LED\_on\_and\_off

TABLE OF CONTENTS:

* Abstract
* Introduction
* Goal
* Materials and Methodology
  + - Power Supply
    - Resistor
    - Diode
    - Light Emitting Diode(LED)
    - Connector
    - HOLTEK MCU
    - Microcontroller
* Software
* Advantages and Limitations
* Conclusion
* Abstraction

In the project LEDs present many advantages over incandescent light sources including lower energy consumption, longer life time, improved robustness, smaller size, faster switching easy installation application, friendly environment, and greater reliability. We designed our LED light scheme with advanced technology which can replace directly the conventional bulbs such as incandescent bulbs, lamppost, sodium as a rod light.

* Introduction

This project is based on microcontroller for the lightning scheme of dancing led from the Website www.instructables.com it was designed by “CHR” I’ve changed this design to fit my purpose and materials while still being able to keep the same.

* Goal

--To know microcontroller and it’s application.

--Create a similar dancing led to the one from intractable.

--Program different behavior.

--Make it look as nice as possible.

* Material and Methodology
* Power supply

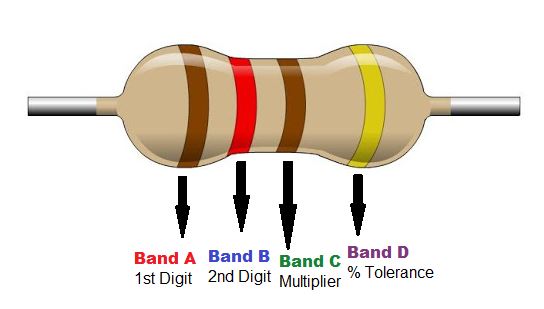
Make sure the total wattage of your LED strip lighting is **at least 20 percent less than the power supply's rated wattage**. For example, if you have an LED strip light that requires 80 watts of power to run, you'll need a power supply that's rated for at least 96 watts.

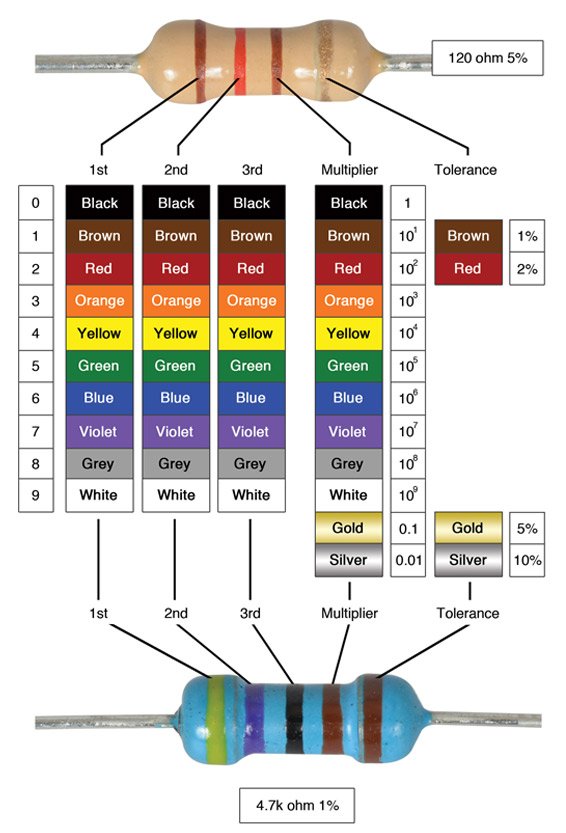


There are two things worth attention concerning the microcontroller power supply circuit. Brown-out is a potentially dangerous state which occurs at the moment the microcontroller is being turned off or in situations when powersupply voltage drops to the limit due to electric noise. As the microcontroller consists of several circuits which have different operating voltage levels, this state can cause its out-of-control performance. In order to prevent it, themicrocontroller usually has built-in circuit for brown out reset. This circuit immediately resets the whole electronicswhen the voltage level drops below the limit. Reset pin is usually marked as MCLR (Master Clear Reset) and serves for external reset of the microcontroller byapplying logic zero (0) or one (1), depending on type of the microcontroller. In case the brown out circuit is not builtin, a simple external circuit for brown out reset can be connected to this pin.

* Resistor

A resistor is a component that ‘resists’ the flow of electricity. The flow of electricity is called current. Every resistor has a certain value telling how much it resists the flow. This resistance is called ohm, it’s most commonly shown with the omega Ω. These resistors are needed on the breadboard to prevent a led from getting to much current. This would kill the LED. It’s the same case with the Infrared LED.

 Resistors are color coded. One gold, silver or blank ring indicating the tolerance the resistor can differ froth given value. And 3 rings which Together make the value in ohm. The fifth ring isn’t used to often. Each color is a number varying from 1 to 9. See the table on the right.

For

example when a resistor is in the above case blue, red, red it means the first digit is “6” the second digit is “2” and it must be multiplied by “2 x 10” or a hundred which would make it 6200 Ω. Once such a resistor is made it will not change its value. The only thing which could happen is it gets burned when its limit is overridden by supplying a high enough voltage killing the entire thing. When this happens the resistor won’t do anything anymore and can 25 Three Dimensional Display be thrown away.

* Diode

**Diode**, an [electrical](https://www.britannica.com/science/electricity) component that allows the flow of [current](https://www.britannica.com/science/electric-current) in only one direction. In [circuit](https://www.britannica.com/technology/electric-circuit) diagrams, a diode a triangle with a line across one vertex.

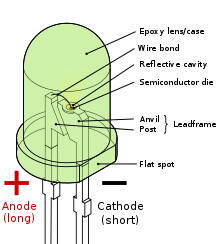
File:Zener diode symbol.svg

The most common type of diode uses a [p-n junction](https://www.britannica.com/technology/p-n-junction). In this type of diode, one material (n) in which [electrons](https://www.britannica.com/science/electron) are charge carriers abuts a second material (p) in which [holes](https://www.britannica.com/science/hole-solid-state-physics) (places depleted of electrons that act as positively charged particles) act as charge carriers. At their interface, a depletion region is formed across which electrons diffuse to fill holes in the p-side.

This stops the further flow of electrons. When this junction is forward [biased](https://www.merriam-webster.com/dictionary/biased) (that is, a positive voltage is applied to the p-side), electrons can easily move across the junction to fill the holes, and a current flows through the diode. When the junction is reverse biased (that is, a negative voltage is applied to the p-side), the depletion region widens and electrons cannot easily move across. The current remains very small until a certain voltage (the breakdown voltage) is reached and the current suddenly increases.

* LED(Light emiting diode)

Light emitting diodes are diodes which produce light when current passes through in the right direction. Just like any diode the other direction is completely blocked. Led’s are nowadays commonly used as light sourcesbecause they are smaller, very efficient and have a longer lifetime.



* Connector

Well wires, these are cables that let through electricity.

This can be as a power supply or as a matter of signals. This cable in the picture is the 10 pins connecter used to connect the USBasp with the 10 pins connecter on the protoboar.

* HOLTEK MCU

HT-IDE3000 The HT-IDE3000 is a simulation software specially developed for HOLTEK 8-bit MCUs. When used together with OCDS EV chip, it will implement a series of operations such as single step, full speed, breakpoint setting, RAM monitor, etc. Refer to the HT-IDE3000 User’s Guide for more information. <http://www.holtek.com.tw/documents/10179/106680/HT-IDE3000UsersGuide%281252%29.pdf?0113>

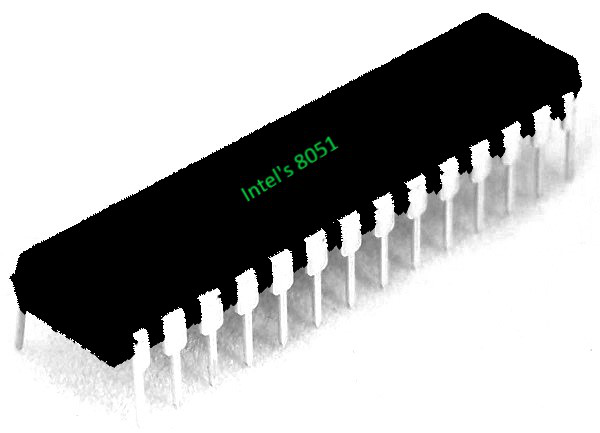
HOPE3000 The HOPE3000 is a programming software specially designed for HOLTEK 8-bit MCUs. The software includes erase, blank check, program, verify, etc. Refer to the e-Writer Series Writer User’s Guide for more information. <http://www.holtek.com.tw/documents/10179/106680/e-Writer+Series+Writer+User%27s+Guide+%28English%29.pdf>



HOPE3000 For e-Link The HOPE3000 For e-Link is a programming software specially designed for HOLTEK 8-bit MCUs. The software includes erase, blank check, program, verify, etc. Refer to the Holtek e-Link ICP User’s Guide for more information. <http://www.holtek.com.tw/documents/10179/106680/e-Link+ICP+User%27s+Guide_ENv100--20150513.pdf>.

* Microcontroller

Microcontrollers are small computers in a small package. They contain several things such as a processor, ram and programmable memory and most last but not least inputs and outputs.



Microcontrollers are used in most automatically controlled devices e.g. microwaves, cellphones and alarms. They are very convenient to use for products produced in huge quantities because they can be programmed very quickly. And they are very small compared to previous techniques in which processor ram and programmable memory are separated. These computers in a chip are nowadays also used in mp3 players in which another function is very useful, this function is the sleep function, and microcontrollers are able to wake up immediately on the touch of a button. Microcontroller can be termed as a single on chip computer which includes number .

* Software

A lot of you may heard about microcontrollers and its applications. Well it is a bit difficult to start learning microcontrollers. And the guides and tutorials also do not start from zero level which makes learning far more difficult than anticipated. I have tried to start from zero level in here also. All you need is the simplest knowledge of electronics or digital circuits. But you surely should have a decent knowledge of C language.

The software for programming Microchip microcontrollers comes by the name MPLAB IDE. MPLAB is the name of the software and IDE stands for Integrated Development Environment. IDE means that the software itself has all or most of the features that is needed. MPLAB IDE can perform the following operations :

* Source code editing
* Project management
* Machine code generation from Assembly
* Device simulation
* Device emulation
* Device programming

Hence it is called an Integrated Development Environment.

Some things should be clear before starting the tutorial. We can write programs for the PIC microcontrollers in MPLAB but the syntax and code depends upon the library, which may be different from family to family and even a lot of libraries are available for the same family of microcontrollers too.

* Source code

#include "HT68F0021.h"

#define INIT\_LED1() (\_pac0 = \_pa0 = 0)

#define SET\_LED1(X) (\_pa0 = (X))

#define INIT\_LED2() (\_pac1 = \_pa1 = 0)

#define SET\_LED2(X) (\_pa1 = (X))

#define INIT\_LED3() (\_pac7 = \_pa7 = 0)

#define SET\_LED3(X) (\_pa7 = (X))

#define INIT\_LED4() (\_pac2 = \_pa2 = 0)

#define SET\_LED4(X) (\_pa2 = (X))

#define INIT\_LED5() (\_pac6 = \_pa6 = 0)

#define SET\_LED5(X) (\_pa6 = (X))

#define RESET\_LEDS() (\_pa0 = \_pa1 = \_pa7 = \_pa2 = \_pa6 = 0)

#define INT\_ISR\_ENABLE() (\_innate = 1)

#define INT\_ISR\_DISABLE() (\_inte = 0)

#define INT\_SET\_FLAG() (\_intf = 1)

#define INT\_CLEAR\_FLAG() (\_intf = 0)

#define EMI\_ENABLE() (\_emi = 1)

#define EMI\_DISABLE() (\_emi = 0)

#define INIT\_SWITCH() (\_pac5 = \_papu5 = 1)

unsigned char state;

unsigned char got\_intt;

volatile int timer\_cnt;

volatile int prev\_timer\_cnt;

volatile int button\_down;

volatile int sleep;

#pragma vector isr\_4 @ 0x04

void isr\_4(){

//\_delay(65000);

if(\_pa5 == 0) {

prev\_timer\_cnt = timer\_cnt = -1;

button\_down = 1;

}/\* else if(timer\_cnt - prev\_timer\_cnt > 3 && button\_down == 1){

button\_down = 0;

prev\_timer\_cnt = 0;

state = 6;

got\_intt = 1;

timer\_cnt = 0;

}\*/ else {

button\_down = 0;

if(sleep == 0){

got\_intt = 1;

} else {

sleep = 0;

}

prev\_timer\_cnt = timer\_cnt = -1;

}

} // external ISR

#pragma vector isr\_c @ 0x0C

void isr\_c(){

timer\_cnt ++;

if((timer\_cnt >= 300) || ((timer\_cnt > 2 && timer\_cnt < 10) && (\_pa5 == 0))) {

sleep = 1;

state = 6;

got\_intt = 1;

timer\_cnt = -1;

} else {

button\_down = 0;

}

} // timer isr

void mydelay(unsigned int times){

while(times--) \_delay(65000);

}

unsigned char leds[5];

void soft\_pwm(unsigned int l) {

unsigned char i,j;

for(i=0; i<l; i++){

for(j=0; j<255; j++){

SET\_LED1(leds[0] > j);

SET\_LED2(leds[1] > j);

SET\_LED3(leds[2] > j);

SET\_LED4(leds[3] > j);

SET\_LED5(leds[4] > j);

if(got\_intt != 0) {

return;

}

//\_delay(65000);

}

}

}

void main()

{

unsigned char leds2[5] = {0,0,0,0,0};

unsigned char step = 0;

unsigned char i = 0;

unsigned char j = 0;

unsigned char k = 0;

unsigned char stretch = 0;

unsigned char step5\_seq[6] = {36, 42, 49, 17, 10, 4};

INIT\_LED1();

INIT\_LED2();

INIT\_LED3();

INIT\_LED4();

INIT\_LED5();

//\_ints1 = 0; \_ints0 = 1; //rising edge trigger external interrupt

\_ints1 = 1; \_ints0 = 1; //rising edge and falling edge triggers external interrupt

\_intps = 0;

INT\_CLEAR\_FLAG();

INT\_ISR\_ENABLE();

EMI\_ENABLE();

\_intf = 0;

\_inte = 1;

\_emi = 1;

//////////////////////// Timer registers ///////////////////////////////////////////

\_tmr = 0;

// interrupt registers

\_emi = 1;

//\_tbe = 1;

\_te = 1;

// TMRC register

\_tm1 = 1;

\_tm0 = 0;

\_ton = 0; // disable timer

\_tpsc0 = 1;

\_tpsc1 = 1;

\_tpsc2 = 1;

\_ton = 1;

//////////////////////// Timer registers ///////////////////////////////////////////

INIT\_SWITCH();

\_wdtc = 0xA8;

//\_pas04 = 0;

//\_pawu5 = 0;

//\_inte = 0;

state = 0;

got\_intt = 0;

prev\_timer\_cnt = -1;

button\_down = 0;

timer\_cnt = -1;

sleep = 0;

RESET\_LEDS();

for(step = 0; step < 5; step ++){

leds[step] = 0;

}

while(1) {

if(got\_intt != 0){

mydelay(20);

state += 1;

if(state > 5) {

state = 0;

}

RESET\_LEDS();

for(step = 0; step < 5; step ++){

leds[step] = 0;

}

i = j = 0;

stretch = 1;

got\_intt = 0;

// button\_down = 0;

}

switch(state) {

case 0:

RESET\_LEDS();

break;

case 1:

//soft\_pwm(&leds);

//mydelay(2);

SET\_LED1(1);

SET\_LED2(1);

SET\_LED3(1);

SET\_LED4(1);

SET\_LED5(1);

break;

case 2:

stretch = 3;

case 3:

if(state == 3) {

stretch = 1;

}

for(k = 0; k < 248; k += 4){

for(i = 0; i < 5; i++) {

leds[i] = k;

}

soft\_pwm(stretch);

if(got\_intt != 0){

RESET\_LEDS();

break;

}

}

for(i = 0; i < 5; i++) {

leds[i] = k;

}

soft\_pwm(stretch);

for(k = 255; k > 5; k -= 4){

for(i = 0; i < 5; i++) {

leds[i] = k;

}

soft\_pwm(stretch);

if(got\_intt != 0){

RESET\_LEDS();

break;

}

}

break;

case 4:

for(k = 0; k < 240; k += 10){

leds[i] = k;

if(j != i) {

leds[j] = 255 - k;

}

soft\_pwm(1);

if(got\_intt != 0){

break;

}

}

if(got\_intt != 0){

break;

}

if(j != i) {

leds[j] = 0;

}

leds[i] = 255;

soft\_pwm(100);

j = i++;

if(i >= 5) {

i = 0;

}

break;

case 5:

for(k=0; k < 6; k++) {

j = (step5\_seq[k] & 32) >> 5;

for(i = 0; i <= 253; i += 2) {

for(step = 0; step < 5; step ++){

if((step5\_seq[k] & (1 << step)) > 0) {

leds[step] = (j == 1) ? i : (255 - i);

}

}

soft\_pwm(1);

if(got\_intt != 0){

break;

}

}

for(step = 0; step < 5; step ++){

if((step5\_seq[k] & (1 << step)) > 0) {

leds[step] = (j == 1) ? 255 : 0;

}

}

soft\_pwm(5);

if(got\_intt != 0){

break;

}

}

break;

}

}

* ADVANTAGES AND LIMITATIONS
* ADVANTAGES

**Advantages of the microcontroller :**

* Low time required for performing operation.
* It is easy to use, troubleshooting and system maintenance is straightforward.
* At an equivalent time, many tasks are often performed therefore the human effect are often saved.
* Processor chip is extremely small and adaptability occurs.
* Cost and size of the system is less.
* Microcontroller is straightforward to interface additional RAM, ROM, and I/O port.
* Once microcontroller is programmed then they can’t be reprogrammed.
* If the digital parts were not present it will look like a microcomputer.
* It is easy to use, troubleshooting and system maintaining is straightforward.
* **LIMITATIONS of the microcontroller**
* It is generally utilized in micro equipment.
* It has a complex structure.
* Microcontroller cannot interface a better power device directly.
* Number of executions is limited.
* As every Microcontrollers does not have analog I/O so there are issues .
* CONCLUSION

I’ve created a dancing led which works and does exactly what the one on intractable net does. I think it looks pretty too. I’m quite happy with it even though I’ve not been able to program different behavior. I hope to be able to do this late.