Application Report
SPRA527

Interfacing TMS320C54x DSPs to TLC320AC01/02 Analog Interface Circuits

Iain Hunter *C5000 DSP Field Applications Engineer*

Abstract

This application report describes how you can gluelessly interface the Texas Instruments (TI™) TLC320AC01/02 Analog Interface Circuits (AICs) to the synchronous serial port (SSP), buffered serial port (BSP), and time-division multiplexed serial port (TDM) on TI's TMS320C54x family of digital signal processors (DSPs). This application report describes how to gluelessly interface the 'C54x DSPs to the 'AC01/02 AICs in Stand-alone mode and provides a full C code listing of the software to implement a basic echo function.

Contents

Serial Ports on the TMS320C54x DSPs....2
Using TLC320AC01 in Stand-alone Mode....3
Audio Loopback Example Using TLC320AC01....4

Figures

Figure 1. Glueless Interface between TMS320C54x and TLC320AC01....2
Figure 2. TLC320AC01 Frame Sync and Clocking....3

Examples

Example 1. Audio Loop Back Code....4
Example 2. Interrupt Vector Table Code....6
Example 3. TMS320C541 Linker File Code....7

TI is trademark of Texas Instruments Incorporated.

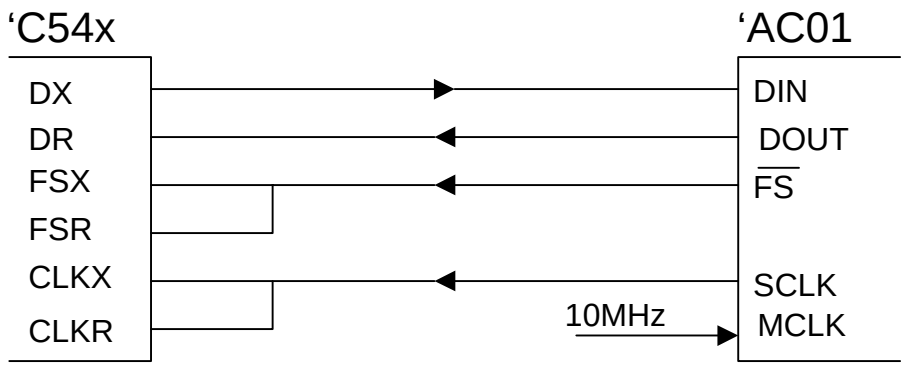
Serial Ports on the TMS320C54x DSPs

The ‘AC01/02 AICs can be gluelessly interfaced to the synchronous serial port (SSP), buffered serial port (BSP), or time-division multiplexed serial port (TDM) on a ‘C54x DSP, as shown in Figure 1. All of these serial ports contain the basic functionality of the SSP; therefore, in this application report they all will be referred to as SSP. This application report does not cover the more powerful multi-channel BSP (McBSP) in other ‘C54x devices, which also provides a glueless interface. Similarly, in this application report, the ‘AC01/02 AICs will be referred to as ‘AC01. From the DSP point of view, these two AICs are identical; the only differences are in their analog characteristics.

The ‘AC01 can be configured to operate in two main modes, Stand-alone and Codec. In Stand-alone mode, the ‘AC01 generates the clock and frame synchronization (FS) signals; in Codec mode, the SSP generates the clock and FS signals. From a system perspective, the Stand-alone mode is useful when the DSP’s algorithm timing is based on the codec’s sample rate, as given by the SSP’s receive and transmit interrupts. In Codec mode, the DSP controls the rate at which samples are transferred to and from the ‘AC01 by generating the FS pulses. Since the SSP on the ‘C54x cannot generate its own FS signal by dividing down the serial port clock, the FS rate must be controlled by DSP timer interrupts. In this case, the timer controls the rate at which samples are transferred to and from the codec. This application report will only consider the use of Stand-alone mode and SSP interrupts.

The ‘AC01 is clocked by MCLK which has a maximum frequency of 10 MHz. The AIC’s sampling rate and internal filtering are based around divide-downs of the MCLK signal using its A and B register values (see ‘AC01 Data Manual for more details). The DSP’s CLKX/R signals are tied together and FSX/R signals are tied together to ensure that the transmit and receive sides of the SSP are using the same clock and FS signals.

Figure 1. Glueless Interface between TMS320C54x and TLC320AC01



Using TLC320AC01 in Stand-alone Mode

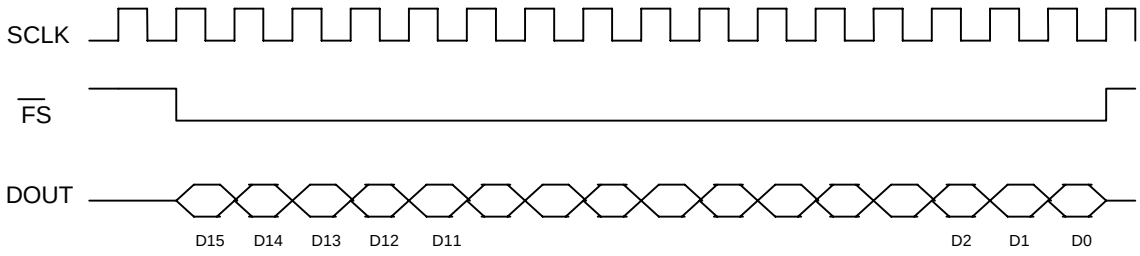
In Stand-alone mode, the AIC generates its own clock and FS signals. This is useful when the DSP's system timing is controlled by the sampling rate of the AIC. To support this mode, the SSP is configured to expect external clock and FS signals. In order to put the 'AC01 into Stand-alone mode, the M/~S pin must be pulled high.

When comparing the data sheets for the 'C54x and 'AC01, an obvious difference is that the FS signal on the DSP is active high, whereas the 'AC01 uses an active-low ~FS signal. However, both devices have different definitions of FS. The DSP expects the FS signal to be a one cycle active-high pulse the cycle before transmission starts; the 'AC01 defines the FS signal as an active-low pulse that lasts for the 16 clock cycles of the transmission. Therefore, both devices actually mark the start of the frame on the negative-going edge of the ~FS signal, so a direct pin-to-pin connection is all that is required (Figure 1). In Stand-alone mode, the 'AC01 generates an active-low ~FS signal for the entire 16 SCLK cycles that data is transmitted, as shown in Figure 2. Both the 'AC01 and the SSP transmit data on the rising edge of SCLK and receive data on the falling edge of SCLK. The SCLK is generated internally on the 'AC01 by dividing down MCLK by a factor of 4.

The SSP is double buffered with a memory-mapped data register and shift register on both the receive and transmit sides. The operation of the double buffering will be discussed using the transmit side of the SSP. The software writes a word to the data transmit register (DXR) and this word is then automatically transferred into the transmit shift register (XSR) when it is empty. When the SSP detects a falling edge on the FS signal, the SSP starts shifting the word out of the XSR, bit by bit, on the DX pin. The receive side is similar with a data receive register (DRR) and a receive shift register (RSR). The contents of RSR are shifted into DRR as soon as a word has been received. It is these transfers between the data registers and the shift registers that trigger the generation of the SSP transmit interrupt (XINT) and receive interrupt (RINT). Associated with these two interrupts are the SSP flags, transmit ready (XRDY) and receive ready (RRDY) that can be polled.

The SSP needs to be configured to run in burst mode (FSM = 1), with external frame synchronization signals (TXM = 0) and external clocks (MCM = 0).

Figure 2. TLC320AC01 Frame Sync and Clocking



Audio Loopback Example Using TLC320AC01

Example 1 shows a C program that implements a basic echo function using the ‘AC01 in Stand-alone mode. This code runs on the SSP and was developed using SSP1 on the ‘C541 EVM. In this system, all the processing of the samples would take place in the RINT interrupt service routine. During the initialization of the ‘AC01, each word is transmitted by writing it to the DXR and then XRDY is polled to detect the transfer from the DXR to XSR. When XRDY is set, a new word can be written into DXR. The internal registers of the ‘AC01 are initialized using a series of primary and secondary transmissions. When the ‘AC01 receives a word (primary transmission) with the 2 LSBs set, the next word received is a secondary transmission that should be used to program an internal register. The file cvectors.asm shown in Example 2 contains the interrupt vector table and Example 3 contains the linker file for the ‘C541. The file c54xregs.h defines all the memory-mapped registers in the ‘C54x available from the TI FTP site.

Example 1. Audio Loop Back Code

```
/* This program sets up an audio loop back using the     */
/* AC01/2 codec to run on the Synchronous Serial Port on */
/* the C54x family.                                      */
/* This version runs the AC01 in Stand-alone Mode        */

#include "c54xregs.h"

/* Define constants for AC01 setup with MCLK = 10MHz  */
#define            A  12 /* A = 12, => FCLK = 432kHz  */
#define            B  18 /* B = 18, => Fs   = 24kHz   */

/* Declare all Function Prototypes */
void inline disable(void);
void inline enable(void);
void inline idle(void);
void interrupt essp_tx(void);
void interrupt essp_rx(void);
void init_core(void);
void init_codec(void);


/* This defines address 14 in IO map for EVM reset */
volatile ioport unsigned port14;

/* Main Program loop */

main() {
   init_core();
   init_codec();

  /* Enable Interrupt Mask      */
  enable();

  /* Idle until next interrupt  */
  while(1);
}
```

```
/* Initialize core*/
void init_core()
{  /* 0 wait states for external memory  and 2 for IO    */
   SWWSR=0x2000;

   /* Set OVLY bit for onchip RAM in prog space*/
   PMST |= 0x0020;

   IFR=0xFF;       /* Clear all Int Flags      */
}


/* Initialize codec (AC01/2) */
void init_codec() {
   disable();         /* Disable all interrupts */

   /* Put AIC in reset on EVM , bit 15 of 0x14@io  */
   port14=0x0;

   /* Initialize SSP 1 as follows        */
   /* FSM=1,       Burst Mode            */
   /* TXM=0,       External Frame Sync   */
   /* MCM=0,       External Clock Source */
   /* FO=0,        16 bit mode           */
   SPC1 = 0x4008;
   SPC1 = 0x40C8;

   /* Pull AC01 out of reset on EVM board   */
   port14=0x8000;

   /* Program the AC01 using secondary comms to   */
   /* give 24kHz sampling, based on 10MHz MCLK       */
   /* Set A for FCLK freq                            */
   DXR1  = 0x0003;         /* Request secondary comm  */
   while(!(SPC1 & 0x800)); /* Loop until word moved   */
                           /* to XSR,  XRDY==1        */
   DXR1  = 0x0100 | A;
   while(!(SPC1 & 0x800));

   /* Set B for Fs */
   DXR1  = 0x0003;
   while(!(SPC1 & 0x800));
   DXR1  = 0x0200 | B;
   while(!(SPC1 & 0x800));

   IMR = 0x40;        /* Enable RINT1          */
   IFR=0x40;          /* Clear all ints        */
   enable();          /* Enable all interrupts */

   /* Send a dummy value to start things off   */
   DXR1=DRR1;

}
```

```
/* Disable all interrupts   */
void inline disable() {
   asm(" ssbx INTM");
}

/* Enable all interrupts    */
void inline enable() {
   asm(" rsbx INTM");
}

/* Idle until interrupt     */
void inline idle() {
   asm(" IDLE 1");
}

/* Codec transmit interrupt */
void interrupt essp_tx() {
}

/* Codec receive interrupt  */
void interrupt essp_rx() {

   int sample_in;

   /* Read in sample */
   sample_in=DRR1;
   /* Retransmit it ensuring that no secondary requested */
   DXR1=(sample_in & 0xFFFC);
}
```

Example 2. Interrupt Vector Table Code

```
* File containing the vector table for the  */
* AC01 <==> C54x interface program       */
   .title          "cvectors.asm"
   .ref            _c_int00,_essp_tx,_essp_rx
   .sect           ".vectors"
reset:             b  _c_int00
                   nop
                   nop

                   .space 4*16*21  ; Fill next 21 vector locations with
zeros
rint1:             b  _essp_rx
                   nop
                   nop
xint1:             b  _essp_tx
                   nop
                   nop
```

Example 3. TMS320C541 Linker File Code

```
MEMORY
                    {
PAGE 0:        /* Pgm.space */
 PROG    : origin = 0x2000, length = 0x2000  /* Ext.Pgm.area   */
 VECS    : origin = 0xff80, length = 0x7f    /* Vectors        */
PAGE 1:        /* Data space */
 REGS    : origin = 0x0000, length = 0x0060  /* MMR's          */
 RAM0    : origin = 0x0060, length = 0x20
 RAM     : origin = 0x0080, length = 0x1380  /* DARAM          */
                    }


SECTIONS
{
.vectors  :{}  > VECS    PAGE 0  /*  Vectors at 0xFF80         */
.text     :{}  > PROG    PAGE 0
.cinit    :{}  > PROG    PAGE 0
.bss      :{}  > RAM     PAGE 1
.const    :{}  > RAM     PAGE 1
.data     :{}  > RAM     PAGE 1
.stack    :{}  > RAM     PAGE 1
}
```

TI Contact Numbers

INTERNET

TI Semiconductor Home Page
www.ti.com/sc

TI Distributors
www.ti.com/sc/docs/distmenu.htm

PRODUCT INFORMATION CENTERS

Americas
Phone +1(972) 644-5580
Fax +1(972) 480-7800
Email sc-infomaster@ti.com

Europe, Middle East, and Africa
Phone
Deutsch +49-(0) 8161 80 3311
English +44-(0) 1604 66 3399
Español +34-(0) 90 23 54 0 28
Francais +33-(0) 1-30 70 11 64
Italiano +33-(0) 1-30 70 11 67
Fax +44-(0) 1604 66 33 34
Email epic@ti.com

Japan
Phone
International +81-3-3457-0972
Domestic 0120-81-0026
Fax
International +81-3-3457-1259
Domestic 0120-81-0036
Email pic-japan@ti.com

Asia
Phone
International +886-2-23786800
Domestic
Australia 1-800-881-011
TI Number -800-800-1450
China 10810
TI Number -800-800-1450
Hong Kong 800-96-1111
TI Number -800-800-1450
India 000-117
TI Number -800-800-1450
Indonesia 001-801-10
TI Number -800-800-1450
Korea 080-551-2804
Malaysia 1-800-800-011
TI Number -800-800-1450
New Zealand 000-911
TI Number -800-800-1450
Philippines 105-11
TI Number -800-800-1450
Singapore 800-0111-111
TI Number -800-800-1450
Taiwan 080-006800
Thailand 0019-991-1111
TI Number -800-800-1450
Fax 886-2-2378-6808
Email tiasia@ti.com

IMPORTANT NOTICE

Texas Instruments and its subsidiaries (TI) reserve the right to make changes to their products or to discontinue any product or service without notice, and advise customers to obtain the latest version of relevant information to verify, before placing orders, that information being relied on is current and complete. All products are sold subject to the terms and conditions of sale supplied at the time of order acknowledgement, including those pertaining to warranty, patent infringement, and limitation of liability.

TI warrants performance of its semiconductor products 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, AUTHORIZED, OR WARRANTED TO BE SUITABLE FOR USE IN LIFE-SUPPORT DEVICES OR SYSTEMS OR OTHER CRITICAL APPLICATIONS. INCLUSION OF TI PRODUCTS IN SUCH APPLICATIONS IS UNDERSTOOD TO BE FULLY AT THE CUSTOMER'S RISK.

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

TI assumes no liability for applications assistance or customer product design. TI does not 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. TI's publication of information regarding any third party's products or services does not constitute TI's approval, warranty, or endorsement thereof.

Copyright © 1999 Texas Instruments Incorporated