Audio Hardware Interface ‘96
Design Guide

Version 1.3, January 9, 1996

Table of Contents

List of Figures

List of Tables

1. Introduction

The goal of this document is to define a minimum set of audio system functional requirements that will facilitate the development and availability of full featured, cost effective, audio components for **"Communicating PC"** systems shipped in 1996. Audio hardware component IHVs, with assumed PC audio technical expertise, are the target audience of this document.

The Communicating PC is expected to support all of the following audio capabilities:

- **System audio** High quality audio for business, games, and multimedia
- **CD audio:** CD-ROM audio with internal connections to Codec mixer
- **Telephony:** DSVD MODEM & speakerphone which utilizes system mic & speakers
- **Video capture:** Video Camera or TV tuner audio with internal connections to Codec mixer

PC "games compatible" audio has established itself as the last de facto industry standard subsystem. Soundblaster and 8237 DMA hardware compatibility are required to run the majority of the most popular multimedia, and entertainment software titles. This document comprehends these market realities.

In addition, as the Communicating PC platform becomes more responsive to the demands placed upon it by multimedia and entertainment software, it becomes possible to reduce system cost without sacrificing performance or feature richness. Audio will be one of the first platform subsystems that will be able to take advantage of the Communicating PC system's performance headroom.

This Audio Hardware Interface '96 Design Guide covers the following topics:

- Host interface
- MPC2 & MPC3 capabilities
- 3D audio
- Codec mixer functionality
- Support for software echo cancellation
- Independent ADC and DAC sample rates
- Volume control
- Target Audio System "socket" implementation

The final section is a Communicating PC audio requirements summary, detailing the following:

- Codec Audio Hardware
- System Audio Hardware
- System Audio Quality
- Windows Audio Drivers & API's

1.1 Related documents and specifications

This document details Intel's "Communicating PC" platform requirements. Other related specifications should be considered (this is not a complete list):

- MPC Working Group's MPC2 & MPC3 specifications (http://www.spa.org/mpc/default.htm)
- Microsoft's "Hardware Design Guide for Windows 95", chapter 5 (Microsoft Press)
- Microsoft's "The Desktop PC 95", chapter 3 (Microsoft Press)
- Microsoft's "PC 96" suppliment to "Hardware Design Guide for Windows 95"
- Microsoft's Games SDK & DirectSound API (MSDN CD-ROM)

2. Host Interface

This section describes the current "games compatible" host interface. The systems issues associated with the current interface will be explored, and a cost effective solution will be presented.

2.1 The Audio Sample Datapath

Current "games compatible" PC audio subsystems utilize the 8237 DMA controller to move audio sample data between the audio interface and the CPU/Memory subsystem. The 8237 DMA programming and hardware models have become de facto standards and the majority of industry leading games titles require 8237 register level compatibility in order for their audio to function properly[1].

However, while the 8237 DMA controller is a well understood common development vehicle, the extremely slow ISA bus timing associated with its use, coupled with the need to command the entire system for the full duration of its accesses, severely bottlenecks the system's ability to keep the CPU/Memory subsystem executing at the rate that would be sufficient to keep up in real time (assuming more than just audio is running in the system).

In a PCI Release 2.1 compliant system the maximum access latency requirements on the PCI bus cannot be satisfied when the 8237 DMA controller masters on PCI at ISA timings. Therefore the ISA and PCI busses must be decoupled from one another to enable more efficient use of the PCI bus, while at the same time 8237 register level compatibility be maintained.

The following two figures illustrate the impact on system utilization for decoupled, versus coupled PCI / ISA bus audio sample playback transactions.

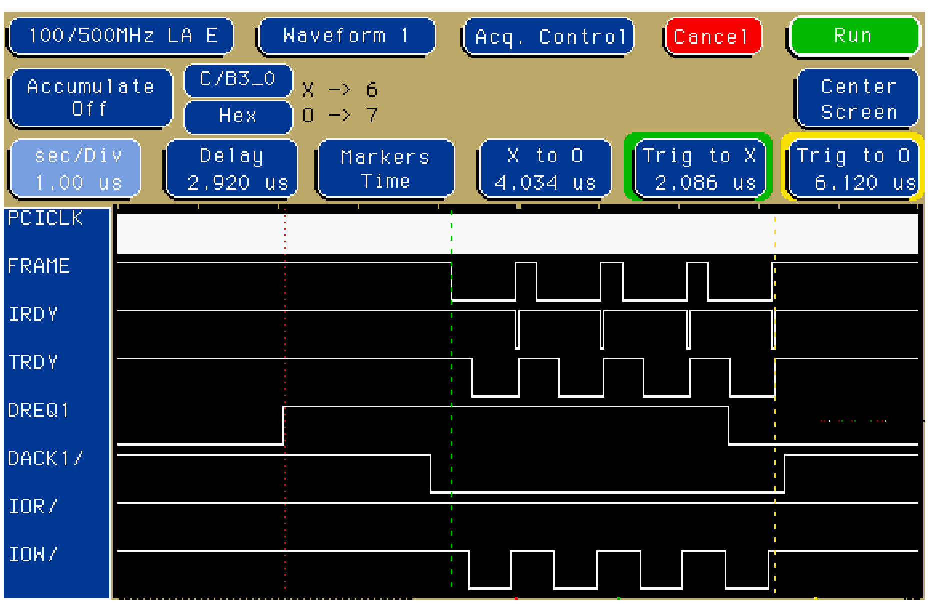


Figure 1. Unconditioned PCI / ISA DMA Transactions

The figure above illustrates the system utilization for a 16 bit stereo playback sample. Since there is no decoupling of the PCI and ISA busses, the transactions occur along both busses in real time consuming approximately 4uS where nothing else in the system can occur except for the audio sample transport from main memory to the audio subsystem.

[1] Most major game titles require a "Real Mode DOS" environment to run.

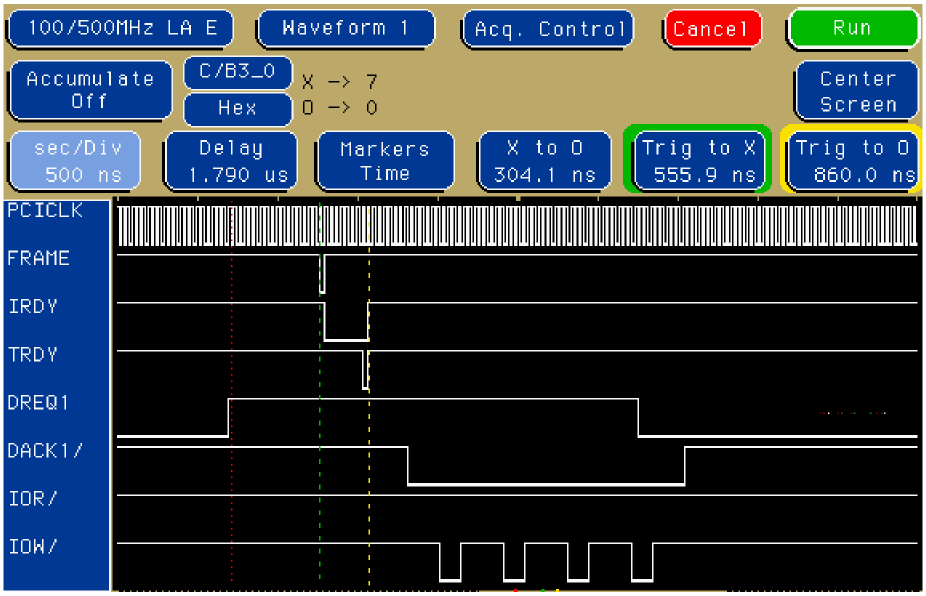


Figure 2. Decoupled PCI / ISA DMA Transactions

Figure 2. shows significantly reduced system utilization for audio playback sample transport. With the busses decoupled, the sample data is prefetched from main memory (prior to the assertion of DACK# to the audio subsystem) across PCI in a single DWORD transaction that is stored in the expansion bus bridge. The PCI, Host and Memory busses are now freed for other activities while audio sample transport completes between the expansion bus bridge and the audio subsystem, "offline" with respect to the rest of the system.

The following table summarizes the system utilization savings (Host, Memory, and PCI Bus) realized when the ISA DMA agent (audio system) is decoupled from the PCI bus with data collection buffering in the PCI to ISA bridge[2].

Full Duplex Audio (Playback and Record)
(all units = ns)

Codec Sample Transport	**Playback Sample**	**Record Sample**	**% Host/PCI Bus utilization**
8237 8-bit compat.	3750	3750	34
8237 16-bit compat.	1875	1875	17
Buffered Datapath (8 or 16 bit)	350	120	2

- **8237 ISA 8 bit Timing**
  - Playback = 4 x 1 byte read transfers (8 bit 8237 compatible DMA timing)
  - Record = 4 x 1 byte write transfers (8 bit 8237 compatible DMA timing)
- **8237 ISA 16 bit Timing**
  - Playback = 2 x 1 word read transfers (16 bit 8237 compatible DMA timing)
  - Record = 2 x 1 word write transfers (16 bit 8237 compatible DMA timing)
- **Buffered Datapath (Demand Mode Codec)**
  - Playback = 1 x 32 bit read transfer (350 nS)

[2] To sustain the Communicating PC audio environment, 1 DWORD write (record), and 1 DWORD read (playback) must be pushed/pulled respectively from DRAM once every 22.6 uS (sample period). Data was measured on an Intel 82430 FX PCIset based platform.

- ♦ Record = 1 x 32 bit write transfer (120 nS)

As can be seen from the data, the isolation of the ISA bus from the PCI bus brings the CPU/PCI system utilization down to 2% for audio sample transport. ISA bus utilization is cut in half going from an 8 bit ISA bus device to a 16 bit ISA bus device, however the far larger system utilization problems of the PCI, CPU/Cache, and Memory busses are addressed equally well for 8 or 16 bit ISA bus interfaces.

Additionally, systems will most likely begin to appear without ISA "slots" prior to the demise of "Real Mode DOS". These systems will still provide an 8 bit interface for peripherals such as the system's Flash BIOS. An 8 bit bus interface is suitable for either system configuration.

Finally, the audio subsystem's host inteface should support on-chip sample FIFOs to guard against long instantaneous latencies that could be encountered in the system. Separate FIFOs, providing mutually exclusive support for both the record and playback channels are required. The minimum FIFO depth should be ten 16-bit stereo samples for each channel.

When preloading the FIFO with intial playback data, or loading multiple samples into the FIFO after having fallen behind due to long latency that was encountered, the FIFO should be loaded at a rate of no more than one sample for each demand mode DREQ/DACK# arbitration cycle. This sample oriented technicque will preclude potential preemption of the codec during the process which would otherwise adversly affect performance.

Audio Sample Datapath Minimum Requirements

Given the data, the most cost effective,Communicating PC audio subsystem datapath consists of:

- 8 bit ISA Bus interface
- Type "F" DMA Timing Capable (3 SYSCLK Cycles)
- Graceful ISA bus preemption via deassertion of the Codec's DACK#
- 2 x "Demand Mode[3]" DMA Slave interface (Full Duplex operation)
- On-chip, record and play, 16 bit stereo sample FIFOs with buffering for 10 samples in either direction simultaneously
- 1 ISA Interrupt Request

2.2 Plug and Play

The audio hardware interface must provide support for industry standard (ISA) Plug and Play functionality. This enables auto-configuration of ISA resources under software control contributing to the platform's ease of use.

Generally speaking, audio may be supported in the platform either by integrating it onto the motherboard or by providing audio on an ISA add-in. For integrated motherboard solutions it may be advantageous for the OEM to provide full PnP BIOS support for audio by creating BIOS

[3] The most cost effective bridge solution would house a single address tag for a given DREQ/DACK# sequence using a 32 bit collection buffer. "Single Transfer" mode is not recommended as the buffer would need to be flushed, or filled, for each ISA bus transaction. For an 8 bit audio interface in Single Transfer Mode there would be 4 independent PCI transactions vs. the Demand Mode protocol which would collect all four byte in the 32 bit bridge buffer prior to executing a single PCI transaction.

Device Nodes for each of the audio subfunctions. In this way all runtime, as well as boot time, configurations may be managed by the BIOS (even in a Windows 95* environment).

ISA resource configuration in a DOS/Windows 3.x environment is currently managed exclusively by the motherboard's ISA Plug and Play enabled BIOS. This will no longer be true with the introduction of Windows 95, as the operating system will manage any runtime resource reconfigurations, as well as any boot time configurations for devices that are bus enumerated (vs. those enumerated by the BIOS).

Plug and Play software configuration routines moving into the domain of the operating system could present system issues, the nature of which will now be described. The BIOS has a complete understanding of the platform's feature set, whereas the operating system does not. For example, an OEM may incorporate performance enhancements that are applied to ISA resources in a proprietary fashion in order to differentiate their product. While the Plug and Play BIOS comprehends all platform attributes both standard and proprietary, the operating system would have no way of discovering the presence of any feature that is not identifiable via standard Plug and Play algorithms. In the example above, the performance enhancements could be lost following a runtime reconfiguration of ISA resources performed by Windows 95. To provide a flexible approach for addressing this issue, the ISA Plug and Play interface should be implemented such that a component specific resource key (in addition to the ISA PnP key sequence) could also be executed by the BIOS for any runtime configurations. The next section provides an example of how this "back door" hook could be used.

2.2.1 Motherboard Audio

The following illustrates how to apply the "back door" functionality to a motherboard audio implementation in a Windows 95 environment:

1. BIOS performs the standard ISA PnP isolation, identification and configuration of the PnP motherboard devices in the system, namely the audio subsystem.
2. BIOS enumerates the audio system by creating device nodes for DSP (Wave), FM (OPL3), MIDI I/O (UART), and Game Port (Joystick) subfunctions.
3. Finally, BIOS disables the audio system's ISA PnP Key Sequence decode, thereby taking the audio system out of any future runtime ISA PnP reconfigurations that could be attempted directly by Windows 95.

If Windows 95 needs to reassign ISA resources during runtime for reasons such as the insertion of a hot swapable device, the audio host interface would not appear in the ISA PnP Isolation Sequence, but would be reported to the OS as BIOS dev nodes which would ONLY be directly managed by the BIOS. The OS would request changes for the audio system's resources through the BIOS which would now access the component via a "back door" to perform any reconfigurations required.

2.2.2 ISA Add-in Audio

The add-in case would impose no BIOS overhead beyond the boot time ISA PnP functionality, and would always report into the system through the ISA PnP algorithms.

Plug and Play Minimum Requirements:

The audio system's host interface must provide a fully compliant ISA PnP front end, as well as an alternative, proprietary means of programming the component's ISA resources. When the audio hardware is configured to respond to its proprietary key sequence, the ISA PnP key sequence decode should be disabled.

ISA resource mapping should be supported for the following DMA and Interrupt options

3 DMA DREQ/DACK# pairs options

- CH0, CH1, CH3

4 IRQ pin options

- IRQ5, IRQ7, IRQ9, IRQ11

2.3 Soundblaster*, Soundblaster Pro*, and WSS* Compatibility

The audio system must be 100% Soundblaster, Soundblaster Pro, and optionally WSS compatible. The Soundblaster/Soundblaster Pro functionality addresses Real Mode DOS games compatibility.

Sound System Support Minimum Requirements:

- 100% Soundblaster Compatible
- 100% Soundblaster Pro Compatible

WSS Compatibility isoptional.

2.3.1 Soundblaster/Soundblaster Pro Compatible Register Files

The following tables map the Soundblaster/Soundblaster Pro compatible I/O and Mixer register files. It is assumed that the audio IHV has an expert understanding of the underlying programming model. These register maps are included for quick reference only.[4]

I/O Address	Register Description	Accessibility
SB_base + 0	Left FM Status Port	Read
SB_base + 0	Left FM Register Status Port	Write
SB_base + 1	Left FM Data Port	WO
SB_base + 2	Right FM Status Port	Read
SB_base + 2	Right FM Register Status Port	Write
SB_base + 3	Right FM Data Port	WO
SB_base + 4	Mixer Register Address	WO
SB_base + 5	Mixer Data Port	R/W
SB_base + 6	Reset	WO
SB_base + 8	FM Status Port	Read
SB_base + 8	FM Register Port	Write
SB_base + 9	FM Data Port	WO
SB_base + A	Read Data Port	RO
SB_base + C	Command / Write Data	Write
SB_base + C	Write Buffer Status	Read
SB_base + E	Data Available Status	Read

Table 1. SoundBlaster / SoundBlaster Pro Compatible I/O Map

[4] Addresses are provided as “Base + Offset” as it is assumed that the Soundblaster* interface is relocatable via ISA PnP

The following table outlines the Soundblaster/Soundblaster Pro compatible mixer I/O interface. The mixer address map is accessed via an index/data register pair found in the SB I/O register file. The mixer index register resides at address = SB_base + 4 and the mixer data port resides at address = SB_base + 5 (see table 2.1).

Index	D7	D6	D5	D4	D3	D2	D1	D0
00h	Data Reset							
02h	RESERVED							
04h	Voice Volume Left				Voice Volume Right			
06h	RESERVED							
08h	RESERVED							
0Ah	X	X	X	X	X	Mic Mixing		
0Ch	X	X	Input Filter			Input Select		X
0Eh	X	X	X	X	X	X	VSTC	X
20h	RESERVED							
22h	Master Volume Left				Master Volume Right			
24h	RESERVED							
26h	FM Volume Left				FM Volume Right			
28h	CD Volume Left				CD Volume Right			
2Ah	RESERVED							
2Ch	RESERVED							
2Eh	Line Volume Left				Line Volume Right			

Table 2. SoundBlaster / SoundBlaster Pro Compatible Mixer I/O Map

2.3.2 WSS (Wave Audio) Compatible Register Files

The following tables outline the WSS (Wave Audio) compatible I/O interface. It is assumed that the audio IHV has an expert understanding of the underlying programming model. These register maps are included for quick reference only[5].

I/O Address	Register	Register Description
WSS_base + 0	R0	Index Address Register
WSS_base + 1	R1	Data Port
WSS_base + 2	R2	Status Register
WSS_base + 3	R3	PIO Data Register

Table 3. WSS Codec Direct Mapped Register Interface

Index	Register Name	Index	Register Name
I0	Left ADC Input Control	I16	Alternate Feature Enable I
I1	Right ADC Input Control	I17	Alternate Feature Enable II
I2	Left AUX #1 Input Control	I18	Left Line Input Control
I3	Right AUX #1 Input Control	I19	Right Line Input Control
I4	Left AUX #2 Input Control	I20	Timer Low Byte
I5	Right AUX #2 Input Control	I21	Timer High Byte
I6	Left DAC Output Control	I22	Alternate Sample Frequency
I7	Right DAC Output Control	I23	Alternate Feature Enable III
I8	Fs & Playback Data Format	I24	Alternate Feature Status
I9	Interface Configuration	I25	Version/Chip ID
I10	Pin Control	I26	Mono Input & Output Control
I11	Error Status and Initialization	I27	Left Output Attenuation Control
I12	Mode and ID	I28	Capture Data Format
I13	Loopback Control	I29	Right Output Attenuation Control
I14	Playback Upper Base Count	I30	Capture Upper Base Count
I15	Playback Lower Base Count	I31	Capture Lower Base Count

Table 4. WSS Codec Indirect Reg. Interface

[5] Addresses are provided as “Base + Offset” as it is assumed that the WSS* interface is relocatable via ISA PnP

3. MPC-2, MPC-3 Capabilities

MPC-2, and MPC-3 compliance requires a minimum feature set as well as certain functionality from the Codec mixer. The desired mixer configuration for the Audio System of ‘96 will be described in the Mixer section. In this section, the feature set necessary to support Communicating PC audio requirements and the relationship to MPC-2 and MPC-3 will be introduced.

3.1 FM Synthesis

MPC-2 compliance requires a synthesis mechanism in the hardware, independent of operating system. For this reason, it is desired that the audio subsystem support, at a minimum, an FM synthesis hardware mechanism. The FM synthesis capabilities must be compatible with the OPL3 hardware solution.

Address	Register	Access
0x388	FM Status	RO
0x388	FM Address 0	WO
0x389	FM Data 0	WO
0x38A	FM Address 1	WO
0x38B	FM Data 1	RO

Table 5. Standard Adlib* Compatible Synthesizer Register Map

3.2 WaveTable MIDI Synthesis

MPC-3 requires WaveTable MIDI music synthesis, which may be implemented in software, or hardware assisted software or via an external MIDI device as described in the next section.

3.3 MIDI

A standard serial MIDI (UART) port is required in the Audio subsystem.

Address	Register	Access
0x330	MIDI Xmt/Rcv Port	R/W
0x331	Command Register	WO
0x331	Status Register	RO

Table 6. Standard MIDI Register Address Map

3.4 Game (Joystick) Port

The joystick is not an integral part of the audio subsystem. However, joystick use is associated with games and traditionally, this capability has been integrated with audio cards. As a result, base platforms w/o audio do not support joysticks, but audio enabled platforms need to support joysticks. The audio IHV should consider that ISVs writing new games for the host processing environment are being encouraged to move the game port to a digital, interrupt driven interface. System utilization would be dramatically reduced when operating with a digital USB joystick. That being said, the games compatible audio system for 1996, must at a minimum, provide support for the de facto standard analog joystick interface.

D7	D6	D5	D4	D3	D2	D1	D0
J2B2	J2B1	J1B2	J1B1	J2Y	J2X	J1Y	J1X

J2B2 : Joystick 2 Button 2
J2B1 : Joystick 2 Button 1
J1B2 : Joystick 1 Button 2
J1B1 : Joystick 1 Button 1
J2Y : Joystick 2 Y Position
J2X : Joystick 2 X Position
J1Y : Joystick 1 Y Position
J1X : Joystick 1 X Position

Table 7. Standard Game Port (0x200-0x207) Bit Assignments

4. 3D Audio

The Communicating PC audio environment provides the ability to process audio streams with 3D software algorithms. The cost of implementing hardware to create 3D effects is thereby minimized.

4.1 Disable feature

If hardware is provided to create 3D effects, it is required that there be a mechanism to allow the bypassing of the hardware 3D effect. The reason for this requirement is that multiple 3D effect filters distort the desired effects of the individual algorithms. If a 3D algorithm is running in software, it becomes necessary to disable the hardware 3D effect.

The mechanism to disable hardware 3D must include support in the audio device driver and the Mixer Application as well.

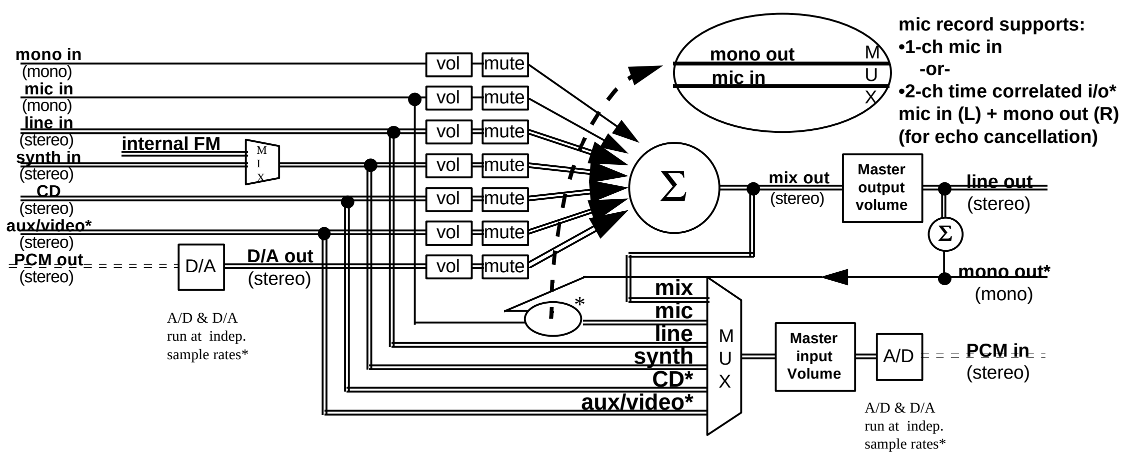
4.2 Width control

If the hardware 3D implementation offers “width” control, it must be possible to achieve the control by writing to a register. The Communicating PC audio device driver and the mixer application must include support for this function.

MPC Compliance Minimum Requirements:

- FM Synthesis (OPL3 compatible) hardware
- MPU-401 compatible MIDI port
- Game (Joystick) Port
- If 3D Sound Hardware is provided (optional), it must be capable of being disabled via software

5. Codec Mixer Functionality



NOTE: * features recommended

source	function (typical)	connection (typical)
mono in	telephony card out	cabled from telephony card
mic in	desktop microphone	from mic connector
line in	external audio source	from line in connector
synth in	FM synth or H/W wavetable synth	within CODEC or external source
CD in	audio from CD-ROM drive	cabled from CD-ROM
aux/video in	audio from TV tuner or video camera	cabled from TV/VidCap card
PCM out	digital audio output from PC	within CODEC
D/A out	analog audio output from PC	within CODEC
mono out	mono mix of all audio sources	cabled to telephony card
line out	stereo mix of all audio sources	to speaker out & line out connectors
PCM in	digital audio input to PC	within CODEC

Figure 3. Analog Mixer Functionality

The Codec audio mixer should be capable of accepting data from the sources shown in Figure 3. There are inputs for mono in, microphone, external line source, MIDI synthesis source, internal CD player, internal video source, and the computer's own D/A output. Each source must have an independent volume control. Record and playback must have an over-all master volume control. Single chip audio solutions which implement FM synth internally should still support an external **synth in** input for upgrade to hardware wavetable.

With this design **mix out** can be any combination of the sources, and each source can be represented on a S/W mixer panel with a persistent volume setting. The input MUX can select any one of the sources, or the the entire **mix out**, and a persistent master input volume can be associated with each MUX option.

An alternate version of this design replaces the input MUX with a second set of volume controls and sums to create an independent mix for input.

6. Enhanced full-duplex mode capabilities

6.1 Independent ADC and DAC sample rates

Full-duplex audio environments require additional flexability in the programming of the Codec sample rates for input and output. The requirements here are to:

- support high quality games output (22Kss stereo) while capturing telephony quality input (8Kss mono)
- Support DSVD games via headsets and "hands free" speakerphone
- match the input sample rate to the transmission channel (typically 8Kss mono)
- accomodate software echo cancellation algorithms (which currently require mono 8Kss data)
- minimize CPU utilization, by avoiding the need to perform sample rate conversions in software

The input A/D converter should have the capability to run at a sample rate and channel width that is independent of the output D/A converter.

In a DSVD games scenario using "headset", for example, the system audio can output games audio at 22Kss stereo (perhaps with the 8Kss far side voice digitally mixed in) while supporting an input channel which can run at the mono 8K sample rate which supports transmission of the near side voice back across the telephony channel.

6.2 Support for Host-based Echo Cancellation

Echo cancellation is a requirement for speakerphone functionality in full-duplex (simultaneous input and output) PC telephony and conferencing environments, and will become a popular feature for "hands free" DSVD games. It involves removing all reflections of the output signal (which plays thru the speakers) from the incoming signal (captured at the microphone). An echo cancellation filter requires digital representations of both the input and output data streams.

In a “hands free” DSVD games scenario using speakerphone, for example, the system audio can play the games & telephony mix at 22Kss stereo while supporting an input channel which can run echo cancellation at the mono 8K sample rate and supports transmission of the echo free near side voice back across the telephony channel. As processor MIPS and communications channel bandwith scale upward this same architecture enables echo cancellation for wide-band audio using 16K (or higher) sample rates.

6.2.1 2-channel time correlated i/o data format

The current generation of host-based echo cancellors operate on a data format known as “ ***time correlated i/o***”. This format is only meaningful when the Codec is operating in full-duplex mode. The time correlated i/o format is a 2-channel format which resembles the traditional interleaved stereo format. Each sample, instead of containing left and right inputs, contains an input (captured from mic) in the left channel and the current output (destined for speakers) in the right channel.

6.2.2 Software support for time correlated i/o

Audio drivers which support a full-duplex Codecs should be able to correlate the Codec’s **PCM out** and **PCM in** streams (refer to figure 3) to produce the 2-channel time correlated i/o format. If mono PCM is playing out, the audio driver simply interleaves the output samples (which just played) with the incoming mono PCM samples (which were just collected). If stereo PCM is playing out, the audio driver preferably adds the left and right output samples and interleaves the sum with the incoming mono PCM samples. In either case, 2-channel time correlated data is made available to the input stream. This technique has been successfully used to develop host-based echo cancellors running at 8Kss on a variety of Codec’s.

6.2.3 Hardware support for time correlated i/o

On a multimedia Codec there are additional analog audio sources in the output signal (refer to figure 3). There is no way to capture this analog data at the software driver level. The mixer diagram in figure 3 defines a simple addition that can be made to the multimedia Codec input MUX to support the capture of 2-channel input data in the time correlated i/o data format. Recording 2-channel data from mic returns **mic in** in the left channel and the Codec’s internally generated **mono out**for the right channel.

7. Volume Control

A “master” volume control allows the user to control the overall volume of the audio produced by the computer from all sources. This theoretically provides a single point of control that can be used to adapt to the changing environment and to changes in individual user preferences. In practice, there can be as many as three independent “master” controls in a PC audio system:

Software applications with audio content typically allow control of the software master volume supported by the Codec mixer. This is achieved through the use of a setup program, or some other screen based software representation of the Codec mixer interface. Applications also exist that control the master volume under based on such things as the arrival of a phone call, time of day, feedback from the room, etc.

Almost all PC speakers have their own built-in amplifiers and also provide a hardware volume control. This gives the user immediate access to volume adjustments, which can be made even when a software mixer interface is unavailable (such as before boot time, or after a crash).

Some OEM's have also added a user accessible hardware master volume control located on the system front panel, which provides immediate access, and is inherently easier to operate then the software mixer applets. Some Codecs implement a digital control to support this feature.

The ideal PC solution would be speakers without volume controls and a hardware volume control conveniently located on the system front panel (or monitor), which can be tracked by the Codec so that the whenever there is a hardware volume change it is also reflected in software. Our recommendation is that the Codec support a minimum of 2 digital inputs (momentary on switches) for decoding of three functions: volume up, volume down, & mute. Changes should introduce an interrupt to the audio driver so that the changes can be reflected in S/W.

8. Target "Socket" Design

This section describes a functional "socket" for the target Audio Hardware Interface '96 design. The 1995 Audio subsystem is described, followed by the recommended Audio '96 socket.

8.1 1995 Audio Subsystem

The 1995 audio subsystem employs an audio chipset that delivers on most of the requirements for a Communicating PC audio subsystem.

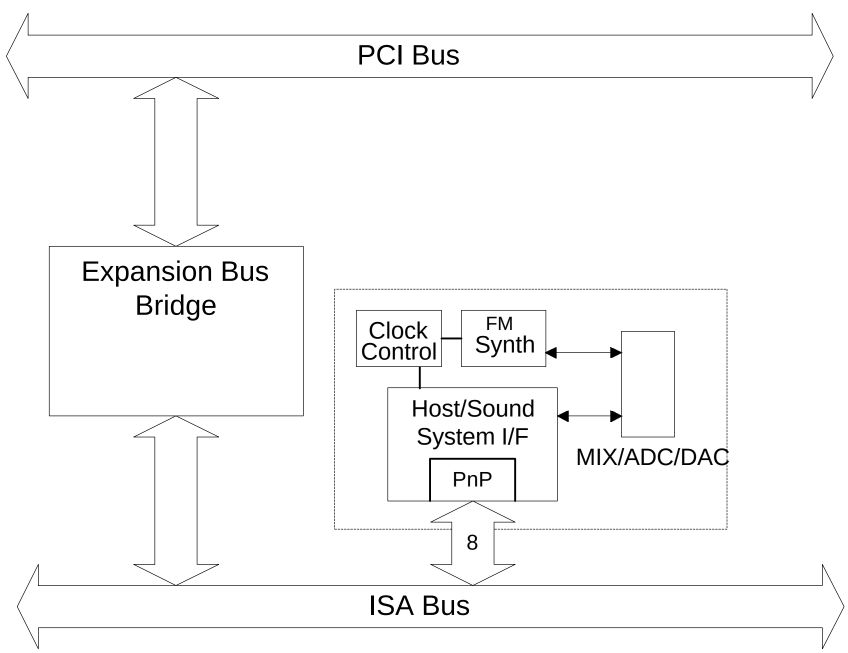


Figure 4. 1995 Audio Solution

This solution, while satisfying most of the Communicating PC audio functional requirements, added substantially to the system cost for audio due to low levels of integration which required 4 components along with the external discretes.

8.2 Audio '96 Target Implementation

This implementation represents the target for the Audio Hardware Interface '96 Design Guide.

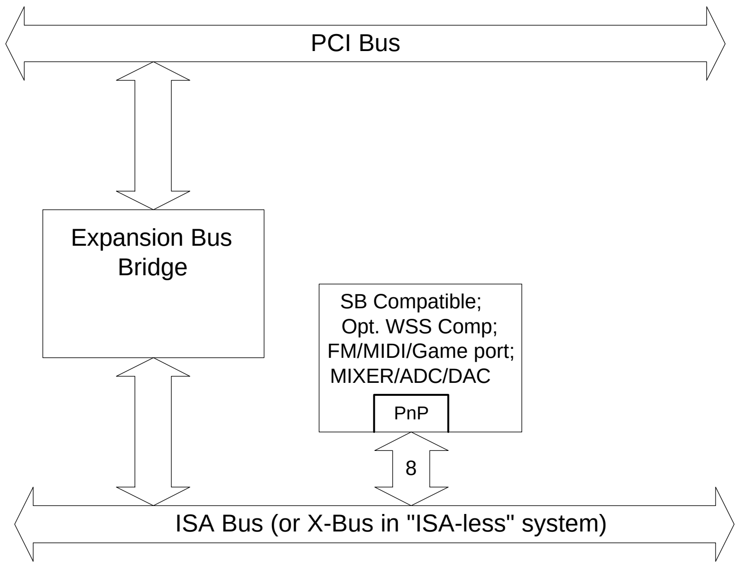


Figure 5. Audio Hardware Interface '96 Design Guide Target Solution

The single component solution integrates all of the functionality required for the Communicating PC. However now, with its higher levels of integration, the audio subsystem should be able to realize a dramatic cost savings. This component could be integrated onto the motherboard as an ISA (System Data Bus) or X-Bus (motherboard ISA peripheral bus) peripheral (in systems with no ISA Bus[6]) and would leverage the required enhancements in the Expansion Bus Bridge to obtain good performance with low Host/Memory/PCI bus utilization. This component would perform equally well if sold into the installed base of systems as an ISA Plug and Play compliant add-in for audio.

[6] Because the audio interface does all of its own address space decoding, it should be wired directly to the lower 8 bits of the SD bus. That is, unless there is no SD bus and all ISA bound traffic makes its way to the 8 bit (X-Bus) datapath.

9. Audio Requirements Summary

9.1 Codec Audio Hardware

Function	Requirement	1996[1]
Codec Sampling Characteristics	• Software programmable sample rates of 8.0, 11.025, 16.0, 22.05, 44.1 and 48.0 Kss[2]	*R*
	• 8-bit mono, full-duplex operation, with PCM data encoding	*R*
	• 16-bit stereo, full-duplex operation, with PCM data encoding.	*R*
	• Additional software programmable sample rates of 7.2 and 9.6 Kss[2] (support for modem)	*O*
	• Input A/D sample rate and channel width (1 or 2-ch) are independent of D/A output sample rate and channel width (support for "head to head DSVD games, conferencing, & echo cancellation)	*O*
	• Record from **mic in** supports mono or the "time correlated i/o" data format described in section 4 (support for echo cancellation).	*O*
Codec PCI bus utilization	≤2% PCI bus utilization[3] for simultaneous play and record, 16-bit stereo, full-duplex 44.1Kss audio	*R*

Notes:

1. R=Required Capability, O=Optional (Recommended) Capability, NA=Not Applicable
2. KiloSamples per second. Not to be confused with analog signal bandwidth in KHz.
3. For ISA based audio systems, this requirement is placed on the audio hardware interface, ISA DMA controller and PCI-to-ISA bridge. Although < 360 KB/s is required (44100 samples per second, 16 bits per sample, stereo, record and playback), 2% PCI bus utilization is equivalent to ≤ 7 PCI clock reads and write cycles assuming single DWORD transfers to an ideal PCI slave. The recommended implementation for the audio hardware interface is demand-mode Type-F DMA transfers and a DMA controller in the PCI-to-ISA bridge that is able to buffer several ISA transfers at ISA speeds and transfer a single DWORD at PCI bus speeds.

Codec Audio Hardware (cont.)

Function	Requirement	1996[1]
Codec Inputs	• **mono in**: A mono line-level input used for telephony card output	*R*
	• **mic in**: A mono mic-level input with 20dB boost (in addition to 20dB S/W programmable gain)	*R*
	• **line in**: A stereo line-level input with programmable left and right channel gain control	*R*
	• **synth in**: A stereo line-level input with programmable left and right channel gain control which is dedicated for an external H/W MIDI upgrade to internal FM synth (mix and/or mux w/ internal synth)	*R*
	• **CD in**: A stereo line-level input with programmable left and right channel gain control which is dedicated for an internal CD-ROM drive connection	*R*
	• **aux/video in**: A stereo line-level input with programmable left and right channel gain control which is dedicated for an internal TV/VidCam connection	*O*
Codec Outputs	• **line out**: A stereo mix of all audio sources w/ line-level output and left and right channel programmable output gain control.	*R*
	• **mono out** A mono line-level mix of all audio sources used for telephony card input (may be generated internally even if not brought off chip)	*O*
Codec control inputs	• a minimum of 2 digital inputs used to control the master output volume: **up, down, & mute** **volume up** momentary on (1st) **volume down** momentary on (2nd) **mute** momentary on (both)	*O*

Notes:
1. R=Required Capability, O=Optional (Recommended) Capability, NA=Not Applicable.

Codec Audio Hardware (cont.)

Function	Requirement	1996
Codec Output Selection	• **mono in, mic in**, **line in**, **synth in**, **CD in**, **aux/video in**, and **PCM out** are ***mixable*** with each source having a persistent gain and mute setting	*R*
Codec Input Selection	• **mic in**, **line in**, **synth in**, or **mix out** sources are ***selectable via MUX*** at the input A/D, with a persistent master gain setting for each MUX option • **mic in**, **line in**, **synth in**, **CD in**, **aux/video in**, or **mix out** sources are ***selectable via MUX*** at the input A/D, with a persistent master gain setting for each MUX option • **mic in**, **line in**, **synth in**, **CD in**, **aux/video in,** and **PCM out** sources are ***mixable*** at the input A/D with each source having a persistent gain and mute setting which is independent of the output mix	*R* *R* *O*
DOS Games Compatibility	• Sound Blaster compatible registers and FM Synthesis support • Joystick and MIDI I/O	*R* *R*
Plug n Play	• PnP compliant audio add-in or motherboard Codec	*R*
Power Mgt.	• Support of low power state through software control	*R*

Notes:

1. R=Required Capability, O=Optional (Recommended) Capability, NA=Not Applicable.

9.2 System Audio Hardware

Function	Requirement	1996[1]
System Audio Inputs	• **mic in**: High impedance electret and dynamic mono-microphone input support[2]	R
	• **line in**: stereo 1 Vrms line-level input	R
	• **CD in**: (Internal) stereo 1 Vrms line-level input	R
	• **aux/video in**: (Internal) stereo 1 Vrms line-level input	O
System Audio Outputs	• **line out**: Stereo line-level output	R
	• **spk out**: Stereo amplified output that provides ≥ 1 Watt into a 4 ohm speaker	O
Audio Connectors	• **Back Panel (riser card or system board)** **Type** / **Use**	
	line in — stereo-mini[3] — external source	R
	mic in — mono-mini[3] — desktop mic	R
	line out — stereo-mini[3] — line level out	R
	spk out — stereo-mini[3] — speakers	O
	• **Front Panel (optional)** **Type** / **Use**	
	headset in — mono-mini[3] — headset mic	O
	headset out — stereo-mini[3] — headset	O
	If the Front Panel option is supported, the Back Panel **mic in** and **spkr out** output must be mechanically disabled when the corresponding front panel **headset in** or **headset out** connection is made.	
	• **Digital Volume Controls (optional)** **Type** / **Use**	
	volume up — momentary on — (1st)	O
	volume down — momentary on — (2nd)	O
	mute — momentary on — (both)	O

Notes:
1. R=Required Capability, O=Optional Capability, NA=Not Applicable
2. Nominal input voltage is 10 mV and 200 mVpp for dynamic and electret microphones respectively. Dynamic mics use mono (2 conductor) connector. Electret mics use stereo (3 conductor) connector.
3. 3.5 mm stereo jack

9.3 System Audio quality

Function	Requirement	1996[1]
Audio Quality[2]	• -3 dB Frequency Response Limits[3] 8.0 Kss[4] 50 Hz - 3.2 KHz 11.025 Kss 50 Hz - 4.4 KHz 22.05 Kss 50 Hz - 8.8 KHz 44.1 Kss 50 Hz - 17.6 KHz	*R*
	• Signal-to-Noise Ratio better than 70 dB[3]	*R*
	• Channel separation better than 70 dB[3]	*R*
	• Total harmonic distortion below -60dB (0.1%)[3]	*R*

Notes:

1. R=Required Capability, O=Optional Capability, NA=Not Applicable
2. Audio quality measurements refer to a complete signal path from the analog input connectors thru A/D back thru D/A to the output connector (via digital monitor or to disk and back).
3. Left and right channels must be tested separately. Sample rate may be 8000, 22050 or 44100 samples per second. Test waveform frequency must be between 200 and 2000 Hz for signal-to-noise, channel separation, and total harmonic distortion measurements.
4. KiloSamples per second. Not to be confused with analog signal bandwidth in KHz.

9.4 Windows Audio Drivers & API's

Function	Requirement	1996
Audio Drivers/APIs	• MMSYSTEM API support for Wave, AUX, and Mixer	*R*
	• Microsoft DirectSound	*R*
	• MIDI support	*R*
	• DOS box support under Windows	*R*
	• Full duplex simultaneous capture and playback of audio streams	*R*
	• Supports software based wavetable synthesis for MIDI	*R*
	• Device virtualization to allow multiple applications to share a single audio device which includes the support of digital mixing and sample rate conversion of audio streams	*O*
Interrupts	• Interrupts never disabled through the processor interrupt flag (IF) for longer than 5 ms.	*R*
Power Mgt.	• Support of low power state through software control using VPOWERD STANDBY events	*R*
Management	• Provide a MIF file for the audio controller and addin board that implements the DMTF Standard *ComponentID*group.	*R*
	• Provide a MIF file for the audio driver that implements the DMTF Standard *ComponentID*group	*R*
	• Implement vendor private attributes as defined in the DMI Specification	*O*

Notes:
1. R=Required Capability, O=Optional (Recommended) Capability, NA=Not Applicable