Designing With the TLC320AC01 Analog Interface for DSPs

SLAA006
May 1995

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*

Contents

List of Illustrations

1 INTRODUCTION

This application report was prepared by John Walliker and Julian Daley of University College London. It is based on their experience of using the device as part of a specialized signal processing hearing aid. However the techniques described, for both the analog and digital interfaces, are appropriate for a wide variety of applications.

Measurements of performance quoted in this application note are those achieved with the particular samples and test set-up. For the full device specification see the TLC320AC01 data manual, reference SLAS057A.

Some features of the TLC320AC01 were not used in this design and therefore have not been covered here. They are phase adjustment and the use of multiple devices.

1.1 Overview of Device

The TLC320AC01 is a 14-bit resolution, audio frequency (approximately 12-kHz bandwidth) analog interface for DSP with integral anti-aliasing and reconstruction filters. It has a synchronous, serial, digital interface designed for ease of connection to many DSP chips.

The internal circuit configuration and the performance parameters, such as input source, sampling rate, filter bandwidths and gain, are determined by writing in control information to eight data registers. These registers are used to set-up the device for a given mode of operation and given application. The ADC channel and the DAC channel operate synchronously and data is transferred in 2's complement format.

The anti-aliasing filter is a switched-capacitor low-pass filter with a sixth-order elliptic characteristic. The high-pass is a single-pole filter, which can be switched out if required. There is a 3-pole continuous-time filter that precedes the switched-capacitor filter to eliminate aliasing caused by sampling in the switched-capacitor filter.

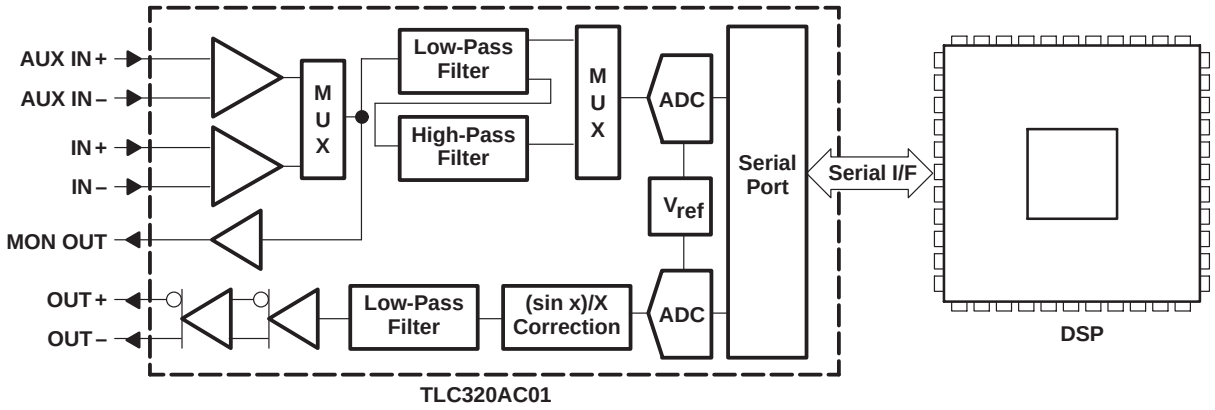
The output-reconstruction filter is also a switched-capacitor low-pass filter with a sixth-order elliptic characteristic and it is followed by a second-order (sin x)/x correction filter. This is followed by a three-pole continuous-time filter to eliminate images caused by sampling in the switched-capacitor filter.

There are three basic modes of operation available:

- Stand-alone analog-interface mode, where the TLC320AC01 generates the shift clock and the frame sync for the data transfers and is the only AIC used.
- Master-slave mode, where the master TLC320AC01 generates the shift clock and the frame sync and the rest are slaves to these signals.
- Linear-codec mode, where the shift clock and the frame sync are generated externally and the timing can be any of the standard codec timing patterns.

The TLC320AC01 is available in a standard 28-pin plastic J-lead chip carrier (FN suffix) and a 64-pin plastic-quad-flat-pack (PM suffix) which is only 1,5 mm thick, making it suitable for use in portable systems.

The device has a maximum power dissipation of 110 mW in the active mode and 10 mW in the power down mode. It runs from a single 5-V supply, both for digital and analog circuitry. This is particularly useful for portable equipment, but does require extra care in the design of the analog input and output stages.



- 14-bit A/D converter
- 14-bit D/A converter
- Antialiasing filter
- Anti-imaging filter
- (sin x)/x correction
- Input and output amplifiers
- Programmable bandwidth, sampling rate, gain, I/P source
- Single 5 V supply
- 70 dB typical signal to noise+distortion ratio
- 100 mW typical power consumption
- Analog bandwidth up to 10.8 kHz

Figure 1. TLC320AC01 Analog Interface for DSP

2 ANALOG INPUT

2.1 Signal-to-Noise and Signal-to-Distortion Measurements

With the internal gain of the TLC320AC01 set to 0 dB, a full scale signal corresponds to 6 V peak-peak at the analog input (equivalent to $6/(2\sqrt{2}) = 2.12$ V RMS).

The input signal-to-noise ratio of the TLC320AC01 can be expressed in terms of the number of least significant bits (LSB) of noise present in the digital signal, when both its inputs are connected to V_{MID}. The RMS value of the noise was measured on the test boards at 0.5 LSB. This corresponds to a noise voltage of approximately 180 μV RMS at the input (i.e., a signal-to-noise ratio of 81 dB). The intermodulation measurements are shown in Figure 2. The stimulus was the sum of a 1 kHz signal at −6 dB referred to full scale plus a 1.2 kHz signal at −12 dB referred to full scale. Distortion products are approximately 80 dB down throughout the pass band. The low frequency peaks that can be seen are multiples of 50 Hz interference.

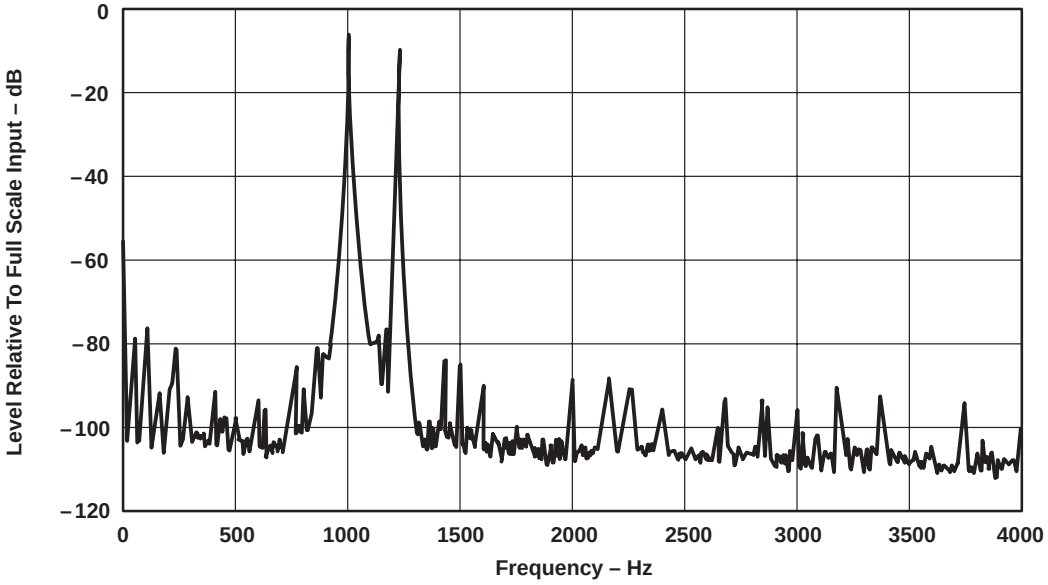


Figure 2. ADC Noise and Distortion Measurement

In the test circuit, the dc accuracy on the samples measured was 14 LSB, equivalent to 5 mV of dc offset.

2.2 Input Preamp Design

2.2.1 Noise Considerations

In order that the input preamp does not significantly affect the noise performance of the system, it should produce a noise level at least 6 dB below the TLC320AC01, (i.e., less than 90 μV RMS) at the TLC320AC01 input.

Consider the case of a microphone producing 20 mV peak-to-peak at the maximum sound level, a preamp is needed with a gain of 6 V.20 mV = 300 to get a full scale input at the ADC. So the input noise produced by the preamp must be less than 90 μV/300 = 300 nV RMS.

For a preamp with a bandwidth of 10 kHz the input noise voltage should be less than

$$\frac{300\ \text{nV}}{\sqrt{10\ \text{kHz}}} = 3\ \text{nV}/\sqrt{\text{Hz}}$$

This noise is made up of the operational amplifier's noise voltage combined with the thermal noise of the equivalent series resistance of the input source. Resistor values need to be carefully chosen, since a 10 kΩ resistor produces thermal noise of 14 nV/$\sqrt{\text{Hz}}$ at room temperature. (spectral noise voltage density for a resistor is given by $\sqrt{4KTR}$, where K is Boltzman's constant, T is the absolute temperature and R the resistance). In this case, a 100-Ω resistor was chosen

(producing a thermal noise of 1.4 nV/$\sqrt{\text{Hz}}$ at room temperature). A MAX410 operational amplifier was chosen for the first gain stage as this has a noise voltage of 2.4 nV/$\sqrt{\text{Hz}}$. Noise voltages combine as the root of the sum of the squares, so the total noise is given by:

$$\sqrt{\left(2.4\ \text{nV}^2 + 1.4\ \text{nV}^2\right)} = 2.8\ \text{nV}/\sqrt{\text{Hz}}$$

The gain is split between two operational amplifiers. The first, low noise, operational amplifier configured as a noninverting amplifier with a gain of 100, followed by a second noninverting stage with a gain of three. This second operational amplifier does not need such a low noise voltage specification since its input noise is only being amplified by three. The TLC2272 dual operational amplifier, which has a noise voltage of 9 nV/$\sqrt{\text{Hz}}$, is chosen for its low power consumption, low input offset and well behaved performance under overload. (These operational amplifiers do not exhibit the behavior of BiFETs which can produce phase reversal of the output when the inputs go out of negative common mode range).

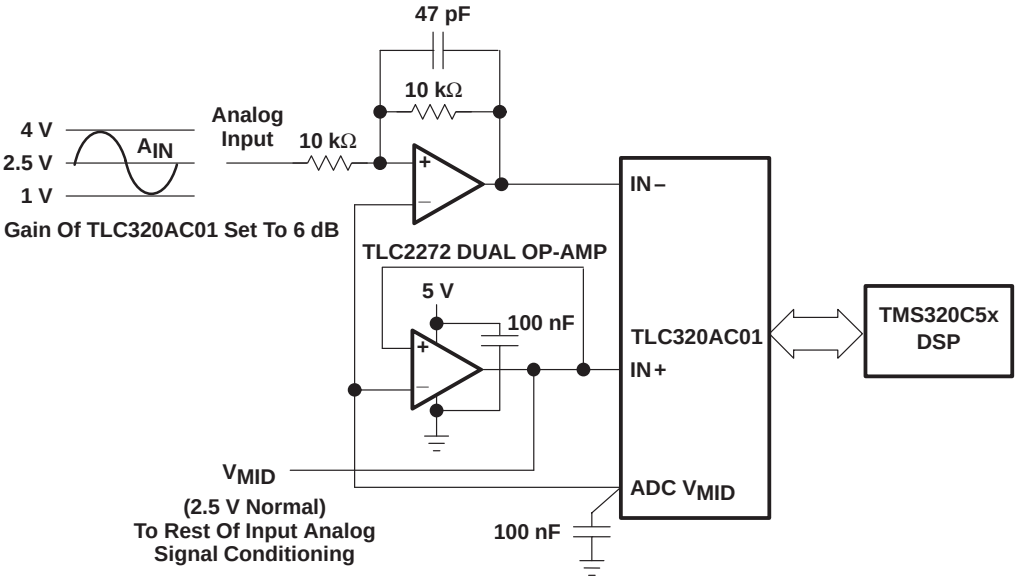


Figure 3. V_{MID} Referenced Input Circuit

2.2.2 V_{MID} Referenced Input Circuit Configuration

The configuration of the input circuitry requires extra care since all internal signals are referenced to V_{MID} rather than ground, to allow single supply operation. The PSRR at the internally generated V_{MID} point is low, so it is important that both the differential inputs are referenced to V_{MID} with any noise on V_{MID} appearing equally on both inputs. There are two ways of fulfilling this criterion. The first is to reference the whole input circuit to V_{MID} (using this as a virtual ground) as shown in Figure 3.

This configuration has the advantage of simplicity although there are some drawbacks. The buffered V_{MID} point has to be capable of driving the virtual ground and since many operational amplifiers are unhappy driving large capacitive loads this problem must not be overlooked. The TLC2272 is a good choice for this application. The input needs to be referenced to V_{MID}, which may cause a problem if interfacing to an externally powered, ground referenced signal. In this case the input needs to be ac coupled.

2.2.3 0-V Referenced Input Circuit Configuration

The second method is to level shift the signal just before the ADC inputs as shown in Figure 4. In this circuit, the preamp input is referenced to 0 V. This circuit allows a full range input swing ($V_{MID} \pm 1.5$ V on each input) for an input signal of ± 1.5 V. Any noise on V_{MID} appears equally on both differential inputs and is therefore cancelled. The common mode range of the inputs does not exceed the supply rails, so V_{MID} noise must not take the input signal outside the supply rails. The eight resistors can conveniently be in one thin film resistor package, giving good matching of resistor values and hence good power supply rejection ratio (PSRR) and dc accuracy. Amplifier A1 must have ± 5 V or greater power rails but A2 to A4 only need a single 5-V rail.

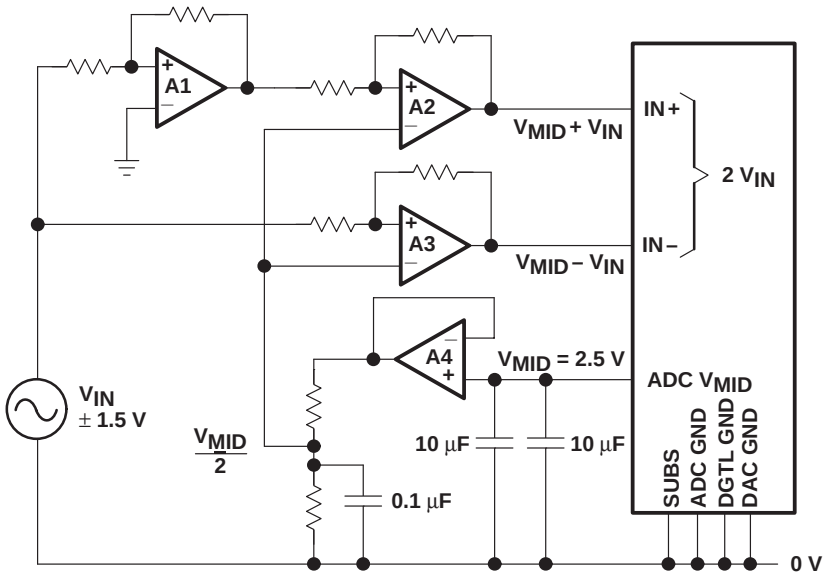


Figure 4. 0-V Referenced Input Circuit

2.2.4 Gain Control

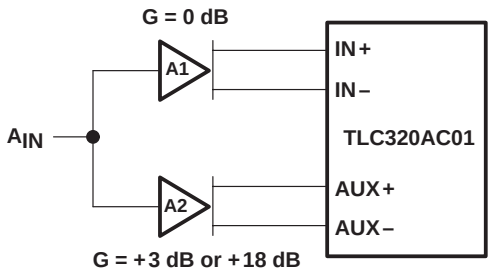
The internal preamp of the TLC320AC01 has software selectable internal gain of 0 dB, 6 dB or 12 dB plus a squelch mode (–60 dB). With 0 dB gain, plus or minus full scale result is given for a differential input of ±3 V. With a single ended input configuration (one input tied to V_{MID}), this would not allow plus or minus full scale before the operational amplifiers run out of headroom, so the gain must be set to 6 dB or 12 dB which would give plus or minus full scale with ±1.5 V and ±0.75 V, respectively, at the TLC320AC01 input.

Most of the input noise is associated with the converter itself, rather than the input amplifiers or multiplexers. Therefore, the signal-to-noise ratio is hardly affected by the chosen input gain. However, it is easier to ensure good rejection of power supply noise coupled through V_{MID} at low gains.

The TLC320AC01 has two sets of differential inputs, IN and AUX IN which can be individually selected (or both selected simultaneously for mixing).

If more gain settings are required, a combination of software switching of input source and input gain coupled with an extra hardware gain stage (see Figure 5) allows six software selectable gain steps as shown in the following table.

EXTERNAL GAIN 3 dB			EXTERNAL GAIN 18 dB		
NORM/AUX INPUT	INTERNAL GAIN (dB)	TOTAL GAIN (dB)	NORM/AUX INPUT	INTERNAL GAIN (dB)	TOTAL GAIN (dB)
NORM	0	0	NORM	0	0
AUX	0	3	NORM	6	6
NORM	6	6	NORM	12	12
AUX	6	9	AUX	0	18
NORM	12	12	AUX	6	24
AUX	12	15	AUX	12	30



Internal Gain Settings 0 dB, 6 dB, 12 dB

Figure 5. Input Circuit for 6 S/W Selectable Gain Settings

2.3 Layout and Grounding

Although earthing and PCB layout do not seem to be too critical for this device, it is good practice to ensure that the ground current from sensitive devices such as the ADC does not flow in the same copper as currents from other devices. This means having a central ground point near the device or using power planes with splits where necessary to isolate return current from other devices.

The substrate (SUBS) should be connected to ADC ground. Failure to do so can result in noisy and unstable operation. The circuit should be well decoupled for low and high frequencies to minimize noise injection from the supplies.

2.4 Power Supply

With a master clock frequency of 10 MHz, the TLC320AC01 samples typically drew 10 mA at 5 V with default register values. The supply current depends principally on the filter clock frequency. If a negative supply is needed for operational amplifiers, etc., it may be convenient to generate it using a negative voltage converter. Since the negative supply generally draws little current this is a feasible solution and avoids the need for a second battery in portable systems. The ICL7660 needs no external inductors and is available in an 8-pin small outline package. As the internal oscillator of the ICL7660 free runs at about 10 kHz, noise generated from this oscillator can find its way into the ADC input, often beating with the sampling clock creating a whirring type noise. It is however possible to lock this oscillator to the ADC clock by linking the conversion complete signal to the oscillator input on the ICL7660. Coupling via a 100-pF capacitor allows the converter to free-run if the ADC is not operative (e.g., during start-up). Any noise that now gets coupled into the ADC will be the same for each sample, creating a dc result that is much easier to deal with. Alternatively the TLE2682 provides a negative rail generator supplying up to 100 mA (which can be phase locked) plus a dual operational amplifier in one 16-pin wide body SO package).

2.5 Sampling Rate and Filters

Within limits, the sampling rate of the device (both ADC and DAC are inherently synchronous) can be set under software control. If the DAC is not used then the ADC can run at up to 43.2k samples/sec. However, if the DAC is to be used the sampling rate must be limited to 25k samples/sec.

The anti-aliasing filters (switched capacitor type) track the sampling rate by setting the corner frequency of the filter to some fraction of the sampling rate. This allows for the possibility of sub-Nyquist sampling, which should be avoided in most cases. The ratio of sampling rate to anti-aliasing filter corner frequency is set by the B register value (REG_B). The anti-aliasing corner frequency is set by the A register value (REG_A) within the TLC320AC01.

Conversion rate is given by:

$$f_{sample} = \frac{f_{MCLK}}{\left(2 \times Reg_A \times Reg_B\right)}$$

The anti-aliasing corner frequency is given by:

$$f_{lp} = \frac{f_{MCLK}}{80 \times Reg_A}$$

$$\frac{f_{sample}}{f_{lp}} = \frac{40}{Reg_B}$$

To satisfy Nyquist's sampling theorem:

$$\frac{f_{sample}}{f_{lp}} \geq 2$$

$$\therefore Reg_B \leq 20$$

The default of 18 for the B register gives $f_{sample}/f_{lp} = 2.2$. This ensures that energy above the Nyquist frequency is well into the filter's stop band.

The product of the A and B registers must be greater than 65 to allow for 17 serial clock cycles between conversions (16 data bits plus one extra cycle for frame sync in master or standalone mode). The B register must not be less than 10, since the ADC conversion takes 10 B register counts to complete. The A and B registers have a maximum value of 255.

2.5.1 High-Pass Filter

The TLC320AC01 also has a high-pass filter which can be used to attenuate subsonic noise and remove dc offsets. The importance of subsonic noise filtering should not be underestimated. For example: air conditioning systems are a notorious source of low frequency noise and a slamming door can produce extremely high levels of subsonic energy. The filter in the TLC320AC01 has a corner frequency of $f_S/200$ and a slope of 6 dB per octave. The corner frequency cannot be changed independently of the sampling frequency.

3 ANALOG OUTPUT

As previously mentioned, the maximum sample rate for the DAC, at 25 kHz, is lower than for the ADC. This limits the bandwidth of the output signal to less than 12.5 kHz.

3.1 Signal-to-Noise and Signal-to-Distortion Ratio

Figure 6 shows the result of intermodulation distortion measurements for the DAC made on the test boards. The noise floor can be seen at approximately –90 dB in the pass band, falling to approximately –108 dB at frequencies above $f_s/2$. There are some distortion products in the pass band at approximately –85 dB. The double peaks at approximately 7 kHz and 9 kHz are images of the signal that have been only partially attenuated by the reconstruction filter. Images are the digital to analog equivalents of aliases in analog to digital conversion. They occur at frequencies given by:

$$f_{image} = N\ f_S \pm f_{in}$$

N = 1, 2, 3 . . .

f_S = sampling frequency

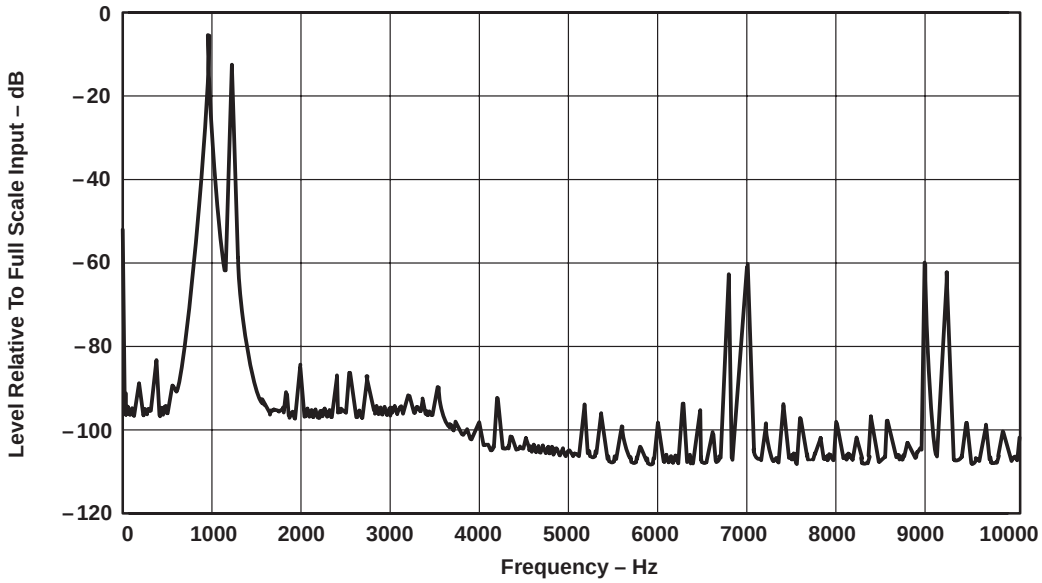


Figure 6. DAC Noise and Distortion Measurements

If the images are to large for a given application they can be removed by continuous-time low-pass filtering at the output of the DAC. The size of the images reflects the 45 dB stop-band attenuation of the reconstruction filter.

3.2 Voltage Swing and PSRR

The voltage swing at the differential output is ±6 V for a full scale output. There are software selectable attenuators giving outputs of 0 dB, –12 dB and a squelch mode of –60 dB. Although there is not a large improvement in SNR ratio by using a differential output stage, it has the added advantage of increasing the PSRR and allowing level shifting to a ground referenced output without having to ac couple the signal. Using a thin film resistor pack for the differential amplifier gives the well matched resistors needed for good common mode rejection and accurate gain. Using a differential amplifier in this way the PSRR was improved from 49 dB to 53 dB (see Figure 7).

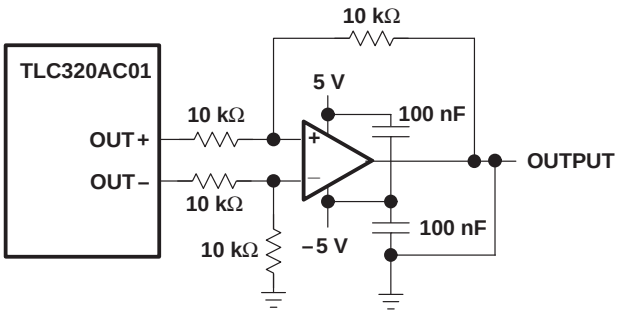


Figure 7. Differential to Single Ended Output Circuit

3.3 (Sin x)/x Correction

(Sin x)/x error arises because the output from a digital-to-analog converter is held constant between samples rather than smoothly joining them up. The TLC320AC01 has a (sin x)/x correction filter. It gives a correct response for a B register value of 15, which gives a ratio of sample rate to ADC anti-aliasing filter of 2.67. But as it does not track the B register, other values for the B register will produce an error in the magnitude of a given output frequency. Figure 8 shows a graph of calculated error versus frequency for various values of B register with a master clock of 10 MHz and sample rate of approximately 8 kHz. Other values can be calculated using the equation given in section 2.15.7 of the TLC320AC01 data manual.

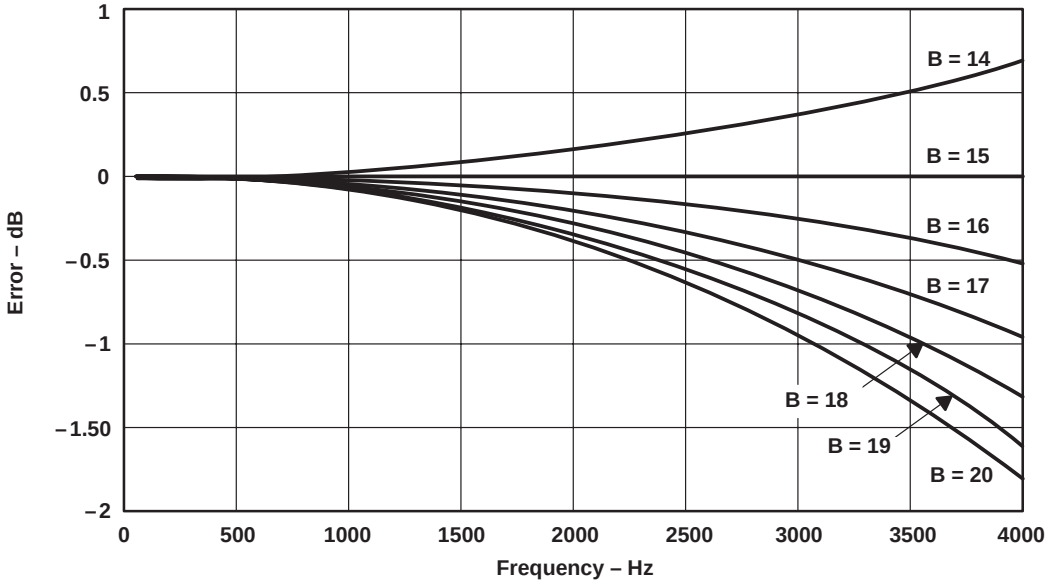


Figure 8. (Sin x)/x Error

4 DIGITAL DESIGN CONSIDERATIONS

4.1 DSP Serial Interface

The TLC320AC01 can be connected directly to the synchronous serial port of a TMS320C25 as shown in Figures 6–1 and 6–3 of the TLC320AC01 data manual. Interfacing the TMS320C50 family requires some caution because the CLKOUT signal will usually exceed the maximum 15 MHz MCLK frequency of the TLC320AC01. So a divider is required. Most makes of DSP chip support the synchronous serial interface and should connect directly to the TLC320AC01.

4.1.1 Maximum Clock Rate

It is highly desirable that the DSP chip and TLC320AC01 are clocked from a common master oscillator. This ensures that the digital noise which is often coupled into the ADC and DAC of the TLC320AC01 in miniature systems is aliased to a stable frequency, preferably dc. Texas Instruments has found that the signal-to-noise ratio can be degraded by as much as 6 dB in a breadboard system when independent clocks are used compared with a fully synchronous system.

A convenient way of phase locking the TLC320AC01 to the processor is to drive the MCLK input of the TLC320AC01 from the CLKOUT of a TMS320C25 or TMS320C50 or from the H1 or H3 output of a TMS320C30. However, because of the higher speed of the TMS320C50 an external one or two stage divider must be used to lower the CLKOUT frequency (which can be between 20 MHz and 40 MHz depending on which speed grade of processor is used) to 15 MHz or less. A SN74ACT74 was chosen for its high maximum clock frequency, relatively low power consumption and availability in surface mount package. The divider power supply current was measured at 6.5 mA at 5 V and 3.17 mA at 3 V when dividing by 2 at 20 MHz; and 20.3 mA at 5 V and 9.6 mA at 3 V when dividing by 4 at 40 MHz.

4.1.2 Synchronization of Negative Rail Generator

If switching power supplies are used in the system, for example to generate a negative supply rail, it is advantageous to phase lock the switching frequency to the sampling frequency. The EOC output from the TLC320AC01 is close to a square wave for reasonable sampling frequencies and can be connected to the OSC pin of a ICL7660 negative supply generator through a small (100 pF) capacitor to force the normally free running oscillator. The small capacitor allows the ICL7660 to free run in the absence of an EOC signal. The ICL7660 divides this signal by two internally, so that power supply ripple is at exactly the Nyquist frequency and hence appears as a small dc offset rather than as an unstable whistle.

4.1.3 Edge Timing

It is important to ensure that the rise and fall times of the serial clock signal between the codec and DSP chip are within specification, particularly if level shifting circuits are used for mixed 5 V and 3 V operation. Failure to do so can result in data or frame sync signals being sampled on the wrong clock edge, causing erratic errors. The shift clock output rise and fall times for the TLC320AC01 in master mode are specified at 19 ns maximum and 13 ns typical in the data manual. The maximum serial clock input rise and fall times for the TMS320C25 are 25 ns, and for the TMS320C30 and TMS320C50 are 8 ns. While it might seem from these specifications that the TLC320AC01 cannot satisfactorily drive the TMS320C30 or TMS320C50 without buffering, these devices do work reliably together as long as very short connections are used.

In order to discover whether the system was operating with a reasonable safety margin, the rise and fall times of SCLK were measured. The measurements were made using a LeCroy oscilloscope sampling at 1 GHz and a FET probe with SCLK from the TLC320AC01 driving the parallel inputs CLKX and CLKR on the TMS320C50. The PCB track length was approximately 10 cm and the width 0.25 mm. The results (shown below) were well within the requirements of the TLC320C50.

	V_{DD} = 5 V	V_{DD} = 4 V
Rise time (0.8 V to 2 V)	2.3 ns	4.1 ns
Fall time (2 V to 0.8 V)	4.4 ns	5.6 ns

4.2 Hardware Design of TMS320C50 Based DSP System

A relatively simple yet powerful DSP system can be built using a TMS320C50 family digital signal processor and a TLC320AC01 AIC, as shown in Figure 9. The circuit shown is a simplified version of one that we have used extensively. It can readily be expanded to include parallel input and output ports. The TMS320C50 has 10K words of on-chip RAM, allowing complex algorithms to be implemented without the need to use external RAM. Some means of program storage is needed. A pair of 8-bit wide 1-Mbit flash EPROMs (N28F001BX-B120 from Intel) were used which completely fill the program and data address spaces, allowing the use of very large data tables. They have the advantages of reasonably low power consumption, especially when idle, the ability to be reprogrammed in-circuit or in a standard programmer (if socketed) and have a hardware protected boot block. The boot block is an 8K byte segment starting at address zero which can be protected against erasure by opening a switch. This allows the EPROM programming algorithm to be safely stored within the EPROM itself, downloaded to on-chip memory and executed from there to reprogram the rest of the flash EPROM with data transmitted through one of the serial ports. This is very convenient when portable equipment is to be reprogrammed in the field, especially when surface mounted devices are permanently soldered into the circuit. The initial bootstrap code can either be loaded using a standard programmer before assembly, or afterwards using the XDS510 in-circuit emulator interface which is brought out to a 14-pin header. These flash EPROMs have an internal state machine to control the erasure and programming algorithms. This is very important, not only because it simplifies programming, but it ensures that the essential precharge step before erasure is applied to all locations. This cannot easily be done with earlier generations of flash memory because those addresses that overlap internal registers and memory cannot easily be accessed. 100 kΩ pull-up resistor packs were used on the data bus and serial port signals to minimize power consumption when the bus is in a high impedance condition, which is the normal condition when executing from on-chip RAM.

A standard, 40 MHz, third overtone crystal oscillator was used to clock the TMS320C50. Exactly 8 kHz or 16 kHz sampling frequencies cannot be obtained with a 40 MHz MCLK. If this is a requirement, an MCLK of 41.475 MHz should be used. A 2.2 μH surface mount inductor blocks oscillation at the fundamental frequency of the crystal. The 330 kΩ resistor biases the on-chip oscillator inverter. Some care is needed in the choice of this value to ensure stable operation and reliable start-up. With the component values shown, the oscillator starts at a supply voltage of approximately 2 V and is stable up to the absolute maximum of 7 V. Alternatively, CLKMD2 of the TMS320C50 can be grounded and an external 20 MHz clock fed into CLKIN2. This provides a divide by 1 option whereby the CPU clock operates at the same 20-MHz frequency. The 20 MHz, CLKOUT1 signal from the TMS320C50 is divided by two using half a SN74ACT74 D-type flip-flop to provide a 10 MHz MCLK to the TLC320AC01.

4.3 Battery Operation

4.3.1 Reset Considerations

The TLC320AC01 undergoes a power-on reset when V_{DD} falls below 4 V with the samples tested. In battery powered systems it is important to ensure that the supply never dips this low, otherwise all the programmed registers will return to their default values. To guard against undetected resetting, the system supply should be monitored, using a comparator as shown in Figure 9 or a supply voltage supervisor such as the TL7702B. Also, one of the registers that has been changed from its default should periodically be read back and checked.

4.3.2 Interfacing to a 3-V DSP Processor

There is a strong incentive to operate DSPs at 3 V or 3.3 V to save power. As the TLC320AC01 resets at a V_{DD} of about 4 V, separate power supplies and level shifting circuits must be used. The signals from a true CMOS DSP such as the TMS320C50 swing from 0 to V_{DD}, that is from 0 V to 3 V. As this is greater than the 2.2-V logic high threshold of the TLC320AC01, all signals from the DSP to the TLC320AC01 can be directly connected, provided that the 5-V supply rises and falls faster than the logic supply at switch on and off, respectively. If the power sequence cannot be guaranteed, the TLC320AC01 inputs should be protected with a series resistor of about 3.3 kΩ compensated with a parallel capacitor of about 1 nF. There will be a small increase in I_{DD} of the TLC320AC01 compared with driving from 0 V to 5 V due to simultaneous conduction by both FETs in the input circuits. Signals from the TLC320AC01 to the DSP cannot be directly connected, however. The simplest interface circuit is a series resistor, to limit the current flowing through the upper protection diode, with parallel compensating capacitor to preserve the rise and fall times, as shown in Figure 10.

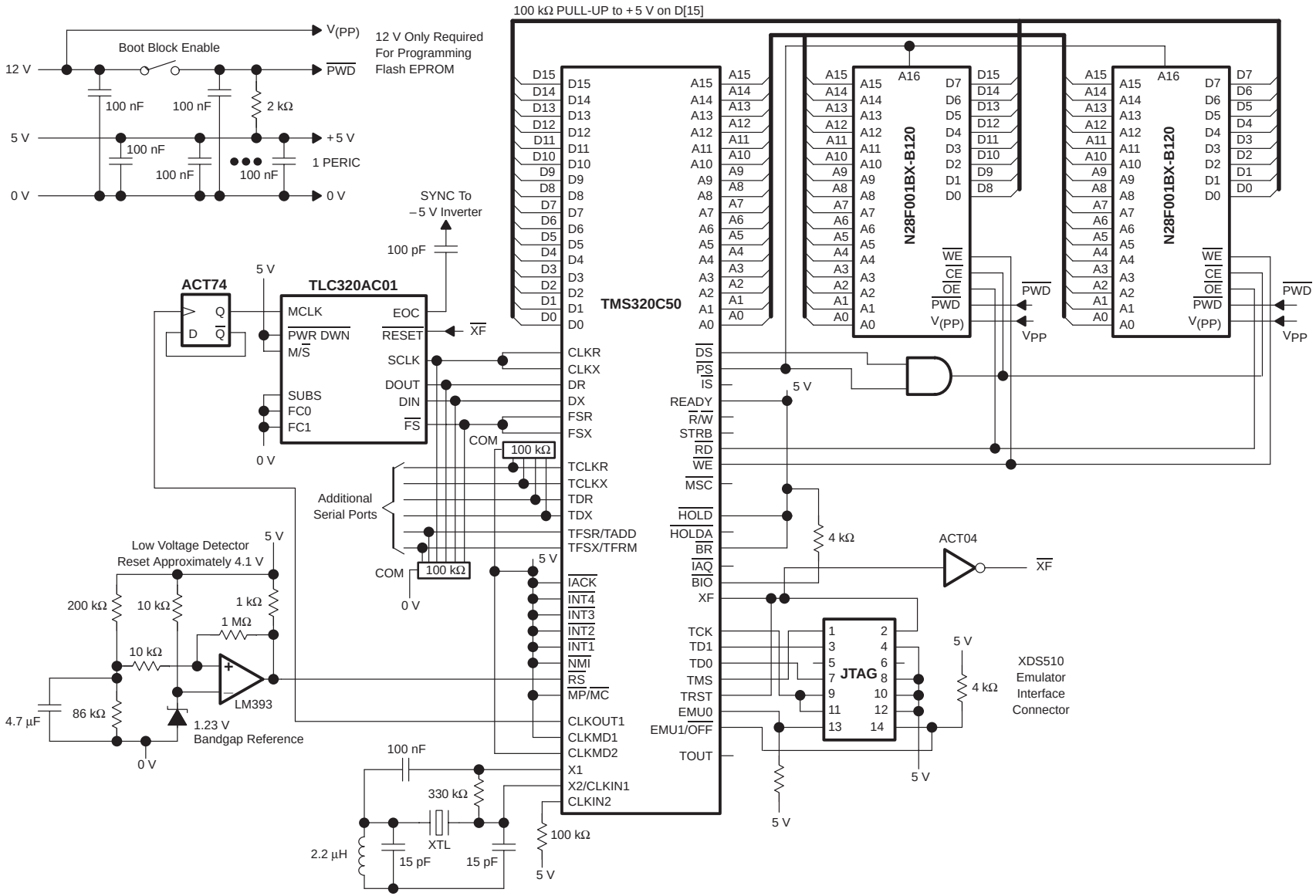


Figure 9. TLC320AC01 to TMS320C50 Hardware Schematic

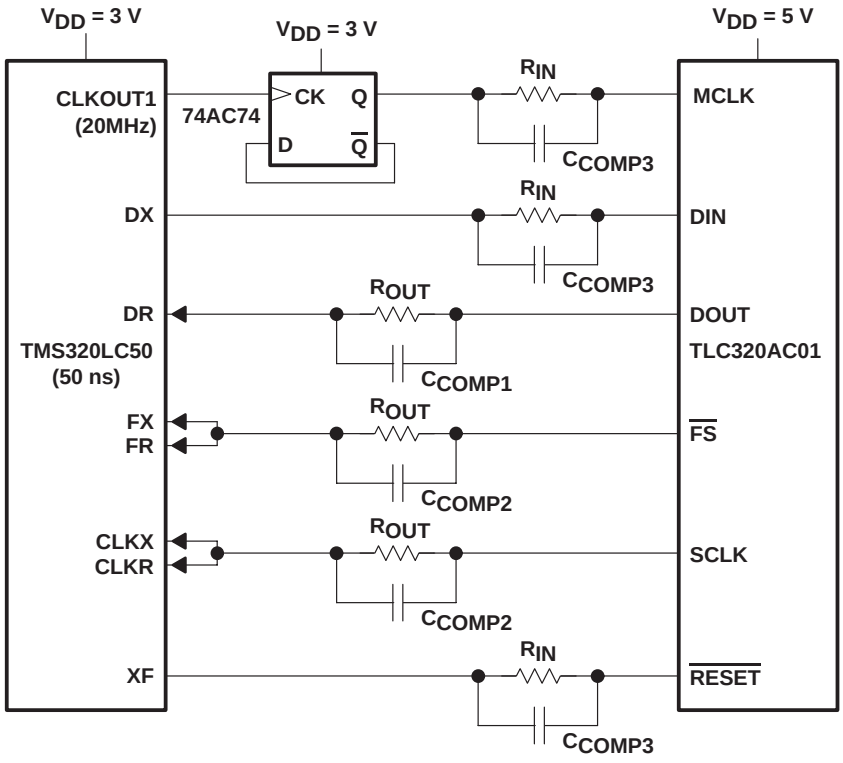


Figure 10. Interfacing to 3-V DSP

4.3.3 Calculation of Interface Component Values

Assuming that a transient input current through the protection diodes of 1 mA at power-up is safe for both devices and that the order of rise and fall is unknown. Then, 3.3 kΩ resistors (R_{IN}) in series with the TLC320AC01 inputs and 5.6 kΩ resistors (R_{OUT}) in series with the outputs provide full protection.

To calculate the appropriate compensation capacitor for signals from the TLC320AC01 to the TMS320C50, treat the parallel combination of C_{IN} and C_{STRAY} in conjunction with C_{COMP} as a capacitive divider where:

$$V_{OUT} = \frac{V_{IN} C_{COMP}}{C_{COMP} + C_{IN+STRAY}}$$

rearranging:

$$C_{COMP} = \frac{C_{IN+STRAY} V_{OUT}}{V_{IN} - V_{OUT}}$$

Substituting for worst case power supplies of 3.3 V and 4.5 V with one TMS320C50 input load of 15 nF and 10 pF stray capacitance:

For one input:

$$C_{COMP1} = \frac{(15 + 10) \times 3.3}{4.5–3.3} \approx 68 \text{ pF}$$

For two inputs:

$$C_{COMP2} = \frac{(15 = 15 + 10) \times 3.3}{4.5–3.3} \approx 120 \text{ pF}$$

These capacitor values should not be greatly increased because the input protection diodes of the TMS320C50 would then be driven into transient conduction on each rising logic edge.

The same method is applied to calculate the compensation capacitor for signals going to the TLC320AC01 (e.g., $\overline{\text{RESET}}$, MCLK and DIN). Assume a TLC320AC01 input capacitance of 5 pF and 10 pF stray capacitance, a worst case V_{DD} for the TMS320C50 and TLC320AC01 input threshold of 2.2 V.

$$C_{COMP3} = 68 \text{ pF}$$

This is a minimum value. C_{COMP} should be as large as possible for lowest power consumption and best noise margin. The maximum value of C_{COMP} depends upon the DSP chip power supply rise time. Switching three series connected 1.2 Ah NiCd cells with a total internal resistance of 30 mΩ into 200 µF of decoupling capacitance gives a maximum dV/dt of approximately 1 V/µs. In practice, wiring inductance and the resistance of protective fuses limit dV/dt to <0.1 V/µs.

$$C_{IN}\Big|_{MAX} = I_{IN}\Big|_{MAX} * \frac{dV_{DD}}{dt}\Big|_{MAX}$$

Therefore, C_{IN} should not greatly exceed 1 nF.

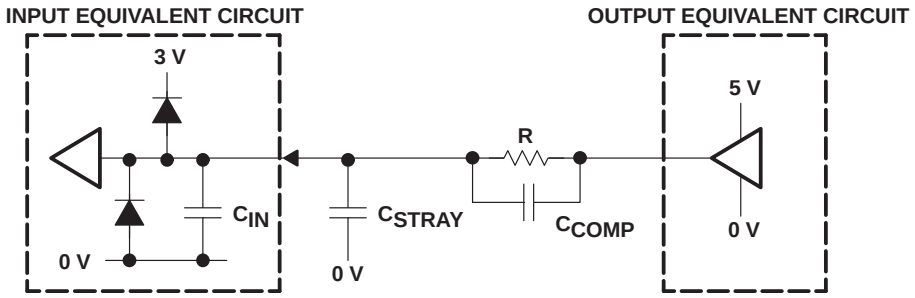


Figure 11. Interfacing to 3-V DSP – Component Values

4.4 Programming

4.4.1 Initialization

Only those registers that have to be changed from their defaults need to be reprogrammed. The initialization process consists of sending pairs of data values from the 16-bit synchronous serial interface of the DSP chip to the TLC320AC01. In most cases the first word of the pair will be 0000 0000 0000 0011B. The 14 most significant bits of this value (bits 15 to 2) specify that an output sample of zero be sent and the two least significant bits (bits 1 and 0) specify that the next word transmitted will be interpreted as a secondary communication.

The secondary data value is used to reprogram one of the nine registers. Bits 15 and 14, which control phase shifting in modem applications will usually be zero, bit 13 = 0 specifies that data is to be written to a register, bits 12 to 8 define the address of the register that is to be changed. Bits 7 to 0 contain the data to be stored in the register.

4.5 Register Descriptions

4.5.1 Pseudo Register 0 (no-op)

The main purpose of R0 is to allow phase shift commands to be sent as secondary communications without reprogramming any other register. It is not needed for most applications.

4.5.2 Register 1 (A Register)

The A register sets half the number by which the master clock input (MCLK) is divided to provide the switched capacitor filter clock (FCLK). This is also the principal method for setting the sampling frequency.

4.5.3 Register 2 (B Register)

The B register sets the ratio between the low-pass filter corner frequency and the sampling frequency. For most purposes the default, or a value close to it will be appropriate.

4.5.4 Register 3 (A′ Register)

The A′ register is used for phase shift control and can be ignored for most purposes.

4.5.5 Register 4 (Amplifier Gain Select Register)

This allows the gains of the analog input and output to be varied by –6 dB or –12 dB or disabled. The monitor output can be varied by –8 dB or –18 dB or disabled.

4.5.6 Register 5 (Analog Configuration Register)

This selects whether the high-pass filter is to be disabled, thus allowing the codec to respond to dc, and controls the input multiplexer.

4.5.7 Register 6 (Digital Configuration Register)

Control operating modes and power-down options. The defaults will be suitable for many applications.

4.5.8 Register 7 (Frame-Sync Delay Register)

This controls the timing of the serial data transmission of a slave converter in multi-channel multiplexed systems.

4.5.9 Register 8 (Frame-Sync Number Register)

This controls the number of frame-sync pulses generated, corresponding to the number of channels being multiplexed on to the bused serial interface. The default of 1 is suitable for single channel operation.

4.6 TLC320AC01/TMS320C50 Demonstration Program

The demonstration program AC01DEMO.ASM carries out the simplest possible operation, reading in a sample from the ADC in the TLC320AC01 and then writing it to the DAC. It is assembled and linked using the commands in the batch file MAKE.BAT. This is called as follows: MAKE AC01DEMO.

The program begins with the definition of some variables and allocation of their memory locations. The COFF assembler used here does not assign absolute addresses, but instead relative positions within named blocks of memory. The linker then resolves these references in conjunction with information stored in the linker command file AC01DEMO.CMD.

The next section of the program is the definition of macros that will be used later. Using macros makes the main program listing easier to understand by hiding some of the frequently repeated details.

The section called vectors is loaded into flash EPROM at address zero, which is where the TMS320C50 starts executing after reset. The .text section is the main body of the program, and would typically be loaded into the memory section flashp defined in the command file. In this example, however, it has been placed immediately after the vectors section in the protected bootstrap area for convenience of testing.

The main program starts by initializing certain processor registers that are undefined or have unsuitable defaults at start-up, then clearing the memory variables. The code which is to run in real-time is copied from flash EPROM to the on-chip single-access RAM block for maximum speed of execution. The serial port is initialized to use external clock and frame synchronization pulses, and to transfer 16-bit data words. The initial behavior of the serial port is unpredictable when it is reset with the frame sync input high, as is the case when the TLC320AC01 is inactive. Therefore, a dummy value of zero is sent, and afterwards the TLC320AC01 is again reset briefly. Now the interface is properly initialized and the program branches to the real-time processing loop.

The processor waits in a low power idle mode until an interrupt is received. Then it determines whether the interrupt was from the serial port, in which case it executes the processing loop. The processed results from the previous sample are written to the serial port data transmit register, then the fresh ADC data is read in from the data receive register and processed. The results are stored ready to be sent to the DAC on the next interrupt. This double buffering method maximizes the processing time available because processing can take place while serial data is being transmitted and received. The two least significant bits of the output data are masked out to ensure that a secondary communication request is not inadvertently sent.

The serial port receive interrupt routine simply sets a flag to indicate that data is available. There is no need to have a separate transmit interrupt because the transmit and receive operations are inherently synchronous with each other.

4.7 TMS320C50 Assembler Listing

```
    .title "'AC01 demonstration program"
    .width 200
    .version 50         ; Makes assembler generate C50 code
    .mmregs             ; Predefine names for memory mapped registers

; This program initializes the 'C50 processor and serial port, then initializes the '
; 'AC01 codec and starts the main signal processing loop. In this example, a data
; sample is read from the adc and written back to the dac unchanged.

; Using rev 6.40 or higher assembler tools, use the following make file

; @echo off
; if "%1" == "" goto  :nofile
; dspa %1.asm –x –w –s –v50 –1
; if not errorlevel 1 dsplnk %1.obj –o %1.out –m %1.map %1.cmd
' gpt  "dpme
; :nofile
; echo no source file!
; :done

; –s option makes all symbols global, and thus accessible to the emulator and
; simulator.
; –w option warns about pipeline conflicts.
; –x option makes a cross reference table.
; –1 option generates listing file

; If an eprom programmer is used it may also be necessary to use the DSPHEX conversion
; program to split the linker output file which is in COFF format into a pair of high
; and low byte files in HEX format.

FSAMP     .set 16      ; 8 seslects 8 kHz, 16 selects 16 kHz

; The following 2 variables must be in memory block b2 and dp set to 0 because they
; are accessed by direct addressing

gotdataflag  .usect    "b2", 1   ; used to signal that an interrupt came from the
                                 ; serial input port
outputbuffer .usect    "b2", 1   ; temporarily store output sample

; Macro definitions

waitint .macro
waitint?
       lacc   gotdataflag
       bz     waitint?         ; wait for semaphore to be changed
       splk   #0, gotdataflag  ; set it again
       .endm

progreg .macro progval
       splk   #11b, dxr        ; request secondary comms
       waitint                 ; wait for transmission
       splk   #:progval:, dxr  ; send value
       waiting                 ; wait . . . .
       .endm

       .sect "vectors"         ; vectors is the starting point of a block of program
                               ; memory starting at address zero

Interrupt vectors – these start at address zero
; unused interrupts branch to themselves so that if they are inadvertently activated
; they can be identified using the xds510 emulator
rs     b  mainentry
intl   b  intl
int2   b  int2
int3   b  int3
tint   b  tint
rint   b  getdata
xint   b  xint
trnt   b  trnt
txnt   b  txnt
int4   b  int4
rsvd14 b  rsvd14
rsvd16 b  rsvd16
rsvd18 b  rsvd18
rsvd1A b  rsvd1A
rsvd1C b  rsvd1C
```

```
rsvd1E  b  rsvd1E
rsvd20  b  rsvd20
trap    b  trap
nmi     b  nmi
rsvd26  b  rsvd26
rsvd28  b  rsvd28

       .text     ; .text indicates start of main program storage block in flash eprom

mainentry        ; this is the startu entry point
   ldp    #0     ; data page pointer to page zero
   setc   INTM   ; globally disable interrupts
   setc   SXM    ; set sign extension mode
   setc   OVM    ; set saturation on arithmetic overflow

; Disable address visibility (to save power by not driving address bus)
; set up on-chip single access ram and B0 to be in data space for initialization.
   splk   #0000000010101000b, PMST
   circ   CNF    ; B0 is in data space

; Set up wait state control registers for 2 wait states when accessing flash eprom
   splk   #00000b,  CWSR
   splk   #1010101010101010,  PDWSR

   splk   #0, gotdataflag     ; zero data received flag
   splk   #0, outputbuffer    ; zero output storage buffer

; relocate speed critical part of program to on chip ram
   lrlk   AR1, 800h           ; address in data memory of start of ram block
   larp   AR1
   lacc   #ocramstart         ; address in flash memory of start of code
   rpt    #ocramend–ocramstart–1
   tblr   *+

   apl    #1111111111011111b, PMST      ; remove ram from data space
   opl    #0000000000100000b, PMST      ; put it in program space

; set all interrupt masks except serial receive
   splk   #000010000b, IMR

; set up serial port
   splk   #0, dxr             ; zero the data transmit register
   splk   #0001000b, SPC      ; use ext clock and frame sync
   opl    #0c0h, SPC          ; take it out of reset

; clear all interrupt flag bits
   splk   #0ffffh, IFR
   clrc   intm                ; enable interrupts

   clrc   xf        ; release codec from reset
waitint             ; waiat for interrupt from serial port

; this code assumes that XF is inverted in hardware
; reset the codec and release it again to make it ignore first garbage word
; generated by serial port in revision 1 'C50 silicon
   setc   xf
   rpt #10          ; hold 'AC01 reset low for at least 1 MCLK period
   nop              ; ie > 100 ns for MCLK = 10 MHz
   clrc   xf

; serial interface and 'AC01 are now in a stable state

; setup codec – only need to reprogram those registers that need to be changed from
; their defaults
                                    ; reg 0 = no op
                                    ; reg 1 = A register (18 = default)
      .if FSAMP = 8
         progreg 0000000101000101b  ; 36 –> 8 kHz @10.368 MHz clockin
      ;endif                        ; 35 –> 7.937 kHz @10 MHz

      .if FSAMP = 16
         progreg 0000000100100010b  ; 18 –> 16 kHz @10.368 MHz clockin
      .endif                        ; 17 –> 16.34 kHz @10 MHz
```

```
          progreg   0000001000010010b      ; reg 2 = B register (18 = default)
;                   ||||||||||||||||
;                   |||||||++++++++----- data
;                   |||+++++------------- address
;                   ||+------------------ 0 = write
;                   ++------------------- Phase shift

                                           ; reg 3 = A' register
          progreg   0000010000011001b      ; reg 4 = amplifier gain select
;                   ||||||||||||||||
;                   ||||||||||||++----- output     0=sq, 1=0 dB, 2=-6 dB, 3=-12 dB
;                   ||||||||||++------- input      0=sq, 1=0 dB, 2=+6 dB, 3=+12 dB
;                   ||||||||++--------- monitor    0=sq, 1=0 dB, 2=-8 dB, 3=-18 dB
;                   ||||||++----------- no used
;                   |||+++++------------- address
;                   ||+------------------ 0 = write
;                   ++------------------- Phase shift
          progreg   0000010100000101b      ; reg 5 = analog configuration
;                   ||||||||||||++----- 0=loopback, 1=norm i/p, 2=aux i/p, 3=both
;                   ||||||||||||+------- 0=hp filter on, 1=hp filter off
;                   ||||||||||+--------- 0=echo off, 1=echo on
;                   |||||||++++--------- not used
;                   |||+++++------------- address
;                   ||+------------------ 0=write
;                   ++------------------- Phase shift

                                           ; reg 6 = digital configuratjon
                                           ; reg 7 = frame sync delay
                                           ; reg 8 = frame sync number

    b   passthrough      ; branch to real-time code in on-chip ram
; This section is relocated to on-chip single access ram block for faster operation

       .sect "ocram"
       .label ocramstart   ; this label referes to the address where the following
                           ; code is stored in eprom, not the address from which it
                           ; is executed

passsthrough               ; this label refers to the execution address in on-chip ram
; Main signal processing loop

       clrc   intm         ; Wait for any interrupt, determine whether
       nop                 ; it is caused by serial data input and branch
       idle                ; back to idle if not.
       setc   intm         ; WARNING - The manipulation of INTM and the
       lacc   gotdataflag  ; nop, idle sequence are necessary to prevent
       bz     passthrough  ; serial interrupts from being missed if they
       clrc   intm         ; occur just after another interrupt!
       splk   #0, gotdataflag  ; clear the data received flag

       out outputbuffer, dxr   ; write the data derived from the previous input sample
                               ; to the serial port data transmit register
       lacc   drr    ; read a codec input sample from the serial port data receive
                     ; register. Data is in low accumulator
       ; do the signal processing here, leaving result in accumulator
       ; . . .
       ; . . .
       ; . . .

       and    #1111111111111100b  ; mask out bottom two bits to ensure that secondary
                                  ; communications are not accidentally requested

       sacl outputbuffer          ; save the result of the prcessing until the next
                                  ; interrupt, and only then write it to the serial
                                  ; port. This maximizes the processing time available.

       b      passthrough
; Interrupt handlers
; because the transmit and receive operations of the 'AC01 are synchronous
; only one serial port interrupt handler is needed
```

```
getdata   splk   #1, gotdataflag  ; set a flag to indicate data available
          rete   ; return from interrupt, restoring context and re-enabling interrupts

          .label ocramend  ; end of block transferred to on-chip ram
.end
```

4.8 Linker Command File: AC01DEMO.CMD Listing

```
MEMORY       /* memory map for C50 */
{
page 0  :/* program memory */
reset   : origin = 0,   length = 800h /* booth block up to start of on chip ram +/
onchipp :origin = 800h, length = 2400h /* on chip program memory*/


flashp : origin = 8000h, length = 7200h /* top half flash prog except b0 */
param2 : origin = 3000h, length = 1000h /* second parameter block in eprom */
                              /* par. block is overlaid by on-chip ram */

page 1 : /* data memory */
   b3  : origin = 60 h, length = 20h
   b0b1   :origin = 800h, length = 240h /* ocram is on-chip data if OVLY=1 */
                               /* external flash eprom if OVLY=0 */

}

SECTIONS
{      vectors   : load = reset page 0
       .text  : load = flashp page 0
       param2 : load = param2 page 0
       ocram  : load = flashp page 0 run = onchipp page 0

       be     : load = b2 page 1 /* data page 0 on-chip ram */
       .bss   : load = b0b1 page 1
}
```

4.9 Measuring the DAC Filter Response with a White Noise Generator

A convenient way of measuring the frequency response of a linear system is to excite it with white noise and measure the response with a spectrum analyzer. This example shows how white noise can be generated by a very short random bit generator program and used to measure the response of the TLC320AC01's DAC reconstruction filter.

The random bit generator implements a recurrence relation in a primitive polynomial modulo 2 of order 31 (reference 1). This gives a maximal length sequence of pseudo-random bits which only repeats after 2^{31} –1 iterations. The polynomial used is $x^{31} + x^3 + x^0$, although there are many others to choose from. A 32-bit variable, noise_sr, which is initially seeded with any non zero value, stores the state between iterations. On each iteration, the accumulator is loaded from noise_sr and shifted left one bit. The most significant bit (now in the carry bit) is then exclusive ORed (XOR) with the remaining non zero terms. Each bit that has been XORed is stored back into the same location in the accumulator and the result is saved. This is implemented by testing the carry bit after the shift with the execute conditional (XC) instruction. If C was 0, do nothing because anything XORed with 0 is 0 and bit zero of the accumulator is filled with a 0 after a shift. If C was 1, XOR the low accumulator with the constant 10010b which achieves the desired result.

There are three main limitations to this technique. Because the XOR is only carried out on the 16 least significant bits there must not be any non zero terms in the polynomial above x15 apart from x31. The contents of the accumulator should not be used directly as a random number because successive values are correlated as the bits work their way to the left. This is overcome in the example by iterating the code 14 times for each sample. Although the noise has a white long-term spectrum (that is equal power per unit frequency), it is nongaussian. This does not matter for frequency response measurements.

Figure 12 shows the response of the TLC320AC01 reconstruction filter measured at a sampling rate of 7.937 kHz. The A register value is 42, the B register value is 15 and the MCLK frequency is 10 MHz.

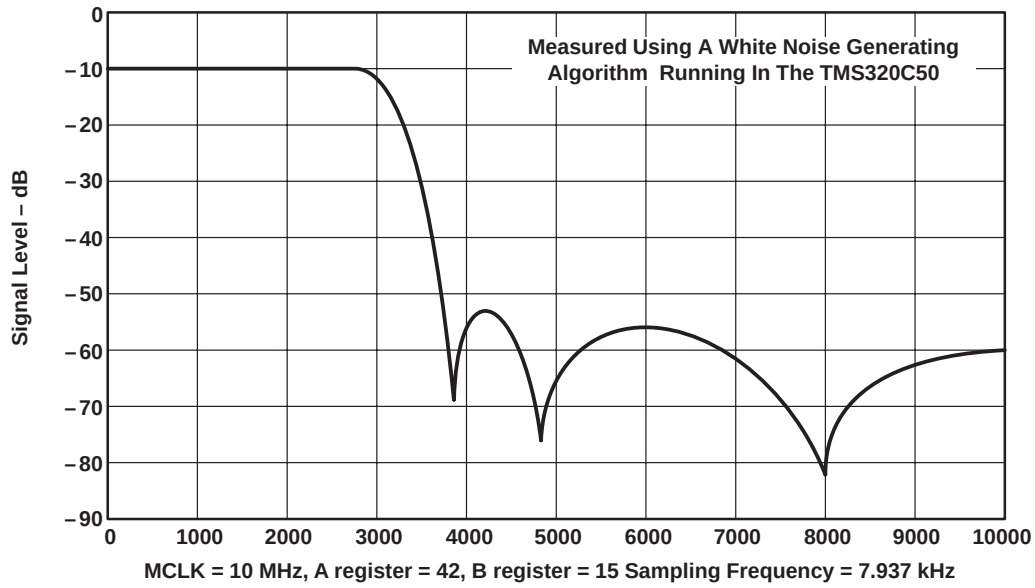


Figure 12. DAC Channel Frequency Response

4.10 Example Noise Generator Code Listing

```
noise_sr      .usect “b2”, 2   ; allocate 32 bits of data memory
;. . .
       larp   AR1    ; seed random bit generator with 2
       lrlk   AR1, noise_sr
       lac    #2
       sacl   *+
       zac
       sacl   *

; . . .

; execute this section once per dac output sample
       larp   AR1
       lrlk   AR1, noise_sr
       lacl   *+           ; load low accumulator from data memory
       add    *, 16        ; load high accumulator
       splk   #13, brcr    ; repeat block 14 times to decorrelate sequential bits
       rptb   end_noise – 1
       sfl                 ; need a 1 cycle gap between sf1 and xc to
       nop                 ; allow for pipeline delay
       xc     1, C         ; execute next instruction if carry set
       xor #10010b         ; xor bit 3 with bit 31, copy bit 31 to bit 0
end_noise

       sach   *–           ; save high accumulator to data memory
       sacl   *            ; save low accumulator

; Write low accumulator to transmit data register (after masking out bottom 2 bits)
```

5 SAMPLING AND QUANTIZATION – TUTORIAL

5.1 Sampling

5.1.1 Ideal Sampling

In converting a continuous time signal into a discrete digital representation, the process of sampling is a fundamental requirement. In an ideal case, the sampling signal is a train of impulses (infinitesimally narrow with unit area). The frequency of these impulses is the sampling rate (f_s). The input signal can also be idealized by considering it to be truly band limited, containing no components in its spectrum above a certain frequency.

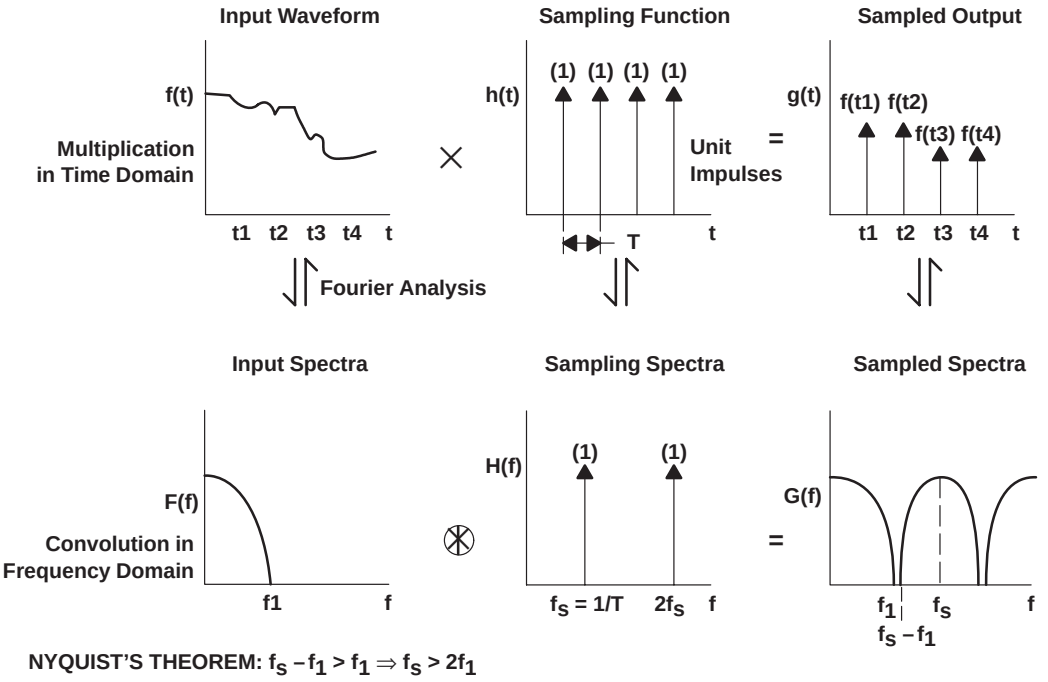


Figure 13. Ideal Sampling

The ideal sampling condition is shown in Figure 13, represented in both the frequency and time domains. The effect of sampling in the time domain is to produce an amplitude modulated train of impulses representing the value of the input signal at the instant of sampling. In the frequency domain, the spectrum of the pulse train is a series of discrete frequencies at multiples of the sampling rate. Sampling convolves the spectrum of the input signal with that of the pulse train to produce the combined spectrum shown, with double sidebands around each discrete frequency which are produced by the amplitude modulation. In effect, some of the higher frequencies are folded back so that they produce interference at lower ones. This interference causes distortion which is called aliasing. Aliases cannot be removed by subsequent processing.

As shown in the diagram, if the input signal is band limited to a frequency f_1 and is sampled at frequency f_s, the overlap (and hence aliasing) cannot occur if

$$f_1 < f_s - f_1 \text{ or } 2f_1 < f_s$$

Therefore, if sampling is performed at a frequency at least twice as great as the maximum frequency of the input signal, no aliasing occurs and all of the signal information can be extracted. This is *Nyquist's Sampling Theorem*, and it provides a basic criterion for the selection of the sampling rate required by the converter to process an input signal of a given bandwidth.

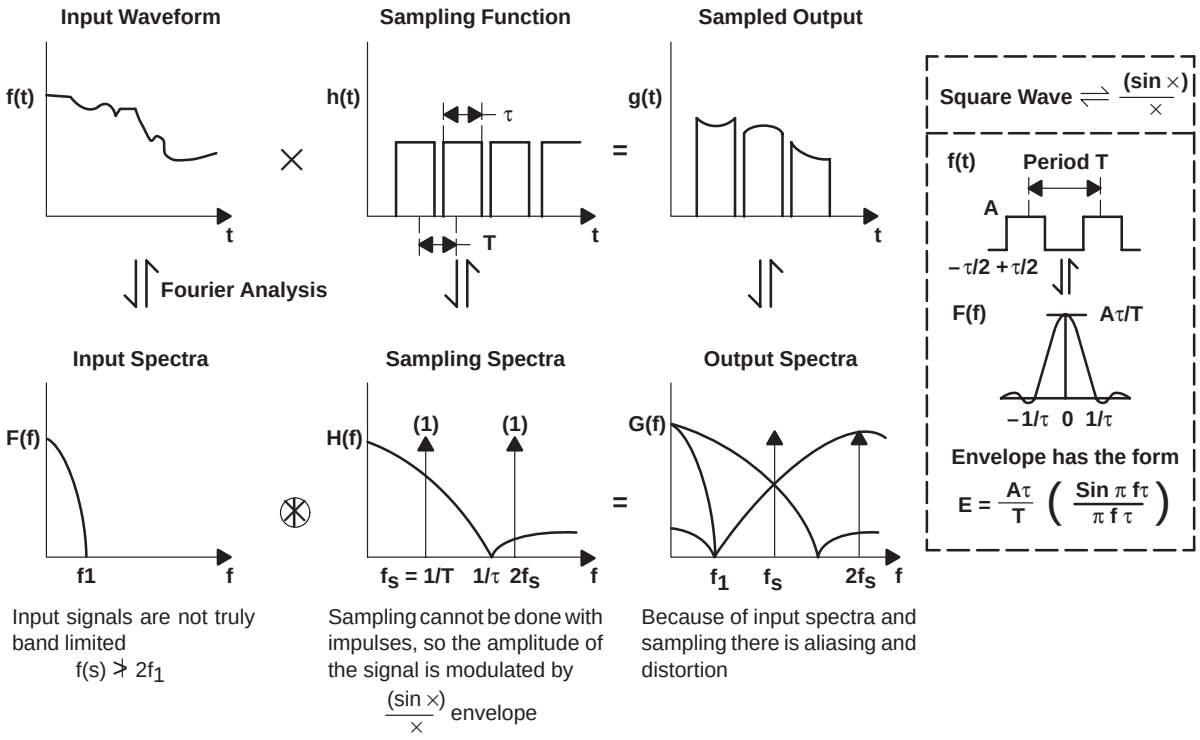


Figure 14. Real Sampling

5.1.2 Real Sampling

The concept of an impulse is a useful one to simplify the analysis of sampling. However, it is a theoretical ideal which can be approached but never reached in practice. Instead the real signal is a series of pulses with a period equalling the reciprocal of the sampling frequency. The result of sampling with this pulse train is a series of amplitude modulated pulses.

Examining the spectrum of a square wave pulse train shows a series of discrete frequencies, as with the impulse train, but the amplitude of these frequencies is modified by an envelope which is defined by (sin x)/x (sometimes written sinc(x)) where x in this case is πf_S. For a square wave of amplitude A, the envelope of the spectrum is defined as

$$\text{Envelope} = A\left(\frac{\tau}{T}\right)\left[\sin(\pi f_S \tau)\right]/\pi f_S \tau$$

The error resulting from this can be controlled with a filter which compensates for the sinc envelope. This can be implemented as a digital filter, in a DSP, or using conventional analog techniques. (The TLC320AC01 analog interface circuit has an on-chip (sin x)/x correction filter after its DAC output for this purpose.)

5.1.3 Aliasing Effects and Considerations

In practice, any real signal has infinite bandwidth. However, the energy of the higher frequency components become increasingly smaller so that at a certain value they can be considered to be irrelevant. This value is a choice that must be made by the system designer.

As shown, the amount of aliasing is affected by the sampling frequency and by the relevant bandwidth of the input signal, filtered as required. The factor that determines how much aliasing can be tolerated is ultimately the resolution of the system. If the system has low resolution, the noise floor is already relatively high and aliasing can have an insignificant effect. However, with a high resolution system, aliasing can increase the noise floor considerably and therefore needs to be controlled.

As shown, increasing the sampling rate is one way to prevent aliasing. However, there is a limit on what frequency this can be, determined by the type of converter used and also by the maximum clock rate of the digital processor receiving and transmitting the data. Therefore, to reduce the effects of aliasing to within acceptable levels, analog filters must be used to alter the input signal spectrum.

5.2 Theoretical SNR for a 14-Bit Device

The analog input to an ADC is a continuous signal with an infinite number of possible states, whereas the digital output is by its nature a discrete function with a number of different states determined by the resolution of the device. It follows from this therefore, that in converting from one form to the other, certain parts of the analog signal that were represented by a different voltage on the input, are represented by the same digital code at the output. Some information has been lost and distortion has been introduced into the signal. This is quantization noise.

For an ideal staircase transfer function of an ADC, the error between the actual input and its digital form has a uniform probability density function if the input signal is assumed to be random. It can vary in the range of ±1/2 least significant bit (LSB) or ±q/2 where q is the width of one step.

$p(\varepsilon) = 1/q$ for $-q/2 \leq \varepsilon \leq +q/2$

$p(\varepsilon) = 0$ otherwise

The average noise power (mean square) of the error over a step is given by

$$E^2(\epsilon) = \frac{1}{q} \int_{q/2}^{+q/2} \epsilon^2 d\epsilon$$

which gives $E^2(\varepsilon) = q^2/12$

The total mean square error, N^2, over the whole conversion area is the sum of each quantization level's mean square multiplied by its associated probability. Assuming the converter is ideal, the width of each code step is identical and therefore has an equal probability. Hence for the ideal case

$$N^2 = \frac{q^2}{12}$$

Considering a sine wave input F(t) of amplitude A so that

$F(t) = A \sin \omega t$

which has a mean square value of $F^2(t)$, where

$$F^2(t) = \frac{1}{2\pi} \int_0^{2\pi} A^2 \sin^2(\omega t) dt$$

which is the signal power. Therefore the signal-to-noise ratio (SNR) is given by

$$\text{SNR(dB)} = 10 \text{ Log} \left[\left(\frac{A^2}{2} \right) \Big/ \left(\frac{q^2}{12} \right) \right]$$

But $q = 1 \text{ LSB} = 2A/2^n = A/2^{n-1}$

Substituting for q gives

$$\text{SNR} = 10 \text{ Log} \left[\left(\frac{A^2}{2} \right) \Big/ \left(\frac{A^2}{3 \times 2^{2n}} \right) \right] = 10 \text{ Log} \left(\frac{3 \times 2^{2n}}{2} \right)$$

$$\Rightarrow 6.02n + 1.76 \text{ dB}$$

This gives the ideal value for a perfect n-bit converter and shows that each extra bit of resolution provides approximately 6 dB improvement in the SNR. In practice, errors in the ADC introduce non-linearities that lead to a reduction of this value.

For a perfect 14-bit converter, the SNR is:

$$6.02 \times 14 + 1.76 \approx 86 \text{ dB}$$

5.3 References

1. William H. Press, Brian P. Flannery, Saul A. Teukolsky and William T. Vetterling, 1988, Numerical Recipies in C – The Art of Scientific Computing, (Cambridge: Cambridge University Press) pp 224–228.
2. TLC320AC01 Analog Interface Circuit Data Manual; SLAS057A
3. TMS320C2x User's Guide; SPRU014C
4. TMS320C5x User's Guide; SPRU056B
5. TMS320 Fixed-Point DSP Assembly Language Tools; SPRU018C