BY KENNETH CISZEWSKI • TECH ELECTRONICS

Beware of what you think you know!

DANIEL VASCONCELLOS

I once helped design a digital-guitar amplifier that models the sound of various analog-guitar amplifiers. This amplifier was the first major digital-audio product in what had been an analog-only guitar-amplifier line. The task presented two unusual challenges: generating a synchronized serial-connection clock and correctly programming a flash EPROM.

I used a DSP and a stereo-audio codec that communicated via a synchronous serial port using four connections: data out from DSP to codec, data into DSP from codec, frame synchronization, and a synchronous-clock line that controlled serial-data transfer. I got a sample board up and running with a loopback-audio routine, connected an audio source, and connected a monitor amplifier, but there was no audio coming out of the codec's analog ports. Checking the serial-data connection with an oscilloscope, I found that serial data was running in both directions between the DSP and the codec. The synchronous-clock line and the frame-sync line looked OK.

I wondered whether the external 12-MHz logic clock had to connect to the codec to synchronize with the serial-data-connection clock. Without synchronization, the audio coming out of some codecs would sound like white noise loud enough to damage both loudspeakers and listeners' ears. The data sheet for the device I was using didn't list this requirement. The codec field-application engineer told me that clock synchronization was not a requirement.

A PLL in the DSP converted the 12-MHz clock to higher frequencies, which were then divided down to generate the serial-data-connection clock. The 12-MHz clock came straight from a crystal oscillator that also supplied the 12-MHz clock to the codec. Testing showed that the logic clock and the serial-connection clock were not synchronous but instead sliding asynchronously past each other.

I divided down the 12-MHz logic clock using a CMOS 4040 IC and fed it to both the DSP and the codec. This method removed the PLL in the DSP from the picture and guaranteed that the clocks would synchronize. Upon firing up the board, audio came forth from the codec outputs!

The second challenge came when I was programming flash EPROMs using our EPROM programmer and inserting them into various sample boards. The boards would work fine. I would turn them off and power them back up, and they would be dead. I put one of the flash EPROMs back into the programmer and read back the data that was in the EPROM. The power-up-reset vector, all of the interrupt vectors, and a good portion of the EPROM's contents had changed.

An analog engineer thought the problem might be the power-up sequence at the flash EPROM. We explored that possibility for a while, providing some capacitive delays, but nothing helped. Early EEPROMs required a huge amount of analog-support circuitry, due to the analog-sequencing circuitry necessary to prevent them from altering their contents during power-up, but the device I was using had no such requirements. I found that the flash-EPROM programmer lacked a proper sector-relocking sequence as part of its programming algorithm.

The analog engineer I worked with often said, "It's what you think you know that's holding you back." We thought we knew how to generate a serial-connection clock and that our EPROM programmer worked correctly. We were surprised and wrong on both counts. EDN

Kenneth Ciszewski is a senior project manager for Tech Electronics in St Louis. Like Kenneth, you can share your Tales from the Cube and receive $200. Contact Maury Wright at mgwright@edn.com.