ACE411 - Embedded Systems
LAB 2
7-Segment Display Decoder w/ 8 Digit Multiplexing
Odysseas Stavrou 2018030199
October 2021
Technical University of Crete
Prof.: A. Dollas

In the 2^{nd} Lab Excercise of the course on Embedded Systems (ACE411), the goal was to create a 7-Segment Display Decoder, with the ability to multiplex simultaneously 8 digits. The language of Implementation was AVR (Alf and Vegard's RISC / Advanced Virtual RISC) Assembly.

1. **Timer Confugiration**

   If each display was to blink 30 times (or more) a second, then the Human Eye wouldn't be able to tell the difference between that and an always on display. Since there are 8 displays in total, the total 'blink' amount is 240 per second. Meaning that to achieve a seemingly always on setup, each display has to 'blink' every 4ms and each time only one has to be active.

   Using Timer0 with a prescaler of (4) 256 the tick frequency is:

   $$t_{clk} = \frac{10 \cdot 10^6}{256} = 39062Hz$$

   With a TOP value of 255 the overflow frequency becomes:

   $$f_{t0} = \frac{39062}{255} = 140Hz$$

   If the Timer0 register TCNT0 is preloaded with a starting value then the overflow will become more frequent:

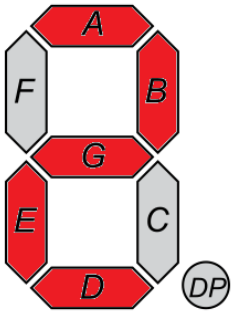
   $$TCNT0 = 255 - \frac{10 \cdot 10^6}{256 \cdot f_{target}}$$

   With a targeted frequency of 250 Hz, TCNT0 has to be preloaded with:

   $$TCNT0 = 255 - \frac{10 \cdot 10^6}{256 \cdot 250} \approx 99$$

2. **Decoder**

   Each display is a CA (Common Anode) Display, meaning the segemnts that need to be light up have to be pulled LOW. So the decoder needs to specify which pins will be set to 0.



If the number 2 is to be displayed then pins A,B,G,E and D have to be set LOW, while the remaining pins should remain HIGH. Pin A is the LSBit and DP is the MSBit so the output in the PORTA register should be 0b10100100 or 0xA4.

There is a special place in SRAM where the decoder's masks are stored. Each time a digit is to be displayed, the decoder will use the mask in the base address offseted by that specific digit. In Example: If the current digit is number 5 the decoder will use the contents of the address: base + 5.

The digit itself, is the value of another place in SRAM, that stores the number, and gets offseted by a register that holds the place of the current digit. This register gets reset, each time the One-Hot register shifts out a carry (More on that below)

```
display_digit:
    ; load our number's address
    ldi xH, high(data)
    ldi xL, low(data)

    ; add digit position and subtract 1
    add xL, digit
    dec xL
    ; dereference pointer and get digit
    ld r0, x

    ; load masks' address
    ldi xH, high(segments)
    ldi xL, low(segments)

    ; offset by digit
    add xL, r0
    ; dereference and write to porta
    ld r0, x
    out PORTA, r0

    ret
```

3. **Digit Multiplexing**

   Two Registers, ring and digit, are used to keep track the digit being displayed and that digit's place, respectively. Ring is a one-hot register meaning of the 8 bits, only 1 is set and the others are zero. Each time there is an interrupt the digit register is incremented and the ring register is shifted left, if a carry is being shifted out, it means tha last digit was displayed last time and the register is reset again with the value of 1 along with the digit register.

4. **Testing And Confirming**

   To confirm that the system is functioning properly the Debugger was used, breaking on each interrupt and observing which bit is enabled in PORTC reveals that only one

display is active on any given moment, and the transitions & reset of the register work as intented. Also the light up digit should be corresponding to the correct display been driven, to confirm that each digit is displayed in the correct place and sequence.

Lastly to evaluate whether the timing works, there's only need to measure the difference in cycles between two consecutive Interrupts:

$$80390 - 40198 = 40192 \Rightarrow 40192 \cdot \frac{1}{T_{clk}} = 40192 \cdot \frac{1}{10 \cdot 10^6}$$

$$T_{interr} \approx 0.004s \approx 4ms \Rightarrow 250Hz$$

Note: This should had been included in the last Report in order to confirm the real Time difference between two interrupts, rather than just relying on formulas.

5. **Implementation**

The timer was setup in the excact same manner as the first Lab. The X,Y 16-bit registers are needed in order to refer to memory locations. (Both for loading and storing data). There is need to allocate memory in the data section, using the .byte directive preceded with a label, in order to be able to refer to it:

```
.dseg
.org 0x200
    data: .byte 8
.org 0x250
    segments: .byte 10
```

Storing the number has to be done in BCD form, ie: if output will be 1956, then 4 bytes in RAM have to be reserved with value 0x01090506. With the help of the Y register and using post-increment addressing mode the data and the decoder's masks are initialized in RAM.

```
ldi yH, high(segments)
ldi yL, low(segments)

ldi r16, 0xC0
st y+, r16
```