Aaron Vaughan
ECE375

Prelab 8

1. In this lab, you will be given a set of behaviors/actions that you need to have a proof-of-concept "toy" perform. Think of a toy you know of (or look around online for a toy) that is likely implemented using a microcontroller, and describe the behaviors it performs. Here is an example behavior:
"If you press button X on the toy, it takes action Y (or makes sound Z)".

I had a toy called a glow worm. When it is squeezed it makes a light turn on. After some amount of time, the light turns off on its own.

2. For each behavior you described in the previous question, explain which microcontroller feature was likely used to implement that behavior, and give a brief code example indicating how that feature should be configured. Make your explanation as ATmega128-specific as possible (e.g., discuss which I/O registers would need to be configured, and if any interrupts will be used), and also mention if any additional mechanical and/or electronic devices are needed.

The microcontroller may have used an interrupt sequence to execute a time delay function using a timer/counter. When a button was pressed the interrupt routine sets the output to high and excites the light circuit and jumps to the timer function. The output circuit would have an amplifier and an incandescent light bulb to complete a circuit.

```
.org   $0000
             rjmp   INIT                ; reset interrupt
.org   $003A
             rcall TurnOnLight
             reti
…………………………………………………………….
; Initialize the Stack Pointer
             ldi          mpr, low(RAMEND)
             out          SPL, mpr             ; Load SPL with low byte of RAMEND
             ldi          mpr, high(RAMEND)
             out          SPH, mpr             ; Load SPH with high byte of RAMEND

             ; Configure I/O ports
             ; Initialize Port B for output
             ldi          mpr, $FF             ; Set Port B Data Direction Register
             out          DDRB, mpr            ; for output
             ldi          mpr, $00             ; Initialize Port B Data Register
             out          PORTB, mpr           ; so all Port B outputs are low

             ; Initialize Port D for input
             ldi          mpr, $00             ; Set Port D Data Direction Register
             mov          counter, mpr
```

```
        out         DDRD, mpr           ; for input
        ldi         mpr, $FF            ; Initialize Port D Data Register
        out         PORTD, mpr          ; so all Port D inputs are Tri-State
    ; init the counter/timer3 normal, 256, interrupt enable
        ldi         mpr, high(3036)           ; Set the starting value to 3036
        sts         TCNT3H, mpr
        ldi         mpr, low(3036)
        sts         TCNT3L, mpr
```

3. Each ATmega128 USART module has two flags used to indicate its current transmitter state: the Data Register Empty (UDRE) flag and Transmit Complete (TXC) flag. What is the difference between these two flags, and which one always gets set first as the transmitter runs? You will probably need to read about the Data Transmission process in the datasheet (including looking at any relevant USART diagrams) to answer this question.

The UDRE flag is set when the data register is empty and cleared when the transmit buffer contains data to be transmitted that has not yet been moved into the Shift Register. The TXC flag is set when the entire frame in the Transmit Shift Register has been shifted out and there are no new data currently present in the transmit buffer. This means that the UDRE flag will get set before the TXC flag when the transmission is complete. By looking at the diagram we can confirm this observation knowing the architecture of the microcontroller is working with an 8-bit piece of data first, then putting that 8-bit data string into a shift register to be shifted out serially over the transmission mdedium.

4. Each ATmega128 USART module has one flag used to indicate its current receiver state (not including the error flags). For USART1 specifically, what is the name of this flag, and what is the interrupt vector address for the interrupt associated with this flag? This time, you will probably need to read about Data Reception in the datasheet to answer this question.

For USART1, the RXC1 flag (bit-7 in UCSR1A) is set when there are unread data in the receive buffer and cleared when the receive buffer is empty. So a zero means ready to receive and a 1 means it is busy with data. The interrupt vector is located at address $003C.