Technical notes on using Analog Devices DSPs, processors and development tools
Contact our technical support at dsp.support@analog.com and at dsptools.support@analog.com
Or visit our on-line resources http://www.analog.com/ee-notes and http://www.analog.com/processors

Programming S/PDIF on ADSP-2136x SHARC® Processors

Contributed by Aseem Vasudev Prabhugaonkar *Rev 1 – April 1, 2005*

Introduction

This application note provides a basic description of the Sony Philips Digital Interface (S/PDIF), its frame structure, and how to program S/PDIF on ADSP-21364 SHARC® processors to transmit and receive data. This application note also explains the Precision Clock Generator (PCG) setup required to provide necessary signals for an S/PDIF transmit section. Code examples are provided with this application note to demonstrate programming of S/PDIF transmit and receive sections of ADSP-21364 SHARC processor. The examples consist of code that demonstrates talk-through and data loop back performed on S/PDIF interface.

Basics of S/PDIF

S/PDIF is a standard that defines a serial interface for transferring digital audio data between various audio equipment like CD players, DVD players, and amplifiers. S/PDIF describes a serial, uni-directional, self-clocking interface for the interconnection of digital audio equipment for consumer and professional applications that use linear PCM coded audio samples. The data can be transferred between devices without having to convert it to an analog signal. This is the biggest advantage of S/PDIF. For instance, when audio is transferred from a CD player to an audio amplifier over an analog link, noise is introduced; filtering out this noise is not easy. This problem is overcome when audio data is transferred over a digital link instead of an analog link.

S/PDIF is used to transmit digital signals of many formats, the most common being the 48 kHz sample rate format used in digital audio tape (DAT) and the 44.1 kHz format used in CD audio. To support both systems, as well as others that might be needed, the format has no defined data rate. Instead, the data is sent using a bi-phase mark code, which has one or two transitions for every bit. This allows the original word clock to be extracted from the signal itself. S/PDIF is also polarity independent, which makes it a lot easier to work with.

S/PDIF was developed from the AES/EBU standard used in the professional audio field, which is commonly used in DAT systems. S/PDIF remained identical at the protocol level, but changed the physical connectors from XLR to electrical RCA jacks or optical TOSLINK, both of which cost less and are easier to use.

Essentially, S/PDIF is a consumer version of the AES/EBU format. The S/PDIF interface specification consists mainly of the hardware and software. The software usually deals with S/PDIF frame format, and the hardware interface is the actual physical connection medium used to transfer data between two devices. The various interfaces used for the physical medium could be TTL (Transistor-Transistor Logic), COAX (Coaxial cable - 75 ohm cable connected with RCA plugs), and TOSLINK - an optical fiber connection. The S/PDIF signal is independent of

polarity; hence, you do not have to worry about the absolute signal polarity.

S/PDIF Bi-Phase Encoding

S/PDIF is a single-wire serial interface, and the bit clock is embedded within the S/PDIF data stream. The bit clock and the frame sync are recovered from the data stream at the receiver end. The simple coding of digital data (i.e., `1` as logic high, and `0` as logic low) is not an ideal format, because another signal (a bit clock) is required to transmit and sample data bits in the stream. This is where the bi-phase encoding comes in – the bit clock is embedded within the transmitted stream and recovered at the receiver. Bi-phase coded PCM has a mean voltage of zero, which eliminates DC on the interface, so the resulting data can be AC-coupled through a transformer or series capacitor. Each data bit in the S/PDIF stream has a time slot that begins with a transition and ends with a transition. The second transition serves as the beginning transition for the next time slot. If the data bit is a `1`, an additional transition is made in the middle of the time slot; a data bit of `0` has no additional transition. Figure 1 illustrates bi-phase encoding. Unit interval (UI) is the shortest interval between transitions.

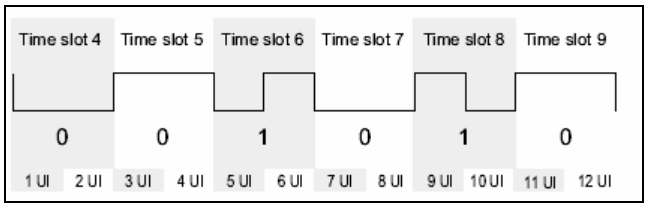

Figure 1. Bi-Phase Encoding in S/PDIF Stream

As noted in Figure 1, even if the data is a stream of `0`s, there will be transitions at each time slot, and the direction of these transitions (signal polarity) becomes irrelevant.

S/PDIF Frame Format

Each S/PDIF frame is 64 timeslots (128 UIs). Each frame consists of two sub-frames, which are 32 timeslots (64 UIs). The sub-frame comprises of preamble, 24 bits of audio data, and 4 bits that carry other information such as user data and channel status information. Figure 2 shows S/PDIF sub-frame format.

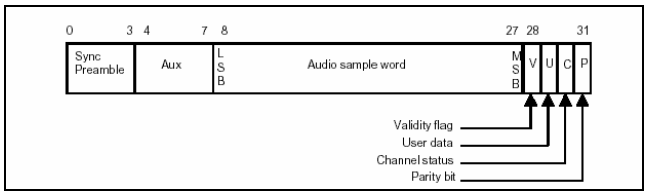

Figure 2. S/PDIF Sub-Frame Format

The LSB of the audio data is driven first. The first four time slots of a sub-frame carry a distinctive data pattern called a preamble, which is used to indicate sub-frame and block starts. There are three preambles, each of which break the bi-phase coding rule by containing one or two pulses, which have duration of 3 UIs. Breaking this rule would mean that the pattern cannot occur anywhere else in the stream.

Figure 3 illustrates S/PDIF frame and block format.

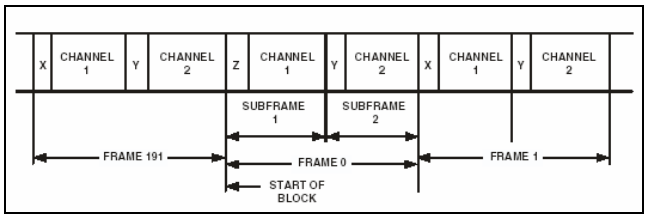

Figure 3. S/PDIF Frame and Block Format

Each sub-frame begins with a 4-bit preamble. Preamble `Z` indicates the start of a block and the start of sub-frame channel A. Preamble `X` indicates the start of a channel A sub-frame when not at the start of a block. Preamble `Y` indicates the start of a channel B sub-frame.

S/PDIF Transmitter and Receiver of ADSP-21364 SHARC Processor

The S/PDIF transmitter in SHARC processors usually receives data from a SPORT. The serial data input to the transmitter can be formatted as left justified, I2S, or right justified with word widths of 16, 18, 20, or 24 bits. The serial data, clock, and frame sync inputs to the S/PDIF

transmitter are routed through the Signal Routing Unit (SRU). These inputs, which are controlled by the SRU control registers, come from a variety of sources such as SPORTs, external pins, the precision clock generator (PCG), or the sample rate converter (SRC). The S/PDIF transmitter output may be routed to an output pin via the SRU and tied directly to the physical layer interface of an end-user product. The output is also available to the S/PDIF receiver for loop-back testing through the SRU.

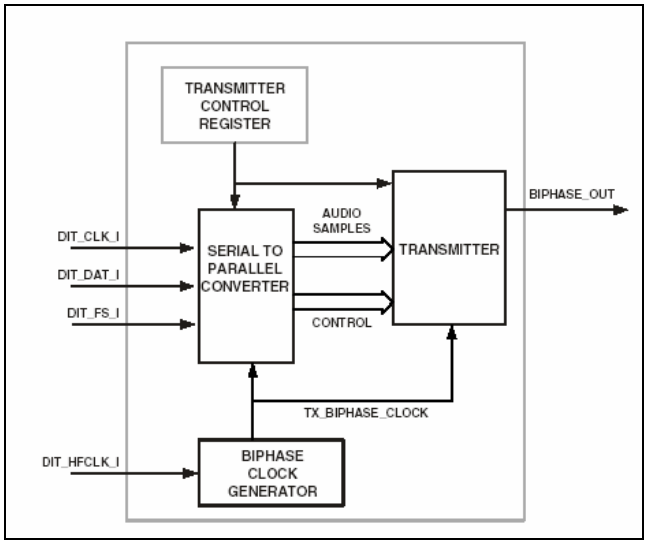

Figure 4. ADSP-21364 S/PDIF Transmitter Block Diagram

The S/PDIF receiver complies with all common serial digital audio interface standards, including IEC-60958, IEC-61937, AES3, and AES11. These protocols encompass everything from simple stereo streams to high sample rate, compressed surround sound, such as those defined by Dolby or DTS. The input to the receiver is a bi-phase encoded signal that may contain any number of audio channels (compressed or linear PCM) or non-audio data. The receiver decodes the single bi-phase encoded signal to produce serial data, `LRCLK`, and high-frequency serial clocks. The receiver derives the clock from the on-chip digital phased-locked loop (PLL). In applications that are not sensitive to moderate jitter, the on-chip PLL can provide the clock source derived from the bi-phase encoded signal.

Example: Loop Back from Transmitter to Receiver

This section discusses programming the ADSP-21364 SHARC processor's S/PDIF transmitter to receive data, bit clock, and frame sync from a SPORT to generate a bi-phase encoded data stream. The Precision Clock Generators (PCG) are used to provide an over-sampling clock, frame sync, and bit clock to the SPORT and S/PDIF TX.

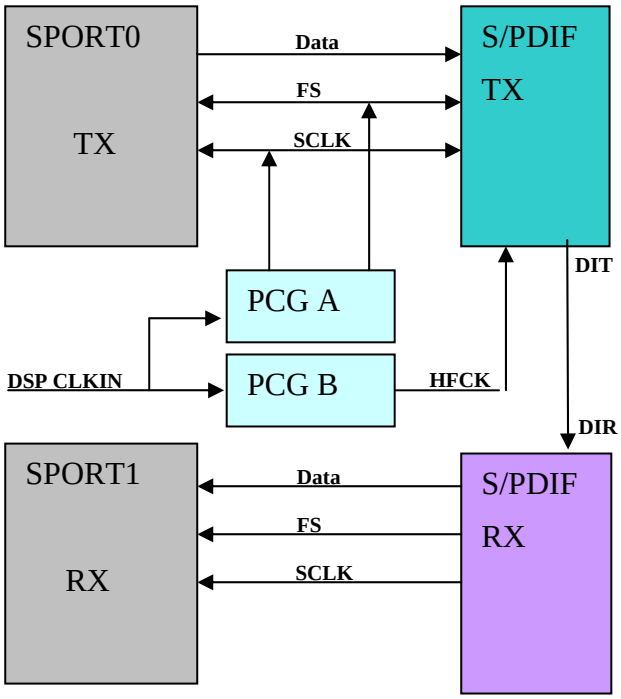

Figure 5. S/PDIF TX – RX Loop-Back Setup

`DIT` is the output of S/PDIF transmitter, and the bi-phase encoded data stream is sent over this signal. `DIT` can be routed to any of the external DAI pins. The S/PDIF receiver receives bi-phase encoded data on `DIR`, and it is an input signal to the S/PDIF receiver. `DIR` can be routed to any of DAI pins or receive data from `DIT` for internal loop back. SPORT0 is configured to transmit data in I2S format with data word length of 32 bits. SPORT0 and S/PDIF TX receive the bit clock and frame sync from PCG A. PCG B supplies an over-sampling `HFCLK` clock signal to S/PDIF TX. The transmitter is configured for an over-sampling ratio of 256 x FS. Hence, for a

specific frame sync (FS) frequency, HFCLK should be 256 x FS, and the bit-clock should be 64 x FS. The over-sampling ratio in S/PDIF transmitter is determined by the DIT_FREQ bits of the DITCTL register. The DIT_SMODE_IN bits, which determine the serial data input format, are configured for I2S format. Refer to the *ADSP-2136x SHARC Hardware Reference Manual* [1] for details on the DITCTL register bits.

PCG A provides bit clock and frame sync signals to SPORT0 and S/PDIF TX. To comply with I2S format, the PCG_CTLA1 register should also be programmed appropriately with FSAPHASE_HI bits. This value should align the rising edge of FS with the falling edge of SCLK from PCG A for I2S mode.

For programming details, control register bit settings, and so on, refer to main.c in the attached ZIP file. This file also has information on PCG, SPORT, and SRU configuration. Note that some of these signals are also brought out on DAI pins to capture them on scope.

The screen captures were taken on an oscilloscope during S/PDIF transmission.

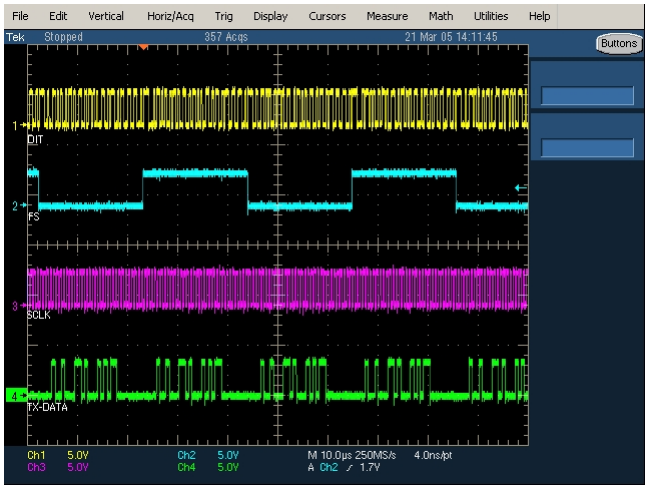

Figure 6. SPORT0 and S/PDIF TX Signals

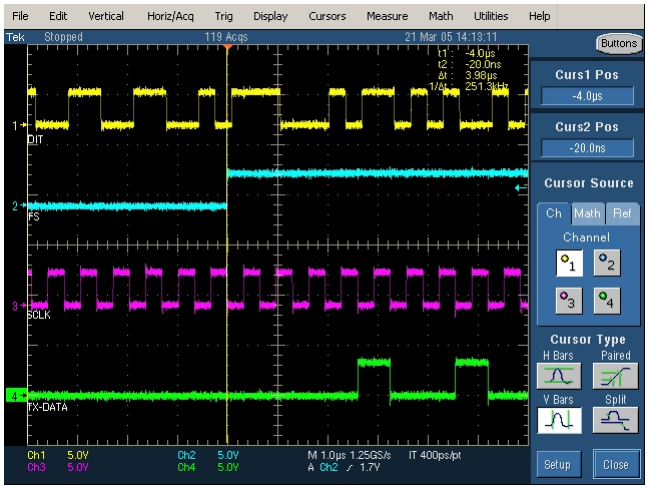

Figure 7. SCLK and FS Relationship

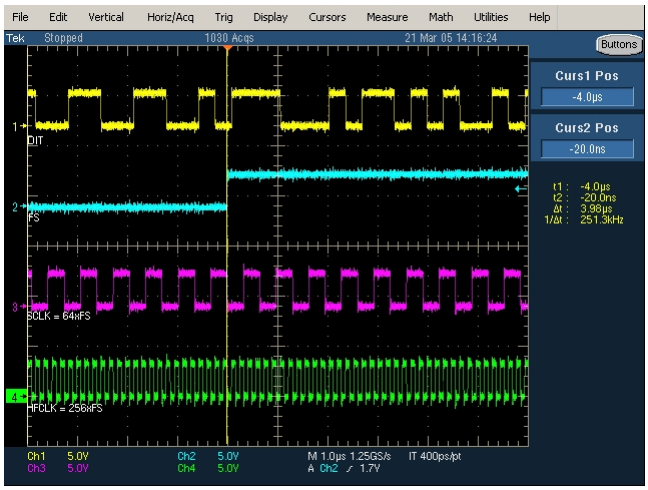

Figure 8. FS, SCLK, DIT and HFCLK of S/PDIF TX

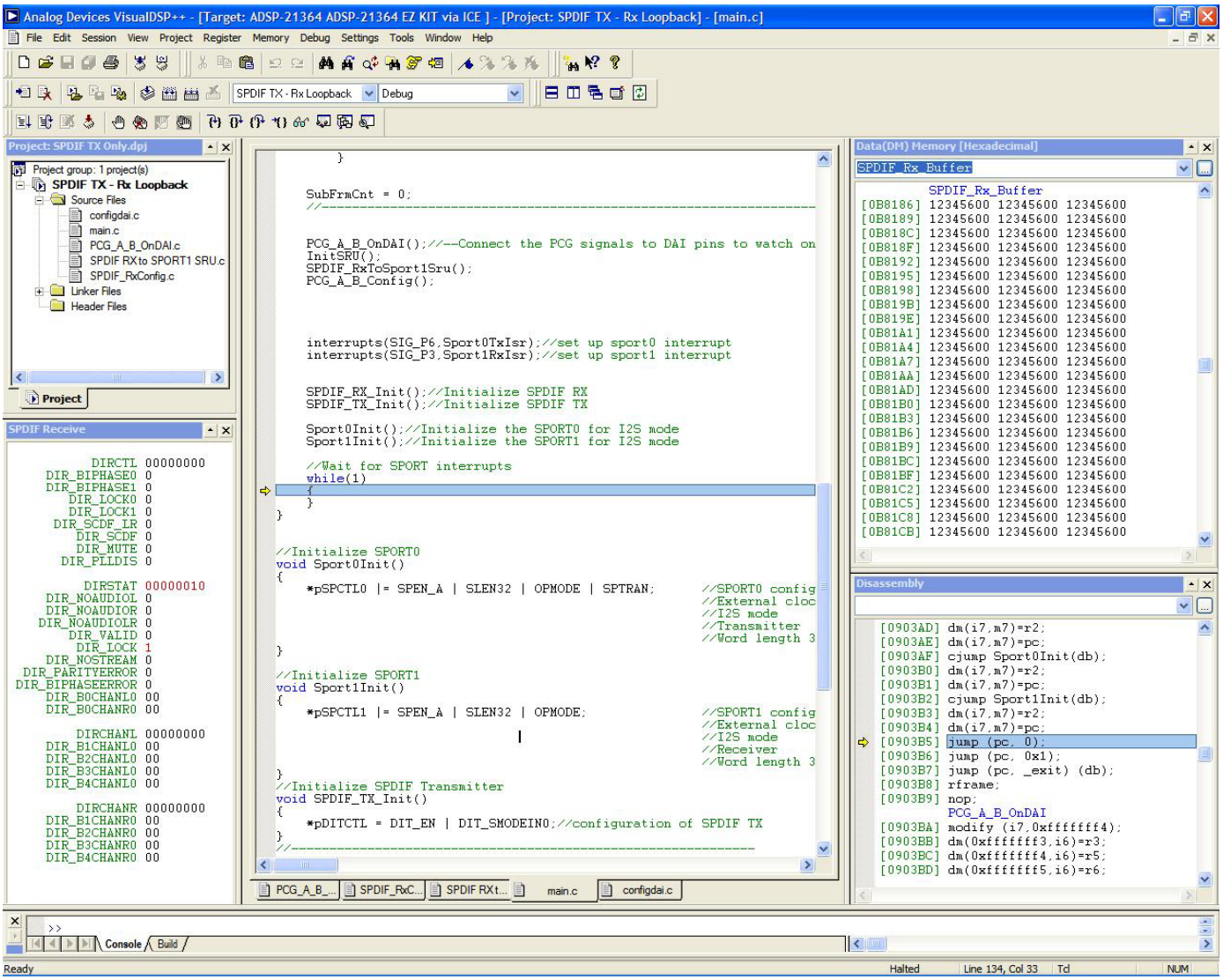

Figure 9. Viewing Received Data and S/PDIF RX Register Values in VisualDSP++

Example: Talk-Through Code

This code example demonstrates the receipt of data on S/PDIF RX from an external source such as a DVD player. The bi-phase encoded PCM stream is received by S/PDIF RX. SPORT1 is used to receive data from S/PDIF RX. SPORT0 is used to transmit this data to S/PDIF TX. S/PDIF TX encodes this data in to bi-phase stream, which is provided to AVR.

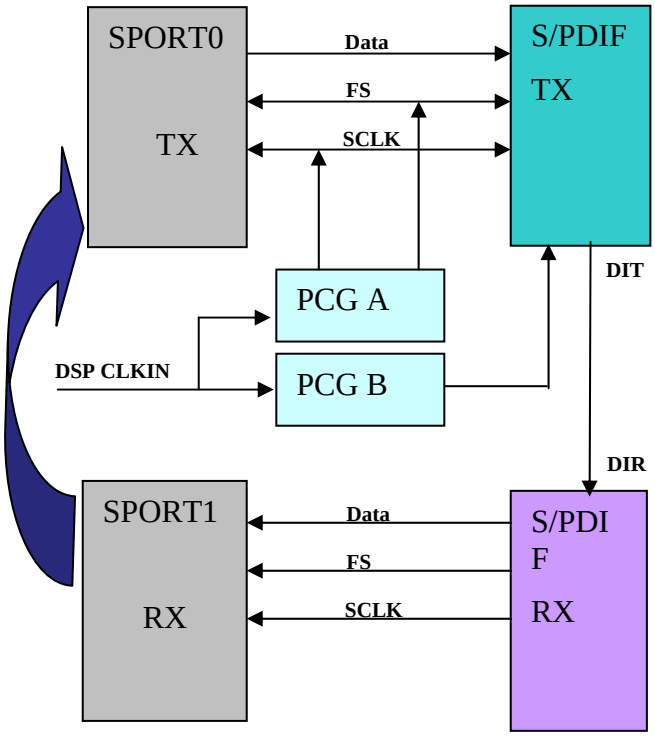

Figure 10. S/PDIF Talk-Through Setup

For programming details, control register bit settings, and so on, refer to `main.c` in the attached ZIP file. This file also has information on PCG, SPORT, and SRU configuration. Note that some of these signals are also brought out on DAI pins to capture them on an oscilloscope.

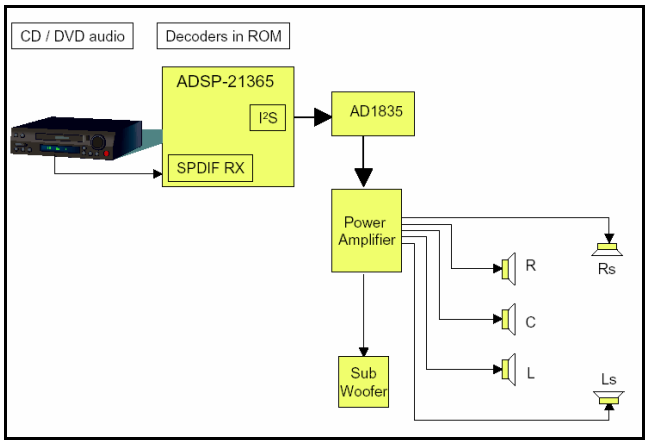

Figure 11. Possible System Configuration Using S/PDIF on ADSP-21364 Processors

The block diagram in Figure 11 shows a possible system configuration. The ADSP-2136x processor would receive the S/PDIF stream over its receiver, implement decode or post-decode algorithms on the received data, and then send the decoded audio data over serial ports in I2S or TDM mode to codec.

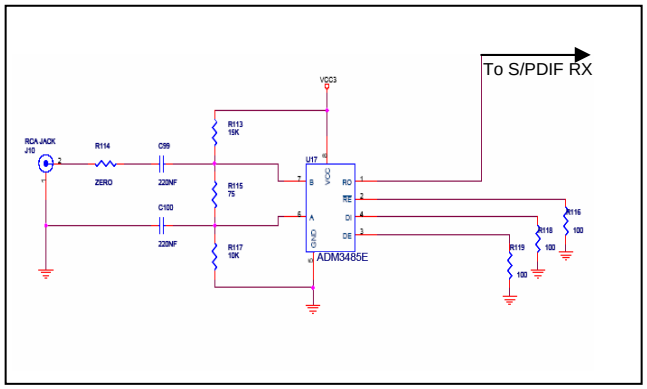

Figure 12. Suggested S/PDIF RX Input Circuitry

The circuit diagram in Figure 12 shows a suggested receive circuitry in which an ADM3485E (RS-485 transceiver) is used.

References

[1] *ADSP-2136x SHARC Processor Hardware Reference.* Revision 0.3, February 2005. Analog Devices, Inc.

[2] *ADSP-21364 Processor EZ-KIT Lite User's Manual.* Analog Devices, Inc.

[3] *TN-26 the AES3 and IEC60958 Digital Interface TECHNOTE.* Julian Dunn, Audio Precision.

[4] *FAE distance learning presentation on S/PDIF*. Richard Murphy, Analog Devices, Inc.

[5] *International Standard, IEC, 60958-1*. First edition1999-12.

Document History

Revision	Description
Rev 1 – April 1, 2005 *by Aseem Vasudev Prabhugaonkar*	Initial version