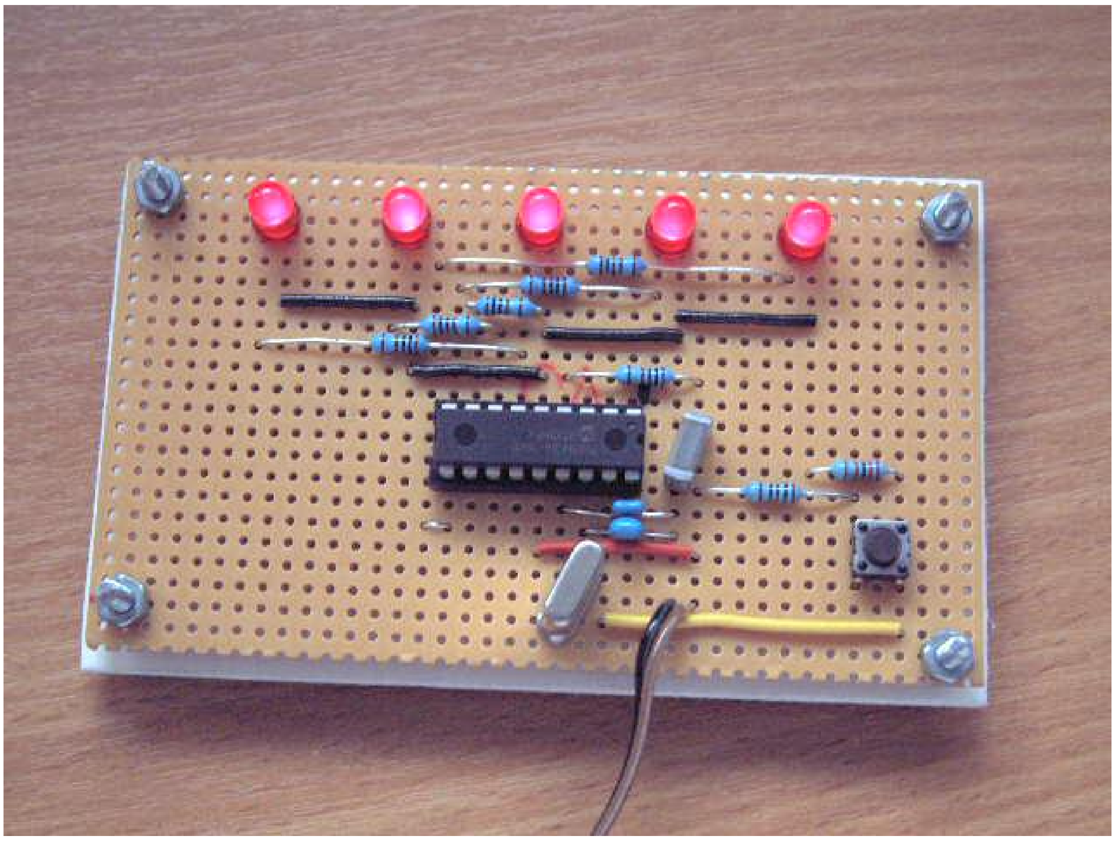
Formula One Starting Grid Lights



A Project By

Perry Andrews

Based on the PIC16F84A Micro controller.

06 January 2002

Contents

Introduction

As this was my first project using the PIC16F84A micro-controller I decided to tackle something not too demanding. At the start of each Formula One grand prix five lights are used to start the race. The sequence is that each of the five lights is turned on at one-second intervals after the cars are stationary on the grid. Then a delay of between one and five seconds occurs before the lights are turned off for the race to begin. This project simulates the above description and can be used to start slot car races.

To develop this project I used the PIC Tutor development board and C for PICmicro micro-controllers, both by Matrix Multimedia Ltd. I decided on using C because I am already familiar with this programming language and this would speed up the development time. The project is not demanding in performance or space requirements so there is no disadvantage using a high level language.

The project is split into two sections. The Design and Development section includes writing and testing the program and the design of the circuit. The construction section includes building the circuit on Vero strip board and testing the final circuit.

Design and Development

I started by listing the steps the program requires:

1. Initialise the PIC and set all outputs to off.
2. Turn each of the five outputs on with a delay of one second between each.
3. Generate a random number.
4. Turn all outputs off after a delay specified by the random number.

Based on these steps I created a flow diagram to help create the program.

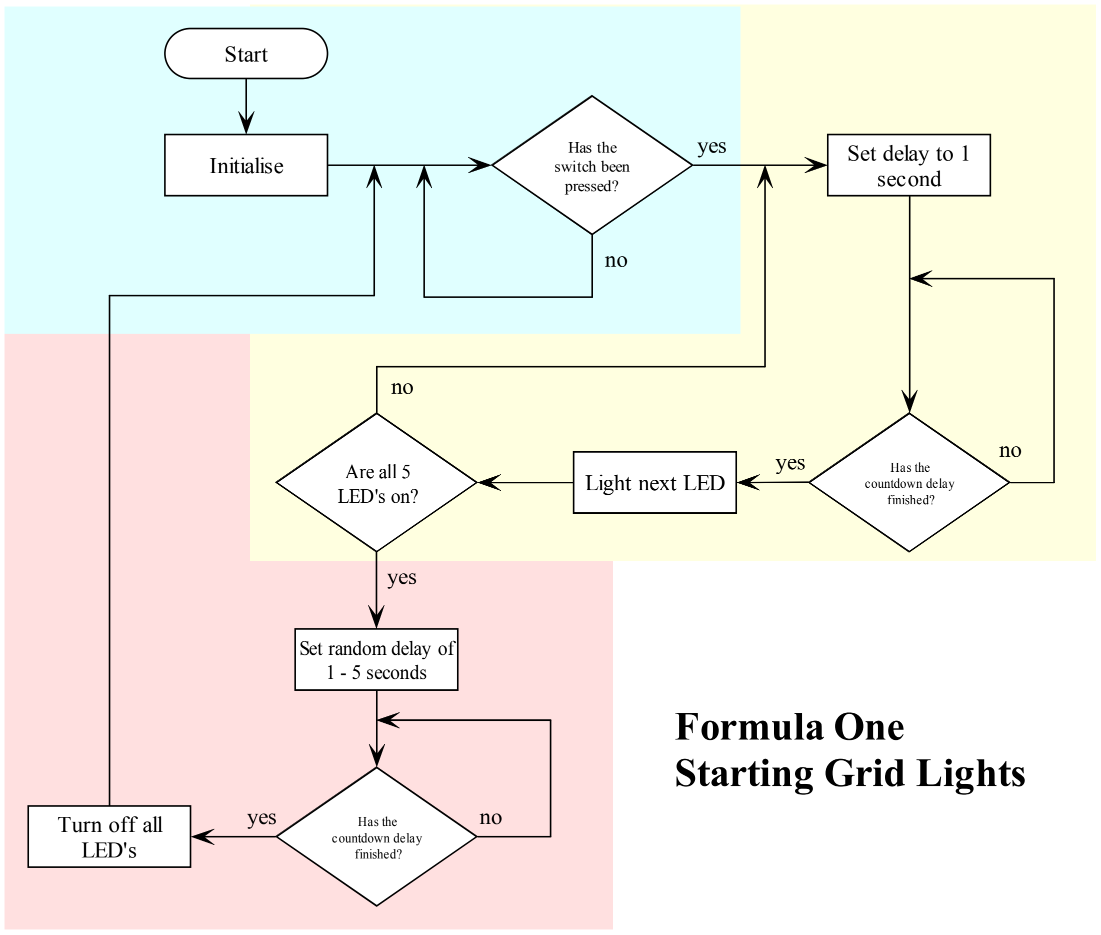

Start
Initialise
Has the switch been pressed?
yes
no
Set delay to 1 second
Has the countdown delay finished?
no
yes
Light next LED
Are all 5 LED's on?
no
yes
Set random delay of 1 - 5 seconds
Has the countdown delay finished?
no
yes
Turn off all LED's
Formula One Starting Grid Lights

Figure 1 - Flow Diagram

The three distinct sections are marked with different colour backgrounds. The blue background section is the section where initialisation takes place and the program polls for the start button to be pressed. The yellow section is the section where the LED's are turned on sequentially. The pink section is the section where the LED's are turned off after a random delay.

The C program is shown below:

```
/*  Formula One LED Start Lights */
/*  (c) Perry Andrews Jan. 2002  */

/* use LFSR based algorithm for random   */
/* number generation                      */

unsigned char seed = 1 ;

unsigned char random ( void )
{
   unsigned char top, mid ;

   /* get new input bit  */

   if ( seed & 12 ) {
      mid = 1 ;
   } else {
      mid = 0 ;
   }

   if ( seed & 0x80 ) {
      top = 1 ;
   } else {
      top = 0 ;
   }

   seed = ( seed << 1 ) | ( mid ^ top ) ;
   return seed ;
}

void main (void)
{
   unsigned int i ;
   unsigned int j ;
   unsigned char r;

   /* Select the Register bank 1 */
   set_bit ( STATUS, RP0 ) ;
   /* set all of PORTB for output */
   TRISB = 0x00 ;
   /* set all of PORTA for input */
   TRISA = 0x1f ;
   /* now use Register bank 0    */
   clear_bit ( STATUS, RP0 ) ;

   // set all LEDs off
   for ( j=0; j<5; j=j+1 )
           output_low_port_b(j);
```

```
  // Main program loop
  while (1) {
        // Wait for Switch A0
        while ( input_pin_port_a(0)==0 );

        // set the LEDs 1 per second
          for ( j=0; j<5; j=j+1) {
            for ( i=0 ; i < 35200 ; i= i + 1 ) ;
            output_high_port_b ( j ) ;
          }
          for ( i=0 ; i < 35200 ; i= i + 1 ) ;
        // generate a random time period
          r = random () % 4;
          for ( j=0; j<r; j=j+1 )
            for ( i=0 ; i < 35200 ; i= i + 1 ) ;
        // set all LEDs off
          for ( j=0; j<5; j=j+1 )
            output_low_port_b(j);
  }
}
```

The random function was taken from the C for PICmicro. This is probably not the best program but it does the job! The program was built up using the coloured sections in the flow diagram. I concentrated on one section at a time and programmed and tested the PIC before moving onto the next section. This is achieved in the quickest time possible as the C development environment compiles, assembles and programs all in one hit!

The circuit diagram is very straightforward in comparison to a design using digital electronics as the PIC microcontroller does all the work. The diagram is shown below:

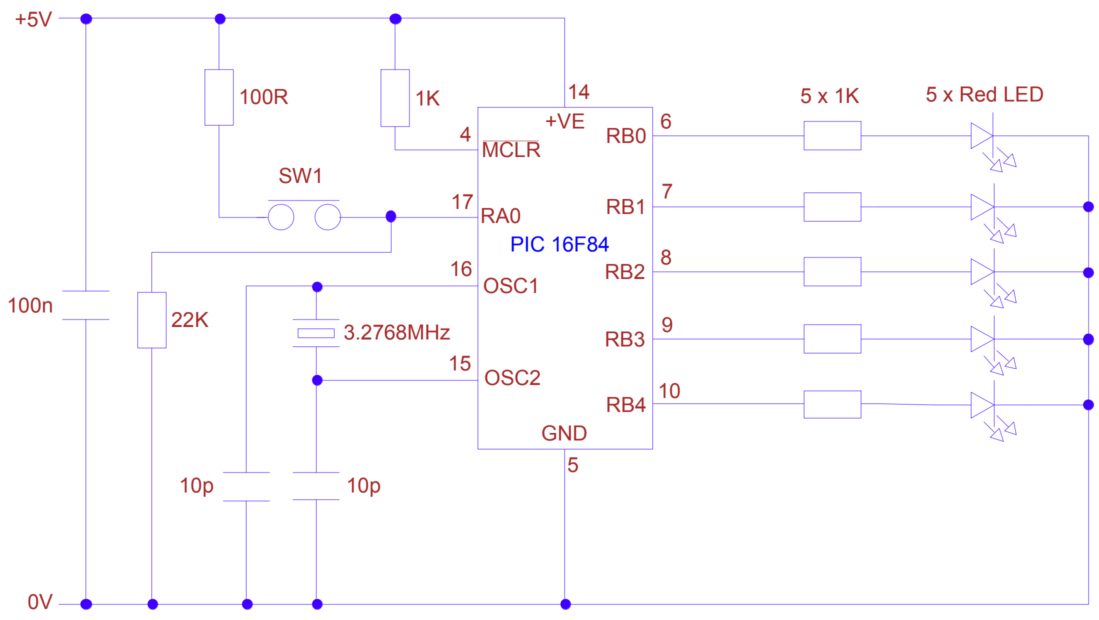


Figure 2 - Circuit Diagram

Construction

This project was constructed using Vero strip board, as it is quick to work with. A parts list is shown below:

Description	Quantity	Maplin Stock Code
Vero Strip Board	1 pc 100mm x 60mm	JP49D
PIC16F84A	1	VS88V
Red LED's	5	WL27E
3.2768MHz Crystal	1	FY86T
10pf Capacitors	2	RA33L
100nf Capacitor	1	RA49D
1K0 Resistor	6	M1K
100R Resistor	1	M100R
22K Resistor	1	M22K
PCB Switch	1	KR88V
18Pin DIL Socket	1	HQ76H

Table 1 - Parts List

I used a crystal as the timing source because of its accuracy. The R-C network would have worked just as well in this application and would have been a bit cheaper. If this is desired then replace the crystal with a resistor and variable resistor. The variable resistor can then be altered to adjust the timing.

To start constructing the project I marked the positions of the five LED's. The middle LED was located first by finding the middle two holes. I also left a gap at the top in case some jumper wires were needed. Next the position of the IC socket was found. The socket was orientated so that the outputs used for the LED's were on the same side as the LED's. I also located pin 5 of the socket on the same copper strip as the middle LED cathode to save an additional jumper wire. I left seven rows to allow room for the LED resistors. The socket was then soldered in place and the copper strips cut between the pins of the IC socket. The resistors were soldered in next along with the 7 jumper wires. Next the switch and capacitors were located and soldered in place. Last were the five LED's and the crystal.

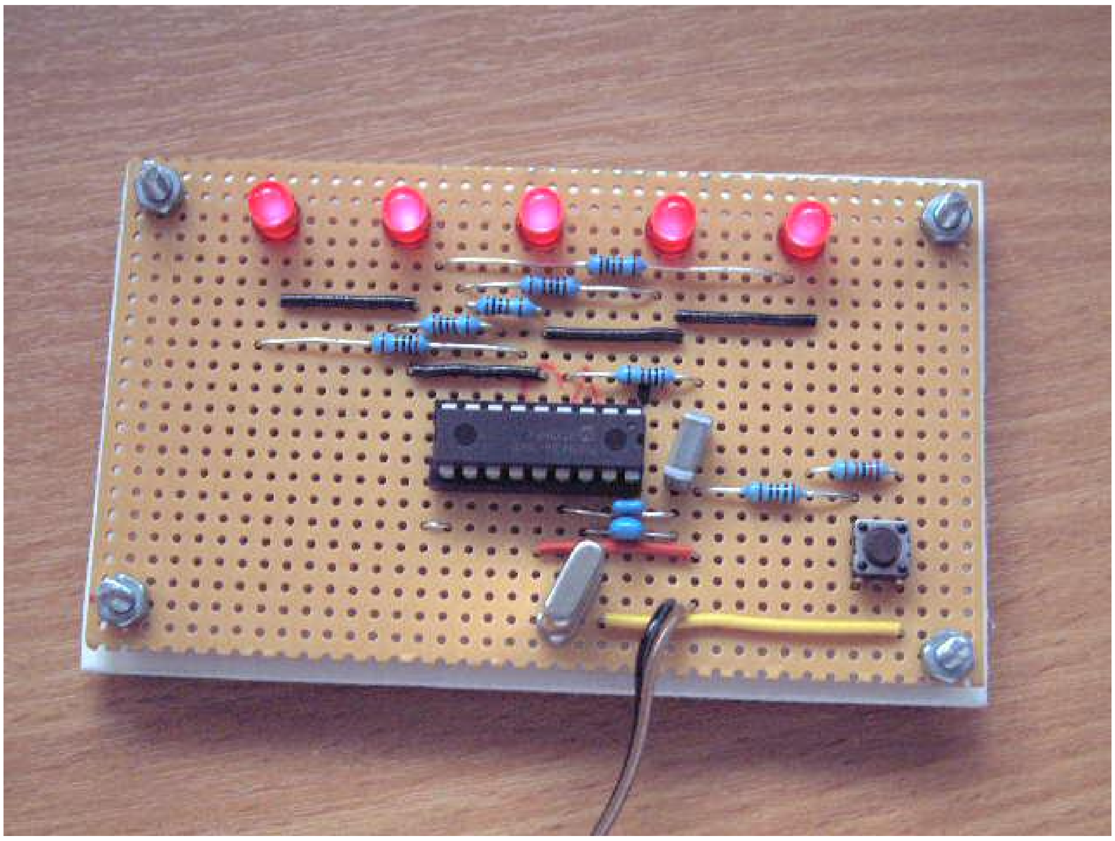


Figure 3 - The Completed Project

All that was left to do was to plug in the PIC and connect the 5V supply to test.

Conclusion

The aim was to produce a small project in a short space of time to help learn how to program and use a PIC microcontroller. The project took about five hours to complete including the program development and construction. The construction was the longest part of the project and more time spent here reduced the possibility of errors. The circuit design was simple which also made construction easier compared to a version using digital electronics. Using the development environment that came with the PIC Tutor board was simple and allowed things to be tried on the PIC very quickly. I did have to change the default settings of the programming software however, as it would only work with the supplied PIC. I had two slightly different versions of the PIC, which did not program using the default delay value.

Below I have listed some future developments for this project:

1. Add a second button so the start can be aborted before all five lights are on. This is signified by all five lights blinking,
2. Add two sensors for slot car racing so that a jump-start can be detected. This would be signalled by flashing two LED's on the side that caused the jump-start.
3. Revisit the assembly language and improve the code. This would be done after learning the assembly language.

References

Matrix Multimedia Ltd
C for PICmicro microcontrollers by Rob Miles
PIC Tutor development board

Maplin Electronic Supplies
Supplier of all components including the PIC development board.