Passive Sonar

Project Design

Eric Chang
Mike Ilardi
Jess Kaneshiro
Jonathan Steiner

Introduction

In developing the *Passive Sonar*, our group intendes to incorporate lessons from both Embedded Systems and E:4986, the latter of which three of our members are currently enrolled in. Our device works by triangulating the location of a sound source along a horizontal axis by calculating the difference between the arrival times in two spaced microphones. A marker projected onto a screen will represent calculated location of the sound source.

Trigonometric calculations in the form of ROM-encoded lookup tables will be used to find the location based upon the difference in the arrival times of the peak of the sound wave for any given period of time. Due to complications involved in calculating the vertical position of a sound source, we will be only able to find the position along the horizontal axis.

Abstract

Our basic setup is shown on Figure 1. Two microphones are spaced a distance x apart. A sound source is located on a plane a distance l perpendicular to the plane in which the microphones are situated. A projection screen is located equidistant between the microphones. d1 and d2 represent the distance from the sound source to the respective microphones. By means of the trigonometric calculations outlined on the attached diagram, we can triangulate the horizontal position of the sound source.

Process Flow

Figure 2 contains a block diagram outlining at an abstract level the logical flow of our Passive Sonar.

Stage 1:

Analog sound signals enter the left and right microphone inputs on the audio codec, labeled A on the block diagram. Serial digital audio samples corresponding to the left and right analog inputs feed out of two digital outputs on the codec and into separate 20-bit shift registers, labeled B and C on the block diagram. Configuration pins on the codec are connected to the microblaze processor to allow for easy configuration through a software interface.

Stage 2:

Data from the shift registers are read into the peak detectors, labeled F and G, once an entire 20 bit word has been loaded into the shift registers. The peak detectors operate by first comparing the sample with noise constant stored in the register labeled I on the diagram. Register I is configurable through the microblaze. If the sample is above the noise constant then the peak detector is activated for a constant period of time (to be determined by the length of a normal transient). All subsequent samples taken during this period of activation are compared against the previous greatest value and the time of occurrence based upon counter H's count are stored in internal registers. At the end of the transient sample period the *time* of the greatest value is output to the subtractor (D in diagram) to await the signal from the inactivated peak detector. While the subtractor is 'waiting', that peak detector is disabled, pending peak detection of the second signal.

Stage 3:

The inactivated peak detector (F or G) is now awaiting the signal to break the noise threshold. The second peak detector functions exactly like the first one, storing and tagging the maximum sample value. When the second peak detector reaches the end of the transient detection period the value is sent to the subtractor.

Note: the peak detectors send an extra bit (notated +1 on our diagram) along with the time to tell the subtractor whether or not it is a new word. The subtractor will only output the difference when two new words have been received.

Stage 4:

The subtractor (D) receives the first sample and holds it until it receives the second sample. The extra bit coming from the peak detector governs whether the subtractor is waiting or subtracting (to send a new difference).

Stage 5:

Our time difference is then sent to our lookup table (J). The lookup table will be filled with an array based on the trigonometry we did in Figure 1. The time difference will represent the index of the LUT which will output information to the microblaze (K) representing the horizontal position of the sound source.

Stage 6:

The microblaze has final control over the VGA framebuffer. It arbitrates between the output of diagnostic data obtained from the registers and the visualization of the sound source. In addition, the registers, including those in the codec, will be configurable through a software interface controlled by the microblaze.

Stage 7:

Video output from the framebuffer appears on a display.

Project design issues

1) We are placing the microphones a set distance apart from each other and placing that value into a register vs. making the distance variable. This is because leaving the distance as a variable would involve leaving all the trigonometry in terms of that variable. We would then need to build a multiplier in VHDL for the LUTs, which would unnecessarily complicate the project.

2) We briefly considered designating one mic as the dominant mic, meaning it would always receive the signal before the other mic. This would simplify the code and eliminate the need for some left/right variables, loops, if statements, et cetera. But a dominant mic would limit the range of direction that the sound could come from; it would have to come from closer to the dominant mic than the other mic.

3) Initially we had the system test for sound by comparing the signal against a noise floor, or threshold. Then whichever comparator that received the signal second would test it against the same floor. There was problem of finding the range to apply to the signal received by the second mic, as the same noise heard by the second mic will not be exactly the same as the noise heard by the first mic.

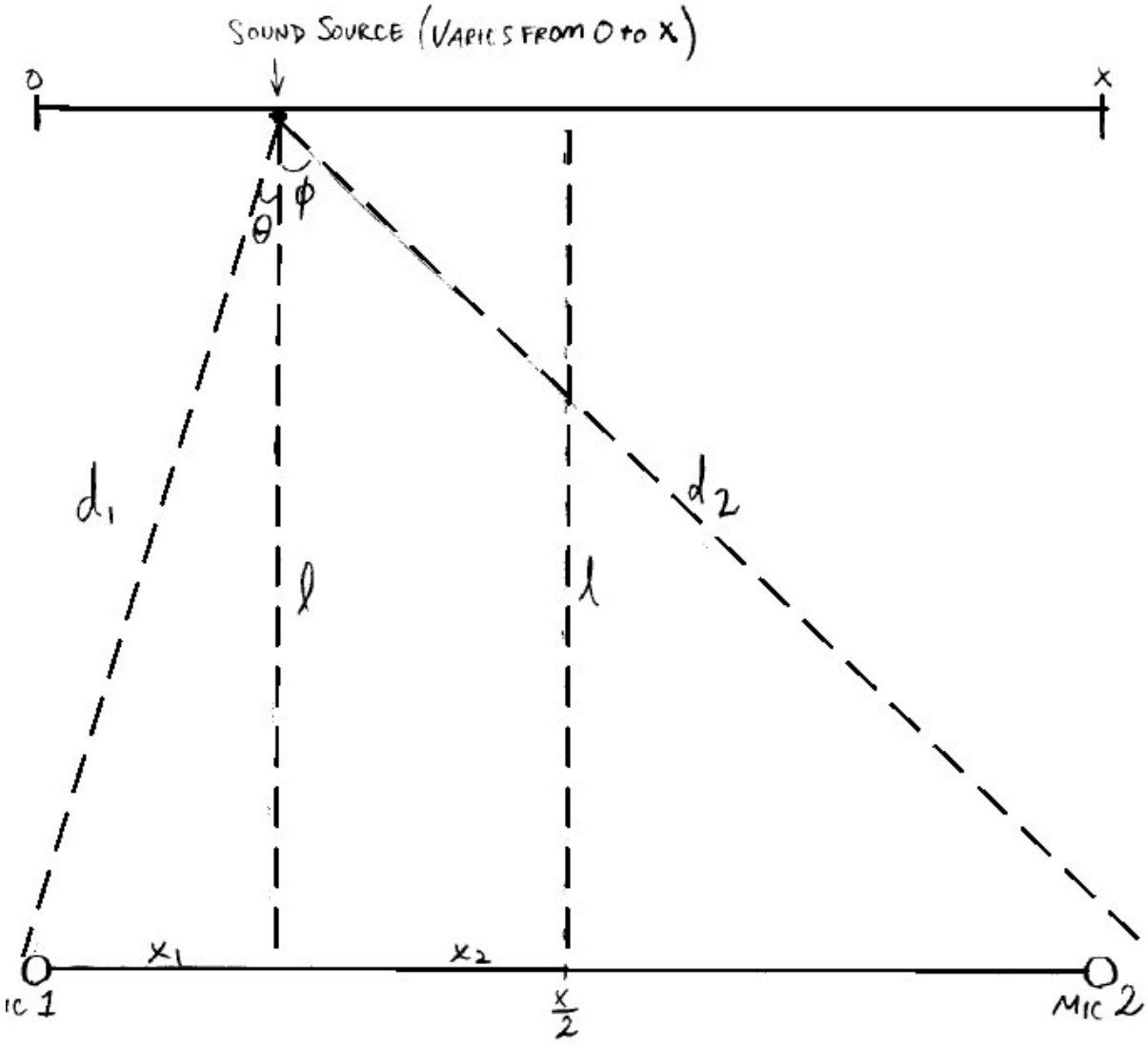
4) Now we are continuously sampling the signal once it passes a noise threshold for the largest amplitude within that sample. This peak detection will be much better able to accurately detect the delay between the left and right mics.

5) Much like the human hearing system, if the system receives a signal of constant amplitude, it will be confused because the system detects the sound by peaks in the sin wave. It will detect the initial attack and then the mics will pick up the peaks in the signal, but it will not be able to distinguish which period the peak is from. So the left and right mics will read the same signal, but at different periods, instead of the same signal's peak at one period. The system will actually be reading the phase difference of the shifted sin wave (the wave received by the second mic) as the delay. This phase difference will most likely be small enough that the vertical bar shined by the projector will stay near the center.

6) The system will work best with transience/pulses, i.e. cymbals or clapping. To detect a person speaking as he walks back and forth might require the person to enunciate in an unnatural manner. A discrete sound source will be simpler to work with than a continuous source.

7) Diagram is missing multiple resets, it will basically run through one iteration.

Figure 1



$$d_1 = \frac{l}{\cos\theta} \qquad d_2 = \frac{l}{\cos\phi} \qquad \theta + \phi = \frac{\pi}{4}$$

$$\frac{l}{\cos\theta} = v t_1 = 340 \frac{m}{s} \cdot t_1 \qquad t_1 = \frac{l}{\cos\theta \cdot 340}$$

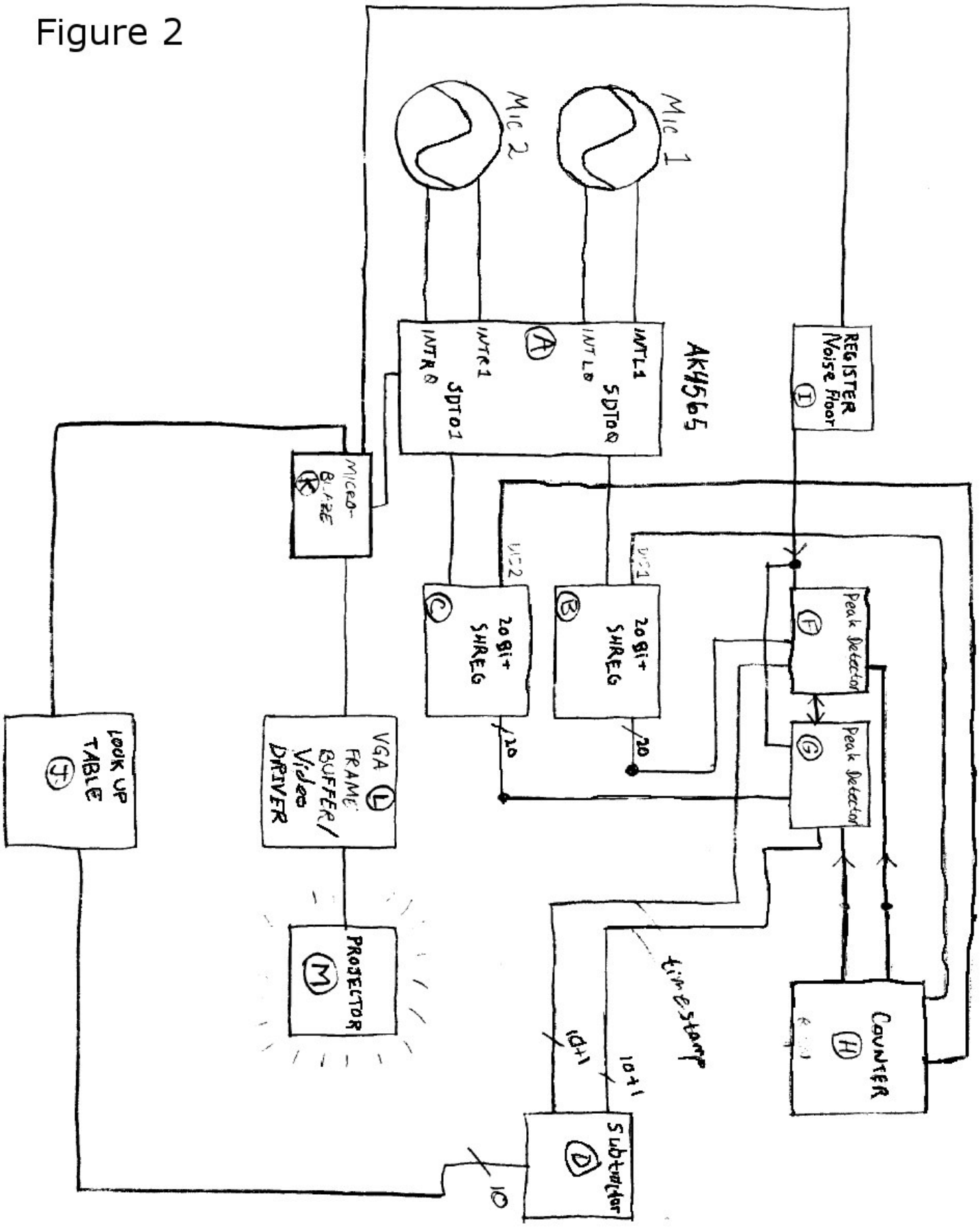
$$\frac{l}{\cos\phi} = v t_2 = 340 \frac{m}{s} \cdot t_2 \qquad t_2 = \frac{l}{\cos\phi \cdot 340}$$

When $t_1 = t_2$, $\phi = \theta$, $d_1 = d_2$

$x_1 = x_2 = \frac{x}{2} = \text{Delay}(0)$

$\text{Delay} = t_1 - t_2$ $\quad \text{Delay} < 0,\ x_1 < x_2$; $\text{Delay} > 0,\ x_2 > x_1$

Figure 2


MIC 1
MIC 2
AK4565
INTL1
SDTOQ
INTR1
SDTO1
INTR0
A
REGISTER
Noise Floor
I
MICRO-
BUFFE
K
20 Bit
SHREG
B
20 Bit
SHREG
C
Peak Detector
F
Peak Detector
G
20
20
VGA
FRAME
BUFFER/
Video
DRIVER
L
LOOK UP
TABLE
J
PROJECTOR
M
timestamp
10+1
10+1
Subtractor
D
10
COUNTER
H