SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

RIAS Robotics & Intelligent Autonomous Systems

Introduction to Mobile Robotics

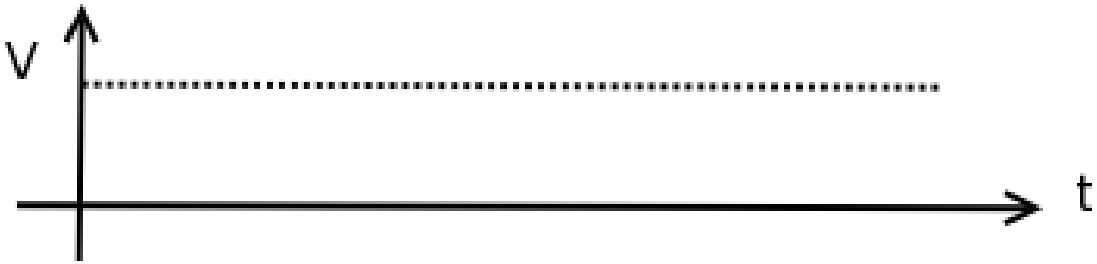
Jeff McGough

Department of Computer Science
South Dakota School of Mines and Technology
Rapid City, SD 57701, USA

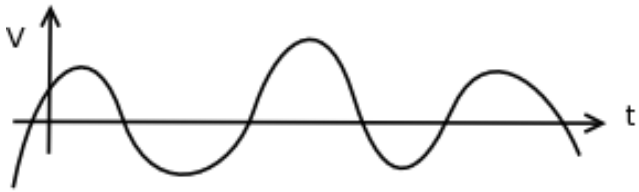
October 1, 2012

Basic Power Delivery

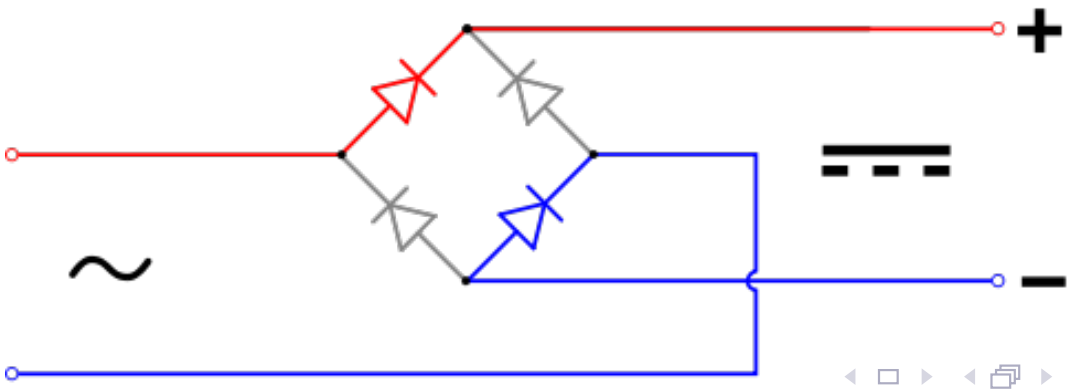
Direct Current:



Alternating Current:

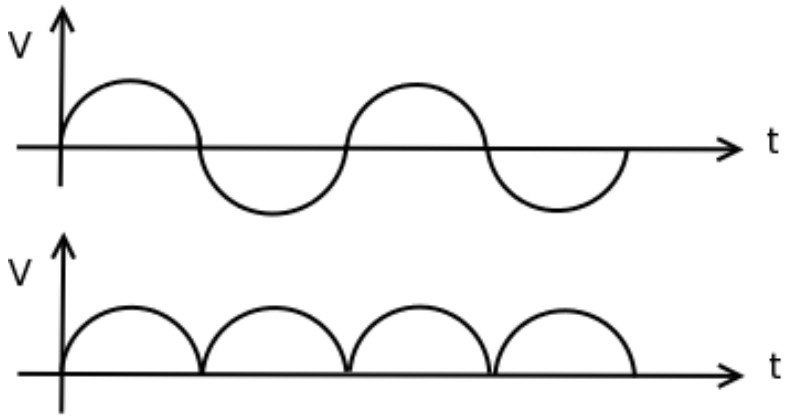


Diode bridge reroutes power

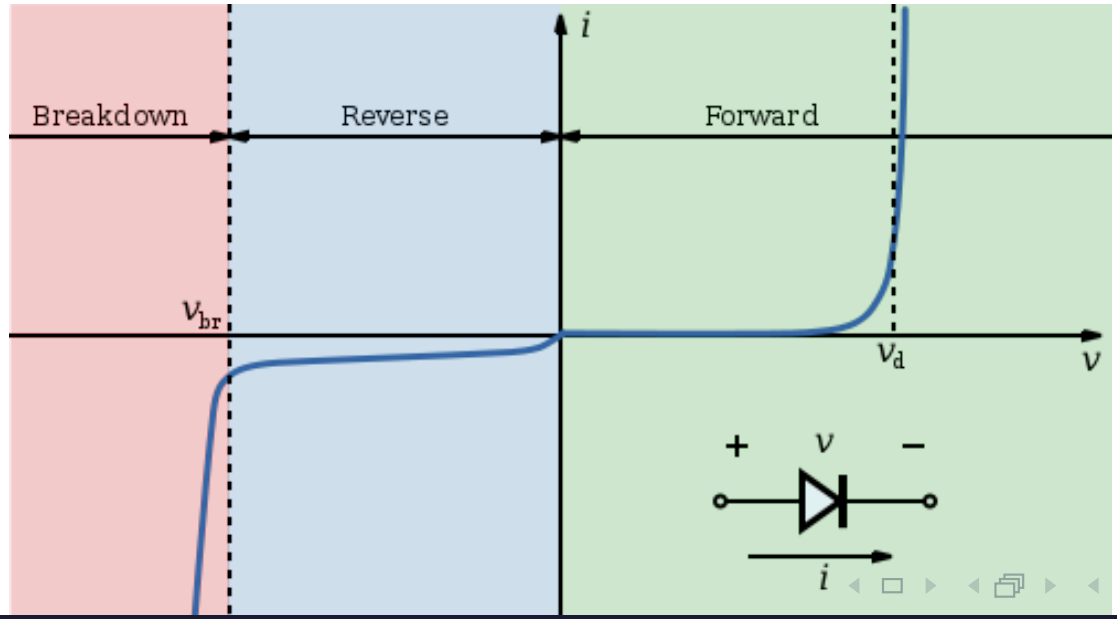


Basic Power Delivery

Bridge conversion to direct current:

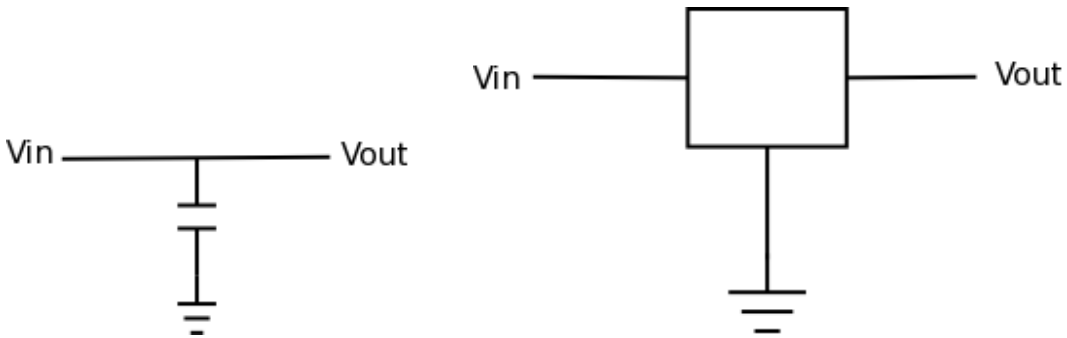


Diode:

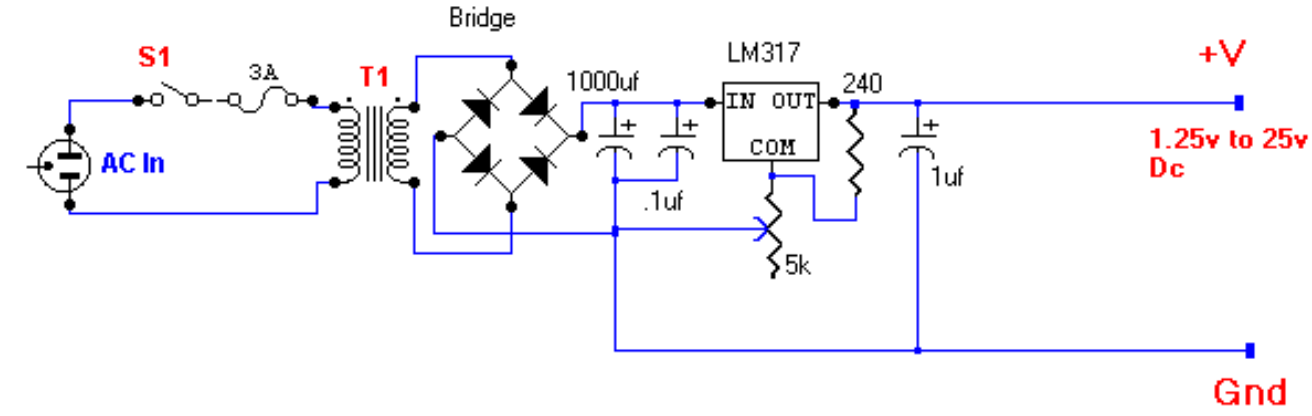


Basic Power Delivery

Smoothing using a capacitor and voltage regulation:

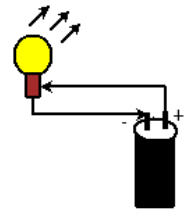


Power supply:

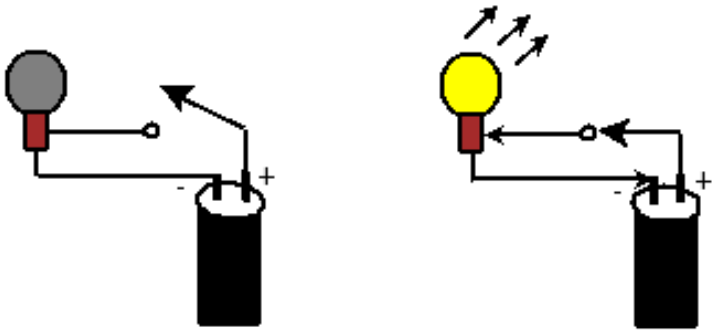


Power Delivery

Say that you have a basic electric light circuit (like a flashlight):

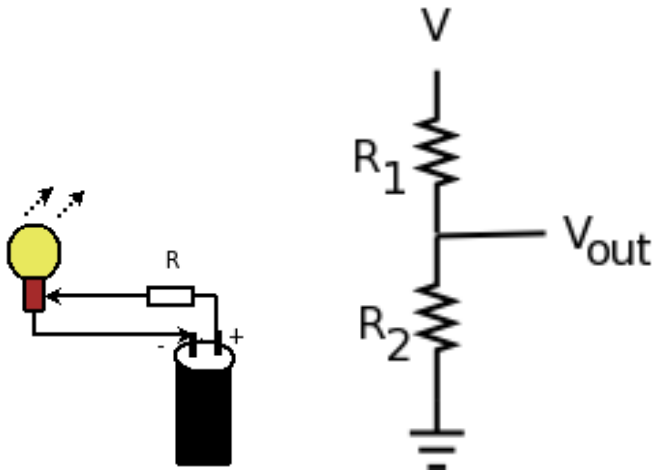


We are able to turn this on and off using a switch:



Power Dissipation

This is a great system until someone asks to dim the light. A simple approach is to place a device along the flow of current that will resist the current flow - called a resistor.



The problem with this design is that some of the energy is wasted as heat in the resistor.

Waste

Current through the resistors is

$$i = \frac{V}{R_1 + R_2}.$$

Voltage drop across R1 is

$$V_{R_1} = \left(\frac{R_1}{R_1 + R_2}\right) V.$$

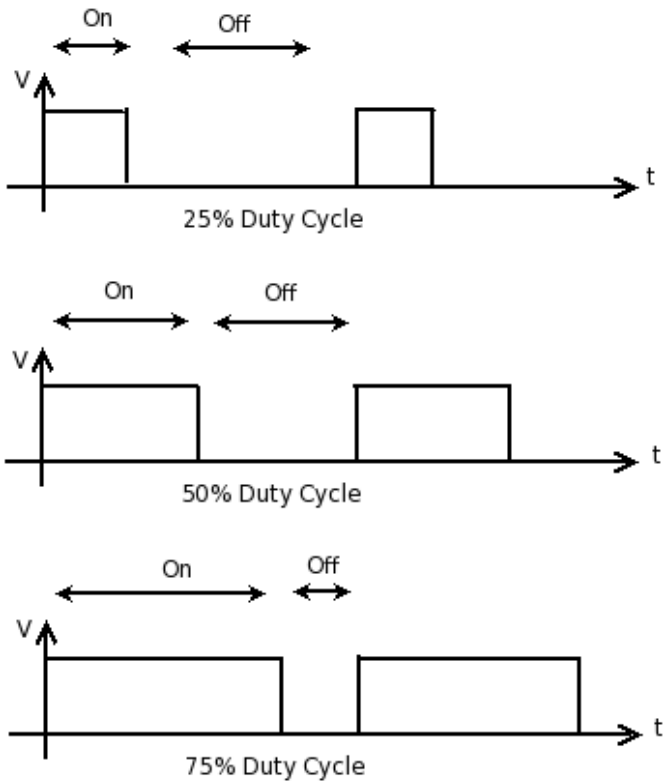
Power is

$$W = i * V_{R_1} = R_1 \left(\frac{V}{R_1 + R_2}\right)^2$$

For low power circuits, this may not be a problem, but for higher power devices like for electric motors, considerable energy is wasted as heat. [As well as shorter battery life and maybe melted circuits!]

Waste

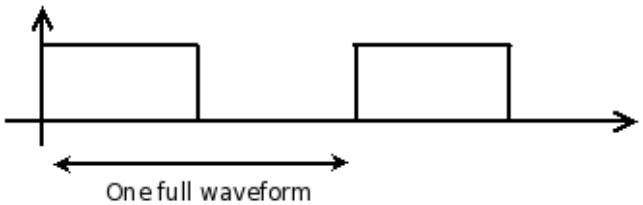
The solution is to switch on and off the power very quickly. To see what we mean, here is a picture of the voltage though time.



The amount of time the pulse is high compared to low is the duty cycle. Often this is a percent of the pulse length which is called the period.

Pulses

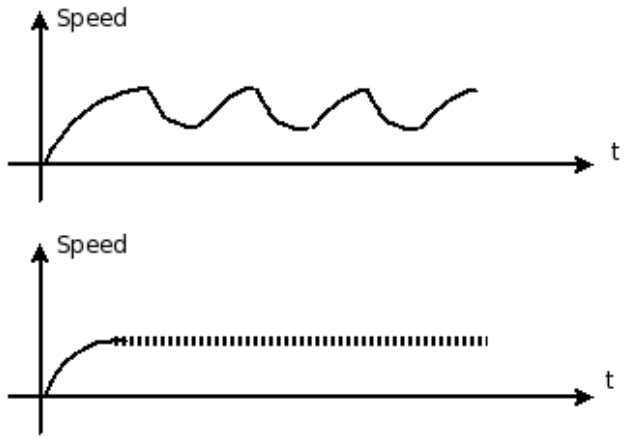
Why does this matter? By this method, we deliver a fraction of the energy which then makes light dimmer. It does not have the energy waste as compared to using a resistor. If we run the on and off fast enough, our eyes will not see the flicker and it will just appear dimmer.



This is also the method by which we control an electric motor. The frequency of this waveform does not change (because the duration of a single waveform is unchanged). The time that the voltage is high compared to the voltage is low does change.

PWM Motor Control

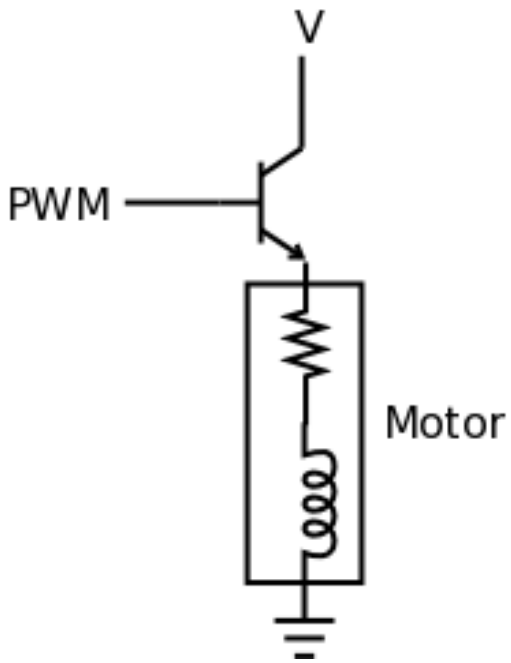
During the high part of the waveform an electric motor will start to increase in speed. During the low part the motor will coast and slow down.



You may ask how we switch the power on and off really fast. It is not like we have a little light switch and 87 cups of coffee. Hard for us, trivial for a computer. In fact, this is the basic way computers operate. They switch lines on and off millions or even billions of times per second. A program can be used to switch on and off an output line at a variety of frequencies and duty cycles.

PWM Motor Control

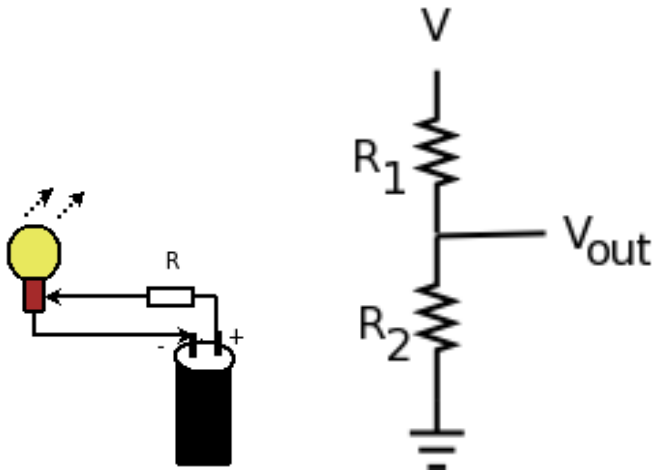
Using a pwm to drive a transistor is the way to get power delivered.



One minor problem is that this only runs one way. An H-bridge is a way to provide a reverse current.

Power Dissipation

How to dim a light? A simple approach is to place a device along the flow of current that will resist the current flow - called a resistor.



The problem with this design is that some of the energy is wasted as heat in the resistor.

Waste

Current through the resistors is

$$i = \frac{V}{R_1 + R_2}.$$

Voltage drop across R1 is

$$V_{R_1} = \left(\frac{R_1}{R_1 + R_2}\right) V.$$

Power is

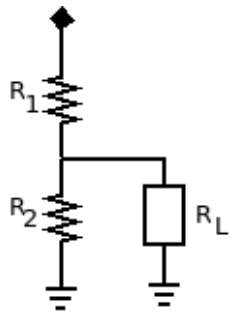
$$W = i * V_{R_1} = R_1 \left(\frac{V}{R_1 + R_2}\right)^2$$

For low power circuits, this may not be a problem, but for higher power devices like for electric motors, considerable energy is wasted as heat. [As well as shorter battery life and maybe melted circuits!]

Waste example

A quick example:

Assume we are using a 12V power source and we want to use a voltage divider to provide 9V.



Assume that the load is a simple resistor with resistance 10 ohms. Since R_2 is in parallel with the load, we get an effective resistor for the parallel combination of the load and R_2:

$$R_p = \frac{R_2 R_L}{(R_2 + R_L)} = \frac{10 R_2}{(R_2 + 10)}.$$

Waste example

The total resistance is

$$R = R_1 + R_p = R_1 + \frac{10R_2}{(R_2 + 10)}$$

The voltage drop across R_1 is $(12 - 9) = 3$ volts and the current is given by

$$i = V/R = \frac{12}{R_1 + \frac{10R_2}{(R_2+10)}} = 3/R_1$$

so ...

$$\frac{1}{4} = \left(\frac{R_1}{R_1 + \frac{10R_2}{(R_2+10)}} \right)$$

some algebra ...

$$R_1 = \frac{5R_2}{(R_2 + 10)}$$

If $R_2 = 10$ Ohms, then $R_1 = 2.5$.

Waste example

So the load uses: $W_L = iV = (9/10)9 = 8.1$ Watts.

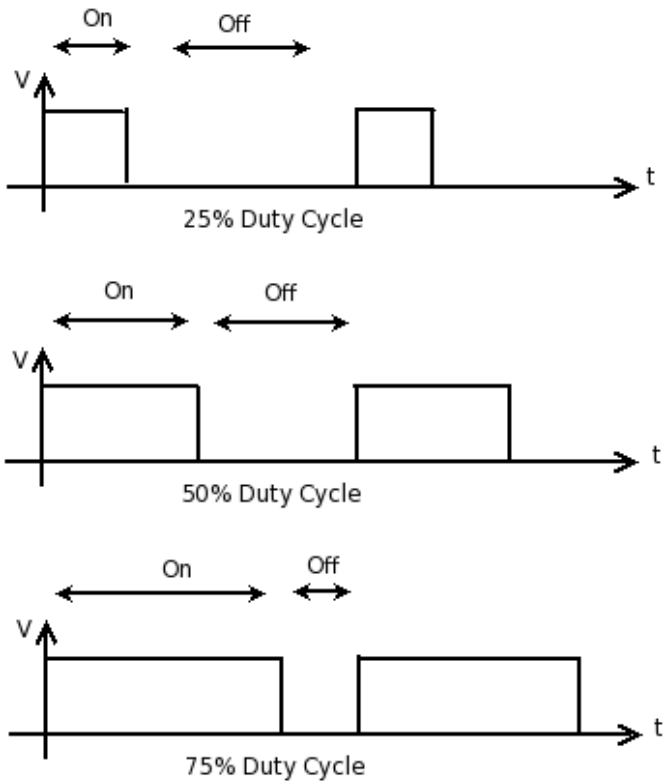
The whole circuit uses

$$W = V^2/R = \frac{12^2}{R_1 + \frac{10R_2}{(R_2+10)}} = \frac{12^2}{2.5 + \frac{100}{(20)}} = 19.2$$

A waste of 19.2 - 8.1 = 11.1 Watts.

Efficient

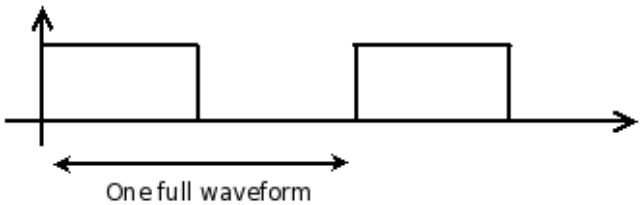
The solution is to switch on and off the power very quickly. To see what we mean, here is a picture of the voltage though time.



The amount of time the pulse is high compared to low is the duty cycle. Often this is a percent of the pulse length which is called the period.

Pulses

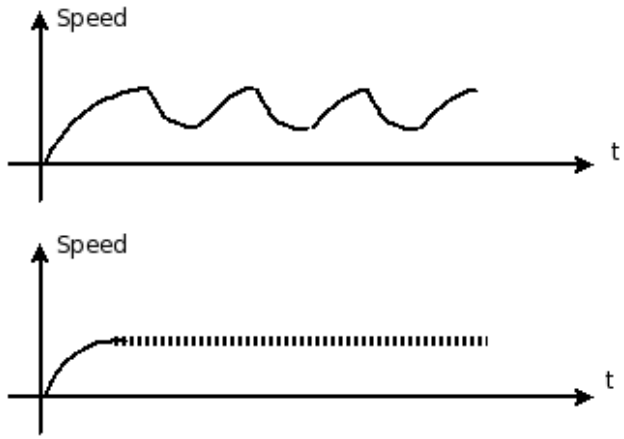
Why does this matter? By this method, we deliver a fraction of the energy which then makes light dimmer. It does not have the energy waste as compared to using a resistor. If we run the on and off fast enough, our eyes will not see the flicker and it will just appear dimmer.



This is also the method by which we control an electric motor. The frequency of this waveform does not change (because the duration of a single waveform is unchanged). The time that the voltage is high compared to the voltage is low does change.

PWM Motor Control

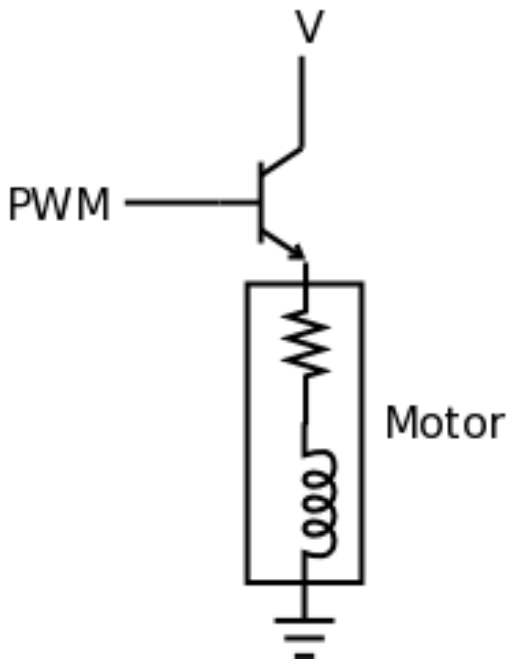
During the high part of the waveform an electric motor will start to increase in speed. During the low part the motor will coast and slow down.



You may ask how we switch the power on and off really fast. It is not like we have a little light switch and 87 cups of coffee. Hard for us, trivial for a computer. In fact, this is the basic way computers operate. They switch lines on and off millions or even billions of times per second. A program can be used to switch on and off an output line at a variety of frequencies and duty cycles.

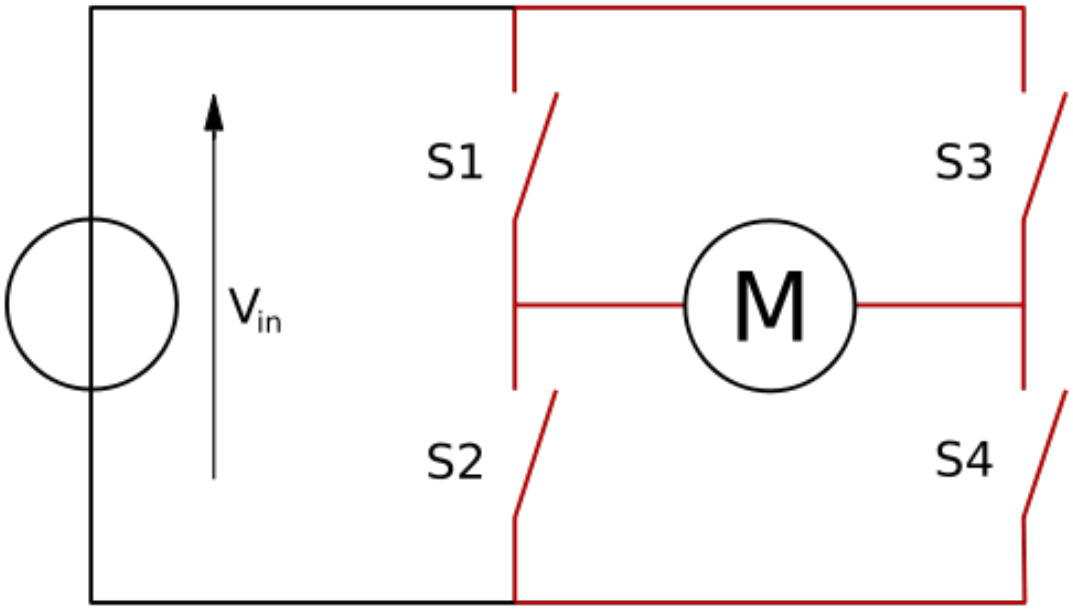
PWM Motor Control

Using a pwm to drive a transistor is the way to get power delivered.

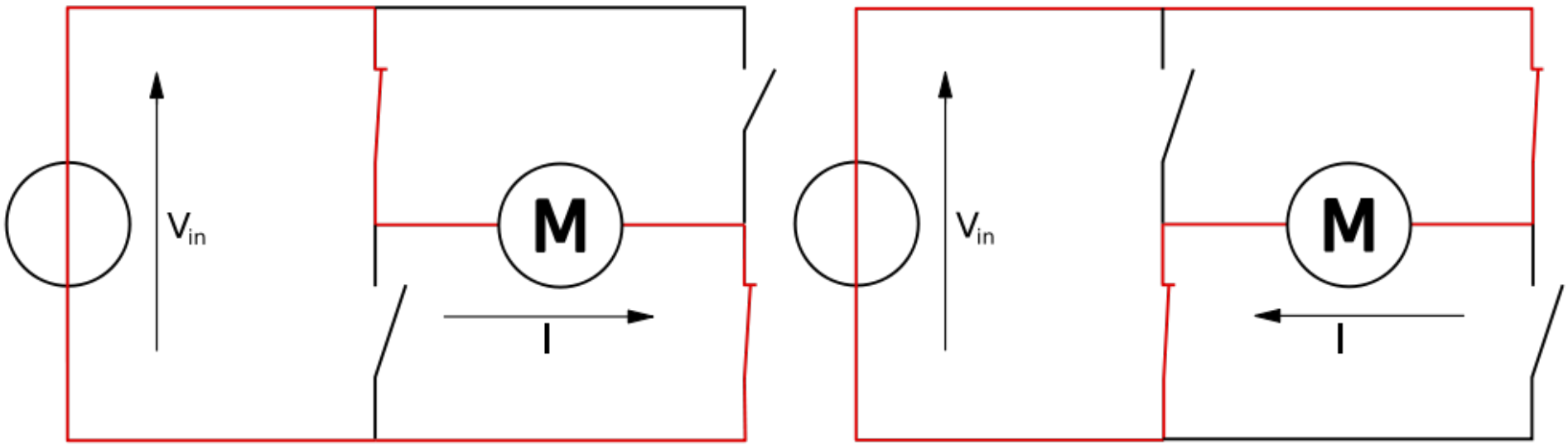


One minor problem is that this only runs one way. An H-bridge is a way to provide a reverse current.

H-Bridge



H-Bridge

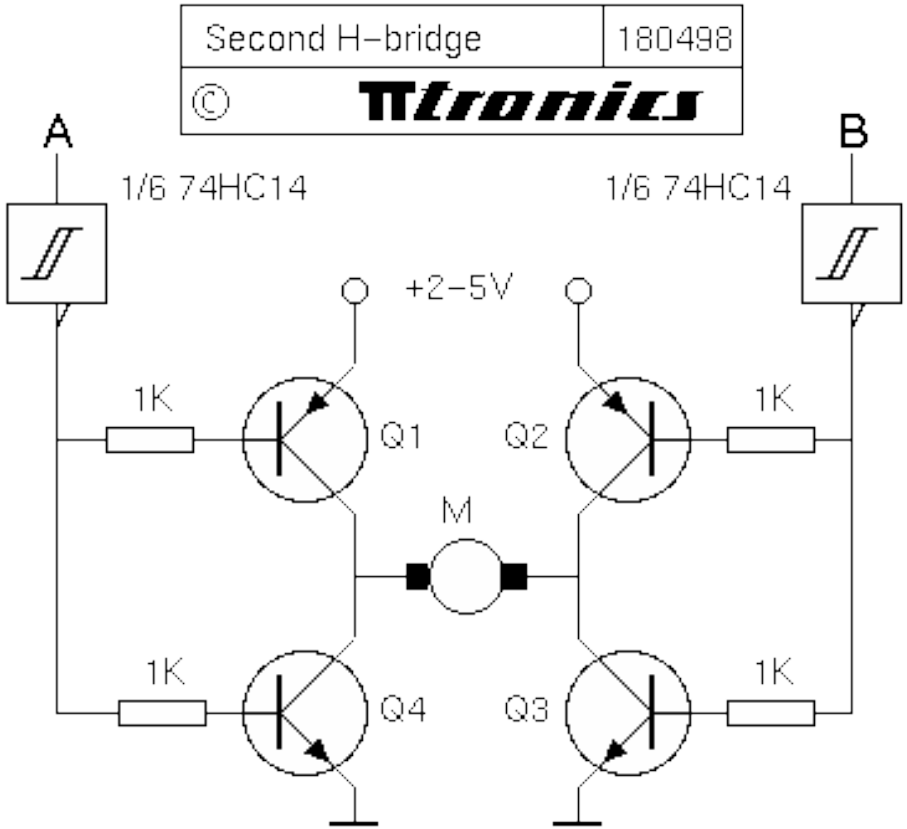


H-Bridge

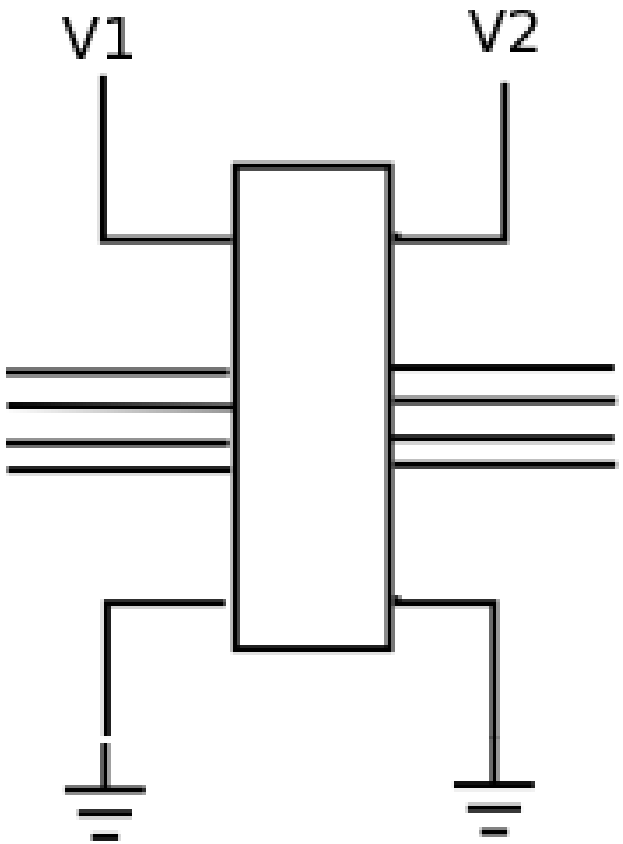
Q1, Q2 = BC327–25 or 2N2905A
Q3, Q4 = BC337–25 or 2N2219A
M = Servo motor

When A and B are equal (both 0 or both 1), the motor doesn't run
When A = 1 and B = 0, the motor turns in one direction; when A = 0 and B = 1, it turns the other way.

The circuit has handled a stalled motor @ 360mA with no trouble.
@ Vcc = 4.5V, the 74HC standard output sinks and sources 4mA

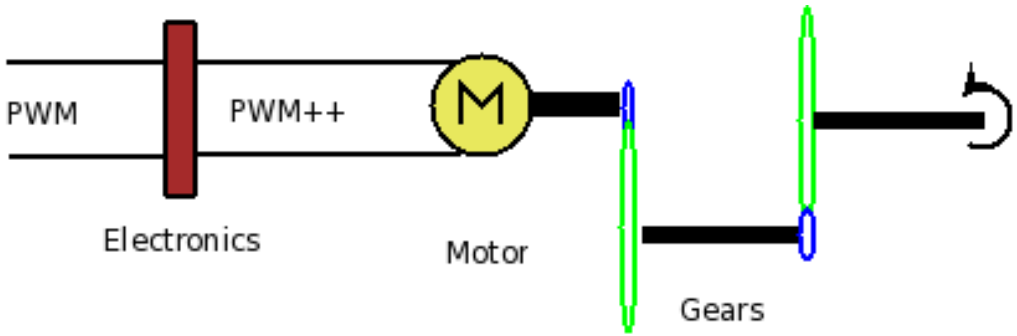


Level Shifters - interfacing hardware

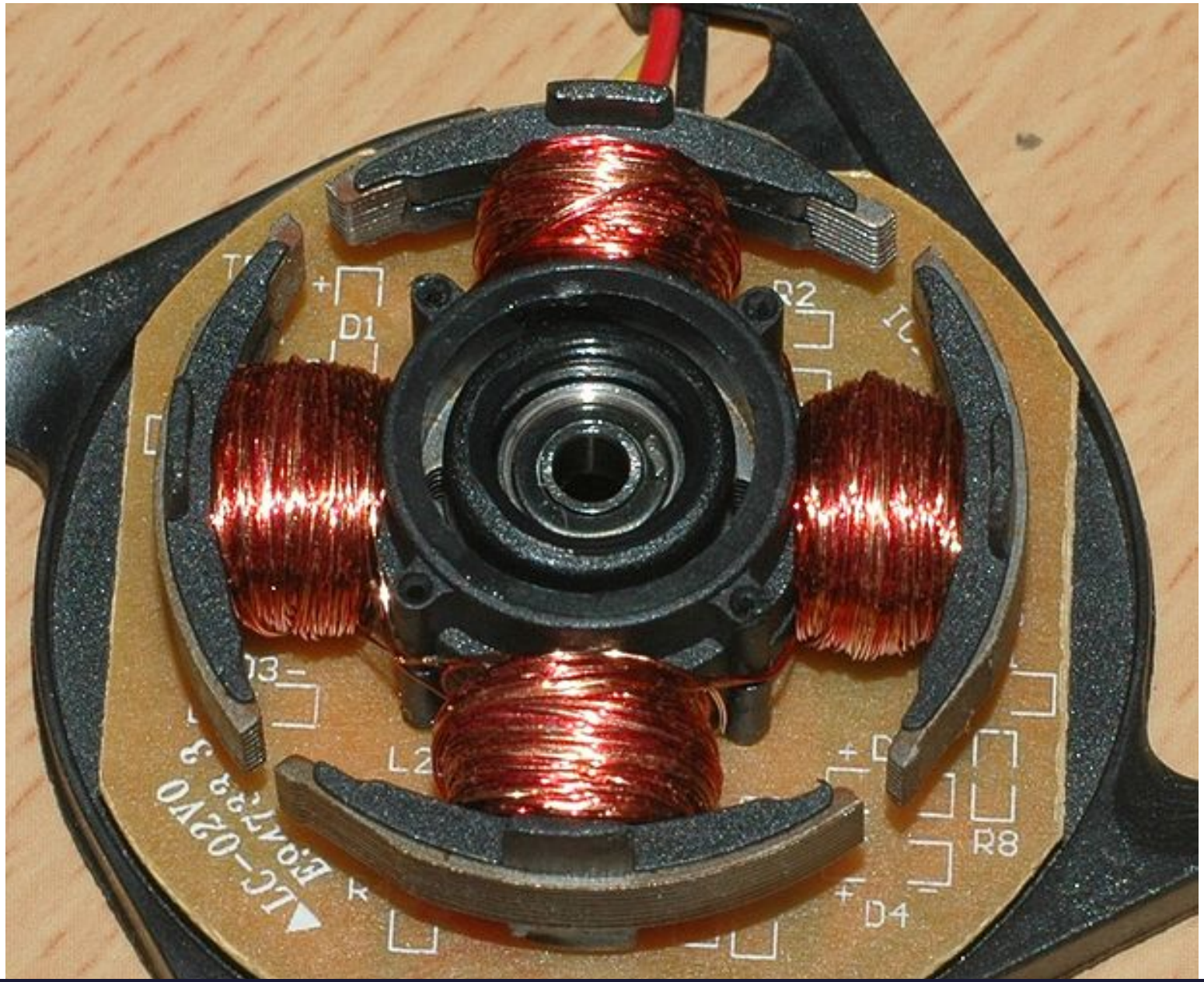


Servo

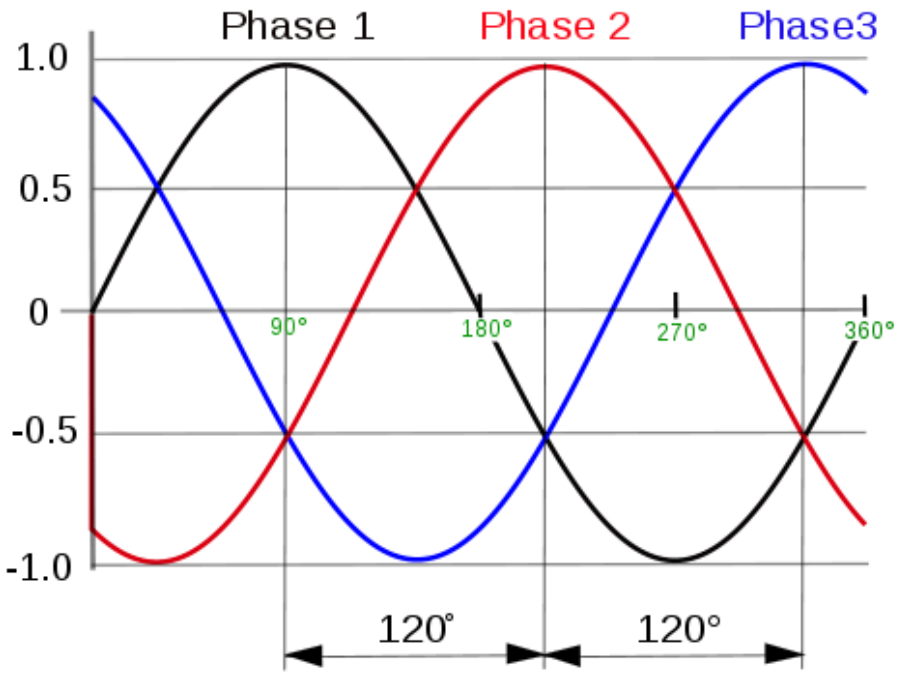
A servo is an electric motor, some electronics and a gear box

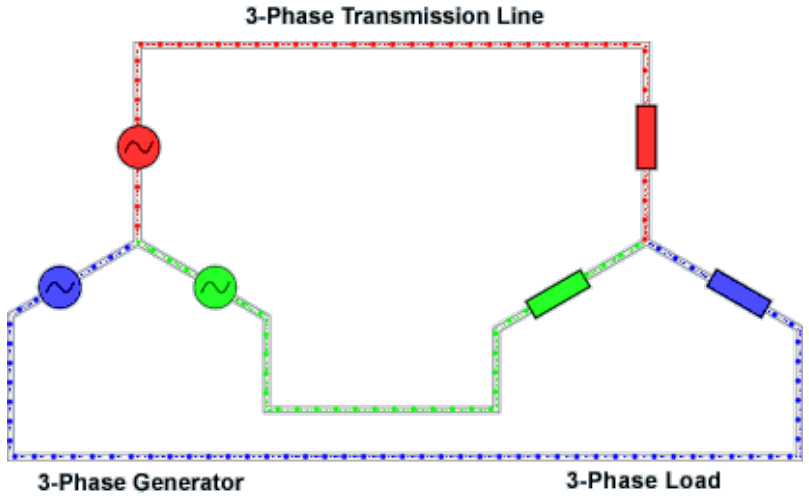


Motor detail

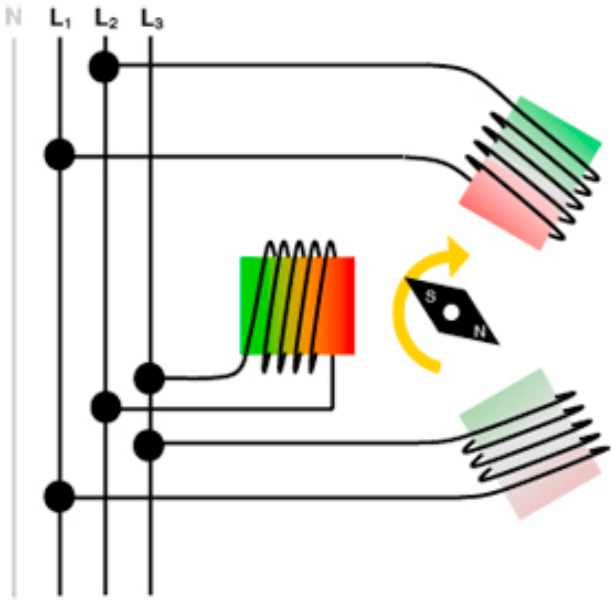


Three Phase

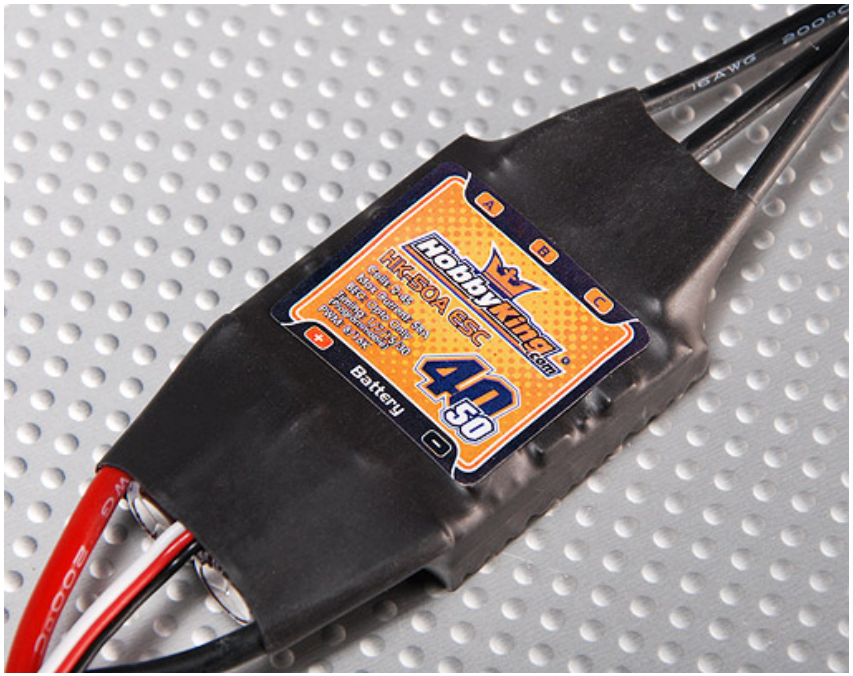

Phase 1
Phase 2
Phase3
1.0
0.5
0
-0.5
-1.0
90°
180°
270°
360°
120°
120°


3-Phase Transmission Line
3-Phase Generator
3-Phase Load

More three phase



More three phase

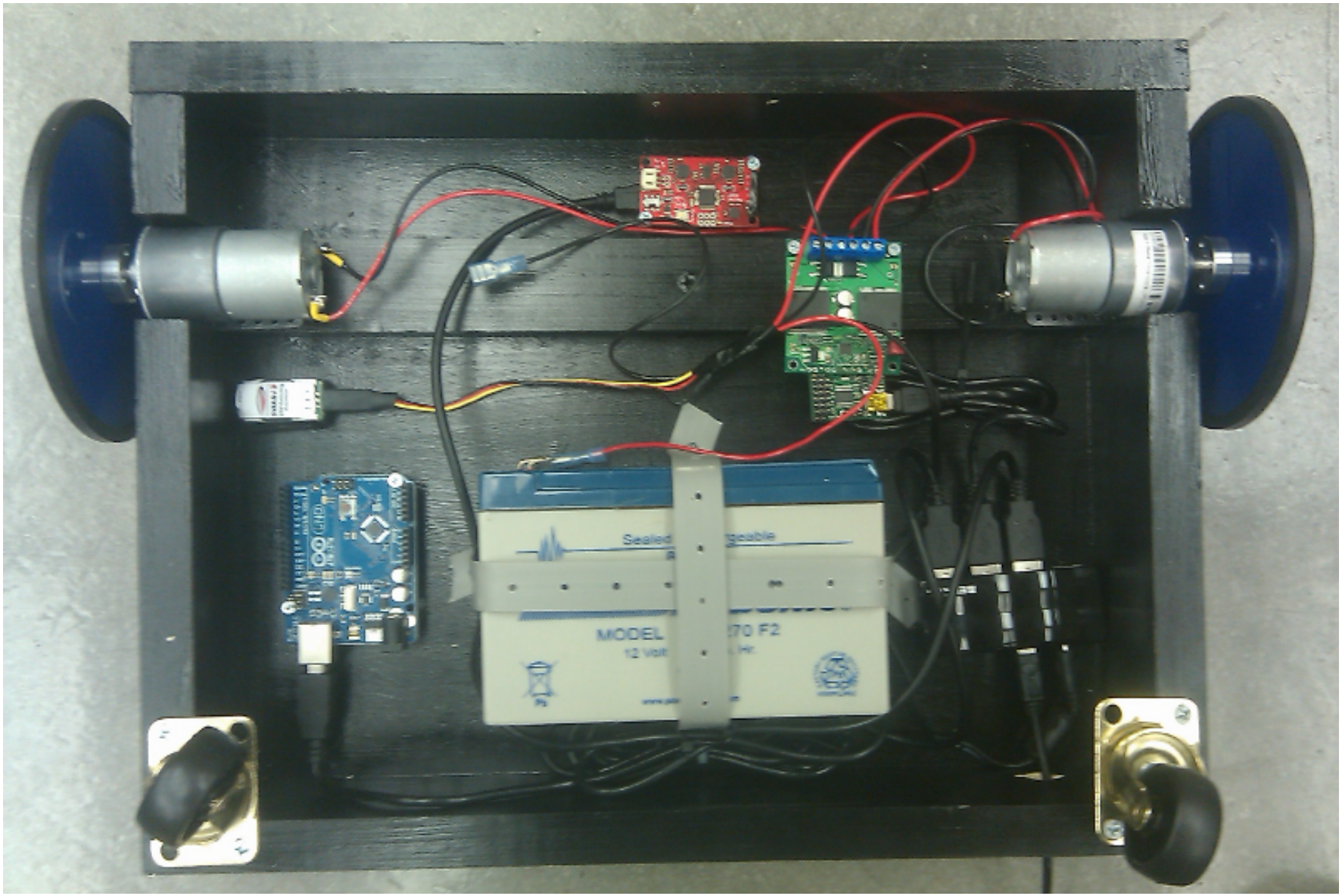


Class Robots

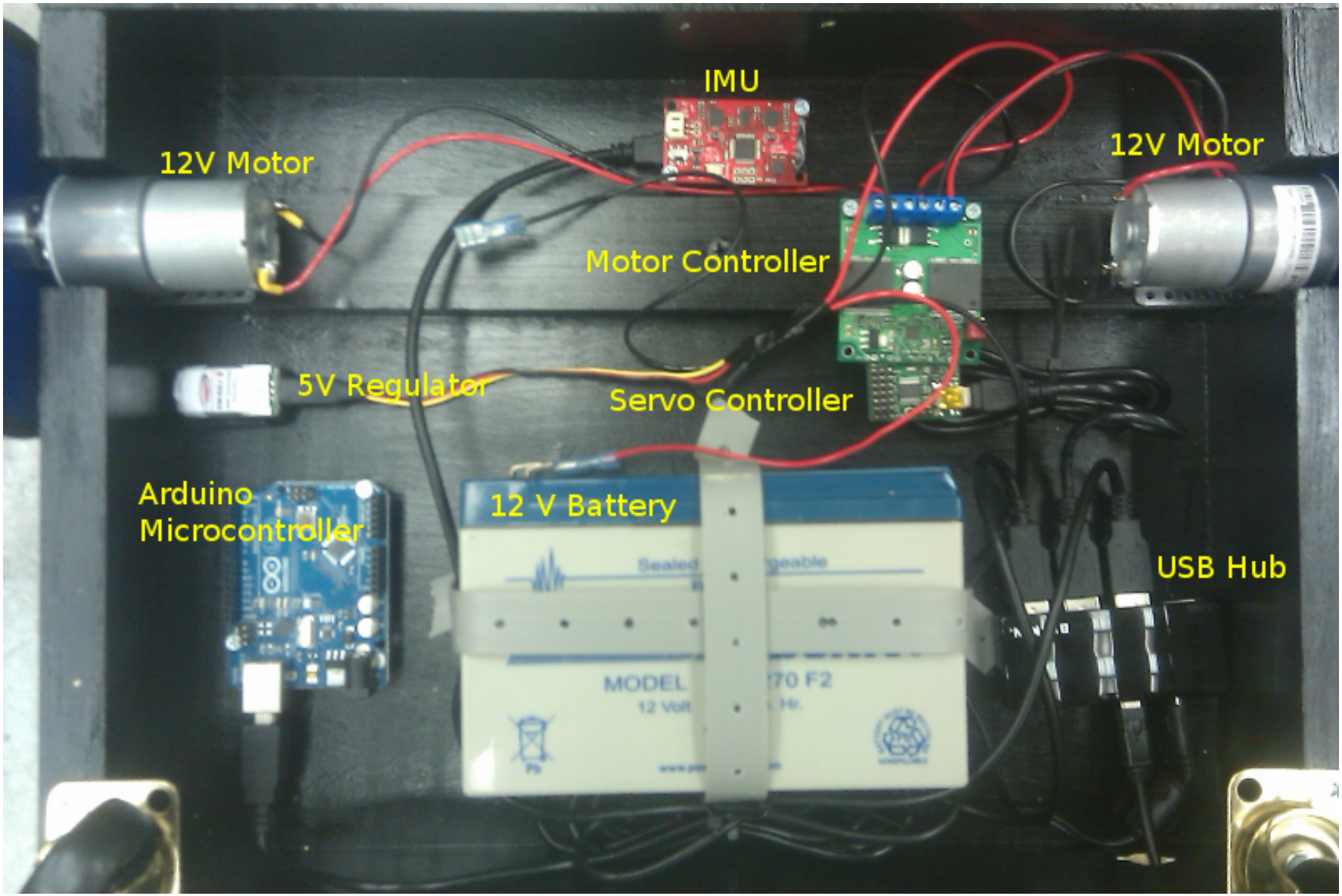
We will be using custom robots, **Scooterbots**, for CSC415/515.[1]

[1] Built by our own Ryan Housh with help from Prashanta Gyawali

Scooterbot - bottom view



Scooterbot - components



Scooterbot - control

The robot is controlled by a laptop or other device which can connect via USB.

One runs a program that sends simple commands out the USB to the robot hardware.

Any programming language that can write to a file can access the hardware. C, C++, Java, Python, etc will work.

Linux devices

POSIX (unix, linux, etc) treats everything as a file. One accesses hardware using the same metaphor as files - meaning through the file system.

The serial devices such as the USB system may be found under a branch of the file system.

Hardware may be found under the device tree: `/dev`

```
jmcgough@alta:/$ ls
bin    dev   initrd.img  lost+found  opt       sbin  sys       var
boot   etc   media       proc        selinux   tmp   vmlinuz   cdrom
home   lib
```

Device Tree

```
ls -al /dev

....
crw-rw----   1 root dialout 166,   0 2011-10-14 10:00 ttyACM0
crw-rw----   1 root dialout 166,   1 2011-10-14 10:00 ttyACM1
crw-rw----   1 root dialout 166,   2 2011-10-14 10:00 ttyACM2
crw-rw----   1 root dialout   4,  64 2011-10-14 08:58 ttyS0
crw-rw----   1 root dialout   4,  65 2011-10-14 08:58 ttyS1
crw-rw----   1 root dialout   4,  66 2011-10-14 08:58 ttyS2
crw-rw----   1 root dialout   4,  67 2011-10-14 08:58 ttyS3
crw-rw----   1 root dialout 188,   0 2011-10-14 10:00 ttyUSB0
...
```

Devices

Devices to focus on:

1. `ttyACM0, ttyACM1, ttyACM2`

are the Arduino, Servo Controller and Motor Controller.

2. `ttyUSB0, ttyUSB1`

are the IMU and GPS.

The mount order can vary. You need to check which is which.

First one must set permissions for all devices:

```
sudo chmod a+rw /dev/ttyACM*
sudo chmod a+rw /dev/ttyUSB*
```

Finding the device

To determine which file is mapped to which device:

```
udevadm info -q all -n /dev/ttyUSB0
udevadm info -q all -n /dev/ttyUSB1
udevadm info -q all -n /dev/ttyACM0
udevadm info -q all -n /dev/ttyACM1
udevadm info -q all -n /dev/ttyACM2
```

BAUD

To query the baud rate:

```
jmcgough@alta:/dev$ stty -F /dev/ttyUSB0
speed 57600 baud; line = 0;
eof = ^A; min = 1; time = 0;
-brkint -icrnl -imaxbel
-opost -onlcr
-icanon -echo -echoe
```

To set the baud rate:

```
stty -F /dev/ttyUSB0 9600
```

Baud rates for `/dev/ttyACM*` = 9600[2], `/dev/ttyUSB*` = 4800

[2] must be done individually

Check devices

Check the device on USB0:

```
jmcgough@alta:/dev$ cat /dev/ttyUSB0
373
381
384
-4
52
-31
$ANG,-0.04,0.04,-0.04
```

IMU appears to be working ...

Motors

Motor 0 = Right Motor, Motor 1 = Left Motor

Initializing motors: byte: 0xAA

Setting motor speed:

Motor 0 - byte1: 0x88 (forward), byte2: [0-127]
Motor 0 - byte1: 0x8A (reverse), byte2: [0-127]
Motor 1 - byte1: 0x8E (forward), byte2: [0-127]
Motor 1 - byte1: 0x8C (reverse), byte2: [0-127]

Sample commands:
0xAA
0x88 0x5A
0x8E 0x5A

C++ Code to run motors

```
#include <iostream>
#include <fstream>
using namespace std;

int main( int argc, char **argv ){
    fstream motor;
    motor.open( argv[1], fstream::in | fstream::out );

    sleep(1);

    char temp = 0xaa;
    motor.write((char*)&temp, 1);
    //if( motor.tellg() == -1 );

    sleep(.5);

    temp = 0x8e;
    motor.write((char*)&temp, 1);
    temp = 127;
    motor.write((char*)&temp, 1);
    if( motor.tellg() == -1 );

    temp = 0x88;
    motor.write((char*)&temp, 1);
    temp = 127;
    motor.write((char*)&temp, 1);
    if( motor.tellg() == -1 );

    sleep(2);

    temp = 0x8c;
    motor.write((char*)&temp, 1);
    temp = 0;
    motor.write((char*)&temp, 1);
    if( motor.tellg() == -1 );

    temp = 0x88;
    motor.write((char*)&temp, 1);
    temp = 0;
    motor.write((char*)&temp, 1);
    if( motor.tellg() == -1 );

}
```

C Code to run motors

```
#include<stdio.h>

int main()
{
  char cmd;
  FILE * fp;
  fp = fopen ("/dev/ttyACM1","w+");

  cmd = 0xAA;
  fprintf(fp, "%c\n", cmd);

  cmd = 0x8e;
  fprintf(fp, "%c", cmd);
  cmd = 0x5A;
  fprintf(fp, "%c", cmd);

  cmd = 0x88;
  fprintf(fp, "%c", cmd);
  cmd = 0x5A;
  fprintf(fp, "%c", cmd);

  fflush(fp);

  sleep(20);

  cmd = 0x8e;
  fprintf(fp, "%c", cmd);
  cmd = 0x00;
  fprintf(fp, "%c", cmd);

  cmd = 0x88;
  fprintf(fp, "%c", cmd);
  cmd = 0x00;
  fprintf(fp, "%c", cmd);

  close(fp);

}
```

Sensor Background

- Why should a robotics engineer know about sensors?
  - Key technology for perceiving the environment
  - Understanding the physical principles enables appropriate use
- Understanding the physical principle behind sensors enables us
  - To properly select the sensors for a given application
  - To properly model the sensor system, e.g. resultion, bandwidth, uncertainties
  - To define the needs in collaboration with the sensor system suppliers

Sensor Classes

Classes

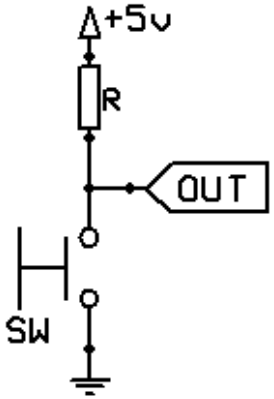
- Properioceptive
  - measure values internally to the system (robot)
  - values like motor speed, wheel load, orientation, battery status
- Exteroceptive
  - information from the robot's environment
  - distances to objects, intensity of ambient light, unique features

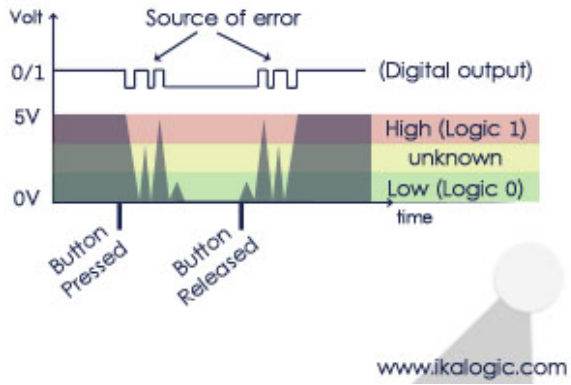
How

- Passive sensors
  - energy from environment
- Active sensors
  - emit energy in the correct configuration and measure the reaction
  - better performance but influences the environment

Switches

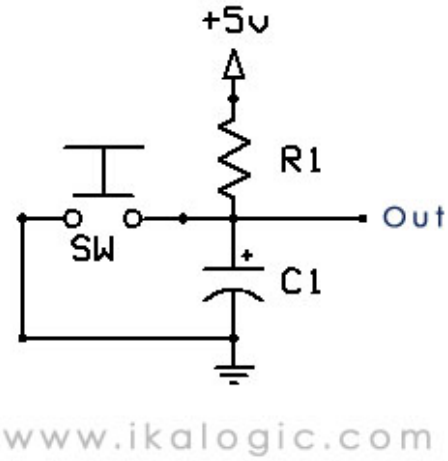
The most elementary sensor - Switches ...

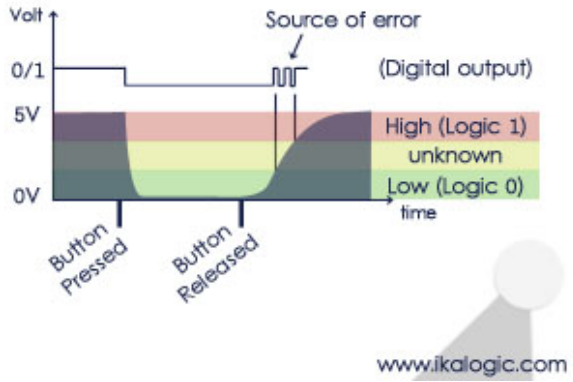




Switches

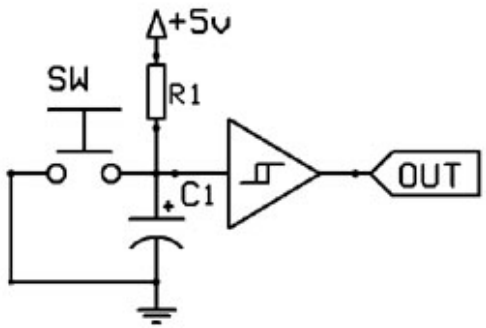
Debounce issues

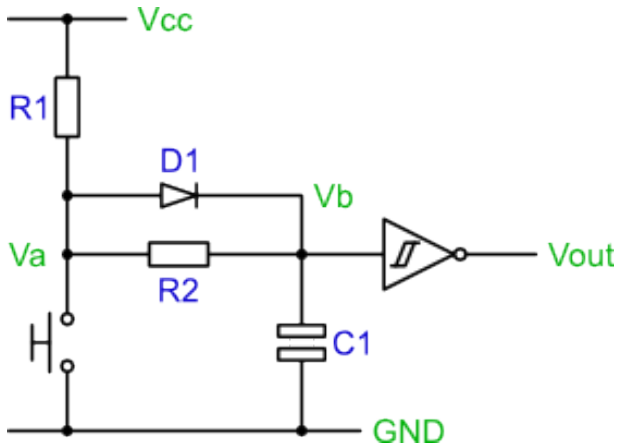




Switches

Debounce solutions in hardware

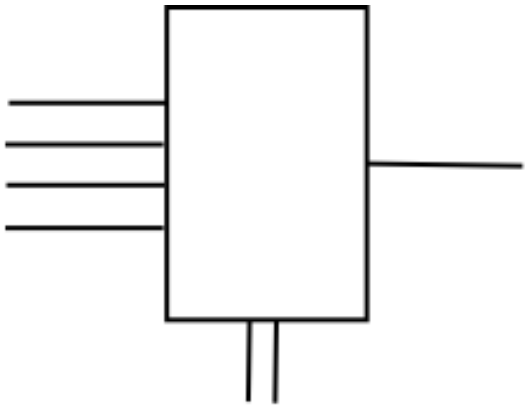


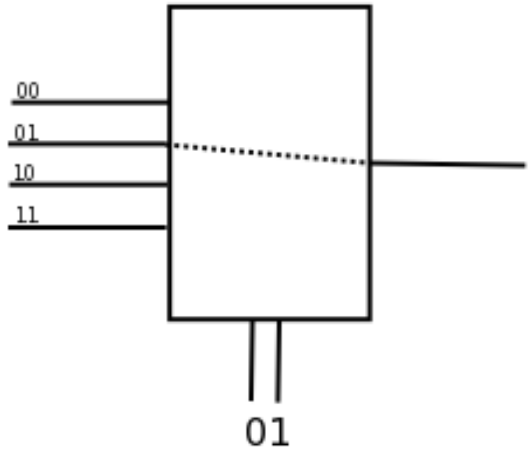


Software solutions are also available and normally depend on waiting a short time.

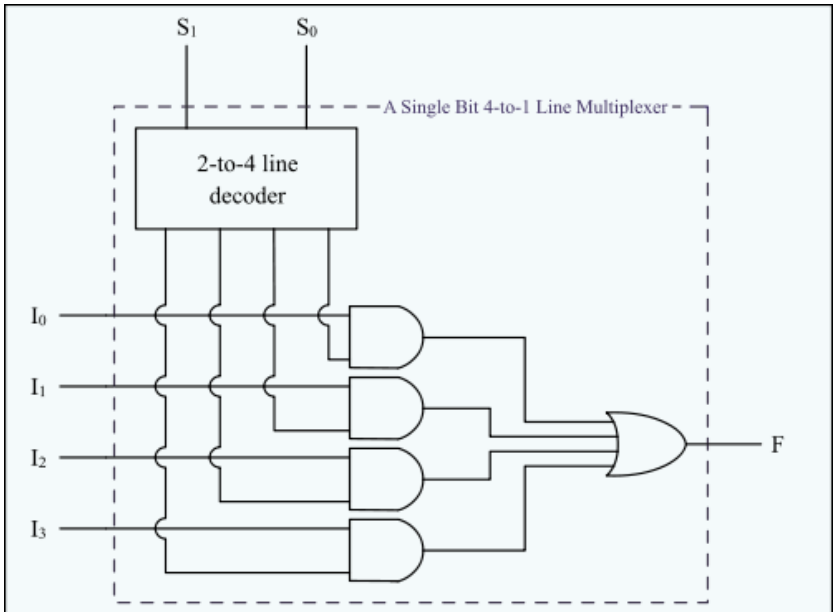
Multiplexing and demultiplexing

How does one deal with dozens of devices and a few GPIO?





Multiplexing and demultiplexing



Truth Table

S_1	S_0	I_3	I_2	I_1	I_0	F
0	0	0	0	0	0	0
		0	0	0	1	1
		0	0	1	0	0
		0	0	1	1	1
		0	1	0	0	0
		0	1	0	1	1
		0	1	1	0	0
		0	1	1	1	1
		1	0	0	0	0
		1	0	0	1	1
		1	0	1	0	0
		1	0	1	1	1
		1	1	0	0	0
		1	1	0	1	1
		1	1	1	0	0
		1	1	1	1	1

S_1	S_0	I_3	I_2	I_1	I_0	F
0	1	0	0	0	0	0
		0	0	0	1	0
		0	0	1	0	1
		0	0	1	1	1
		0	1	0	0	0
		0	1	0	1	0
		0	1	1	0	1
		0	1	1	1	1
		1	0	0	0	0
		1	0	0	1	0
		1	0	1	0	1
		1	0	1	1	1
		1	1	0	0	0
		1	1	0	1	0
		1	1	1	0	1
		1	1	1	1	1

S_1	S_0	I_3	I_2	I_1	I_0	F
1	0	0	0	0	0	0
		0	0	0	1	0
		0	0	1	0	0
		0	0	1	1	0
		0	1	0	0	1
		0	1	0	1	1
		0	1	1	0	1
		0	1	1	1	1
		1	0	0	0	0
		1	0	0	1	0
		1	0	1	0	0
		1	0	1	1	0
		1	1	0	0	1
		1	1	0	1	1
		1	1	1	0	1
		1	1	1	1	1

S_1	S_0	I_3	I_2	I_1	I_0	F
1	1	0	0	0	0	0
		0	0	0	1	0
		0	0	1	0	0
		0	0	1	1	0
		0	1	0	0	0
		0	1	0	1	0
		0	1	1	0	0
		0	1	1	1	0
		1	0	0	0	1
		1	0	0	1	1
		1	0	1	0	1
		1	0	1	1	1
		1	1	0	0	1
		1	1	0	1	1
		1	1	1	0	1
		1	1	1	1	1

LEDs

Light emiting diodes - LEDs

These are diodes which emit at specific EM bands.

LEDs have a variety of operating specs.

Assume we have one that operates in the 3-6 volt range and current level is 20mA. If we select V = 5 then the resistor should be R = V/I = 5/.02 = 250. Since 250 is not a standard value, we select R =270 ohms.