ENEL206: Principles of Computing
Laboratory 101
UCFK Testing and LED Display Simulator Part 2

Andrew Bainbridge-Smith, A404

July 15, 2008

1 Objective

To test your UCFK microcontroller kits and elaborate your earlier simulated LED matrix display code in Laboratory session 100 to show a series of visual patterns.

The arrangement used in this simulator is similar, but not identical, to the UCFK kit you will build.

2 Housekeeping

Download the zip file for the laboratory (lab101.zip), it contains 3 files: *avr-load, blink.c, rotate.c.* You will also need to use the zip file from the previous laboratory session (lab100.zip).

3 Testing UCFK board

Compiling a C program to produce an executable targeted for your UCFK board requires using a cross-compiler — this is a compiler that runs on one machine but produces code for a different target (AVR). We will be using a gcc compiler very similar to one you are use to when compiling in and for your linux/intel environment, but this version of gcc is targeted for the for AVR family of microcontrollers. Consider the sample c files supplied (blink.c and rotate.c), we compile the first using the command:

```
avr-gcc -g -Wall -mmcu=atmega8 -I. blink.c -o blink.elf
```

The difference between this form of compilation and what you have seen in the past is the use of the "avr-gcc" cross-compiler, and the option "-mmcu-atmega8" which specifies we are targeting an atmega8 micro control unit (MCU). The output file (blink.elf and also rotate.elf) will be downloaded onto your board, follow these steps:

1. Power up your board. You can use the bench power supplies in the "Grotto", but note there are very few of these, so please be mindful others and don't camp. Alternatively you can power your board with your own supply.

2. Plug the D25 programming cable into the parallel port of the computer — note, not all machines in the "Glade" have parallel ports.
3. Load the first program onto the micro using the command:

   ```
   avr-load blink.elf
   ```

   Remove the D25 connector to see the program running and to satisfy your self that the PC is not running the program.
4. Now turn off the the power, the program should stop running.
5. Turn the power back on and observe that the program is running again.
6. Now load the rotate.elf program and repeat.
7. You are ready to test your assignment code.

4 Simulating multiple display patterns

Use the code from the previous lab and modify the last version "displaying a complex pattern" to display a series of complex patterns, where the program rotates through the patterns at a constant fixed speed. You could use two timers for this task.

The technique of controlling the regular interval of execution will use our *paced loop*, and for the example below a second timer to control switching of the patterns:

```
while(1)
{
   TCNT1 = 0;

   if (TCNT0 > 50) // period is 1 second
   {
      pattern_number++
      if (pattern_number == MAX_PATTERNS)
         pattern_number = 0;
      TCNT0 = 0;
   }

   display_pattern(pattern_number);

   while (TCNT1 < 2)  // period is 8 milliseconds
      continue;
}
```