Patrick Harrington
ECEN 2020
Homework 2
September 23, 2013

1 AVR Status Register (SREG)

The AVR status register contains information about the state of the microcontroller's processor - specificaly, the most recently executed arithmetic instruction. The memory-mapped address of the SREG starts at `0x3F`, and continues to `0x5f`. The default value is 0. Bit 1, also called Z, is the **Zero Flag**. It indicates "false" for arithmetic and logic operations, and is useful for branching conditional statements.

2 Checking AVR Status Register SREG

See attached files.

The output was:

```
Bit 0 Status:  On
Bit 1 Status:  On
Bit 2 Status:  On
Bit 3 Status:  On
Bit 4 Status:  On
Bit 5 Status:  On
Bit 6 Status:  Off
Bit 7 Status:  Off
```

3 Low-Level Blink

See attached files

4 Lowest Frequency for Low-Level Blink

The slowest I could get the LED to blink was 14.4Hz. The pulse width was found to be about 37 ms. This was using unmodified code from problem 3; as I did not use nested `for` loops.

5 Memory Addresses, Variables and Definitions

(a) Memory Range

If a variable is defined as `int Count_Rotations=0`, it is likely stored in the `0x0100 - 0x08F7` range, assuming a total of 2KBytes of memory. This address range is the range of addresses that exist for the Atmega SRAM, minus half of the width of an integer, which is 2 Bytes long.

(b) Array Definition

There is a problem with the following array definition:

```
volatile int my_array[2048]={0};,
```

Namely, it is with the value it is initialized with; it matches the size of all SRAM on the Atmega328! It would cause an immediate overflow.

(c) Two Variables

Assuming that the first variable is at address `0x0120`:

```
double my_number = 5;
char my_letter = 'b';
```

The size of a `double` is 4 Bytes, and the size of a `char` is only one Byte. As a result, the address of `b` is most likely `0x0140`.

6 Harvard vs. Von Neumann Architectures

The key difference between the Von Neumann and Harcard architectures is in the nature of the memory bus; with Von Neumann, the bus between program memory and data memory is shared, leading to a bottleneck as data and instructions have to be scheduled. The Harvard architecture has seperate buses for data and program memory, allowing data and instructions to be moving around at the same time. So while although the Von Neumann architecture is simpler, it performs poorly compared to the Harvard architecture. Another difference is in the use of caches. A Von Neumann processor would have a single cache storing everything, whereas a Harvard processor would need a cache for every bus to be efficient.