Engineering 90 Report
High Speed Time Domain Reflectometer

Aron P. Dobos
Advisor, Lynne Molter, Sc.D.

May 4, 2006

Abstract

Time domain reflectometry has proven to be an indispensible technology in cable characterization, geological survey, and many other applications. Low cost, high resolution, and portability remain the critical design constraints that the designer must cope with when creating a measurement instrument capable of performing well in a wide variety of target domains. The design and construction of a low cost yet high bandwidth Time Domain Reflectometer (TDR) instrument is documented. The instrument is intended for 50 Ω SMA cable characterization applications in which discontinuities spaced as low as 4 cm must be discerned. A modular data acquisition platform allows flexibility in the choice of a host control interface, allowing the TDR to be integrated into various specialized target applications. Two functioning prototypes were built and tested for under $600.

Contents

List of Figures

List of Tables

Acknowledgements The efforts of numerous people have been instrumental in making this senior design project feasible and successful. Professor Lynne Molter's advice and encouragment throughout the course of the project proved to be significant help and motivation towards its successful completion. Professor Erik Cheever's lab space and assistance with purchasing and setting up a special surface mount electronics lab was essential to the project, as well as technician Ed Jaoudi's help with ordering components. Dr. Agoston Agoston at Hyperlabs, Inc. provided several specialized components for the project, as well as generous technical support and consulting. Laszlo Dobos provided significant technical help through numerous design reviews and debugging assistance. Lastly, many thanks are extended to the Swarthmore College Engineering Department, and also to all the friends and family who provided much appreciated support and encouragement over the duration of the project.

1 Introduction

1.1 Technology Overview

Time domain reflectometry is a method of characterizing conductors and other test mediums by measuring the nature of energy propagation through the device under test. It was first developed in the 1950s by the power and communication industries for cable fault finding, but many other applications have since emerged. Some of these include geological surveying, groundwater table level and soil moisture measurement, localization of oil fields by modern drilling companies, and monitoring of the structural integrity of large concrete structures. The Time Domain Reflectometer (TDR) instrument documented herein focuses on precise cable characterization, although its adaptation to other areas remains straightforward. Using such a TDR, telecommunication companies can characterize long cables that are either buried underground or otherwise evade direct observation, and problems such as breaks and damage from water infiltration can be directly located for localized repairs.

Making a TDR measurement involves propagating a pulse of energy into a test medium and measuring any reflected energies as a function of time. Reflection may occur at the interface between two different materials, an impedance change in a wire due to disfigurement or other perturbation, or any change in the transmission characteristic along the path of the energy propagation in the test medium. The energy amplitude at the point of injection as a function of time thus gives a temporal characterization, but the test cable can be also characterized spatially by converting from a time to a distance scale. This is possible if the cable dielectric constant is known, since then the velocity of propagation (VOP) of the signal in the cable can be calculated, and hence conversion from time to distance is straighforward. In this way, the distance to a cable fault (e.g. short or open circuit) from the point of injection can be readily determined.

1.2 Related Technologies

A related method of cable characterization is Frequency Domain Reflectometry (FDR). In this method, the instrument creates a high frequency sinusoidal sweep that is propagated through the test medium. By measuring the interference pattern between the injected signal and the reflections arising from faults and other discontinuities, a ripple vs. frequency waveform is obtained. Conversion to a time domain signal is achieved with the fast Fourier transform (FFT), and thus conversion to a distance scale is possible. Unlike the TDR method, distance measurements using FDR are generally more precise since they do not rely on the accuracy and jitter characteristics of the TDR's timebase control circuits. The FDR technique may also provide finer detail in the frequency spectrum of the actual cable charcteristic, and problems like vapor intrusion and contact corrosion that might evade conventional TDR can be detected. The FDR method, however, requires an extremely high frequency sweep for shorter cables (<10 ft) to obtain usable resolution. For basic cable fault testing and characterization, the TDR technique was adopted to acheive potentially lower cost and greater applicability to both short and long distance cable testing.

1.3 TDR Instrument Goals and Direction

The TDR instrument outlined in this document is capable of resolving discontinuities spaced as close together as 5 cm. The high bandwidth discrete sampling technology was provided by Hyperlabs Inc., a company in Beaverton, Oregon, specializing in high performance sampler, pulser, and

TDR technologies. With assistance from Hyperlabs, the sampler circuit was modified for experimentation in this project. The primary objectives of the project were to develop a high accuracy timebase circuit with very low jitter, to achieve low power consumption by using a highly integrated microcontroller for the acquisition engine, and to provide a very flexible (although power hungry) 2-channel pulse generator circuit with a sub-100 ps rise time. The timebase, acquisition microcontroller, and step generator circuits, as well as the accompanying software controls may be integrated into future Hyperlabs TDR products, thereby confirming the practicality of the approach taken in this project.

A brief mathematical survey of transmission lines and TDR theory will be pursued. Thereafter, the overall instrument structure will be outlined, followed by detailed documentation of each component of the TDR. A discussion of testing, results, and potential future improvements will conclude the report.

2 Transmission Lines and TDR

2.1 Motivation

The mathematical foundations of TDR for cable testing can be arrived at from the development of a methodology and equations for looking at the behavior of transmission lines. A simple transmission line can be modeled as a parallel plate waveguide structure, and the electromagnetic field descriptions of the waveguide can be translated into the standard voltage, current, and impedance forms.

2.2 Failure of Lumped-Parameter Models

Unlike straightforward circuit elements, transmission lines cannot be modeled as lumped parameter systems. In lumped parameter models, it is assumed that the parameter (voltage or current) does not change across a circuit element. In other words, the length of the component is assumed to be much shorter than the wavelength of the applied signal passing through it. At very high frequencies

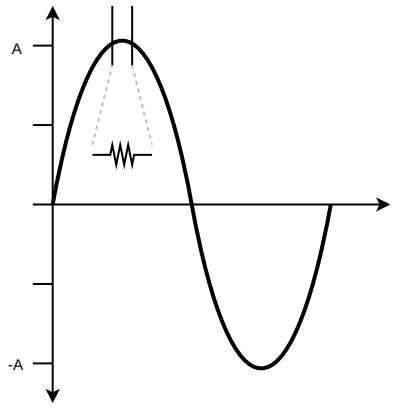


Figure 1: Lumped Parameter Models for Low Frequencies

or very long distances, however, such assumptions no longer hold. Therefore,

$$i \neq C\frac{dV}{dt} \text{ and } v \neq L\frac{dI}{dt}.$$

2.3 Transmission Line Equations

To develop equations for the propagation of energy waves on transmissions in terms of standard circuit parameters, the parameters are defined as functions of distance along the transmission line, as they are no longer constant for the entire length. Evaluating a line integral of the electric field along the line yields the voltage as a function of distance. The equations for both the voltage and current are shown below.

$$V(z) = \alpha_1 \int \bar{E} \cdot d\bar{l} \text{ and } I(z) = \alpha_2 \oint_C \bar{H} \cdot d\bar{l}$$

The solutions for $\bar{E}$ and $\bar{H}$ for a parallel plate waveguide are given below.

$$\bar{E} = \hat{x}E_0 e^{-jkz} \text{ and } \bar{H} = \hat{y}\frac{E_0}{\eta}e^{-jkz}$$

For a plate separation a and length w, the voltage and current along the waveguide can be defined as

$$V(z) = \alpha_1 \int_0^a \bar{E} \cdot \hat{x}dx = \alpha_1 E_0 a e^{-jkz}$$

$$I(z) = \alpha_2 \oint_{C_0} \bar{H} \cdot ds = \alpha_2 \frac{E_0}{\eta} w e^{-jkz}.$$

Applying Maxwell's equations $\nabla \times \underline{\overline{E}} = -j\omega\mu\underline{\overline{H}}$ and $\nabla \times \underline{\overline{H}} = j\omega\epsilon\underline{\overline{E}}$ to the distance-dependent voltage and current functions above results in the transmission line equations,

$$\frac{dV}{dz} = -j\omega LI \text{ and } \frac{dI}{dz} = -j\omega CV,$$

where $L = \frac{\mu a}{w}$ (Henrys/m) and $C = \frac{\epsilon w}{a}$ (Farads/m). The characteristic impedance of the transmission line is defined as the ratio of the voltage to the current along the line.

$$Z_0 = \frac{V(z)}{I(z)} = \eta\frac{a}{w} \text{ ohms}$$

2.4 Energy Propagation and Reflections

A transmission line with characteristic impedance Z_0 and load impedance Z_L is shown below in Figure 2. The voltage pulse sent down the transmission line can be written as the sum of the injected pulse and any reflected wave from the load impedance. This is expressed below in terms of a forward traveling wave and a backwards traveling wave.

$$V(z) = V_+ e^{-jkz} + V_- e^{jkz}$$

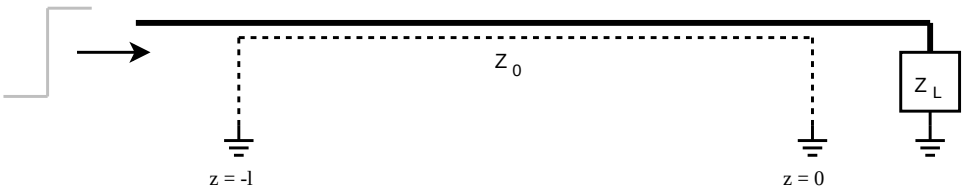


Figure 2: Transmission Line Z_0 with Load Z_L

The reflection coefficient Γ_L is defined as the ratio of the magnitude of the backwards traveling wave to the magnitude of the forward traveling wave. Note that it expresses the reflection *at* the load (hence the subscript $_{\mathrm{L}}$), and as such the voltage expression can be rewritten as

$$V(z) = V_+(e^{-jkz} + \Gamma_L e^{jkz}).$$

Given that the impedance $Z(z)$ along the transmission is

$$Z_L(z) = \frac{V(z)}{I(z)} = Z_0 \frac{e^{-jkz} + \Gamma_L e^{jkz}}{e^{-jkz} - \Gamma_L e^{jkz}},$$

it can be seen that at $z = 0$, the exponential terms reduce to unity. Rearranging the expression evaluated at $z = 0$, the reflection coefficient can be obtained in terms of the load impedance, given below.

$$\Gamma_L = \frac{Z_L - Z_0}{Z_L + Z_0}$$

This result is central to TDR technology, as it describes how the reflection is related to the characteristic impedance Z_0 of the test cable and the load impedance Z_L seen at the end of the line.

2.5 Load Terminations

The behavior of the transmission line voltage can be summarized into a few different cases of load terminations. An ideal voltage pulse is assumed to be the stimulus in the following cases.

If the load impedance is infinite, that is, the cable is somehow disconnected or broken at the end, the reflection coefficient becomes

$$\Gamma_L = \frac{\infty}{\infty} = 1.$$

A reflection coefficient of unity indicates that the reflected pulse from the 'load termination' is not inverted and returns with the same magnitude as the transmitted pulse. This situation will appear on a TDR waveform as a stair step shape, where the width of the step indicates the length of the open-ended ($Z_L = \infty$) test cable.

In the short circuit case, Z_L is 0. The reflection coefficient then becomes

$$\Gamma_L = \frac{-Z_0}{Z_0} = -1,$$

indicating that the reflection from the short at the end of the cable has the same magnitude as the stimulus but with opposite polarity. This condition translates to square pulse in the TDR waveform whose width again indicates the length of the shorted cable.

When $Z_L = Z_0$, the load impedance is said to be matched to the line impedance. In this case, it is clear that

$$\Gamma_L = 0.$$

In the "matched termination" condition, there is no reflection from the load, meaning that all of the energy in the stimulus was dissipated in the load. This is the ideal situation for transmitting electrical energy for long distances (e.g. a power grid), since all of the transmitted power ends up at the destination without incurring losses from reflections along the transmission line. On a TDR waveform, only the initial stimulus will be visible as there are no reflections from the load.

3 Device Structure Overview

3.1 PC Host/Microcontroller Interface

The TDR instrument is a PC-based measurement device consisting of three main blocks as shown in Figure 3: the host control software, an embedded ADuC7026 microcontroller, and the TDR timebase and acquisition circuits. The PC host provides a graphical user interface (GUI) for manipulating the various TDR parameters, and presents the acquired waveform on the screen with the appropriate time or distance scaling on the horizontal axis. The PC host software communicates with the TDR hardware through the parallel port, using a generic 8-bit bidirectional data bus along with a few additional control and handshaking lines. The PC software transfers high level acquisition commands and parameters to the TDR hardware, and when the acquisition is complete, the waveform data is transferred from the microcontroller memory to the PC for display. Debugging commands are also provided in the software interface to allow the developer to exercise individual components of the acquisition subsystem.

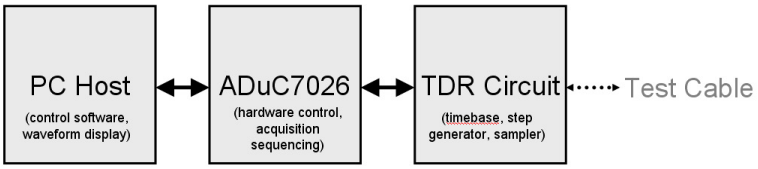


Figure 3: TDR Instrument Architecture Overview

The ADuC7026 microcontroller is the heart of the acquisition hardware. The ADuc7026 software translates the high level commands issued by the PC host into the hardware level signals to control the timebase, step generator, and sampling circuits, and performs the equivalent-time sampling sequence for the waveform.

3.2 Microcontroller/TDR Acquisition Hardware Interface

The timebase circuit determines the horizontal axis of the waveform. Using various clocks, counters, and D/A converters, the timebase generates precise delays for positioning the step and the time at which the signal is sampled. The timebase is controlled directly by the ADuC7026 through an SPI interface.

The step generator circuit is based around a 12.5 Gbps crosspoint switch SiGe-IC to create the high speed voltage step in the test cable. Two channels are available, and the polarity (positive or negative going) can be independently specified. The channels can also individually be turned on or off. Only one channel is sampled, but differential TDR measurements can still be made by measuring the common mode as well as the differential mode response, and post-processing the two waveforms. The polarity and enable controls are set directly by the ADuC7026, and the timebase initiates the step.

The sampler circuit outputs an analog voltage proportional to the value of the sampled signal, which is then read by the ADuC7026's A/D converter and is recorded in the internal microcontroller memory. The time at which the sampling occurs is controlled of course by the timebase.

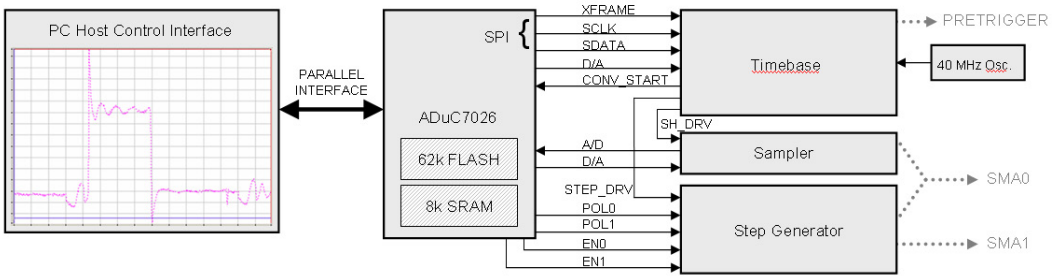


Figure 4: TDR Instrument Architecture Block Diagram

3.3 Equivalent Time Sampling

Due to the very high bandwidth and timebase accuracy required to accurately characterize the rapidly changing voltage waveforms, the TDR operates in equivalent-time mode, not real time. In an equivalent-time sampling system, each data point is acquired independently at a different delay from the (arbitrary) 'start time.'

To acquire one data point, the microcontroller (ADuC7026) configures the time base to 1) start the step generator and 2) initiate the sample/hold circuit after a precise time delay. Once the sample/hold has been triggered and the sampled voltage stabilizes, the voltage is read by the A/D converter, stored in memory, and the process repeats with a different time delay for the next data point in the waveform. The step stimulus must be generated once per data point, and it is assumed that all transients die out and the system is in steady state with zero initial conditions before the next data point is acquired. Since the sampling time delay value effectively corresponds to a single point along the test wire, a characterization of the wire as a function of distance can be determined by incrementing the time delay by an appropriate amount. An illustration of this process is shown in Figure 5.

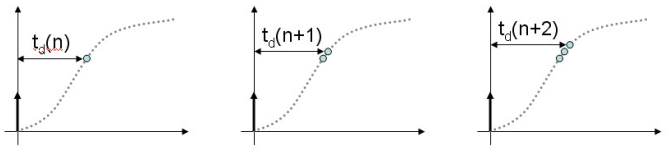


Figure 5: Equivalent Time Sampling

By controlling the delay between the simulus time and the sampling time, the shape of the voltage signal as a function of time can be reconstructed, and would appear as if it were acquired in real-time. Equivalent time sampling negates the need for a very high sampling rate; in the current configuration, each data point requires about 50 μs for the delay calculations, step generation, sampling, and data storage. This translates into about a 20 ksps sampling rate, yielding a theoretical throughput of about 19 1024 point waveform acquisitions per second. Due to the memory size limitations of the ADuC7026 SRAM, only one waveform is acquired per host request.

4 Step Generator

4.1 Circuit Overview

The step generator circuit is responsible for creating the voltage step stimulus that propagates down the test cable. Since the TDR is designed for 50 Ω cable characterization, the circuit must provide proper local termination as well as some level of electrostatic discharge (ESD) protection at the external world interface. The rise/fall time of the voltage step, along with the sampler bandwidth, are the two parameters that determine the TDR instrument's resolution, and thus it is essential to minimize the rise/fall times so that the stimulus approaches as closely as possible the ideal step. It is, however, inevitable that parasitic capacitances in the sampling bridge and impedance discontinuties at microstrip/SMA connector junctions will degrade the theoretical performance of the step generator.

To achieve very fast rise/fall times, the MAX3841 integrated circuit from Maxim Integrated Products is used. It is a 2x2 CML crosspoint switch IC manufactured on a process with 60 GHz f_T bipolar transistors, designed for switching data at fiber optic interfaces at up to 12.5 Gbps. Early performance estimations suggested that a step generator based on this chip would yield rise/fall times on the order of 50 ps, a very respectable performance figure for this TDR project. The main difficulty faced in the choice of this IC was the 24 pin 4x4 mm Thin QFN package that required special mounting techniques.

The MAX3841's two independent channels automatically provide the capability for a two channel TDR, although only one channel is sampled in the current implementation. Nonetheless, differential mode TDR measurements are still possible since the step polarity of the two channels are independently selectable. Given the measurement on one channel of both a differential and common mode step stimulus, the true differential mode behavior of the device under test can be calculated. The two channels can also be independently disabled if desired.

4.2 External Interface

Five CMOS-level control lines govern the operation of the step generator, and the output is taken from two symmetrical 50 Ω microstrip lines. The polarity selection and output enable lines are controlled directly by the ADuC7026 microcontroller, while the STEP_DRV signal that creates the stimulus is controlled by the timebase circuit. The microstrip lines provide proper impedance matching as the signal leaves the step generator to the semi-rigid SMA test cable connector. The sampler circuit is connected to the output of channel 0, at the point where the semi-rigid cable attaches to the microstrip line.

4.3 Implementation Details

The step generator circuit is designed using negative logic levels to make the 50 Ω back termination connect to ground (0 V) instead of VCC. This is required due to the output interface of the MAX3841, shown in Figure 6. The circuit operates essentially by switching a 10 mA bias current on and off with the differential bipolar transistors. If the current source is disconnected from the output, the 50 Ω termination pulls the output to the positive supply connection (0 V), and when the transistor is on, the 10 mA current develops a 500 mV voltage drop across the termination resistor, resulting in the 500 mV step. The very high speed transistors allow this switching to happen far faster than would be possible in a discrete implementation due to the parasitics involved.

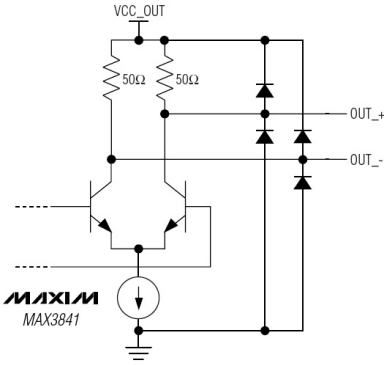


Figure 6: MAX3841 Equivalent Output Circuit

If VCC_OUT is tied to 0 V and the ground connection to VEE, the 50 Ω internal termination resistors pull the test line to 0 V instead of otherwise VCC. As a result, the chance of accidental shorts due to external ground connections is reduced. The high frequency operation may be improved since the test connector terminates on a large high frequency ground plane instead of the TDR supply voltage that is most likely noisy with noticable transients and clock feedthrough.

Connecting the MAX3841 ground to VEE (-3.3 V) introduces complications for interfacing the circuit to the timebase, and achieving perfect timing of the two independent channels while allowing for polarity selection further complicates the design. To meet these demands, negative emitter-coupled logic (NECL) circuits from ON Semiconductor were used. Low voltage CMOS (LVCMOS) to differential NECL level shifter ICs provide the interface between the 3.3 V CMOS-based timebase circuit, and a packaged dual differential ECL 2-to-1 multiplexer IC provided the

polarity selection capability. It is assumed that the propagation delays through the two multiplexer channels are essentially identical since they are in a single packaged IC, and a symmetrical layout ensures that the signal transmission delays from the multiplexer to the MAX3841 chip are the same for both channels. The cross-connected multiplexer input topology means that the polarity of each step channel can be selected by adjusting the appropriate SEL lines, while maintaining proper synchronization. The circuit schematic is presented in Figure 7.

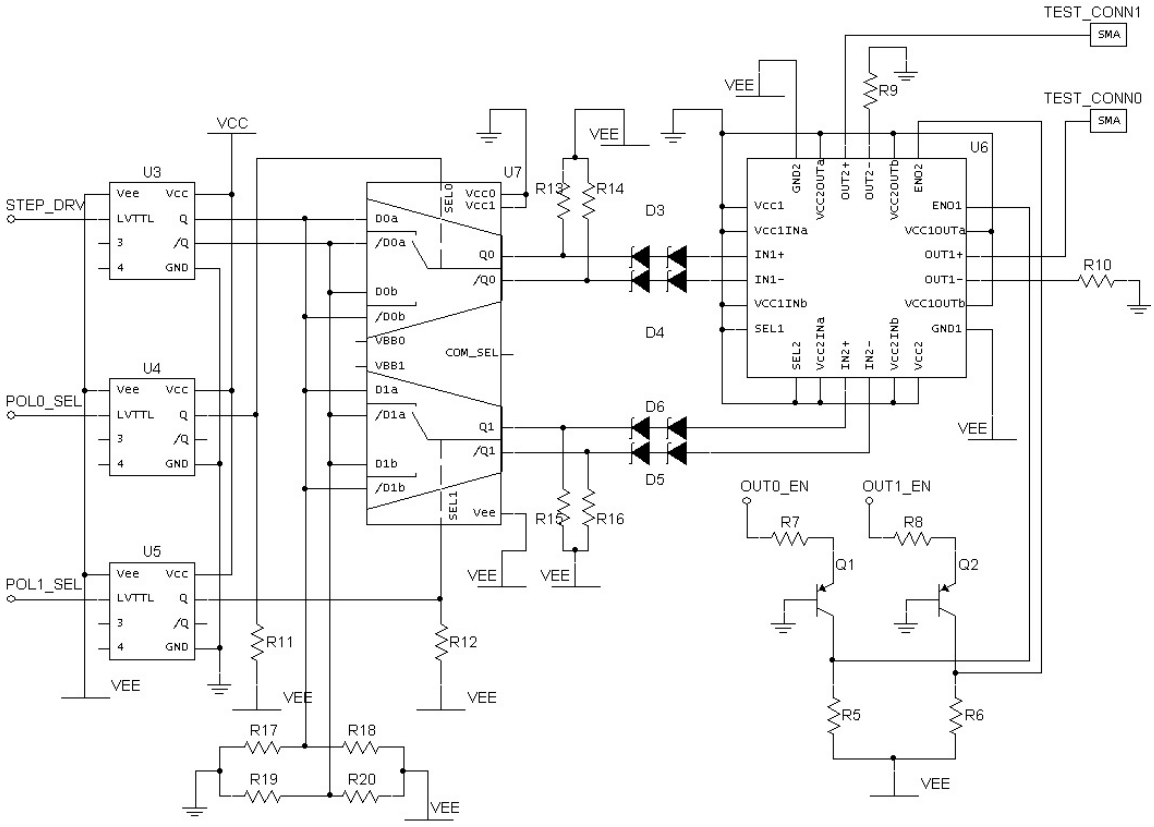


Figure 7: Step Generator Circuit Schematic

The enable/disable controls for the MAX3841 are manipulated from standard +3.3 V CMOS logic with the simple transistor level shifting circuit. When the OUT0_EN line is low, the base-emitter junction has 0 V across it, and no current flows in the transistor, pulling the select line to the negative supply. In the case that OUT0_EN is high, the transistor saturates, and the voltage developed across the two resistors is more or less the same. As such, the select line is pulled to near 0 V or slighly above due to the transistor's base residing at 0 V.

Such simple level shifting circuits are not sufficent for the high performance sections of the circuit, as proper external termination networks are essential for correct operation of the NECL/ECL logic family. The output of the polarity selection level shifter ICs are simply terminated with 1 kΩ (R11, R12) to the negative supply, since high performance is not essential for setting the polarities. For optimal performance, each differential output of the STEP_DRV level shifter must be DC biased at

2 V below the positive supply voltage, which in this case would be -2 V, and the impedance seen at the output node must be 50 Ω. The choice of 130 Ω (R17, R19) and 82 Ω (R18, R20) resistors give an parallel impedance of about 50 Ω, and at the same time form an appropriate voltage divider between 0 V and VEE to give the -2 V DC level.

At the output of the multiplexer, however, the termination network is different due to the fact that the MAX3841 uses current-mode logic (CML) instead of ECL. As such, two Schottky diodes are placed in series, and are pulled up to 0 V through the internal 50 Ω input terminations inside the MAX3841. Forward biasing them with resistors R13-R16 (100 Ω) shift the ECL voltage up by about 0.6 V to achieve sufficent turn on/off voltages for the bipolar CML input of the MAX3841. Originally, 82 Ω resistors were used in place of the diodes to obtain the same termination scheme as discussed before, but the step rise/fall times were quite sluggish due to the improper interface between the ECL and CML logic families.

4.4 Power Consumption

Due to the small resistor values in the termination networks, the power consumption of the step generator circuit is by no means small. With the non-negligible dissipation of the logic ICs coupled with the numerous termination networks, the circuit draws about 390 mA from the negative supply. Disabling the step generators does not significantly reduce the current draw, and a complete power management scheme must be developed if this circuit is to be viable in battery operated field applications. In fact, between waveform acquisitions, the entire negative supply could be shut down with minimal extra circuitry that could be readily controlled by the ADuC7026 microcontroller.

5 Sampler

5.1 Overview

The sampling circuit is a two stage circuit that very quickly samples the input signal and converts the signal into a proportional output that is read by the ADuC7026 A/D converter. The sampler's rise time must be comparable to that of the bandwidth of the signals of interest. The faster the sampler, the sharper the edges in the TDR waveform will be, and thus the precision with which discontinuities and ranging measurements can be made is larger. Because the sampler circuit is based on proprietary technology from Hyperlabs Inc., only a cursory overview will be pursued. Correct functioning of the circuitry is also extremely sensitive to circuit board layout considerations and the choice of specialized components.

5.2 Circuit Description

A standard diode bridge switch forms the point of interaction between the test cable connecter and the first (high frequency) stage of the sampler circuit, as shown in Figure 8.

Normally, the DC voltage source V_R reverse biases the diode junctions of the serially connected dual Schottky diodes D1 and D2. In this reverse bias condition, essentially no current flows through the diodes, presenting an extremely high impedance node at the SMA test connector M1. If the diode bridge were forward biased, the voltage at the common terminal of D2 would track the test connector voltage directly due to the constant voltage drop across D1 and D2 from their common anode, provided that the forward bias current generators have sufficent voltage headroom. The

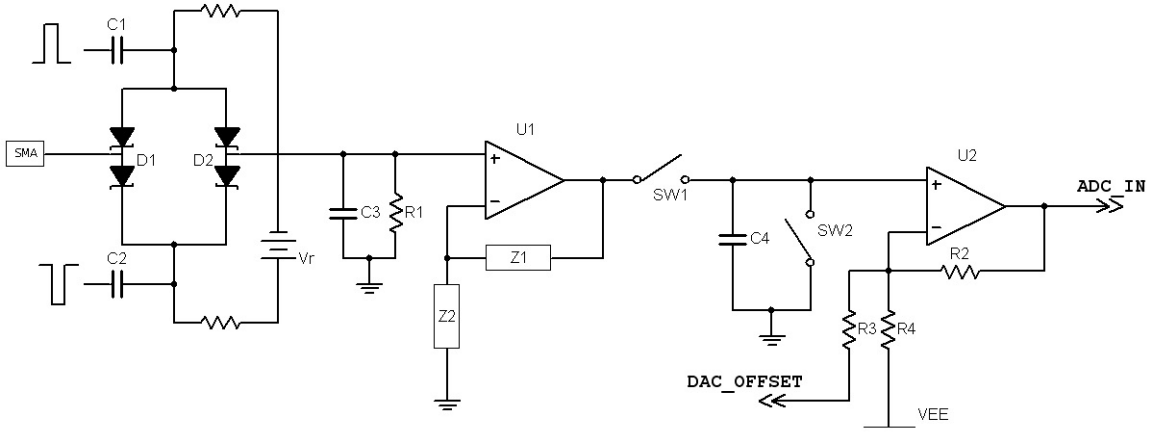


Figure 8: Sampler Circuit

goal, then, is to 'open' the sampling bridge by forward biasing it for a very short instant at the time when the test signal is to be sampled. This is achieved by injecting extremely narrow high amplitude differential pulses into coupling capacitors C1 and C2. In steady state, C1 and C2 decouple the sampling bridge from the differential strobe generator circuit, allowing for the DC reverse bias condition described above. When the positive and negative strobe pulses are applied, C1 and C2 act as AC shorts, and the reverse bias voltage is temporarily overcome by the strobe signal, thus forward biasing D1 and D2. In the short time that the sampling gate is open, a small amount of charge is injected into C3 that is proportional to the voltage on the test connector. R1 simply provides a discharge path for C3 to reset the capacitor in between samples. The time constant is chosen to be large enough for the rest of the sampling circuit to respond before C3 is drained, but at the same time short compared to the natural frequency of the impulse response of the next stage.

In the second stage, the charge transfer into C3 acts as a small impulse excitation of opamp U1 and its resonant feedback network formed by Z1 and Z2. Z1 and Z2 are chosen to give a slightly underdamped second order impulse response as shown in Figure 9, taking into consideration the frequency response of the opamp itself. The undamped natural frequency ω_0 of the opamp response is thus fixed, and hence the amount of charge transfer from the sampling gate determines essentially the amplitude of the second-order response. Since the excitation generally lasts on the order of 1 to 3 μs, there is no longer a need for extremely high bandwidth buffers and high frequency switching to capture the test signal voltage onto a capacitor.

The sampling switches SW1 and SW2 control the charge accumulation on the hold capacitor C4. SW1 is controlled by a monostable multivibrator that closes it for a fixed time each time the sampling gate is opened and the impulse response of U1 is excited. The charge transferred to C4 will be proportional to the magnitude of the impulse response, and thus also to the voltage at the test connector. After SW1 opens, the voltage on C4 is amplified and an offset is added to bring it into an appropriate range and scale for the A/D converter. Once the voltage has been digitized and the value stored in the microcontroller memory, SW2 is closed to reset the hold capacitor. By this time, C3 has also discharged through R1, and the sampler circuit is ready to capture the next

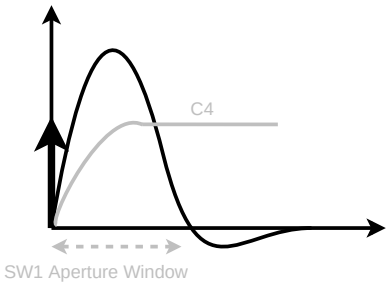


Figure 9: 2nd Order Charge Amplifier Impulse Response

data point.

5.3 Stability and Linearity Considerations

The open-loop configuration of this sampler architecture renders it unconditionally stable, regardless of the forward path gain. Optimal linearity is not necessarily achieved with this architecture, however. It is worth noting that vertical scale linearity is not essential for basic TDR distance measurements, since only the relative time positions of the stimulus and the reflection are of interest. If close observation of changes in the cable dielectric are desired, vertical scale linearity becomes more of a concern. As such, the main factors affecting sampler linearity are discussed next.

5.3.1 Sampler Reverse Bias Condition Symmetry

The sampler linearity is very highly dependent on the midpoint of the DC reverse bias voltage (e.g. the DC voltage appearing at the common node of D2) when the bridge is off. Although it is not shown in sampler circuit (Figure 8), a D/A converter from the microcontroller is used to set the midpoint of V_R, so that it can be adjusted to the center of the dynamic range expected for the test signal. By moving the apparent voltage across the diodes in accordance with the value of the input, better linearity is achieved since the symmetry of the diode switch relative to the input level is maintained.

The current implementation, however, sets a fixed reverse bias point that is assumed to be in the middle of the dynamic range of the expected signal. An open circuit at the end of the test cable will result in a ΔV_{IN} of about -500 mV, since the step amplitude with 50 Ω termination is -250 mV. As a result, the D/A converter is programmed to set the midpoint of V_R to approximately -250 mV, and it is assumed that even if the acquired waveform is not perfectly linear, it should be at least symmetrical around the -250 mV level.

A digital feedback mechanism could be implemented to adjust the reverse bias midpoint based on previously acquired data values to improve linearity. Given a mechanism for scaling the A/D converter output to the actual voltage seen at the test connector, linearity could be increased while still maintaining unconditional stability in the analog circuit. One possible way to achieve this would be to generate lookup tables of the sampler output at a known reverse bias midpoint when appling a known DC voltage to the test connecter. Such a table could be used to map the A/D converter output to an appropriate D/A value to set the reverse bias, thereby implementing a

digital feedback mechanism. Another benefit would be the capability to calibrate the vertical scale to actual voltages, instead of raw A/D converter codes.

5.3.2 Differential Strobe Amplitude and Symmetry

Sampler linearity is also affected by the amplitude and symmetry of the differential pulses that temporarily open the sampling gate. If the amplitude of the strobe is not high enough to both overcome the reverse bias and sufficiently forward bias the sampling gate, the charge transfer to the C3 will be reduced, and the signal-to-noise ratio of the sampler will suffer as well as the linearity given a fixed bias midpoint voltage. If the positive and negative strobe pulses are not symmetrical, the upper and lower halves of the sampling bridge will not be equally forward biased at the sampling instant, and thus test input voltages that are higher than the DC midpoint will result in a different charge transfer than lower input voltages. Designing and constructing a circuit to generate such pulses is indeed a rather involved enterprise, and will not be pursued in this document. The bandwidth of the sampler circuit is determined by the width of the strobe pulses, rendering the overall functionality of the TDR instrument extremely dependent on the characteristics of the strobe signals.

5.3.3 Closed Loop Error-Sampled Feedback Topology

Analog closed loop error-sampled feedback loop topologies in which the voltage on the hold capacitor is fed back directly to adjust the reverse bias midpoint can potentially achieve greater linearity. This is often at the expense of requiring heavy oversampling at the same time delay to allow the feedback loop to 'catch up' to the correct voltage level due to a small loop gain. Experimentation shows that in practice the loop gain must be limited to much less than unity to maintain stability and to avoid unnecessary overshoot, especially given component variations. In the case that the waveform is not sufficiently oversampled, the measured rise times of the stimulus and energy reflections will appear longer than they are in reality due to the small loop gain.

5.4 Example SPICE Simulation

A simplified SPICE simulation of the impulse response of the charge amplifier and the voltage developed on the hold capacitor is shown. A generic CMOS opamp model from Texas Instruments is used, and the excitation is a 100 ps wide voltage pulse.

The SPICE modeling appears to yield reasonable results. A better modeling of the system would include the diode bridge, bias generators, and strobe injection, as well as parasitic effects, to fully simulate the inherent nonlinearities in the semiconductors and the true charge injection and amplification mechanisms. Such simulations however are not pursued in this report.

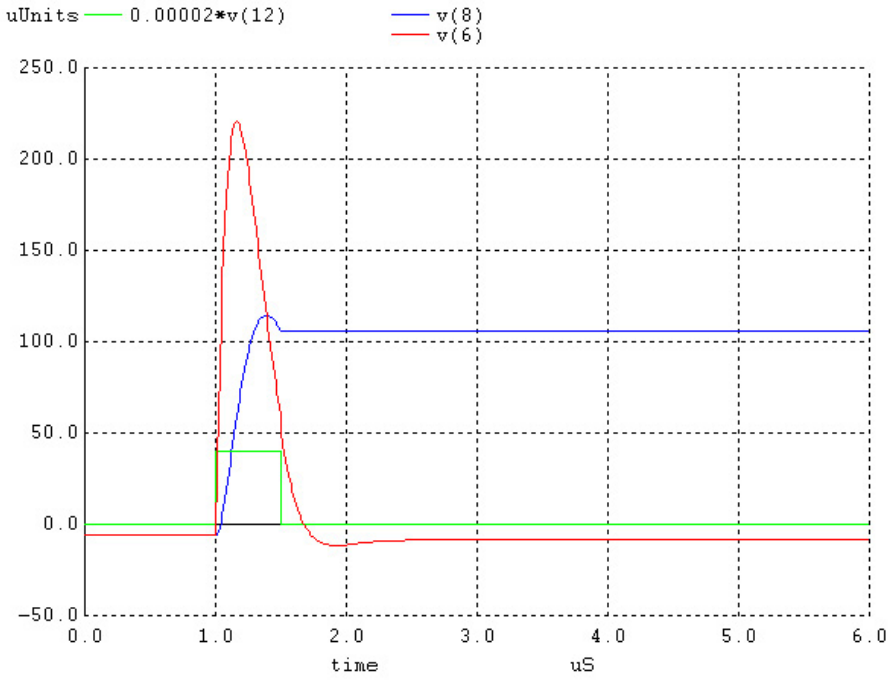


Figure 10: Sampler Charge Amplifier SPICE Simulation

6 Timebase

6.1 Functional Overview

The timebase circuit is responsible for controlling the precise synchronization between the step generator and the sampling circuit necessary to produce the TDR waveforms in equivalent time. It also provides timing capabilities for starting the A/D converter after the sampling has completed. Timing resolution on the order of tens of picoseconds (or less) must be achieved, and to this end a hybrid digital/analog timing circuit is used. A time delay is split into 'integer' counts and 'fractional' counts, where the integer part of the delay is controlled by a digital timer run off a crystal oscillator, and the fractional part is generated by comparing an analog ramp voltage and a DC reference set by a D/A converter. The architecture is summarized in Figure 11.

The digital integer delay counters are implemented in a Xilinx CPLD using VHDL, and the analog fractional delay generator is based on current source injecting charge onto a capacitor. The timebase circuit is controlled directly by the ADuC7026 microcontroller through an SPI interface that programs the initial step and strobe delay count values, and then initiates the timing sequence. The fractional delay is set directly by one of the 12-bit D/A converters built into the ADuC7026.

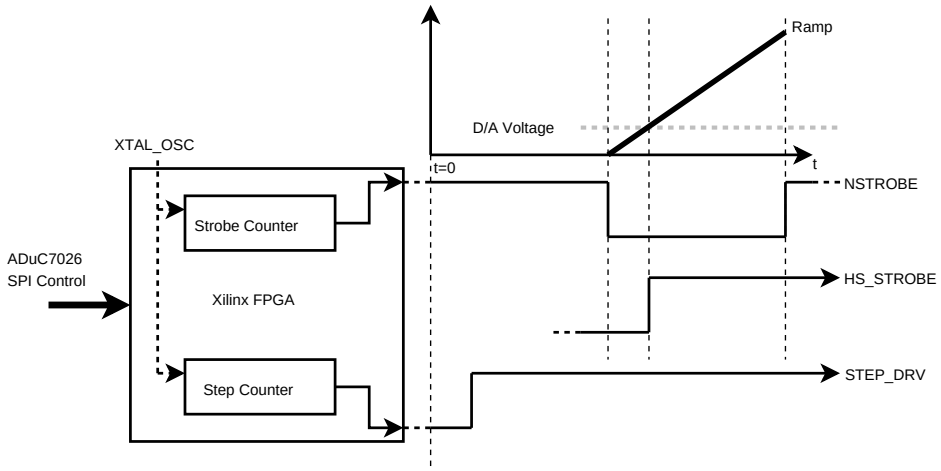


Figure 11: Timebase Block Diagram

6.2 Timing Requirements

A hybrid timing method is necessary because it is not possible to achieve fine enough resolution with a purely digital approach due to the unreasonably high clock frequencies that would be required (for 12 ps timebase resolution, an 83 GHz clock is needed), and a purely analog approach would have very poor jitter performance and reduced accuracy because of a slow ramp and the resulting time uncertainty of the comparision with the D/A converter reference level. Given a 20 MHz clock, 12-bit D/A converter, and a ramp that spans the entire 12-bit D/A voltage range, the minimum time resolution can be readily calculated.

The clock period is $\frac{1}{20 \cdot 10^6} = 50$ ns, meaning that each digital integer count corresponds to a time delay of 50 ns. Given a ramp that covers the entire 12 bit D/A converter voltage range in one integer count means that each 50 ns can be subdivided into 4096 spacings, giving a time resolution of $\frac{50 \text{ ns}}{4096} = 12.2$ ps. It is possible to obtain very precise timing without the need for outrageous clock frequencies, while keeping the analog ramp steep enough to reduce jitter resulting from uncertainty and variation at the comparator.

6.3 Debugging Facilities

In order to allow for proper debugging of the timing signals produced by the timebase circuit, a special *pretrigger* output is included. The pretrigger signal is a positive going pulse that always occurs at the time when the timers are enabled. By triggering an oscilloscope on this signal, both the step and strobe delay outputs can be simultaneously viewed relative to a fixed time, instead of just relative to each other. The pretigger output includes a 450 Ω series resistor connecting to a SMA connector, giving a built-in 10x attenuation for connecting directly to high speed sampling oscilloscopes. The 3.3 V CMOS level pretrigger output is also available on a pin header for use with standard real-time scopes.

6.4 Xilinx CPLD Digital Timers

6.4.1 Clocking and Sychronization

The timebase is clocked using an external 40.000 MHz crystal oscillator connected to one of the global clock input pins on the Xilinx CPLD. The 40 MHz clock is internally divided by two to generate a 20 MHz clock that is brought out to a pin, which is externally connected straight to a second global clock input. This way, the built-in clock distribution network of the Xilinx CPLD is fully utilized, maximizing the time synchronization between the various counters and reducing internal clock skew.

The step delay counter is run off the 40 MHz clock, allowing the stimulus to be positioned in 25 ns increments from the reference time. The strobe delay counter is clocked off the externally routed 20 MHz clock, which translates to each integer count representing 50 ns of real world delay. This dual clock scheme was pursued initially to allow the stimulus to be positioned at two different positions while keeping the integer strobe delay constant, thereby presenting a potential means of calibrating the fractional delay ramp circuit. This mechanism did not prove successful, as even higher step placement resolution would have been required to properly detect the most linear 50 ns section of the ramp. Other potential hardware calibration mechanisms are explored in Section 12.5.

Proper synchronization of asynchronous signals to the clock is required to obtain very low jitter performance. For example, the XFRAME signal from the ADuC7026 microcontroller is asynchronous to the timebase clocks, and is thus stepped through a series of flip-flops to synchronize the signal to the internal clock. The synchronous internal signal is called IFRAME(..), where IFRAME(0), IFRAME(1), IFRAME(2) are each delayed one more clock period from the previous one. The internal load and pretrigger signals are formed by logical combinations of these synchronized IFRAME signals. Synchronization via flip-flop delays is also used in the counters to ensure that the terminal count (and thus STEP_DRV and NSTROBE) signals are properly aligned with the clock transitions to reduce jitter. No clock gating is used to control the timers; instead, the clocks are left freely running, and reset and load signals are used to control the counter operations.

6.4.2 SPI Programming Interface

The digital timebase is programmed through a two byte MSB-first SPI transfer. The serial clock and serial data lines from the ADuC7026 control directly a 16 bit serial shift register. Serial data is read on the rising edge of the clock. After the serial data has been loaded, the upper and lower bytes in the shift register are transferred to the step and strobe 8 bit up counters. The VHDL code implementing the shift register entity is given below.

```
entity sr is
port (
  sck : in std_logic;
  sdi : in std_logic;
  sdo : out std_logic;
  strobe_val : out std_logic_vector(7 downto 0);
  step_val  : out std_logic_vector(7 downto 0)
 );
end sr;

architecture Behavioral of sr is
signal q : std_logic_vector(15 downto 0);
begin

strobe_val <= q(7 downto 0);
step_val <= q(15 downto 8);
```

```
sdo <= q(15);

process(sck)
begin
  if rising_edge(sck) then -- shift in on rising edge of clock
    q <= q(14 downto 0) & sdi;
  end if;
end process;
end Behavioral;
```

A third control line from the microcontroller synchronizes the loading of the data from the SPI shift register into the counters, and starts counting. When the XFRAME signal is low, the counters are disabled. It is at this time that the serial data is loaded. When XFRAME goes high, the counters are enabled and the data point acquisition commences.

6.4.3 Counter Structure

The step and strobe delay counters are 8 bits and count upwards to 0xFF. Upon reaching terminal count, the counter disables itself and does nothing until it is reprogrammed with a new start value. The relevant portion of the counter code is reproduced below.

```
tc <= '1' when q="11111111" else '0'; -- terminal count
tcasync <= tc;  -- asynchronous terminal count output
process ( clk, srst )
begin
  if srst = '0' then -- hard reset signal
     q <= "00000000";
  elsif rising_edge(clk) then
    if load='1' then -- load = sync'd XFRAME
        q <= input;
    elsif tc='0' then
      q <= q + 1;
    end if;

    tc0 <= tc;
    tc1 <= tc0;
    st  <= tc and (not tc1);
    nst <= (not tc) or tc1;  -- not st (synchronized to clock)
    st_high <= tc;
  end if;
end process;
```

6.4.4 A/D Conversion Timing

The A/D converter in the ADuC7026 is triggered by a signal from the timebase that goes high 3 μs after the STROBE signal is asserted and thus the sampler is started. The 3 μs delay provides ample time for the sampling circuit's charge amplifier impulse response to settle, and for the voltage on the hold capacitor to stabilize. The delay is implemented by a down counter with a preprogrammed initial value of 60. At a 20 MHz clock frequency, this value results in the desired delay. The VHDL implementation of the convert start counter is provided below.

```
entity convs is
port(
  rst : in std_logic;
  clk : in std_logic;
  cs : out std_logic
  );
end convs;

architecture Behavioral of convs is
```

```
  signal tc : std_logic;
  signal q : std_logic_vector(5 downto 0);
begin
tc <= '1' when q="000000" else '0';
process ( clk, rst )
begin
  if rst = '0' then
    q <= "111100"; -- start value = 60 (20mhz clock (50ns period): 3us delay
  elsif rising_edge(clk) then
    if tc='0' then
      q <= q - 1;
    end if;
      cs <= tc;
  end if;
end process;
end Behavioral;
```

6.4.5 Sequencing Control

A timing diagram of the internal top-level timebase sequencing is shown in Figure 12.

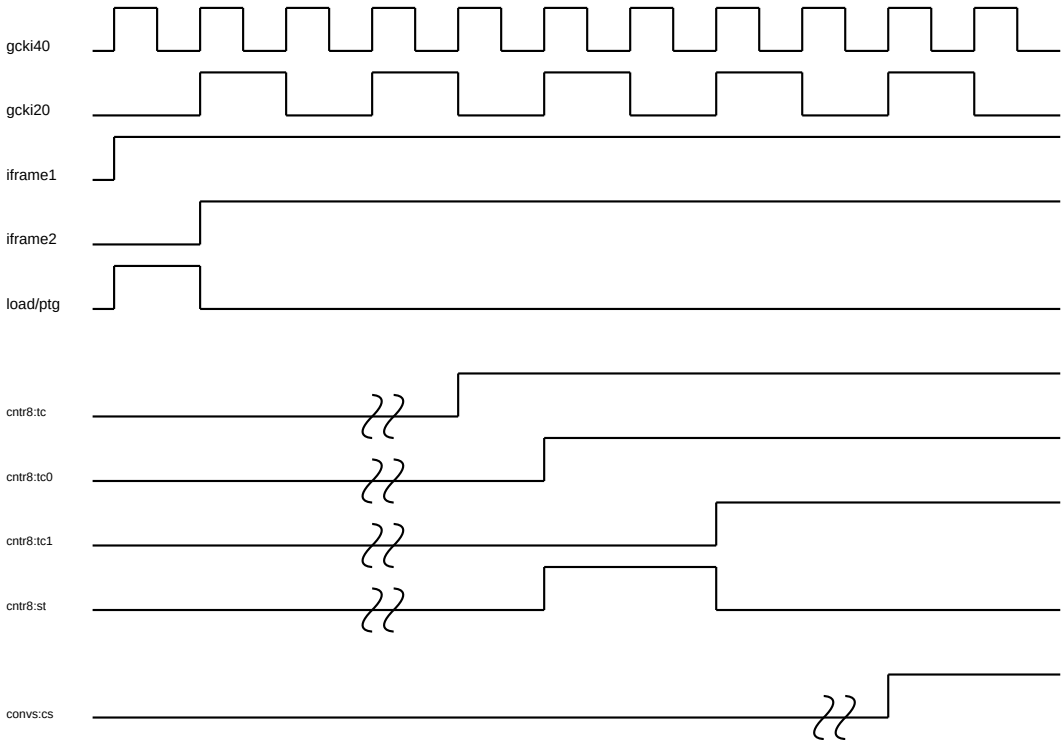


Figure 12: Timebase Implementation Timing Diagram

The XFRAME signal is synchronized on the 40 MHz clock to the internal IFRAME signals, and the LOAD/PRETRIGGER signal is generated by IFRAME1 $\cdot$ $\overline{\text{IFRAME2}}$. When IFRAME1 is low, the counters are held in reset, and the new delay values are copied from the SPI shift register to the counters when LOAD is high. After the load, the counters are enabled and the data point acquisition begins.

6.5 Analog Fractional Delay Generator

The ramp generator circuit consists of two current sources, a charge capacitor, diode-based current switch, and a high speed comparator IC. A simplified schematic diagram is provided in Figure 13.

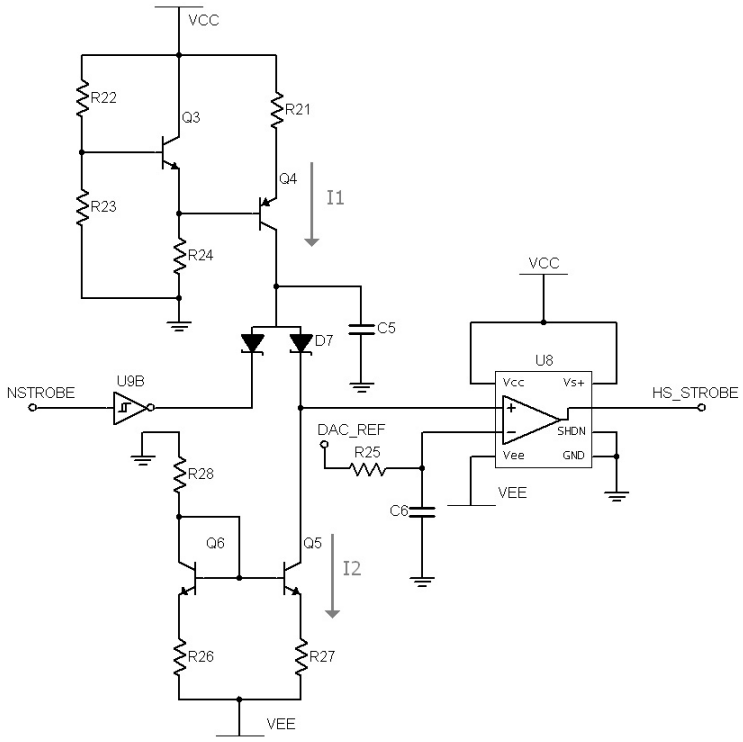


Figure 13: Ramp Generator Circuit

The upper current source pushes about 2 mA into the common anode of diodes and onto the ramp capacitor C5. Resistors R22 and R23 set a fixed voltage at the base of buffering transistor Q3. The voltage seen at the base of Q4 is one base-emitter drop lower than at the common terminal of R22 and R23, but because of the npn-pnp structure, the voltage across R21 and R22 is essentially the same, given a matched pair of transistors. Setting the value of R21 fixes the collector current of Q4. R24 simply pulls down the emitter of Q3 to provide a DC current path to keep Q3 turned on. This pnp-npn structure has good thermal characteristics, as the changes in base-emitter voltages of the two transistors due to temperature cancel each other out, thereby stabilizing the current. The current source is somewhat sensitive to supply variation, and must be locally decoupled with a capacitor.

The lower current source pulls down about 1 mA from the cathode of the right half of D7. Thus, the capacitor charging current is the difference between I1 and I2. When NSTROBE is high, the inverter output is low, and both halves of D7 are forward biased. This keeps the voltage across the capacitor one diode drop above ground, introducing a fixed pedestal into the ramp at the capacitor. Due to the good component matching in the dual packaged Schottky diode D7, the voltage at the postive input of the comparator should remain close to 0 V. When NSTROBE goes low, the left

half of D7 becomes reverse biased, and the current difference between I1 and I2 is forced into C5. As this is a constant current, the voltage on C5 will increase linearly, as will the voltage seen at the comparator, except it is a few tenths of a volt lower. The slope of the ramp is given by

$$\frac{dV}{dt} = \frac{I_c}{C}$$

By trimming the currents via resistors R21 and R28 and choosing an appropriate capacitor, the ramp can be configured to span as much as possible of the D/A converter's range in the 50 ns time window. A very high speed comparator from Linear Technologies provides very fast transistion times and introduces approximately only a 4 ns delay from input to output.

This Schottky diode current switching scheme is used to avoid the problems of bipolar transistor saturation. Using a transistor to short the ramp capacitor would introduce inconsistent delays when the transistor is pulled out of saturation to enable the ramp, due to the time necessary to clear out the carriers in the base.

The D/A reference voltage has a small RC network placed close to the comparator to stabilize the voltage and immunize it somewhat from noise. The time constant is chosen to be small enough to have no appreciable effect on the circuit timing.

6.6 Performance Characterization

An oscilloscope screenshot of the various timing signals generated by the timebase circuit is shown in Figure 14.

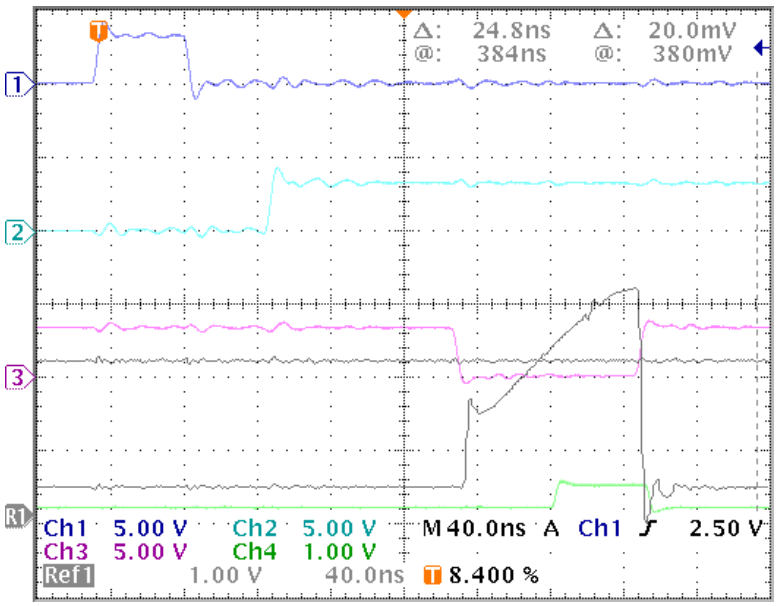


Figure 14: Timebase Oscilloscope Screenshot

Trace 1 is the pretrigger signal, trace 2 the STEP_DRV signal, trace 3 the NSTROBE signal, and trace 4 the HS_STROBE signal. The remaining waveforms show the analog ramp and the DC

reference set by the D/A converter. It is clear that the timebase circuit is functioning properly.

7 Microcontroller

7.1 Organization

The ADuC7026 microcontroller controls the acquisition hardware and communicates with the host controller. It provides a 16 channel 12-bit A/D converter and a 4 D/A converters, a master/slave SPI port, five general purpose I/O ports, and contains an 8k data SRAM and 62k program FLASH memories. The 32-bit ARM7TDMI core allows for the fast 32-bit x 32-bit multiplies necessary to compute the timebase parameters for each data point acqusition. The hardware resources used are discussed next.

7.2 Hardware Resource Allocation

Efforts were made to fully utilize the hardware capabilities of the ADuC7026 to simplify the hardware design and to reduce the overall TDR's power consumption. The microcontroller's pulse width modulation, RS-232 serial port, external SRAM interface, programmable logic (PLA) unit, comparator, I^2C, and internal timer hardware did not find application in this project.

7.2.1 Digital I/O

The digital resources used are listed in Table 1 from the hardware port perspective.

Port/Pin	Chosen Functionality	Purpose
P1.4	SCLK	SPI Clock (SCK) for timebase configuration
P1.6	MOSI	SPI Master Out Slave In (SDA) for timebase configuration
P1.7	GPIO (out)	FRAME signal for timebase configuration
P2.0	GPIO (in)	A/D converter start signal from timebase
P2.4	GPIO (out)	Channel 0 step polarity (0=negative going, 1=positive going)
P2.5	GPIO (out)	Channel 1 step polarity (0=negative going, 1=positive going
P2.6	GPIO (out)	Channel 0 step enable (inverted)
P2.7	GPIO (out)	Channel 1 step enable (inverted)
P3.7-3.0	GPIO (in/out)	Bi-directional 8-bit data port for host communications
P4.0	GPIO (out)	Communication RDY signal
P4.1	GPIO (out)	Communication ACK signal
P4.2	GPIO (in)	Communication CLK signal
P4.3	GPIO (in)	Communication RQ signal

Table 1: Digital Resource Allocation

The SPI port was used in master mode, with phase='1', polarity='1', MSB first, and with loopback and continuous transfer modes disabled. The SPI clock divider was set to give a comfortably slow serial clock of about 3 MHz.

7.2.2 Analog I/O

The analog I/O hardware allocation in the ADuC7026 is listed in Table 2. The internal 2.5 V bandgap reference was used to set the A/D and D/A converter voltage ranges, and the A/D converter was used in software-initiated conversion mode.

Resource	Purpose
ADC0	Reads sampler circuit output to collect waveform data
DAC0	Sets the fractional time delay in the timebase
DAC1	Sets the upper value of the DC reverse bias voltage for the sampling gate
DAC2	Adds a DC offset to the output of the sampler voltage

Table 2: Analog I/O Resource Allocation

7.3 Embedded Software Overview

7.3.1 Programming Model

Upon startup, the ADuC7026 configures all of the hardware resources previously described by writing to the appropriate memory-mapped control registers. After the hardware has been set up, the microcontroller enters an idle loop, awaiting a communication request from the host. If the communication request is determined to be a device-to-host transfer, the raw data stored in the waveform buffer is written to the host. Otherwise, the microcontroller reads a parameter vector from the host that specifies what action should be taken, as well as various acqusition time and offset values.

Depending on the first byte in the vector (the command parameter), the microcontroller may initiate a waveform acquistion, or it may enter a debug mode in which no waveform is acquired but the various D/A and timebase control parameters are manually configured to allow the developer to test the functionality of each subcomponent. A *freerun* mode is also provided that repeatedly applies the parameters and retriggers the step generator and timebase circuits to allow the signals to be viewed properly using high bandwidth sampling oscilloscopes. The microcontroller remains in this mode until a new parameter vector is read from the host and the *freerun* parameter is turned off.

If the command byte specifies a waveform acquisition, the waveform parameters are applied, and the microcontroller initiates an acquisition loop that reads and stores a data point in memory, increments the timebase, and repeats for each data point. The waveform data is only transferred to the host when a device-to-host transfer is initiated.

7.3.2 Parameter Vector Definition

In the current implementation, the parameter vector is 29 bytes long. A complete list of the parameters and allowed values is given in Table 3.

The parameter vector allocation was chosen to provide backwards compatibility with all versions of the debugging software and streamlined TDR interface software developed. Even though parameter vector is probably longer than necessary, it maintains simplicity in both the host and microcontroller. Optimizations could involve combining true/false fields into one byte instead of

wasting 7 bits of information per true/false parameter, as well as mapping different meanings onto the same field depending on the command byte. Such optimizations were not pursued for the purposes of this project, and the slight performance degredation was simply neglected.

The debugging TDR host interface application described in Section 9.2 provides direct access to these parameters for testing TDR functionality. The streamlined interface (Section 9.3) uses a higher level abstraction by specifying start and end time values for the waveform and relies on the microcontroller to generate the appropriate delays and D/A converter codes to produce the waveform.

7.4 Debugging/Acq1 Mode

In debugging mode, the microcontroller simply writes the timebase delay values (IDX_STEP_DLY, IDX_STROBE_DLY), sets the step generator enable and polarity controls, and writes the three D/A converter values. It then returns to idle mode, awaiting a new set of parameters. If the freerun (IDX_FREERUN) parameter is set, then the timebase is repeatedly reprogrammed with the same values in the idle loop to provide retriggering capabilities for debugging.

In the debug acquisition mode (Acq1), the microcontroller first sets the step generator enable and polarity controls, and applies the reverse bias (DAC1) and offset (DAC2) values. It then reads the D/A converter calibration parameters for the start and end values of the D/A that map to a 50 ns linear section on the analog timebase ramp, and uses these values to calculate the 'time' position of the current data point in terms of the number of available D/A values. From this time value, the integer delay count and fractional delay count are calculated. The loop iterates 1024 times to capture the entire waveform. The C code that implements this simple acquisition sequencing is reproduced below.

```
...

int i;
UINT8 step_dly = params[IDX_STEP_DLY];
UINT8 incr = params[IDX_INCREMENT];
UINT16 dac0start, dac0end;
dac0start = (((UINT16)params[IDX_CALSTART_UPPER]) << 8) | ((UINT16)params[IDX_CALSTART_LOWER]);
dac0end = (((UINT16)params[IDX_CALEND_UPPER]) << 8) | ((UINT16)params[IDX_CALEND_LOWER]);

// configure the step generators
step_enable( params[IDX_CHAN0_EN], params[IDX_CHAN1_EN] );
steppolarity( params[IDX_CHAN0_POL], params[IDX_CHAN1_POL] );

// set the sampler reverse bias, adc offset
WRITE_DAC1( dacvals[1] );
WRITE_DAC2( dacvals[2] );

// run acquisition sequence for a fixed record length (1024)
for (i=0;i<NPOINTS;i++)
{
  // calculate the strobe integer and fractional delays
  UINT32 ndaclevels = dac0end - dac0start;
  UINT32 delay = ndaclevels * params[IDX_START_DLY] / 255 + params[IDX_STROBE_DLY] * ndaclevels;

  UINT32 time = delay + incr * i; // this the current sample time
  UINT32 intg = time / ndaclevels;
  UINT32 frac = time % ndaclevels;
  UINT16 dac0val = dac0start + frac;

  // set the ramp level (fractional delay, DAC0)
  WRITE_DAC0( dac0val );

  // write the timer and start the point acquisition
```

```
  xspi_write_delays( step_dly, 254 - intg );

  // Polling the CONV_START pin that is driven by the xilinx timebase
  while(!READ_CONVS);
  // perform the conversion
  ADCCON = 0xA3; // software conv., single-ended, conv. enabled
  while (!ADCSTA); // wait for end of conversion

  // extract the ADC result
  UINT16 result = (UINT16) ((ADCDAT & 0x0FFF0000) >> 16);

  // copy the result into the waveform data
  rawdata[2*i] = (UINT8) ((result & 0xFF00) >> 8);
  rawdata[2*i+1] = (UINT8) (result & 0x00FF);
}

...
```

The `ndaclevels` variable represents the number of fractional delay values available, and the `time` variable is expressed as a count of D/A levels. As such, the integer delay count for the timebase is simply the integer division of `time` by `ndaclevels`. The fractional delay value is then the remainder of this division, and the ramp D/A converter value is the calibration start value added to the remainder. This is a rather clunky and non-intuitive mechanism for sequencing the waveform data points, but was implemented simply to exercise the concept in the debugging interface. The software delay generation model used by the other acquisition command (Acq2) is much more intuitive and user-friendly, as it is designed to be easily controlled from a TDR interface rather than from a debugging tool, by a person unfamiliar with the inner workings of the instrumentation.

7.5 TDR Acquisition (Acq2) Software Model

In Acq2 mode, the host control interface specifies only the waveform start time, end time, record length, ramp calibration parameters, and the usual step generator configuration values. The burden is then left to the microcontroller to determine the proper delay values.

7.5.1 Time Representation

Time in Acq2 mode is represented essentially as a 32-bit unsigned integer. The upper 16 bits represent the integer count delay, while the lower 16 bits represent the fractional delay. To allow easy access to the integer and fractional parts of a time value in the microcontroller code, structures and unions are used to define the data types `_delay8`, `_delay16`, and `timeinf`. The C definitions are reproduced below. Note that the ordering of the fields is specific to a little-endian processor core, which is the case for both the Intel x86 based PC host as well as the ARM7 core in the ADuC7026.

```
typedef unsigned char UINT8;
typedef unsigned short UINT16;
typedef unsigned long UINT32;

struct _delay16
{
  UINT16 frac_val;
  UINT16 int_val;
};

struct _delay8
{
  UINT8 b0;
```

```
  UINT8 b1;
  UINT8 b2;
  UINT8 b3;
};

union _timeinf
{
  UINT32 time;
  struct _delay16 time_s;
  struct _delay8 time_b;
};

typedef union _timeinf timeinf;
```

The `timeinf` union is convenient because it allows simultaneous access to all 32 bits, the logical integer/fractional sections, as well as individual bytes (useful for breaking down time representations for transferring via the communication protocol). It is straightforward to increment the time counter for the next data point in the waveform, because the fractional delay automatically 'wraps' to the correct integer count value without need for modulo arithmetic or other computations.

Each integer count in the time representation corresponds directly to the period of the clock that runs the counters in the timebase. With a 20.000 MHz clock, each integer count represents 50 ns in real world time, and each 16 bit fractional count represents $\frac{50 \cdot 10^{-9} \text{ S}}{2^{16}} = 0.763$ ps. Thus, the PC host software can correctly calibrate the time scale on the waveform graph.

7.5.2 Delay Calculation and Sequencing

Using the time representation described above, it is relatively simple to generate the proper sequencing of the counters and the ramp D/A for each data point. Given the 32-bit start time, 32-bit end time, and the record length in the parameter vector, the 32-bit increment value is given by

$$\text{incr} = \frac{\text{end.time} - \text{start.time}}{\text{reclen}}.$$

The integer delay for the timebase is directly available from the `timeinf` structure without any computation. To determine the fractional delay, the 65536 possible fractional delay values must be mapped onto the range of values for the ramp D/A converter that yield a 50 ns linear ramp. Given the D/A calibration parameters, it is straighforward to linearly scale the fractional time representation onto the D/A range, thereby giving the desired D/A converter code. The microcontroller code implementing this improved Acq2 sequencing method is included below.

```
...

timeinf start, cur, end;
UINT32 incr, reclen;
UINT16 dacmin, dacmax, dacval;
UINT8 step_dly, strobe_dly;

// get the step delay
step_dly = params[IDX_STEP_DLY];

// obtain the ramp calibration parameters
dacmin = (((UINT16)params[IDX_CALSTART_UPPER]) << 8) | ((UINT16)params[IDX_CALSTART_LOWER]);
dacmax = (((UINT16)params[IDX_CALEND_UPPER]) << 8) | ((UINT16)params[IDX_CALEND_LOWER]);

// obtain the start time, end time, and record length
start.time_b.b3 = params[IDX_TMSTART_B3];
start.time_b.b2 = params[IDX_TMSTART_B2];
start.time_b.b1 = params[IDX_TMSTART_B1];
```

```
start.time_b.b0 = params[IDX_TMSTART_B0];

end.time_b.b3 = params[IDX_TMEND_B3];
end.time_b.b2 = params[IDX_TMEND_B2];
end.time_b.b1 = params[IDX_TMEND_B1];
end.time_b.b0 = params[IDX_TMEND_B0];

// record length: reclen = 2^(params[IDX_RECLENPWR])
if (params[IDX_RECLENPWR] < 6)
  params[IDX_RECLENPWR] = 6;
if (params[IDX_RECLENPWR] > 10)
  params[IDX_RECLENPWR] = 10;

reclen = m2pow(params[IDX_RECLENPWR]);

// enable/disable the step generators, set the polarities
step_enable( params[IDX_CHAN0_EN], params[IDX_CHAN1_EN] );
step_polarity( params[IDX_CHAN0_POL], params[IDX_CHAN1_POL] );

// set the sampler reverse bias, set the ADC offset
WRITE_DAC1( dacvals[1] );
WRITE_DAC2( dacvals[2] );

// calculate the time increment
incr = (end.time - start.time)/reclen;
// set the start time
cur.time = start.time;

// run acquisition sequence for the specified record length
for (i=0;i<reclen;i++)
{
  // calculate the integer strobe delay
  strobe_dly = 254 - (UINT8)cur.time_s.int_val;
  // calculate the fractional delay by mapping the fractional value to the d/a range
  dacval = dacmin + ((dacmax-dacmin)*cur.time_s.frac_val)/0xFFFF;
  // set the ramp level (fractional delay, DAC0)
  WRITE_DAC0( dacval );

  // write the timer and start the point acquisition
  xspi_write_delays( step_dly, strobe_dly );

  // Polling the CONV_START pin that is driven by the xilinx timebase
  while(!READ_CONVS);
  // perform the conversion
  ADCCON = 0xA3; // software conv., single-ended, conv. enabled
  while (!ADCSTA); // wait for end of conversion
  // extract the ADC result
  UINT16 result = (UINT16) ((ADCDAT & 0x0FFF0000) >> 16);

  // copy the result into the waveform data
  rawdata[2*i] = (UINT8) ((result & 0xFF00) >> 8);
  rawdata[2*i+1] = (UINT8) (result & 0x00FF);

  // increment to the next time value
  cur.time += incr;
}

...
```

This sequencing algorithm is more intuitive, and it is much more easily programmed and controlled by the host application. It maps a very high resolution virtual time scale onto the capabilities of the underlying acquisition hardware, and provides a clearer abstraction between the user interface and the hardware.

i	Parameter ID	Purpose	Applies To
0	IDX_COMMAND	Command byte. (255:Dbg 254:Acq1 253:Acq2)	N/A
1	IDX_STEP_DLY	Step integer delay counter initial value	Dbg
2	IDX_STROBE_DLY	Strobe integer delay counter initial value	Dbg, Acq1
3	IDX_CHAN0_EN	Channel 0 Step /Enable Select	*all*
4	IDX_CHAN1_EN	Channel 1 Step /Enable Select	*all*
5	IDX_CHAN0_POL	Channel 0 Step Polarity Select	*all*
6	IDX_CHAN1_POL	Channel 1 Step Polarity Select	*all*
7	IDX_DAC0_UPPER	Ramp Compare Value (MSB)	Dbg
8	IDX_DAC0_LOWER	Ramp Compare Value (LSB)	Dbg
9	IDX_DAC1_UPPER	Sampler Reverse Bias Adjust (MSB)	*all*
10	IDX_DAC1_LOWER	Sampler Reverse Bias Adjust (LSB)	*all*
11	IDX_DAC2_UPPER	A/D Offset Adjust (MSB)	*all*
12	IDX_DAC2_LOWER	A/D Offset Adjust (LSB)	*all*
13	IDX_INCREMENT	Time Increment (int. multiple of min. time spacing)	Acq1
14	IDX_START_DLY	Start Delay (int. multiple of min. time spacing)	Acq1
15	IDX_FREERUN	Enable/disable freerun (1/0)	Dbg
16	IDX_CALSTART_UPPER	Ramp D/A Calibration Start Value (MSB)	Acq1, Acq2
17	IDX_CALSTART_LOWER	Ramp D/A Calibration Start Value (LSB)	Acq1, Acq2
18	IDX_CALEND_UPPER	Ramp D/A Calibration End Value (MSB)	Acq1, Acq2
19	IDX_CALEND_LOWER	Ramp D/A Calibration End Value (LSB)	Acq1, Acq2
20	IDX_TMSTART_B3	32-bit Start Time Value (MSB)	Acq2
21	IDX_TMSTART_B2	32-bit Start Time Value	Acq2
22	IDX_TMSTART_B1	32-bit Start Time Value	Acq2
23	IDX_TMSTART_B0	32-bit Start Time Value (LSB)	Acq2
24	IDX_TMEND_B3	32-bit End Time Value (MSB)	Acq2
25	IDX_TMEND_B2	32-bit End Time Value	Acq2
26	IDX_TMEND_B1	32-bit End Time Value	Acq2
27	IDX_TMEND_B0	32-bit End Time Value (LSB)	Acq2
28	IDX_RECLEN	Record length = $2^x \vert 6 \leq x \leq 10$	Acq2

Table 3: Microcontroller Parameter Vector Description

8 PC/Microcontroller Communication

8.1 Subsystem Overview

A communication protocol was designed and implemented in C/C++ on both the PC and ADuC7026 microcontroller, and the necessary hardware adaptor was built to allow the device to be connected to the standard DB-25 parallel port. With the host operating in standard parallel port (SPP) mode, data transfer rates of 83.3 kbytes/sec were achieved using full software handshaking for concurrency.

8.2 PC Parallel Port

The PC parallel port operating in standard mode (SPP) was used. The parallel port consists of three registers/ports. The 8-bit data port was originally designed by IBM as output only, but nearly all relatively recent implementations allow the data port to operate in both directions. The lower nibble of the control port can also be used as input or output, and the upper five bits of the status port are permanently configured as inputs. Some of the bits in the control and status ports are inverted at the connector, requiring appropriate adjustments in the software to achieve the desired behavior. The upper nibble of the control port is reserved. Writing a 1 to C5 changes the data port to high impedance state, thereby allowing the port to read the value placed on the pins.

It was determined experimentally that the port in the computer used had 2.2 kΩ internal pullup resistors. The pulled up voltage on the data and status ports was measured to be approximately 4.3 V. A diagram of the parallel port connecter is shown in Figure 15.

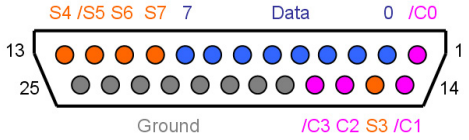


Figure 15: PC Parallel Port Connector

8.3 Interface Resource Allocation

The communication schematic is shown Figure 16.

The 8-bit wide bi-directional data port sends or receives data. The request (RQ) line is used to initiate a transaction with the microcontroller, and the clock (CLK) line is used to synchronize each byte as it is transferred. The ready (RDY) and acknowledge (ACK) lines provide status indicators as to the state of the current transaction.

8.4 Low-Level Protocol Definition

The PC host interface is defined as the master, and the ADuC7026 as the slave. The master initiates all data transfers, and controls the transfer sequence with the clock signal. In the current implementation, the transfer length is fixed in the code and must be the same on both the master and slave. The basic protocol outlined below can easily be upgraded later to allow for variable length transfers. The direction of the transfer is specified by the state of the CK line at the time

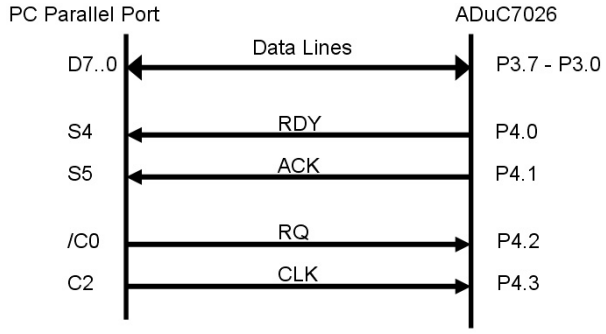


Figure 16: Parallel Communication Hardware Resource Allocation

that the request (RQ) line is raised to initiate a transfer. If the CK=1, the transfer is from the device to the host, otherwise from the host to the device. The basic transfer protocol is outlined in detail below.

8.4.1 Host-to-Device Transfer

1. Host enables outputs on the data port, CK line is low to indicate direction of transfer.
2. Host raises the RQ line.
3. Host waits for RDY line to be asserted by the device.
4. Host lowers the RQ line.
5. Data transfer sequence commences
   (a) Host writes data to the data port.
   (b) Host raises the CK line to signal that the data is ready.
   (c) Host waits for the ACK signal to be asserted by the slave, indicating that the data was read.
   (d) Host lowers the CK signal.
   (e) Host waits for ACK signal to go low, signaling that the byte transaction was completed, and the slave is ready for the next byte. (repeat a-e as necessary)
6. Host puts the data port back to INPUT mode for reduce the chance of both the parallel port and microcontroller writing at the same time.
7. Host clears both RQ and CLK lines to return to idle state.

8.4.2 Device-to-Host Transfer

1. Host raises the CK line to indicate the direction of transfer.
2. Host raises the RQ line. (the data port is already in INPUT mode, default state)

3. Host waits for RDY line to be asserted by the device.
4. Host lowers the RQ line.
5. Data transfer sequence commences
   (a) Host lowers the CK line to signal that it awaits the data byte.
   (b) Host waits for the ACK signal to go high, indicating that the data byte is ready for reading.
   (c) Host reads the data.
   (d) Host raises the clock to indicate that the data was read.
   (e) Host waits for the ACK signal to go low, signaling that the byte transaction was completed, and the slave is ready to transfer the next byte. (repeat a-e as necessary)
6. Host clears both RQ and CLK lines to return to idle state.

Due to the full software handshaking in place, the protocol as outlined is not time sensitive. To prevent either the host or device from locking up in the case of excessively long response times (or no response/error), iteration limitations are used in the wait cycles to impose a timeout on the transfer sequence. In this way, the robustness of the protocol is ensured. No error checking provisions are made in the basic transfer sequence. For most applications, this is probably acceptable due to the rarity of bit errors, especially considering that the data being transferred in the intended target application is relatively non-critical.

The protocol from the device perspective is analogous to the host side, and can be readily arrived at by considering what sequence is required by the host for a successful data transfer.

8.5 Hardware Considerations

Although the ADuC7026 GPIO pins are 5 V tolerant, series 180 Ω resistors are placed between the parallel port and the microcontroller to help protect against high currents flowing in the event that outputs on both sides are enabled at the same time. The resistors are used on the control port and the data port, but not on the status port since it cannot ever be driven by the parallel port.

8.6 Driver Software

The PC host software was implemented using the DriverLinx DlPortIO generic port driver for Windows 2000/XP. The driver is freely downloadable from the `http://www.driverlinx.com` website, and provides direct port access under the protected-mode operating system. A basic communication protocol test interface was developed using the wxWidgets GUI toolkit from `http://www.wxwidgets.org` to exercise a simple 9 byte read/write sequence with the microcontroller and to test the bidirectional functionality of the parallel port. This software is independent of the TDR control software used elsewhere in the project.

The ADuC7026 is programmed in plain C with the Keil μVision3 development studio using the GCC-ARM7 compiler. These tools were provided on the microcontroller's development kit CDROM.

8.7 Testing and Performance Characterization

Correct operation of the bidirectional transfer was readily verified using the Commhost interface application, and further documented using a oscilloscope. A write operation is shown below in Figure 17.

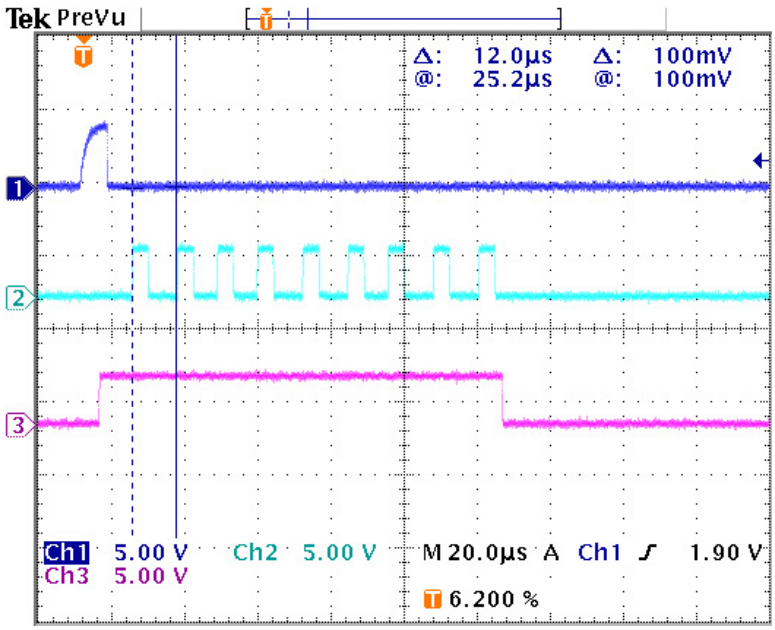


Figure 17: Parallel Port 9 Byte Write Sequence

Using the oscilloscope cursors shown in Figure 17, the data transfer rate is readily determined.

$$12\ \mu\text{s/byte} \longrightarrow \frac{1\ \text{byte}}{12 \cdot 10^{-6}\ \text{s}} = 83.33\ \text{kbyte/s}$$

This is a reasonably fast transfer rate of about 0.5 MBit/sec, considering that the all of the synchronization is done in software. It is indeed much faster than the RS-232 option for bulk data transfer, and should not pose a barrier in achieving the waveform transfer throughput desired in the TDR project.

9 PC Host Control Software

9.1 Organization

The PC-based TDR control software is based on two main modules: the graphical user interface (GUI), and the communication module used to transfer data to and from the acquisition hardware. The GUI is written using the National Instruments CVI 6.0 development tools, allowing for a rapid development cycle using the standard C language. The communication module is also written in C, and is based on a freely available Windows parallel port driver. Two separate control interfaces exist for the TDR, discussed next.

9.2 Debugging Interface

The debugging interface program allows the operator to exercise each of the individual subcomponents of the acquisition hardware for testing and verification. It also provides rudimentary mechanisms for simple timebase sequencing, calibration testing, and can acquire waveforms. A screenshot of the front panel is included below in Figure 18.

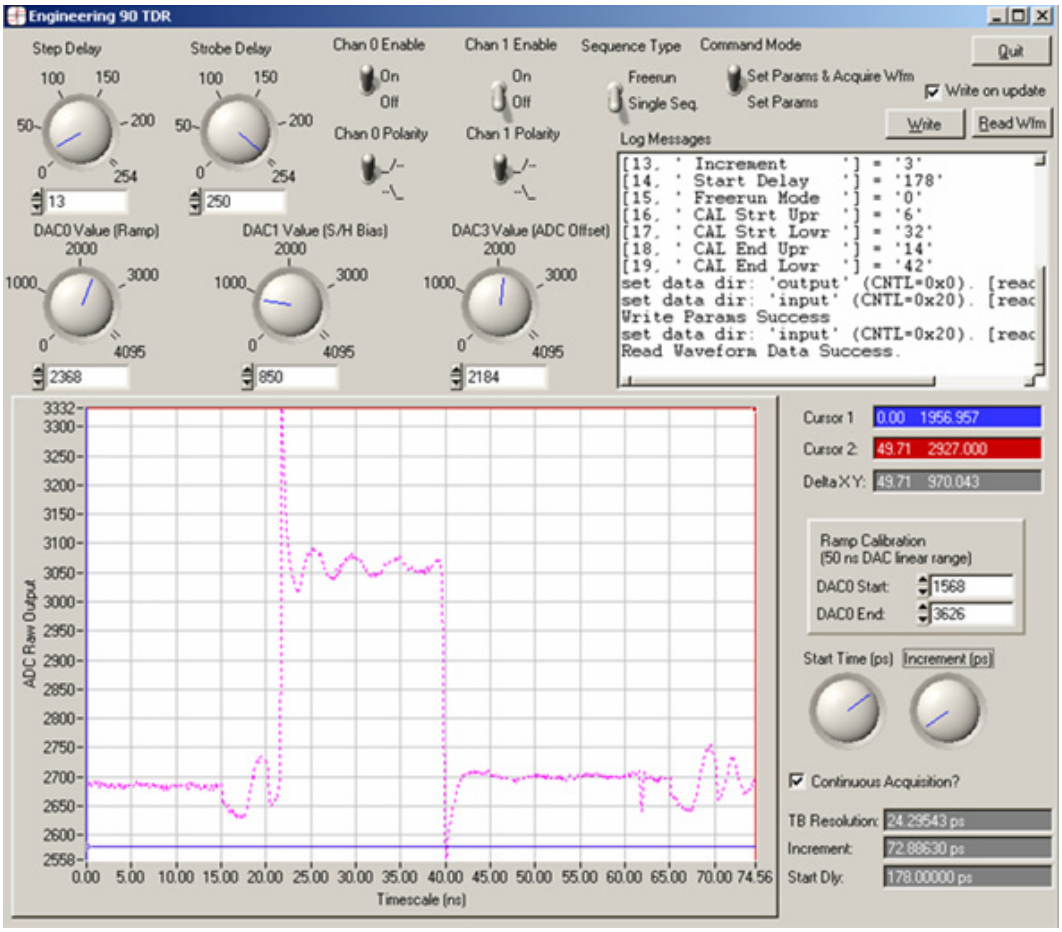


Figure 18: Debugging Interface Program

The controls at the top allow the user to set the raw 8-bit delay codes in the Xilinx timebase counter, as well as the 12-bit D/A converter values for the ramp comparator level, the sampling bridge reverse bias voltage, and the DC offset of the sampler output voltage. Enable/disable and polarity select controls for the step generator are also available. The *Sequence Type* switch changes whether the microcontroller repeated reprograms the Xilinx timer, thereby periodically generating the step and sample signals. This feature is useful for retriggering an oscilloscope (triggered on the *pretrigger* output) to probe various timing signals for verification. Changing the step delay, for example, would immediately show up on the oscilloscope screen. In *Single Sequence* mode, the

Xilinx timer is programmed only once, thus generating only one step/sample sequence. The freerun mode is especially essential for properly triggering high bandwidth sampling scopes during testing.

The *Command Mode* option sets whether a rudimentary waveform acquisition should be run after the parameters specified in the interface have been applied. In the set and acquire mode, the raw waveform data can be read back and displayed with the *Read Wfm* button. However, if the *Continuous Acquisition* checkbox is selected, a new waveform acquisition is initiated every second by a timer, and the waveform display graph is updated accordingly. The *Start Time* and *Increment* controls specify the timebase parameters for the waveform, setting the fractional start delay and the time spacing between individual data points, respectivel. The integer start delay count is still set by the *Strobe* value dial, although in this mode it is the two's complement representation $(255 - val)$. For basic timebase calibration, the lower and upper bounds of the ramp level D/A converter are specified. The D/A values specified should correspond to a 50 ns wide linear section of the ramp, and must be measured manually using an oscilloscope with the TDR in *freerun* mode.

The *Write* button simply resends the update parameters to the TDR hardware. The *Write on update* checkbox sets whether the parameters are automatically resent when a parameter value is changed by the user.

Due to the very fine grain control of all aspects of the TDR hardware, it can be quite challenging to find reasonable parameter values for acquiring a simple waveform with which to start debugging. As such, the parameter values shown in Figure 18 are a good starting point from which to experiment with the TDR functionality.

9.3 Streamlined TDR Interface

Although the debugging program provides direct access to the TDR acquisition hardware, it is too cumbersome for making actual TDR measurements. A streamlined interface is thus provided for usability once the underlying functionality has been verified. A screenshot is shown in Figure 19.

With this interface, it is simple to specify the acquisition start time and the width of the viewing window in any of the available horizontal scales (time, feet, or meters). If a distance scale is selected, the expected cable dielectric constant can be entered to facilitate the time-to-distance conversion. In the current setup, the timebase window can be controlled from 0 to 1500 ns, which corresponds to a distance of approximately 152 m in a standard 50 Ω SMA cable.

On the right of the waveform viewer are the waveform acquistion controls. If the *Acquire Continuously* checkbox is selected, the waveform is reacquired every 0.2 seconds. The record length can also be changed, with 64 points being the shortest and 1024 points the longest, stepping in powers of 2.

Directly below the acquire controls are the step generator parameters. Each channel can be independently enabled or disabled, and the polarity of each can be set as well. The step delay knob specifies the time at which the stimulus is generated, although the time indicator is probably only an approximation of the actual step time in the current implementation. This indicator is thus useful for generally positioning the step within a desirable viewing window, and the actual position can be readily measured using the graph cursors.

Direct access to the timebase calibration parameters is given below the step generator controls. Upon application startup, the initial values provide a reasonably accurate 50 ns ramp range (arrived at from manual calibration), so it should not be necessary to change them for most measurements.

An application status indicator and a *Quit* button round out the streamlined TDR interface.

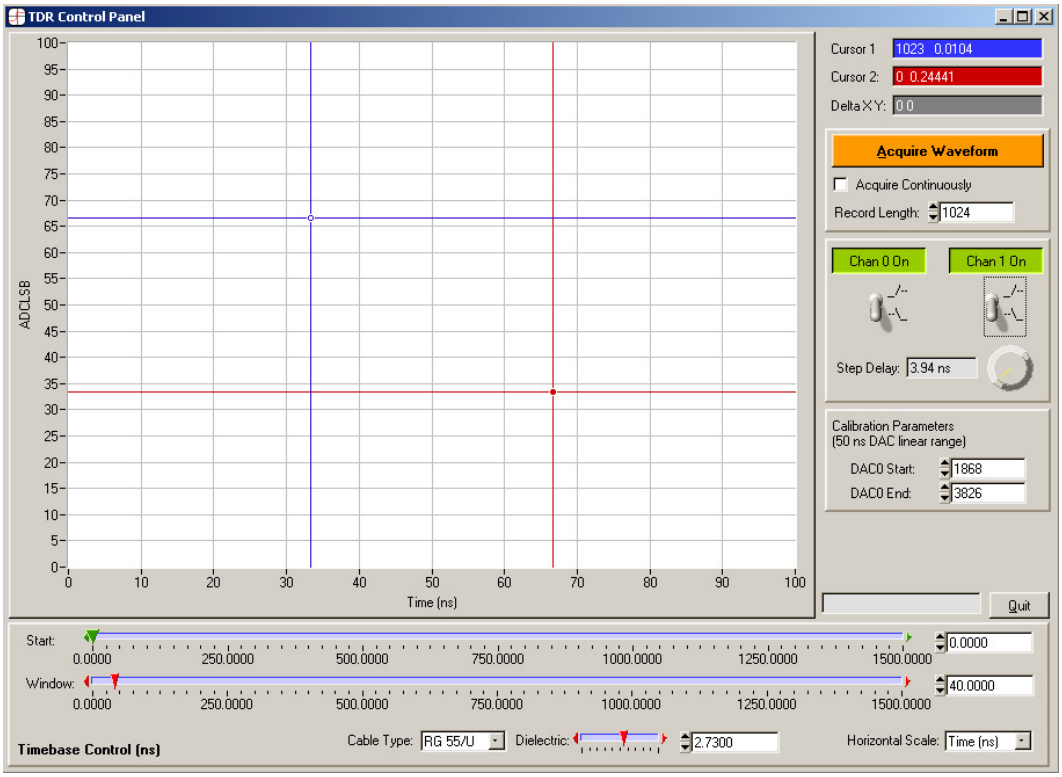


Figure 19: Streamlined TDR Control Interface

9.4 CVI 6.0 Code Overview

A listing of the source files in the streamlined TDR interface application is now presented with brief descriptions.

Source Code Files

1. `tdrui.c` - The source code file containing all of the interface callback functions, the application entry point *main()*, and the TDR control parameter manipulation code.
2. `tdrui.h` - The user interface header file generated automatically by the CVI 6.0 development environment.
3. `commadi.h` - Low-level parallel port communication function definitions.
4. `commadi.c` - Implementation of the low-level parallel port communcation functions.
5. `dlportio.h` - Header file for the `dlportio.lib` parallel port driver library.

Other Application Files

1. `tdrui.uir` - Binary user interface definition file generated by the CVI 6.0 user interface editor.
2. `dlportio.lib` - Precompiled parallel port driver library.

10 Printed Circuit Board and Construction

10.1 Layout Considerations

A four layer circuit board was designed to allow for ample flexibility in providing an internal high frequency ground plane while properly routing the supply lines and the signal interconnections. The digital components were placed at a distance from the sensitive analog sampling and timing circuits, and symmetrical layout techniques traces were used to maintain proper sychronization in sensitive sections. A top view of the layout is shown in Figure 20.

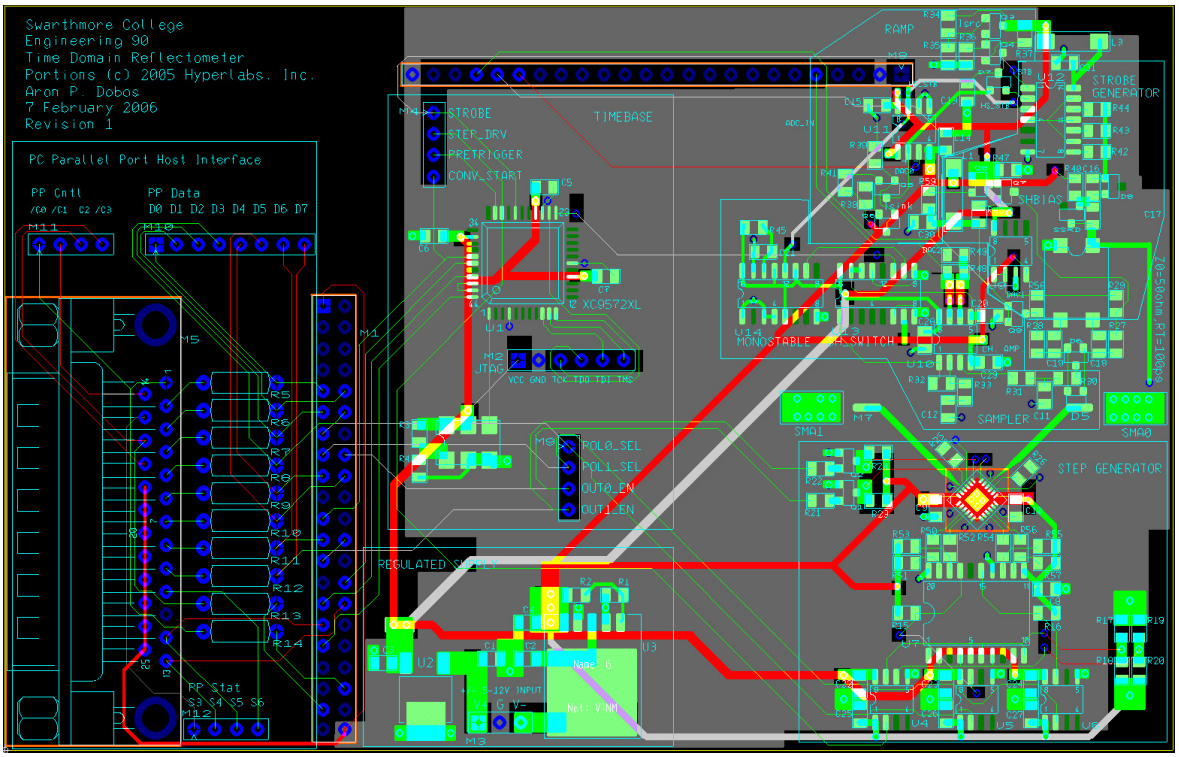


Figure 20: 4 Layer PCB Layout

10.2 Construction Techniques

Surface mount technology (SMT) components were used throughout the TDR implementation to achieve the desired performance. A manual reflow soldering technique was developed based on

the one used at Maxim Integrated Products in their evaluation board assembly laboratory. Using a shot-meter type dispenser, solder paste was applied in controlled amounts to the pads on the circuit board, and the parts were manually positioned using fine tweezers. The circuit board was then placed on a low-profile hot plate, allowing heat to pass up through the circuit board and melt the solder paste, thereby mounting the components. Any shorts on fine pitch parts were removed using a solder removal wick and a microtip soldering iron. All of these procedures were conducted under an illuminated stereo dissection microscope to achieve the necessary precision.

A picture of the completed TDR circuit board is shown in Figure 21. Several modifications to the original circuit are visible that were necessary to obtain correct functionality. The TDR circuit board attaches via pin headers underneath the board to the standard ADuC7026 microcontroller development board.

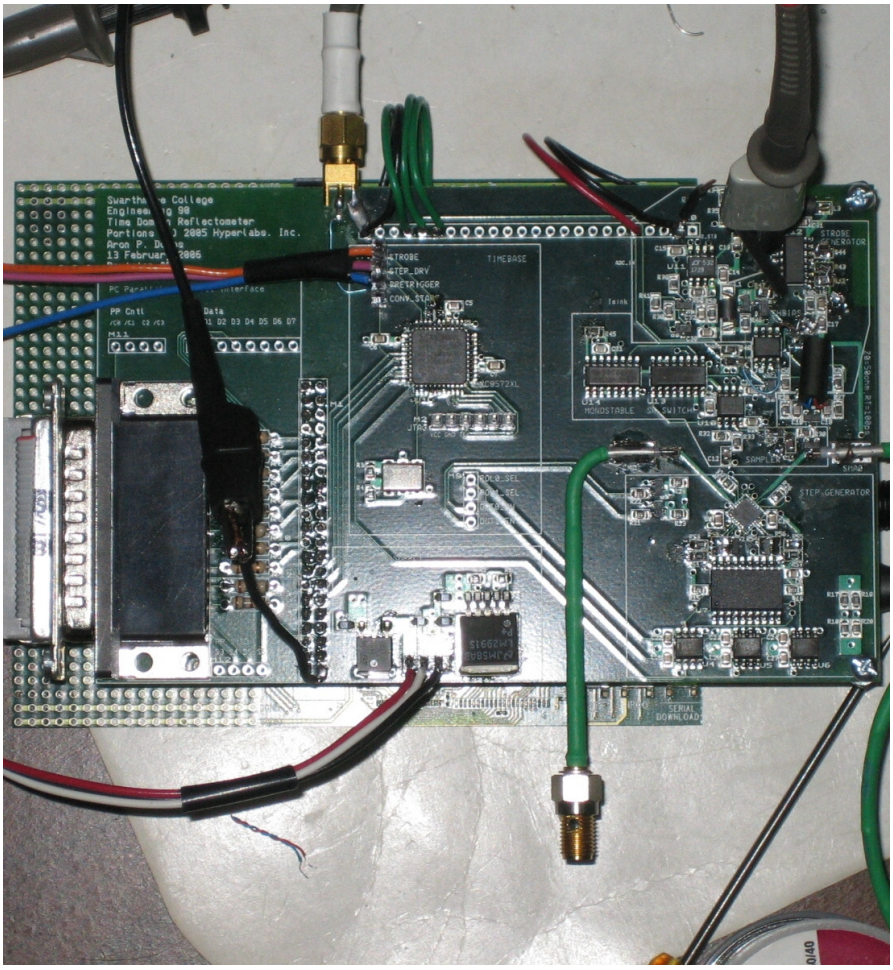


Figure 21: Completed TDR Circuit Board

11 Testing and Results

11.1 Procedures and Instrumentation

Measurements of the timebase circuit were made using 100 and 500 MHz real-time oscilloscopes with the TDR hardware in *freerun* mode using the pretrigger output to trigger the scope. Using the 10x attenuated pretrigger output connected to a 20 GHz sampling oscilloscope, proper measurements of the differential strobe, step stimulus, and timebase jitter were possible. Custom low-inductance ground probes were used to obtain clean signal measurements of the timebase ramp and charge amplifier response in the sampler.

Evidence of the proper functioning of the various parts of the TDR instrument have already been presented. A variety of TDR measurements are presented next to give an estimation of the capabilities of the completed instrument.

11.2 TDR Measurements

Three test cases were run with different cables using the streamlined TDR interface to control the instrumentation.

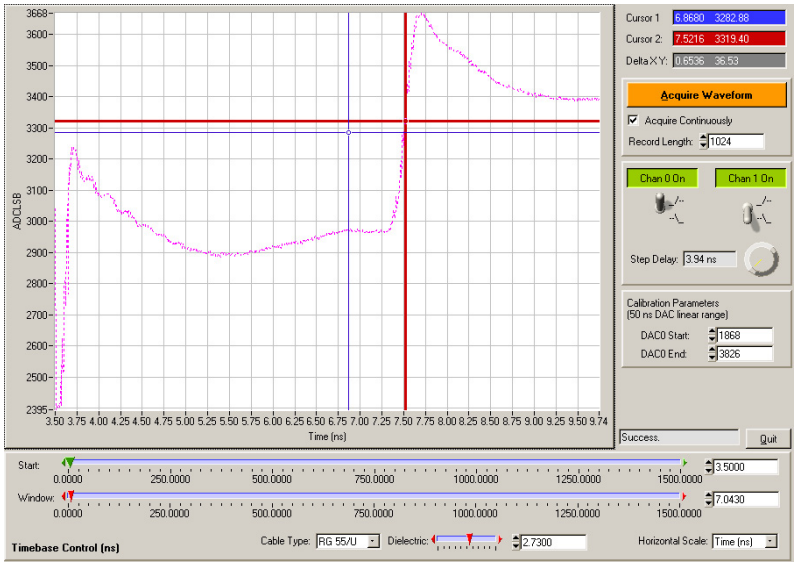
The first test run (Figure 22) is without any test cable connected. The width of the half-way step corresponds to the length of the 50 Ω semirigid cable between the step generator/sampler interface to the SMA test connector. Normally, this known length of cable would be used for distance calibration. The TDR waveforms clearly indicate the proper shapes for open, short, and matched termination conditions.

The second test run (Figure 23) is with a 155 cm SMA cable attached. The timescale is unchanged from the first test, and thus the round trip time from the end of the semirigid cable to the end of the test cable can be readily measured using the graph cursors. Again, the three waveforms are correct for the open, short, and matched termination conditions.

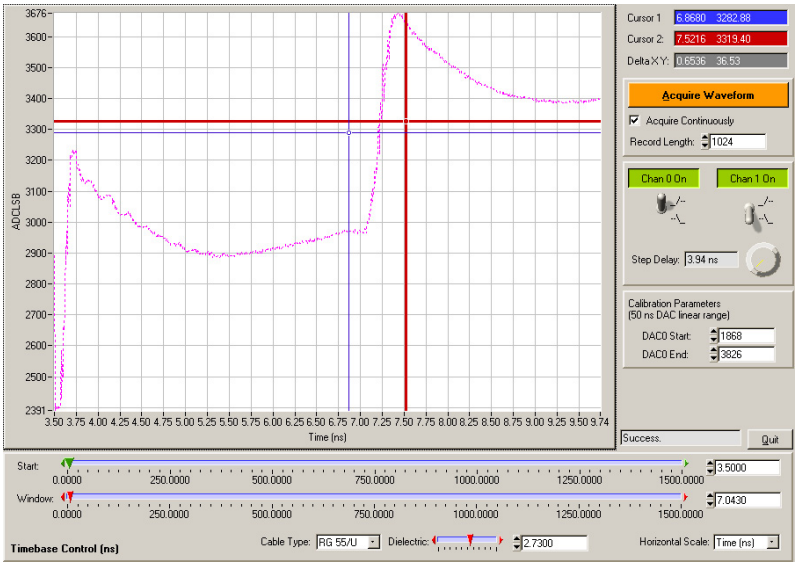
For the third test, the timescale was changed to show a very short window. First, a waveform was acquired without any test cable connected, and the blue cursor was positioned to the time at the end of semirigid connector. Next, an 8 cm semirigid open circuit test stub was connected, and thus the difference between the red and blue cursor in Figure 24a indicates that the round trip time along this test cable is about 650 ps. In Figure 24b, the 8 cm stub was cut in half, and it is apparent as the step occurs essentially halfway between the red and blue cursors, as expected. Tt is shown that the TDR instrument is capable of detecting discontinuities with at least 4 cm resolution, if not even finer.

Figure 22: Waveforms For No Test Cable (Open, Short, 50 Ω Termination)

Figure 23: Waveforms For 155 cm Test Cable (Open, Short, 50 Ω Termination)



(a)

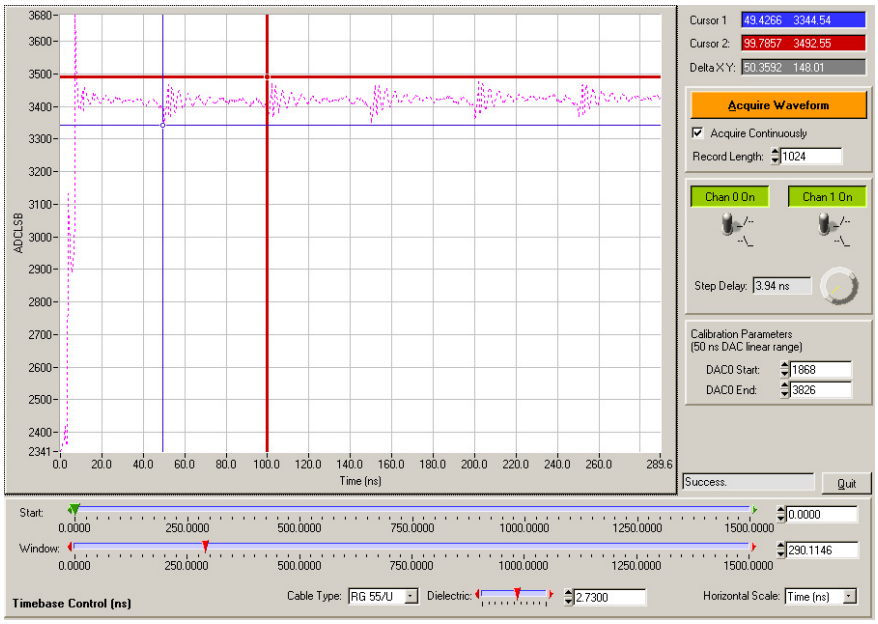


(b)

Figure 24: Waveforms For 8 and 4 cm Test Cable (Open)

11.3 Clock Feedthrough and SNR

By extending the viewing window, it is clear in Figure 11.3 that there is significant clock feedthrough in the sampled vertical scale. Positioning the cursors appropriately reveals that the noise has a period of almost exactly 50 ns, meaning that it most likely results from the 20 MHz clock in the digital timebase. It remains unclear how this clock noise can be properly eradicated from the vertical scale.



A simple signal-to-noise ratio (SNR) can be calculated for the sampler. The amplitude of the clock noise is 148 A/D converter values, and the amplitude (without overshoot) of the signal waveform is about $3450 - 2340 = 1110$. Thus the SNR is calculated to be

$$\text{SNR} = 20 \cdot \log \frac{1100}{148} = 17.42 \text{ dB}.$$

This is not a very good figure, and it is clear that the sampler circuit and circuit board layout are both worthy of improvement efforts.

12 Future Work

There are many aspects of the TDR instrument as presented that remain worthy of improvement. The most beneficial improvement would be to increase the signal-to-noise ratio of the sampler circuit and to reduce the clock-feedthrough that injects appreciable periodic noise into the waveform. A method for calibration would also be quite beneficial from the end user's point of view. These and more potential improvements are now pursued in the following sections.

12.1 Timebase Resolution and Jitter

Many improvements could be made to the timebase circuit to improve resolution. A very simple modification that would essentially double the minimum time spacing between data points would be to use two of the 12-bit D/A converters in the ADuC7026 to set ramp compare level. Using a current sharing resistive network, the D/A converters would be programmed in software to output either the same voltage V, or one V and the other V+ΔV. As a result of the voltage division at the output resistors, the effective voltage seen at the comparator input would be that of the lower D/A value plus half of the normal D/A ΔV. With some additional software complexity, a 13-bit converter could be mimicked simply by utilizing the last unused D/A converter in the microcontroller.

Timebase jitter is mostly a function of variation in the analog ramp circuit. If the ramp time were shortened, the jitter would be reduced, and the time resolution given the same D/A precision would be increased. This is easily accomplished by increasing the clock frequency that the timebase runs on, and also increasing the ramp generator current source or decreasing the ramp capacitor value to appropriately increase the slope of the ramp to maximize the available D/A voltage range. The Xilinx FPGA can be operated upwards of 150 MHz, giving generous room for timebase improvements, although at the expense of increased power consumption.

12.2 Sampler Improvements

The sampler circuit is probably the most sensitive and intricate component in the TDR instrument. The open loop sampler topology pursued in this project gives good rise times without oversampling, but at the expense of linearity that could be provided by negative feedback. Using sub 1 pF reverse bias capacitance diodes at the sampling bridge could improve the signal-to-noise ratio by reducing capacitative coupling to other parts of the circuit, and would also reduce the performance degredation of the step stimulus. Implementing a digital feedback mechanism to set the sampling gate reverse bias could also help improve linearity as discussed previously.

Entirely different circuit topologies could yield a sampling circuit with comparable performance. Instead of a sample-and-hold type circuit, a track-and-hold sampler could be implemented by buffering the test connecter using new gigahertz bandwidth opamps (e.g. THS4303 with 1.8 GHz GBWP and 5500 V/μs slew rate) from Texas Instruments. Such an opamp could drive a small hold capacitor through a bipolar current-switched diode bridge controlled by the timebase. In this way, the test signal would be directly transferred to the hold capacitor during the track phase, avoiding the linearity problems introduced by the sampling gate biasing and charge amplifier impulse response gating method. The bandwidth of a buffered track-and-hold sampler might be severely limited by parasitics and opamp characteristics, and it is still not clear how to go about disconnecting the hold capacitor from the opamp quickly enough to achieve the desired sampler rise time.

12.3 Layout

Potential improvements in the PCB layout could decrease clock feedthrough and other couplings between circuits. Heavier use of star topologies for power supply routing could reduce crosstalk through the supply rails by providing each component or subsection its own direct path back to the main supply source. Introducing small series inductors with the power supply lines to the components could also help reduce coupling between sections by isolating supply current transients. Separating the analog and digital ground planes, and perhaps even the low frequency and high frequency analog ground planes, would most likely result in a dramatic reduction in clock noise in

the sampling circuit. Reducing the length the traces from the digital timebase to the analog ramp and strobe generator, or at least resistively decoupling them, would also result in cleaner signal transmission between parts of the circuit.

12.4 Power Consumption

The idle power consumption of the TDR circuit could be greatly reduced by introducing circuitry to physically turn off the sampler, timebase, and step generator circuits when they are not in use. Two logic controlled MOSFETs and the appropriate power network wiring would achieve this. The microcontroller would then only enable the TDR circuit during a waveform acquisition. Power could also be reduced by using a simpler (perhaps single channel) step generator circuit to reduce the number of power hungry termination networks, and by even reducing the clock speed of the microcontroller.

12.5 Hardware Calibration

Hardware calibration is a feature sorely missing from the current TDR implementation. Currently, timebase calibration is achieved by measuring the linear range of the ramp using an oscilloscope, and then providing the appropriate D/A converter codes that map to this range. In terms of the vertical scale, there is no calibration whatsoever.

One way to implement automatic ramp calibration would be to position the step stimulus at various places within the fractional delay range at a fixed integer delay. By measuring the resulting waveform and extracting the location of the step, the ramp D/A converter codes could be adjusted programmatically so that the step would appear at the expected position. If the step delay counter were run off a much higher frequency clock than the integer time delay counter, relatively fine grain positioning of the step could be achieved. Given the possibility of positioning the step at various places along the ramp, the linearity of the ramp could be determined from the resulting waveforms, and the best D/A converter calibration could be (somehow) arrived at.

Providing a vertical calibration mechanism for the open loop charge amplifier-based sampler circuit is not simple. One way would be to design a circuit that presents a precisely known fixed DC voltage at the test connector, and by measuring the raw output of the sampler, a mapping table could be generated to translate the raw A/D converter codes to test connecter voltages, at a fixed sampler reverse bias point. As simple as this might sound, it in fact is not quite feasible to introduce another circuit on the high frequency microstrip line to apply such a DC voltage, since again the parasitic capacitances would even further degrade the stimulus rise/fall time. It is not clear how to provide an efficient calibration if the sampler reverse bias point is changing due to either analog or digital feedback mechanisms. It is conceivable to create a 2-dimensional mapping table between the raw A/D converter result and the current sampler reverse bias position to the applied DC calibration voltage, although such an approach might not give very good results unless non-linear interpolation algorithms are used to map the space between calibrated points. It might be good to keep in mind that such a mechanism, regardless of how clunky, would still be a fantastic improvement over the current implementation, provided that the problems posed by the parasitics can be overcome.

12.6 Integration and Construction

The current TDR implementation is circuit board that plugs into the standard Analog Devices Microconverter ADuC7026 development board. As such, it requires separate power supplies, separate clocks, and overall presents itself as a clunky package for actual TDR measurements. In practice, it would be ideal to integrate the ADuC7026 microcontroller and all of its accompanying circuitry onto the same board with the TDR circuits, and use the same power supply and clock to reduce period interference noise from multiple clocks in the circuit. If the circuit were used in the field, a proper enclosure would clearly be required as well.

13 Environmental Impact and Sustainability

The environmental impact is probably the same as for any piece of small electronics, meaning that recycling is expensive and involved due to the inherent difficulties in separating out the numerous different metals and plastics that are so tightly integrated in electronics. Since TDR instruments do not tend to become obsolete as quickly as personal computers, nor are they produced in huge quantities, it is unlikely that they will be landfilled in any significant volume. Provided that appropriate *e*-waste recycling facilites exist and are appropriately funded, a broken or otherwise unused TDR instrument could probably be harvested for its metals and perhaps even plastics.

14 Conclusions

The design and assessment of a high performance Time Domain Reflectometer instrument for 50 Ω cable characterization purposes was presented. While many significant areas of improvement remain, the instrument is indeed capable of discerning discontinuities spaced 4 cm apart with its sub-200 ps sampler and step generater rise times. The various host interface applications provide access to the TDR hardware from a very detailed debugging level to a high level TDR measurement interface designed for an end user.

References

[1] Axelson, Jan. *Parallel Port Complete*. Madison, WI: Lakeview Research, 2000.

[2] Boni, Andrea, Matteo Parenti, and Lorenzo Agnetti. "*A 1-GS/s 2.7-V Track-and-Hold Amplifier with 10-b Resolution at Nyquist in SiGe BiCMOS*." ESSCIRC (2002).

[3] Brown, Stephen, and Zvonko Vranesic. *Fundamentals of Digital Logic with VHDL Design*. 2nd ed. New York: McGraw Hill, 2005.

[4] Choi, Michael, and Asad A. Abidi. "*A 6-b 1.3-Gsample/s A/D Converter in 0.35-μm CMOS*." IEEE Journal of Solid-State Circuits 36 (2001).

[5] Dobos, Aron P. "*Time Domain Reflectometry*." E75 Project Report. 3 Apr. 2006 <http://www.engin.swarthmore.edu/∼adobos1/e75/>

[6] Dobos, Aron P. "*Parallel Port Communication with the ADuC7026 Microcontroller*." Engineering 15 Project. 3 Apr. 2006 <http://www.engin.swarthmore.edu/∼adobos1/e15/project/>.

[7] Grebene, Alan. *Bipolar and MOS Analog Integrated Circuit Design*. New York: John Wiley & Sons, Inc., 1984.

[8] Gupta, K C., Ramesh Garg, Inder Bahl, and Prakash Bhartia. *Microstrip Lines and Slotlines*. 2nd ed. Boston: Artech House, 1996.

[9] Hager, N E., and R C. Domszy. "*Monitoring of Cement Hydration by Broadband Time-Domain-Reflectometry Dielectric Spectroscopy*." Journal of Applied Physics 96 (2004). 3 Apr. 2006 <http://www.msi-sensing.com/pubs/msi_cement_jap_2004.pdf>

[10] Harvey, Ken. "*Frequency Domain Reflectometry Vs. Time Domain Reflectometry*." MRT. 3 Apr. 2006 <http://mrtmag.com/mag/radio_frequency_domain_reflectometry_2/>.

[11] Horowitz, Paul, and Winfield Hill. *The Art of Electronics*. 2nd ed. New York: Cambridge UP, 1989.

[12] "*TDR Concrete Cure Monitoring*." Material Sensing and Instrumentation. 3 Apr. 2006 <http://www.msi-sensing.com/tdr_concrete.htm>.

[13] Razavi, Behzad. "*A 200-MHz 15-mW BiCMOS Sample-and-Hold Amplifier with 3-V Supply*." IEEE Journal of Solid-State Circuits 30 (1995).

[14] Razavi, Behzad. "*Design of a 100-MHz 10-mW 3-V Sample-and-Hold Amplifier in Digital Bipolar Technology*." IEEE Journal of Solid-State Circuits 30 (1995).

[15] Razavi, Behzad. "*Principles of Data Conversion System Design*. New York: John Wiley & Sons, Inc., 1995.

[16] Sedra, Adel S., and Kenneth C. Smith. *Microelectronic Circuits*. 5th ed. New York: Oxford UP, 2004.

[17] Shen, Liang C., and Jin A. Kong. *Applied Electromagnetism*. Monterey, California: Brooks/Cole Engineering Division, 1983.

[18] Vladimirescu, Andrei. *The SPICE Book*. New York: John Wiley and Sons, 1994.