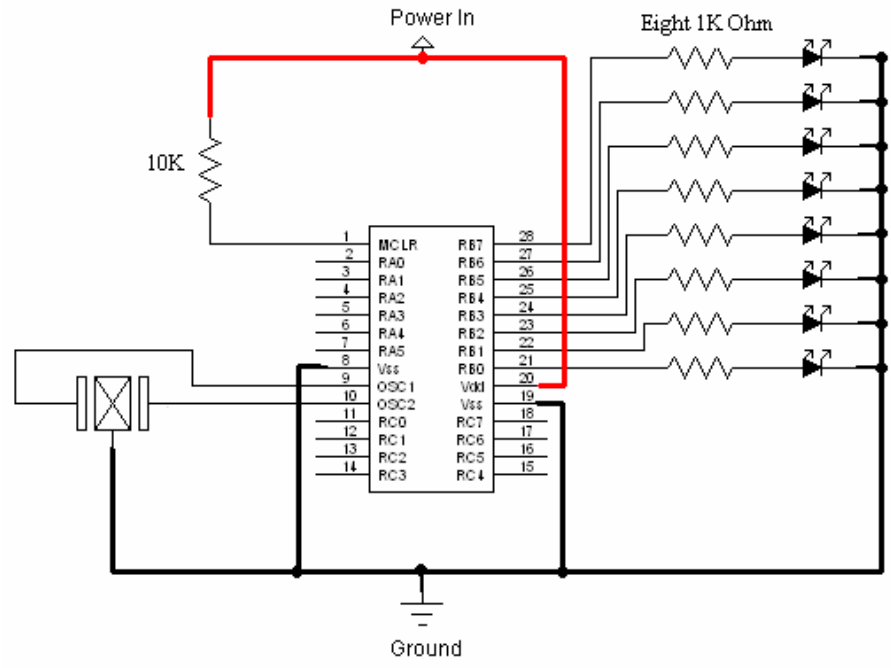
8 Flashing Lights
Binary Counter

The PIC16F876 Section

Okay, here we are again. Let's see if we can follow along with the PIC16F876 chip. This should be a piece of cake.

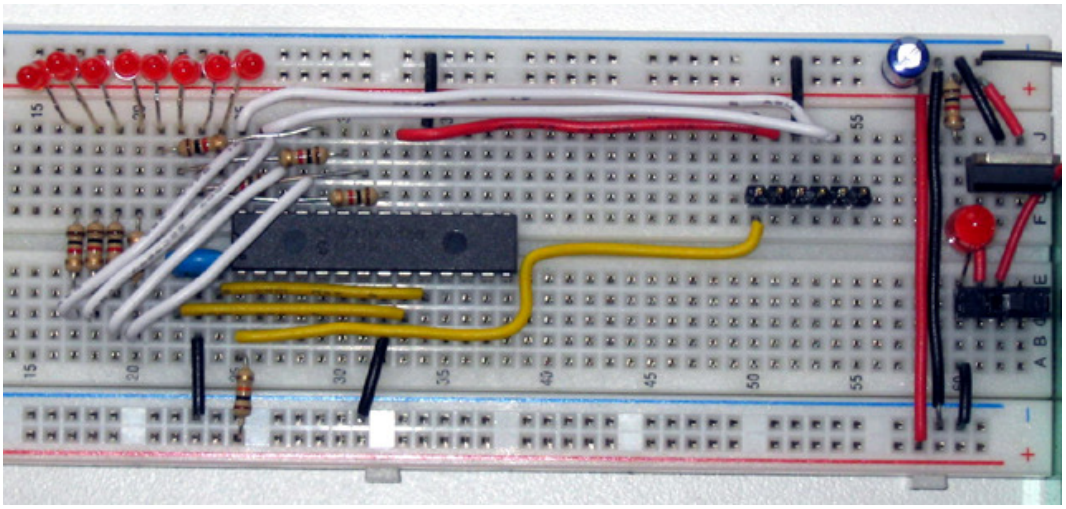
The Eight LED Circuit.



Parts	Value	Number
PIC16F876	N/A	1
Resonator	4MHz	1
Resistor	1K Ohm	8
LED's	any	8
Resistor	10K Ohm	1

The Board

See why you needed the space for the LED's?



The Breadboard with 8 LED's and ICSP

100 calories will propel a bicycle 3 miles and drive a car 280 feet…..Wow.

The Code

```
'****************8 LEDS************************
program CountingUp
'************Variables****************************
Dim value as Byte
'*********************************************
sub procedure Dlay100
   delay_ms(100)          ' Chill for 100 milliseconds
end sub
'*********************************************
main:
   TrisB = %00000000    ' Configure pins of PortB as output
   value = 0             ' Start this code off with value = 0
Count_Up:
   PortB = value         ' Turn on LED on PortB
   value = value +1      ' Add one to "value"
   If value = 255 Then   ' If value is equal to 255 then…..
     value = 0           ' Value switches back to being equal to 0
   End If                ' Always need to end an If Then statement
   Dlay100               ' Jump up to that Dlay100 sub procedure
goto Count_Up            ' Go back up to the Count_Up portion
End.                     ' All good things must end
'*********************************************
```

Okay. Did ya load it up? Did it run?

Cool huh?

Let's talk a bit about what the heck happened here. Check out the line:

```
Dim value as Byte
```

Simple enough, eh'?

Well, what is going on is we have declared a variable. I like to think of variables as boxes. A variable is a box that holds numbers or letters. Like any box, you got'ta pick the right size box! You can't expect a wheelbarrow to fit in a shoebox can ya? Well, the same thing applies with variables. Think about this. Let's say you have 100 wheelbarrow-sized boxes in your house and only one wheelbarrow. Where are you going to put all those boxes?

I bet they couldn't even fit in your house. So if you needed boxes for your stuff, it sure would be nice to pick the right sized box for your stuff…right? Well, you do the same darn thing with variables. You need to choose the size of box or variable you're going to use. That is what the "Byte" is all about. It tells the compiler that I need a box or variable of a certain size. Check out this table below:

Type	Size	Range
byte	8-bit	0 – 255
char	8-bit	0 – 255
word	16-bit	0 – 65535
short	8-bit	-128 – 127
integer	16-bit	-32768 – 32767
Float	32-bit	$1.17549435082*10^{38}$ – $6.80564774407*10^{38}$
Longint	32-bit	-2147483648 – 2147483647

So, if you need a variable that will only hold a number as big as 13 then you need a byte sized box. End If. If you need a box that will hold a number like 3.567 then you need a float. End If. If you need a box to hold a number like –23 then you need a short. End If.

In the case of our experiment, we only needed a little box that will hold a number less than 256. A byte was just the right sized bite.

Okay, so now we have set up the right sized box for our variable. Now we need to put something in the box. Oddly, we are going to make that first something a …well…um…..nothing. However, when you think back to that little section on binary, you will likely recall that even a nothing like zero is something. That something in binary is %00000000. We achieve sticking nothing or zero into the variable with this line

```
value = 0                    ' Start this code off with value = 0
```

I never thought that nothing could be so cool! Think about it. To a computer it's eight zeros long.

Wow!

Check out this line:

```
PortB = value                ' Turn on LED on PortB
```

Now, I know you guys are getting familiar with PortB, but check it out, we made it equal to our variable! It starts out equal to zero as we said before but now it is able to be anything our variable is.

Now, check out this cool line:

```
value = value +1             ' Add one to "value"
```

Value is equal to value plus one. Well Peter Pan on a peanut butter sandwich! Gives me the chills! Check it out! If we started with value being equal to zero, I do believe that we are saying that value is now equal to zero plus one. Now let me think about this. Hold on..I'm going to get my calculator.

Yes!

I was right!

Value is now equal to one!

Wow…I can hardly wait to get to the next line!

Oh geeez! It's one of those If-Then thingies! Goodness gracious!

```
If value = 255 Then          ' If value is equal to 255 then….
```

If value is equal to 255 Then…….What! What…..I can't wait. Let's get the next line. I'm so excited.

```
value = 0                    ' Value switches back to being equal to 0
```

Holy smokes!
If value is equal to 255 Then value is equal to zero. Wow! But wait, we said earlier that value was equal to just zero. I know that is where we started. But then we added one. So then value is equal to only one. We are meters away from 255.

Let's go on.

```
End If              ' Always need to end an If Then statement
```

End If…Yeah yeah. Enough already. I think we all get that you need to end every If -Then statement with an End If.

Okay…what's next?

```
Dlay100             ' Jump up to that Dlay100 subroutine
```

Okay, Okay….Yeah ,yeah, yeah I know we jump up to the Dlay100 sub procedure.

What's next?

```
goto Count_Up    ' Go back up to the Count_Up portion
```

Wow! Check it out! We jump right back up.

But wait, we only jump back up to the Count_Up portion of the code. That section is below the value = 0 part. So that means we get to keep value equal to one. Oh geez! Think it through. That means that we load a one into PortB. Wow! Now we add one to value!!!

I get it! I get it!

We keep repeating the code until value equals 255!

Then it clears back to zero!

Then it all starts over again!

Leaping lizards! Gosh this stuff is cool!

Gratuitous Knowledge Bonus

There are flies called Cyrtidae that make their living as parasites to spiders. The eggs are laid on the ground and when they hatch the larva wait for a passing spider. Once they peg one they jump up and attach themselves. Once there, they slowly eat their way though the skin and start eating the whole spider from the inside out.

And you think having a little brother tag along is a drag?