# System Overview

Our application leverages the power and flexibility of the myRIO’s ARM® Cortex™-A9 real-time processing and Xilinx FPGA customizable I/O to create a device that provides the functionality of a four-track recorder with some unique add-ons. With our application, we are able to:

* Perform real-time DSP by leveraging the FPGA for our audio acquisition, processing and playback.
* Quickly stream binary data to and from disc on the real-time host.
* Synthesize waveforms on the real-time host, pass them to the FPGA for processing, and then return them to the host for recording and playback.
* Communicate over a 3rd party protocol (MIDI) quickly and efficiently by implementing our driver on the FPGA.
* Leverage the myRIO’s web server to host a thin client—or custom web page—that allows the user to monitor the system.

With the myRIO’s wide feature set, flexibility, and power—developing a full-featured embedded system is within the reach of students and professional engineers alike.

# Digital Signal Processing

Using the myRIO’s FPGA, we are able to implement our digital signal processing in hardware rather than software. This gives us increased speed and performance over a real-time implementation.

## Combining Signals

Our data for each channel comes from the Real-Time host through a DMA FIFO in the form of an unsigned 64-bit number. We then split that 64-bit number in to 4 equal parts, each representing a track of audio. We then add each split part together on the FPGA to create a composite signal.



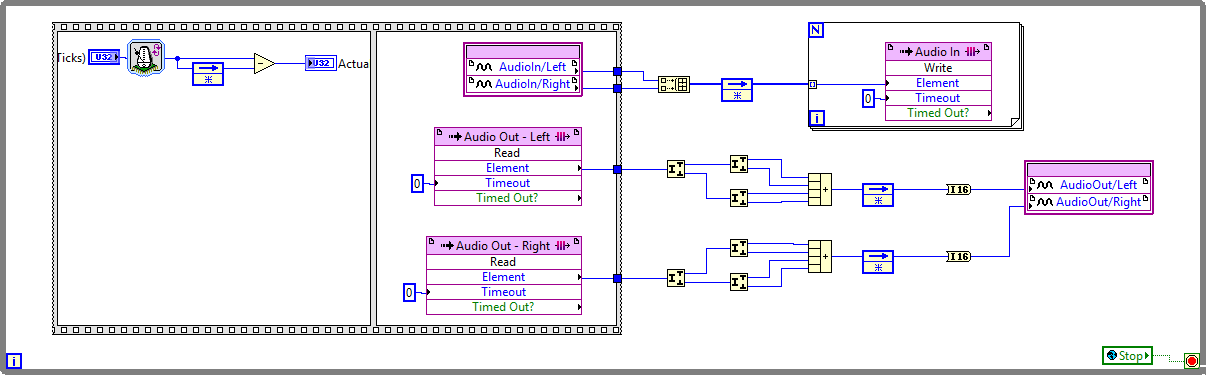
## Envelope Filter

How it works

Highlights

# FPGA Audio playback and recording

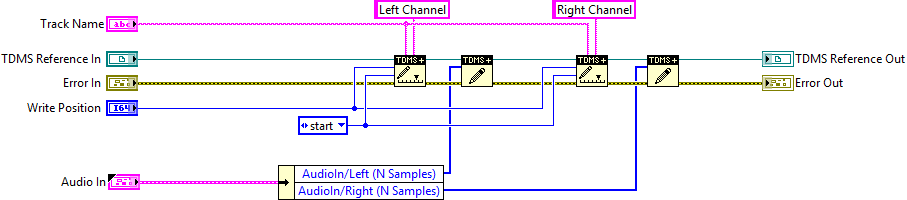
Using the myRIO’s on-board Audio I/O, we are able to capture and play back audio at rates in excess of CD-quality, or 44101 samples per second. As you can see from the code below, using LabVIEW FPGA makes creating and using DMA FIFOs for streaming to and from the host trivial in comparison to tradition implementation methods.



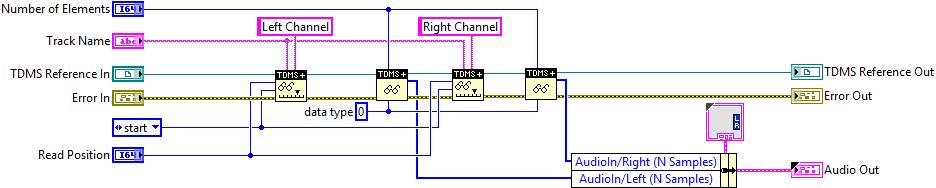
FPGA Implementation

# Streaming from TDMS

TDMS—or the Technical Data Management System format—has many advantages. It’s exchangeable, compact, searchable, capable of high-speed streaming, and allows for random access. By leveraging TDMS, we are able to quickly read and write data to the myRIO’s internal storage.



Write to TDMS File

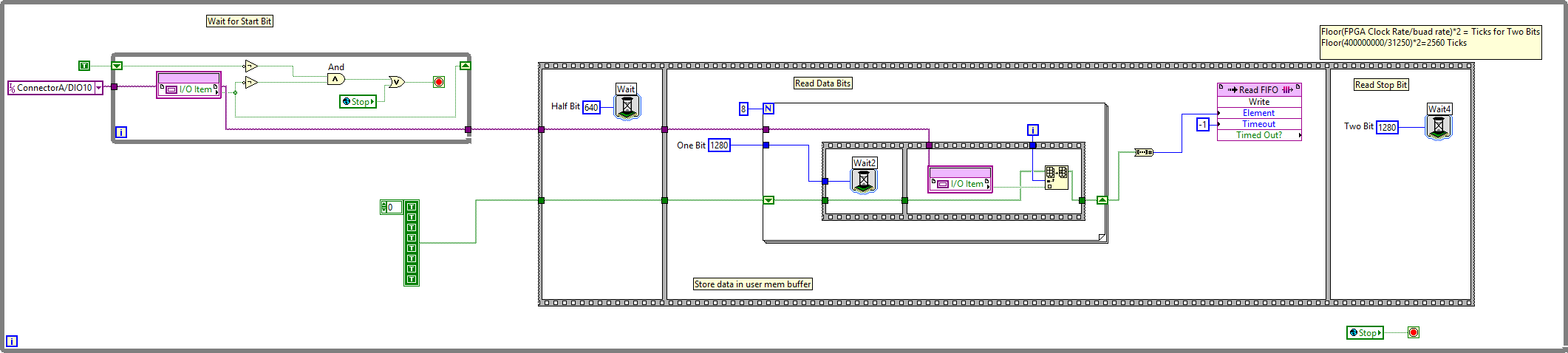


Read from TDMS File

# Communicating over MIDI

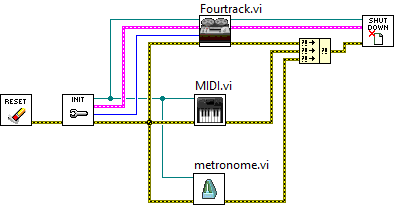
MIDI—or the Musical Instrument Digital Interface—is a 3rd party communication protocol that allows a wide variety of electronic musical devices to interface with each other. By creating a MIDI driver for the myRIO, we are able to use a wide variety of 3rd party devices to control our application.

The baud rate of MIDI communication is 31250 bits per second. This is a non-standard baud rate, so a custom driver implementation was necessary. We began with our FPGA implementation, where we were able to implement our desired baud rate using the LabVIEW FPGA timing Vis to synchronize reading the MIDI input. We then use a DMA FIFO to pass each byte up the the real-time host where it is used to control our application.



FPGA Serial Read

# Real-Time Application



# Thin Client