Operating Systems Latency Measurement and Analysis for Sound Synthesis and Processing Applications

Adrian Freed
Amar Chaudhary
Brian Davila
CNMAT, UC Berkeley
adrian,amar,davila@cnmat.berkeley.edu, http://www.cnmat.berkeley.edu/People

Abstract

There is increasing interest in using personal and workstation computer platforms for reactive sound synthesis and processing applications. Much experimentation and tuning is required with each operating system and hardware platform to achieve acceptable real-time performance for such applications. Tools to measure I/O latencies are critical for this tuning process. This paper introduces an affordable hardware-oriented measuring system.

1 Introduction

Desktop and workstation computers are now very attractive for reactive sound synthesis and processing applications. Unfortunately few operating systems were designed to deliver real-time performance and vendors generally don't guarantee or even specify the temporal performance of operating systems. Considerable experimentation is required with each operating system to achieve acceptable musical results.

Measurement of I/O latency is fundamental to achieving reliable real-time performance for musical applications and is the primary focus of this paper. The first challenge is to synchronously log stimulus events communicated with MIDI, ethernet, USB, IEEE-1394, audio input, etc., and subsequent changes in processed or synthesized audio sound streams.

"In vivo" measurement of latencies in most computing and networking systems is challenging. Even in the rare systems where time stamping is available, most logging techniques interfere with the performance of the system being evaluated. Also, many of the latencies to be characterized are hardware buffers inaccessible to software, e.g. FIR filter delay in audio codec reconstruction filters.

In this paper we describe an affordable hardware-based latency evaluation strategy applicable to any operating system or computer.

2 Event Logging

2.1 Audio Transcoding

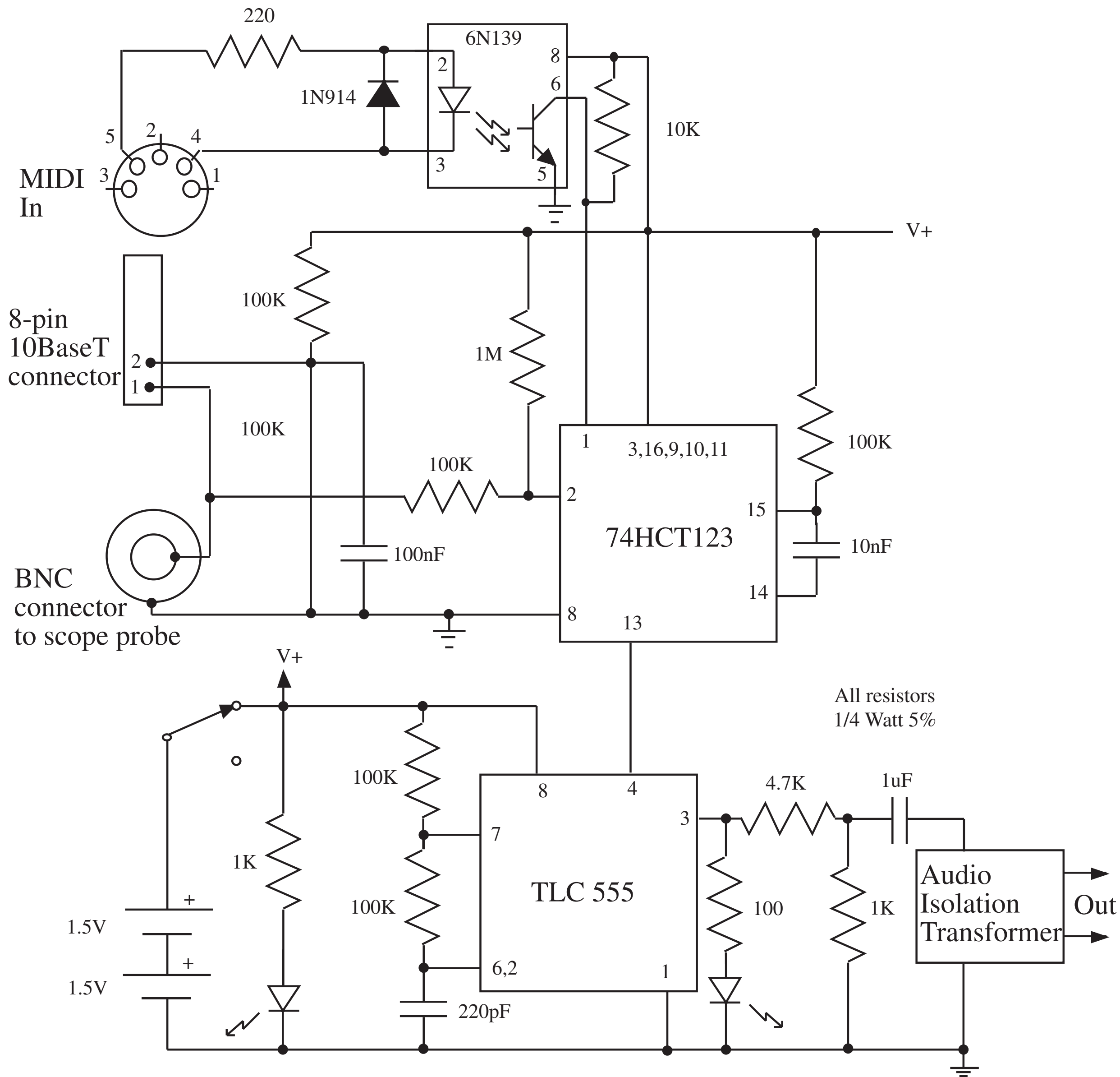
Modern instrumentation tools such as logic analyzers and digital storage oscilloscopes can perform event logging but are expensive and challenging to configure.

The simpler approach used here is to convert events from a wide range of sources into audio frequency signals. These signals and synthesized sound outputs are synchronously analyzed or recorded using standard multichannel audio recording tools, such as an ADAT recorder connected to an SGI O2 computer.

The multi-channel audio stream serves to bond together the event streams maintaining their temporal relationships despite latencies in the analyzing system.

This technique has been used [1] in stereo form to analyze timing relationship between key displacement and sound from a pipe organ.

The logging system requires a circuit to interface cleanly and non-invasively with the source of event information and a way to modulate events in the audio frequency range.



Circuit Diagram of the Event Transducer.

2.2 Interfacing

For MIDI events the logging circuit uses optical isolation as specified in the MIDI standard. This is inherently non-invasive since the logging circuit may be connected to a MIDI thru output.

A standard 8-pin T-connector is used For 10BaseT Ethernet to tap into the transmission or reception differential pair. Input resistors and the CMOS inputs of a 74HCT123 satisfies the requirement for a high impedance, non-invasive connection.

2.3 Down Sampling Circuitry

Pulse streams from Ethernet and MIDI are clocked at higher rates than available audio bandwidth, so some information will be lost down-sampling. A 74HCT123 monostable multivibrator is used to stretch out data transitions by a fixed time interval that is long enough to gate an audio oscillator implemented with a CMOS 555 timer. Details of each bit transition are lost in this scheme, but the event beginnings are accurately captured.

The time constant of the retriggerable monostable is the temporal uncertainty of the end of an event. The time constant is set to provide adequate accuracy when logging at audio rates.

The oscillator output drives an audio frequency isolation transformer. This isolates the circuit from the audio system, eliminating ground loops. The use of battery power supply allows the input to float, minimizing common mode problems.

The choice of 3V supply is important. It allows the front-end to successfully capture transitions from a wide variety of inputs including RS422, TTL, RS232 and MIDI current loop. The high resistance values chosen in the front end are important: they provide current limitation for the protection diodes built into the 74HCT123's inputs which are used here for clamping.

Note that the circuit exploits the two trigger inputs of the multivibrator avoiding a switch between the current loop and other event sources. The desired source is simply plugged in. Both these inputs have special internal circuits with hysteresis to minimize false triggering and improve noise immunity.

3 Results

The circuit described here is one of a set of tools being developed as part of a broad initiative, the Responsive Systems Validation Project (RSVP) [3]. These tools have been used to measure sound synthesis software performance on SGI, Macintosh and PC machines with MIDI, Ethernet and gestural input devices such as digitizing tablets [2].

The original thought behind the latency measurement tools was to use them to tune sound synthesis software on each platform towards the latency goal of 10 ± 1mS. Although such a goal is within sight on these systems, the measurement tools revealed significant bugs and design flaws in the underlying operating systems and computers with respect to real-time performance. Since work to address these flaws is being vigorously pursued by some of the vendors, it would be unfair to publish particular numbers at this stage - numbers that would be meaningless by the time you read this. Progress may be gauged by monitoring the aforementioned RSVP web pages.

It is encouraging that the performance difficulties identified by latency measurement tools were not from inadequate real-time features in the operating systems, but design and implementation decisions that defeat real-time requirements. A pervasive difficulty stems from poor implementations of I/O drivers. Programmers still cut corners by holding off interrupts for long periods. This is especially troublesome on systems assembled from many vendors, i.e. Intel PC's, since there is no industry wide real-time validation suite in use to catch this sloppiness. Widespread use of affordable tools such as the one described here will eventually result in affordable reactive computing.

4 Future Work

More sophisticated front-end hardware is required to monitor USB and Firewire.

A small run of a multichannel version of the circuit outlined here is planned.

5 Acknowledgments

We gratefully acknowledge the financial support of Emu Systems, Gibson, and Silicon Graphics Inc. Doug Cook and Chris Pirazzi clarified many performance and timing issues on SGI machines.

References

[1] Kostek, B., A. Czyzewski, 1993, “Investigation of Articulation features in Organ Pipe Sound,” Archives of Acoustics, vol.18, (no.3):417-34.

[2] Wright, M., D. Wessel, A. Freed, 1997. “New Musical Control Structures from Standard Gestural Controllers,” proc. ICMC, Thessaloniki, Greece.

[3] http://www.cnmat.berkeley.edu/RSVP