections, i.e. points A and B shown in Fig.1.1.
(i) For getting a stable display in CH-1 and CH-2, it is best to use 'AC Line' as the trigger source. (Use: Trigger Menu > Source > AC Line)

Lab Note Book: Observation and Measurement:

a) Sketch the V_{AB} and V_{out} waveforms and note down their peak-to-peak voltages.

Note: switch-OFF power to the transformer

1.2 Half-wave Rectifier Circuit with a Capacitor Filter

The half-wave rectifier circuit of Fig. 1.1 is modified and re-drawn in Fig.1.3. In the modified circuit, a large value capacitor (1000 µF) is connected across R_L. Please note from Fig.1.2 the '-' terminal of the capacitor (the '-' terminal is generally the one with the shorter lead length).

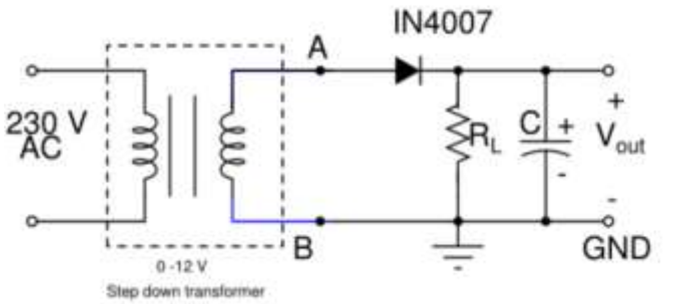


Fig 1.3 Half-wave rectifier circuit with a capacitor filter

Fig. 1.4 Images of the 1000 µF capacitor

Experiment:

i) Switch-OFF power to the transformer. Modify the circuit of Fig.1.1 so as to connect the 1000 µF electrolytic capacitor C across R_L. **Take extreme care to see that '-' terminal of the electrolytic capacitor is connected to the GND and the '+' terminal to V_{out}.**

ii) Switch ON power to the transformer.

Lab Note Book: Observation and Measurement:

a) Sketch the V_{AB} and V_{out} waveforms for the following values
i) $R_L = 1\ \text{k}\Omega$, $C = 1000\ \mu\text{F}$.

- For the above case, note down the peak-to-peak ripple voltage and the mean value of V_{out}. (Use the 'Measure' feature of the DSO to obtain the mean value of V_{out}). Please also use your DMM to measure V_{out}. For this measurement the DMM should be in the DC voltage setting.

Note: Peak-to-peak ripple voltage is relatively much smaller compared to the mean value of V_{out} (say 0.5 V of ripple riding over 16 or 17 V DC). In order to measure the small ripple voltage correctly, you should put the DSO channel to the ***AC mode*** *and choose an appropriate scale.*

Part B – Unregulated DC Power Supply using the Bridge Rectifier Circuit and a Capacitor Filter

2.1 Bridge Rectifier Circuit

Circuit diagram of the bridge rectifier circuit is shown below. We will be using the same 230 V/0-12 V step-down transformer of Part A. In addition, we will use four IN4007 diodes (peak forward current: 1A, peak-inverse voltage: 1000 V) to realize the circuit. As shown in the diode image (see Fig.2.2), the cathode lead is identified by the band close to it.

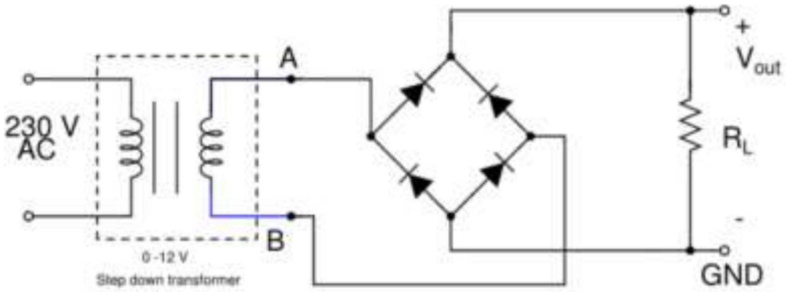


Fig 2.1 Bridge rectifier circuit

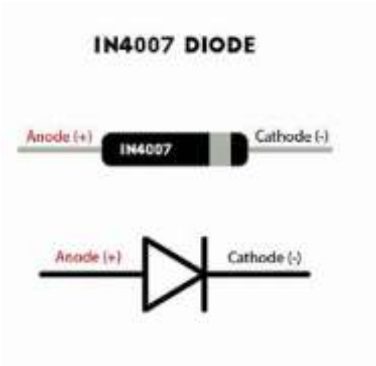


Fig. 2.2 IN4007 diode image

Experiment:

i) Wire the bridge rectifier circuit neatly and carefully on the breadboard. R_L = 10 kΩ. Take extra care in wiring the diodes correctly with the right polarity as shown. **Please note even though the wire from output B of the transformer is crossing the GND line, there is no connection there.**

ii) Connect DSO (CH-1 or CH-2) only to the rectifier output (V_{out}). <u>Do not connect any DSO probe to the transformer secondary outputs A or B. (i.e., do not try to display V_{AB} on the DSO). This will result the transformer secondary getting shorted through one of the diodes.</u>

Note:

a) For getting a stable display in **CH-1 or CH-2**, it is best to use 'AC Line' as the trigger source. (Use: Trigger Menu > Source > AC Line)

Lab Note Book: Observation and Measurement:

a) Sketch the V_{out} waveform (for R_L = 10 kΩ) and the peak-to-peak voltage.
b) After competing (a), switch-OFF power to the transformer.

1.3 Bridge Rectifier Circuit with a Capacitor Filter

The bridge rectifier circuit of Fig.2.1 is modified and re-drawn in Fig.2.3. In the modified circuit, a large value capacitor (100 µF or 1000 µF) is connected across R_L.

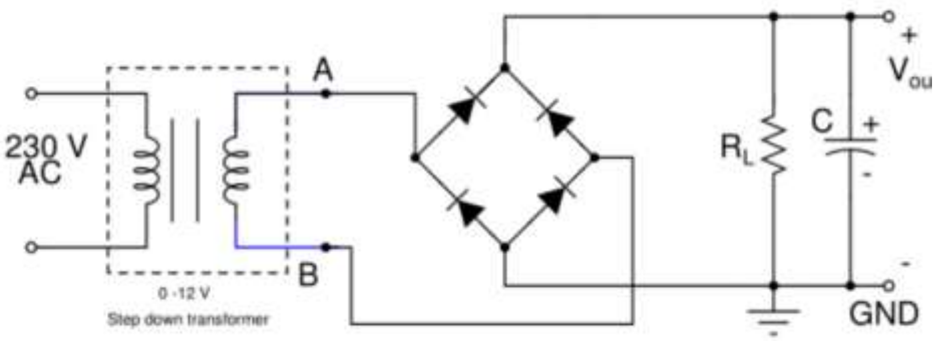


Fig 2.3 Bridge rectifier circuit with a capacitor filter

Experiment:

i) Switch-OFF power to the transformer. Modify the circuit of Fig. 2.1 so as to connect the 100 µF electrolytic capacitor C across R_L. **Take extreme care to see that '-' terminal of the electrolytic capacitor is connected to the GND and the '+' terminal to V_{out}.**

ii) Switch ON power to the transformer.

Lab Note Book: Observation and Measurement:

Sketch the V_{out} waveforms for the following values

i) R_L = 10 kΩ, C = 100 µF.

ii) R_L = 3.3 kΩ, C = 100 µF.

iii) R_L = 1 kΩ, C = 100 µF.

- For each of the above cases, note down the peak-to-peak ripple voltage and the mean value of V_{out} (= $V_{out\text{-}mean}$). Use the 'Measure' feature of the DSO to obtain $V_{out\text{-}mean}$. Use also your DMM to measure V_{out}. For this measurement the DMM should be in the DC voltage setting.

- Determine the output resistance of the bridge rectifier power supply (with capacitive filter).
  Output resistance = $|\Delta V_{out}/\Delta I_L|$.
  ΔV_{out} = ($V_{out\text{-}mean}$ for R_L = 10 kΩ) – ($V_{out\text{-}mean}$ for R_L = 1 kΩ). ΔI_L is the currents for these cases. For example, I_L (for R_L = 10 kΩ) = $V_{out\text{-}mean}$/10 kΩ. Similarly find I_L for R_L = 1 kΩ.

b) Switch-OFF power to the transformer. **Change *C* to 1000 µF, once again ensuring that '-'of the electrolytic capacitor is connected to the GND and the '+' terminal to V_{out}.**

Switch ON power to the transformer.

Lab Note Book: Observe and sketch the V_{out} waveforms for the following values

i) R_L = 10 kΩ, C = 1000 µF.

ii) R_L = 3.3 kΩ, C = 1000 µF.

iii) R_L = 1 kΩ, C = 1000 µF.

- For each of the above cases, note down the peak-to-peak ripple voltage and the mean value of V_{out} (=$V_{out\text{-}mean}$).
- Determine the output resistance of the above power supply.
  Output resistance = $|\Delta V_{out}/\Delta I_L|$. Use the method as was described earlier to determine ΔV_{out} and ΔI_L.
- Compare the output resistance of the bridge rectifier with C = 100 µF with the case when C = 1000 µF. Which case is better? Justify your answer.

Part C – IC Regulator (LM7805)

We shall use a very commonly used three-terminal 5V regulator IC to realize a regulated DC power supply. This regulator IC is capable of giving a constant V_{out} (= 5 V) for a large range of V_{in} (for V_{in} varying from 7 V to 25 V). Also, V_{out} will be a constant for a large range of I_L (for I_L varying from 0 to 1 A).

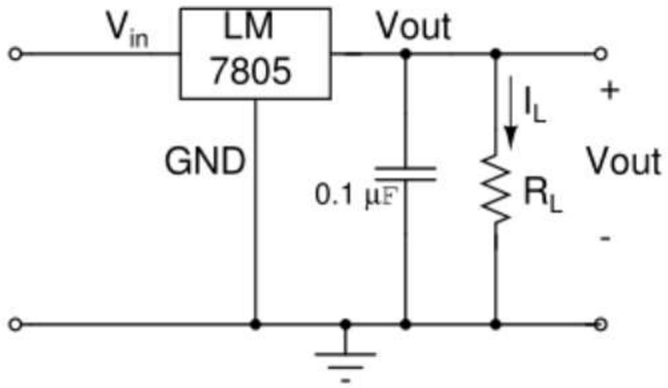


Fig. 3.1 LM7805 regulator circuit

Fig. 3.2 LM7805 IC image

Part C.1 – Measurement of Output Voltage Fluctuations with Load Currents (Output Resistance)

Circuit values: LM7805, 0.1 µF ; R_L = 680 Ω, 1 kΩ, and 3.3 kΩ

Note: V_{in} (of LM7805) = V_{out} of the Bridge rectifier circuit of Fig. 2.3 with R_L removed and C = 1000 µF;

For measuring output resistance of the LM7805 IC, we need to vary the load currents while keeping V_{in} to the circuit at a constant value. The output resistance is indicative of the load regulation of LM7805.

Experiment:

1. Wire the circuit of Fig.3.1. Identify the V_{in}, *GND* and V_{out} pins of the LM7805 and take special care to connect the three terminals correctly. The IC terminals may be thicker compared to other components; hence you need to insert the pins carefully. Also, connect the 0.1 µF ceramic capacitor as shown.
2. We shall use the output of the bridge rectifier (circuit shown in Fig.2.3, but with $_{RL}$ removed and C = 1000 µF) as V_{in} to the LM 7805 regulator IC.
3. Vary R_L values: 680 Ω, 1 kΩ, and 3.3 kΩ. For each R_L value, measure V_{out} using your DMM.

   Lab Note Book: Observation and Measurement

   a) For noting down your measurement values the Tabulate your results in to four columns, viz., V_{in} , R_L, V_{out}, I_L (in mA). ($I_L = V_{out} / R_L$)
   b) Calculate output resistance of LM7805 as: $|\Delta V_{out} / \Delta I_L|$, where ΔV_{out} is the variation in V_{out} corresponding to the minimum and maximum values of I_L.
   c) Compare the output resistance value you obtained above with that of the bridge rectifier with C = 1000 µF. Which is a better DC supply? Justify your answer.

d) Refer to the Datasheet of LM7805 IC (see the 'Electrical Characteristics' on page 2) and note the various parameters of the IC. In particular, note down the specified line regulation, load regulation and the output resistance.

Note: Since variations in V_{out} will be very small (in the mV range), your DMM may not be able to indicate these small variations; you will most likely find V_{out} to be a constant for all values of R_L.

Optional Part – (Bonus Points to those who do this properly)

Part C.2 – Accurate Measurement of Output Voltage Fluctuations with Load Currents (Output Resistance)

You would have observed that there are hardly any variations in the V_{out} for the given load resistors. This is because of the fact that the V_{out} variations are in the *mV* range. One way to measure these variations is to use a difference amplifier which measures variations of V_{out} with reference to the nominal V_{out} value.

Circuit values: + Vcc = + 12 V, - Vcc = - 12 V; $R_1 = R_3 = 10\ k\Omega$, $R_2 = R_4 = 100\ k\Omega$

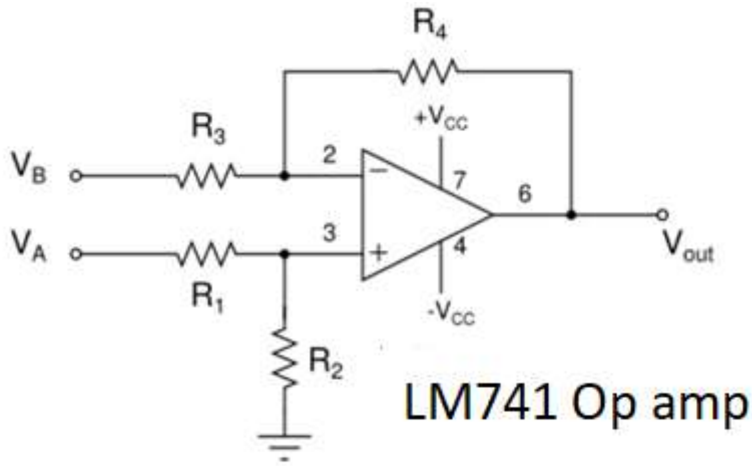


Fig. 3.3 Single-op amp difference amplifier (for measuring V_{out} variations with load)

Fig. 3.3 gives the circuit diagram of a difference amplifier. The voltage gain of the amplifier is 10. We want the difference amplifier to measure the small variations in the LM7805 variations with reference to the nominal V_{out} value (of +5V). With $V_A = V_B$, the difference amplifier output will be zero. V_{out} of the difference amplifier will amplify any voltage difference between V_A and V_B by a factor of 10. In this case this will amplify the difference LM7805 output with reference to its nominal value.

We shall connect V_A input to the LM7805 V_{out} and V_B input to a good +5 V regulated power supply. LM7805 V_{out} variations for three R_L values will be measured using the above circuit.

Experiment

1. Wire the op amp difference amplifier circuit of Fig.3.3.
2. Connect V_A input of the difference amplifier to the V_{out} of the LM7805 (Fig.3.1).
3. Using the Ch1 and Ch2 channels of the Keithley Power Supply, set the +Vcc and -Vcc voltages as +12 V and -12 V respectively for the LM741 op amp.
4. Adjust Ch3 of the Keithley Power Supply to +5V and connect it to V_B input of the difference amplifier.
5. Switch on the Keithley Power Supply.

6. Connect $R_L = 3.3\ k\Omega$. Measure V_{out} of the LM7805 and that of the difference amplifier using your DMM. It should be very close to zero (should be in the range of 10 to 20 mV). The non-zero, but small difference amplifier output voltage may be taken as the error voltage which must be subtracted from all diff amp V_{out} readings.
   (In case V_{out} of the difference amplifier is high, measure V_A and V_B values. If they are around +5 V, then the difference amplifier V_{out} will be within 100 mV).
7. For the other two R_L values of LM7805 (1 kΩ and 680 Ω) measure V_{out} of the LM7805 and that of the difference amplifier using your DMM.
8. Write your results in a Table with the following columns:
   R_L, V_{out} (LM741), I_L (mA), V_{out} (Diff amp) $\Delta V_{out\text{-}Diff}$ = [V_{out} (Diff amp)]/10
9. Calculate output resistance of LM7805 as: $|\Delta V_{out} / \Delta I_L|$, where ΔV_{out} is the variation in V_{out} corresponding to the minimum and maximum values of I_L.

Expt. 4: Arduino board Familiarization

Dinesh Sharma
D. Chakraborty, K. Chatterjee, B.G. Fernandes, J. John,
P.C. Pandey, N.S. Shiradkar, K.R. Tuckley

Department of Electrical Engineering
Indian Institute of Technology, Bombay

February 12, 2024

Arduino Nano Board

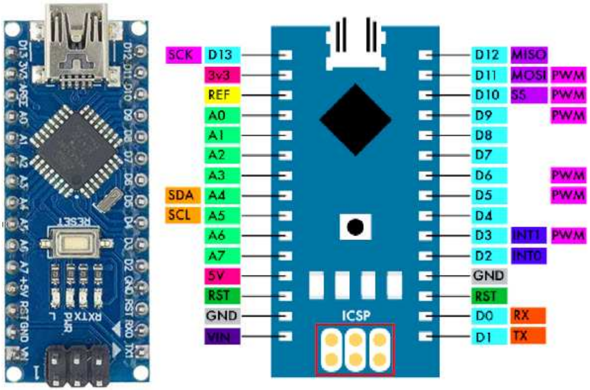


Diagram showing pins

Arduino Nano uses a USB mini connector to connect to a laptop/PC.

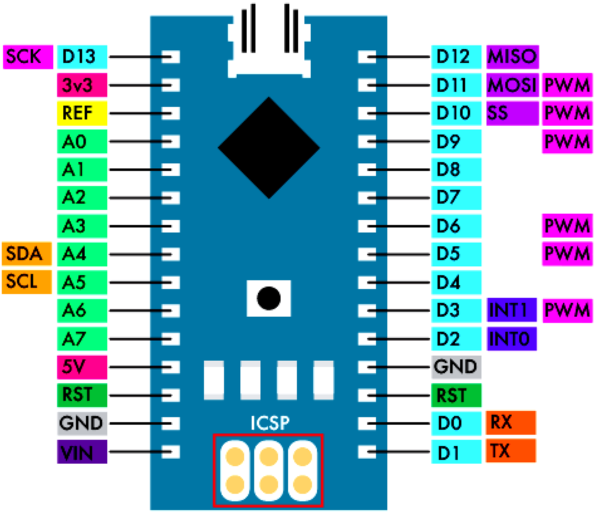
The board can be plugged in directly into a breadboard for trial circuits.

Some of the inexpensive nano boards need special drivers (CH 340) with Windows. The driver can be downloaded from
https://www.wch-ic.com/downloads/CH341SER_ZIP.html
A tutorial for installing it is available at:
https://learn.sparkfun.com/tutorials/how-to-install-ch340-drivers/all

Powering up the Arduino board

Power to the Arduino Nano board may be supplied:

1. Through the USB connector. (Nano uses a mini USB connector)
2. Through the Vin pin (bottom right in the figure above).



The "power on" LED will light up when power is applied.

Programming Arduino Boards

- Arduino boards are programmed using an Integrated Development Environment (IDE) running on a laptop or PC.
- Detailed instructions were put up on moodle for installing the IDE depending on your operating system.
- YOU SHOULD HAVE INSTALLED THE IDE ON YOUR LAPTOP BEFORE COMING TO THE LAB.
  Note that the current version of the IDE is version 2.3.0.
- In order to communicate with your board, the IDE needs to know
  1. The type of your board (Uno/Nano/...), and
  2. The serial line you will use for connecting the card to the laptop/PC.
- The write up on moodle provides all details which are required for setting up these two parameters for the IDE.

Programming Arduino Boards: Some terminology

Arduino programs which follow a pattern suitable for event-driven applications are called *sketches*.

- A sketch has two parts – the initialization part which is run first (and only once), and the repetitive part which runs in a perpetual loop.
- You need to define two functions (called "setup" and "loop") which provide these two parts. These functions receive no parameters from the calling code and return nothing to the caller (void type).
- These functions are to be written in Arduino flavour of C or C++.
- Many pre-defined functions for specific actions are available as libraries. These functions can be called from your **setup( )** or **loop( )** code, which makes it easy to write software for applications.

Experiments for Lab. 4

We shall carry out 5 simple experiments using Arduino.

A: LED Blink experiment for Digital output – to be done before coming to the lab to test your board and IDE installation.

B: LED fade experiment for Pulse Width Modulated outputs.

C: Switch reading (digital input) and serial output,

D: Reading voltage output from a potentiometer as an illustration of analog input, and

E: Using a light dependent resistor (LDR) to sense light intensity.

For this lab, we'll use ready made example programs available in the Arduino IDE with minimal modifications.

In the next lab, we shall write our own programs.

Experiments for Lab. 4

Before starting the experiments, it is assumed that

1. you have read this document and an earlier file (Introduction to Arduino) that we had put on moodle.
2. You have acquired an Arduino board.
3. You have installed and checked the IDE according to your operating system.
4. You have checked the Arduino board by connecting it to your laptop using an appropriate type of cable – USB A on the laptop side and USB B on micro-controller side for Arduino Uno and a mini USB connector for Arduino Nano.
5. You have run the blink experiment (described next) **before** coming to the lab.

Connect the board to your laptop through the USB cable and start the IDE according to the conventions of your OS (for example, by double clicking on the program name).

Expt. 4 – IDE set up

- In the IDE, click on the down arrow in the window labelled as “Select Board” and choose to “Select Other Board and Port”.
- In the search window, type Nano and the board option for Arduino Nano will come up. Click on it and a tick mark will appear against the name.
  (You could also scroll down the alphabetical list of boards to select your board).
- Click on the serial port shown on the right half of the window. A tick mark will appear here as well. Click on OK to select these choices.
- The code for a previously run program may be already displayed in the editing window. However, we shall begin with a fresh ready-made program.

Expt. 4-A – Digital Output and delay: Blinking an LED

This program does not require any external circuit or instruments. You should perform this experiment before coming to the lab to check that your board is working properly and the IDE is installed correctly.

- Click on the File tab and choose “Examples->01-Basics->Blink. A fresh edit window will open with the sketch for Blink loaded in it.
- Read through the comments to understand what the program is supposed to do and notice the function calls used.
- Click on the “Sketch” tab and choose to verify/compile. The IDE responds with “Done compiling”. (A short cut for this is control-R).
- Click on the “Sketch” tab again and choose to “Upload”. The IDE responds with “Done uploading”. (A short cut for this is control-U).
- The $\rightarrow$ icon on top can be used to compile and uploade in one go.
- The Blink program starts running. Admire your handiwork lovingly.

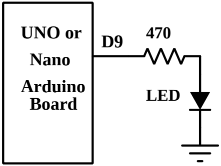
Expt. 4A – Digital Output and delay: Blinking an LED

Now we'll tinker with the sketch . . .
This part will be done in the lab.

- Change the on-time/off-time to 500 ms/1500 ms in the sketch by modifying the appropriate numbers in the code.
- Compile and upload again and run it to observe a different ratio of on and off time for the LED.
- Change the on time/off-time to 1500 ms/500 ms, recompile, upload and run.
- Call your TA to participate in your excitement . . .
  (who may spoil the fun somewhat by asking you about the function calls used and the structure of the program).
- Have your lab book signed for completion of Experiment 4-A.

Expt. 4B – Pulse Width Modulation: Fade

This experiment requires making a small circuit on your breadboard.



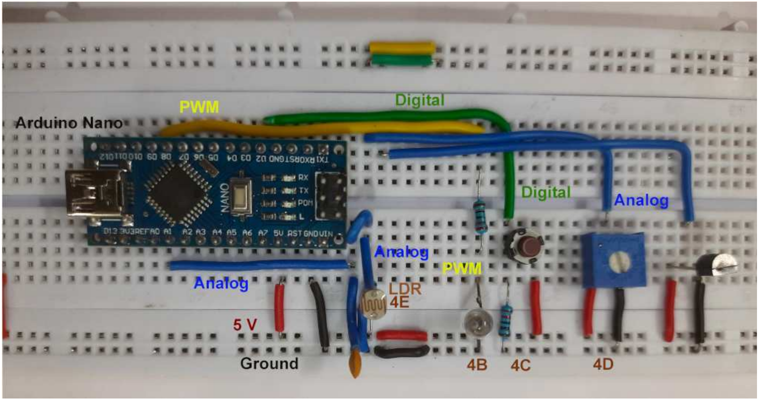
The on-board LED is connected to pin 13, which does not support Pulse Width Modulation. Hence the need for this circuit.

- Connect a wire from Digital pin D-9 on your UNO/Nano board to the breadboard.
- Connect a resistor (any value between 220Ω to 470Ω will do) from this point to the anode of an LED (its longer lead).
- Return the cathode of the LED (its shorter lead) to a long line of the breadboard for the ground connection.
- Connect a (black) wire from this long line to the GND terminal of Arduino (in the group of pins labelled as “power”).

Done!

Expt. 4B – Pulse Width Modulation: Fade

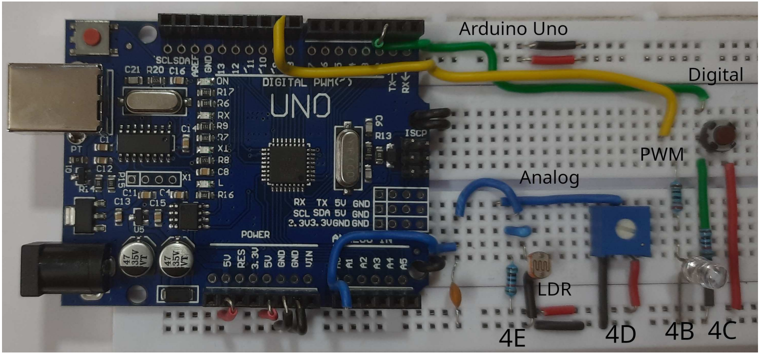
This is how I wired up all parts of expt. 4 on a breadboard, using a Nano board.



(Beware – due to parallax, wires may appear to go to the wrong pins on Arduino – follow the text, not what you perceive from the figure.)
Wire colours: Red: 5V, Black: Ground,
Green: Digital, Yellow: PWM, Blue: Analog

Expt. 4 with an Uno board

The photograph shows all parts of expt. 4 wired up using an Uno board.



(Beware – due to parallax, wires may appear to go to the wrong pins on Arduino – follow the text, not what you perceive from the figure.)
Wire colours: Red: 5V, Black: Ground,
Green: Digital, Yellow: PWM, Blue: Analog

Expt. 4B – Pulse Width Modulation: Fade

- Click on the File tab and choose "Examples->01-Basics->Fade. A fresh edit window will open with the sketch for Fade loaded in it.
- Read through the comments to understand what the program is supposed to do and notice the function calls used.
- Click on the "Sketch" tab and choose to verify/compile. The IDE responds with "Done compiling".
- Click on the "Sketch" tab again and choose to "Upload". The IDE responds with "Done uploading".
- Again, you could click on the $\rightarrow$ icon to combine the compile and upload operations.
- The Fade program starts running.
  Watch the LED on the breadboard fade in and out. View the waveform on pin D9 using the Digital Storage Oscilloscope (DSO).

Expt. 4B – Pulse Width Modulation: Fade

- Modify the sketch to use a different fading increment.
- Re-compile, upload and run. Watch the effect of slower and faster fading on the LED.
- Define separate values for increment/decrement amount so that the rate of fading is not symmetrical during fade in and fade out phases. (For this, you will have to modify the program code).

For code modification, you will define two values for fade amounts in the setup( ) function.

```
int brightness = 0; // how bright the LED is
int fd1 = 5; // how many points to fade the LED by
int fd2 = 20;
int fadeAmount = fd1; // Initial fading rate
```

(Only the changed part of the program has been described above)

Expt. 4B – Pulse Width Modulation: Fade

In the loop( ) function, you have to check for two termination conditions:

```
 //change the brightness for next time:
brightness = brightness + fadeAmount;
// reverse the direction of the fading
// when you reach max brightness
 if (brightness > 255) {
     brightness = 255-fd2;
    fadeAmount = -fd2;
  }

 if (brightness < 0) {
   brightness = fd1;
  fadeAmount = fd1;
}
// wait for 50 milliseconds per brightness change
delay(50);\\
```

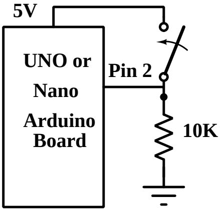
Expt. 4B – Pulse Width Modulation: Fade

After modifying the code,

- Re-compile, re-upload, re-run, and rejoice.
- View the waveform on pin 9 using the DSO.
  Draw this in your note book.
- Call the RA/TA to show your program running with asymmetrical fading rate and get the completion of Experiment 4B signed.

Do not dismantle this circuit when you go to the next part!

We need to make another simple circuit for this experiment.



Digital pin 2 will be used as the digital input. The circuit shows a switch, but we can simulate the switch by just touching a wire manually.

Connect the ground pin (among the power pins of Arduino) to the long ground line of your breadboard. Plug in a 10KΩ resistor between ground and any group of shorted 5 pins.
(Any value between 10KΩ and 100KΩ will do)

Connect a wire from digital pin 2 to the top of this resistor. Connect a wire from the 5V output in the group of pins marked “power” on Arduino.

We’ll touch the other end of this wire by hand to the top of the resistor to simulate a switch.

Expt. 4C – Digital Input, Serial Output

- Click on the File tab and choose "Examples->01-Basics->DigitalReadSerial.
  A fresh edit window will open with the sketch for DigitalReadSerial loaded in it.
- Read through the comments to understand what the program is supposed to do and notice the function calls used. Pay particular attention to the way the serial link will be established between the laptop and Arduino.
- Click on the "Sketch" tab and choose to verify/compile. The IDE responds with "Done compiling".
- Click on the "Sketch" tab again and choose to "Upload". The IDE responds with "Done uploading".

Expt. 4C – Digital Input, Serial Output

- To see the measured values, open the serial monitor.
  For this, click on the Tools tab and then on the Serial Monitor choice. This will open a window below the edit window to show serial communication between the Laptop and Arduino.
- Watch the string of '0's appearing in the serial monitor. This is because Arduino is reading a Low voltage on pin 2 through the grounded 10KΩ resistor.
- Press the push button (or touch the wire you had connected from the 5V output pin on Arduino to the top of 10KΩ resistor). You will see a series of '1's now in the serial monitor window.

On the top right of IDE window are two icons.

These can be used to invoke the digital monitor and digital plotter provided by the IDE.

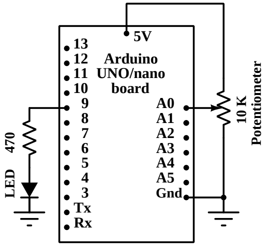
Expt. 4C – Digital Input, Serial Output

- Press and release the push button (or touch and remove the 5V wire from the top of the 10KΩ resistor) and check that the digital value read on pin 2 changes from '0' to '1' and back.
- Call your TA and claim that you have learnt:
  1. how to read digital input,
  2. how to send data through the serial line, and
  3. how to plot incoming data from Arduino on the laptop.

(The TA may be tempted to test whether you have actually learnt it)!

Get your lab book signed for completion of Experiment 4C.

Another experiment . . . another circuit.
You are now veterans of making circuits on the bread board.
So just make the circuit shown below without detailed instructions.



- Do not forget to connect the 5V and Ground lines to the 5V and ground pins in the “power” group of pins on the Arduino board.
- The breadboard compatible potentiometer covers the nearby pins – so make a note of the label on the group of 5 pins on the breadboard to which each of its terminals is connected.

The LED connected to pin D9 through a ($\approx$ 470)Ω resistor should still be there. (If you removed these by mistake, connect these again as shown).

Expt. 4D – Analog Input, Serial Output

- Click on the File tab and choose "Examples->03.Analog->AnalogInOutSerial.
  A fresh edit window will open with the sketch for AnalogInOutSerial loaded in it.
- Read through the comments to understand what the program is supposed to do and notice the function calls used.
- Pay particular attention to the way the range of 10bit ADC on AVR 328P is mapped to a range of 8bits (0 to 255), and how a serial link is established between the laptop and Arduino.
- Click on the "Sketch" tab and choose to verify/compile. The IDE responds with "Done compiling". (Short cut: control-R).
- Click on the "Sketch" tab again and choose to "Upload". (Short cut: control-U). The IDE responds with "Done uploading".

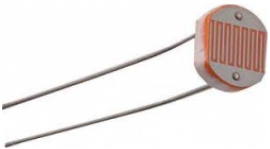
Expt. 4D – Analog Input, Serial Output

- Open the serial monitor. For this, click on the Tools tab and then on the Serial Monitor choice. (You can also click on the icon at the top right of the IDE). This will open a window below the edit window to show serial communication between the Laptop and Arduino.
- Adjust the position of the potentiometer by rotating the slider using a small screw driver. Watch the intensity of the LED being controlled by the position of the potentiometer.
- Also see the sensor output values (ADC outputs with a 10 bit range and the mapped values to 8 bit ranges) being sent through the serial port. These appear in the serial monitor. You can also use the plotter function to see how these change when you turn the potentiometer slider.

Show the operation to your TA and explain how it all works. Get your notebook signed for observations of experiment 4D.

Light detection using a Light Dependent Resistor

We'll use an LDR for detecting light intensity.

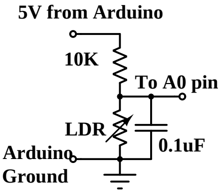


- An LDR is a resistor whose resistance value changes with light intensity.
- The LDR uses a semiconductor instead of a metal or carbon as the resistive material.

- When light falls on the resistor, electron-hole pairs are generated. Since more current carriers are available now, the current increases for a given applied voltage.
- Thus the resistance of an LDR **decreases** with increasing light intensity.
- We can use the LDR as one arm of a potential divider circuit and measure the output voltage to measure the light intensity.

Light detection using a Light Dependent Resistor

Make the potential divider circuit as shown below, on your breadboard. The only change from the circuit for experiment 4D is replacing the potentiometer with the potential divider circuit shown below.



- Notice the 0.1μF capacitor used for bypassing noise.
- 5V and Ground are taken from the Arduino board. The voltage divider output is connected to the analog input pin A0.

We shall use the same program as was used in experiment 4D.

Leave the LED connected to pin 9 through the 470Ω resistor as in experiment 4D.

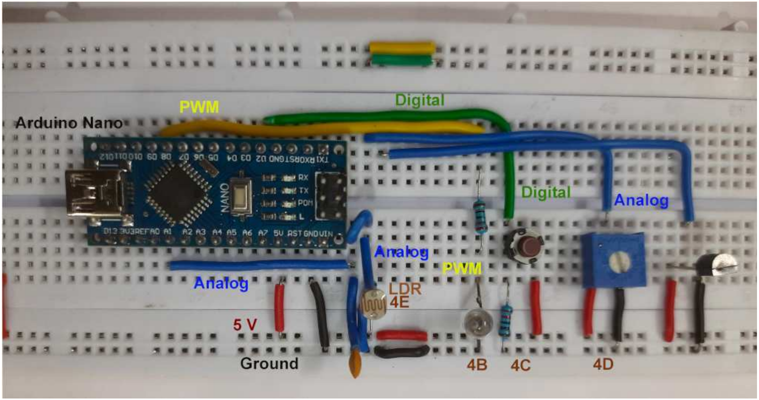
Expt. 4E – Light Intensity Detection

- compile, upload and run your sketch to send the voltage value to the serial monitor.
- Notice that the voltage will decrease as the light intensity increases.
- The effect of light intensity variation will cause the LED driven from pin D9 to be brighter when the light is less and dimmer when there is more light.
- Cover the LDR with your hand and note the output voltage.
- Shine light from your mobile on the LDR and note the output voltage.

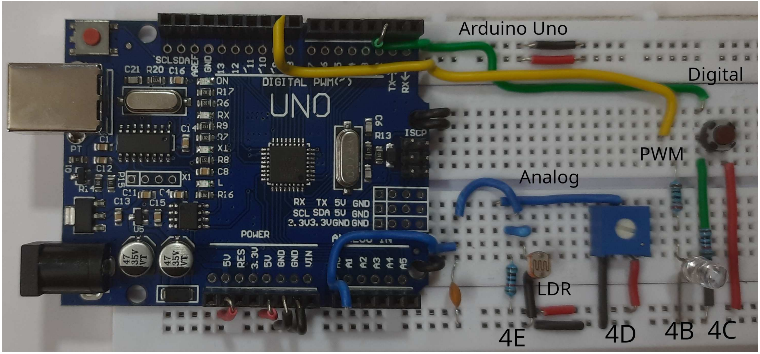
Note down and Show your readings to the TA and have these signed.

Can we use this arrangement to adjust the light intensity in a hall?

Expt. 4 with a Nano board



(Beware – due to parallax, wires may appear to go to the wrong pins on Arduino – follow the text, not what you perceive from the figure.)
Wire colours: Red: 5V, Black: Ground,
Green: Digital, Yellow: PWM, Blue: Analog



(Beware – due to parallax, wires may appear to go to the wrong pins on Arduino – follow the text, not what you perceive from the figure.)
Wire colours: Red: 5V, Black: Ground,
Green: Digital, Yellow: PWM, Blue: Analog