Session 3226

Interfacing to a DSP through a Micro-controller via a DMA Port – Vertical Integration of DSP and Micro-controller Courses in an EET Program.

Anthony J. A. Oxtoby, Adam G. Sanderson
Purdue University, West Lafayette

Abstract

This paper describes a hardware interface that allows data transfer between the 8-bit 80C552 micro-controller and the 16-bit fixed point ADSP2181 digital signal processor. The interface provides a mechanism to connect slow external devices to the DSP processor without incurring wait states that are necessary when interfacing directly to the DSP memory spaces. It also provides a link connecting content of the two required microprocessor courses in the EET undergraduate program. A description of the operation of the interface is included along with an audio application that has been incorporated into the DSP course to take advantage of this data transfer method.

I. Introduction

The sequence of required microprocessor based courses in the EET program at Purdue University consists of an introductory sophomore level course based on the 80C552 micro-controller and an introductory DSP course at junior/senior level using the ADSP2181digital signal processor. The DSP course includes the operation and programming of a fixed-point processor and then moves on to cover the theory and implementation of common DSP applications such as filtering, audio effects and the DFT/FFT.

Because of the hardware emphasis in the course, interfacing to I/O devices has been included, permitting some external interaction to take place with the DSP algorithms. Directly interfacing such devices to the processor memory spaces reduces algorithm speed because wait states are needed in any transactions with these slower I/O devices. One alternative approach is to use serial data transfer through the processor’s serial ports and perform all the necessary scaling in the DSP. Another is to use an external controller to handle the acquisition and scaling of data for slower I/O devices and relay the data to the DSP processor via direct memory access. Simple circuitry to implement the data transfer and handshaking has been developed and is now included in selected laboratory exercises. This way one can extend the interfacing options in the course, take advantage of a wider range of I/O devices such as push buttons, LCD displays, multi-channel ADCs etc. and provide a direct link to earlier micro-processor courses. Student projects can also take advantage of this technique.

II. Microprocessor Hardware

The DSP course uses the ADSP-2181 based EZ-Kit Lite, an inexpensive yet versatile development system from Analog Devices. Incorporated onto the EZ_KIT Lite is a 16-bit fixed point, 30ns ADSP-2181 processor with 16k words of internal data memory, 16k words of program memory and several additional off-chip memory spaces. In addition, there is an AD1847 stereo audio CODEC with programmable sampling rate. Access is provided to all external data, address and control bus lines through expansion connectors on the board. Additional analog peripherals can be connected to the processor through the 2k word 16-bit I/O memory space but would require individual glue logic circuitry to decode bus signals along with wait state insertions to obtain reliable data exchange with slow I/O devices. In contrast, the introductory microprocessor course uses the 80C552 micro-controller along with a variety of I/O devices such as push buttons, potentiometers, LEDs, and LCD displays. All devices connect easily to the various ports on the 80C552 and students have a semester's experience of working with the system before taking the DSP class. It was in an effort to take advantage of both the 80C552 hardware and the students' experience and thus provide some continuity between the two courses that the interface between the two systems was developed.

III. The ADSP-2181 IDMA Port

The ADSP-2181 provides several means to interface with peripheral processors and devices. In addition to the interface buses used to access the various memory spaces, it contains an Internal Direct Memory Access (IDMA) port. The port supports boot loading and run-time access to the program memory (PM) and data memory (DM) spaces within the ADSP-2181 but does not permit access to internal memory mapped control registers.

Control of the IDMA port is achieved through an internal register mapped into DM(0x3FE0) accessible to both the ADSP-2181and the host device (80C552). In this application, only the 80C552 host accesses this register. Data transfer through the IDMA port takes place via the 16-bit IDMA port address/data bus and is controlled using the handshaking signals listed in table 1 below.

Table 1 IDMA Handshaking Signals

Pin Name(s)	Active State	Input/Outpt	Function
IRD	Low	Input	IDMA Port Read Strobe
IWR	Low	Input	IDMA Port Write Strobe
IAL	High	Input	IDMA Port Address Latch Enable
IS	Low	Input	IDMA Port Select
IAD15-0	▨	Input/Output	IDMA Port Address/Data Bus
IACK	Low	Output	IDMA Port Access Ready Acknowledge

Data transfer is initiated by writing, to the IDMA control register at DM(0x3FE0), a 16-bit word which defines the memory space to be accessed, PM or DM, and a 14-bit starting address. In the next IDMA bus cycle, data is either written to or read from the specified address in the ADSP-2181 memory, depending on the whether the port write (IWR) or port read (IRD) strobe lines respectively, are asserted. Following the memory access the address in the IDMA control register is automatically incremented thus avoiding additional address latch cycles when accessing contiguous memory locations. In all IDMA transfers, the port select (IS) must also be asserted. Note also that data transfer to and from program memory requires two bus cycles since the memory space is 24 bits wide and that the IDMA register address is thus only incremented after the two cycles are completed.

IV. The 80C552 to ADSP-2181 IDMA Interface

The IDMA port on the ADSP-2181 is specifically designed for a seamless interface to another 16-bit device whereas the 80C552 micro-controller is an 8-bit device. In order to transfer 16-bit data via the IDMA port, two 8-bit latches are used to create a bi-directional port to temporary hold the lower 8-bits of data during transfers. A block diagram of the interface is shown below in figure 1.

Two complete cycles of the 80C552 are needed for each cycle of the IDMA port. The operation when writing to the ADSP2181 data memory or the IDMA control register is as follows. During the first cycle, the LS byte of the address to be written to the IDMA control register is written to latch A. In the second cycle, the MS byte is written to the 80C552 bus and the output enable of latch A is asserted as is the IS and either the IWR or the IAL control lines respectively, of the IDMA port. Performing the transfers in this order supplies the IDMA port with the complete 16-bit value at one time.

When conducting an IDMA read cycle to transfer data to the 80C552, the process described above is reversed. During the first 80C552 external bus cycle, the IS and IRD control lines of the IDMA port are asserted and the MS byte of the data is read and the LS byte is written to latch B. For the second bus cycle, the output enable of latch B is asserted and the LS byte of the data is read. Glue logic for the interface is programmed into a GAL16V8 PLD to reduce chip count.

V. Gain Control of a Filter Using a Slide Potentiometer and Look-Up Table

Suppose we have a digital filter and we wish to control the pass band gain over a specified range of boost and cut using an external potentiometer. Specifically, consider the example shown below in figure 2 that consists of a low pass FIR filter implemented on the ADSP-2181. The relative gain of the filter is shown for the potentiometer positions at maximum (+10dB of boost), minimum (-10 dB of cut) and mid-point (0dB). Gain variation is simply obtained by multiplying the output signal w[n] from the basic FIR filter by a 16-bit 1.15 number derived from the raw 8-bit data representing the position of the potentiometer. It is thus necessary to scale the raw pot data to a range of 1.15 numbers which will produce the ±10dB of gain variation and, since the potentiometers are linear, to scale so as to establish a logarithmic relationship. Since the range of positive 1.15 numbers is 0 to 0.99996948…., then the dB range must re-scaled to maintain the ratios but keep it inside the 1.15 range which effectively means scaling to a gain range of 0 to -20 dB.

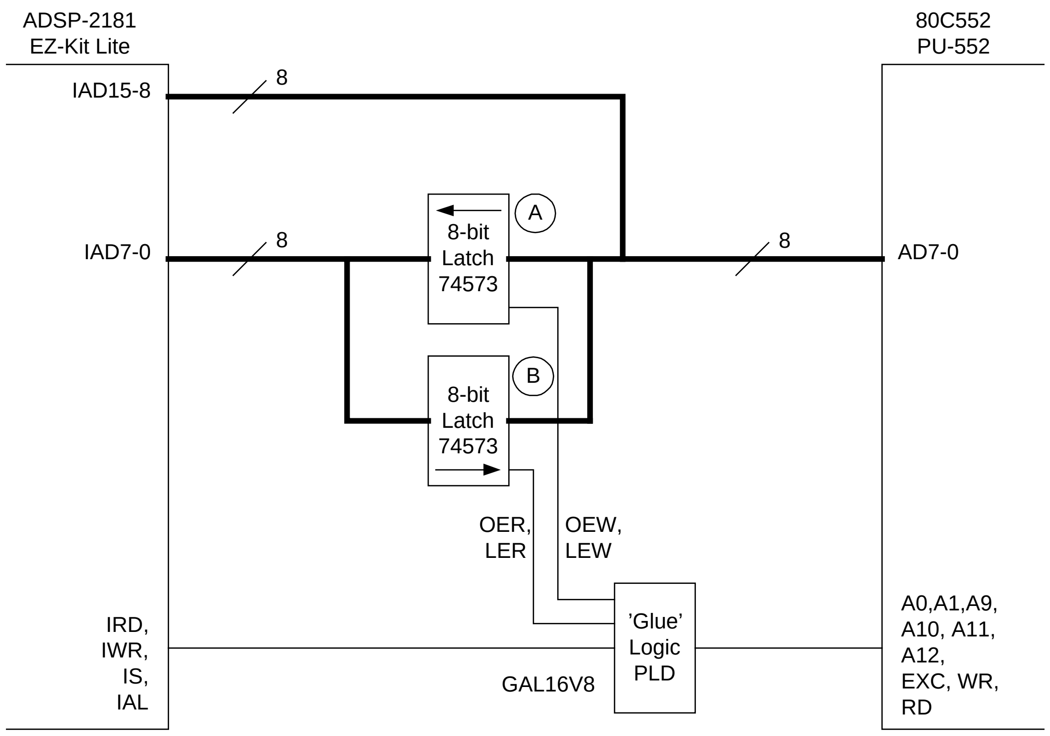

Figure 1 Block diagram of the 80C552 to ADSP-2181 IDMA port interface

In order to achieve the logarithmic variation a 256-point look-up table is used, with the potentiometer position being used to determine the point in the table from which the scaled data is fetched. Since, however, the 80C552 is an 8-bit machine, the 256-point 16-bit data must be organized in memory into two 256-byte segments. One segment contains the 256 LS bytes of the data and the other the 256 MS bytes. Movement through the look-up table is simply implemented by initializing a pointer to the base address of the LS byte table prior to looking up each new value, and then offsetting the pointer address with an increment value (0 - 255) derived from the potentiometer raw data. The LS byte is then transferred to latch B as described earlier. Retrieval of the MS byte is achieved from the other memory segment by offsetting the pointer by 256 from its current position, assuming the two segments are contiguous. The MS byte is then transferred through the interface along with the LS byte into the IDMA port ready for use by the ADSP-2181. The 8-bit table increment value is trivially obtained in the 80C552 by using the 8 MS bits of the 10-bit raw data from the on board ADC. Generation of the 16-bit look-up table data is performed using EXCEL. The process simply involves dividing the range 0 to -20dB into 256 increments, converting from dB to numerical gain and then converting these fractional gain values into 1.15 hexadecimal values. The data is then split into the two 256 by 8-bit LS and MS segments ready for initialization into the 80C552 memory.

Three of the 256 values required in the lookup table are given in table 2 below. Address values yy are determined by the base address of the table.

Table 2 Lookup Table Sample Values

ADC data	**Table address**	**±10 dB Gain change**	**Scaled data table**
0xFF	0xyyFF	+10dB ⇒ 0dB ⇒ 0.999969…	**0x7FFF**
0x80	0xyy80	0dB ⇒ -10dB ⇒ 0.316218…	**0x287A**
0x00	0xyy00	-10dB ⇒ -20dB ⇒ 0.0999969..	**0x0CCC**

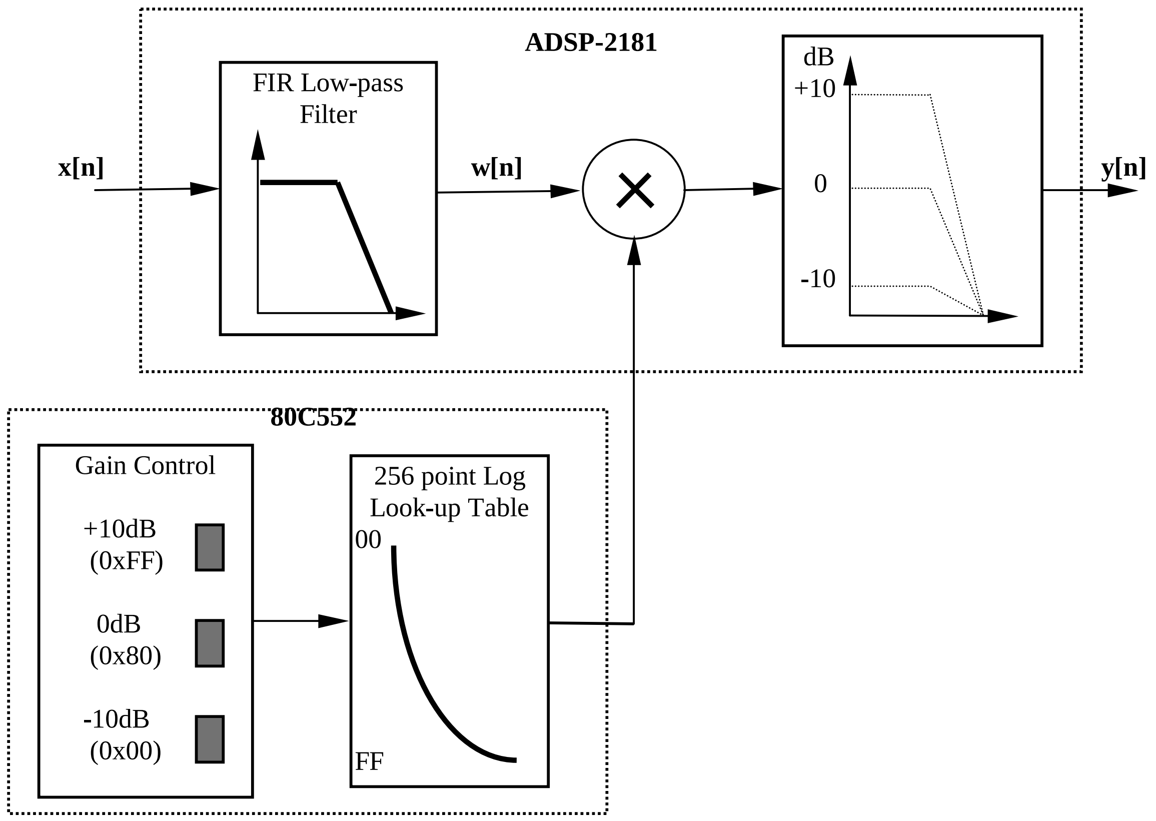

Figure 2 FIR Gain Control

VI. Conclusion

The implementation of the interface between the 80C552 micro-controller and the ADSP-2181 digital signal processor provides an opportunity for students to put to good use some of the experience gained in their first microprocessor class. The ability to exercise some control over the DSP algorithms they develop, especially in the exercise described above, provides a genuinely motivating experience. Students benefit from the exposure to a multiprocessor solution to realistic applications that serves to demonstrate the complementary aspects to using two processors. Other applications include the control of audio effects such as reverberation and

the control of the amplitude, phase, frequency and offset in a wavetable based function generator.
One other benefit is the experience the students gain in working with circuitry, timing and programming issues associated with interfacing a host processor to a DSP processor.

Bibliography
1. Analog Devices, ADSP-2100 Family EZ Tools Manual. 1994.
2. Alkin, O., Digital Signal Processing – A Laboratory Approach Using PCDSP. Prentice-Hall, 1994.
3. Vaidyanathan, P.P.,. *Multirate Systems and Filter Banks.* Prentice-Hall, 157-158 (1993)

ANTHONY J. A. OXTOBY
Tony Oxtoby is an Associate Professor of Electrical Engineering Technology at Purdue University in West Lafayette, Indiana and was educated in England at the University of Hull and the University of Manchester Institute of Science and Technology. He developed the course in Digital Signal Processing now required at three Purdue campuses. In addition, he has presented DSP workshops for industry and academics.

ADAM G. SANDERSON
Adam Sanderson is a senior in the Electrical Engineering Technology department at Purdue University in West Lafayette, Indiana. He has spent a considerable amount of time working on PC and digital systems for Advanced Micro Devices in Austin, TX and Sunnyvale, CA.