SONAR PROJECT

Test Bed Design

* Oscillator board
* Phase difference board
  + phase shift board to simulate pinger location
  + function generator tests

Analog Board

* ~~Hydrophone maintenance~~
  + ~~(3x) RESON TC4013 Hydrophones~~
  + ~~Spaced 24mm apart in triangle at bottom of sub~~
* ~~Pinger~~
  + ~~Output: 22, 23, 24, 25, 26, 27, 28, 29 or 30 kHz~~
  + ~~Pings at 0.5 Hz (2 seconds) for 1.3ms with a sound level of 187 dB~~
  + ~~Separated by 0.9 seconds from practice pool ping~~
* Filter Design
  + Switched Capacitor Filter
    - Narrow bandpass for 20Khz to 40Khz range
    - Controlled by clock frequency in the 0-40Mhz range
    - MAX263
      * Supply voltage: 15V
      * Clock input (2.4min High, 0.8max Low)
  + Infinite Gain Amplifier
    - opamp
  + Phase Difference

Digital Board

* Microcontroller Board (interface with phase diff)
* Programmable Oscillator
  + Maxim DS1077, DS1085
  + Frequency range of 66-133MHz, adjustable over the I2C connection in 10KHz intervals
  + This is fed into a programmable divisor of 1-1025, making for a wide range of available output values
  + Pinout, voltage/current requirements
* Carrier Board >> Arduino >> Progammable Oscillator >> Switched Capacitor Filter
* Microcontroller software (Arduino)

ROS Sonar Node

* Messages (Dave’s implementation)
  + Pinger ID (#0 correct, #1 incorrect)
  + Heading
  + Magnitude
  + TimeSince
* PingerArray

Pinger >> water >> hydrophones >> VGA >> filter >> ADC >> Cheetah SPI >> ROS Sonar Node

Pinger >> water >> hydrophones >> preamp >> filter >> opamp >> ADC >> Cheetah SPI >> Sonar Node

Pinger >> water >> hydrophones >> preamp >> filter >> >> opamp >> phase comparator >> ADC >> Sonar Node

~~Pinger operation (turn on, set frequency, mount underwater, two pingers, interference)~~

~~Signal capture through hydrophones/connecters (compare/test signal amplitudes, signal degradation?)~~

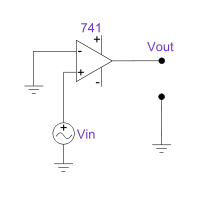
~~Pre-amp to boost signal for filter? VGA. Signal conditioning~~

Signal filtering (test for filtered output, add noise)

* Switched cap filter input (function generator sine wave at 25kHz, 100mV)
* Clock input (function generator, square wave at 1MHz, 5V, interface with oscillator after)
* Variable frequency oscillator
* Supply voltage, control words (pin programmable with Arduino), operating modes, Qs, fclk/f0 ratio, cascading?
* MAX7490

Amplification

* Input (func gen to simulate filtered signal)
* Op-amp (gain to clip signal and turn into square wave), differential amplifier for low noise
* Zero crossing detector, rectifier, multilateration



* San Diego City used 100X gain

Filtered signal acquisition (ADCs, Cheetah sampling)

* Hook up ADC to func gen (DC voltage, sinusoid, square wave)
* Arduino to take in bytes from ADC (plot and verify)
* Repeat with Cheetah at faster sampling rates
* AD7865 simultaneous sampling four channel 14-bit ADC
* Clipped input from amplifier does not exceed voltage requirements? Clamping? Op-amp supply voltage
* FFT to find phase difference

Phase comparator

* Outputs a PWM signal whose pulse width is proportional to the difference in phase between the two signals
* MAX9382
* PWM signal goes to microcontroller to determine how long the pulse width is
* Pulse width is converted to a left or right degree error (directional error) by a look up table
* San Diego city uses 4 hydrophones mounted at 45 degrees to vertical, to determine when sub is right above the pinger
* PLL phase locked loop
  + A type II detector is sensitive only to the relative timing of the edges of the input and reference pulses, and produces a constant output proportional to phase difference when both signals are at the same frequency.

Ping detection (measure length of time, bin, windows)

* Analog voltage threshold triggers sampling and storing in serial memory
* Cross-correlation algorithm is performed on binned signal return samples to find phase differences
* Wait for min amplitude, accumulate samples, apply band pass filter on samples, calculate phase difference between hydrophones (reference channel and remaining), send phase difference, compare in lookup table for heading and elevation (value in the table with smallest difference)

Phase difference analyzer (angle-of-arrival calculation, closest hydrophone pair, 3rd hydrophone)

* From each hydrophone pair, two candidate rotation angles are found. The two angles which are closest to each other gives the final estimate of rotation needed to aim toward the pinger

Localization (sensor data for direction, phase difference tells you distance from source)

Test on SeaBee

Run 3-channels, triangulation

Analog filtering, digital signal processing

Cornell can compute heading and elevation to the pinger within one degree (4 hydrophones)

Hyperbolic positioning to determine directional vector towards the pinger

Distance between hydrophones is less than half the wavelength, to ensure unambiguous phase difference 🡪 along with physical placement of h-phones and speed of sound underwater, can determine the time difference and calculate angle of sound source

FPGA? DSP?

repurpose VGA for pressure sensor?

Parts lists and cost breakdown

Break up into smaller tasks (amplifier, phase comparator, ADC, find pinger)

**Jose**  
***Current Milestone***: Finish bandpass filter  
***Date:*** October 11th

**Dylan**  
***Current Milestone:*** Test bed oscillator  
***Date:*** September 20th

I'll start with the system as it was for the competition last year:

**Hardware:**

* (3x) RESON TC4013 Hydrophones (<http://www.reson.com/wp-content/uploads/2010/12/TC4013.pdf>)
* Cheetah SPI Host Adapter (<http://www.totalphase.com/products/cheetah_spi/>)
* LTC1400 12-Bit, 400ksps ADC (<http://cds.linear.com/docs/Datasheet/1400fa.pdf>)
* Custom PCB (Beosub Repository)
  + trunk/Electrical/Eagle/Wave Packet Sonar/SonarCheetah.brd
  + trunk/Electrical/Eagle/Wave Packet Sonar/SonarCheetah.sch

**Signal Path:**

Pinger  >>  ~~~ water ~~~  >>  hydrophones  >>  ADC  >>  Cheetah SPI  >>  ROS Sonar Node

**Software:**

* Initial Dev
  + Saliency Repository
    - src/Robots/SeaBeeIII/Sonar
* Current
  + ROS Node (<http://code.google.com/p/seabee3-ros-pkg/>)
* FFTW3 - FFT Library (<http://www.fftw.org/>)

**Walkthrough:**

* The initial dev software folder contains all of the testing software and development/experimentation code I initially used to determine how to approach coding the phase difference algorithm, frequency testing, Cheetah communications, etc...
* The ROS node is the most current version of the code (originally written for Box Turtle, but should be ok with C Turtle as well)
* Cheetah SPI (as configured in ROS node code)
  + **NOTE:** At the time only the 32-bit pre-compiled shared library was provided, so be aware of this when running / compiling on 64-bit systems
  + Communicates in asynchronous mode to keep actual sampling rates up
  + "True" sampling rate is calculated based on the speed of the Cheetah at runtime using a linear interpolation method from experimental data (signal generator, spectrogram, and max frequency bin identifier)
    - Initially I tried calculating this real-time in the principle loop but had major inaccuracies
    - Talked with Rand & Edward about it, but didn't come up with a function fix / solution, so went with the existing approximation
  + Uses slave select (SS) signals as convert (CONV) signals for ADCs
  + Generates a batch of conversions, which will occur in rapid succession without interruption once submitted
  + The collected data stream comes back as individual bytes, but the ADC output data is a combination of two consecutive bytes and must be combined so relevant information can be extracted (data, ready bit, etc.)  See ADC data sheet for spec and code for implementation
  + The extracted (valid) data is stored in their respective signal buffers corresponding to the 3 hydrophones
* Ping Detection
  + Each buffer is examined with a short sliding window for significant frequency content in the expected FFT bin
    - Since the window is short, the frequency resolution is poor but the temporal resolution is high so a full ping can be isolated in time with reasonable granularity
  + A running sample mean is kept to evaluate whether or not the current value of the FFT bin is large enough to be considered a peak
  + If peaks are found in all channels, an indicator is noted for that time step
    - This continues until the current time step does not have an indicator
  + Once a peak is not detected, the run of indicators is checked for sufficient length (i.e. was it a full ping, or just a small error)
  + The history data corresponding to all blocks marked with a positive indicator are extracted an an FFT is performed on the full ping sequence
    - The increased length of the FFT will provide more accurate frequency resolution and allow the phase information from just the ping frequency to be extracted
* Angle Calculation
  + An optimal-pair phase difference angle-of-arrival (<http://en.wikipedia.org/wiki/Angle_of_arrival>) algorithm was developed by Rand and I for relative angle estimation of sonar pingers
  + The best hydrophone pair is determined by finding the minimum phase difference between hydrophone pairs
  + The phase difference is adjusted for a phase factor (rear vs. front) using the remaining hydrophone
  + The phase difference is adjusted for an angular offset base on the chosen hydrophone pair
  + Angle estimate is calculated using the physical separation of the hydrophones

**Problems:**

* Signal strength
  + When the cables were redone just before the competition, the signal strength dropped significantly to the point that the peak detection stopped working
  + The cable attachements need to be redone and the analog front end (following section) will help with this
* Signal quality (SNR)
  + Without hardware filtering, the only signal isolation occurs through the FFT
  + Ambient noise and common hydrophone grounds cause significant phase distortions that decrease the accuracy of the angle estimate

Now the more recent work & future directions:

**Proposed New Hardware (Experimental):**

* LTC1569CS8-7 - Linear Phase, DC Accurate, Tunable 10th Order Lowpass Filter (<http://cds.linear.com/docs/Datasheet/15697fs.pdf>)
* AD8331ARQZ-ND - Single Variable-Gain Amplifier (VGA) with Ultralow Noise Preamplifier and Programmable RIN (<http://www.analog.com/static/imported-files/Data_Sheets/AD8331_8332_8334.pdf>)
* LTC1403ACMSE-1 - Serial 14-Bit, 2.8Msps ADC (<http://www.linear.com/pc/downloadDocument.do?navId=H0,C1,C1155,C1001,C1158,P2481,D1475>)
* Simple Arduino / Amtel chip to sample the signal path & control the VGA

This set of hardware was my initial suggestion for a revised analog frontend that filters out high-frequency noise (tunable filter), keeps the signal amplitude high (VGA), and samples with more precision at higher frequency (new ADC).  I'm happy with the idea, but I think to really address the problems we had with the sampling we need to use differential pair hardware and separate the grounds of the signal channels, which should remove a significant amount of interference; Since the signal paths would be independent differential pairs, the cross-talk & external noise will essentially be removed.  More research needs to be done to figure out what hardware should be used.

**Signal Path:**

Pinger  >>  ~~~ water ~~~  >>  hydrophones  >>  **VGA  >>  filter  >>  ADC** >>  Cheetah SPI  >>  ROS Sonar Node

DIAGNOSTIC BOARD