



## Application Note: Transport Delay

**DATE:** April 17, 1997  
**TO:** All PC/SBC31 and PC50 users  
**FROM:** Technical Staff  
**RE:** Analog I/O Transport Delay  
**CC:** Users

The analog I/O section on the PC31, SBC31 and PC50 products are based on the Burr-Brown DSP 102 A/D and DSP 202 D/A converters. These converters are not memory-mapped peripherals. Instead, they are interfaced to the synchronous serial port of the DSP. Because these converters reside on the synchronous serial port of the DSP, communication with them is accomplished via accesses to the serial transmit and receive registers of the DSP. Communications with serial devices such as these involve transport delays which are dependent on the bit rate, data word size and onchip buffering. This document discusses each of these issues in detail.

### Architecture

Each analog I/O channel consists of the elements depicted below.

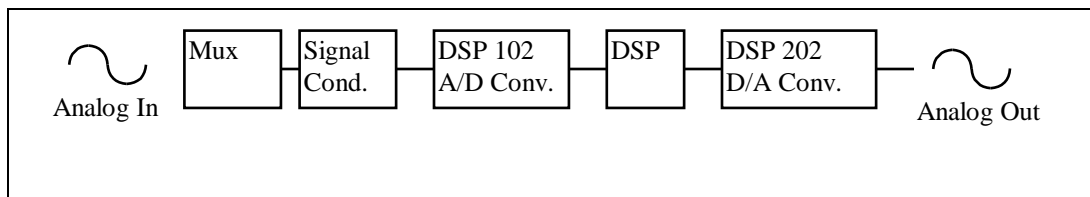


Figure 1: Analog I/O channel elements

Analog signals are routed through a mux, anti-alias filtering and other analog signal conditioning circuitry into the A/D converter. Data is transmitted from the A/D to the DSP via the synchronous serial receive channel. The D/A interface is similar, except that the serial transmit channel is used to send data to the D/A.

The data transfer rate from the DSP to and from the converters is dependent on the speed and type of DSP. For the PC31 and SBC31, the bit rate is H1/2 or 10 Mbits/second on a 20 MIPS machine (when the serial channel is initialized using the II initialization routine `enable_analog()`). On a PC50, the bit rate is CLKOUT1/2 or 5 Mbits/second on a 20 MIPS machine. The transmitted data is thirty-two bits wide on the PC31/SBC31 (plus eight bits of framing information) and sixteen -bits wide on the PC50 (plus four bits of framing information). Thus, regardless of the platform, serial data transfers from the A/D to the DSP take approximately 4  $\mu$ S.

## A/D Operation

The A/D converter has a dedicated input pin called CONVERT which is used to simultaneously initiate a conversion cycle and a serial data transfer operation. When the CONVERT pin is strobed, the A/D converter employs its onchip sample and hold circuitry to acquire a snapshot of the analog waveform within 30 nS after the convert pulse to the A/D. The A/D begins converting the sampled analog signal into a digital value. *Simultaneously* the A/D transfers data from its *holding register* (containing the *previously converted* analog sample) to its *shift register* which clocks the data out to the DSP one bit at a time via the synchronous serial interface to the DSP. Within 5 uS of the CONVERT signal, the conversion will be complete. When the new conversion is complete, the converted data is placed in the holding register within the DSP102. This characteristic is known as *double-buffering*. Because the DSP102 is double-buffered, data transmitted from the A/D is ***always delayed by one sampling interval***.

The DSP's serial receive port is also double-buffered. A/D data is clocked into the DSP's *receive shift register*. The data will remain in the receive shift register until the *serial holding register* is empty, at which time the data is transferred from the receive shift register to the receive holding register. The receive holding register is emptied when the DSP reads from the receive holding register.

## D/A Operation

The D/A converter has a dedicated input pin called CONVERT which is used to simultaneously initiate a conversion cycle and a data transfer operation. When the CONVERT pin is strobed, the D/A converter transfers the contents of its *shift register* into its *holding register* and begins converting the data in the holding register into an analog voltage. *Simultaneously* it begins clocking serial data representing the next conversion value from the DSP into its receive register via the synchronous serial interface. This characteristic is known as *double-buffering*. Because the DSP202 is double-buffered, data transmitted from the DSP to the A/D is ***always delayed by one sampling interval***.

The DSP's serial transmit port is also double-buffered. D/A data is written to the DSP's transmit holding register. The data will remain in the holding register until the *transmit shift register* is empty, at which time the data is transferred from the transmit holding register to the transmit shift register. When the serial port receives a framing pulse from the DSP202, data will be clocked from the DSP to the D/A one bit at a time via the synchronous interface.

## Worst-Case Data Transfer Latency

To illustrate the effects of this architecture, consider this example, ECHO.C, from the PC31 C Developers Package.

```
/*
 *
 * ECHO.C
 * Echo both native A/D input channels A & B (each set to mux channel 0)
 * to native DAC channels A & B at specified A/D conversion rate.
 *
 * It is assumed that the ADC and DAC0 are configured for CTC0-clocked
 * conversions using jumpers JP50 and JPADC on the PC31.
 */

#include "stdio.h"
#include "periph.h"

volatile int serviced; /* Tally of ISR executions */

main()
{
    int col, row, previous, freq, rate;

    MHZ = 40;

    enable_cache();
    enable_interrupts();
    enable_monitor();
    enable_clock(); /* Sets up for 1 kHz timebase on */

    clrscr();
    gotoxy(28, 0);
    bold();
    printf("\7Analog Loopback for PC31\n\n");
    normal();
    cursor(OFF);

    printf("\n\nEnter desired loopback rate (kHz): ");
    scanf("%d", &freq);
    timer(0, freq * 1000);

    enable_analog(); /* Initialize serial port to which */

    /* onboard analog I/O is connected */

    asm(" OR 2, IE"); /* Enable CTC0 interrupt */

    printf("\n\nMeasured loopback rate: ");
    wherexy(&col, &row);
    printf(" kHz");

    while(!kbd_hit()){
        previous = clock();
        serviced = 0;
        ms(50);

        gotoxy(col, row);
        printf("%8.3f", ((float)serviced/((float)(clock() - previous))));
    }

    monitor();
}
```

```

/*
 *
 * A/D, D/A interrupt routine
 *
 * This handler is invoked whenever CTC0 expires.
 *
 * The PC31's DSP102 A/D converter simultaneously converts
 * both its A & B input channels in approximately 5 uS. When strobed, it
 * transmits the two 16-bit results as a 32-bit integer
 * to the synchronous serial port of the C31. Upon receipt, the C31
 * generates an interrupt, which causes this ISR to be serviced.
 * In this routine, the A/D results are echoed out to DAC channels A & B
 * of one of the two onboard BB DSP202 DACs.
 *
 * Note: The analog converter data is transceived to/from the C31 via
 * an 8.25 Mhz synchronous serial link. This results in a transport delay
 * of ten microseconds (40 + 40 clks/8.25E6) from input to output.
 */
void c_int02()
{
    *SER_TD = *SER_RD;    /* Echo received A/D data to DACs */

    serviced++;          /* # times this ISR serviced */
}

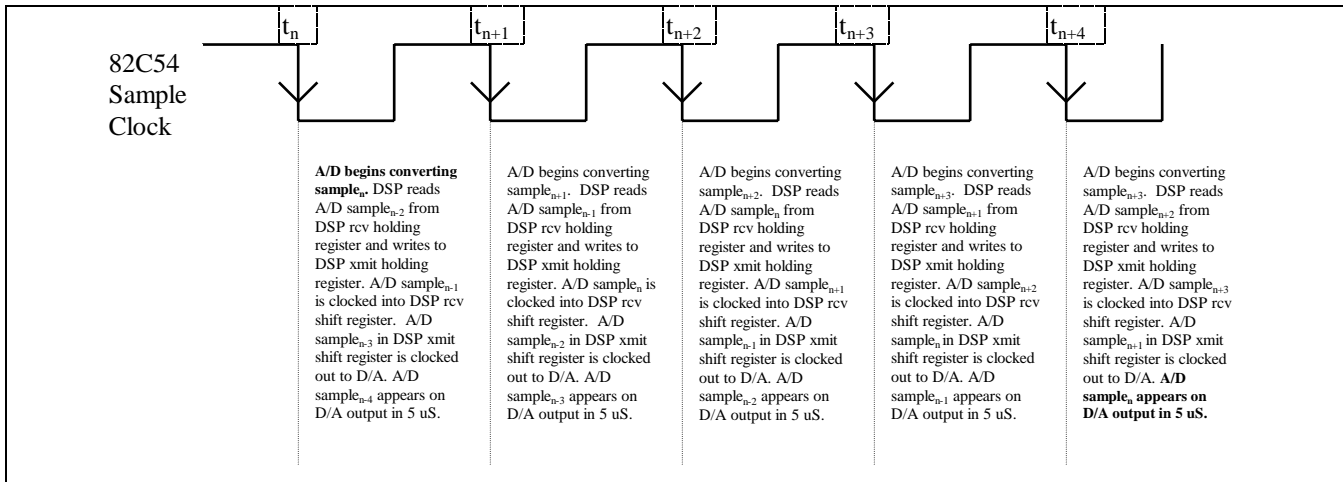
```

This program simply digitizes the analog input to channel A of the DSP102 A/D converter and echoes the converted sample out to channel A of DSP202 D/A converter # 0. This “loopback” function executes at a user specified rate from 1 to 200 kHz using the onboard 82C54 counter timer channel 0 output as a timebase for both analog conversion pulses and initiating the interrupt service routine.

This core of this program (this is the PC31 version) is the interrupt routine `c_int02()`, which simply reads the serial receive holding register and writes the data to the serial transmit holding register.

The total latency involved in the analog loopback is equal to five sampling intervals plus the transport delay of data clocked out to the D/A from the DSP. To understand why this is so, consider the following timing diagram.

At the falling edge  $t_0$  the A/D and D/A are signaled to begin another conversion. At that instant, the DSP's transmit shift register contains the  $A/D_{n-3}$  data point, written in by the DSP during the previous execution of the service routine. The DSP's receive holding register contains the  $A/D_{n-2}$  data point, clocked in from the DSP102 during the previous sampling interval.



The transport delay from input to output is dependent on the user-specified loop frequency. At the maximum frequency of 200 kHz, the transport delay is:

$$(4 \text{ sampling intervals}) \times (5\mu\text{S}/\text{sampling interval}) + 5 \mu\text{S} (\text{final dac convert}) = 25 \mu\text{S}.$$

Assuming a 1 kHz sinusoidal input signal, this corresponds to a phase delay of:

$$30 \mu\text{S} / 1000\mu\text{S} * (360 \text{ degrees}) = 9 \text{ degrees}$$

Note however that if the loop frequency is set to 10 kHz, the transport delay becomes:

$$(4 \text{ sampling intervals}) \times (100\mu\text{S}/\text{sampling interval}) + 5 \mu\text{S} (\text{final dac convert}) = 405 \mu\text{S}.$$

With the same 1 kHz sinusoidal input signal, this corresponds to a phase delay of:

$$405 \mu\text{S} / 1000\mu\text{S} * (360 \text{ degrees}) = 146 \text{ degrees}$$

Bear in mind that since the DSP's synch serial port is also double buffered, it is possible (but undesirable) to code applications where an extra sample is inadvertently left in either the DSP serial receive or transmit buffer (or both) which introduces additional transport delay(s).

Note also that the anti-alias input filter on the cards will typically add approximately another 1.2 degrees of phase shift and the D/A anti-alias filter will add approximately .6 degrees of phase shift. This effect is generally negligible in data acquisition and control applications.