Acoustical Radar

Leonard Sokoloff
DeVRY Technical Institute

Virtual Instrumentation is making a significant impact in today’s industry, education and research. DeVRY Technical Institute selected LabVIEW as an excellent representative of this technology and is implementing LabVIEW into its curriculum at all DeVRY campuses in the United States and Canada.

LabVIEW® (Laboratory Virtual Instrument Engineering Workbench) a product of National Instruments@, is a software system that incorporates data acquisition, analysis and presentation, and instrument control. LabVIEW which runs on the PC under Windows, on Sun SPARCstations and on Apple Macintosh computers, uses the graphical programming language, instead of the traditional high level languages such as C language or Basic.

All LabVIEW programs called virtual instruments (VIs), consist of a Front Panel and a Block Diagram. The Front Panel contains various controls and indicators while the Block Diagram includes functions. The functions (icons) are wired together inside the Block Diagram (wiring indicates data flow) to create a VI. The execution of a VI is data dependent which means that a node inside the Block Diagram will execute only if data is available at all its input terminals. This differs once again from the execution of a traditional program which executes instructions in the order they are written.

As stated earlier, LabVIEW incorporates data acquisition, analysis and presentation into one system. For acquiring data and controlling instruments, LabVIEW supports IEEE-488 (GPIB) and RS-232/422 protocols as well as other D/A, A/D and digital I/O interface boards. The Analysis library offers the user a comprehensive set of tools for signal processing, filtering, statistical analysis, linear algebra operations and many others.

This article describes an application of LabVIEW to the acquisition, processing and the display of data. In order to perform data acquisition, LabVIEW software (latest version is 4.01) and data acquisition driver software such as NI-DAQ must be installed. The DAQ board (data acquisition board) must also be installed along with the extension board that plugs into the DAQ board. The extension board provides the user with access to various pins on the DAQ board such the I./O channels.

This article is a result of a research project on implementing LabVIEW in our physics courses. Projectile flight simulation is another LabVIEW program that will also be implemented into the physics courses. Other areas of LabVIEW implementation include industrial controls and communication courses.

LabVIEW is typical of new skills that students will need in today’s highly competitive job market.

System Overview

This implementation of LabVIEW illustrates the fundamental radar principles at acoustical frequencies. An ultrasonic transducer is the heart of the system. It acts first as a transmitter and later as a receiver. In the transmit mode, the ultrasonic transducer transmits 16 sonic pulses at 49.4 kHz. These pulses propagate through space with the speed of sound toward the nearest object and are reflected by it. The reflected pulses, or echoes, travel back toward the ultrasonic transducer which now switches to the receive mode. The time difference between the transmitted pulse and the echo can be converted into distance to the object by appropriate scaling.

In this implementation, LabVIEW is used to determine the time difference between the transmitted pulse and the received echo. LabVIEW applies an appropriate scaling factor to the time difference in order to convert it to distance, and then displays this data on the vertical slide (one of LabVIEW's digital indicators) and on the waveform chart. It also displays the transmitted pulses and the echoes on the waveform graph.

Storing the acquired data into a spread sheet can also be done, but in this application the data processing VI was set up with a While Loop, allowing the data to be acquired and processed indefinitely until the user decides to terminate the VI execution by clicking on the Stop switch.

Because the time between the emitted pulse and the returned echo was relatively short, the software overhead associated with acquiring and processing the data was too large to conduct repetitive measurements in real time. A quasi real time data acquisition approach was adopted instead. With this method, one block of data is acquired and processed, then the next block and so on. One can see the flicker effect on waveform graph as the data is displayed on the block by block basis. This approach to data acquisition is adequate for stationary or slow moving objects.

System Hardware

Driver Board

The Driver Board is part of the commercially available 6500 series sonar ranging module which is typically used in camera focusing application. After the power is applied to the Driver Board, at least 5 milliseconds must elapse before applying the input to the INIT pin. This time is necessary to initialize and stabilize various circuits on the board. Fig. 1 shows the Driver Board and its output being interfaced with the LabVIEW environment.

Once the board is stabilized, TTL pulses at 20 Hz are applied to the INIT input. During the LOW to HIGH transition of each pulse, the board generates 16 pulses at 49.4 kHZ and 400 volt amplitude and applies these pulses to the ultrasonic transducer which acts as electrical to acoustical transducer. The resulting sound waves travel at approximately 1100 feet/second toward a nearby object which reflects some of the incident energy. The reflected energy, or echo, travels back to the ultrasonic transducer which switches to the receive mode and acts this time as an acoustical to electrical transducer converting the acoustical energy of the echo to a voltage pulse.

As shown in the timing diagram of Fig. 2, the time difference between the leading edges of the INIT and the echo pulses must be measured and converted to distance. As it takes the sound wave 0.9 millisecond to travel a distance of 1 foot, an object 5 feet away will return an echo in 9 milliseconds.

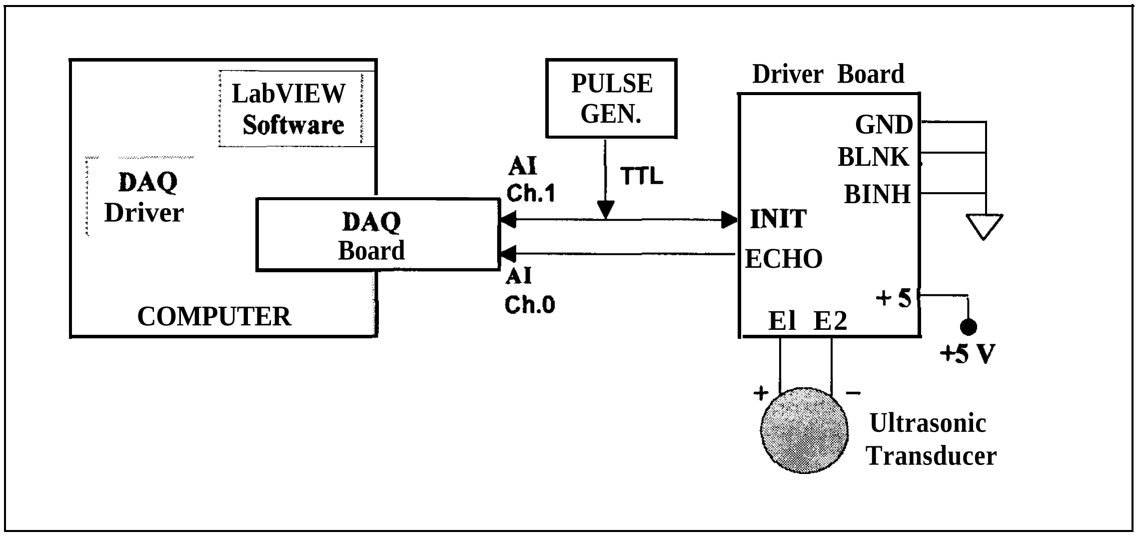

Fig. 1 The Driver Board and its output being interfaced to the LabVIEW environment

In order to suppress a false echo which can occur due to vibration of the ultrasonic transducer immediately following the transmission of the 16 pulses, the driver board generates a 2.38 millisecond blanking pulse which inhibits echo reception. This means that the closest object must be at least 1.3 feet away from the transducer. The user, at his discretion, may override the internal blanking by raising the BINH (internal blanking inhibit) pin HIGH. By so doing, the minimum distance to the object can be reduced to 0.5 feet. The maximum distance is 35 feet.

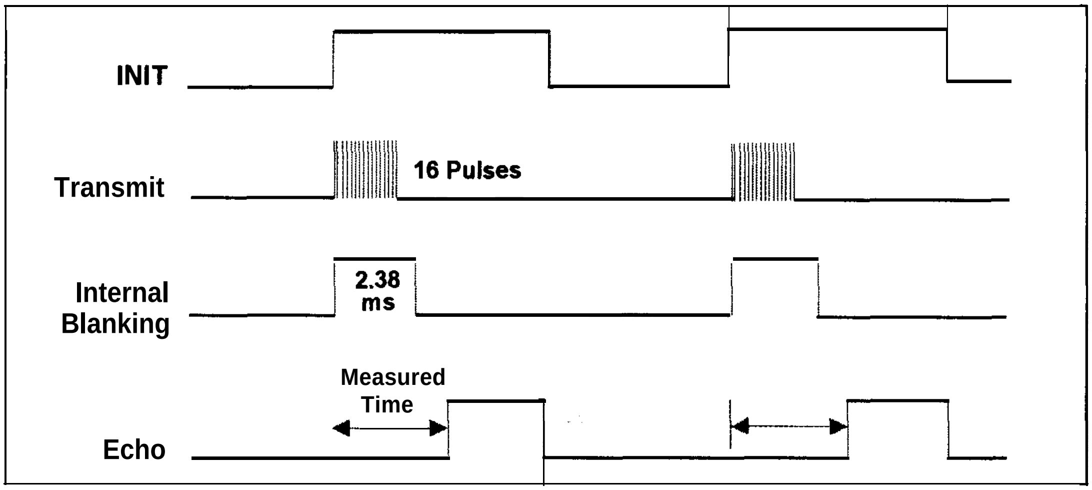

Fig. 2 Timing Diagram

System Software

LabVIEW offers three categories of data acquisition VIs: *Easy VIs Intermediate VIs and Advanced VIs* Easy VIs perform most common VI operations. They are simple to use because the configuration complexity is designed into the VI icon. These icons usually include some of the Intermediate VIs which in turn are made up of the Advanced VIs. Advanced VIs are the fundamental building blocks for all data acquisition VIs. They have the most programming power and flexibility.

Analog input data acquisition options include: *immediate single point input and waveform input.* In using the immediate single point input option, data is acquired one point at a time. Software time delay to time the acquisition of the data points, which is typically used with this option, makes this process somewhat slow.

Waveform input data acquisition is *buffered and hardware timed* The timing is provided by the hardware clock that is activated to guide the acquired data points quickly and accurately. The acquired data is stored temporarily in the memory buffer until it is retrieved by the data acquiring VI.

The LabVIEW program includes two main components: **Delta_t.vi,** a subVI, and the **Radar.vi** which is the main program. Delta_t.vi has the responsibility of processing a block of data and determining the unscaled round trip time to the object.

The main program Radar.vi acquires a block of data, instructs Delta_t.vi subVI to determine the unscaled time, converts the unscaled time to distance and displays the distance on the vertical pointer slide and on a waveform chart. It also displays the 2000 points of the TTL and the Echo array on the waveform graph. As the main program is placed inside the While Loop, the program repeats indefinitely until the user pushes the Stop button.

The use of the While Loop provides the user with a friendly, interactive environment. Any new data is updated on the Front Panel indicators every half second.

Radar.vi Front Panel

The Front Panel of the Radar.vi as shown in Fig. 3 includes the waveform graph which displays the TTL and the Echo block of data. The waveform chart (Distance Chart) and the vertical pointer slide (Distance (ft)) display the calculated distance for each data block. The digital control Calibrate is used to convert the uncalibrated time to distance in feet.

Radar.vi Block Diagram

AI Acquire Waveforms.vi is an Easy VI that is used to acquire two channels of data. This VI provides buffered and hardware timed data acquisition. The user must specify the number of samples to be acquired for each channel and the scan rate. The output is a 2-D array where each row represents one complete scan.

In this application the Acquire Waveforms.vi as shown in Fig. 3 is instructed to acquire 2000 samples from each channel at the scan rate of 10,000 scans/sec. Channel 0 receives the echo data

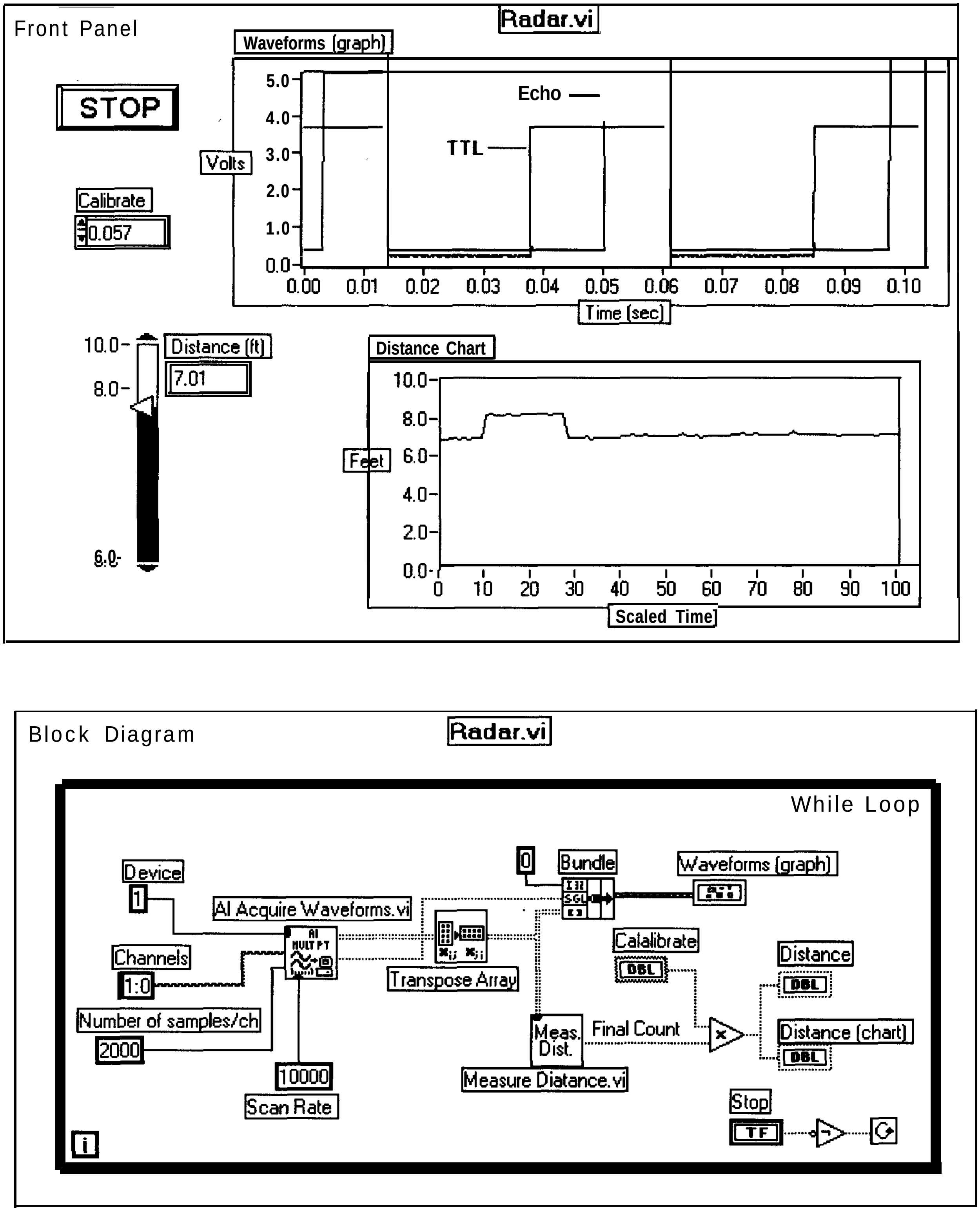

Fig. 3 The Front Panel and the Block Diagram of Radar.vi.

from the driver board and channel 1 receives the TTL pulses. Because the channel scanning is done in reverse order, column 0 of the output array contains 2000 samples of the TTL data and column. 1 contains 2000 samples of the echo data. During data acquisition, LabVIEW places the data into the buffer and when completed retrieves the array from the buffer and outputs it.

The Transpose Array function takes this array and switches rows and columns. Row 0 in the transposed array now contains the TTL data, and the echo data is in row 1,

The transposed 2-D array is then applied to the Waveform graph for display and also to the Delta_ t.vi subVI for data processing.

The Final Count output from the Delta t.vi is an unscaled time corresponding to the time difference between the leading edges of the TTL pulse and the nearest echo pulse. This unscaled time is then multiplied by the value of the Calibration digital control and applied for display on the vertical pointer slide (Distance) and on the waveform chart (Distance (chart)). This completes one execution of the While Loop. The above process repeats until the user pushes the Stop button.

Delta_t.vi Block Diagram

As shown in Fig.4, Index Array 1 extracts row 0, the 2000 samples of the TTL data, from the incoming 2-D array. The resulting 1-D array is next processed to determine the first occurrence of the LOW to HIGH transition in the TTL data. This is done by checking the first element of the TTL array in the For Loop which is executed only once. If it is LOW, the TRUE Boolean Case will be executed and if it is HIGH, the FALSE Boolean Case will be executed.

Let’s take the FALSE Boolean Case first. Once again, the purpose of this case is to determine the occurrence of the first LOW to HIGH transition in the TTL array. While Loop 1 starts by searching the array for the first occurrence of a LOW and when its finds it, the index il is used to extract the subset of the 1-D TTL array. The resulting subset which begins at index il is next applied to While Loop 2 which continues the search for the first occurrence of the HIGH in the subset array. When the HIGH is found, index i2 marks that point in the subset array.

As was done with the TTL array, Index Array 2 extracts the Echo array from the incoming 2-D array and applies it to Array Subset 2 function. Because While Loop 2 began its execution at i2=0 the index i2 is added to il, marking the point in the TTL array where the first LOW to HIGH transition occurred. The Array Subset 2 forms a subset of the echo array beginning at the index i1+i2 and applies the resulting subset to While Loop 3 which searches for the first occurrence of a HIGH. When While Loop 3 finds the first HIGH, it outputs index i3 which represents the final count. Because While Loop 3 begins its count at i3=0, the final count represents the uncalibrated differential time interval between the two consecutive leading edges of the TTL and Echo pulses. This final count value is available at the output terminal of the Delta_t.vi subVI.

The situation in the True Boolean Case shown in Fig. 4 is simpler because the first element of the TTL input is already LOW. This eliminates one While Loop from the True Case. The only thing that has to be done here is to find the first occurrence of the HIGH in the TTL array. When

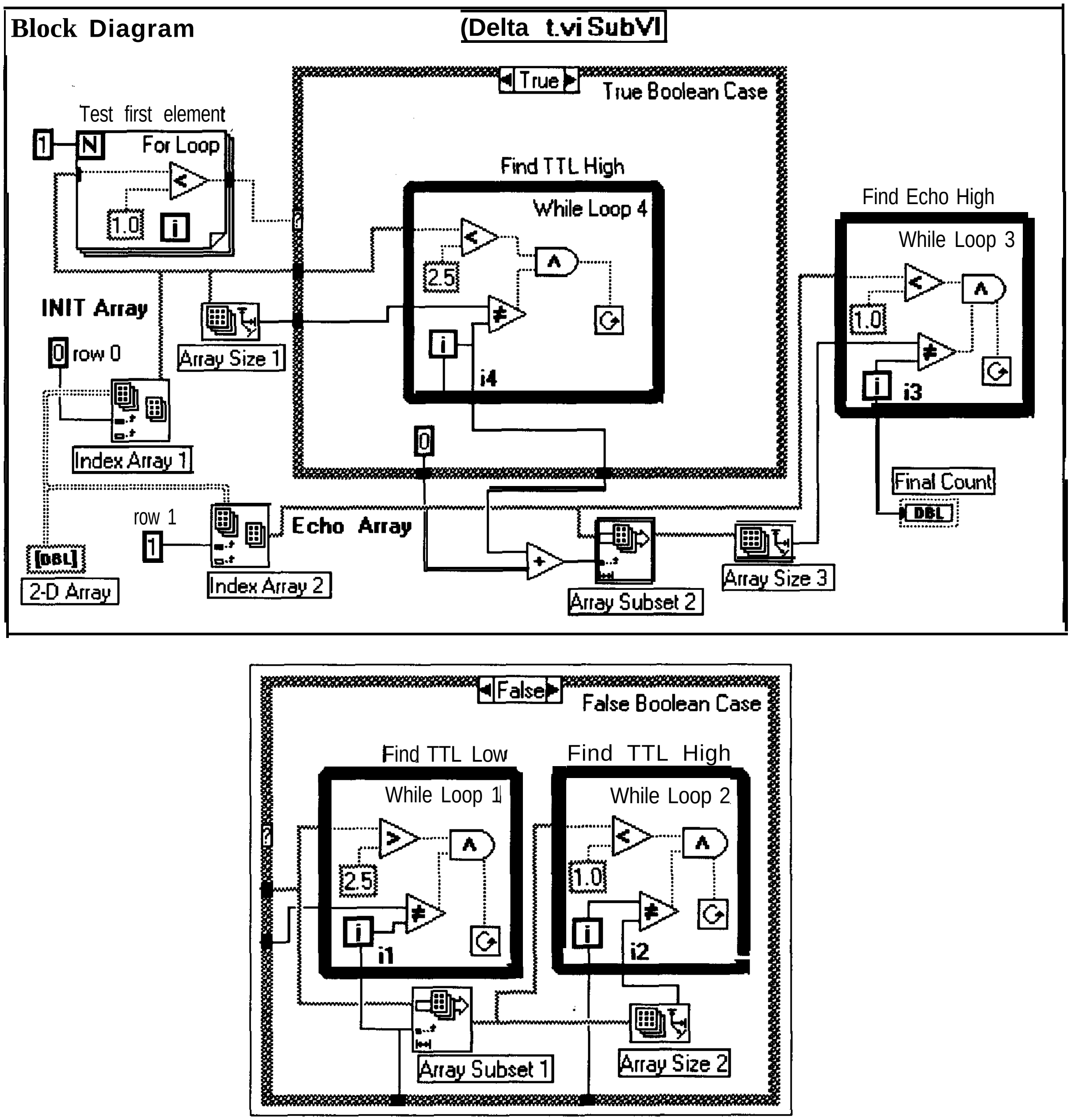

Fig. 4 The Block Diagram of the Delta_t.vi subVI.

While Loop 4 finds the first HIGH, it outputs the index i4 which marks the first LOW to HIGH transition in the TTL array. The rest of the procedure is similar to that described above. Array Subset 2 forms a subset of the echo array beginning at index i4 and applies the subset to While Loop 3 which finds the first HIGH in the echo array:

The remaining nodes in the block diagram which do not participate in determining the final count include the Array Size, Not Equal To and the AND gate. They provide the system time-out feature which forces the While Loops 1 through 4 to terminate execution if no data is present.

Conclusion

The AI Acquire Waveforms.vi scans two channels and acquires 2000 samples from each channel at 10,000 scans/sec. As there are 2 samples in each scan (one from each channel), the resulting sampling rate is 20,000 samples/sec. Hence the time required to complete the acquisition of one block (4,000 samples) of data is 0.2 seconds.

As noted before, this type of data acquisition is hardware timed, activating the scan clock on the DAQ board which provides precise timing in acquiring samples. The acquired 2-D array (2000x2 in this application) is stored in the buffer during the acquisition and then retrieved by the AI Acquire Waveforms.vi. The data has to be processed and displayed which takes additional time.

The Project utility, one of the new features of LabVIEW v. 4, was used to assess system performance. The results indicate that on the average it takes 0.2 seconds to acquire a block of data and 0.3 seconds to process the data.

This application was executed using a 75 MHz pentium PC under Windows 3.1. The observed update rate was approximately one distance data point every 0.5 seconds.

The TTL square wave in this application was set to 20 Hz. This means that approximately two cycles of the square wave will be acquired on each iteration of the While Loop. The waveform display in the Front Panel of Fig. 3 is a snapshot of the actual run.

The quasi real time data acquisition approach works well for stationary or slow moving objects and is an attractive alternative in applications where the software is unable to keep up with the incoming data.

Bibliography

LabVIEW®: Basic Concepts by L. Sokoloff, Prentice Ball, 1997
LabVIEW® Graphical Programming by G. W. Johnson, McGraw-Bill 1994
LabVIEW® Data Acquisition VI Reference Manual, National Instruments@
LabVIEW® for Windows User Manual, National Instruments@
LabVIEW® Function Reference Manual, National Instruments@
LabVIEW® for Windows Tutorial, National Instruments@
LabVIEW® Getting Started with LabVIEW for Windows, National Instruments@

Biography

Leonard Sokoloff was born in Russia and immigrated to the United States in 1950 and was awarded the BSEE degree from Stevens Institute of Technology (1959), the MS Applied Science degree from Adelphi University (1964) and the PhDEE (candidate) from Stevens Institute of Technology. Worked in industry as semiconductor application and circuit design engineer (1959 - 1970). For the past 27 years with DeVRY Technical Institute, currently as senior professor, teaching associate level and bachelor level courses in advanced mathematics and electrical engineering.