Multiple TLC320AC01/02 Analog Interface Circuits on One TMS320C5X DSP Serial Port

SLAA016
January 1996

IMPORTANT NOTICE

Texas Instruments (TI) reserves the right to make changes to its products or to discontinue any semiconductor product or service without notice, and advises its customers to obtain the latest version of relevant information to verify, before placing orders, that the information being relied on is current.

TI warrants performance of its semiconductor products and related software to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are utilized to the extent TI deems necessary to support this warranty. Specific testing of all parameters of each device is not necessarily performed, except those mandated by government requirements.

Certain applications using semiconductor products may involve potential risks of death, personal injury, or severe property or environmental damage ("Critical Applications").

TI SEMICONDUCTOR PRODUCTS ARE NOT DESIGNED, INTENDED, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT APPLICATIONS, DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS.

Inclusion of TI products in such applications is understood to be fully at the risk of the customer. Use of TI products in such applications requires the written approval of an appropriate TI officer. Questions concerning potential risk applications should be directed to TI through a local SC sales office.

In order to minimize risks associated with the customer's applications, adequate design and operating safeguards should be provided by the customer to minimize inherent or procedural hazards.

TI assumes no liability for applications assistance, customer product design, software performance, or infringement of patents or services described herein. Nor does TI warrant or represent that any license, either express or implied, is granted under any patent right, copyright, mask work right, or other intellectual property right of TI covering or relating to any combination, machine, or process in which such semiconductor products or services might be or are used.

Contents

Title *Page*

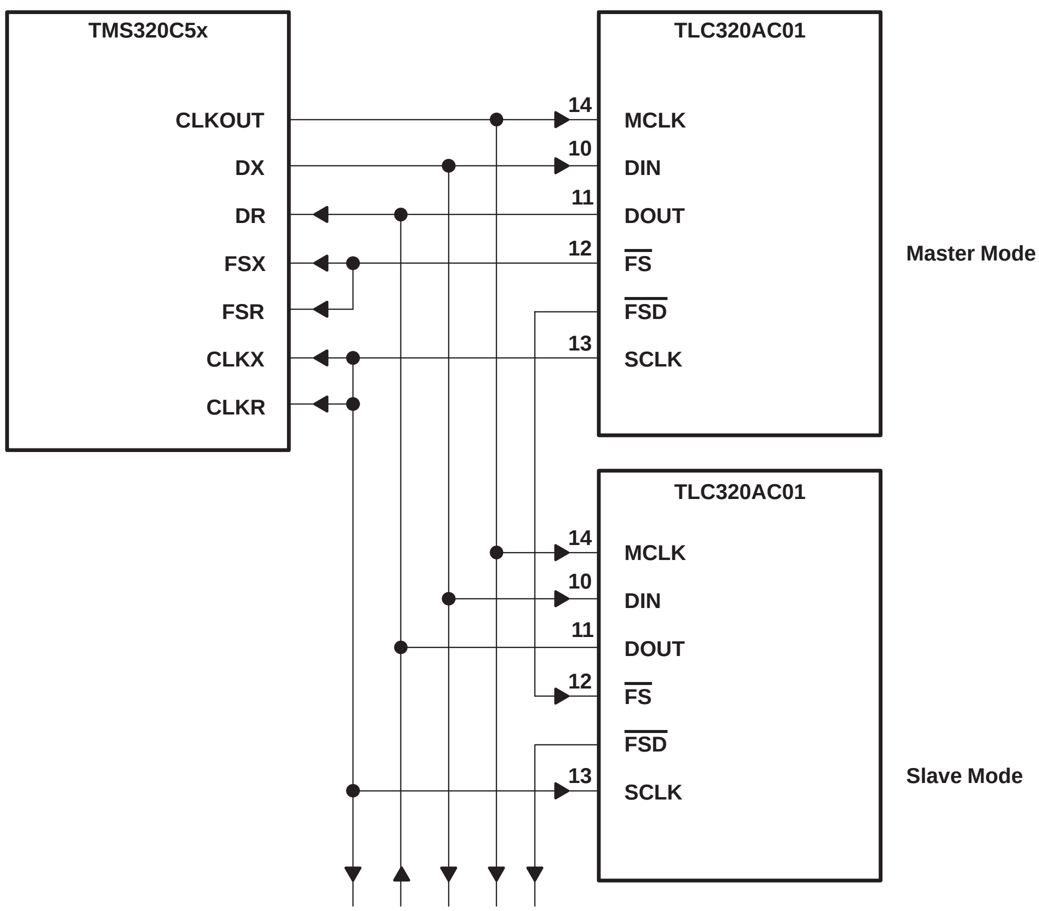
List of Illustrations

INTRODUCTION

In many applications, digital signal processors (DSPs) must obtain information from multiple analog-to-digital (A/D) channels and transmit digital data to multiple digital-to-analog (D/A) conversion channels. The problem is how to do this easily and efficiently.

This application report addresses the issue of connecting two channels of an analog interface circuit (AIC) to one TMS320C5X DSP serial port. In this application report, the AIC is the TLC320AC01.

The TLC320AC01 (and TLC320AC02) AIC contains both A/D and D/A converters and, using the master/slave mode, it is possible to connect two of these AICs to one TMS320C5X DSP serial port with no additional logic. The hardware schematic is shown in Figure 1. The data manual for the TLC320AC01 (SLAS057A) also describes the hardware connection for interfacing multiple 'AC01s to one serial port.



Note A: Terminal numbers shown are for the FN package.

Figure 1. Master AIC With a Slave AIC (to DSP Interface)

HARDWARE AND SOFTWARE SOLUTION

Once the hardware connections are completed, the issue becomes how to distinguish one channel from another. Fortunately, this is very easy to do in software and adds very little overhead. The mode that the ’AC01s run in is called master/slave mode. One TLC320AC01 is the master and all of the rest of the ’AC01s are slaves. The master can be distinguished from all of the slaves by examining the least significant bit (LSB) in the receive word coming from the ’AC01. The master has a 0 in the LSB and all of the slaves have a 1 in the LSB.

The ’AC01s in master/slave mode take turns communicating with the DSP serial port. The devices do this in a round robin or circular fashion. Synchronizing the system involves looking for the master ’AC01 and then starting the software associated with the first ’AC01. All other ’AC01s follow in order. It is possible to have different software for each ’AC01.

A reference design was constructed using a TMS320C5X DSP starter kit (DSK). The ’AC01s were connected to the time-division multiplexer (TDM) serial port which is available at the headers on the edge of the DSK.

A listing of the DSK assembly code for a simple stereo input/output program is included in the Software Module section.

SOFTWARE MODULE

```
*******************************************************************************
*                                                                             *
*   MODULE NAME: INOUTB.ASM                                                   *
*                                                                             *
*   In-out routine for C5X DSK with two TLC320AC01s on the                    *
*   TDM serial port of the C5X in master/slave mode.                          *
*                                                                             *
*   This version performs the in/out task for both the master                 *
*   and slave TLC320AC01 in the receive interrupt service                     *
*   routine.                                                                  *
*                                                                             *
*******************************************************************************
*               .mmregs
                .ds             01000h
PR1             .word           0104h           ;A register
PR2             .word           0219h           ;B register
PR3             .word           0300h           ;A prime register
PR4             .word           0405h           ;amplifier gain register
PR5             .word           0501h           ;analog configuration register
PR6             .word           0600h           ;digital configuration register
PR7             .word           0730h           ;frame synch delay register
PR8             .word           0802h           ;frame synch number register
value           .word           0800h
value2          .word           0800h
val_add         .word           0200h
val_add2        .word           0400h
*******************************************************************************
*                                                                             *
*   Set up the ISR vector                                                     *
*                                                                             *
*******************************************************************************
                .ps             080ah
rint:           B               RECEIVE         ; 0A; Serial port receive interrupt RINT.
xint:           B               TRANSMIT        ; 0C; Serial port transmit interrupt XINT.
trint:          B               TDMREC
txint:          B               TDMTX
                ;———————————————————————————————
*
*******************************************************************************
*                                                                             *
*   TMS320C5X INITIALIZATION                                                  *
*                                                                             *
*******************************************************************************
                .ps 0a00h
                .entry
START:          SETC            INTM            ; Disable interrupts
                LDP             #0              ; Set data page pointer
                OPL             #0834h,PMST
                LACC            #0
                SAMM            CWSR
                SAMM            PDWSR
                splk            #00c8h
                SPLK            082h,IMR
                call            AC01INIT
                CLRC            OVM             ; OVM = 0
                SPM             0               ; PM = 0
                SPLK            #042h,IMR       ; TDMA ser port rec interrupt
```

```
            SPLK        #0C8h,TSPC    ;
            CLRC        INTM          ; enable interrupts

loop                                  ; main program here does nothing.
            nop                       ; a user program can be inserted.
            b           loop          ;
;— — — — — — — — — — end of main program — — — — — — — — — — ;
;
; TDM serial port receiver interrupt service routine
;
TDMREC:
                                      ; This loop insures that the master 'AC01
            ldp         #trcv         ; is the first one that is written to in the
            bit         trcv,15       ; loop. the slave 'AC01(s) will follow in
            bcnd        xxx,tc        ; sequential order. The master 'AC01 has a
                                      ; 0 in the LSB. the slave 'AC01(s) have a 1
                                      ; in the LSB of the receive word.
            ldp         #trcv
            lacc        trcv
            and         #0fffch
                                      ;
                                      ; user code would go here for master 'AC01
                                      ;
            sacl        tdxr
            b           yyy
xxx
            ldp         #trcv
            lacc        trcv
            and         #0fffch
                                      ;
                                      ; user code would go here for slave 'AC01
                                      ;
            sacl        tdxr
yyy
            rete
;
; TDM serial port transmit interrupt service routine
;
TDMTX:
            rete
;
; RECEIVER INTERRUPT SERVICE ROUTINE
;
RECEIVE:
            rete
TRANSMIT:
            RETE
```

```
AC01INIT
                SPLK            #020h,TCR
                SPLK            #01h,PRD
                MAR             *,AR0
                LACC            #0008h
                SACL            TSPC
                LACC            #00c8h
                SACL            TSPC
                SETC            SXM
;------------------------------------------------------------------
                LDP             #PR1
                LACC            PR1
                CALL            AC01_2ND
;------------------------------------------------------------------
                LDP             #PR2
                LACC            PR2
                CALL            AC01_2ND
;------------------------------------------------------------------
                LDP             #PR8
                LACC            PR8
                CALL            AC01_2ND
;------------------------------------------------------------------
                LDP             #PR7
                LACC            PR7
                CALL            AC01_2ND

                ret

AC01_2ND:
                LDP             #0
                SACH            TDXR            ;
                CLRC            INTM
                IDLE
                ADD             #6h, 15         ; 0000 0000 0000 0011 XXXX XXXX XXXX XXXX b
                SACH            TDXR            ;
                IDLE
                SACL            TDXR            ;
                IDLE
                LACL            #0              ;
                SACL            TDXR            ; make sure the word got sent
                IDLE
                SETC            INTM            ;
                RET
```

TLC320AC01/02 MASTER WITH SLAVE OPERATIONAL OVERVIEW

The master AIC with slave AIC operation is summarized in the following sections.

TLC320AC01 AIC Master-Slave Summary

After the initial setup of the devices and the master and slave frame syncs are separated, as secondary communication is needed for a slave device, an 11 must be placed in the two LSBs of each primary data word for all master and slave devices in the system by the host processor. Therefore, all AICs must receive secondary frame requests.

The host processor must issue the command by setting D01 and D00 to a 1 in the primary frame sync data word of all devices. Then the master AIC generates the master primary frame sync interval and, after the number of shift clocks specified in the frame sync delay (FSD) register have occurred, the slave primary frame sync intervals are generated. Then, after (B register value/2) FCLK periods, the master secondary frame sync occurs first, and then the slave secondary frame sync occurs. These are then rippled through the slave devices.

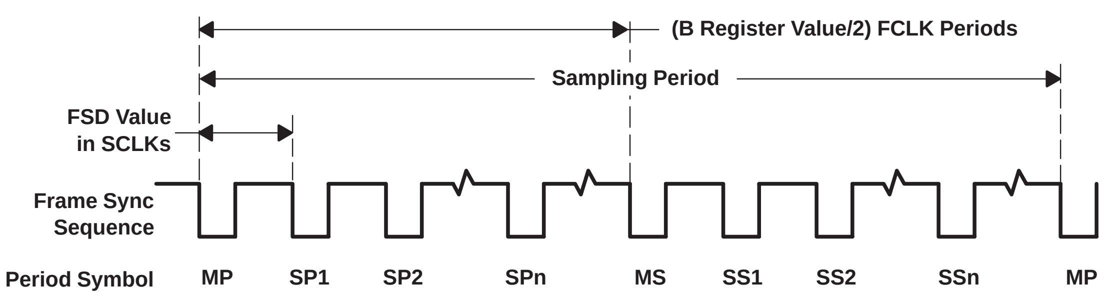
In other words, when a secondary communications interval is requested by the host processor as described above:

1. The master AIC outputs the master primary frame sync interval, and then it outputs the slave primary frame sync intervals after the FSD register value number of shift clocks has occurred.
2. After (B register value/2) FCLK periods, the master outputs the master secondary frame sync interval, and after the FSD register value number of shift clocks, the master outputs the slave secondary frame sync intervals.

This sequence is shown in Figure 2.

The host must keep track of whether the master AIC or a slave AIC is then being addressed, and also track the number of slave devices. The master always outputs a 00 in the last two bits of the DOUT word, and a slave always outputs a 1 in the LSB of the DOUT word. This information allows the system to recognize a starting point by interrogating the LSB of the DOUT word. When the LSB of the DOUT word for a device is 0, then that device is the master, and the system is at the starting point.

NOTE: This identification always happens except in 16-bit mode when the two LSBs are not available for identification purposes.



Each period must be a minimum of 16 SCLKs plus 2 additional SCLKs for the periods shown.

MP	= Master Primary Period	MS	= Master Secondary Period
SP1	= First Slave Primary Period	SS1	= First Slave Secondary Period
SP2	= Second Slave Primary Period	SS2	= Second Slave Secondary Period
SPn	= nth Slave Primary Period	SSn	= nth Slave Secondary Period

Figure 2. Basic Timing Sequence

Notes on TLC320AC01/02 AIC Master-Slave Operation

The details of master/slave operation is summarized in the following notes:

1. The slave devices can be programmed independently of the master device as long as the clock divide register numbers are not changed. For example, the gain settings can be changed independently.
2. To independently program a slave device, a secondary communication of the master and all slaves is requested, and then the delayed frame sync is rippled to the desired slave device to be programmed.
3. Secondary frame syncs must be requested for all devices in the system or none of them. This is required so that the master AIC generates secondary frames for the slave AICs and allows the slave AICs to know that the second frame syncs they receive are secondary frame syncs. Each device in the system must receive a secondary frame request in its corresponding primary frame sync period (11 in the last two LSBs).
4. Calculation of the sampling frequency in terms of the master clock, the shift clock, and the respective register ratios is:

$$\text{Sampling frequency} = f_S = \frac{\text{FCLK}}{\text{B register value}}$$

$$= \frac{f(\text{MCLK})}{2\ (\text{A register value}) \times (\text{B register value})} \tag{1}$$

Therefore,

$$\frac{f(\text{MCLK})}{f_S} = 2 \times (\text{A register value}) \times (\text{B register value}) \tag{2}$$

and in terms of the shift clock frequency, since

$$f(\text{MCLK}) = 4 \times f(\text{SCLK})$$

then

$$\frac{f(\text{SCLK})}{f_S} = \frac{(\text{A register value}) \times (\text{B register value})}{2}$$

$$= \frac{\text{Number of SCLK periods}}{\text{Sampling period}} \tag{3}$$

5. There is a minimum number of 18 shift clocks between falling edges of any two frame syncs because the frame sync delay register minimum number is 18.

   When a secondary communication is requested by the host, the master secondary frame sync begins at the middle of the sampling period followed by the slave secondary frame syncs. All master and slave primary frame sync intervals must occur within one-half of the sampling time.

   Therefore, the first secondary frame sync falling edge occurs at the following time:

$$\text{Time to first secondary frame sync} = \frac{\text{B register value}}{2}\ (\text{FCLK periods}) =$$

$$\text{A register value} \times \text{B register value (number of MCLK periods)} =$$

$$\frac{\text{A register value} \times \text{B register value}}{4}\ (\text{number of SCLK periods}) \tag{4}$$

6. The number of frame sync intervals is determined using equation 4.

   All master and slave primary frame sync intervals must occur within the time calculated in equation 4.

   Since 18 shift clocks are required for each frame sync interval, so the number of frame sync intervals is calculated by using equation 4 is:

$$\text{Number of frame sync intervals} = \frac{\text{A register value} \times \text{B register value}}{4 \times 18 \text{ (SCLKs/frame sync interval)}}$$

$$= \frac{\text{A register value} \times \text{B register value}}{72} \qquad (5)$$

7. The number of master and slave devices that can be used is defined in terms of f(MCLK) and f_S.

   Substituting the value from equation 2 for the A × B register value product gives the total number of devices, including the master and all slaves that can be used, for a given master clock and sampling frequency. Therefore, the number of devices using equation 5 is:

$$\text{Number of devices} = \frac{f(\text{MCLK})}{144 \times f_S} \qquad (6)$$

8. The number of master and slave devices that can be used if the slave devices are reprogrammed is determined by using equation 7.

   Equation 6 does not include reprogramming the slave devices after the frame sync delay occurs. So, when programming is required after shifting the slave frame syncs by the FSD register value, the total number of devices is calculated by:

$$\text{Number of devices} = \frac{f(\text{MCLK})}{288 \times f_S} \qquad (7)$$

9. The following example indicates the maximum number of devices that can be used when the slave devices are reprogrammed and the following values are assumed:

$$f(\text{MCLK}) = 10.368 \text{ MHz}, \; f_S = 8 \text{ kHz}$$

   then from equation 7,

$$\text{Maximum number of devices} = \frac{10.368 \text{ MHz}}{288 \text{ (8 kHz)}} = 4.5$$

   therefore, one master and three slaves can be used.