TMS320 DSP
DESIGNER'S NOTEBOOK

Initializing the TMS320C5x DSK Board

APPLICATION BRIEF: SPRA253

Gerald Capwell
Digital Signal Processing Products
Semiconductor Group

Texas Instruments
March 1995

Texas Instruments

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 application 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.

TRADEMARKS

TI is a trademark of Texas Instruments Incorporated.

Other brands and names are the property of their respective owners.

CONTACT INFORMATION

US TMS320 HOTLINE	(281) 274-2320
US TMS320 FAX	(281) 274-2324
US TMS320 BBS	(281) 274-2323
US TMS320 email	dsph@ti.com

Contents

Figures

Tables

Initializing the TMS320C5x DSK Board

Abstract

The most important task you must perform before starting ANY application is DSK initialization. Initialization is done in software preceding the entry to your application and must be done if the Analog Interface Circuit (AIC) is to be used.

If this initialization is not done, your DSK applications will not run properly. At other times the application seems to run, but nothing will happen. For example, the TRY1.ASM code in the Users Guide does not work. You are seeing the effects of an uninitialized DSK board.

Three things must be initialized:

1) The TMS320C50 on-chip timer,

2) The TMS320C5x serial port,

3) The Analog Interface Circuit (AIC).

This document discusses the details needed to initialize the DSK properly for your configuration.

Design Problem

In some instances, my DSK applications do not run properly. At other times the application seems to run, but nothing happens. For example, the TRY1.ASM code in the Users Guide does not work.

Solution

You are seeing the effects of an uninitialized DSK board. The TRY1 example works correctly only if you initialize the DSK before entering the TRY1 routine. In fact, the most important task you must perform before starting ANY application is DSK initialization. Initialization is done in software preceding the entry to your application and must be done only if the Analog Interface Circuit (AIC) is to be used.

Three things must be initialized:

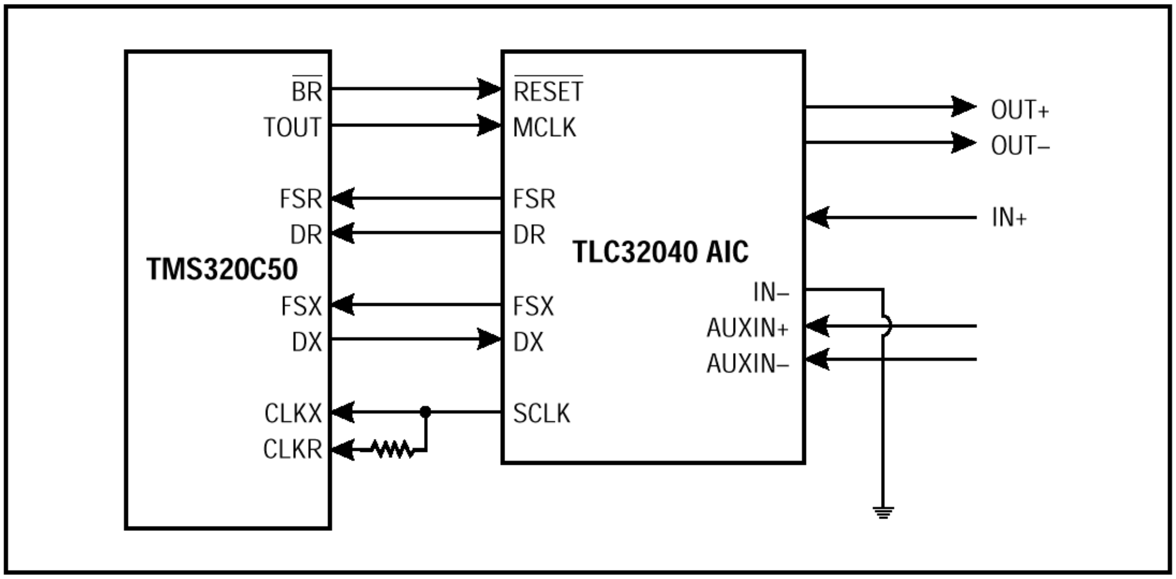
1) The TMS320C50 on-chip timer,

2) The TMS320C5x serial port,

3) The Analog Interface Circuit (AIC).

The AIC is the device that interfaces the outside analog world to the DSK's internal digital world. The AIC is connected to the DSP through the TMS320C50 serial port as shown below in Figure 1.

Figure 1. Hardware Connection Between the TMS320C50 and AIC



In order to communicate with (initialize) the AIC, the programmer must first initialize the DSP on-chip timer to provide the AIC with its master clock and secondly, initialize the serial port. Initialization only has to be performed once each time the DSK is powered up. This explains why TRY1.ASM does not work as the first application after power up, but does work after you have executed FUNC.DSK or some other demo that has the initialization routine included.

The easiest way to include the initialization routine is to extract the AICINIT subroutine from a DSK demo (FUNC.ASM, for example) append it to your application and CALL the subroutine. Be sure to include the constants TA, RA, TB, RB, and AIC_CTR. These constants are used to set the AIC sampling frequency, filtering, etc. These will be explained later. If you wish to create your own initialization routine, continue reading and follow the steps below.

TMS320C50 On-Chip Timer

As you can see from Figure 1, the internal TMS320C50 timer TOUT is used to supply the AIC with its master clock (MCLK). The TOUT pulse is activated each time the DSP's period counter decrements to zero. Refer to page 5-45 of the TMS320C5x Users Guide for the formula used to calculate the TOUT rate. The maximum TOUT rate is calculated by minimizing the denominator (min=2), therefore making the highest TOUT rate to be 1/2 the internal TMS320C50 machine cycle (TMS320C50-40 internal machine cycle is 20 MHz) or 10 MHz. Assembly code for generating the maximum TOUT rate is as follows:

```
SPLK  #01h, PRD
      ; Load PRD reg. for period of 100 nsec TDDR=0
SPLK  #20h, TCR   ; Re-load and begin timer.
```

Upon execution of the second SPLK instruction, TOUT will begin generating a 10-MHz square wave.

Serial-Port Initialization

The TMS320C50 serial port must be initialized by modifying the Serial-Port-Control register (SPC) of the DSP. The SPC is described on page 5-18 of the TMS320C5x Users Guide. In order to transmit and receive data from the AIC, the serial port must be set for Frame Sync Mode (FSM = 1) and reset by writing 0s to the XRST and RRST bits. At this point, a consecutive write to the SPC bits XRST and RRST with a 11 value will bring the serial port out of reset. PLEASE NOTE: A TOTAL OF TWO WRITES SHOULD BE MADE TO THE SPC TO RESET OR RECONFIGURE THE SERIAL PORT. The following code will initialize the serial port correctly:

```
SPLK     #08h, SPC      ; FSM=1, XRST and RRST = 00
SPLK     #0C8h, SPC     ; FSM=1, XRST and RRST = 11
```

After initializing the serial port, it is suggested a dummy word be sent to the DXR in order to clear any unwanted data from the serial-port registers.

AIC Initialization

Once the AIC is supplied with MCLK and the TMS320C50 serial port is initialized, a reset of the AIC should be performed to force the AIC into a known state. The RESET line of the AIC is connected to the BR (Bus Request) pin of the TMS320C50. The BR pin is driven low when external global memory is accessed. Therefore to reset the AIC, we must define global memory and access it. Refer to page 6-29 of the TMS320C5x Users Guide for more information about configuring global memory (GREG register). The code below illustrates how to initialize global memory, and assert BR to reset the AIC.

```
LACC  #80h      ; init 8000h-FFFFh as global memory
SACL  GREG      ; Store to Global Memory Alloc Reg.
LAR   AR0, #0FFFFh
              ; Use AR0 to point to location FFFFh
RPT   #10000  ; Access global memory 10,000 times
LACC  *, 0, AR0   ; to drive pin low for duration.
SACH  GREG        ; Restore GREG to 0000
```

As a result of the reset, the AIC is forced into a known stable state. At this point, AIC initialization of sample rates, etc. can be performed. Sampling rates are determined by the values in the A and B registers of the AIC’s transmit and receive sections. TX counter A and TX counter B determine the D/A conversion timing, whereas Rx counter A and Rx counter B determine the A/D conversion timing.

Page B-11 of the TMS320C5x DSK Users Guide illustrates that for an 8-KHz conversion time (transmit), the MCLK is always divided by TA, 2, and then by TB. Therefore the conversion timing is equal to 10.000 MHz/(TA × 2 × TB). If TA is 17, then TB must be roughly 37.

(10.000 MHz) / $(17 \times 2 \times 37) = 8.0$ kHz

TA is chosen to be 17 so that MCLK/(2 × TA) = 288 K. It will be explained later why this is important. The same rules apply to the receive (Rx) section. Please remember the example in Appendix B of the TMS320C5x DSK Users Guide is based on an MCLK of 10.368 MHz.

In order to initialize the analog chip, the AIC uses primary and secondary communications. Please refer to Appendix pages B-14 and B-15 of the TMS320C5x DSK Users Guide. Primary communications are used to load the TA, TB, RA, and RB counters. Secondary communication is used to load a value to the AIC control register and load the TA, TB, RA, and RB registers. The register values are loaded into the counters each time the counter decrements to zero (I/O new sample).

Table 1. AIC Primary And Secondary Communication Protocols

<table>
<tr><th colspan="17">Primary Serial Communications Protocol</th></tr>
<tr><th>d15</th><th>d14</th><th>d13</th><th>d12</th><th>d11</th><th>d10</th><th>d09</th><th>d08</th><th>d07</th><th>d06</th><th>d05</th><th>d04</th><th>d03</th><th>d02</th><th>d01</th><th>d00</th><th></th></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>0</td><td>0</td><td>Txa=TA, Rxa=RA, Txb=TB, Rxb=RB</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>0</td><td>1</td><td>Txa=TA+TA’, Rxa=RA+RA’, Txb=TB, Rxb=RB</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>1</td><td>0</td><td>Txa=TA–TA’, Rxa=RA–RA’, Txb=TB, Rxb=RB</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>1</td><td>1</td><td>Txa=TA, Rxa=RA, Txb=TB, Rxb=RB
Initiates secondary communication protocol</td></tr>
<tr><th colspan="17">Secondary Serial Communications Protocol</th></tr>
<tr><td>x</td><td>x</td><td colspan="5">TA Register</td><td>x</td><td>x</td><td colspan="5">RA Register</td><td>0</td><td>0</td><td>Unsigned binary</td></tr>
<tr><td>x</td><td colspan="6">TA’ Register</td><td>x</td><td colspan="6">RA’ Register</td><td>0</td><td>1</td><td>Signed two’s complement</td></tr>
<tr><td>x</td><td colspan="6">TB Register</td><td>x</td><td colspan="6">RB Register</td><td>1</td><td>0</td><td>Unsigned binary</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td>x</td><td colspan="6">Control Register</td><td>1</td><td>1</td><td></td></tr>
</table>

Control Register Bit Definitions:

d2 = 0/1 deletes/inserts the bandpass filter

d3 = 0/1 disables/enables the loopback function

d4 = 0/1 disables/enables AUX IN+ and AUX IN– pins

d5 = 0/1 Asynchronous/synchronous transmit and receive sections

d6 = 0/1 gain control bits

d7 = 0/1 gain control bits

The primary communication is defined by the two LSBs of the data word. For example, if the two LSBs are 00, then every time the counters decrement to zero (the next sample time) the A registers are loaded into the A counters and the B registers are loaded into the B counters. However, if the primary communication word LSBs are 01 or 10, then A counters are loaded with A+A’ or A–A’ respectively. Counter B is always loaded with the B register. The TA’ and RA’ are registers that can be used to advance or retard the sampling frequency by shortening or lengthening the sample period. This feature can be used to increase the signal-to-noise performance and is particularly useful in modem applications.

Secondary communication is initiated when the primary communication LSBs are 11. Secondary protocol allows you to load the A, B, and A’ registers and enable/disable other internal features of the AIC. As you can see from the above figure, the LSBs of the secondary communication word must be 00, 01, or 10 in order to load the A, A’, and B registers respectively. If the LSBs are 11, then bits 2–7 are used to initialize/alter the control register.

The control register provides a way to enable the auxiliary input (AUXIN), insert/delete the bandpass filter, change the input gain, and other features. Note: The gain is an input gain (pre-amplification) and not an output gain. The output gain is always 1. The input gain can be changed by setting bits 6 and 7 of the AIC control register to one of the following configurations:

Bit 7	**Bit 6**	**Gain (preamp)**
0	0	1
0	1	2
1	0	4

The bandpass filter can be selected or bypassed by setting bit 3 of the control word to a one or zero. The frequency response of this filter is shown in Appendix B of the TMS320C5x DSK Users Guide and is based upon a switch-capacitor-frequency (SCF) clock of 288 kHz. The SCF is equal to MCLK/(2 × TA) where MCLK is fixed at 10 MHz. As a result, it is impossible to achieve an exact SCF=288 kHz. Therefore, the frequency responses of the filters are scaled by the ratio of the actual SCF to 288 kHz. The closest 1:1 ratio between the actual SCF and 288 kHz, is when TA=17 (SCF=294 kHz).

SCF = (MCLK) / (2 × Counter A)

Conversion frequency = (SCF) / (Counter B)

Shift clock frequency = (MCLK) / 4

The shift clock frequency, shown above, is the rate at which the data is shifted from the AIC through the TMS320C50’s serial port. This is always 2.5 MHz, since MCLK is fixed at 10 MHz.

Conclusion

When initializing the hardware, it is safest to create the subroutine and ALWAYS call it before trying to use the AIC. After you have created your subroutine, it can be cut and pasted to other applications. An alternative is to use the subroutine included with the DSK demonstration programs. The AICINIT subroutine is executed every time in each demo. In fact, the subroutine is written so that easy changes can be made to the AIC’s TX, Rx, and control registers simply by changing the values located at the top of the file named TA, TB, RA, RB, and AIC_CTR.

Another Hint: When entering an Interrupt Service Routine (ISR), the DSP’s interrupts are automatically disabled. Since the debugger uses INT2 to communicate with the PC (and vice-versa), be sure to enable INT2 as one of the first tasks in the ISR.