The trials and tribulations of building a phase-sensitive detector with an Arduino microcontroller

K. D. Schultz

Department of Physics
Hartwick College

MAS-APS, 2014

Original Goals

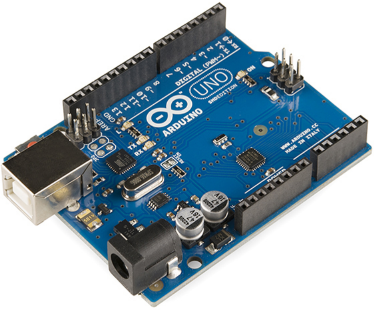
- Use Arduino as a tool for teaching about phase-sensitive detection.
- To do so with only the Arduino, a computer for display purposes, and passive external components (resistors and capacitors)

Why PSD?

- Phase Sensitive Detection (PSD) is the basis of many techniques in physics and engineering
  - Homodyne detection
  - Interferometry
  - Lock-in amplifiers
- Black boxes are useful for application work, but not so much for pedagogical purposes
- Software PSD allows students to peek into the black box.

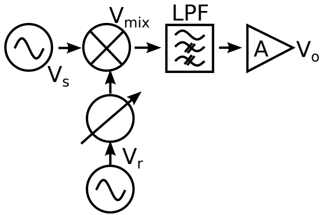
Why Arduino?

- Cheap
- Popular
    - Lots of support
- Simple programming environment
    - Perhaps too simple, IDE has very poor debugging tools.
- Works well with Processing, which is a free and powerful language for visualization
- Both Arduino and Processing are platform agnostic: Windows, Linux, OS X, Raspberry Pi...



Arduino picture from Sparkfun CC-BY-2.0

PSD Basics



Mathematics of PSD

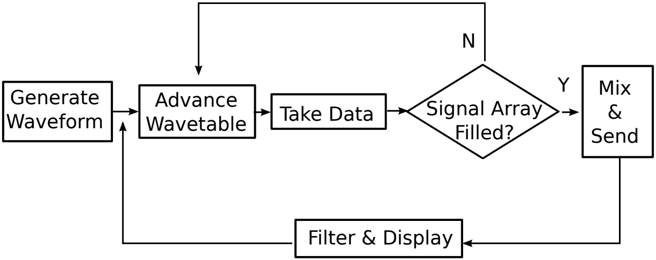
$$V_{mix} = V_s V_r \left[\left(\cos\left(\omega_s - \omega_r\right) t - \left(\phi_s - \phi_r\right) \right) \right]$$

$$V_o = A \frac{V_s V_r}{2} \left[\cos(\phi_s - \phi_r) \right]$$

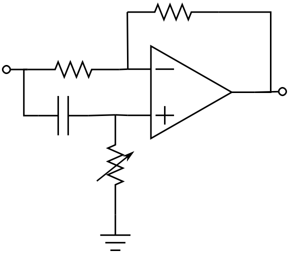
Restrictions

- $\omega_r = \omega_s$
- V_s and V_r have no DC offset

Flowchart

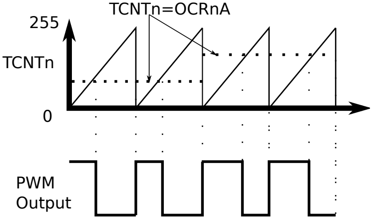

Generate Waveform
Advance Wavetable
Take Data
Signal Array Filled?
N
Y
Mix & Send
Filter & Display

Phase Manipulation



- Need to be able to shift phase to maximize signal
- All-pass phase shifters work, but require a bit more hardware external to Ardunio
- Software solutions difficult to implement
- Use two phase detection.
  - $V_I = V_s \times \cos(\theta)$, $V_Q = V_s \times \sin(\theta)$
  - $R = \sqrt{V_I^2 + V_Q^2}$
  - $\phi = \tan^{-1}(V_Q/V_I)$

Creating a Reference Signal

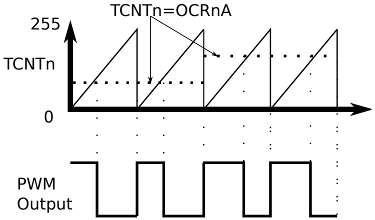


Picture adapted from J. Thompson, MAKE vol. 35

Timers and Interrupts Part I

- The ratio of PWM on to off determines an average “DC” signal
- When register TCNT1 reaches OCR1A PWM goes low
- When TCNT1 overflows PWM goes high again

Creating a Reference Signal



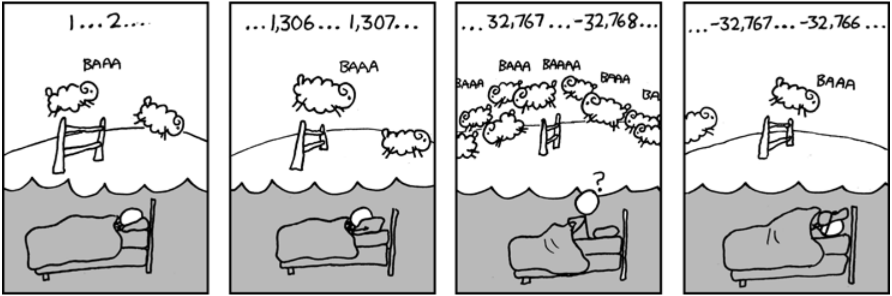
Picture adapted from J. Thompson, MAKE vol. 35

$$f_{ref} = \frac{\text{TCNT2 rate}}{\text{OCR2A value} \times \text{wavetable length}}$$

Timers and Interrupts Part II

- Need fast timer2 and regular timer1, which outputs PWM
- When timer2 reaches OCR2A:
  - Update OCR1A from wavetable
  - read signal at AnalogIn
- When timer1 counts up to OCR1A, PWM goes low
- When timer1 overflows PWM goes high again

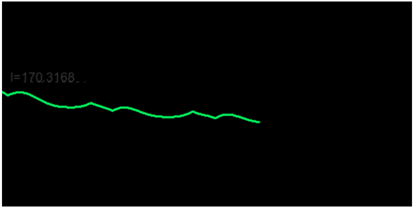
Signal Input



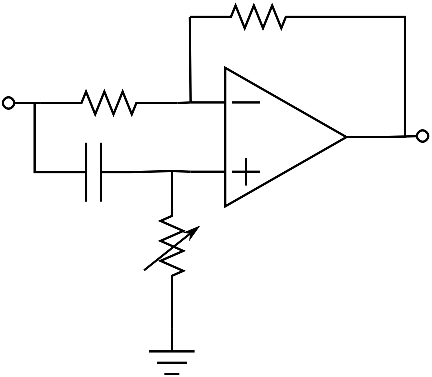
Signal Input

- Arduino can only have positive voltages at its inputs or outputs
  - Integer overflow when mixing
  - Necessary DC offsets makes phase indefinite
- Need to run ADC as fast as we can so as to to interfere with reference generation.
  - Setting pre-scalars and registers can get sampling rates of 50k samples/sec. Not bad for a 30 board!
  - ADC has 10 bit resolution with adjustable V_{ref}, so resolution on the order of a millivolt is possible

Display



In-phase channel as phase is changed



What we accomplished

- Built and tested a working two-channel phase-sensitive detector
- Learned techniques that can be used for other micro-processor based instruments
- Published on github under GPL v3 license

Still to do

- Characterize detector (noise, internal phase shift, etc...)
- Clean up display
- Explore other memory options on Arduino
- Use in an application

For Further Reading I

- The programs!
  https://github.com/HartwickChaosLab/Arduino-Phase-Sensitive-Detector
- Arduino
  http://www.arduino.cc
- Processing
  http://processing.org
- Moding the Arduino ADC
  https://sites.google.com/site/measuringstuff/the-arduino
- J. Thompson
  “Advanced Arduino Sound Synthesis"
  Make Vol. 35