Today:

- Some administrative comments.
- Pulse width digital-to-analog conversion.
- Comments on op-amps and prototyping.

Many graphics from TI materials.

Develop a passion for learning. If you do, you will never cease to grow.
— Anthony J. D’Angelo
Tentative schedule

Week of June 1: Exercise 4, TI One-Day Workshop.
   Tuesday – controlSTICK ADC and DAC.
   Thursday – transfer function measurement.

Week of June 8: Exercise 5, controlSTICK ADC, DAC, xfer meas.
   Tuesday – discuss and decide on project topics.
   Thursday – The DFT and FFTs.

Week of June 15: Exercise 6, real-time FFT and waveform evaluation.
   Tuesday – FFTs using the controlSTICK.
   Thursday – Xilinx 8-bit PicoBlaze microcomputer (VHDL).

Weeks following —
   Lecture and lab complete, focus on projects.
Suggested schedule change

“Enrollment” seems to have dropped significantly over Memorial Day.

Suggest we drop the Tuesday lab period. Lab will be still be available and I and/or Chih-Wei will be on-call. The Thursday lab will remain staffed. The lab is otherwise available 24/7.

This is more consistent with my just-in-time Tuesday posting of the lab exercises.

I will be gone June 23 (Tue) through June 25 (Thu).
Projects

- Contain some aspect of DSP.
- Hopefully use S3SB FPGA and/or controlSTICK.
- If for credit, need to set up agreement on what is to be done and how to be graded.
- Can be individual or team.
- Need to get an idea of what, if any, additional resources might be needed.
- What else?
Exercise 5 evolves

My good intentions:

- Start with the following example code:
  - BlinkingLED,
  - SimpleDACtoADC,
  - FilteredHRPWM.
- Next will do direct digital synthesis.
- Transfer function measurement will involve:
  - DDS under interrupts,
  - Fourier shifting theorem.
  - Moving average filter.
  - $R/\theta$ calculation.
  - Plotting on the FPGA XVGA display.

Might be on the long side. No one is in a hurry. No deadlines.
The motivation being

Practice and making code available for projects.

- Use of the TI structures for I/O control. Lots of examples are available.
- The basic DSP paradigm is analog in, digital processing, analog out. The F28027 has a multichannel A/D, no D/A but four pulse width modulators. How do we use a PWM as a D/A?
- Working with the A/D and PWM.
- Use DDS for analog sinewave generator and for digitally computing $e^{-j2\pi ft}$.
- Application of the Fourier Shifting Theorem.
- “Sliding” average filter.
- Fixed point calculation of magnitude and phase.
- Linking the controlSTICK and the S3SB via a SPI channel.
- Display generation.
Reference material

- **Using PWM Output as a Digital-to-Analog Converter on a TMS320F280x Digital Signal Controller**, SPRAA8.
- Code examples included with the controlSTICK.
Version 1.0 is on the install disk that comes with the controlSTICK. TI’s SPRC835.zip claims to contain V1.0 but it actually contains V1.1. Version 1.1 is also available on the Workshop CD.

I’m assuming use of version 1.1.

The default install path is

C:\Texas Instruments\Piccolo controlSTICK.

You probably want to change this to a more personal directory.

Beware that the .cmd files appear to have been lifted from prior projects and may not accurately describe the F28027 memory. They work for the given application but …. 
Execute from RAM or Flash?

The example codes have two versions. One executes from flash memory and one executes from RAM.
To choose

To execute from RAM, on the top, second box from the left, choose F2802x_RAM.

I’m not how to establish the choice starting a new project. Changing the selection appears to be done by

- changing the active .cmd. There two built into the project, one for flash and one for RAM. Only one is active.
- The compiler option, FLASH is either used or not. If defined this includes the code below:

```c
#ifdef FLASH
    // Copy time critical code and Flash setup code to RAM
    // The RamfuncsLoadStart, RamfuncsLoadEnd, and RamfuncsRunStart
    // symbols are created by the linker. Refer to the linker files.
    MemCopy(&RamfuncsLoadStart, &RamfuncsLoadEnd, &RamfuncsRunStart);

    // Call Flash Initialization to setup flash waitstates
    // This function must reside in RAM
    InitFlash();  // Call the flash wrapper init function
#endif
```

Pulse width modulator DAC

DC level is \( f \times v_{cc} \).

Fundamental is at \( 1/\tau_p \) Hz.

Low pass filter to remove the fundamental and harmonics.

Vary \( f \) slowly (relative to \( \tau_p \)) and a D/A converter results.
This is a periodic pulse train having period, $\tau_p$.

(I unwisely used $f$ to represent fraction, $0 \leq f < 1$.)

$$f_p = 1/\tau_p \text{ Hz.}$$

The periodic pulse train, $p(t)$ has Fourier series representation

$$p(t) = \sum_{k=-\infty}^{\infty} c_k e^{j2\pi kt/\tau_p}.$$ 

For $\tau_p = 32 \times 16.7 \text{ ns}$, $f_p = 1.87 \text{ MHz}$.  

The spectrum consists of a set of lines spaced $1.87 \text{ MHz}$ apart.
Enhanced Pulse Width Modulator Module

- Piccolo 28027 has 4 channels.
- 145 page manual (SPRUGE9).
- Comparable in complexity to a microcomputer of 10 years ago.
- Counts the CPU clock, 60 MHz.
- Generates periods multiples of 16.7 ns.
ePWM subsystems

Figure 1-2. Submodules and Signal Connections for an ePWM Module
High Resolution PWM

- An enhancement to the ePWM.
- Extends the time resolution.
- Nominally divides 16.67 ns clock cycle into 111 subintervals of 150 ps.
- In effect, increases the interval resolution by almost 7 bits.
- Temperature calibration capability provided.
- Each 28027 ePWM channel has this enhancement.
- 36 page manual, SPRUGE8.
High resolution timing

Figure 2. Operating Logic Using MEP

PWM period (N CPU cycles)

PWM duty
(0 to 1.0 in Q15 format)

MEP scale factor
Number of MEP steps in one coarse step

Coarse step size (1 SYSCLK cycle)

Number of coarse steps = \text{integer}(\text{PWMduty} \times \text{PWMperiod})
Number of MEP steps = \text{fraction}(\text{PWMduty} \times \text{PWMperiod}) \times (\text{MEPScaleFactor})

16-bit CMPA register value = number of coarse steps
16-bit CMPAHR register value = (number of MEP steps) \ll 8 + 0x080 \text{ (rounding)} ^{†}

^† For MEP range and rounding adjustment.
Setting the integer width

Integer part: Multiply the period in counts by value in the range [0, 1).

Samples are 12-bits and are in the low 12-bits of a 16-bit word. These are in simple binary form. (0V input maps to 0 and $V_{cc}$ maps to 4095). Can model as being Q12.

The integer width count is the period count times the sample value with the result right shifted 12 places. (Probably need to use a long for the product.)

If the period count is a power of 2, say a count of 32 giving a power of 5, then simply arithmetic shift the sample value by 7 bits.
Setting the fractional width

Using the 12-bit sample and period count of 32.

We need to multiply the high resolution step count. The hardware can be used to determine this. It is sensitive to temperature and voltage. A reasonable guess value is 111.

Having the period count times the sample value. Mask off the top 9 bits keeping the low 7 bits. This is the fractional part in Q7. Multiply by 111. Shift right by 7. This is the fractional count part for setting high resolution.

I think this is correct. Actual code will differ because of how the bits are arranged in the control words. Look before you leap.
The controlSTICK RC filter

\[ R = 470 \text{ Ohms}, \quad C = 10\, \text{nF}. \quad RC = 4.7 \times 10^{-6}. \]

\[ f_{3\text{dB}} = \frac{1}{2\pi RC} = 33.9 \text{ kHz}. \]
Modeling the ripple

\[ V_{cc} \quad R \quad C \quad 0 \quad R \quad C \]

\[ \nu_{cc} \quad \nu_b \quad \nu_a \quad \text{GND} \quad \text{time} \]

\[ f \tau_p \quad (1 - f) \tau_p \]

\[ 0 \leq f < 1 \]
Writing the equations

Assuming steady state. Want to determine \( v_a \) and \( v_b \) as a function of the fractional value \( f \) where \( 0 \leq f < 1 \). Because our introductory circuits course far in the distant past we can immediately write

\[
\begin{align*}
v_b &= v_a + (v_{cc} - v_a)(1 - e^{-f\tau_p/\tau_{RC}}), \\
v_a &= v_b e^{(1-f)\tau_p/\tau_{RC}}.
\end{align*}
\]

where \( \tau_{RC} = RC \). Solving for \( v_a \) gives

\[
v_a = v_{cc} e^{f\tau_p/\tau_{RC}} \frac{1 - e^{-f\tau_p/\tau_{RC}}}{e^{\tau_p/\tau_{RC}} - 1}.
\]

The equation for \( v_b \) follows readily. The plots on the following page were made using MATLAB.
Ripple level plots

![Graph 1: Va (Volts) vs Duty cycle fraction]

![Graph 2: Peak-to-peak ripple (Volts) vs Duty cycle fraction]
Some numbers

The RC values corresponded to those used on the controlSTICK.

- $\tau_p = 32/60E6$, i.e., 32 controlSTICK clock tics.
- The supply, $v_{cc}$ was set to 3.3 Volts.
- The maximum peak-to-peak ripple is 0.0936 Volts.
- The gain at frequency $1/\tau_p$ is 0.0181.
- The peak-to-peak amplitude of the fundamental at the RC output is 0.076 Volts for $f = 1/2$. 
A better filter

Might try one more RC stage.

Use active filter built using an op-amp.

What op-amp, what filter topology, what parameters?
TI OPA340 op-amps

Rail-to-rail input
Rail-to-rail output
Gain bandwidth: 5.5 MHz
Slew rate: 6V/µs
$V_S = 2.7V$ to 5V

Available in 8-pin DIP package. The OPA2340 costs $2.03 quantity 25.
TI OPA350 op-amps

Rail-to-rail input
Rail-to rail output
Gain bandwidth: 38 MHz
Slew rate: 22V/μs
$V_S = 2.7V$ to $5.5V$

Available in 8-pin DIP package. The OPA2350 costs $2.64 quantity 25.
Who makes op-amps?

- Analog Devices,
- Linear Technology,
- National Semiconductor,
- Texas Instruments.
- Others manufacturers exist as well.

Didn’t look for 0.1” DIP packages.
Might be able to squeeze out a bit broader bandwidth.
Too much bandwidth can be a problem itself.
Noise performance is important in many applications.
We have used a myopic bandwidth looking glass.
Not interested in any chips with minimum supply needs > 3 Volts.
Those holey white blocks

Hole mount parts are vanishing… adapt the surface mount parts!

Spark Fun $1.95.
Needs pin strips.

Or, you can adapt your approach!
Dead bugs on perf board
Need a steady hand — 0.05" spacing
Power supply bypass capacitors

- Absolutely essential.
- Charge reserve used by switching devices.
- Prevents (generally) analog amplifiers from oscillating.
- Place as close to the supply pins as possible.
- Use 0.01 $\mu$F minimum. I always use 0.1 $\mu$F.
- Also 0.1 $\mu$F bypass any bulk supply storage electrolytics!!!! When is a capacitor not a capacitor? When it’s an electrolytic.
- Place as early as possible when laying out PWB board!!!!

- Do not place ground planes under amplifier summing junctions!!!
Single-supply Sallen-Key filter?

TI offers assistance:

Two-pole, “low pass”, Sallen-Key filter:

R3=R4 (HIGH)
F0=1/(2πRC)
R1=R2=0.707R
C1=C
C2=2C

Circuit from the TI single supply expert.
More Sallen-Key arrangements

LOW PASS
Unity Gain Butterworth
\[ F_0 = \frac{1}{2\pi RC} \]
\[ R1 = R2 = \frac{R}{\sqrt{2}} \]
\[ R3 = \frac{R}{2\sqrt{2}} \]
\[ C1 = C \]
\[ C2 = 4C \]

HIGH PASS
Unity Gain Butterworth
\[ F_0 = \frac{1}{2\pi RC} \]
\[ R1 = 0.47R \]
\[ R2 = 2.1R \]
\[ C1 = C2 = C3 = C \]

BAND PASS
Gain = 2.3 dB
\[ F_0 = \frac{1}{2.32\pi RC} \]
\[ R1 = 10R \]
\[ R2 = 0.001R \]
\[ R3 = 100R \]
\[ C1 = 10C \]
\[ C2 = C \]

From TI's collection, SLOA058. Available on handouts page.
Delta Sigma modulator DAC

a.k.a pulse density modulation DAC.