Sonomicrometric Arrival Time Detection Using Reconfigurable FPGA

Pong P. Chu and Tamman Shaar
Electrical and Computer Engineering Department
Cleveland State University
Cleveland, Ohio 44115

Abstract

Sonomicrometry is the measurement of distances using sound. Its accuracy depends heavily on the receiver's ability to detect the first arrival of the transmitted ultrasound signal. It has been shown that a coherent signal matching detector can significantly improve the measurement. Our study modifies the original algorithm and implements it utilizing FPGA (Field Programmable Gate Array) chips. We treat the slow varying "template function" of the transmission media as constant coefficients that can be reloaded. This approach takes advantage of the reconfigurability of FPGA and avoids the need of complex multiplication operations. It significantly reduces the circuit complexity and provides a viable alternative for traditional DSP (Digital Signal Processing) processor based design.

1. Overview on Sonomicrometry

Sonomicrometry is the measurement of distances using sound [3]. The conceptual diagram of a single-distance measurement is shown in Figure 1. The sonomicrometer includes a control circuit, a transmitter and a receiver. To do a measurement, the transmitter and receiver are attached to the object. The control circuit starts the timer and triggers the transmitter to generate a burst of ultrasound. The timer continues to count until the receiver detects the arrival of this ultrasonic pulse. Once the elapsed time is known, the physical distance between the transmitter and receiver can be determined using a simple equation:

$$distance = transmission\ speed \times elasped\ time$$

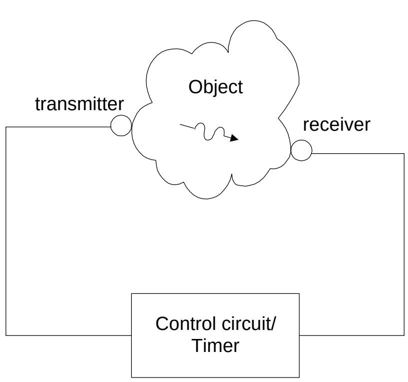

Figure 1. Conceptual Diagram of Sonomicrometer

Current sonomicrometer utilizes transducers made from piezo-electric ceramic material for both transmitter and receiver, operating at frequency around 1 MHz. The measured elapsed time can reach a resolution of 15 nano-seconds. In biological materials, the typical transmission speed is 1540 meters per second, resulting in a distance resolution of 0.024 mm. Sonomicrometry can be used to measure strain, length, area, thickness, volume, and geometry in natural and man-made materials and has been used in a wide variety of applications in biomedical research.

2. Coherent Signal Matching Detector

The resolution of a sonomicrometer depends on its ability to accurately detect the first arrival of the received ultrasound signal. It is a difficult task since the signal is distorted during transmission and is frequently corrupted with external noises. Current sonomicrometers normally use threshold detection and are susceptible to noisy environment. One new proposed method is to use a coherent signal matching detector and simulation showed that it is very effective [2].

The detection algorithm models the object as a noisy channel and utilizes a function that determines the likelihood that the leading edge of the received signal occurs within a particular window. The likelihood function, $l(t)$, is defined as follows:

$$l(t) = \left(\int_0^{t_0} x(t+t)h(t)dt\right)^2 + \left(\int_0^{t_0} x(t+t)h^*(t)dt\right)^2$$

The entities in this equation are defined as follows:

- $h(\cdot)$ is the function describing the characteristics of the transmission channel (i.e., the object to be measured). It is also known as the template function.
- $h^*(\cdot)$ is the Hilbert transformation of $h(\cdot)$.
- $x(\cdot)$ is the signal received by receiver.

The function $l(t)$ measures the likelihood of the leading edge for the window period $(t,\ t+t_0)$. The value of $l(t)$ should be monitored continuously. When it reaches the maximum, the corresponding *t* is the most likely arrival time. The detailed theoretical discussion of the likelihood function can be found in [2,4].

In order to implement the algorithm in digital hardware, the likelihood function has to be in discrete form. The corresponding equation is:

$$l(i) = \left[\sum_{j=i}^{i+k} x(j) \bullet h(j-i+1)\right]^2 + \left[\sum_{j=i}^{i+k} x(j) \bullet h^*(j-i+1)\right]^2$$

The required computation for this function is quite intensive. *k* multiplication operations and *k-1* addition operations are needed for the expression inside each bracket, and *2* extra multiplication operations and *1* extra addition is needed for final square computation and addition. Thus, for every sampled signal, the system has to perform *2k+2* multiplication operations and *2k-1* addition operations. Traditionally, this kind computation intensive algorithm has to utilize a fast DSP processor based system.

3. Distributive Arithmetic Utilizing FPGA

FPGA basically consists of arrays of generic logic cells connected by pre-defined routing paths. Both the cells and interconnections can be programmed to perform a specific function. Since FPGA chips don’t contain any special arithmetic circuit, it cannot effectively support brute-force implementation of multiplier and cannot compete with general DSP processors, which normally contain a customized multiplication-accumulation unit. However, FPGA does perform very well for certain look-up table based computations, known as *distributive arithmetic* [1,5]. It is particularly good to compute the sum of products with one constant multiplicand. The computation is normally expressed as:

$$y(i) = \sum_{j=1}^{k} A_j \bullet x(i+j)$$

In most DSP applications, $x(\cdot)$ is the time-varying signal sampled in real time. A_js, on the other hand, are frequently predetermined coefficients, which do not change with time. Since A_js are treated as constants, there is no need to implement full-blown multiplier. Instead, simple table look-up logic can be used and the entire sum-of-product computation is reduced to addition and shift operations. FPGA is especially good for this approach since it can easily customized and configured for the different coefficients.

4. FPGA Implementation of Coherent Signal Matching Algorithm

If we examine the likelihood function carefully, it shows that it is possible to utilize distributive arithmetic to calculate the value of the function. This comes from the observation that while the received signal, $x(\cdot)$, is sampled in real-time at high rate, the template function, $h(\cdot)$, and its Hilbert transformation, $h^*(\cdot)$, hardly change. The values of $h(\cdot)$ and $h^*(\cdot)$ depend on the type of objects and operation environment, and remain the same in a measurement. They can be treated as constants and only need to be updated when the operating environment changes.

Even we treat $h(\cdot)$ and $h^*(\cdot)$ as constants, two multiplications are still required to perform the square operations. The purpose of the square operation is basically to detect the absolute maximal swing and negate the effect of the negative sign. Since our goal is to find *when* the likelihood function reaches the maximum, the exact value of the function does not matter. Therefore, we utilize the absolute operation to replace the square operation. The desired likelihood computation now can be rewritten as follows ($h(\cdot)$ is replaced by A to emphasize its constant nature):

$$l(i) = \left|\sum_{j=1}^{k} A_j \bullet x(i+j)\right| + \left|\sum_{j=1}^{k} A_j^* \bullet x(i+j)\right|$$

In actual implementation, $x(\cdot)$ is stored in a buffer and shifted for very new sample, and a separate counter will keep track of index i. Thus, the equation only needs j for the buffer location and can be written as:

$$l = \left|\sum_{j=1}^{k} A_j \bullet x(j)\right| + \left|\sum_{j=1}^{k} A_j^* \bullet x(j)\right|$$

The block diagram to implement this algorithm is shown in Figure 2. The system performs two sum-of-product operations, obtains the absolute values and adds them together. The final value is passed to a positive peak detection circuit, which compares the current values with the stored maximum. If the new value is larger, the value and the corresponding time tick will be stored in the arrival time register. One major delay of the operation is the final absolute value operation and addition. Since distributive arithmetic is normally performed in bit-serial manner and the sign bit is obtained in the end, the absolute circuit and adder have to wait.

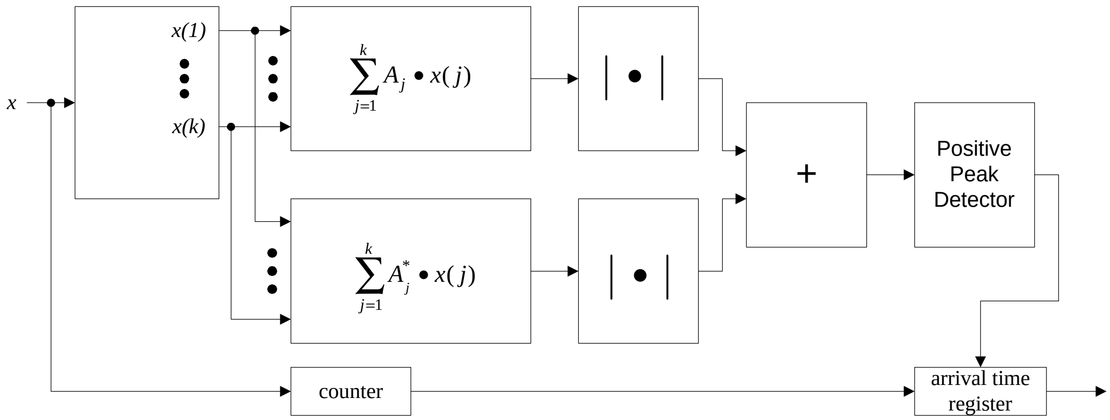

Figure 2. Block Diagram of Coherent Matching Detector

An alternative approach is shown in Figure 3. Instead of computing $\sum_{j=1}^{k} A_j \bullet x(j)$ and $\sum_{j=1}^{k} A_j^* \bullet x(j)$, the new scheme calculates $\sum_{j=1}^{k} (A_j + A_j^*) \bullet x(j)$ and $\sum_{j=1}^{k} (A_j - A_j^*) \bullet x(j)$. . The expression $\sum_{j=1}^{k} (A_j + A_j^*) \bullet x(j)$ assumes that $\sum_{j=1}^{k} A_j \bullet x(j)$ and $\sum_{j=1}^{k} A_j^* \bullet x(j)$ have the same sign and perform addition in advance, and the expression $\sum_{j=1}^{k} (A_j - A_j^*) \bullet x(j)$ assumes that they have different sign. Note that the values $A_j + A_j^*$ and $A_j - A_j^*$ are pre-calculated new constants and require no extra hardware.

The final results of $\sum_{j=1}^{k} (A_j + A_j^*) \bullet x(j)$ and $\sum_{j=1}^{k} (A_j - A_j^*) \bullet x(j)$ can be either positive or negative. Instead of calculating their absolute values, the results are passed a positive peak detector and a negative peak detector. In the end of the measurement, the magnitude comparator chooses the one with a larger value and selects the corresponding time from the arrival time registers. This approaches reduces the need for intermediate absolute-addition circuit and significantly improves this operation speed.

Since a sonomicrometer is expected to work with different types of materials and different operation environments, the values of A_j and A_j^* have to be dynamically adjusted. This requirement can be easily accommodated by SRAM based FPGA. Each set of A_j and A_j^* can be treated as a unique circuit and synthesized to configuration files in advance. During the actual measurement, sonomicrometer will determine the type of the object material and loaded the proper configuration onto the FPGA chip accordingly.

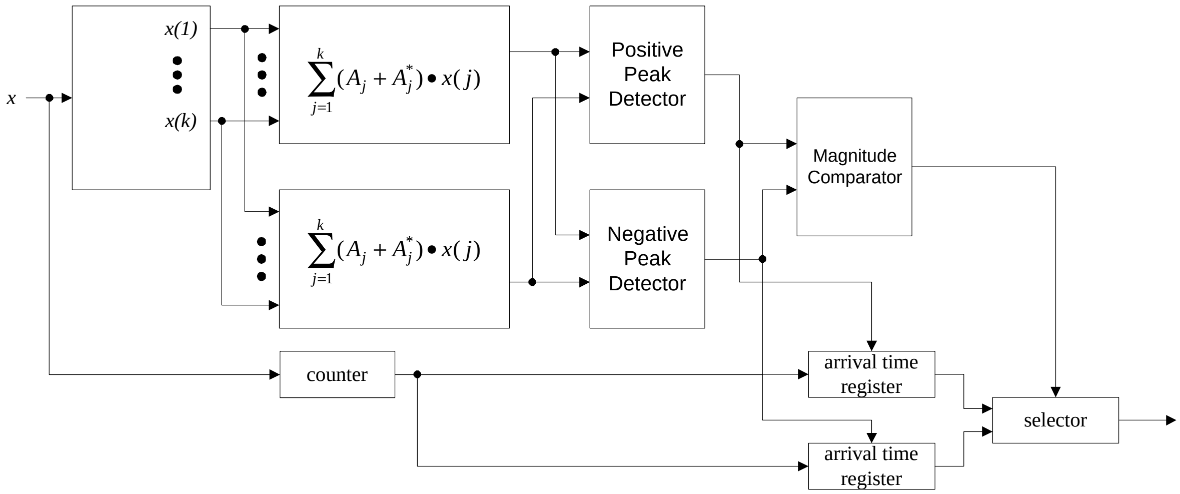

Figure 3. Block Diagram of Coherent Matching Detector

5. Summary

Coherent matching signal detector can improve the accuracy of arrival time detection of sonomicrometers. We treat the template function of the object as constant coefficients, modify the likelihood function and apply the distributed arithmetic to implement this scheme. This approach takes advantage of the reconfigurability of FPGA and avoids the expensive multiplication circuit, and can be a vital alternative for DSP based design.

Acknowledgement

The authors would like to thank Dr. Fuqin Xiong of Cleveland State University and Dr. Ivan Vesely of Cleveland Clinic Foundation for introducing us the topic.

Bibliography

[1]. Altera Corporation, “Implementing FIR Filters in FLEX Devices,” Application Note 73, 1998.
[2]. James W. Davis, “Improved Arrival Time Detection for Cardiac Pulse Transit Sonomicrometry,” *IEEE Computers in Cardiology*, pp. 145-148, 1996.
[3]. Sonometrics Corporation, “General Principles of Sonomicrometry,” http://www.sonomertics.com/sono_101.htm.
[4]. A. D. Whalen, *Detection of Signals in Noise*, Academic Press, 1971.
[5]. Xilinx Corporation, “The Role of Distributed Arithmetic in FPGA-based Signal Processing”, Application Note, 1996.