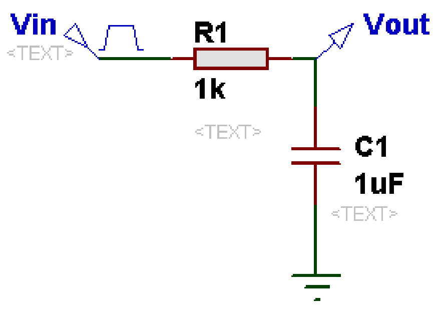
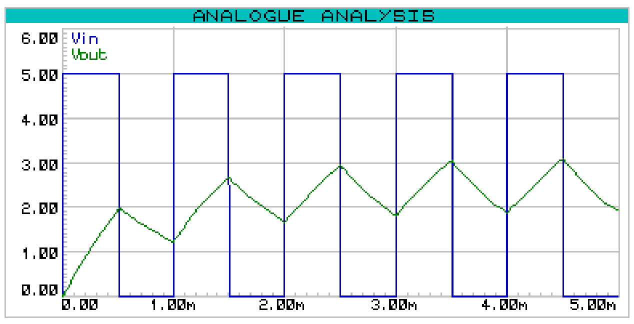
EE443 - Embedded Systems
Experiment 1
Code Compilation and Simulation

Purpose: Become familiar with the development tools used in the experiments. Learn compilation and simulation steps to verify microcontroller programs.

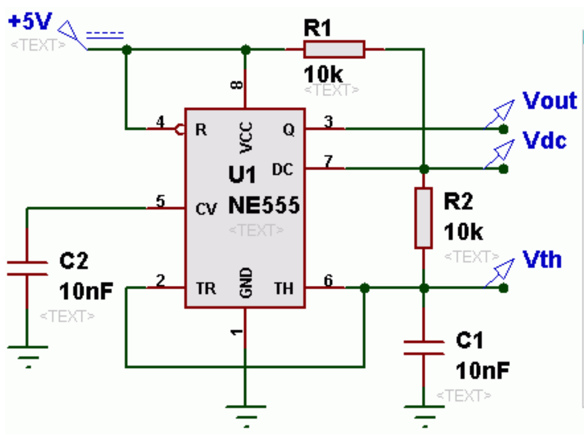
Preliminary Work

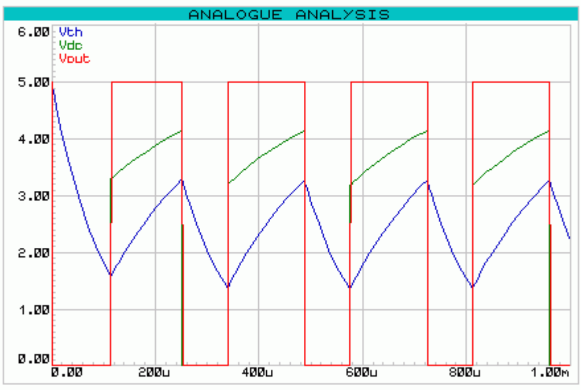
1. Run ISIS Professional design tool, start a new design, and draw the simple R-C circuit shown below. Place a square wave voltage generator as **Vin** and a voltage probe labeled as **Vout**. Place an analog graph, set stop time to **5 ms**, add traces for **Vin** and **Vout**, (right-click on the graph and select "Edit Graph" or "Add Traces..." options) and simulate the graph to obtain the waveforms given in the following figure.





2. Start another design file and simulate the astable multivibrator circuit that uses a 555 timer as shown below. Double-click on the device selector title labeled as "**DEVICES**" or click on the "**P**" button to select the **NE555** timer from the library of available components. NE555 can be found in the **Analog ICs** category.



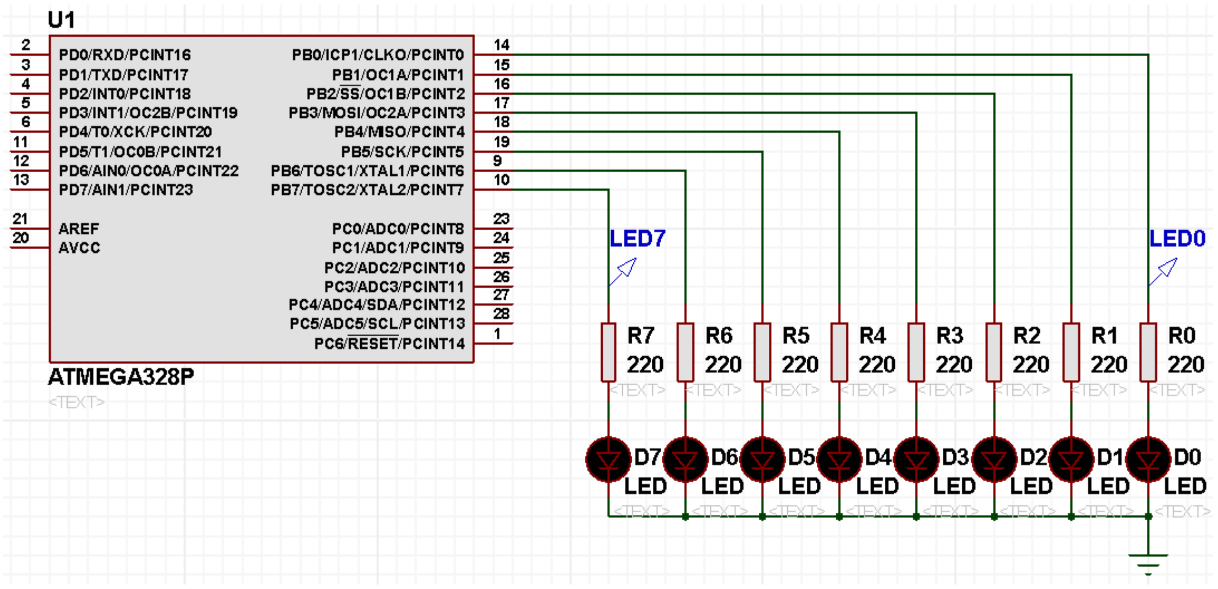


3. The following C program will be used with the circuit schematic given below for this experiment.

```
#include <avr/io.h>
#include <util/delay.h>

int main(void)
{
DDRB |= 0xFF;

while(1)
{
  PORTB = 0x0F;
  _delay_ms(500);
  PORTB = 0x00;
  _delay_ms(500);
};
return 0;
}
```



The C program turns ON and OFF the four LEDs (LED0 through LED3). The macro provided by the WinAVR compiler, **`_delay_ms(500)`**, keeps the processor at idle for 500 ms running dummy instructions in a loop. Go through the program line by line and write the necessary comments describing the purpose of each statement.

Procedure

1. Create a project directory on your hard disk.
Note: Proteus does not support file paths longer than 64 characters. If you place your project directory on the "Desktop" or in the "My Documents" directory, then you may exceed the 64 character limit. It is recommended to open a project directory directly under a drive letter such as **C:\EE443\Exp1**.

2. Start the **Code::Blocks** IDE and create a new **AVR project** in the directory that you created.

- Check the **Create "Release" configuration** box.
- Select **atmega328p** processor, without any external memory.
- Define **F__CPU** as **8000000UL**.
- Make sure that **Create hex files (.hex)** and **Create extended listing file (.lss)** boxes are checked.
- Uncheck **Create Fuse, Lock, Signature files** box.

3. Open the source file, **main.c**, and copy the C program given above. Compile the C program using the **Build** command.

4. Copy the provided Proteus design file, **Exp1_LED.dsn**, into the project directory. Start the ISIS design tool and open the design file that you copied. Right-click on the **ATMEGA328P** microcontroller on the ISIS schematic and select **Edit Properties**.

- For the **Program File**, locate the **.hex** file generated by the compiler in the directory, `<project dir>\bin\Release`.
- Leave these fuse settings "Unprogrammed": **CLKDIV8**, **CKOUT**, **RSTDISBL**, **WDTON**, and **BOOTRST**.
- **CKSEL Fuses:** (0010) Int. RC Ocs. 8MHz
- **Boot Loader Size:** (00) 1024 words
- **SUT Fuses:** (00)

5. Start the simulation using the **Debug > Execute** menu item (or click on the "play" button on the bottom left corner). Verify that the LED blinking period is **1 s** using the simulation time index given at the bottom of ISIS window.

6. Modify the C program so that **LED7** will be **ON** while LED0...LED3 are OFF, and it will be **OFF** while LED0...LED3 are ON. Compile the program again and verify the code execution on ISIS.

7. Delete or comment-out the delay macro statements, "`_delay_ms(500);`", in the C code and compile the program again. Open the extended listing (**.lss**) file and review the assembly statements corresponding to the main program.

8. On the ISIS schematic, open a digital graph window and set the stop time to **10 µs**. Add the traces for **LED0** and **LED7** probing nodes. Simulate the graph (right-click on the graph window) and measure the time required to execute one loop cycle.

Question 1: Calculate the number of clock cycles required to complete one cycle of the `while(1) {...}` loop referring to the extended listing (**.lss**) file and the

Instruction Set Summary given on page-522 of ATmega328 datasheet. What is the processor clock frequency in the simulation? How long (in μs) does it take to execute one loop cycle? Is it the same as the execution time measured on the simulation graph?

Question 2: How can you change the program so that the ON-time of LEDs will be equal to the OFF-time? Compile the modified program and verify the timing.

Question 3: Is it possible to write a more efficient version of the same program using the assembly language, so that the `while(1) {...}` loop execution takes less clock cycles? Why?