6.111 Project Proposal – Digital Sonar
Submitted by Bryan Morrissey, Zhen Li, and Brian Wong
November 2, 2007

Project Overview

Our team will design and implement a digital sonar system based on a single high-power ultrasonic transmitter and an array of receivers; the receiver array will allow us to detect and track the distance and direction to multiple objects in a two-dimensional plane around the array.

The digital sonar project will be broken down into three main sections to be handled by the three members of the team. Bryan Morrissey will be responsible for designing the digital logic to interface with the ultrasonic generation and detection hardware, as well as the separate project of designing and constructing the analog circuitry to interface with the ultrasonic transducers. Zhen Li will be responsible for designing the digital signal processing system to perform the echo signal analysis and environmental modeling. Brian Wong will be responsible for designing and implementing the graphical display and user interface.

(Full Project Block Diagram to be included here...)

Pulse Generation and Signal Detection Block (Bryan Morrissey)

The critical parts of the electronic interface are the ultrasonic pulse generator and the signal detectors. The pulse generator produces 40kHz signal with a controllable number of cycles per pulse and a variable repeat rate. The pulse can be made longer in duration if to improve signal integrity, or it can be made shorter if the signal is strong enough to improve our ability to resolve multiple objects at different distances without echo overlap. Likewise, the pulse repeat rate can also be varied automatically; if all echo signals correspond to nearby objects with short echo times, the system can send shorter but more frequent pulses to improve its ability to detect multiple nearby objects.

The signal from the pulse generator will be amplified by a differential bridged power amplifier circuit. This amplifier design can be built into the 6.111 labkit and produce a 40kHz signal at close to 20V rms while running on the kit's +/-12V power supplies. This is to ensure that we make full use of the capability of the ultrasonic transmitter.

The analog part of the signal detection circuitry will consist of some basic amplification and low-Q bandpass filtering; the idea is do ensure that there is a clean and strong signal to detect while leaving the bulk of the processing to the digital modules implemented on the FPGA.

The signal detection modules will perform synchronization, down-sampling, and digital filtering on up to twelve input channels before buffering the bitstreams for the signal analysis module. Currently we are planning to implement a set of basic one-bit analog to digital converters using a Delta-Sigma topology, similar to those described in lecture (Lecture 11, Slides 16 and 17; <u>not</u> the "Poor Man's ADC"). This has the advantage that it will allow us to implement many more analog input channels than we could do using the AC97 codec, while also providing much higher bandwidth for more precise timing and phase measurements. Instead of implementing a network of high sample-rate serial ADC's to interface with the FPGA, the FPGA itself will serve as an integral part of the analog to digital conversion hardware.

The signal conditioning will consist of a digital low-pass filter and decimation system; this will convert a sequence of one-bit samples at roughly 2 to 4 MHz to a multi-bit data stream at several hundred kHz. High sample rate is more important than signal resolution in this application, but the nature of the Delta-Sigma ADC topology requires that it be down-sampled

and digitally filtered in order to produce a useful, low-noise signal.

Finally, the signal generation and detection block will implement an RS-232 interface that will allow us to acquire and analyze test data from the system on a separate computer. This will also provide a vital debugging and test interface for the system before all of the blocks are ready to be integrated during the final stages of the process; the raw test data acquired in this way can also serve as test data for the other modules as they are being developed.

(Include sub-block diagram for signal generation and detection along with preliminary circuit designs)

Sections: (Some details might be expanded as a discussion section in the report)

- Output pulse: digital signal generation and power amplifier circuitry
- Input signal detection – signal amplification, filtering and A/D conversion
- Discussion of the the two different strategies that we considered
  - Reasons for pursuing far-field R-Theta strategy in the near term
  - Possibility of investigating near-field strategy if signals are clear enough
- Different A/D conversion strategy depending on the near-field vs. far-field strategy
  - far field strategy for r-theta measurement of a limited number of objects
  - near-field strategy for 2-D representation of 3-D space in front of the panel
  - ADC: system requirements and design determined by modelling strategy
    - 2-D just needs binary yes-no for spaces n a 3-D grid
    - R-Theta needs high sample rate for FFT resolution (what will be required?)
- Include request for transmitters, receivers; include data sheet as an appendix
- Describe initial proposed plan to build a test signal generation and detection interface
- Block diagram for detection interface
- Output module: programmable pulse duration (wavelength count to ½ wavelength) and programmable cycle / repeat time
- RS232 serial port to dump data will enable us to do module testing, hardware evalution and data analysis during the pre-integration phase of the project.

Signal Processing Block (Zhen Li)

Description

The signal processing block takes pre-conditioned binary streams from the signal detection module as inputs, and provides the distance and angle information of objects as outputs. The binary streams from the pre-conditioning module represent the reflected signal received from microphones at a sample rate of 1 bit/us. The distances (r) between the transducer array and the objects are calculated from the emit-to-receive time delays. The direction angles (θ) are retrieved from the phase delays among receivers. Finally, a series of distances and angles are passed on to the graphics and user interface module.

Input/Output Interface

The signal processing block reads buffered data from the signal detection and preprocessing module. The preprocessing module stores one binary stream per receiver in the BRAM. When a READY signal is detected from the pre-conditioning module, the signal processing block reads new data from the BRAM.

The signal processing block updates a set of (r, θ) coordinates thirty times per second.

Once all calculations are done, it passes the coordinates to the graphical interface module through a register array. At the same time, a READY signal is asserted.

Retrieving the direction

Retrieving the angle θ to each object is the most challenging part of the signal processing block. The signal reflected from an object arrives at each of the receivers (R_1 to R_4 in Figure 1) at slightly different times. More precisely, a signal which arrives at R_i at time t will arrive at receiver R_{i+n} at time $t+n\Delta t$, where Δt is given by:

$$\Delta t = \frac{d\cos\theta}{c}$$

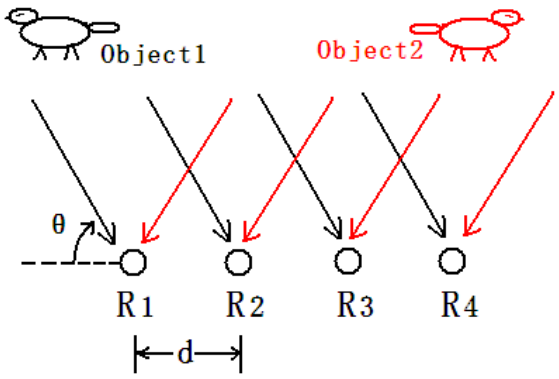

Figure 1: Receiver Array Configuration

where c is the speed of sound in air, 340 meters per second at sea level.

The main challenge will be to detect and measure multiple values for Δt when there is more than one object in view. (Figures 1, 2)

A spatial FFT is introduced to solve this problem. If the received signals are shifted by the proper time interval, namely multiple Δt, they add up to a sharp peak which indicates an incoming signal in the corresponding direction. (Figure 3)

Mathematically,

$$f(\Delta t, t) = R_1(t) + R_2(t+\Delta t) + R_3(t+2\Delta t) + R_4(t+3\Delta t)$$

$$f(\theta, t) = R_1(t) + R_2(t+d\cos\theta/v) + R_3(t+2\mathrm{d}\cos\theta/v) + R_4(t+3\mathrm{d}\cos\theta/v)$$

The peak(s) of $f(\theta, t)$ indicate the incoming reflected signal direction(s).

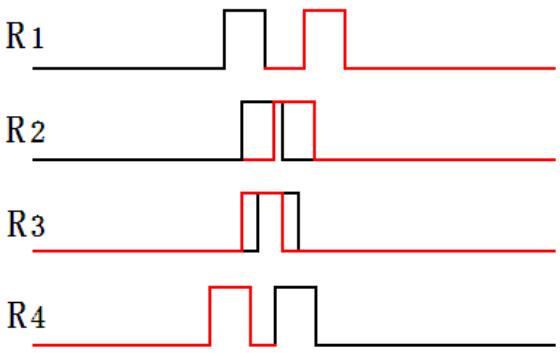

Figure 2: Received Signal (Unprocessed)

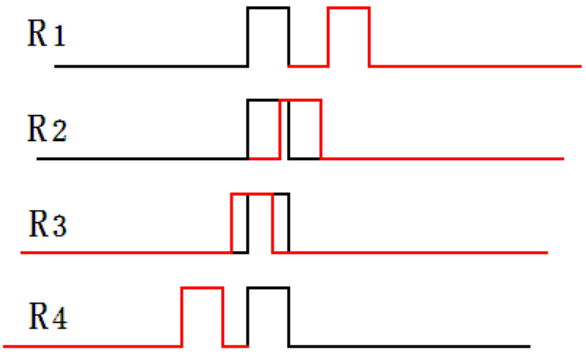

Figure 3: Received Signal (Shifted)

Retrieving the distance

Once the directions of the signals are determined, calculating the distances is just simple math. The distance r from the transducer to the object is retrieved from the time delay from transmitted signal to the peak of received signal t_d, $r = ct_d / 2$

User Interface Block (Brian Wong)

User Interface Modules

In the user interface module, we are aiming to display the incoming sensor data on the a VGA monitor using the 6.111 lab kit VGA signal output capability.We will also utilize the signal output capability of the lab kit to generate an audio tone if the system detects objects within range.

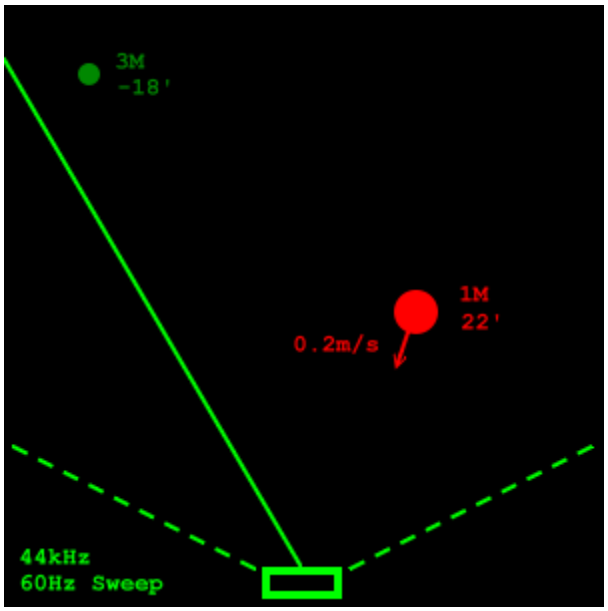

Figure 4: A Proposed look and feel for the graphical user interface

The system has several key features:

1. Graphical representation of sensor-covered area
   The VGA output provides a basic representation of the area covered by the far-field sensor. There are two basic components of the system that are drawn by default:
   I. A representation of the far-field sensor location. This component is drawn at the center bottom of the screen. The green rectangle in Figure 1 represents the sensor. This component can also be represented by other forms, such as a bitmap image imported from the computer to the FPGA memory.
   II. Boundary lines that indicates the covered area of the sensor. The two dashed lines originating form the sensor provides an angular limit on the area covered by the far-field sensor. This line changes depending on the configuration of our sensor arrays.

2. Indication of object distance and angle with respect to sensor
   I. This is represented by two numerical values that appear next to each detected object. The distance represents the meter distance between the object (Approximate range: 1M – 5M) and the sonar sensor, while the angle represents the angle the object makes with the vertical Figure 1 depicts two objects, each containing a distance (in meters) and an angle (in degrees). The angle ranges from -25 to +25 degrees.

3. Color representation of object
   The color of the object can represent the distance between the sensor and the object, i.e. near = red, far = blue. The color can also represent the velocity of the object, i.e. fast = red, slow = blue.

4. Size representation of object
   The size o the object can represent the distance between the sensor and the object, i.e. near = bigger, far = smaller. The color can also represent the velocity of the object, i.e. fast = bigger, slow = smaller.

5. Velocity vector indication of object
   Ideally, we would like to include an indicator that shows the speed and direction at which an object is moving. We want to achieve this by showing a velocity vector attached to each object on the screen. A speed (in m/s) indicates an estimated rate of movement.

6. An indication of the current settings within the system
   The module can indicate sweep rate and frequency settings of the sonar system in the bottom left of the screen.

7. Sweeping motion of sonar line and audio output
   In order for the system to best simulate the functionality of a sonar, we can implement a pseudo sweep line that scans across the detectable area. Every time the sweep line intersects with an object on the screen, an audio signal is initiated to indicate the object's presence.

Other Features of the User Interface Module

1. The interface will provide a ready signal that tells the graphical interface to redraw locations of the detected objects.

2. The graphical module can fetch location data (limited to 10) from the signal processing module (memory or register based).

3. The graphical module can accept both circular and rectangular coordinates.

4. The graphical module works with the same clock frequency as the rest of the system.

5. The user interface module provides an independent testing platform that can show case all the features of the graphical design without the functionality of the signal processing module or the sonar hardware.