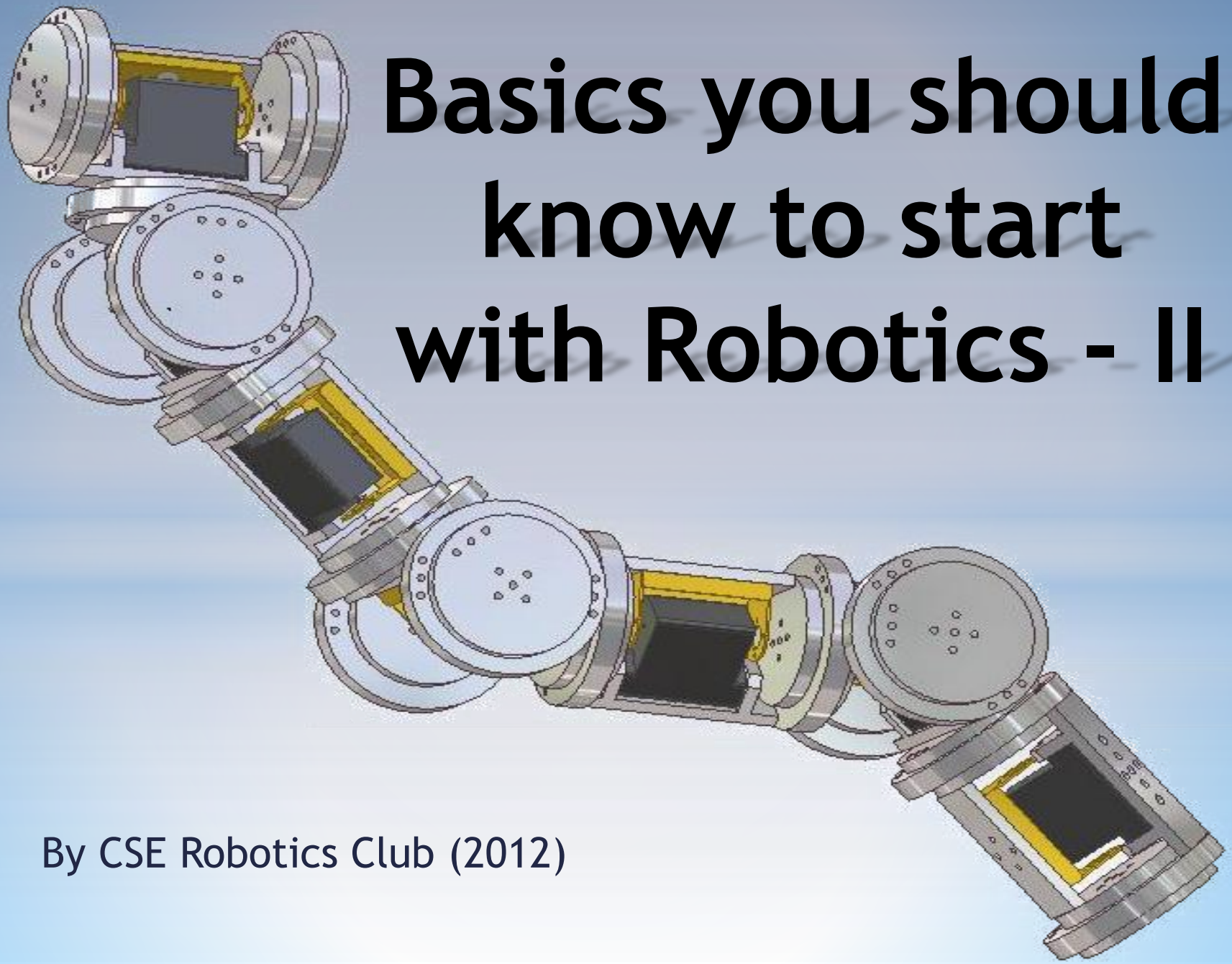
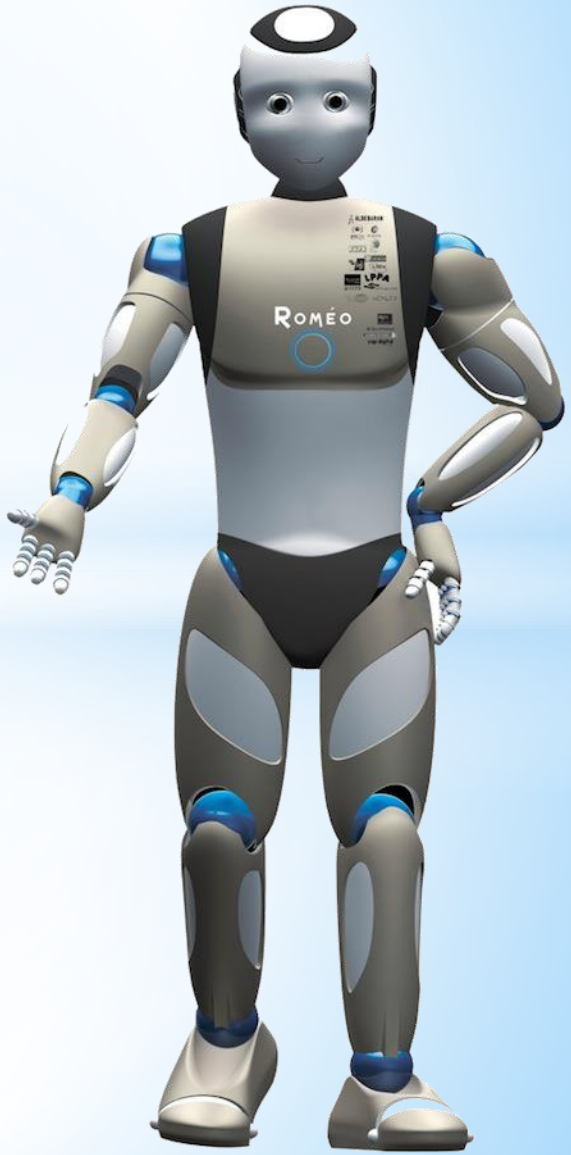
Basics you should know to start with Robotics - II



By CSE Robotics Club (2012)

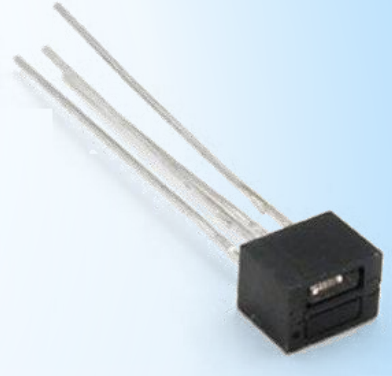
Now you Know…

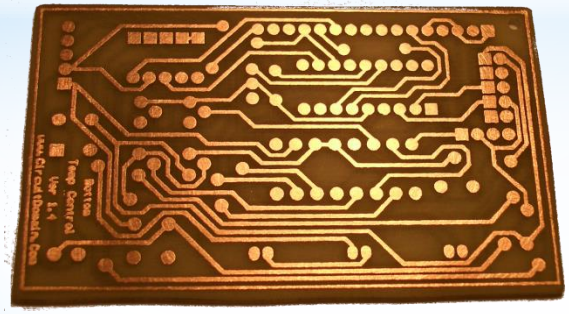
- About Programmable devices
- How to write programs using PIC-C
- Use of software to design and simulate (Proteus, Pic18 Simulator)
- How to load the program to a PIC



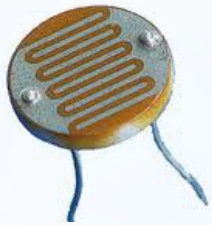
Today we will discuss…

- Sensors to get details from the environment
- Handling movements
- Prepare a PCB (Printed Circuit Board)
- More programming aspects
  - Interrupts
  - Using EEPROM
  - PWM (Pulse Width Modulation)





1. Sensors



Photocells- detect light

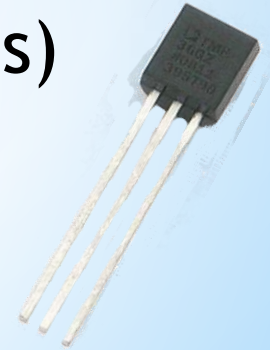
IR receivers- detect IR signals (helps to detect white vs. Black)

Force sensitive resistor –physical pressure

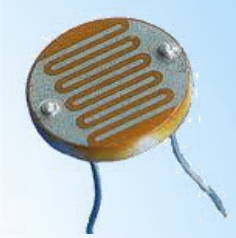
Tilt sensors – detect motion/vibration and orientation

Thermocouple- for temperature measurements

Sonar sensors – sound detection (obstacles)



Detect Colors



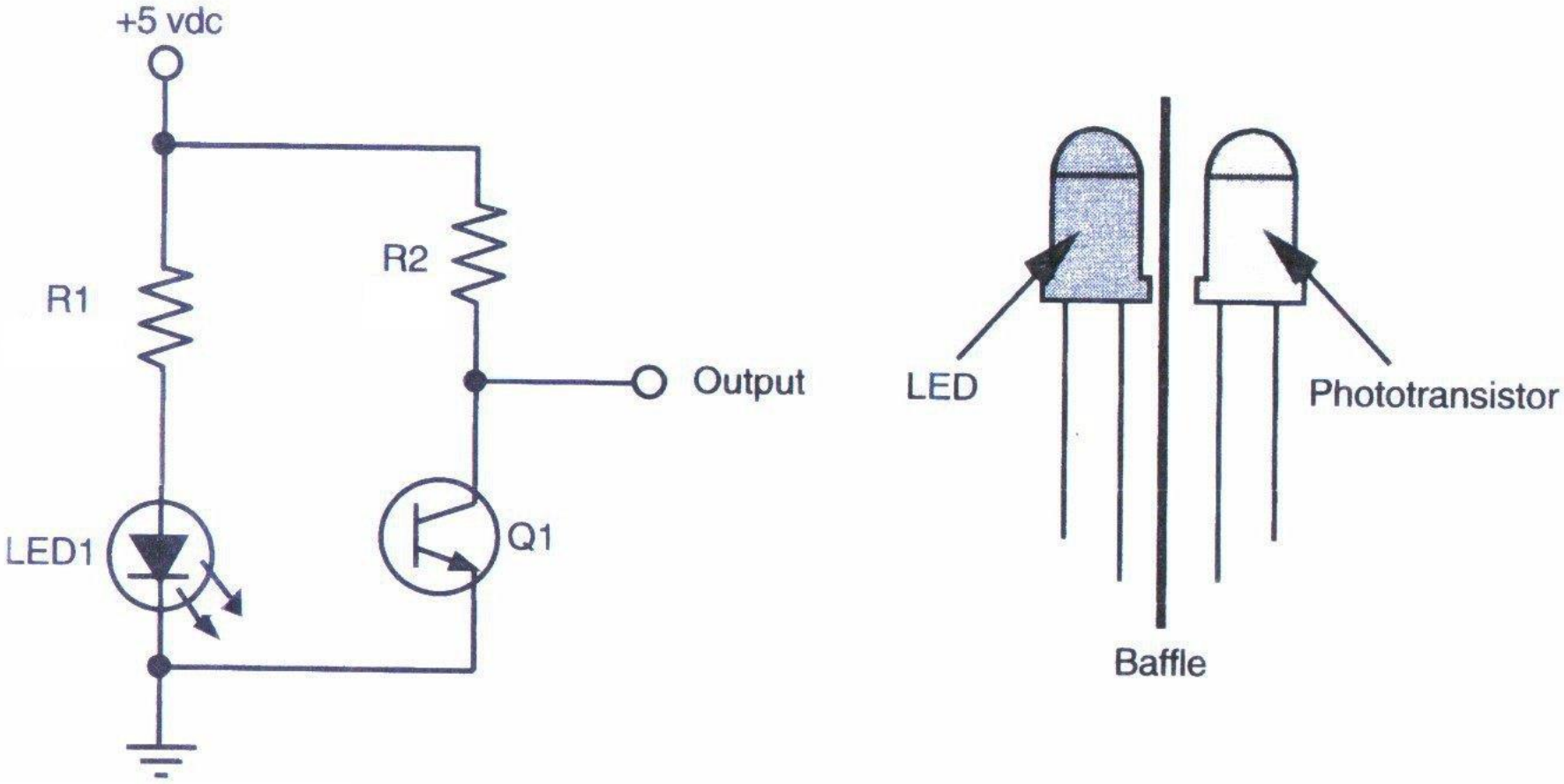
LDR (light dependent resistors)

- Excellent sensitivity between primary colors
- Slow response time
- Easily affected by surrounding light

IR sensors

- Can be used to distinguish between contrastive colours (eg. Black & white)
- Affected by surrounding IR radiation

Using IR sensors

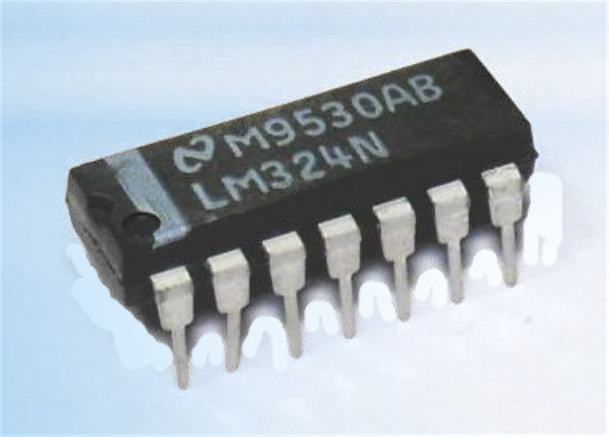


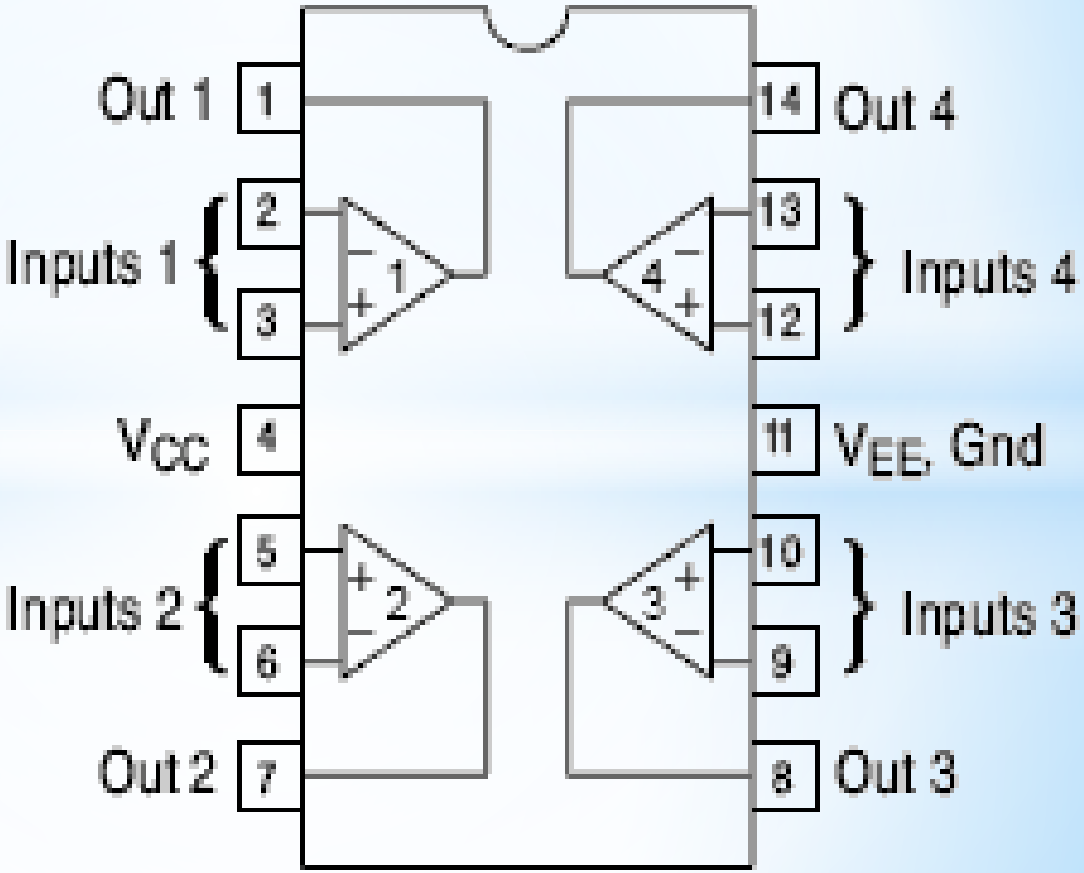
The basic design of the infrared proximity sensor.

Analogue to Digital Conversion

Use op-amps

Ex: LM324

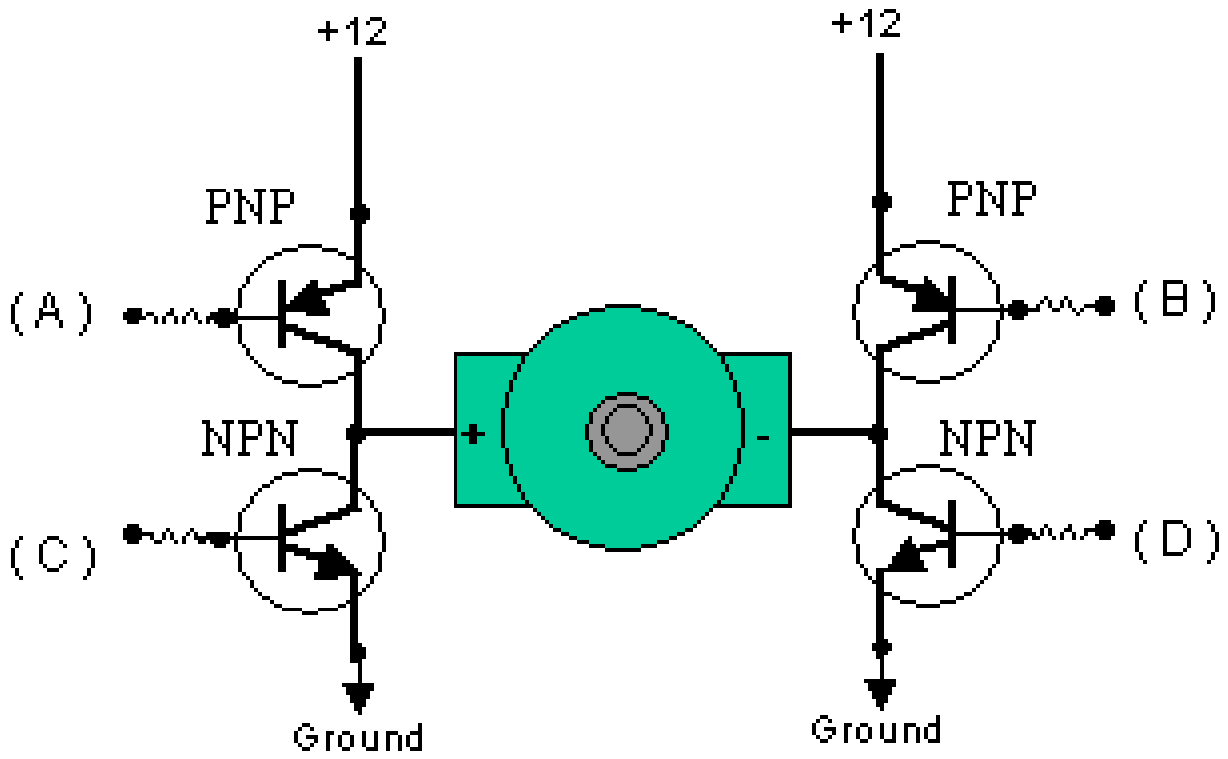




2. Movements

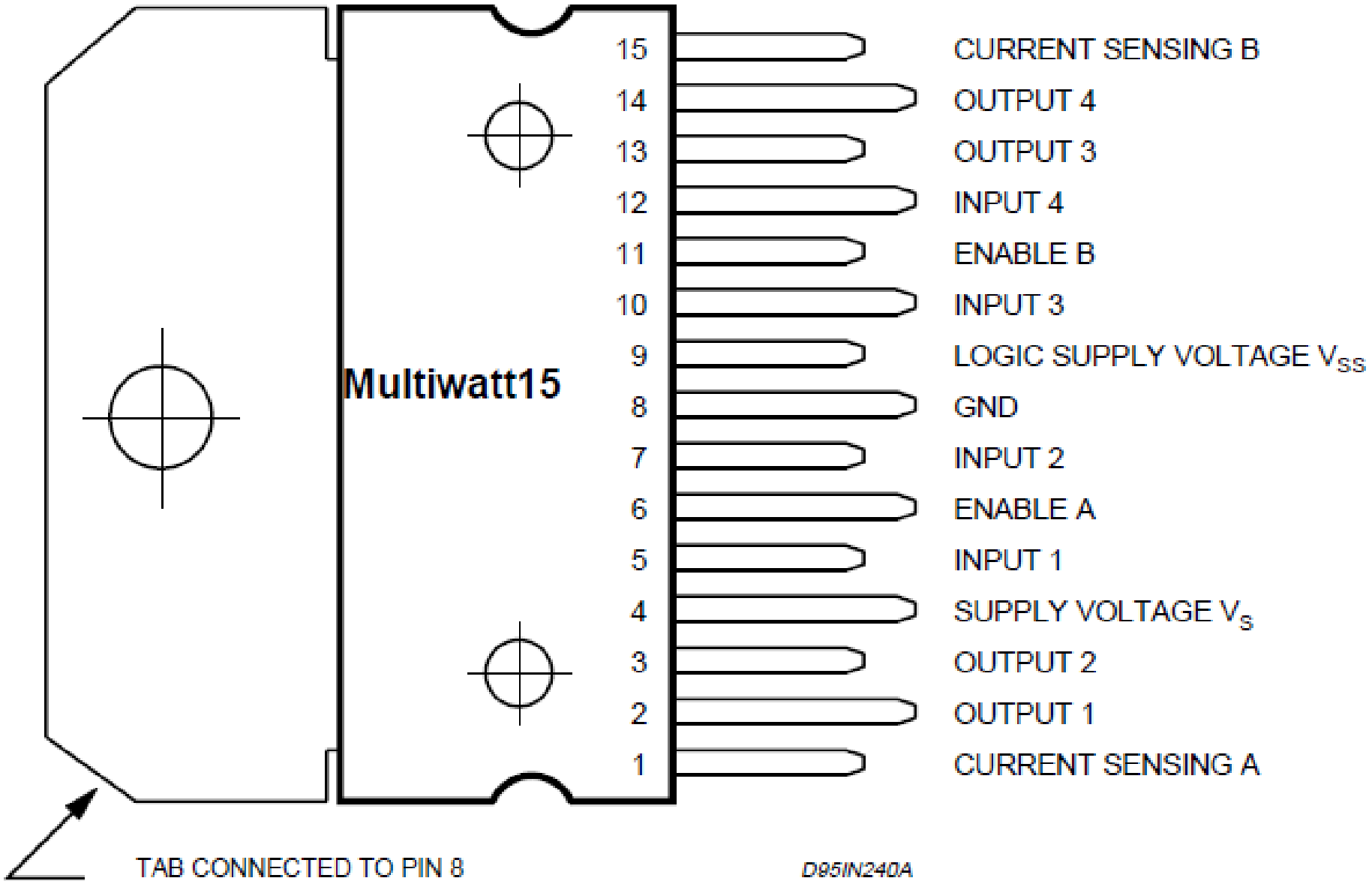
- PIC is the brain. Can only handle logics
- It is not a good practice to power devices through the PIC (most of the time you can't)
- PIC can't handle large currents
- So need drivers / controllers
- Motors absorb larger currents
- So we use motor drivers

Motor Drivers - H Bridge



A	B	C	D	Function
-	-	-	-	--------
1	0	0	1	Forward
0	1	1	0	Reverse
1	1	0	0	Brake
0	0	1	1	Brake

L298


Multiwatt15
15 CURRENT SENSING B
14 OUTPUT 4
13 OUTPUT 3
12 INPUT 4
11 ENABLE B
10 INPUT 3
9 LOGIC SUPPLY VOLTAGE V_{SS}
8 GND
7 INPUT 2
6 ENABLE A
5 INPUT 1
4 SUPPLY VOLTAGE V_S
3 OUTPUT 2
2 OUTPUT 1
1 CURRENT SENSING A
TAB CONNECTED TO PIN 8
D95IN240A

Usage of L298

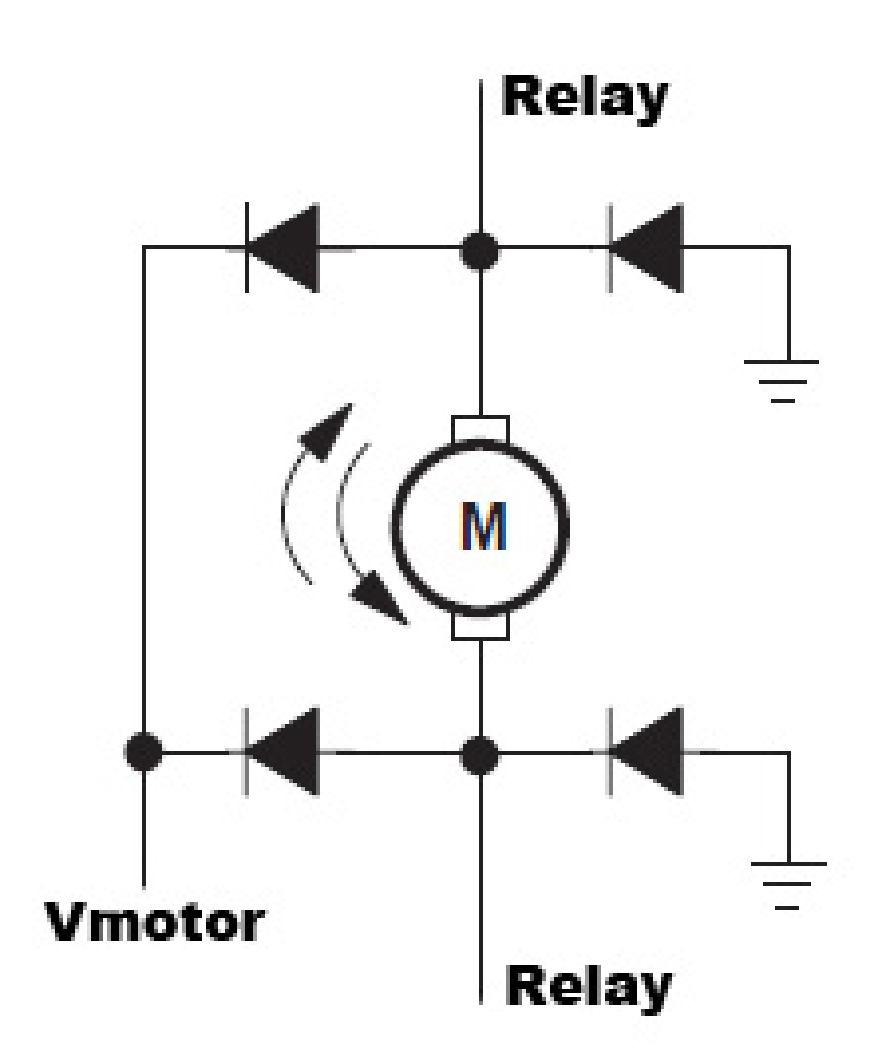
Commonly

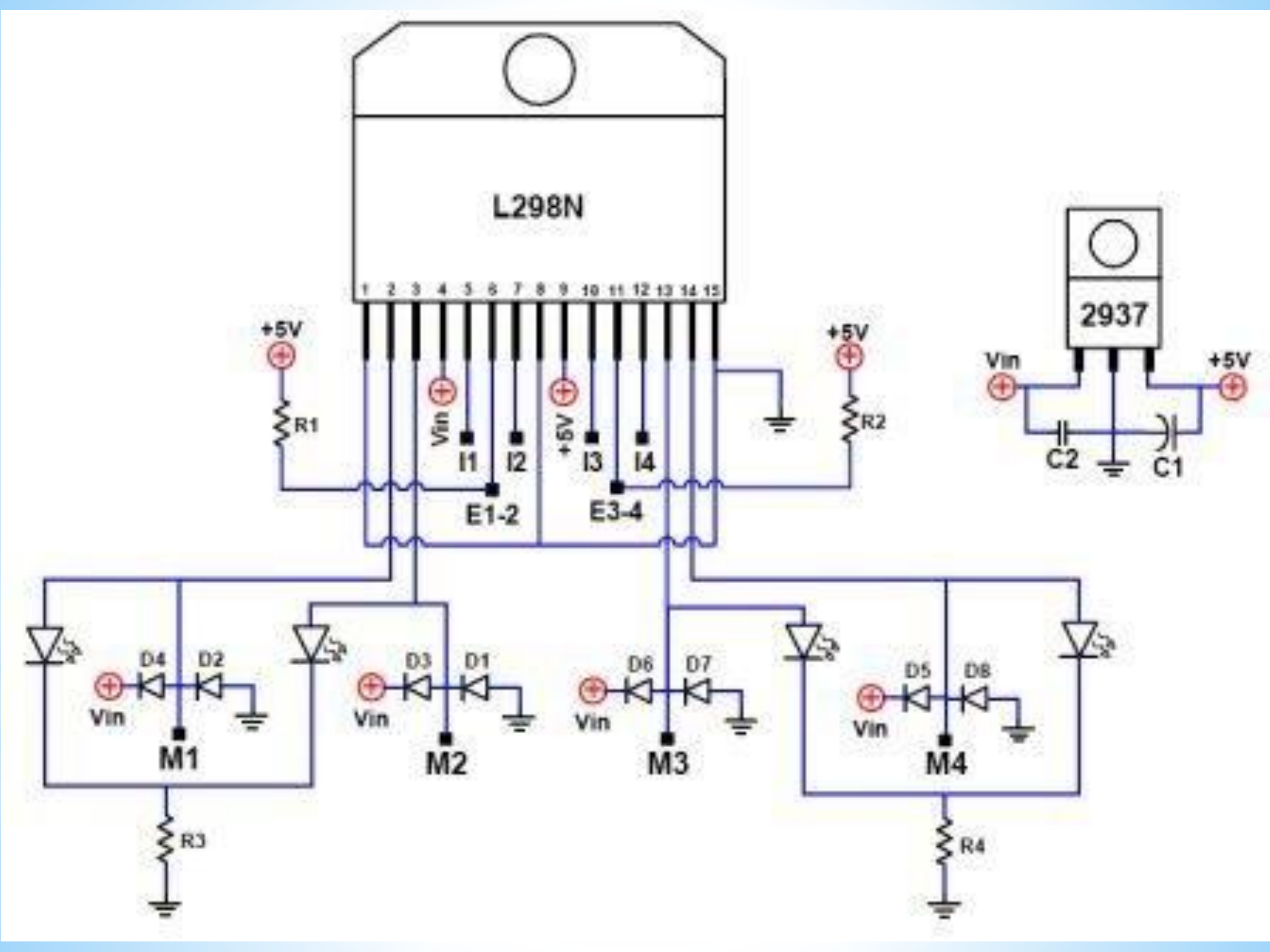
- Logic supply voltage pin should be connected to 5V
- Ground pin should be grounded (common to both supplies)
- Supply voltage pin should be connected to the voltage used to drive the motors

For each motor

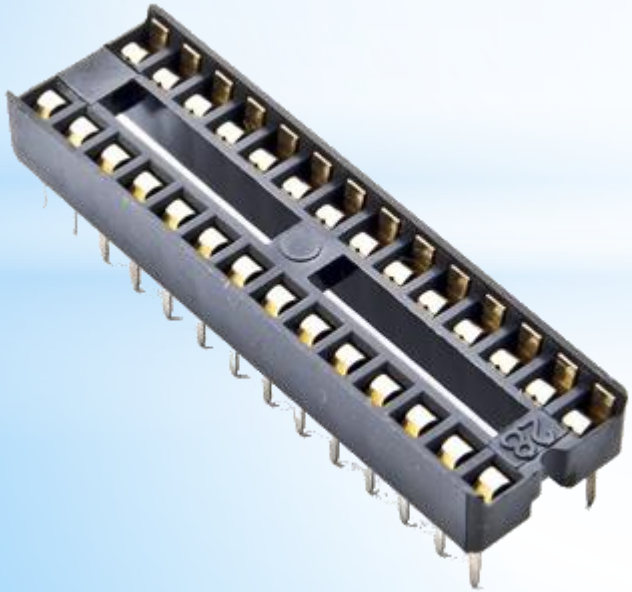
- The 2 input pins (Input 1&2 for motor A & 3&4 for motor B) are used to control the direction.
- The 2 outputs (Output 1&2 for motor A & 3&4 for motor B) should be connected to the motor.
- Current sensing pin should be grounded directly or via a resistor.
- Enable should be given the PWM. (if no speed controlling needed, this pin can be set to high always)

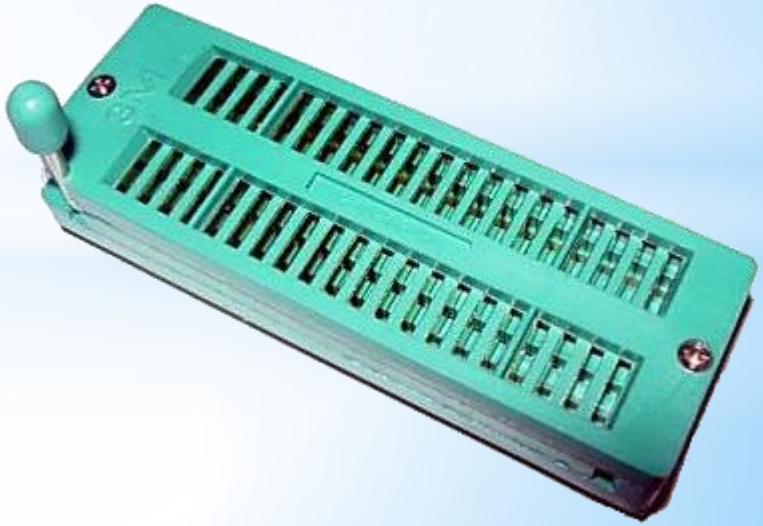
Special precautions that will save your time and money



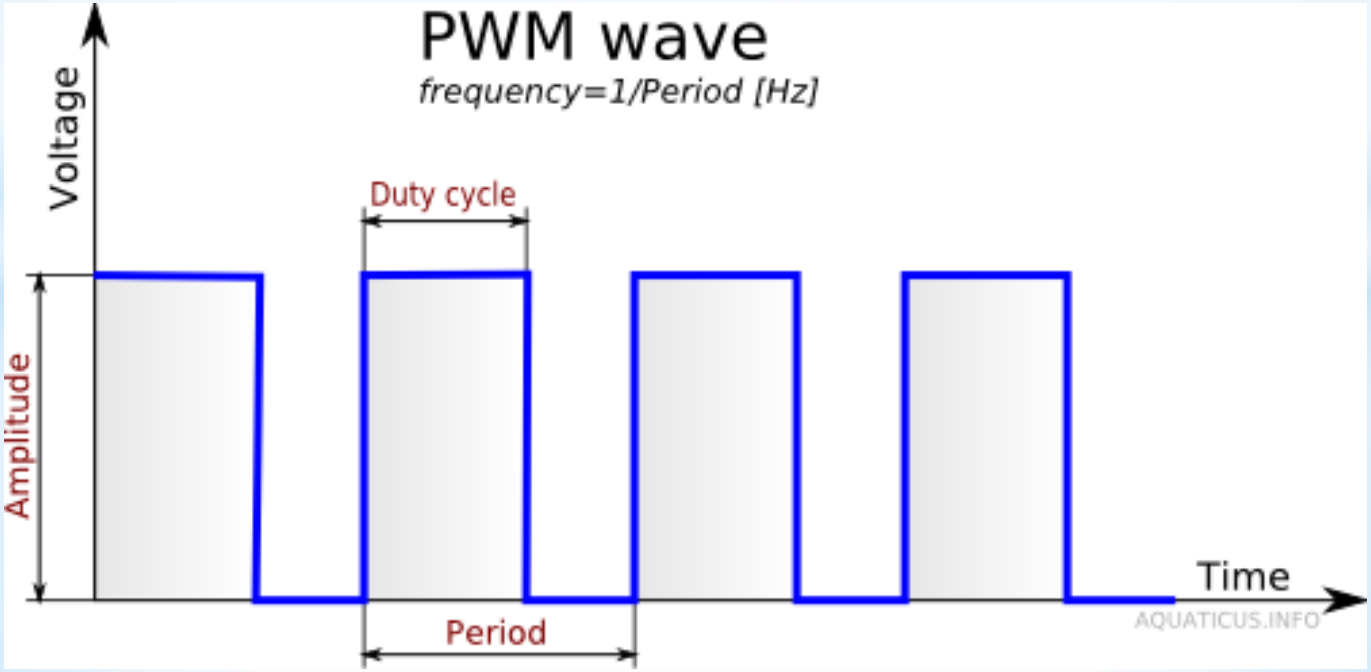

L298N
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
+5V
R1
Vin
I1
I2
+5V
I3
I4
+5V
R2
E1-2
E3-4
2937
Vin
+5V
C2
C1
D4
D2
Vin
M1
D3
D1
Vin
M2
D6
D7
Vin
M3
D5
D8
Vin
M4
R3
R4

- Fly-back diode
- Resistors for inputs
- Use bases for ICs
- Use 2 bases or a ZIF socket to insert the PIC
- If possible design the circuit with an in-circuit progaming facility



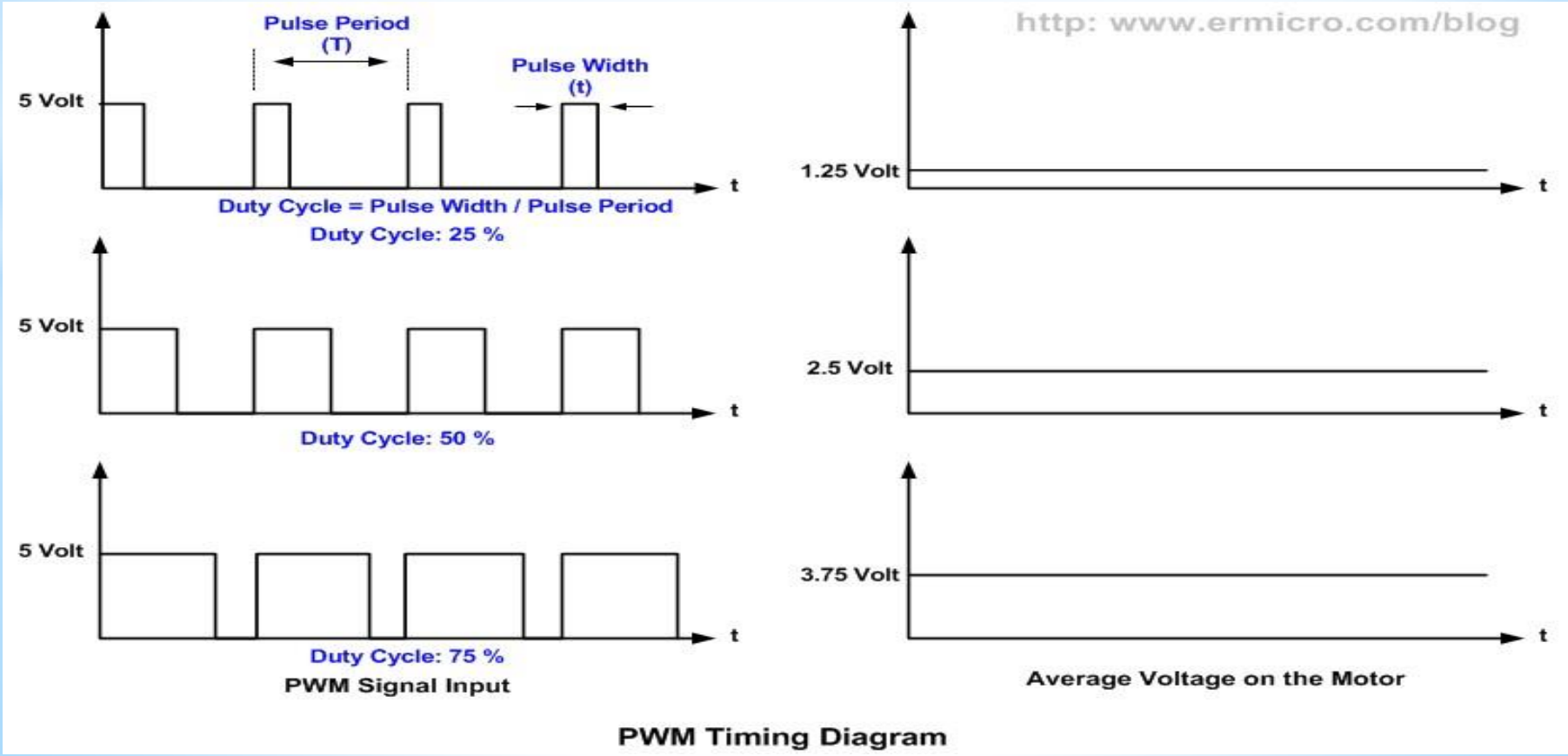


PWM - Pulse Width Modulation

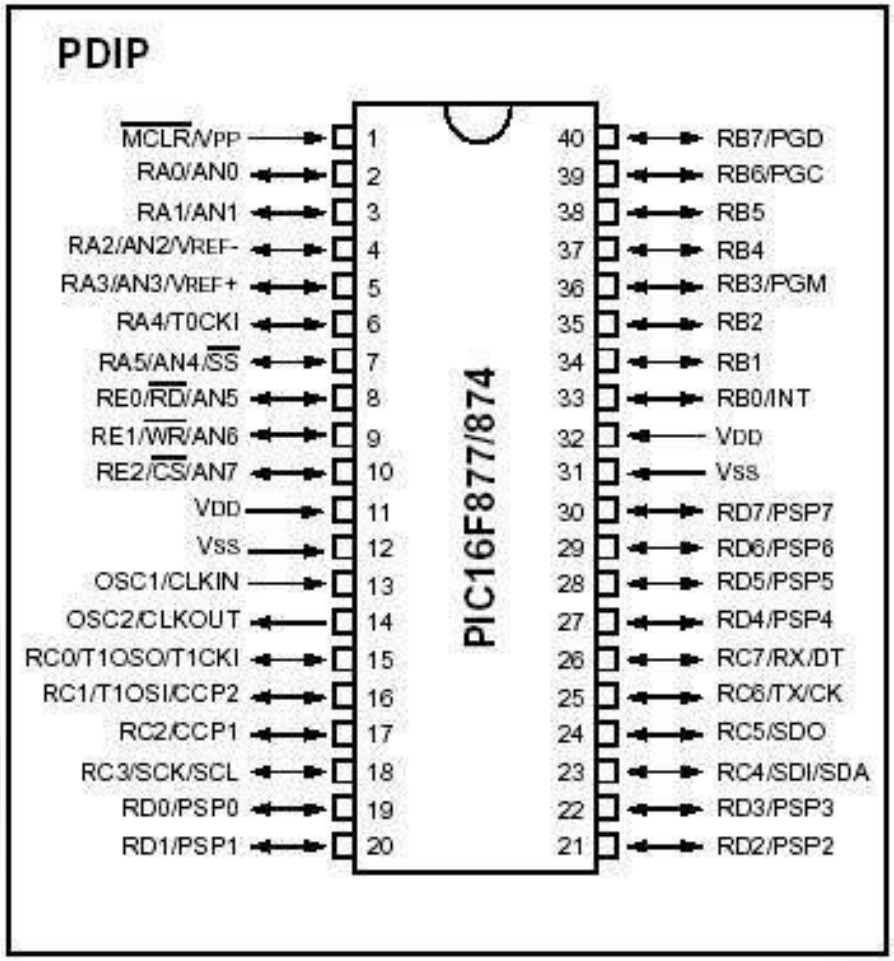


The Principle behind PWM

You can vary average voltage by changing the width of a positive pulse



Use CCP Pins


PDIP
PIC16F877/874
MCLR/VPP 1
RA0/AN0 2
RA1/AN1 3
RA2/AN2/VREF- 4
RA3/AN3/VREF+ 5
RA4/T0CKI 6
RA5/AN4/SS 7
RE0/RD/AN5 8
RE1/WR/AN6 9
RE2/CS/AN7 10
VDD 11
VSS 12
OSC1/CLKIN 13
OSC2/CLKOUT 14
RC0/T1OSO/T1CKI 15
RC1/T1OSI/CCP2 16
RC2/CCP1 17
RC3/SCK/SCL 18
RD0/PSP0 19
RD1/PSP1 20
40 RB7/PGD
39 RB6/PGC
38 RB5
37 RB4
36 RB3/PGM
35 RB2
34 RB1
33 RB0/INT
32 VDD
31 VSS
30 RD7/PSP7
29 RD6/PSP6
28 RD5/PSP5
27 RD4/PSP4
26 RC7/RX/DT
25 RC6/TX/CK
24 RC5/SDO
23 RC4/SDI/SDA
22 RD3/PSP3
21 RD2/PSP2

Programming

1. Initialize PWM

   setup_ccp1(CCP_PWM);

2. Write Value to PWM

   set_pwm1_duty(value);

   0 <= value <= 255

Sample Programme

```
void main{
        int i;
        setup_ccp1(CCP_PWM);  // Configure CCP1 as a PWM
        While(true){
                for(i=0;i<=255;i++){
                        set_pwm1_duty(i);
                        delay_us(100);
                }
        }
}
```

EEPROM - Electrically Erasable Programmable Read-Only Memory

- Non-volatile Memory
- Data will remain even after power off

- Read From EEPROM

  read_eeprom (***address***)

  read_eeprom (***address***,***N***)

N – Number of Bytes to read

By default the function reads a word from EEPROM at the specified address.

- Write to EEPROM

  write_eeprom (***address***, ***value***)

value - a constant or variable to write to EEPROM

Interrupts

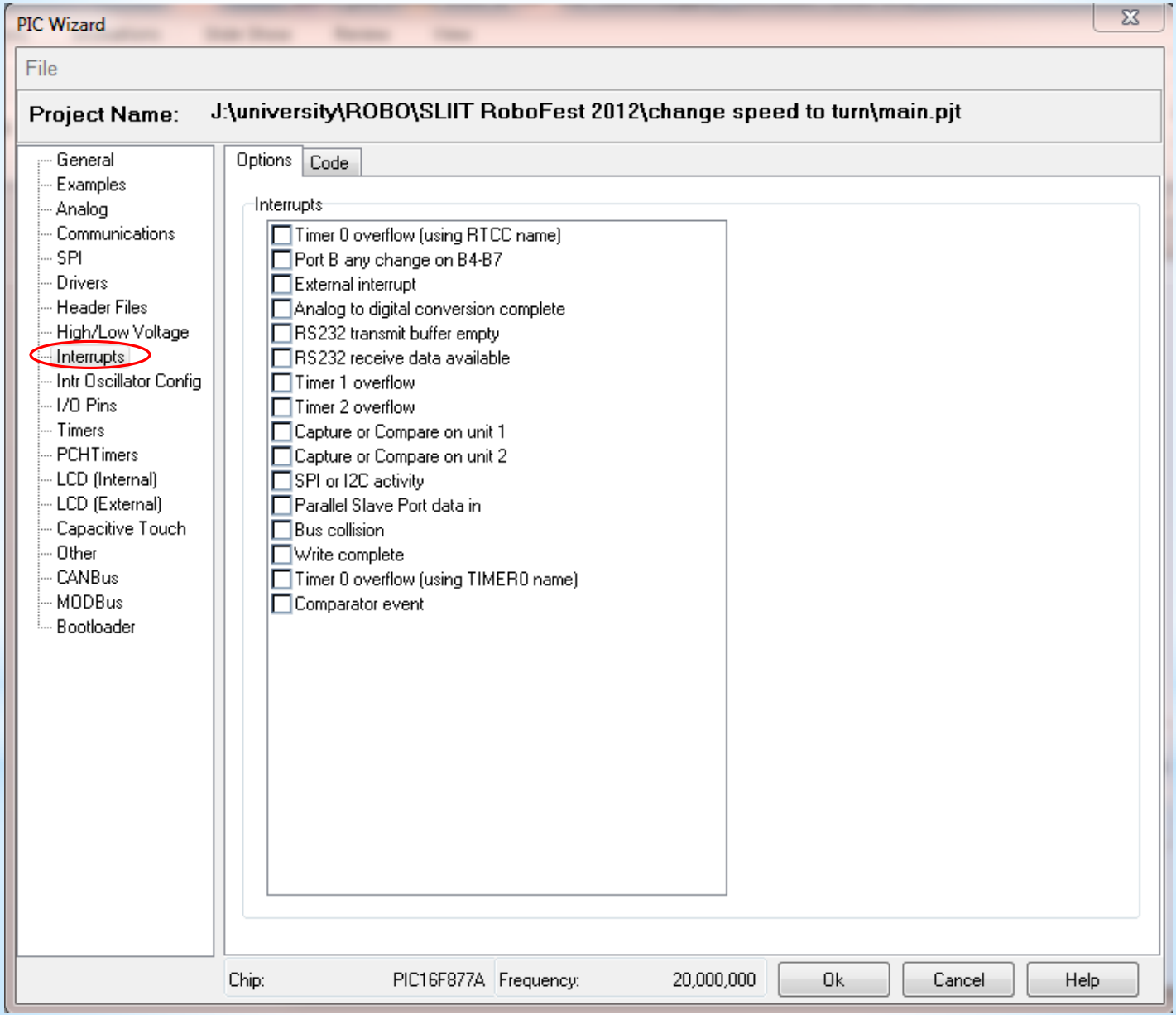
- Interrupts are a mechanism of a microcontroller which enables it to respond to some events at the moment when they occur, regardless of what microcontroller is doing at the time.

- Each interrupt changes the program flow, interrupts it and after executing an interrupt subprogram (interrupt routine) it continues on from that same point.

Examples

- External interrupt on RB0/INT pin of microcontroller
- Interrupt during a TMR0 counter overflow
- Interrupt upon finishing write-subroutine to EEPROM

How to add Interrupts to your programme



Timer Interrupt Example

```
#include <abcd.h>
#int_RTCC
void  RTCC_isr(void)
{
        //do something
}
void main()
{
setup_timer_0(RTCC_EXT_L_TO_H|RTCC_DIV_1|RTCC_8_bit);
                                                //51.2 us overflow
   enable_interrupts(INT_RTCC);
   enable_interrupts(GLOBAL);
}
```

- Disabling Interrupt

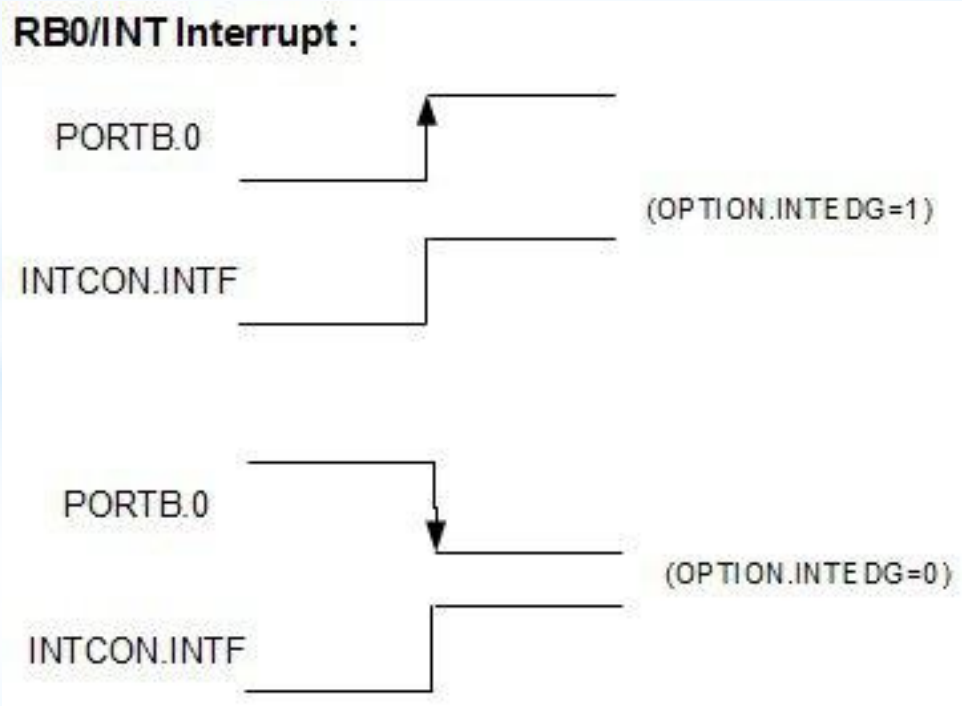
```
disable_interrupts(GLOBAL);
disable_interrupts(INT_RTCC);
```

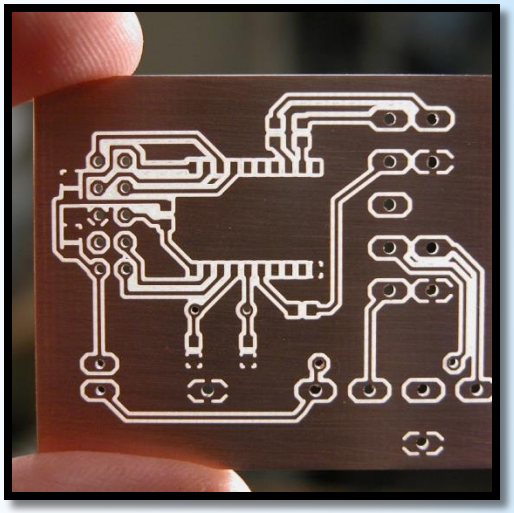
External interrupt on RB0/INT pin of microcontroller

enable_interrupts(INTR_CN_PIN|Pin_B0); or

enable_interrupts(INT_EXT);

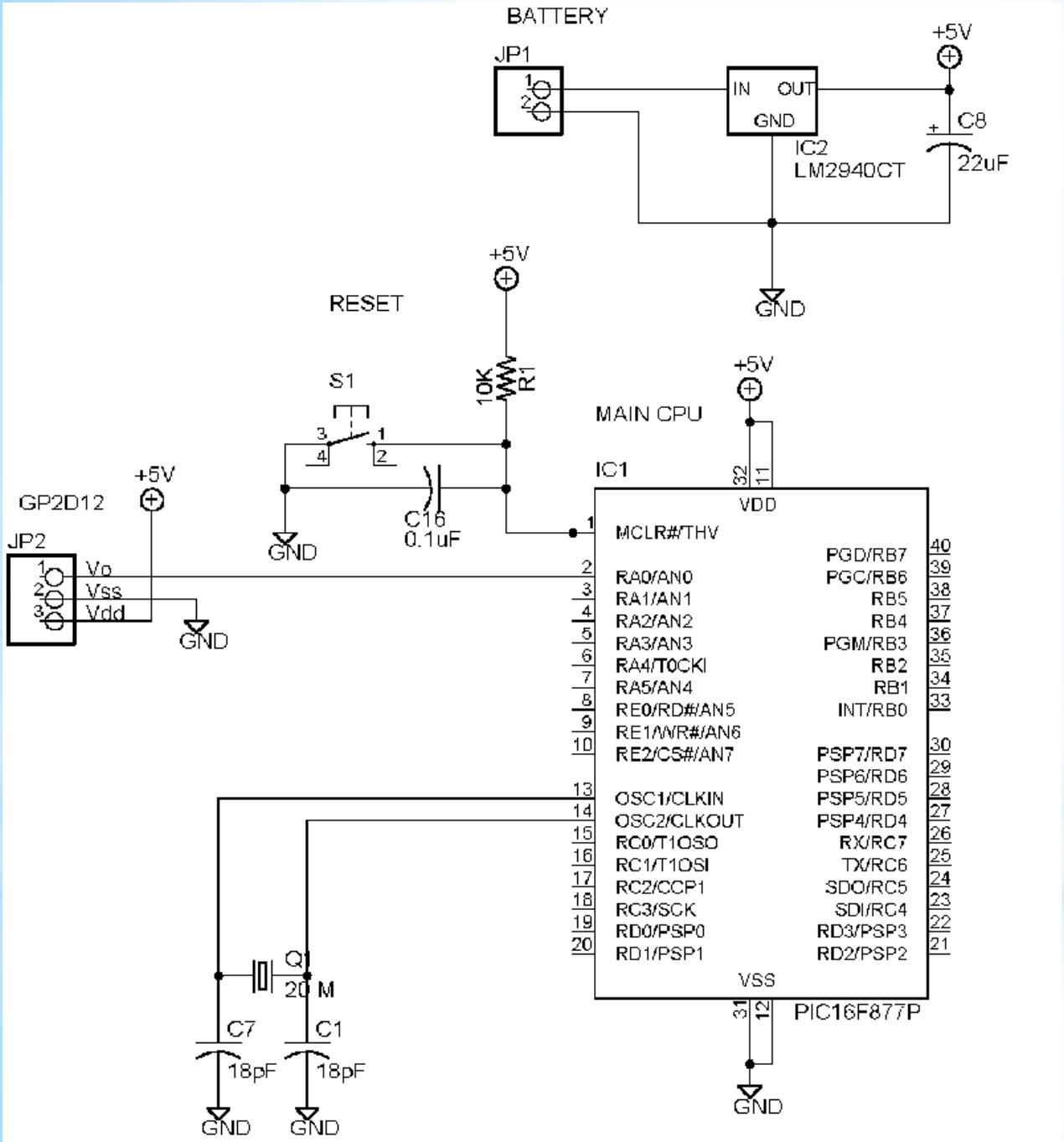
- Interrupt can be triggered on rising edge or falling edge





Create your own PCB

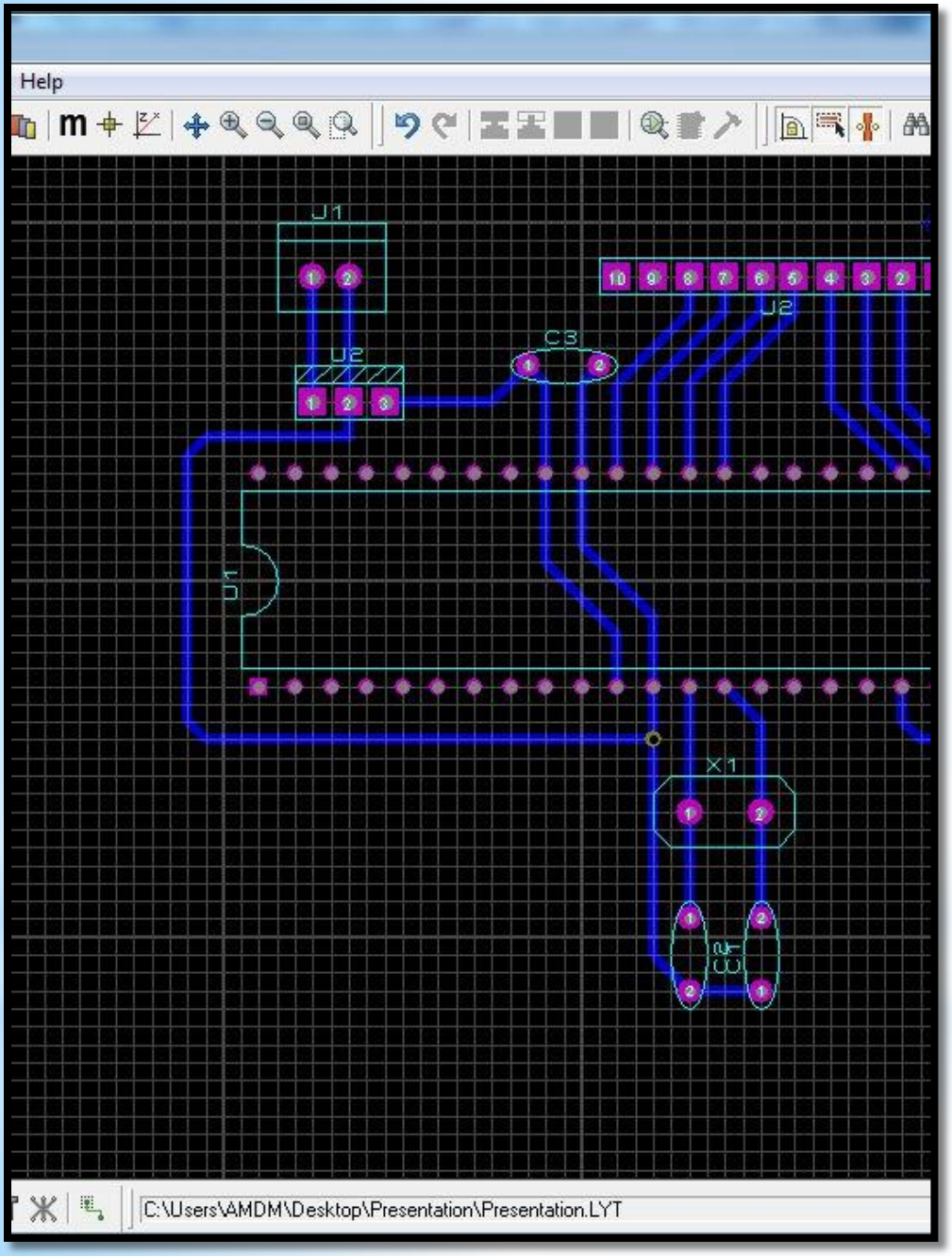
PCB designing and printing techniques



Lets design this circuit

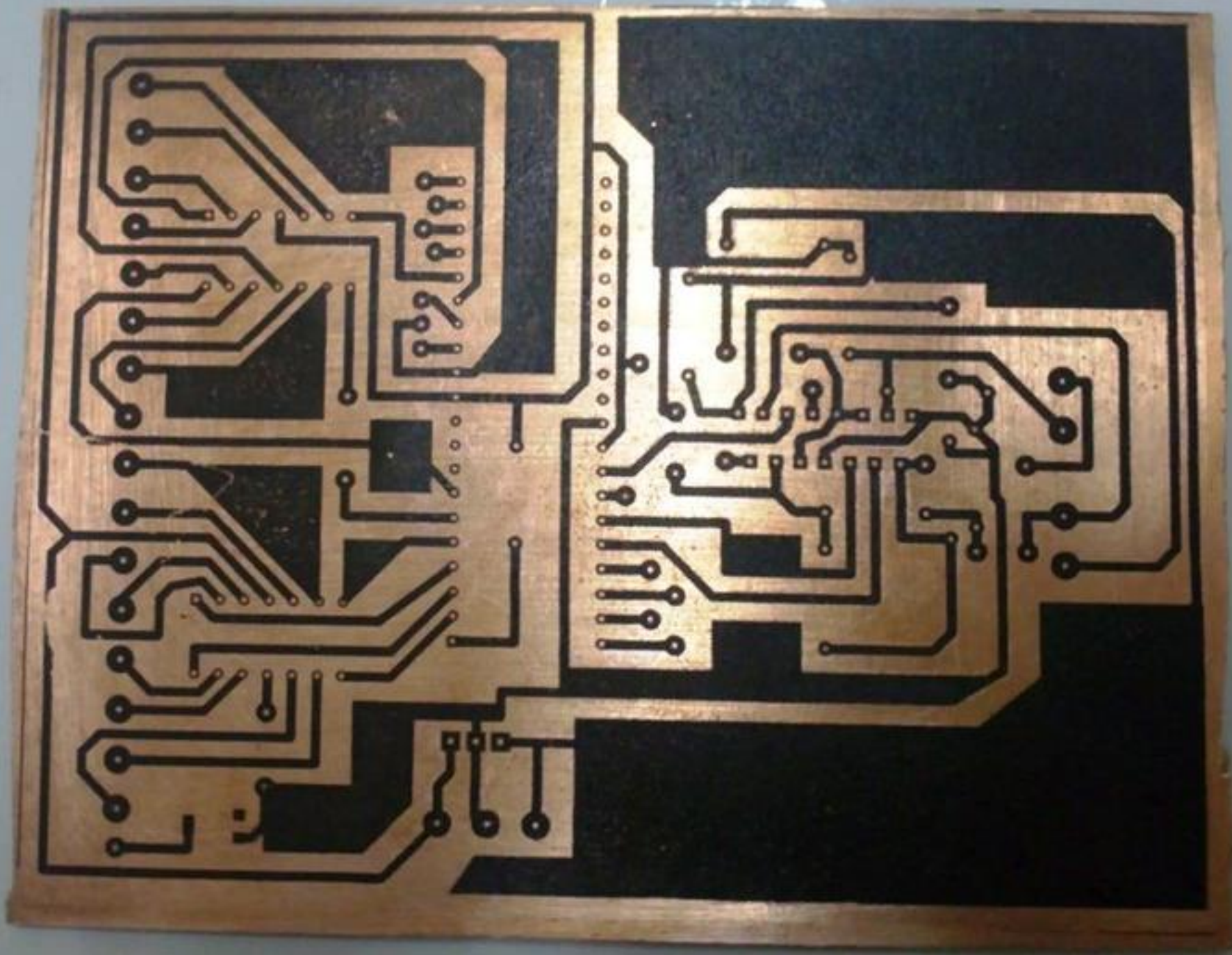
Device List

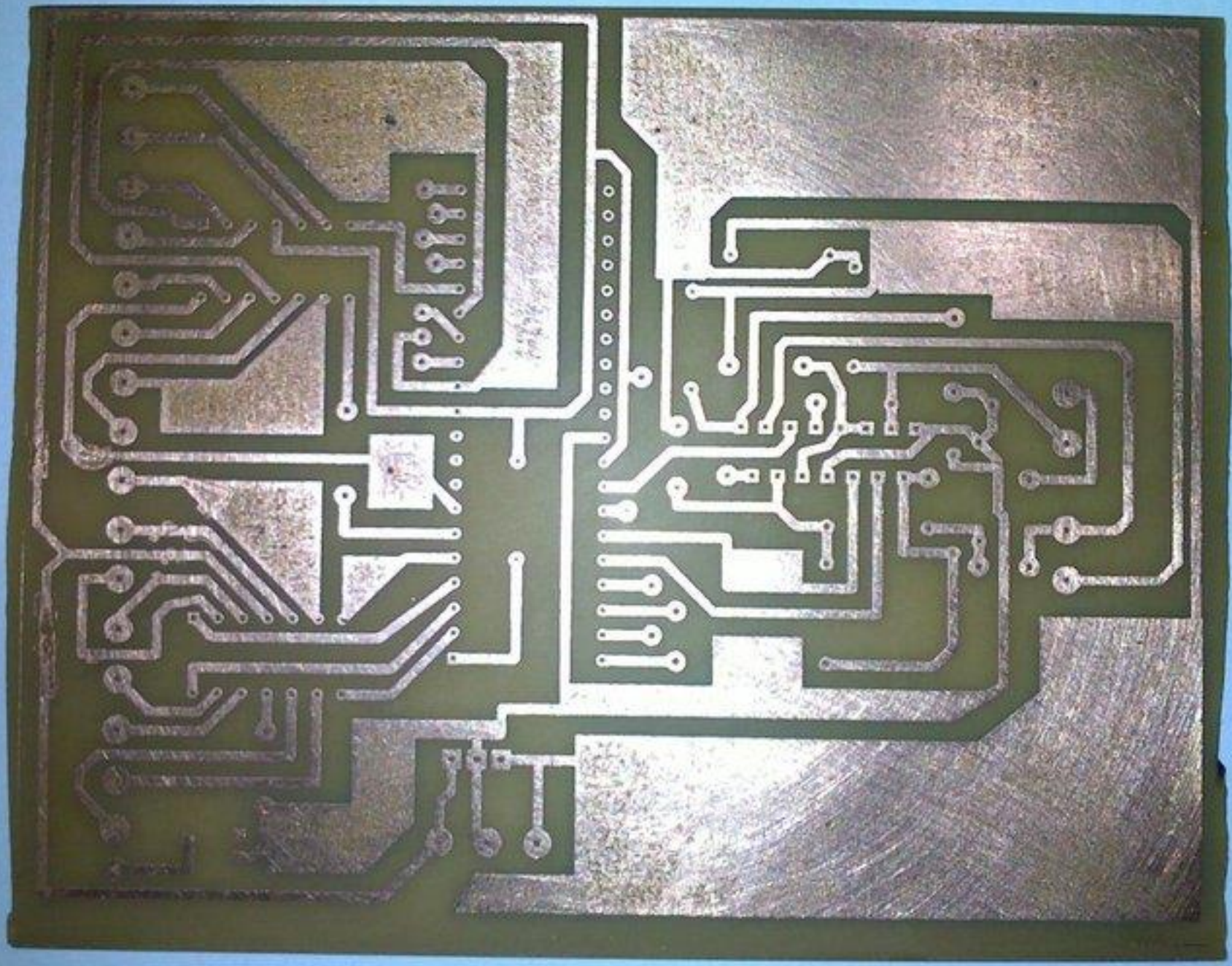
1. PIC 16F877
2. Crystal
3. Ceramic Capacitor – (CEREMIC22N)
4. 7805
5. Connector – (SIL-100-02)
6. Connector – (CONN-H10)



Lets go to ARES to design the PCB layout

- Draw Paths Manually
- Auto Routing
- Exporting
- Printing





Any Questions?

Ask your questions
or
Answer our questions…