ECE 271 – Microcomputer Architecture and Applications Lecture 23

Vince Weaver

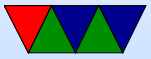
http://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

14 April 2022

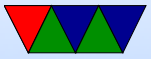
Announcements

- Read Chapter 21
- Turn in Midterm Take-Home Question #5
- Midterm will be graded eventually



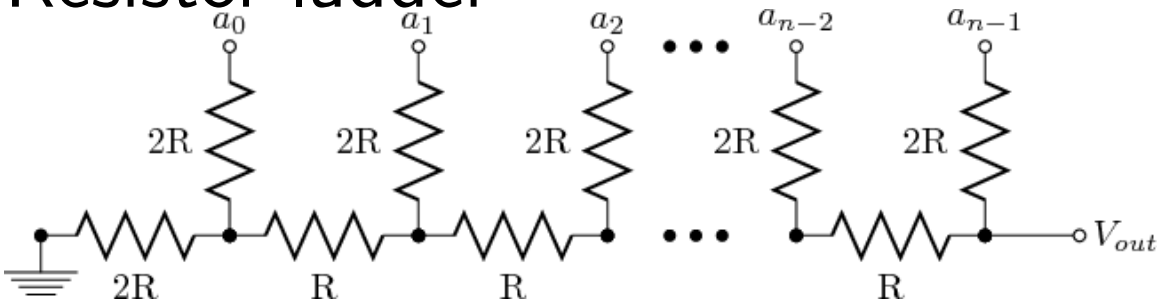
Digital/Analog Converters

- Opposite of Analog/Digital Converter



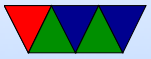
DAC circuit

- Draw one from the textbook?
- Resistor ladder



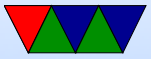
(wikipedia)

- Resistors designed to add voltages, $1/8 + 1/4 + 1/2$, etc.
- You can work this out using Thevanin



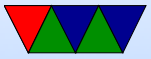
DAC Use

- Commonly used for sound card in your computer
- Also used for VGA graphics adapter (RAMDAC), old analog CRT monitors



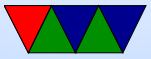
DAC custom circuits

- Story about how I made a sound card out of resistors and parallel port back in the day.
- You can bitbang VGA using GPIOs and resistors to create analog VGA signal on a Pi (uses lots of pins)



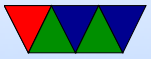
DAC Resolution

- Smallest change that can occur in the analog output.
- 5V 8-bit DAC, each bit increment is $5/2^8$=19.5mV
- CD audio has 16-bit



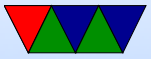
DAC Settling Time / Glitching

- Settling: Time it takes the update of output to settle to within X% of the wanted output.
- Glitch: if on updating the output it overshoots the desired level. “glitch” is often impulse area, area under the graph of glitch



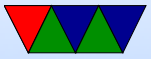
Sampling Frequency

- How many samples do you need to reconstruct a signal?
- Nyquist/Shannon Theorem says twice maximum frequency (there's a mathematical proof for this, a bit beyond this class)
- CD-quality audio at 44.1kHz as it's roughly twice max frequency your ear can hear



DAC on STM32L

- STM32L has two independent DACs
- Can be configured to 8 or 12 bits
- Can run together or separately.
  Why run together? (Stereo audio)

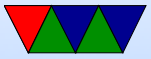


DAC on STM32L Registers

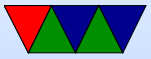
- DHR – data holding registers (DHR12R, DHR12L, DHR8R, DHR8L)
- The L/R is left aligned or right aligned

```
15      bit       0
|         XXXXXXXX 8-bit right align
XXXXXXXX          | 8-bit left align
XXXXXXXXXXXX      | 12-bit left align
|     XXXXXXXXXXXX 12-bit right align
```

- Why align? left align you can treat as a 16-bit sample

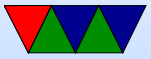


just with low bits missing.



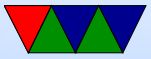
DAC on STM32L Other Features

- Can also add triangle wave and noise
- What is noise good for? Drum effects?



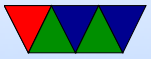
Conversion Trigger

- Can be triggered by hardware (a timer)
- Can be triggered from software, `DAC_SWTRIG`
  reset once data is loaded
- If both channels set to same trigger, will be triggered at same time



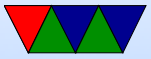
Buffered Output

- When connecting to external device (like headphones) the voltage might sag due to loading.
- If the headphone impedance is close to impedance of output, you get voltage divider and peak goes lower.
- You can enable an output buffer with high input impedance and low output impedance using the `DAC_CR_BOFF` bit in `DAC_CR` register.



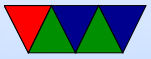
Lab#11

- Generate 440Hz sine wave (musical note A) out to pin PA5



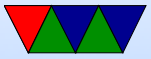
Generating a Sine Wave

- What does a sine wave look like?
- On a fast system with FPU you might do something like $output = (1 + sin(\frac{x}{180}\pi)) \times 2^{11}$
  where x is in degrees. Sine varies from -1 to 1 but we want output 0 to 4096



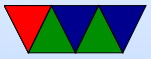
Generating a Sine Wave without sin()

- What to do without a sin() function?
  ARM doesn't have a hardwar FPU instruction so you have to do this in software.
- Can use Taylor series
  $sin(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} \approx x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040}$
- This can be a lot of math for an embedded system, is there a faster way?



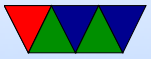
Generating a Sine Wave – table lookup

- Pre-calculate the values, and when time to output just look up in table.
- `output=sine_table[x];`
- Can even optimize (on low memory). sine is symmetrical, so only need 1/4 of it in memory and you can special case 0-90, 90-180, 180-270, 270-360



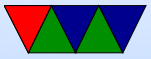
Code to generate a sine lookup

```
for(i=0;i<90;i++){
    sf=sin(M_PI*i/180);
    sine_table[i]=(1+sf)*2048;
    if (sine_table[i]==4096) sine_table[i]=4095;y
}
```



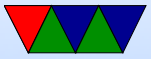
Running it to create header file

- Do this not on STM board, but on your devel machine
- Can write code that uses printf() to print valid C
- If using math.h (for sine) you might need to link with a math library (-lm on Linux)



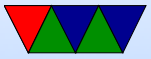
Using the Lookup Table

```
int lookup_sine(int degrees) {
    degrees=degrees%360;  // wrap around
    if (degrees<90) return sine_lookup[degrees];
    if (degrees<180) return sine_lookup[180-degrees]; //flip
    if (degrees<270) return 4096-sine_lookup[degrees+180];
    return 4096-sine_lookup[360-degrees];
}
```



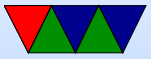
Note on machine without divide

- Make degrees lookup power of 2 (256?)
- Much easier to do a modulus (just AND)



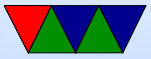
DAC Output

- DAC1_OUT1 goes to pin PA4 and DAC1_OUT2 goes to pin PA5.
- PA4 doesn't actually exposed on the STM32L board, it goes to an on-chip opamp and you can program this via OPAMP and route the output through it to PA3?
- Want to output 12bit audio at 44.1kHz?



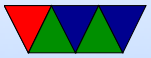
Set up Timer4

- $\frac{f_{HSI}}{(1+PSC)(1+ARR)} = f_{sampling} = 44.1kHz$
- PSC=18 and ARR=18 then you get 44.3kHz which is close



How do you get 440Hz?

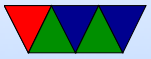
- $Stepsize = \frac{360degrees}{NumberofDACoutputsinsincycle}$
- $= 360degrees / \frac{periodofsine}{timeintervalofDAC}$
- $= 360degrees / (\frac{1}{f} / \frac{1/44.3kHZ}{)}$
- $= 360degrees / \frac{44.3kHz}{f}$
- For $440Hz = 360/44.3k/440 \approx 3.576degrees$



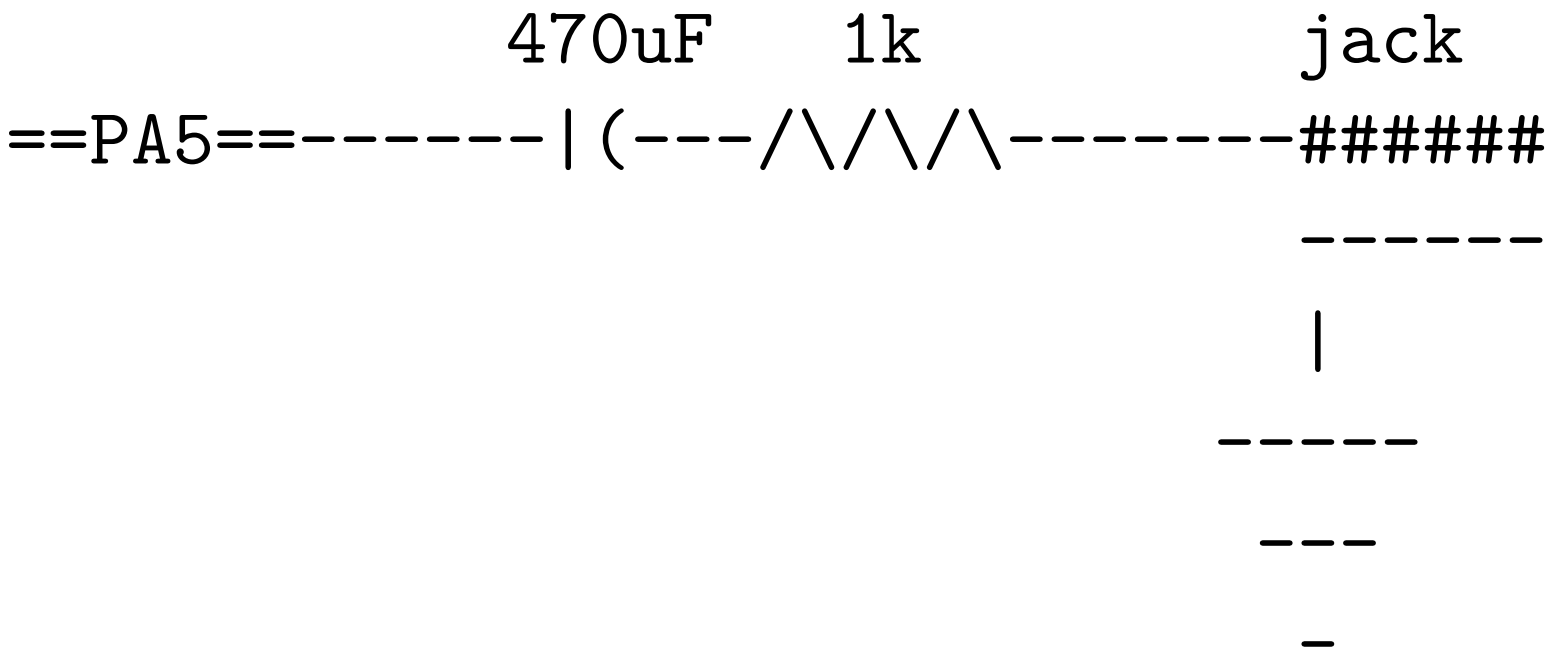
How do you handle decimals without FP?

- Use fixed point

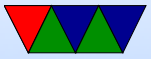
```
current_angle=0;
while(1) {
    output=sine_lookup[current_angle/1000];
    current_angle+=3.576;
}
```



Outputting to headphone/speakers?

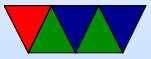


- Capacitor to strip off DC bias
- resistor to adjust impedeance



Digital Music

- Playing samples
- CD quality: 44.1kHz, 16-bit samples, stereo
- Need how many bytes/second?
- Compression, MP3
- How big would a song be?
  44100*2bytes*2channels= 176k/s
  2min song, 25.2MB
  pretty big! MP3 could get that down to maybe 4MB



Digital Synthesizer

- Square waves? Triangle waves? Noise?
- Attack, Decay, Sustain, Release
- Amiga MOD music
- Can you do multi-channel? Add the waves but then divide by number of channels