Contents

1 60Hz Divider **1**
1.1 Overview 1
1.2 Initial Notes: Counting the Hz 1
1.3 MAX7219 8 digit 7 LED segment Display Driver . . . 2
1.4 CPLD Programming 2
1.4.1 6KHz clock 3
1.5 Divide by N Counters 3

1 60Hz Divider

1.1 Overview

Let's count. There is a schematic in Practical Electronics For Beginners 4th edition. I've built that up, and will add some CPLD counter logic, along with a micro to output the SPI to a 7seg counter module.

The goal is relative accuracy. Not absolute. No GPS here. I'm going from 60 to 6,000 cycles.[1] This is just meant to be fun.

1.2 Initial Notes: Counting the Hz

pseudo code goal:

```
Using 1Hz signal
Start counting 1MHz every 1Hz
when next cycle is received,
   display count
   start counting again
```

That's all the objective is here. Easy with a micro, but goal is to complete using cmos or 74 logic.

[1] Due to limitations of CPLD

4553 x 5 74hct132 1MHz clock (or 6MHz clock), or some variation thereof jk flip flop 74376 - quad jk flip flop 7476 - jk flip flop 1mhz clk will be main counter, 6 hz or 1 hz will be latch / reset

I ended up skipping the 74 CMOS, in favor of a CPLD. Practical Electronics also mentions this approach as favored. Even a micro alone could be used. Schematic entry in the CPLD could also be used.

1.3 MAX7219 8 digit 7 LED segment Display Driver

Basic code tested with this was the LedControl arduino library.

```
/*
 Now we need a LedControl to work with.
 ***** These pin numbers will probably not work with your hardware
 pin 12 is connected to the DataIn
 pin 11 is connected to the CLK
 pin 10 is connected to LOAD
 We have only a single MAX72XX.
 */
```

Some of the lines have to be edited to allow for all digits to be read, and also to lower intensity of display. I think also a component package (dark grey clear plastic bag) in front of the leds with intensity 1 is about right.

1.4 CPLD Programming

Using the XC9500XL series. This chip has some limitations - which are good.

As you get faster clocks, you need bigger registers to handle parsing the clocks. Bigger registers, use more power. Maybe this is one reason why high clock speeds mean more power.

1.4.1 6KHz clock

Due to limitations of the XC9500XL FPGA logic blocks, I ended up limiting the counter registers to 12+1 bits[2], so I have around 6,000 (assuming 60Hz), resolution. With this, I need a 6KHz clock. I could do this with the uno, but let's throw an attiny in there because it's a good tool for this kind of purpose and resolution. It should be able to function as a rough 6KHz timer, easily.

1.5 Divide by N Counters

The schematics appear to be incorrect for the divide by 6 counter in the Practical Electronics for Beginners book. Having looked at my built up circuit carefully, I see a 20Hz output from the 60Hz. I managed to get my hands on a copy of the TTL Cookbook by Don Lancaster recently, and that details correct divide by 6 and 10 counters (which are different from what's on my proto board), and while I could fix the divide by 6 counter, instead, I'm going to build another divide by 2 counter, and leave the original incorrect one there as a warning (it's also easier to just build a new one).

[2]Possibly I could use multiple smaller registers in a type of cascade, but let's not bother with that for now. I had 600KHz resolution, until I added the UART out/