**SPEED CONTROL OF DC MOTOR USING 8051 MICROCONTROLLER**

A project submitted in the partial fulfillment of the academic requirement

For the award of the degree of Bachelors in Technology from

Electronics & Communication Engineering.

**VIGNAN INSTITUTE OF TECHNOLOGY AND SCIENCE**

Project done at Bharat Dynamics Limited (BDL),

Submitted by,

**1. KONDA SAITEJA (17891A0425)**

**2. LAGISETTY SAI PRABHU (17891A0428)**

**3. BANDA DIKSHITHA REDDY (17891A04D2)**

**4. RAPELLI SAI RAJESH (17891A04F3)**

Under the esteemed guidance of

**RAVI SINGH**

**Designation:** DM of electronics department

**Bharat Dynamics Limited**

**Hyderabad, Telangana.**



We, the undersigned, declare that the project entitled “**SPEED CONTROL OF DC MOTOR INTERFACING 8051**.**MICROCONTROLLER**” being submitted in partial fulfilment for the award of Bachelor of Engineering Degree in **VIGNAN INSTITUTE OF TECHNOLOGY AND SCIENCES**.

A successful project is a fruitful culmination of the efforts by many people. The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mentioning of the people whose constant guidance and encouragement made it possible. We take pleasure in presenting before you, our project, which is result of studied blend of both research and knowledge. I express my deepest gratitude to everyone who has helped me in completing the undertaken project successfully. We are very much indebted and express our earnest gratitude to our External guide, **RAVI SINGH**, Deputy Manager, our project guide, who has guided us with his erudite sagacity, keen perusal and great attentiveness throughout the 1 month of project duration. He has been a constant source of awe and inspiration and provided valuable guidance and suggestions on timely basis and helped us in various stages of the project work. We are grateful for his cooperation and his valuable suggestions.

We are grateful to Shri Rajesh Singh Verma, Deputy General Manager, for inspiring us with his ideas and suggestions throughout the project.

We extend our veneration to entire staff of Division of Electronics, who encouraged us throughout the course of project.

We are also thankful to Shri M. Sreedhar Rao, Additional General Manager, for allowing us to carry out project in Electronic Division.

**OVERVIEW OF THE PROJECT**

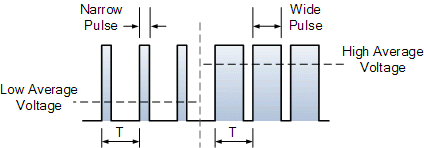
**INTRODUCTION:**

8051microcontroller is designed by Intel in 1981. It is an 8-bit microcontroller. It is built with 40 pins DIP (dual inline package), 4kb of ROM storage and 128 bytes of RAM storage, 2 16-bit timers. It consists of are four parallel 8-bit ports, which are programmable as well as addressable as per the requirement.

**AIM:**

To observe the speed of the DC motor using microcontroller.

**PWM DESCRIPTION**

****

A DC motor consist basically of two parts, the stationary body of the motor called the “Stator” and the inner part which rotates producing the movement called the “Rotor”. For D.C. machines the rotor is commonly termed the “Armature”.

In small light duty DC motor the stator consists of a pair of fixed permanent magnets producing a uniform and stationary magnetic flux inside the motor giving these types of motors their name of “permanent-magnet direct-current” (PMDC) motors.

The motors armature consists of individual electrical coils connected together in a circular configuration around its metallic body producing a North-Pole then a South-Pole then a North-Pole, type of field system configuration.

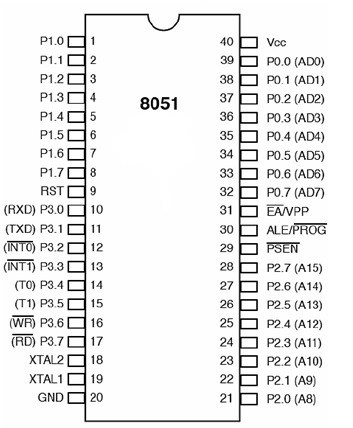
The current flowing within these rotor coils producing the necessary electromagnetic field. The circular magnetic field produced by the armatures windings produces both north and south poles around the armature which are repelled or attracted by the stator’s permanent magnets producing a rotational movement around the motors central axis as shown.

Pulse width modulation is a great method of controlling the amount of power delivered to a load without dissipating any wasted power. The above circuit can also be used to control the speed of a fan or to dim the brightness of DC lamps or LED’s. If you need to control it, then use **Pulse Width Modulation** to do it.

**HARDWARE DESCRIPTION:**

For this purpose, **microcontroller 8051** has an on-chip oscillator which **works** as a clock source for Central Processing Unit of the **microcontroller**. The output pulses of oscillator are stable. Therefore, it enables synchronized **work** of all parts of the **8051 Microcontroller.**

Microcontroller requires a program which is a collection of instructions. This program tells microcontroller to do specific tasks. These programs require a memory on which these can be saved and read by Microcontroller to perform specific operations of a particular task. The memory which is used to store the [program of the microcontroller](https://www.elprocus.com/how-to-program-the-microcontroller/) is known as code memory or Program memory of applications. It is known as ROM memory of microcontroller also requires a memory to store data or operands temporarily of the micro controller. The data memory of the 8051 is used to store data temporarily for operation is known [RAM memory](http://www.edgefx.in/steps-to-build-embedded-c-programming-tutorial/). 8051microcontroller has 4K of code memory or program has 4KB ROM and also 128 bytes of data memory of RAM.



**Pinout Description:**

**Pins 1-8 - PORT 1**. Each of these pins can be configured as an input or an output.

**Pin 9 - RESET**. A logic one on this pin disables the microcontroller and clears the contents of most registers. In other words, the positive voltage on this pin resets the microcontroller. By applying logic zero to this pin, the program starts execution from the beginning.

**Pins10-17 - PORT 3**. Similar to port 1, each of these pins can serve as general input or output. Besides, all of them have alternative functions.

**Pin 10 - RXD**. Serial asynchronous communication input or Serial synchronous communication output.

**Pin 11 - TXD**. Serial asynchronous communication output or Serial synchronous communication clock output.

**Pin 12 - INT0**.External Interrupt 0 input

**Pin 13 - INT1**. External Interrupt 1 input

**Pin 14 - T0**. Counter 0 clock input

**Pin 15 - T1**. Counter 1 clock input

**Pin 16 - WR**. Write to external (additional) RAM

**Pin 17 - RD**. Read from external RAM

**Pin 18, 19 - XTAL2, XTAL1**. Internal oscillator input and output. A quartz crystal which specifies operating frequency is usually connected to these pins.

**Pin 20 - GND**. Ground.

**Pin 21-28 - Port 2**. If there is no intention to use externally then these port pins are configured as general inputs/outputs. In case external memory is used, the higher address byte, i.e. addresses A8-A15 will appear on this port. Even though memory with capacity of 64Kb is not used, which means that not all eight port bits are used for its addressing, the rest of them are not available as inputs/outputs.

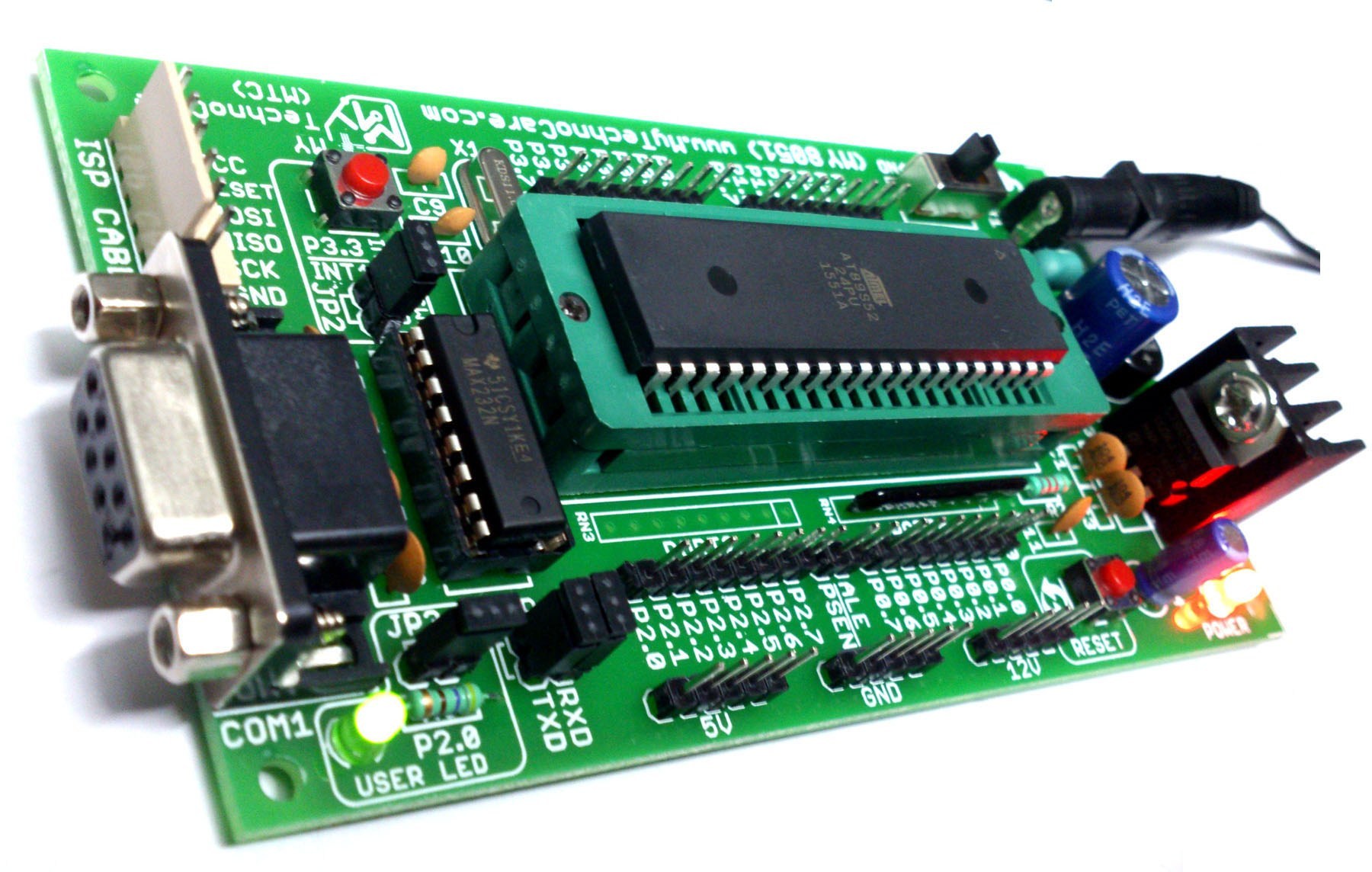
**Pin 29 - PSEN**. If external ROM is used for storing a logic zero (0) appears on it every time the microcontroller reads a byte from memory.

**Pin 30 - ALE**. Prior to reading from external memory, the microcontroller puts the lower address byte (A0-A7) on P0 and activates the ALE output. After receiving signal from the ALE pin, the external latch latches the state of P0 and uses it as a memory chip address. Immediately after that, the ALE pin is returned its previous logic state and P0 is now used as a Data Bus.

**Pin 31 - EA**. By applying logic zero to this pin, P2 and P3 are used for data and address transmission with no regard to whether there is internal memory or not. It means that even there is a program written to the microcontroller, it will not be executed. Instead, the program written to external ROM will be executed. By applying logic one to the EA pin, the microcontroller will use both memories, first internal then external (if exists).

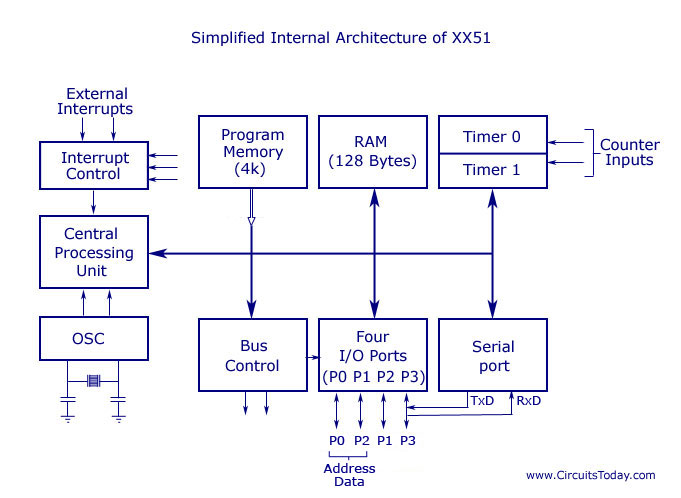
**Pin 32-39 - PORT 0**. Similar to P2, if external memory is not used, these pins can be used as general inputs/outputs. Otherwise, P0 is configured as address output (A0-A7) when the ALE pin is driven high (1) or as data output (Data Bus) when the ALE pin is driven low (0).

**Pin 40 - VCC**. +5V power supply.



The heart of this project is the 8051 Microcontroller. If you have worked with any variant of the 8051 Microcontroller, you might remember that 8051 doesn’t have a dedicated PWM circuitry to enable PWM Mode. So, in order to generate a PWM Signal, we have make use of Timers and switch the I/O pins ON and OFF using the timers.

**8051 INTERNAL ARCHITECTURE**

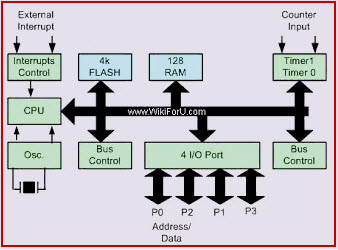
****

**8051 Microcontroller Architecture**

The 8051 Microcontroller is one of the basic type of microcontroller, designed by Intel in 1980’s. This microcontroller  was based on Harvard Architecture and developed primarily for use in [embedded systems](http://www.edgefx.in/embedded-systems-basics-with-applications/) technology. Normally, this microcontroller was developed using NMOS technology, which requires more power to operate. Therefore, Intel redesigned Microcontroller 8051 using CMOS technology and their updated versions came with a letter C in their name, for instance an 80C51 it is an 8bit microcontroller. These latest Microcontrollers requires less power to operate as compared to their previous versions. [The 8051 Microcontroller](http://www.edgefxkits.com/8051-microcontroller-projects)  has two buses and two memory spaces of 64K X 8 size for program and data units. It has an 8bit processing unit and 8 bit accumulator units.

**8051 Microcontroller Architecture**

Following diagram is  [8051 Microcontroller  architecture](https://www.elprocus.com/8051-microcontroller-architecture-and-applications/) . Let us have a look at each part or block of this [Architecture of microcontroller](http://www.edgefx.in/arm-microcontroller-architecture-and-its-programming/).

[](http://www.edgefx.in/wp-content/uploads/2015/02/8051-microcontrolle.jpg)

**8051 Microcontroller Architecture**

**Central Processor Unit (CPU)**

As we know that the CPU is the brain of any processing device of the microcontroller. It monitors and controls all operations that are performed on the Microcontroller units. The User has no control over the work of the CPU directly. It reads program written in ROM memory and executes them and do the expected task of that application.

**Interrupts**

As its name suggests, Interrupt is a subroutine call that interrupts of the microcontrollers main operations or work and causes it to execute any other program, which is more important at the time of operation. The feature of Interrupt is very useful as it helps in case of emergency operations. An Interrupts gives us a mechanism to put on hold the ongoing operations, execute a subroutine and then again resumes to another type of operations.

The Microcontroller 8051 can be configured in such a way that it temporarily terminates or pause the main program at the occurrence of interrupts. When a subroutine is completed, Then the execution of main program starts, five interrupt sources are there in 8051 Microcontroller. There are 5 vectored interrupts are shown in below

* INTO
* TFO
* INT1
* TF1
* R1/T1

Out of these, (INT0) ̅ and (INT1) ̅ are external interrupts that could be negative edge triggered or low level triggered. When All these interrupts are activated, set the corresponding flogs except for serial flags are cleared when the processor branches to the interrupt service routine (ISR). The external interrupt flags are cleared when the processor branches to the interrupt service routine, provides the interrupt is a negative edge triggered whereas the timers and serial port interrupts two of them are external interrupts, two of them are timer interrupts and one serial port interrupt terminal in general.

**Memory**

Microcontroller requires a program which is a collection of instructions. This program tells microcontroller to do specific tasks. These programs require a memory on which these can be saved and read by Microcontroller to perform specific operations of a particular task. The memory which is used to store the [program of the microcontroller](https://www.elprocus.com/how-to-program-the-microcontroller/) is known as code memory or Program memory of applications. It is known as ROM memory of microcontroller also requires a memory to store data or operands temporarily of the micro controller. The data memory of the 8051 is used to store data temporarily for operation is known [RAM memory](http://www.edgefx.in/steps-to-build-embedded-c-programming-tutorial/). 8051microcontroller has 4K of code memory or program memory, that has 4KB ROM and also 128 bytes of data memory of RAM.

**BUS**

Basically Bus is a collection of wires which work as a communication channel or medium for transfer of Data. These buses of 8, 16 or more wires of the microcontroller. Thus, these can carry 8 bits,16 bits simultaneously. Hire two types of buses that are shown in below

* Address Bus
* Data Bus

**Address Bus**: Microcontroller 8051 has a 16 bit address bus for transferring the data. It is used to address memory locations and to transfer the address from CPU to Memory of the microcontroller. It has  four addressing modes that are

* Immediate addressing modes.
* Bank address (or) Register addressing mode.
* Direct Addressing mode.
* Register indirect addressing mode.

**Data Bus**: Microcontroller 8051 has 8 bits of the data bus, which is used to carry data of particular applications.

**Oscillator**

Generally, we know that the microcontroller is a device, therefore it requires clock pulses for its operation of microcontroller applications. For this purpose, microcontroller 8051 has an on-chip oscillator which works as a clock source for Central Processing Unit of the microcontroller. The output pulses of oscillator are stable. Therefore, it enables synchronized work of all parts of the 8051 Microcontroller.

**Input or Output Port**

Normally [microcontroller is used in embedded systems](http://www.edgefx.in/embedded-systems-basics-with-applications/) to control the operation of machines in the microcontroller. Therefore, to connect  it to other machines, devices or peripherals we require I/O interfacing ports in  the [microcontroller interface](http://www.edgefx.in/keypad-interfacing-with-8051-microcontroller/). For this [purpose microcontroller 8051](https://www.elprocus.com/peripherals-interfacing-to-the-microcontroller-8051-in-electronics) has 4 input, output ports to connect it to the other peripherals

**Timers/Counters**

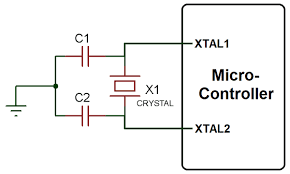
8051 microcontroller has two 16 bit timers and counters. These counters are again divided into 8 bit register. The timers are used for measurement of intervals to determine the pulse width of pulses.

**CRYSTAL OSCILLATOR**

CRYSTAL OSCILLATOR Crystals may be used to generate a popular form of oscillator circuit known as a Pierce oscillator. C Crystal R JFET L VCC Oscillator output (to microcontroller).

A variant of the Pierce oscillator is common in the 8051 family. To create such an oscillator, most of the components are included on the microcontroller itself.

The user of this device must generally only supply the crystal and two small capacitors to complete the oscillator implementation.

****

**Oscillator frequency and machine cycle period**

In the original members of the 8051 family, the machine cycle takes twelve oscillator periods.

In later family members, such as the Infineon C515C, a machine cycle takes six oscillator periods; in more recent devices such as the Dallas 89C420, only one oscillator period is required per machine cycle.

As a result, the later members of the family operating at the same clock frequency execute instructions much more rapidly.

**Keep the clock frequency as low as possible**

Many developers select an oscillator / resonator frequency that is at or near the maximum value supported by a particular device.

This can be a mistake:

Many do not require the levels of performance that a modern 8051 device can provide.

The electromagnetic interference (EMI) generated by a circuit increases with clock frequency.

In most modern (CMOS-based) 8051s, there is an almost linear relationship between the oscillator frequency and the power-supply current. As a result, by using the lowest frequency necessary it is possible to reduce the power requirement: this can be useful in many applications.

When accessing low-speed peripherals (such as slow memory, or LCD displays), programming and hardware design can be greatly simplified - and the cost of peripheral components, such as memory latches, can be reduced - if the chip is operating more slowly.

**Overall strengths and weaknesses**

Crystal oscillators are stable. ±20-100 ppm = ±50 minutes per year (up to ~1 minute / week).

The great majority of 8051-based designs use a variant of the simple crystal-based oscillator circuit presented here: developers are therefore familiar with crystal-based designs.

Quartz crystals are available at reasonable cost for most common frequencies. The only additional components required are usually two small capacitors. Overall, crystal oscillators are more expensive than ceramic resonators.

**BUT:**

Crystal oscillators are susceptible to vibration.

The stability falls with age.

**SOFTWARE PROCEDURE**

**SFRs and ports**

Control of the 8051 ports through software is carried out using what

are known as ‘special function registers’ (SFRs).

Physically, the SFR is area of memory in internal RAM:

P0 is at address 0x80

P1 at address 0x90

P2 at address 0xA0

P3 at address 0xB0

**SFRs and ports**

A typical SFR header file for an 8051 family device will contain the

lines:

sfr P0 = 0x80;

sfr P1 = 0x90;

sfr P2 = 0xA0;

sfr P3 = 0xB0;

Having declared the SFR variables, we can write to the ports in a

straightforward manner. For example, we can send some data to

Port 1 as follows:

unsigned char Port\_data;

Port\_data = 0x0F;

P1 = Port\_data; /\* Write 00001111 to Port 1 \*/

Similarly, we can read from (for example) Port 1 as follows:

unsigned char Port\_data;

P1 = 0xFF; /\* Set the port to ‘read mode’ \*/

Port\_data = P1; /\* Read from the port \*/

**Creating and using sbit variables**

To write to a single pin, we can make use of an sbit variable in the

Keil (C51) compiler to provide a finer level of control.

Here’s a clean way of doing this:

#define LED\_PORT P3

#define LED\_ON 0 /\* Easy to change the logic here \*/

#define LED\_OFF 1

...

sbit Warning\_led = LED\_PORT^0; /\* LED is connected to pin 3.0 \*/

...

Warning\_led = LED\_ON;

... /\* delay \*/

Warning\_led = LED\_OFF;

... /\* delay \*/

Warning\_led = LED\_ON;

... /\* etc \*/

**Creating “software delays”**

**Problem**

How do you create a simple delay without using any hardware

(timer) resources?

Solution

Loop\_Delay()

{

unsigned int x,y;

for (x=0; x <= 65535; x++)

{

y++;

}

}

Longer\_Loop\_Delay()

{

unsigned int x, y, z;

for (x=0; x<=65535; x++)

{

for (y=0; y<=65535; y++);

{

z++;

}

}

}

**PROGRAM CODE**

/\*#define Inc(P3\_bit.P3\_7)

#define Dec(P3\_bit.P3\_5)

//#define Dir(P2\_bit.P2\_1)

#define Pwm(P2\_bit.P2\_0)\*/

sbit Dir=P2^1;

sbit Inc=P3^7;

sbit Dec=P3^5;

sbit Pwm=P2^0;

typedef unsigned char uchar;

typedef unsigned int unit;

void delay(int);

void main(void);

{

int speed;

Dir=1;

if(Dir)

{

speed=1000;

}

else

speed =100;

while(1)

{

if(!Inc)

speed+=1;

if(!Dec)

speed-=1;

}

Pwm=1;

delay(speed);

Pwm=0;

delay(500-speed);

}

}

void delay(int unit)

{

int j=unit;

for(j=unit;j>=0;j--);

}

**APPLICATIONS:**

Some of the applications of 8051 is mainly used in daily life & industrial applications also some of that applications are shown below

* Light sensing and controlling devices
* [Temperature sensing and controlling  devices](http://www.edgefx.in/automatic-fan-controller/)
* Fire detections and safety devices
* Automobile applications
* Defense applications

**Some industrial applications of micro controller and its applications**

* Industrial instrumentation devices
* Process control devices

**Some of 8051 microcontroller devices are used in measurement applications**

* Voltmeter applications
* Measuring and revolving objects
* Current meter objects
* Hand held metering system

**8051 Microcontroller Applications in Embedded Systems**

The applications of 8051 microcontroller involves in 8051 based projects. The list of 8051 projects is listed below.

* Arduino Managed High Sensitive [LDR based Power Saver for Street Light Control System](http://www.edgefxkits.com/ldr-based-power-saver-for-intensity-controlled-street-light)[cheap rolex replica](http://allsignsandbanners.com/Gallery.html)
* The Temperature Humidity Monitoring System of Soil Based on Wireless Sensor Networks using Arduino
* RFID based Electronic Passport System for Easy Governance using Arduino
* Arduino based RFID Sensed Device Access
* [Arduino based DC Motor Speed Control](https://www.edgefx.in/)
* Arduino Based Line Following Robot
* [Zigbee based Automatic Meter Reading System](http://www.edgefxkits.com/zigbee-based-automatic-meter-reading-system)
* GSM based Electricity Energy Meter Billing with Onsite Display
* Android Phone Speech Recognition Sensed Voice Command based Notice Board Display
* Parking Availability Indication System
* Voice Controlled Home Appliances
* [Remote Control Home Appliances](http://www.edgefx.in/remote-control-for-home-appliances/)
* PC Mouse operated Electrical Load Control Using VB Application
* Solar Highway Lighting System with Auto Turn Off in Daytime
* [8051 Microcontroller based Wireless Energy Meter](http://www.edgefx.in/microcontroller-based-wireless-energy-meter/)
* Farmer Friendly Solar Based Electric Fence for Deterring Cattles
* Vehicle Movement Sensed Streetlight with Daytime auto off Features