

MICROPROCESSOR INTERFACE

INTERFACING TO MICROPROCESSORS

Very often, digital information must be exchanged between a data acquisition component and a microprocessor. A fast-growing application area is the microprocessor-based measurement and control system. Let's examine a generalized system to illustrate the interface of a digital-to-analog or analog-to-digital converter to a processor. Afterwards, we will explore some specific interfaces to some of the more popular microprocessors.

The central processing unit is the heart of the system. It requests instructions prepared by the programmer, calls for data, and makes decisions related to the instructions. Based on the data, the processor then determines appropriate actions to be performed by other parts of the system. Since there are many peripherals within a given system, the microprocessor must be capable of selecting a particular device. It identifies each device by means of a unique address code. A typical microprocessor might have 16 binary address lines providing 65,536 addressing codes. Data to and from the processor is carried across a bidirectional 8 or 16 bit wide data bus. Many processors also provide a serial data path. Several microprocessors use a multiplexed address/data bus on which both address and data are transmitted on the same signal paths. In this case, the first portion of the bus cycle transmits the address. Data transfer occurs later in the cycle. This architecture is popular for microprocessors with an 8-bit data bus.

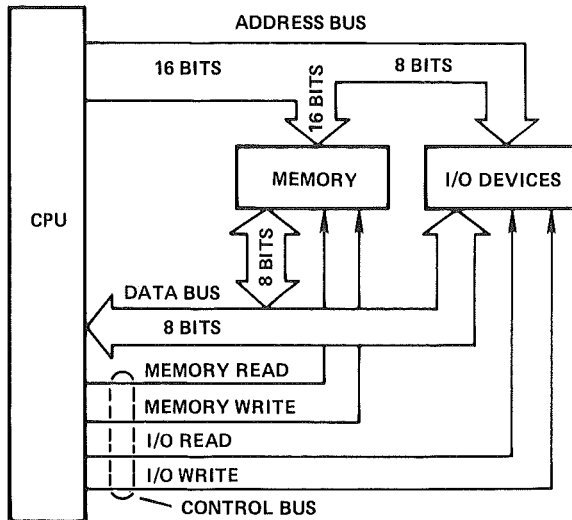
All microprocessors have memory which is usually external. Digital signal processors often have some internal memory to complement external memory. The memory is used by the CPU as its primary source of instructions and also as a means to store information generated for later use. Many of the address locations in a typical system are storage locations in the memory. When a memory location is addressed, the memory may store the information that resides on the data bus. This is called MEMORY WRITE. Addressing stored information to be placed on the data bus for use by the CPU constitutes a MEMORY READ. In addition, the microprocessor provides the system with control information. The control bus issues read or write command signals and indicates a valid address.

Another important link in the system is the set of Input-Output (I/O) interfaces. These interfaces include all the information channels between the system and the real world. There are digital ports through which programs and control commands may be loaded and from which digital data may be transmitted. Many systems also use analog signal ports to measure and control external real-world phenomena.

Digital I/O may be interfaced to the processor by means of one or more of the following three techniques:

- Accumulator or Isolated I/O
- Memory Mapped I/O
- Direct Memory Access

ACCUMULATOR OR ISOLATED I/O



The diagram illustrates the internal architecture of a computer system. On the left, a vertical bar represents the **CPU**. To its right are two main components: **MEMORY** and **I/O DEVICES**.

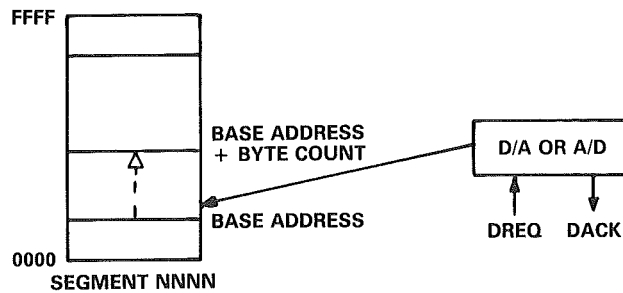
- ADDRESS BUS:** A horizontal line at the top, labeled "ADDRESS BUS". It has two arrows pointing from the CPU to both the MEMORY and I/O DEVICES, each labeled "16 BITS".
- DATA BUS:** A horizontal line below the address bus, labeled "DATA BUS". It has a bidirectional arrow between the CPU and MEMORY labeled "8 BITS". It also has two arrows pointing from both MEMORY and I/O DEVICES to the CPU, each labeled "8 BITS".
- CONTROL BUS:** A horizontal line at the bottom, labeled "CONTROL BUS". It has two control lines: "MEMORY READ" (indicated by a dashed line) and "MEMORY WRITE" (indicated by a solid line). These lines connect to the CPU and both MEMORY and I/O DEVICES.

Accumulator I/O simplifies address decoding. Separate control signals isolate all I/O addresses from memory addresses. In 8085 and Z80 systems, only 256 input and output addresses are available. This 8-bit addressing for I/O increases program execution speed and simplifies programming. Isolated I/O allows flexibility in system hardware design. The system designer has the ability to combine different I/O devices within the I/O space regardless of their speed. Slower I/O will require Wait States which extend the bus cycle thus slowing throughput. The processor extends the current bus cycle until the I/O device issues a READY or DTACK indicating that the cycle should be terminated. Since isolated I/O must pass all data through the accumulator, programming flexibility is limited.

For memory mapped I/O, each input or output function is treated as a location of memory. Full decoding of all address bus signals provides more I/O capability at the expense of address decode complexity. Memory mapped I/O can transfer data without the use of the accumulator thus facilitating ease of programming.

Direct memory access (DMA) is the fastest method for data transfer. Throughputs of 360kbytes/sec are achievable for many microprocessors using DMA. The DMA controller must know where in memory the data from the requesting device is to be stored. This is referred to as the "Base Address". In addition the "Byte or Word Count" or number of items to be stored must be known. Each time the controller processes a DMA request, it "robs" a bus cycle from the processor, issues the appropriate address to memory, and sends an acknowledge signal to the requesting device. The requesting device now can gate its data onto the processor's data bus. The controller will then increment the base address and decrement the byte or word count awaiting the next DMA operation. No software interaction is required permitting the processor to perform other tasks. In DMA operation, only sequential groups of data may be transferred per DMA channel. Many controllers can, however, accommodate several DMA channels. In the case of converter interfacing using DMA, the converter usually represents a single byte or word being transferred to or from a block of memory.

DEVICE-TO-MEMORY DMA MEMORY DIAGRAM



1-BYTE SOURCE TO N-BYTE DESTINATION

GENERAL CONSIDERATIONS FOR INTERFACING

Several key parameters must be determined to successfully interface a D/A or A/D converter to a microprocessor. Power supply requirements, logic compatibility, timing parameters, support hardware, and data formats all need careful consideration. Let's explore some of these issues in greater detail.

GENERAL INTERFACE CONSIDERATIONS

- Power Supply Requirements
- Logic Compatibility
- Timing
- Hardware
- Data Formats

POWER SUPPLY REQUIREMENTS

With the increasing popularity of the personal computer, it seems natural that they have become a prime candidate to host a data acquisition system. The majority of these computers provide ± 12 Volt power supplies to support RS-232 interfaces and may be used to power low current external devices. Many older data acquisition products, however, are not capable of operation with supplies less than ± 15 Volt $\pm 5\%$ and some may also require a $+5$ Volt source. Today however, manufacturers have addressed this problem with products that are specified and tested with both 12 and 15 Volt supplies. Moreover, some newly developed monolithic products are capable of single supply operation at $+12$ or $+5$ Volts. In either case, to achieve optimal performance, proper grounding and decoupling in conjunction with good PC board layout should be incorporated. Quite often, a switching type supply will generate substantial electromagnetic interference to degrade accuracy of the data acquisition system. The user should refer to publications dealing with shielding techniques to alleviate this problem.

POWER SUPPLY REQUIREMENTS

- Dual Supply Operation
- Single Supply Operation
- Low Noise

LOGIC COMPATIBILITY

Today's microprocessor compatible data conversion products require 5 Volt TTL digital inputs. A logic level high is guaranteed 2.4V minimum with a logic level low at 0.8V max. Some older CMOS D/A converters such as the AD7523 and AD7524 have different logic levels for each specified power supply voltage. The AD7524, for example, when powered by a 5V supply, will have TTL compatible inputs. When powered by a +15V supply, the logic levels will shift to +13.5V minimum and 1.5V maximum for logic high and logic low respectfully. In any system, the designer should ensure that the digital logic has adequate output drive to meet the needs of the converter.

LOGIC COMPATIBILITY

- 5 Volt TTL or CMOS
- Logic Levels as a Function of Supply Voltage
- Low Digital Input Currents

TIMING PARAMETERS

Special attention to timing parameters during the design stage of a data acquisition system will limit problems during the production phase. Control signal pulse widths, set-up times, and hold times should conform with timing specifications for the device. Avoid designing to "typical specifications" as these parameters could vary from device to device. If operation over a wide temperature range is expected, shifts in timing may occur. Some CMOS D/A converters will exhibit power supply dependent timing specifications. These devices will generally have faster timing specifications with V_{DD} at 15 Volts as compared to 5 Volts. In some applications, all control inputs to data acquisition component may not be used. Unused inputs should be hardwired high or low per the device specification.

TIMING PARAMETERS

- Avoid Designing to Typical Specifications
- Analyze Temperature Effects
- Analyze Power Supply Voltage Effects
- Terminate Unused Digital Inputs

SUPPORT HARDWARE

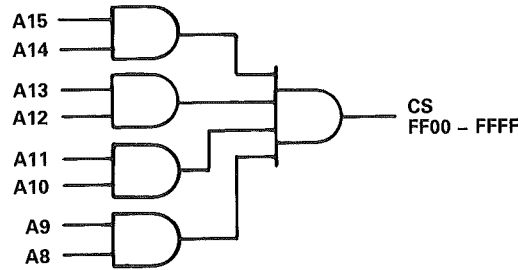
In order to interface a D/A or A/D converter to a processor, additional hardware is usually required. External logic may be used to condition the digital control signals from the processor. Microprocessor compatible D/A converters incorporate on-board data latches. A/D converters with microprocessor compatibility also include these latches with three-state outputs.

SUPPORT HARDWARE

- Address Decoding

The amount of external hardware for address decoding is determined by the I/O technique employed and the number of peripherals in the system. Memory mapped I/O will require more address decode logic than accumulator I/O since all address signals must be decoded. A simple approach for Memory Mapped I/O would be decoding of only the upper 8 address lines of a 16-bit address bus. If, for example, the upper 8 address lines are gated as shown below, any one of 256 addresses from FF00H through FFFFH will decode. The disadvantage is that one peripheral occupies 256 memory locations wasting 255 which could be used for other functions. The same decode logic used with an 8086 or 8088 microprocessor will accommodate accumulator I/O when the upper 8 address lines are replaced by their lower counterparts. These processors can address any one of 256 port locations through the direct I/O instruction and can be interfaced to peripherals using the I/O, read and write control signals. Many other possibilities exist and depend upon available hardware, board space, and system architecture.

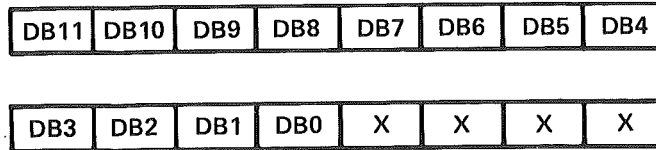
DECODE LOGIC FOR UPPER 8 ADDRESS LINES OF 16-BIT BUS



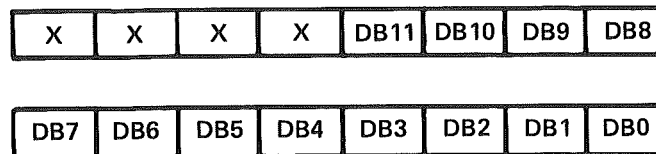
DATA FORMATS

The vast majority of today's microprocessors incorporate either an 8- or 16-bit data bus. An 8-bit peripheral such as a D/A or A/D converter connects directly to an 8-bit bus; however, higher resolution devices require some form of data formatting. A 12-bit D/A connected to an 8-bit bus must have a double-buffered input structure. Two input registers are loaded; the first with the 8 MSBs and the second with the 4 LSBs. The contents of these registers are then loaded into the DAC register at which time the analog output updates. The entire transfer is accomplished in two write operations. The format for the 12 bits of data may be left or right justified. Left justification places the MSB (DB7) at the MSB position of the first address with the LSB (DB0) appearing as DB4 for the next address. Right justification places the MSB on DB3 of the first address with the LSB appearing as the LSB (DB0) of the next address. Some CMOS D/A converters, such as the AD7548, allow the flexibility of either left or right justification via TTL control. Others, such as the AD667, offer this flexibility through pin-strapping. Interfacing a data acquisition device with resolution less than 16 bits to a 16-bit data bus also requires the designer to select a format. The device is wired to the processor's bus for either left or right justification.

LEFT JUSTIFIED



RIGHT JUSTIFIED



A serial data format provides a convenient means of data transmission over long distances. In large systems, a serial data format can often simplify backplane wiring and allow more signals to be passed over the bus. Moreover, serial data acquisition devices have smaller package outlines permitting a higher circuit density. All serial devices require a clock to strobe the data. The CMOS AD7543 12-bit DAC, packaged in a 16-pin DIP, uses an external clock whereas the AD575 10-bit A/D operates from an internal or external clock.

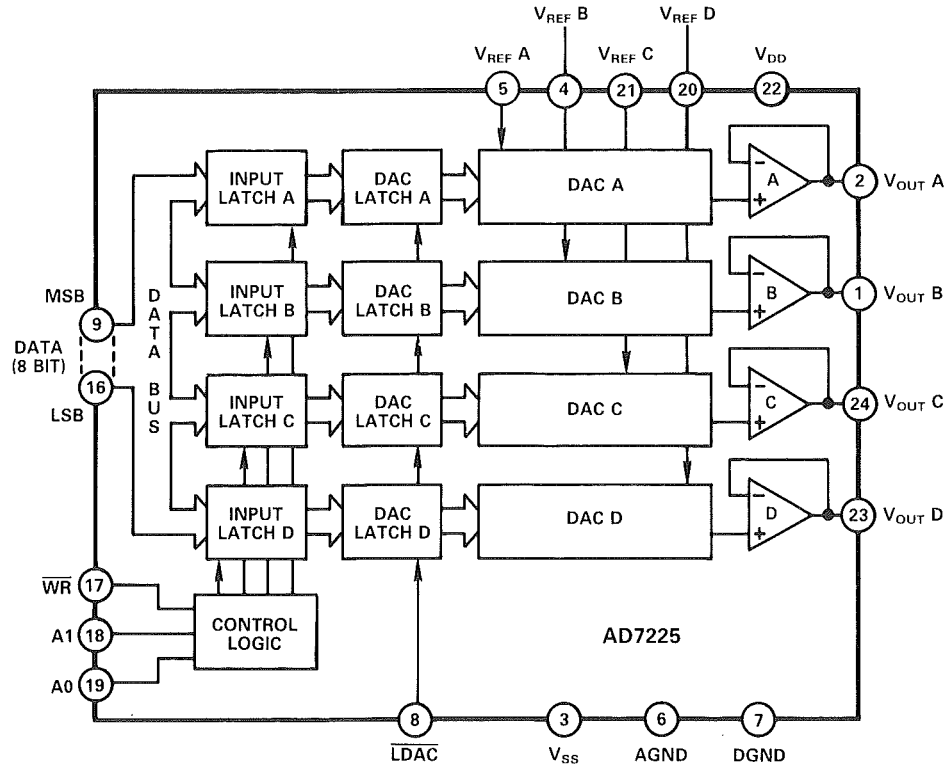
INTERFACING A DIGITAL-TO-ANALOG CONVERTER

Digital-to-analog converters are easily interfaced to microprocessors. Address decoding and some external logic are all that is required to successfully interface the device. Some examples follow to illustrate this point.

8-BIT DATA BUS INTERFACING

AD7225 LC²MOS Quad 8-Bit DAC with Separate References

AD7225 FUNCTIONAL BLOCK DIAGRAM



The AD7225 contains two registers per DAC, an input register and a DAC register. Address lines A0 and A1 select which input register will accept data from the data bus. When the $\overline{WR}$ is LOW the input latches of the selected DAC are transparent. The data is latched into the addressed input register on the rising edge of $\overline{WR}$.

The data held in the DAC register determines the analog output of the converter. The rising edge of $\overline{LDAC}$ controls data transfer from input register to DAC register for all four DACs. The $\overline{LDAC}$ signal is level triggered and therefore the DAC registers may be made transparent by tying $\overline{LDAC}$ LOW. $\overline{LDAC}$ is an asynchronous signal independent of $\overline{WR}$ but proper hold time for $\overline{LDAC}$ must be given to latch the correct data.

Interfacing the AD7225 to an 8085 or 8088 microprocessor is easily accomplished. These processors employ a multiplexed address/data bus. An external latch such as the 8282 latches the lower address byte on the falling edge of ALE. The upper address byte is valid the entire bus cycle and is also decoded with the lower address byte to configure the AD7225 as memory mapped I/O. A0 and A1 connect directly to the DAC to select one of four input registers to be loaded. The high speed digital interface of the AD7225 is compatible with 8MHz processors.

8085A/
8088

A15

A8

WR

ALE

AD7

AD0

ADDRESS DECODE

LATCH

EN

ADDRESS BUS

ADDRESS/DATA BUS

AD7225*

A0

A1

LDAC

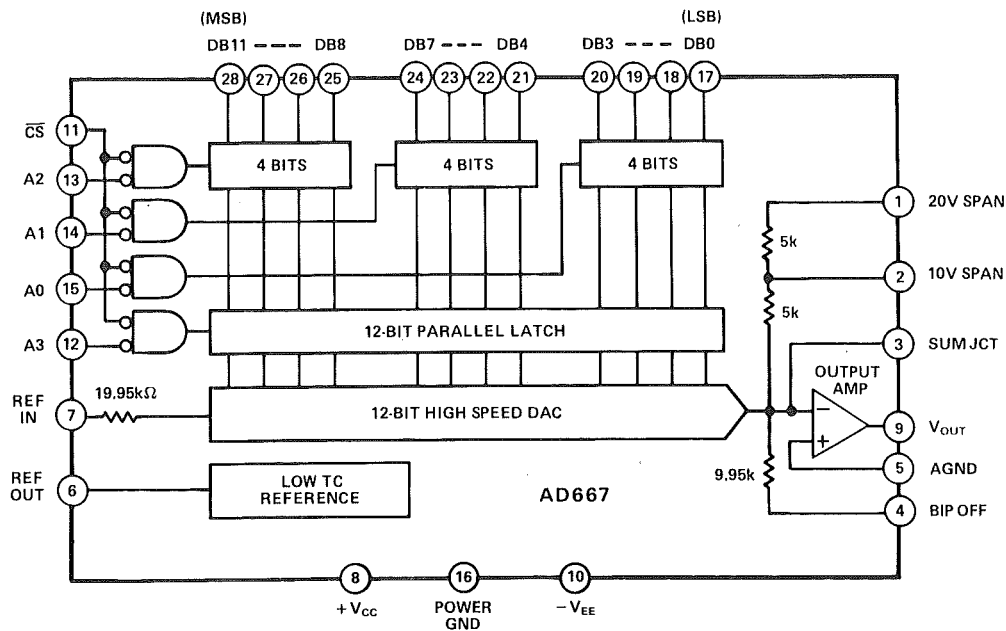
WR

DB7

DB0

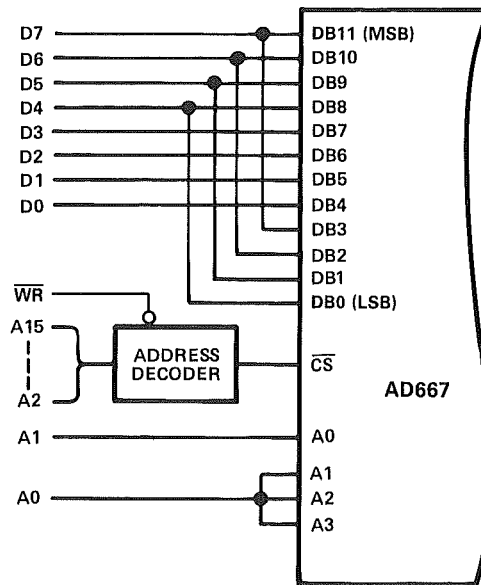
*LINEAR CIRCUITRY OMITTED FOR CLARITY

AD667 FUNCTIONAL BLOCK DIAGRAM



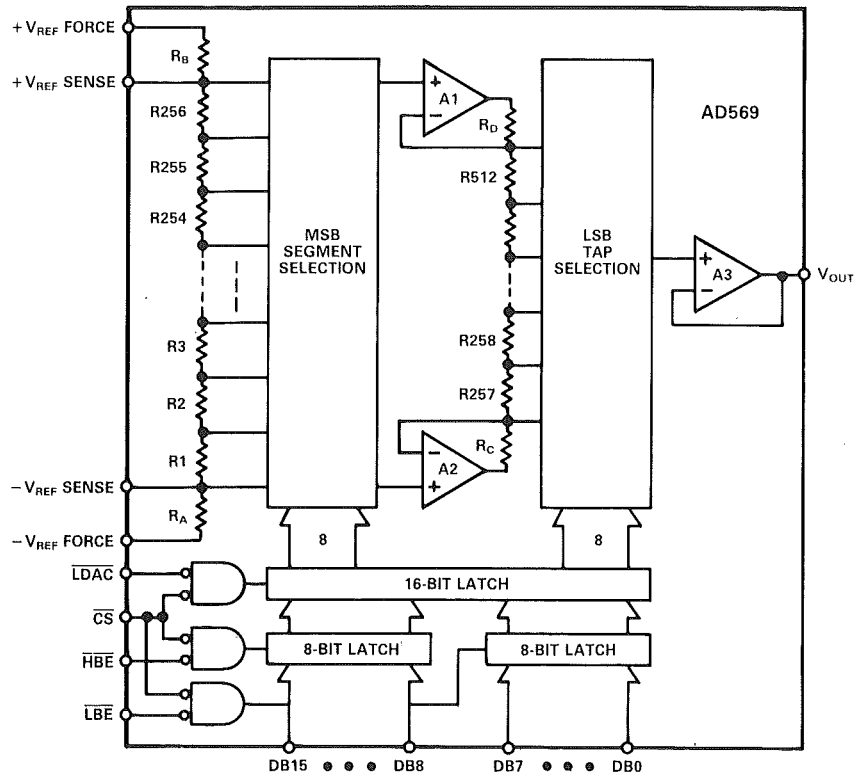
VIII - 7

LEFT-JUSTIFIED 8-BIT BUS INTERFACE



AD569 16-Bit Microprocessor Compatible D/A Converter

AD569 FUNCTIONAL BLOCK DIAGRAM



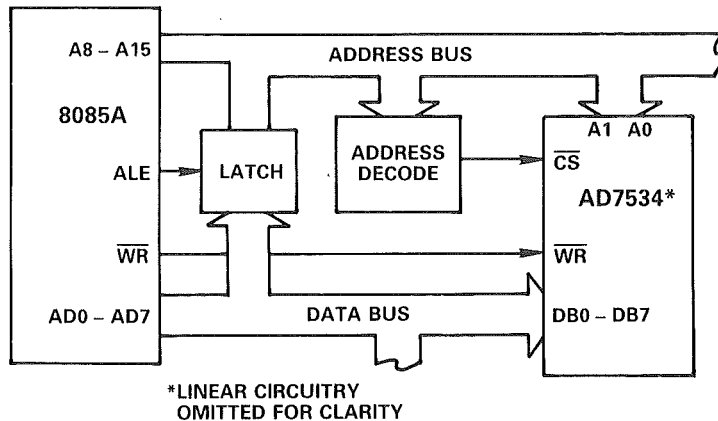
Instrumentation and process control systems often require high resolution converters (>12 bits) to be interfaced directly to a computer. In many instances a personal computer with an 8-bit data bus such as the IBM PC will serve as the controller. Unless the converter is specifically designed to accommodate this narrow bus external hardware will be required. Obviously one solution is to select a converter such as the AD569 with a double buffered input structure. Two write cycles are required to load (in any sequence) the high byte and low byte input registers and to update the DAC register. In addition the high speed control inputs of the AD569 permit operation with today's high speed processors.

The AD7534 14-bit D/A converter is configured to accept right-justified data in two bytes from an 8-bit data bus. Standard chip select and memory write logic is used to access the converter. Address lines A0 and A1 control the internal register loading and transfer.

A typical interface circuit for the AD7534 and the 8085A microprocessor is given below. The microprocessor treats the converter as four memory locations being selected by address lines A0 and A1. A sample program for loading the D/A with a 14-bit word is shown. The converter resides at locations 3000 through 3003.

The six MSBs are written into location 3001, and the eight LSBs are written to 3002. A write instruction to 3003 loads the full 14-bit word to the DAC register and updates the analog output.

AD7534 – 8085A INTERFACE



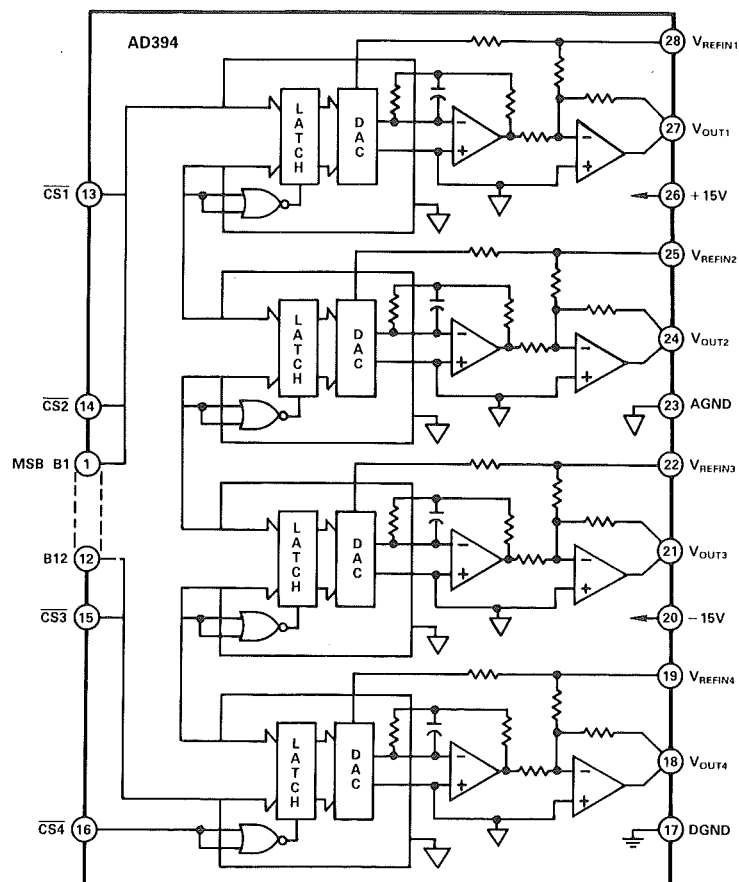
PROGRAM LISTING FOR AD7534 – 8085A INTERFACE

Address	Op-Code	Mnemonic
2000	26	MVIH, # 30
01	30	
02	2E	MVIL, # 01
03	01	
04	3E	MVIA, # "MS"
05	"MS"	
06	77	MOV M, A
07	2C	INR L
08	3E	MVIA, # "LS"
09	"LS"	
0A	77	MOV M, A
0B	2C	INR L
0C	77	MOV M, A
200D	CF	RST I

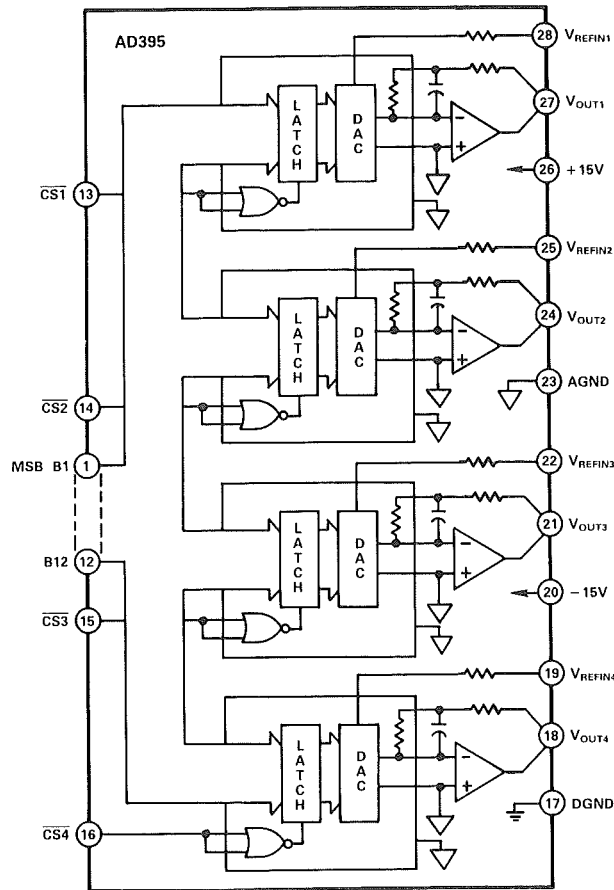
16-BIT DATA BUS INTERFACING

AD394/395 Quad 12-Bit D/A Converter

AD394 (BIPOLAR) FUNCTIONAL BLOCK DIAGRAM



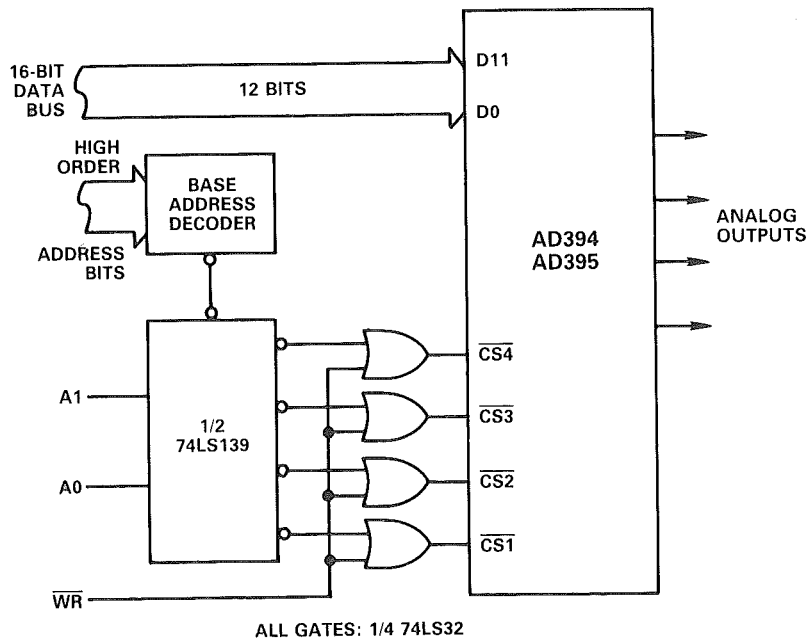
AD395 (UNIPOLAR) FUNCTIONAL BLOCK DIAGRAM



The AD394/395 quad 12-bit D/A converters are easily interfaced to 16-bit wide data buses. The devices' individual latches allow for multi-DAC interfacing to a single data bus.

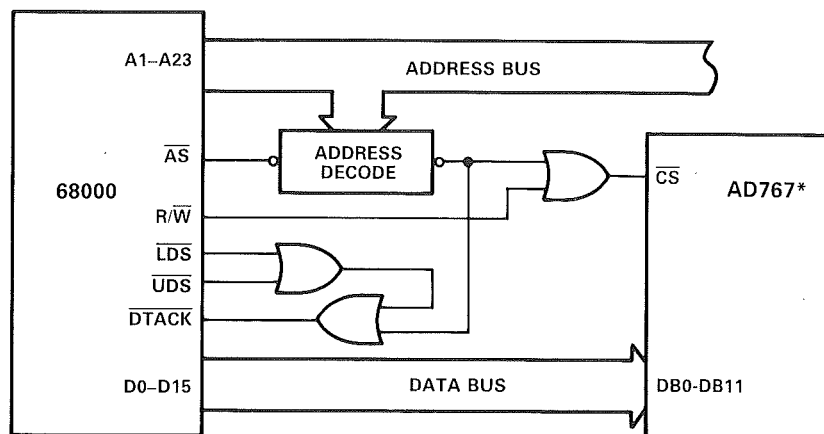
As shown below, a system write signal is used to control the decoded address lines and a 74LS139 decoder driven from the least significant address bits provides the $\overline{CS1}$ through $\overline{CS4}$ control signals. Address lines A0 and A1 each select a single DAC of the four contained within the AD394 or AD395. The use of a separate address line for each DAC allows several DACs to be simultaneously accessed. The address lines are gated by the microprocessor $\overline{WR}$ (400ns minimum) and the appropriately decoded base address.

AD394, AD395 16-BIT BUS INTERFACE

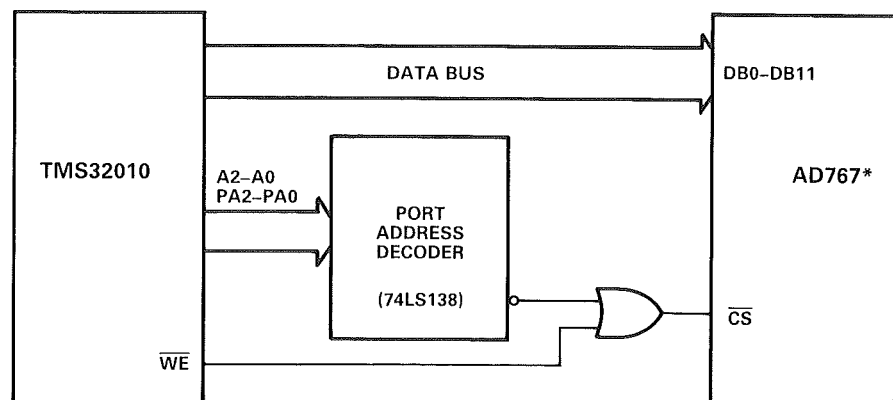


AD767 12-Bit Microprocessor Compatible D/A Converter

68000 – AD767 INTERFACE

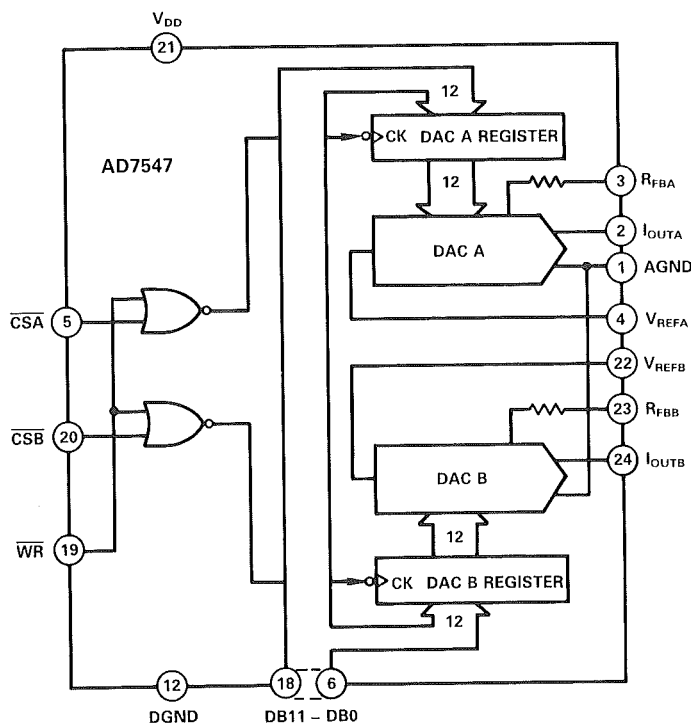


TMS32010 – AD767 INTERFACE



AD7547 DUAL 12 Bit D/A Converter

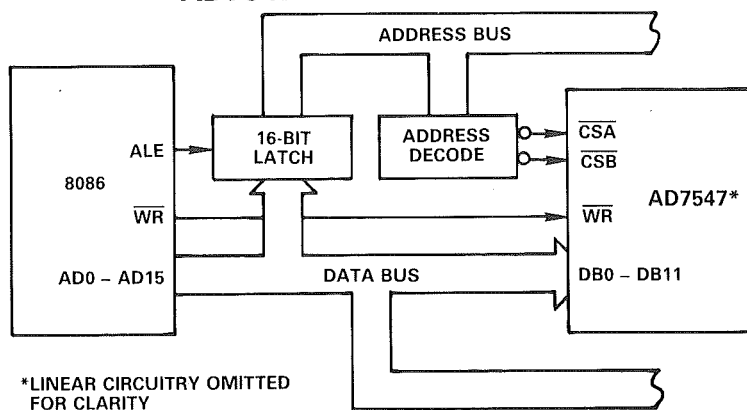
AD7547 FUNCTIONAL BLOCK DIAGRAM



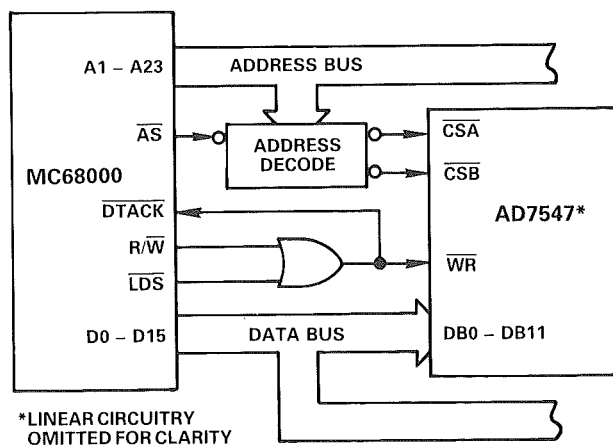
The AD7547 is designed for direct interfacing to 16-bit microprocessors with a minimal amount of external hardware. Shown below are interface circuits for two of the most popular 16-bit microprocessors, the 68000 and the 8086.

Data is loaded into the DAC registers on the rising edge of an 80ns minimum WR pulse. Data setup and hold time are 60 and 25ns respectively.

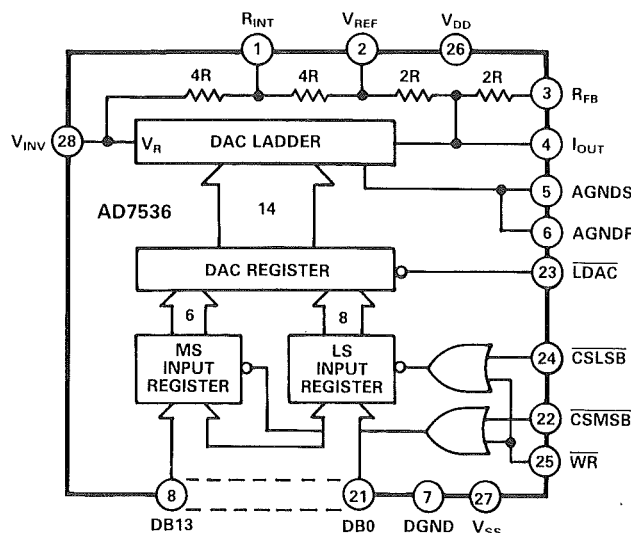
AD7547 – 8086 INTERFACE



AD7547 – MC68000 INTERFACE

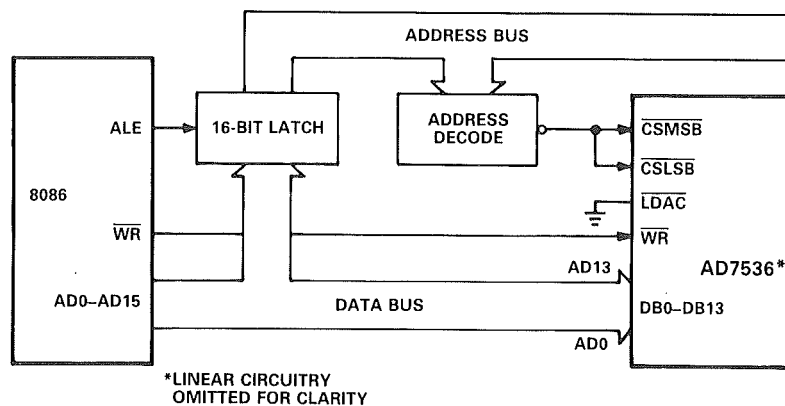


AD7536 FUNCTIONAL BLOCK DIAGRAM



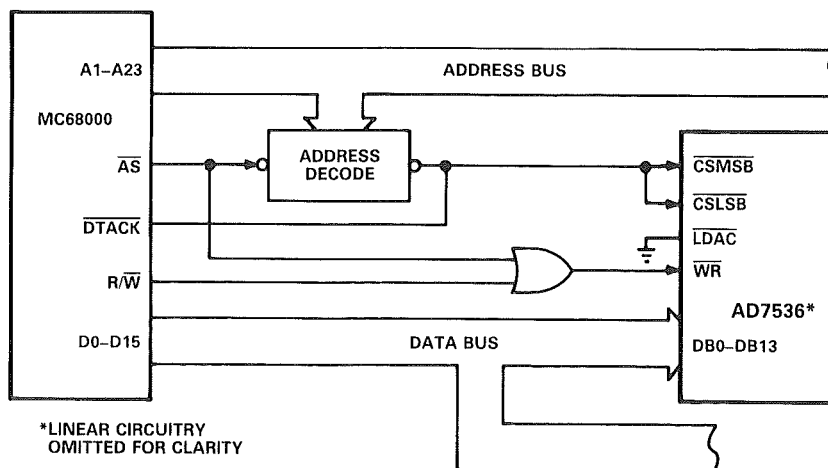
The flexibility of the AD7536 loading structure allows interfacing to both 8 and 16-bit microprocessors. As shown the DAC is interfaced to an 8086 16-bit processor without using the AD7536 double buffering feature. AD0 through AD13 of the 16-bit multiplexed address/data bus are connected to the DAC data bus (DB0–DB13). The 14-bit word is written to the DAC in one MOV instruction updating the analog output.

AD7536 – 8086 INTERFACE CIRCUIT



Interfacing between the 68000 and the AD7536 is accomplished using the circuit shown below. Once again, the double buffering feature of the DAC is not used.

AD7536 – MC68000 INTERFACE



INTERFACING A/D CONVERTERS

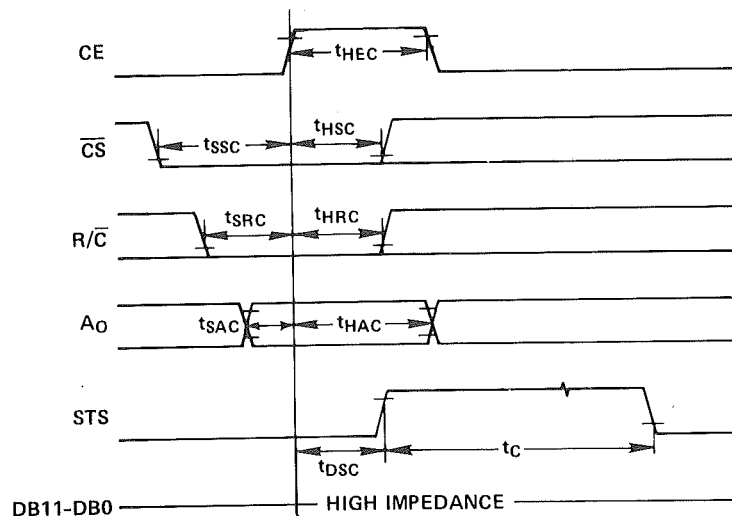
The general considerations for interfacing DACs to microprocessors also apply to ADCs. Whereas a DAC interface consists of a unidirectional transfer of data and control information, the interface of an ADC is bidirectional. Let's investigate this bidirectional interface and some of the common problems associated with it.

TIMING PARAMETERS

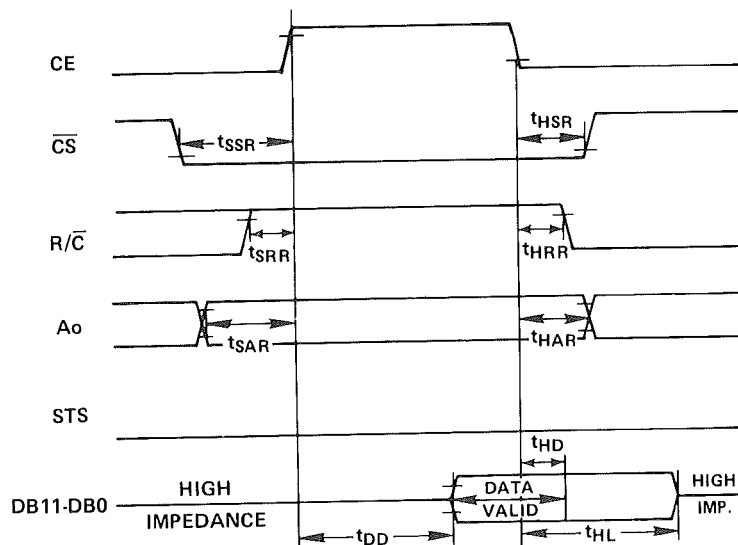
Early microprocessors operated with clock frequencies ranging from 1 to 4MHz and generated control signals with pulse widths of several hundred nanoseconds. As a result, interfacing to an A/D required a handful of logic gates and a few lines of software. Today, however, with microprocessor clock frequencies approaching 25MHz, the designer must carefully analyze all timing parameters. Many older A/D converters will not interface directly with high speed processors due to their faster bus timing. These converters can be interfaced with the addition of a timer or a peripheral interface adapter placed between the processor and converter.

Understanding the converter's timing diagram is the key to a successful interface. Shown below is a typical convert start timing diagram for a 12-bit A/D converter. Note that all timing specifications are referenced to the rising edge of chip enable, (CE). $\overline{CS}$, $R/\overline{C}$, and A_0 must be asserted prior to the rising edge of CE and remain valid sometime thereafter to start a conversion. Read cycle timing functions is similar with additional specifications (bus access t_{DD} and bus float delay t_{HL}) associated with the bidirectional aspects of the interface.

CONVERT START TIMING



READ CYCLE TIMING



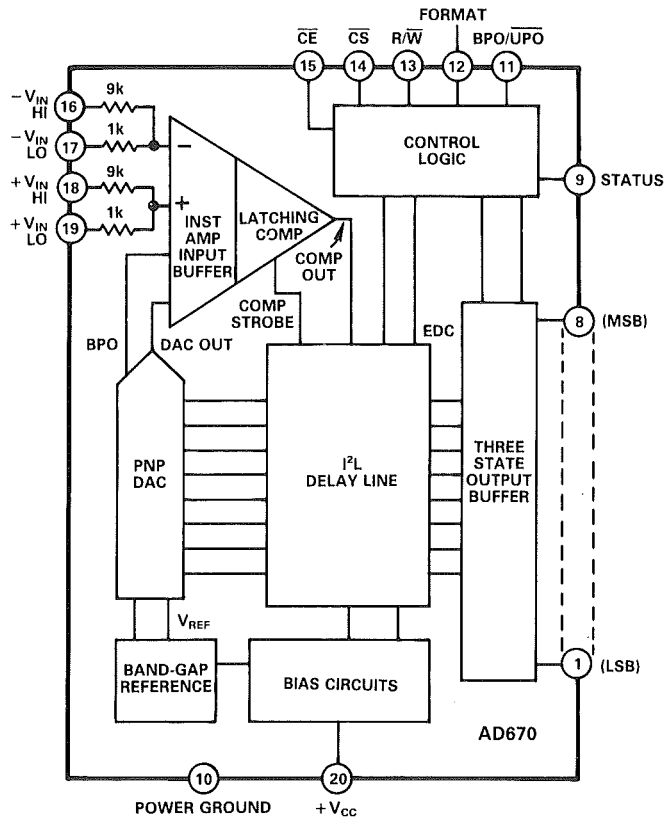
Bus access time defines the amount of time required for the converter's data bus to change from high impedance to valid data. This specification becomes most critical with high speed digital signal processors which may require access times of 100ns or less. Recent developments in converter technology, such as the AD7572 with 90ns access time, have made progress in bus access.

Along with bus access time one must also consider the bus float delay. Bus float delay is defined as the amount of time required for the converter to return the data bus to the high impedance state. This parameter is important to prevent bus contention when dealing with high speed processors. Today, bus float delays of 75ns are becoming the rule rather than the exception. Devices like the AD7572 with its 75ns maximum hold time alleviate many bus interface problems and minimize additional hardware.

8-BIT DATA BUS INTERFACING

AD670 Signal Conditioning 8-Bit ADC

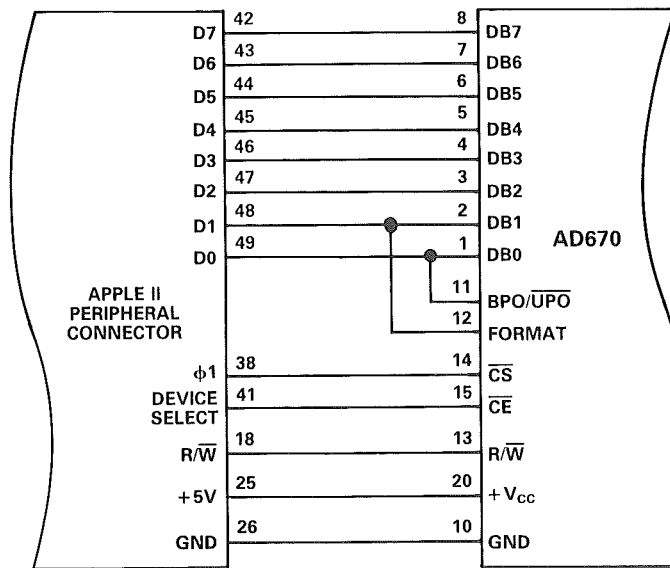
AD670 FUNCTIONAL BLOCK DIAGRAM



In many applications low level signals from transducers require signal conditioning prior to analog-to-digital conversion. A 1mV to 100mV signal by itself cannot utilize the full dynamic range of an 8-bit 10V full scale converter. If however, the signal is amplified by 1000, the same 8-bit converter can now provide meaningful data. Typically an external instrumentation amplifier serves as the gain stage but the AD670 incorporates an internal instrumentation amplifier. With its pin-programmable amplifier, the AD670 can resolve either 1mV or 10mV per LSB. In addition, a TTL compatible control signal allows the user to select unipolar or bipolar input signals.

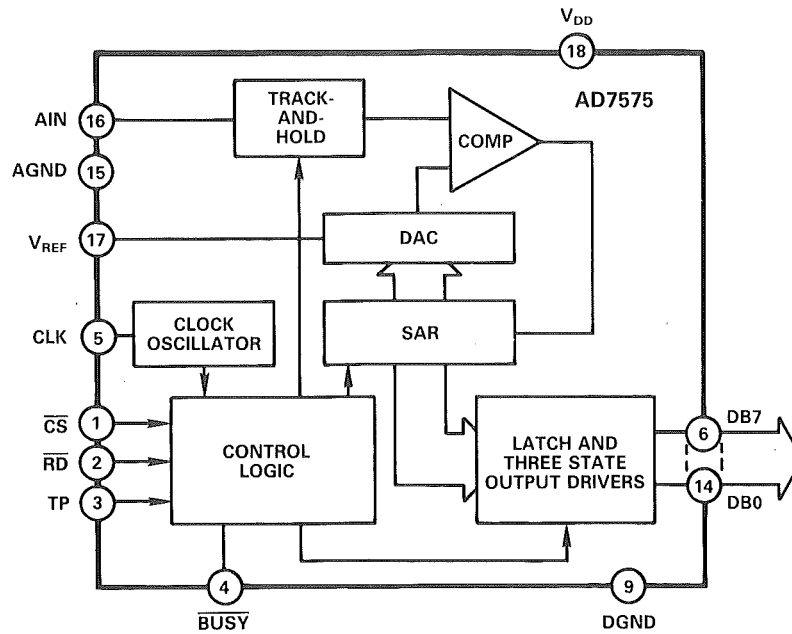
Most signal conditioning applications will use dedicated microcontrollers or the slower microprocessors. In many instances interfacing to a personal computer which provides access to control and data bus may be convenient. The Apple II is shown interfaced to the AD670. A unique feature of this interface is the ability to select data format via software. A POKE 49360, 2 statement prior to conversion selects a 2's complement data format.

APPLE II INTERFACE



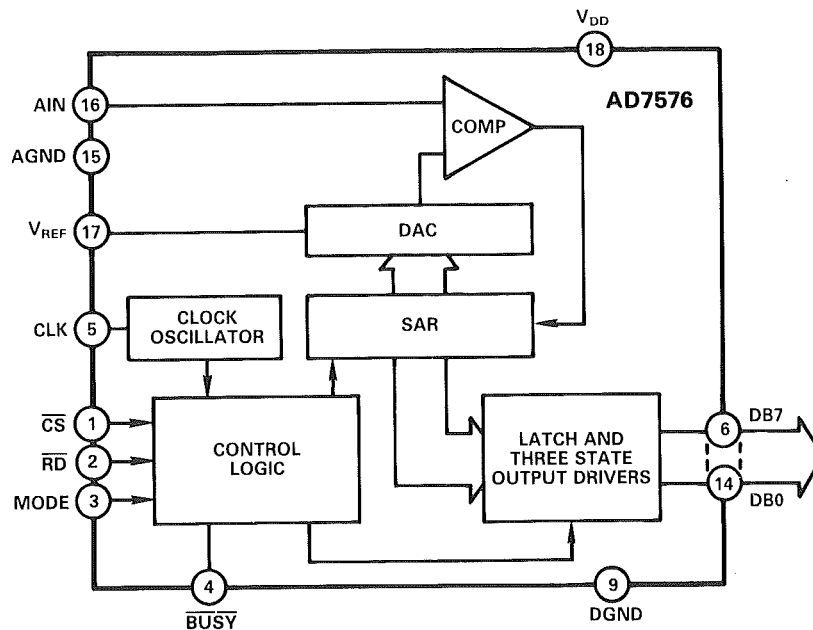
AD7575/76 LC²MOS Microprocessor Compatible ADC

AD7575 FUNCTIONAL BLOCK DIAGRAM



The AD7575 and AD7576 family of 8-bit converters offers complete flexibility when interfacing to 8-bit processors. These devices can be easily configured as a memory mapped device or can be interfaced as SLOW-MEMORY or ROM. Each mode of interfacing will be described in detail.

AD7576 FUNCTIONAL BLOCK DIAGRAM



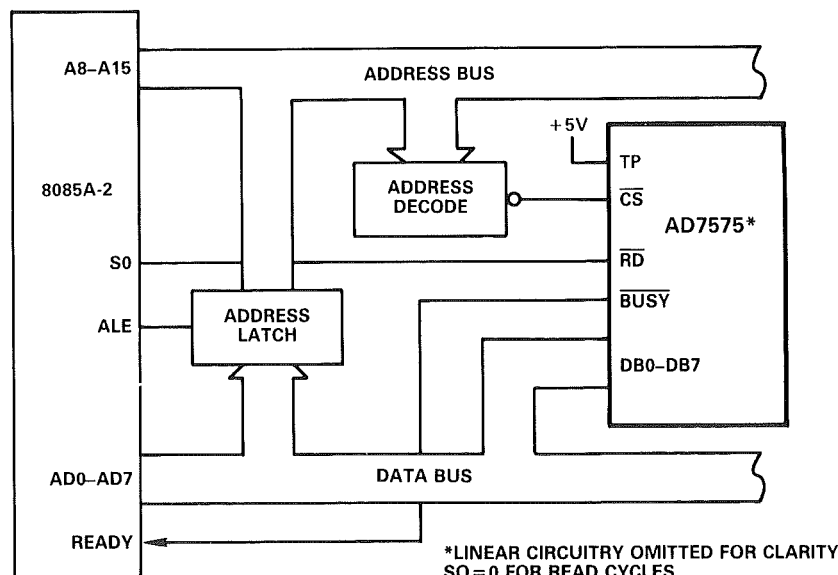
SLOW-MEMORY INTERFACE

The first interface option is designed for use with microprocessors such as the 8086 or 68000 which can be forced into a WAIT STATE for a minimum of 5 microseconds for the AD7575 and 10 microseconds for the AD7576. The processor starts a conversion and is halted until valid data is read from the converter. Conversion is initiated by executing a memory READ to converter's address bringing $\overline{CS}$ and $\overline{RD}$ LOW. $\overline{BUSY}$ goes LOW placing the processor into a wait state. $\overline{BUSY}$ remains LOW for the duration of the conversion and returns HIGH at which time the processor finishes the memory READ and acquires the newly-converted data.

The major advantage of this interface is that it allows the processor to start conversion, wait, and then read data with a single read instruction. The fast conversion times of these converters ensures that the processor is not held in a wait condition too long.

Faster versions of many processors test the condition of the READY input soon after the start of an instruction cycle. For the READY input to force the processor into a WAIT state, the converter's $\overline{BUSY}$ must go LOW early in the bus cycle. If using the 8085-A2, the processor's S0 status signal gives the earliest indication that a READ operation is to occur. The connection diagram for the 8085-A2 Slow-Memory interface is shown below.

AD7575 TO 8085A-2 SLOW MEMORY INTERFACE



ROM INTERFACE

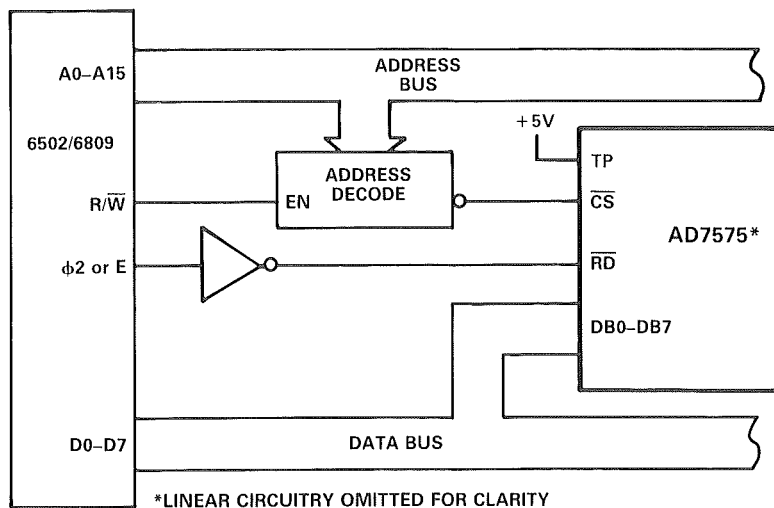
The alternative interface option avoids placing the processor into a WAIT state. A conversion is started with the first READ instruction and the second READ instruction accesses the data and starts a new conversion.

Conversion is initiated by executing a memory READ to the converter placing $\overline{CS}$ and $\overline{RD}$ LOW. Data from the previous conversion becomes available but may be ignored. Busy goes LOW indicating conversion in progress and returns HIGH after completion. The $\overline{BUSY}$ signal may be used to generate an interrupt or polled by the processor. For the converter to operate correctly in the ROM mode $\overline{CS}$ and $\overline{RD}$ should not go LOW before $\overline{BUSY}$ returns HIGH.

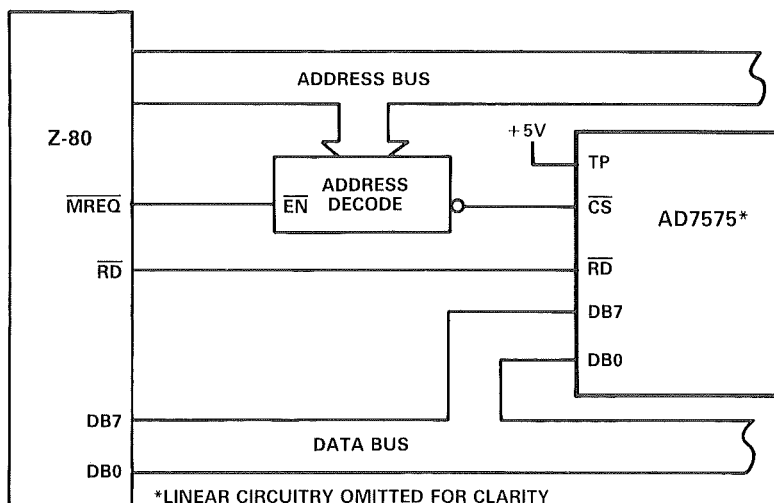
Normally the second READ starts another conversion. If $\overline{CS}$ and $\overline{RD}$ are brought LOW within one external clock period of $\overline{BUSY}$ going HIGH then a second conversion does not occur.

The disadvantage with this interface option is that the age of the data is unknown. That is, the data read is the result of a conversion which was initiated by the previous read cycle.

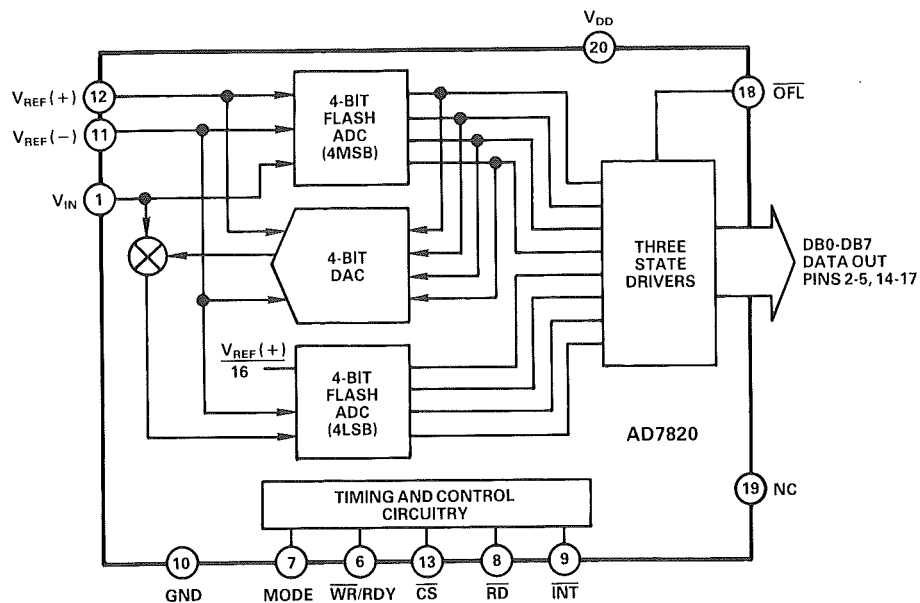
AD7575 TO 6502/6809 ROM INTERFACE



AD7575 TO Z-80 ROM INTERFACE



AD7820 FUNCTIONAL BLOCK DIAGRAM



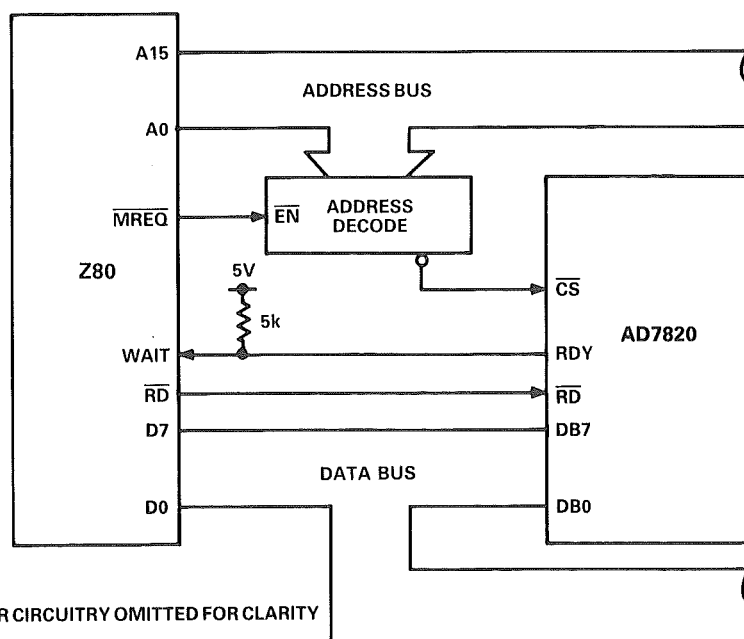
The AD7820 interface techniques are very similar to those shown for the AD7575/76. Here again, two interface modes are available depending upon the status of the **MODE** pin. With this pin **LOW** the AD7820 is in the **RD** mode whereas a **HIGH** places the converter in the **WR-RD** mode.

RD MODE

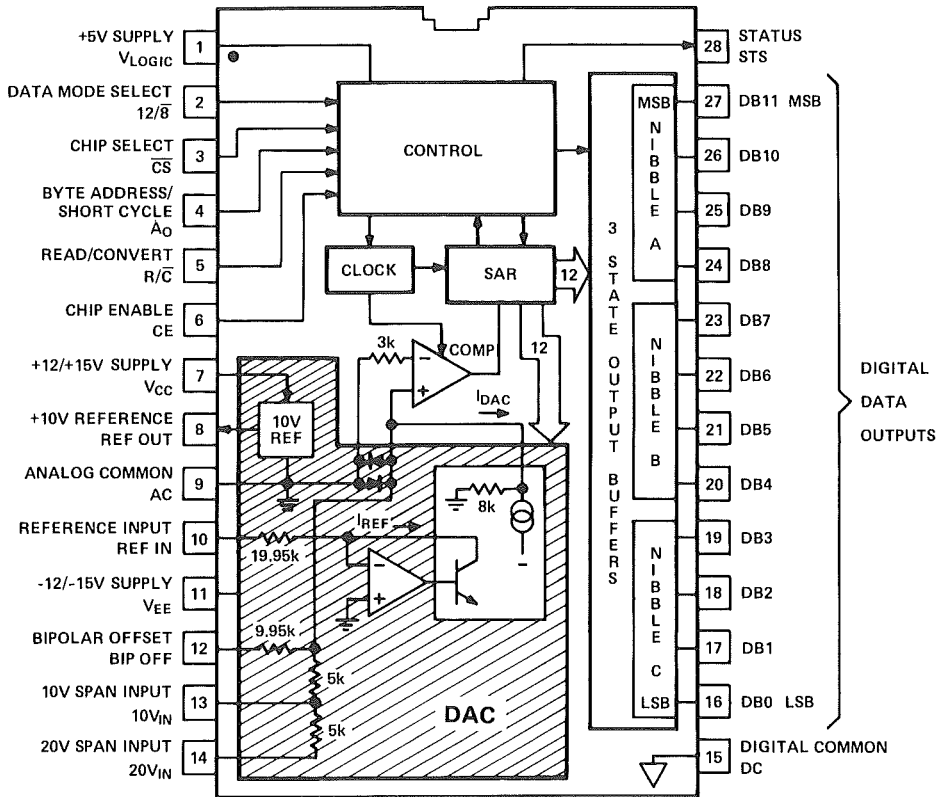
In the RD mode conversion is initiated by pulling $\overline{\text{RD}}$ LOW and holding it there 1.6 microseconds until output data appears. Pin 6 of the converter, RDY, provides a status output and can be used to put the processor into a WAIT state. RDY is an open drain output which goes LOW after the falling edge of $\overline{\text{CS}}$ and returns to the high impedance state at the end of conversion.

A typical interface to a Z80 is shown below. A LD B, (xxxx) statement will start conversion. At the beginning of the instruction cycle when the converter's address is decoded, RDY goes LOW to force the processor into a WAIT state. Upon completion of conversion RDY returns HIGH and the conversion result is stored in the processor's B register.

AD7820-Z80 INTERFACE



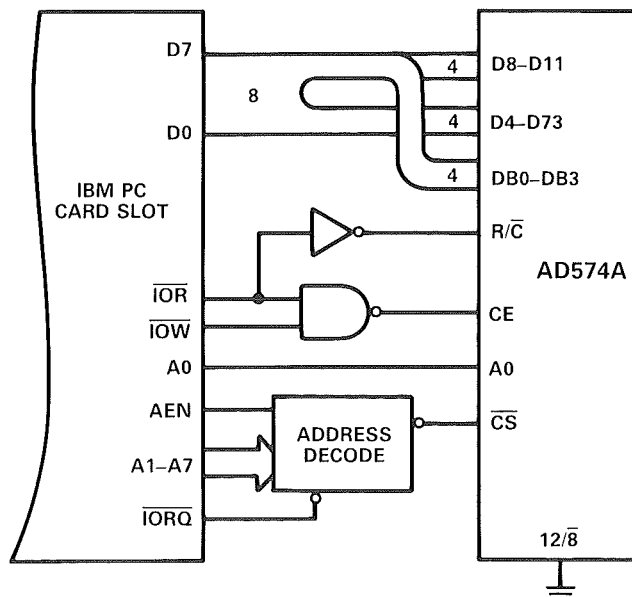
AD574A BLOCK DIAGRAM AND PIN CONFIGURATION



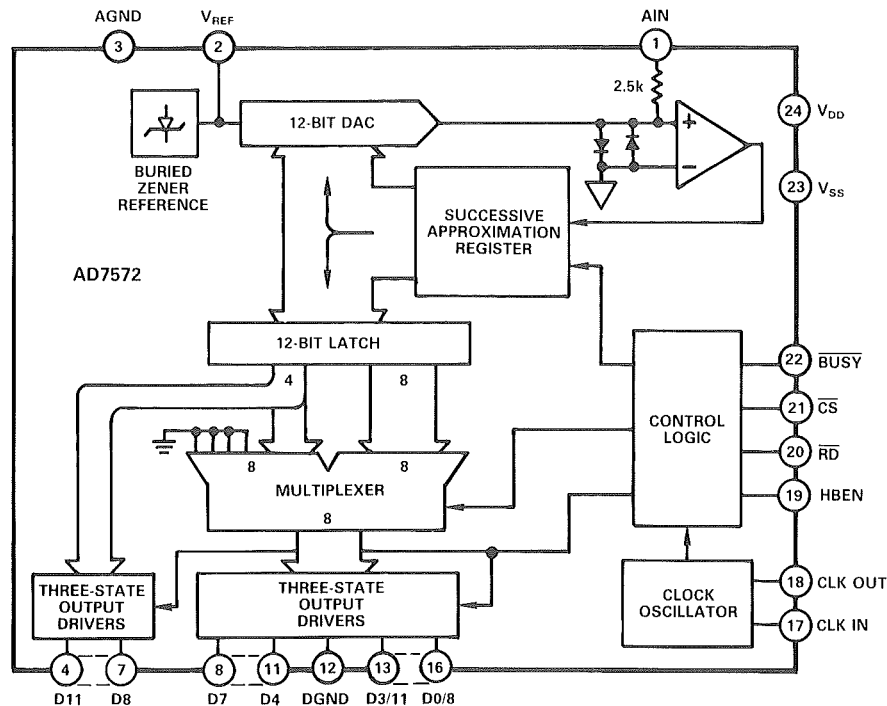
The AD574A has become the industry standard for 12-bit A/D converters. Developed by Analog Devices in the late 1970's this device has probably interfaced to more processors than any other converter in its class.

The AD574A appears below interfaced to a 4MHz. 8088 processor of an IBM PC. Since the converter resides in I/O space its address is decoded from only the lower 10 address lines and must be gated with AEN to mask out internal DMA cycles which use the same I/O space. This active LOW is applied to $\overline{CS}$. $\overline{IOR}$ and $\overline{IOW}$ are used to initiate the conversion and read data. They are gated together to drive the $\overline{CE}$. A_0 selects the 2 adjacent memory locations which store the 8 MSBs and 4 LSBs of the left-justified data.

IBM PC – AD574A INTERFACE



AD7572 FUNCTIONAL BLOCK DIAGRAM



Major breakthroughs in CMOS technology paved the way to the development of the AD7572 A/D converter. Using a mixed bipolar-CMOS process, improvements in conversion speed as well as faster bus timing were all made possible. In addition, this converter became the first CMOS A/D to incorporate an internal buried zener reference.

Conversion start and data read operations are controlled by three TTL compatible inputs: $\overline{CS}$, $\overline{RD}$, and HBEN. These three signals are internally gated such that a LOW on all inputs will begin a conversion. Once a conversion is started it cannot be restarted until the present conversion is finished. During conversion a LOW $\overline{BUSY}$ output indicates a conversion in progress.

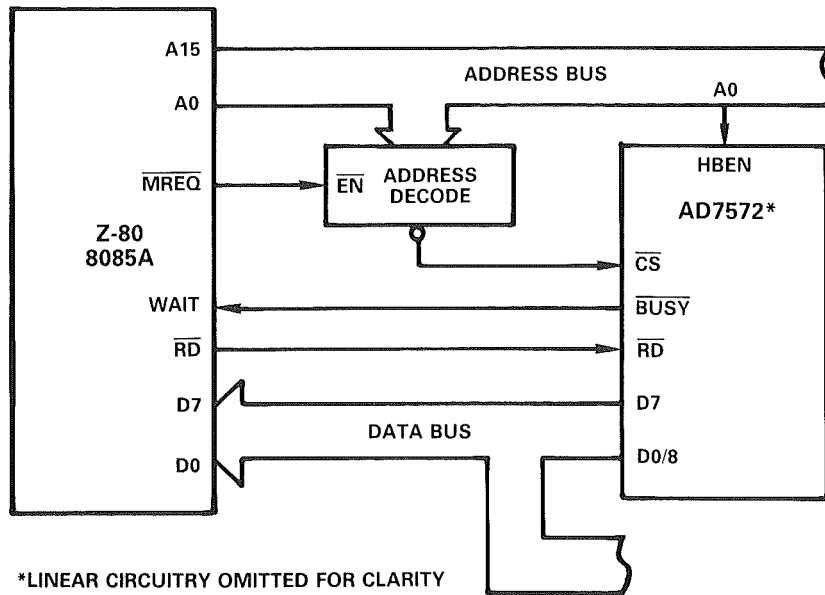
There are two modes of operation available for interfacing the device to a processor. They are the Slow Memory Mode and the ROM mode which were previously described in the section on the AD7575/76. The only difference however is the AD7572's data format which can be 12 bit parallel or two bytes of right-justified data.

The AD7572 is easily interfaced to a Z80 or 8085A. As shown, the converter is operating in the Slow Memory Mode and a two byte read is required. An 8-bit latch such as an 8282 is used to demultiplex the address/data bus with the decoded address providing the AD7572 $\overline{CS}$. An even address (A0 LOW) will start a conversion and read the low data byte. An odd address will read the corresponding high data byte. All this is accomplished with a single 16-bit LOAD instruction:

LDHL (B000) for the 8085A
LDHL (B000) for the Z80

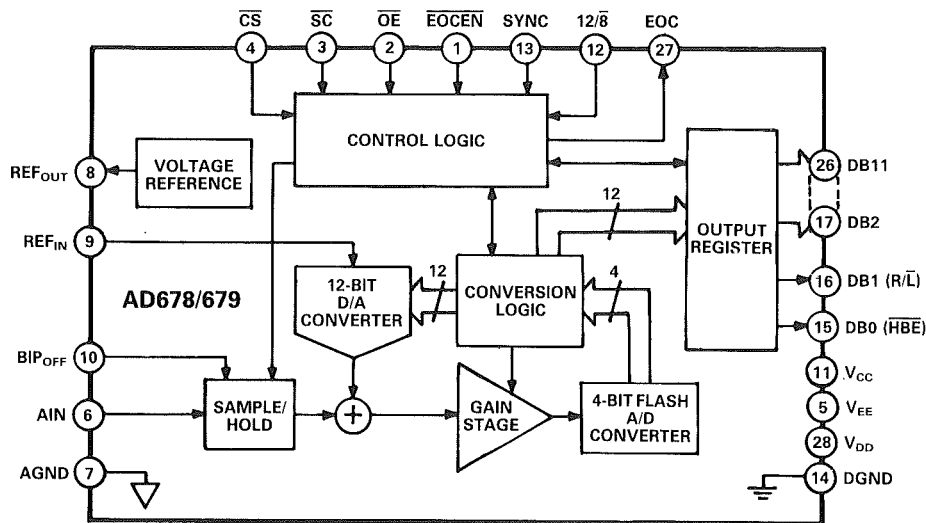
This instruction will load the converter's data (address B000) into the HL register pair. During the first read operation, $\overline{BUSY}$ forces the processor to wait for completion of conversion. The high data byte is then read with a second read operation without the addition of wait states.

AD7572 – 8085A/Z80 INTERFACE



AD678/679 Monolithic 12/14-Bit Sampling A/D Converter

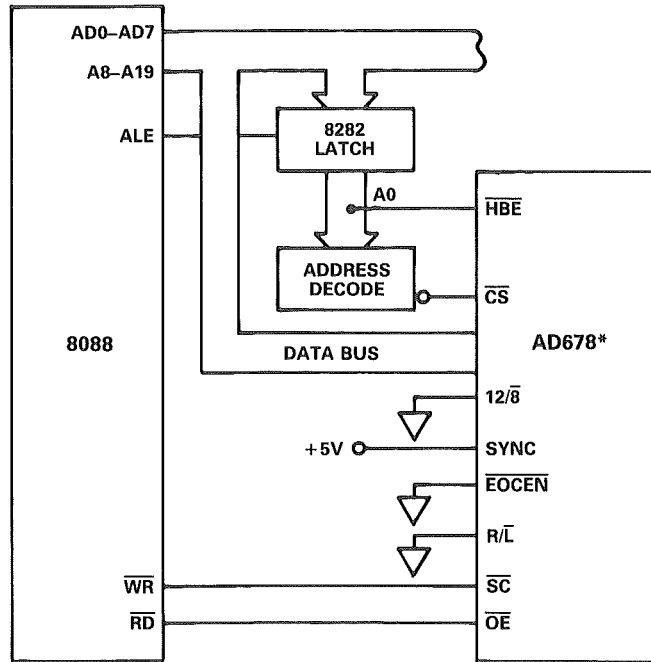
BLOCK DIAGRAM OF AD678/679 SAMPLING 12/14-BIT A/D CONVERTER



The AD678 12-bit A/D converter and the AD679 14-bit converter are the newest addition to Analog Devices' family of high resolution converters. Within a 28-pin package, the device incorporates a sample-and-hold amplifier, voltage reference, and three state output buffers. The high speed digital control logic and fast bus access time ensures operation with the fastest processors. Moreover, the converter may be configured to support 8- or 16-bit bus interfacing with selectable right or left justified data. In addition, the three-state output data buffers may be enabled during conversion allowing data from the previous conversion to be read into the processor via DMA.

Interfacing the AD678 to an 8088 microprocessor is easily accomplished. A 74LS374 address latch is used to demultiplex the address/data bus and the decoded address provides a LOW $\overline{CS}$ to the AD678. A LOW $\overline{WR}$ pulse initiates a conversion and 5 microseconds later conversion is complete. The processor now issues a $\overline{RD}$ to an even decoded address ($A0 = \text{LOW}$) enabling the 8 MSBs. Since the $R/\overline{L}$ control is hardwired LOW this data will be left-justified. A second $\overline{RD}$ to the next decoded address ($A0 = \text{HIGH}$) outputs the 4 LSBs with 4 trailing zeroes.

8088 – AD678 INTERFACE

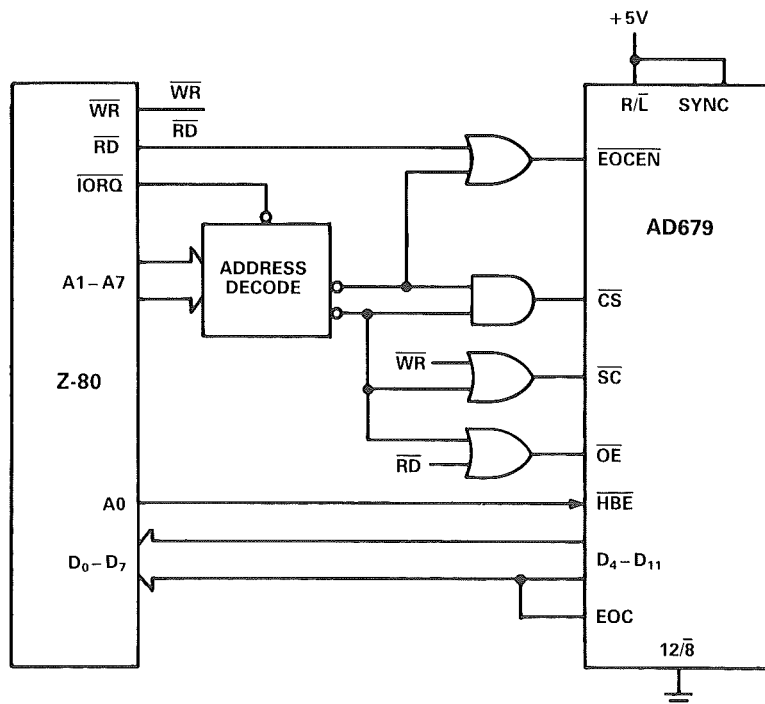


*LINEAR CIRCUITRY OMITTED FOR CLARITY

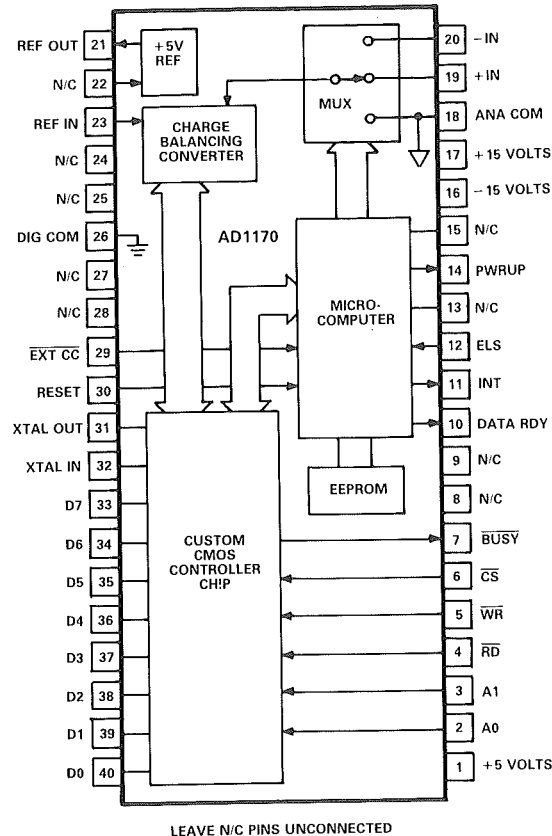
Using an I/O or memory-mapped configuration, the AD678 can be interfaced to the Z80 microprocessor. As shown, the converter occupies several port addresses to allow separate polling of the EOC status and reading of the data. The lower address bit, $A0$, is used to select the high and low-order bytes of the result. The $R/\overline{L}$ control is tied high resulting in right-justified output data.

The Z80 processor will automatically insert a single wait state during I/O operations. This feature allows the AD679 to be used with the Z80 having a clock speed up to 8MHz.

AD679 TO Z-80 INTERFACE



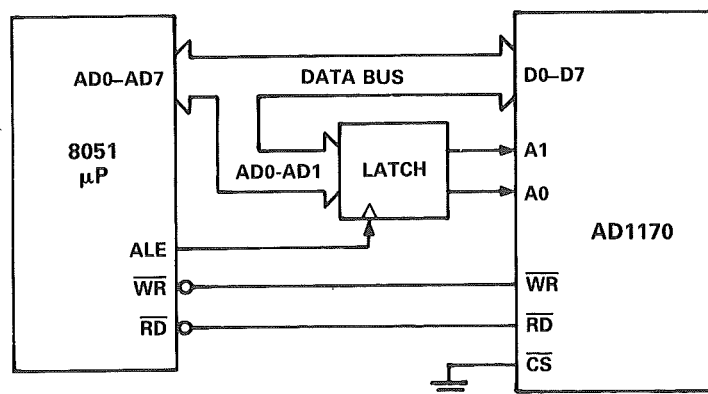
AD1170 FUNCTIONAL BLOCK DIAGRAM



Many high precision applications require more than 16 bits of resolution that is available in monolithic technology. In addition, many of today's 16-bit converters can only achieve 14 bits of accuracy and most often require external trimming for gain and offset. In some instances where external trimming is not feasible, then a software routine to calculate gain and offset errors might be used. Such a routine is illustrated in the Data Acquisition Subsystem section of this book. But these routines require additional memory and software overhead not to mention the time spent debugging the program. The AD1170 however provides an easy solution to high resolution A/D conversion. Using a charge balancing conversion technique and self calibration of gain and offset, the AD1170 offers 16 bits of accuracy in a $1.25 \times 2.5 \times 0.55$ inch package.

As shown, the AD1170 is interfaced to an 8051 microcomputer. An external latch is used to latch the two lowest order address bits from the multiplexed data bus. The indirect external data command MOVX is the only instruction required to access the converter.

8051 – AD1170 INTERFACE



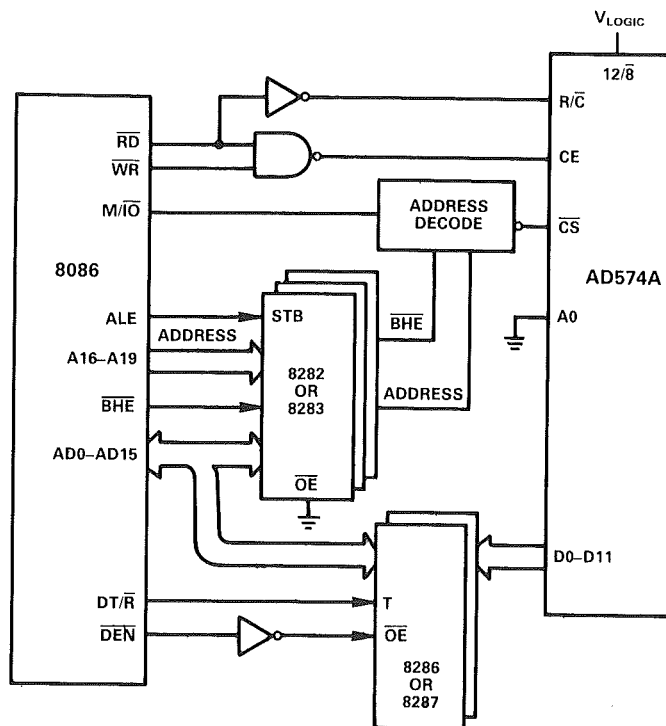
16-BIT BUS INTERFACING

AD574A 12-Bit A/D Converter

Interfacing an AD574A to a 16-bit data bus is quite similar to the 8-bit interfacing techniques previously shown. When using a 16-bit bus the data mode select pin (12/8) should be connected to V_{LOGIC} and A0 is hardwired LOW. The data format is selected by the user either left or right justified by wiring the 12 data outputs to the 12 MSBs or 12 LSBs of the bus.

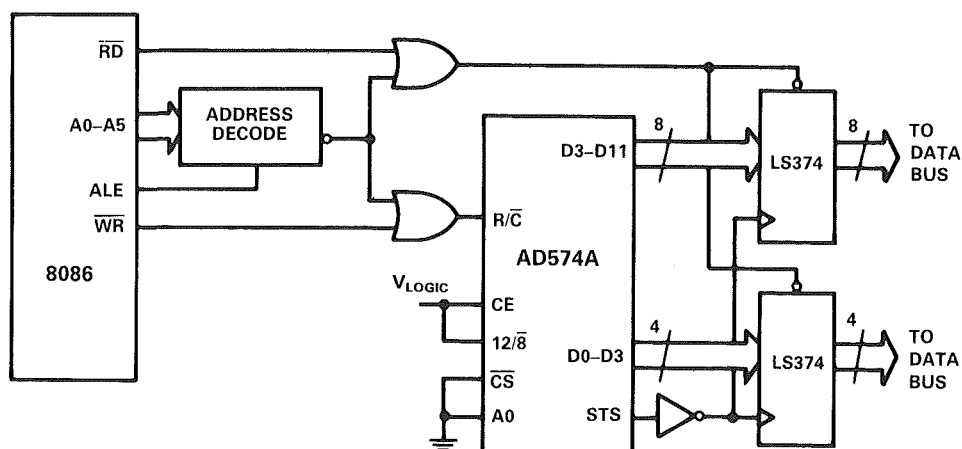
An 8086 interface is shown below. An 8282 latch is connected to the multiplexed address/data bus to demultiplex the address bus and an 8286 provides data bus buffering. This technique is strongly recommended with any multiplexed bus to prevent possible bus contention and to minimize digital feedthrough noise. Note that no wait states are needed for 8086 clock frequencies up to 6MHz.

8086 – AD574A WITH BUFFERED BUS INTERFACE



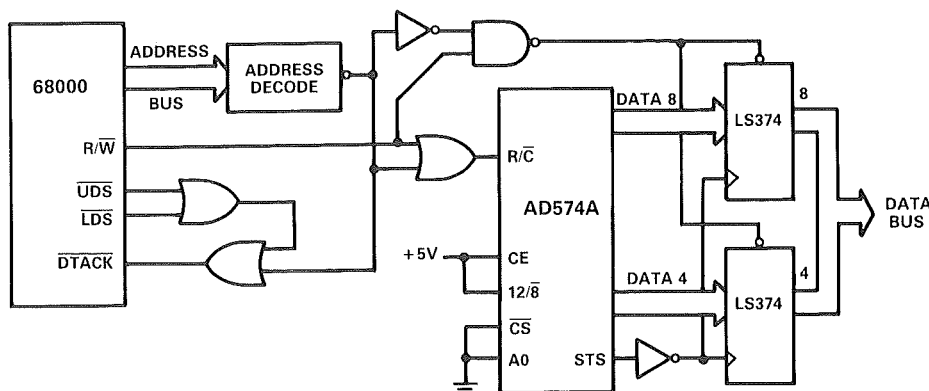
Stand-alone operation of the AD574A often allows the user to interface to high speed processors. In this mode, a single LOW going pulse to the $\text{R}/\overline{\text{C}}$ control line is all that is necessary to start conversion. The converter's STS line will go HIGH and remain high for the duration of conversion. Prior to the falling edge of STS, data becomes valid and can be conveniently latched into a three-state latch such as a 74ALS374. The data is now read by enabling the latch's output enable. This method eliminates problems with bus access and bus float delay that are often encountered with higher speed processors.

8086 – STAND-ALONE CONFIGURATION



Stand-Alone operation is also possible with the 68000 microprocessor. The 68000 $R/\overline{W}$ signal combined with a low address decode initiates conversion. The $\overline{UDS}$ and $\overline{LDS}$ signals with the decoded address generates the $\overline{DTACK}$ input to the processor. A read cycle then reads data from the three-state latches.

68000 – AD574A INTERFACE



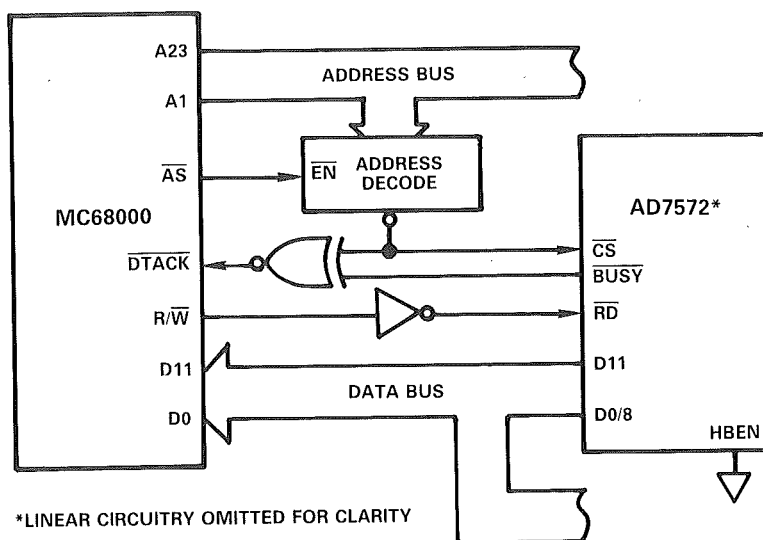
AD7572 5 μ s A/D Converter

A typical interface for the 68000 is shown below. The AD7572 is operating in the Slow Memory Mode. Assuming the AD7572 is located at address B000, then the following single 16-bit MOVE instruction both starts a conversion and reads the conversion results.

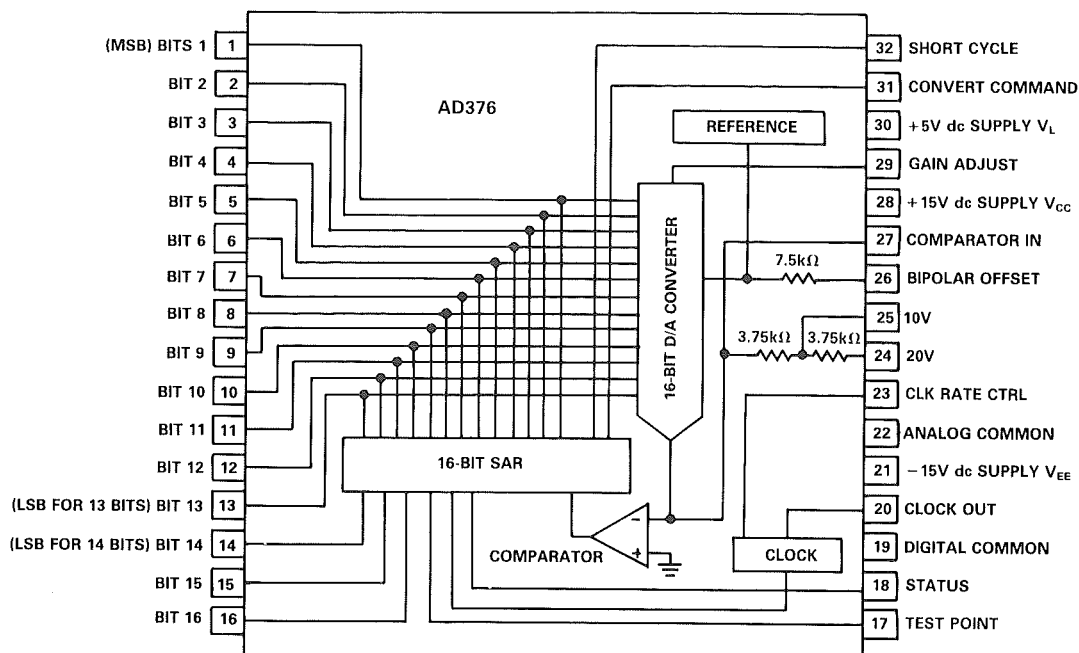
Move.W \$B000,D0

At the beginning of the instruction cycle when the A/D address is selected, $\overline{BUSY}$ and $\overline{CS}$ assert $\overline{DTACK}$ forcing the 68000 into a WAIT state. At the end of conversion $\overline{BUSY}$ returns HIGH and the conversion result is placed in the D0 register of the processor.

AD7572 – MC68000 INTERFACE



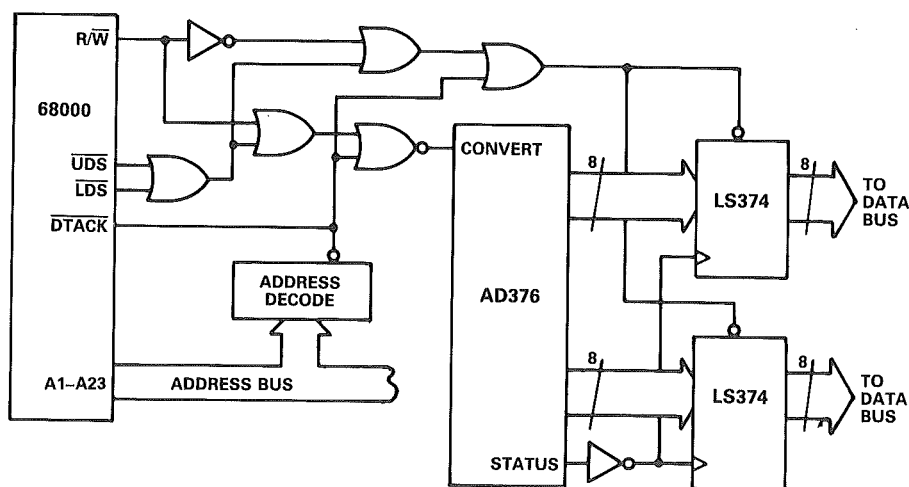
AD376 FUNCTIONAL BLOCK DIAGRAM



The simple control logic of the AD376 makes it an ideal candidate for interfacing to high speed processors. A 50ns high convert start pulse drives the STATUS line HIGH, removes the gated clock inhibit, and starts the 16-bit successive approximation process. One clock cycle after the falling edge of CONVERT START the MSB bit decision is made and data becomes valid on the rising edge of the internal clock. This negative true data (Logic "1" = 0.4V max and Logic "0" = 2.4V min) remains latched during the entire conversion process. The subsequent bit decisions are made and STATUS goes LOW to indicate end of conversion.

Interfacing the AD376 to a 68000 microprocessor requires minimal hardware. A LOW decoded address gated with the processor's $\overline{WR}$ generates a pulse. Two 74ALS244 octal bus buffers isolate the AD376 from the data bus and offer fast bus access and float delay necessary for operation with high speed processors.

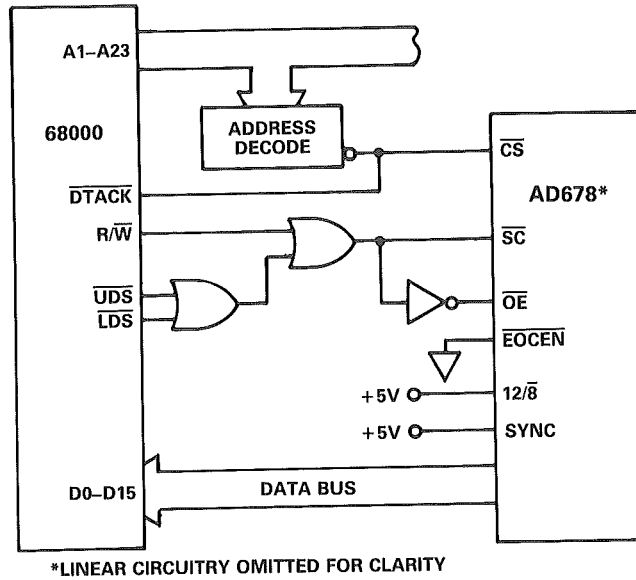
68000 – AD376 INTERFACE



AD678 12-Bit Sampling A/D Converter

The high speed bus interface of the AD678 12-bit sampling A/D converter supports a direct bus interface to a 12.5MHz 68000 system. A LOW address decode asserts the AD678 $\overline{\text{CS}}$ and $\overline{\text{DTACK}}$ allowing the processor to run at full speed without the need for wait states. To start conversion, a LOW going R/W signal OR'ed with the OR gate output of $\overline{\text{UDS}}$ and $\overline{\text{LDS}}$ produce a LOW $\overline{\text{SC}}$ pulse. 5 μs later, conversion is complete and the processor can now read data.

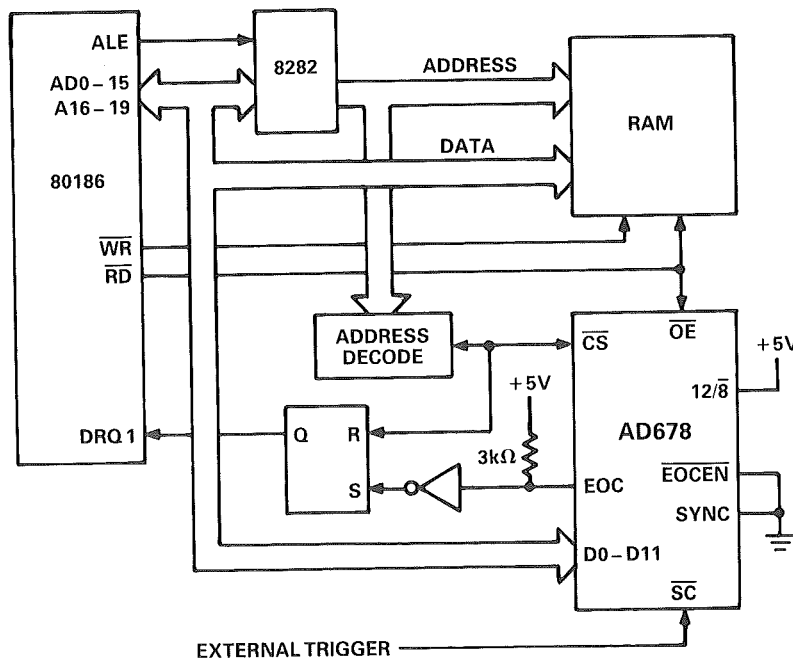
68000 – AD678 INTERFACE



A 6MHz or 8MHz 80186 processor is shown interfaced to the AD678 converter. This configuration allows the processor's internal DMA controller to transfer the AD678 output into a RAM-based FIFO buffer of any length with no processor intervention.

The AD678 is configured in the Asynchronous mode allowing conversions to be initiated by an external trigger source independent of the processor clock. After each conversion, the AD678 EOC signal generates a DMA request to Channel 1 (DRQ1). The subsequent DMA read operation resets the interrupt latch. The system designer must assign a sufficient priority to the DMA channel to ensure that the DMA request will be serviced before the completion of the next conversion.

AD678 TO 80186 DMA INTERFACE (ASYNCHRONOUS)

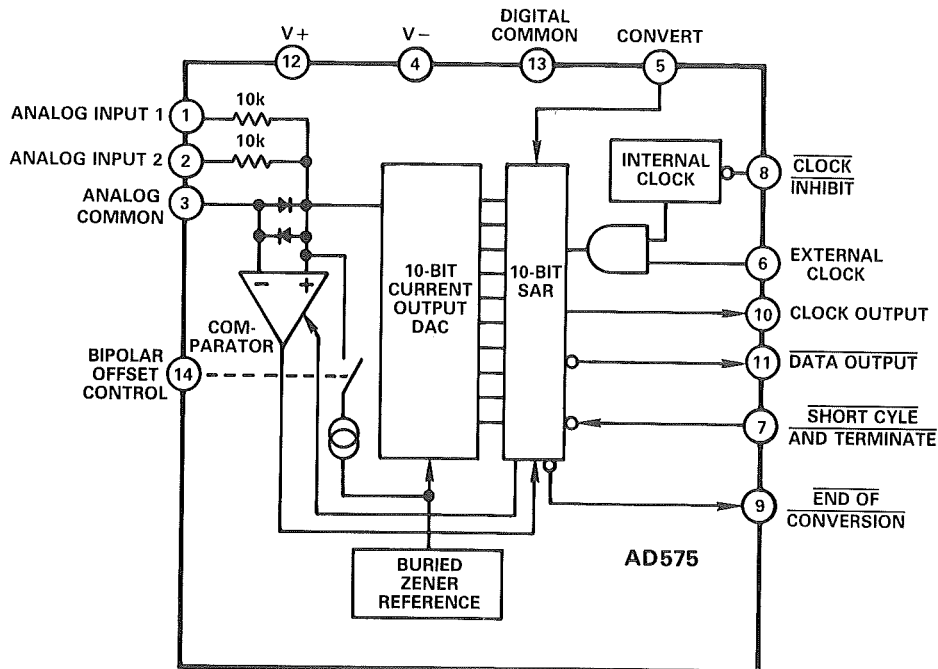


INTERFACING CONVERTERS WITH SERIAL OUTPUT

Some microprocessors such as the 8085 and the TMS32020 can accommodate serial I/O. Generally speaking most A/D converters have parallel data formats. In order to connect these converters to a serial input port, a shift register would be required to convert to serial format. The increasing popularity however of digital signal processors in audio applications has focused attention to developing more converters with serial outputs. Let's look at some of these developments.

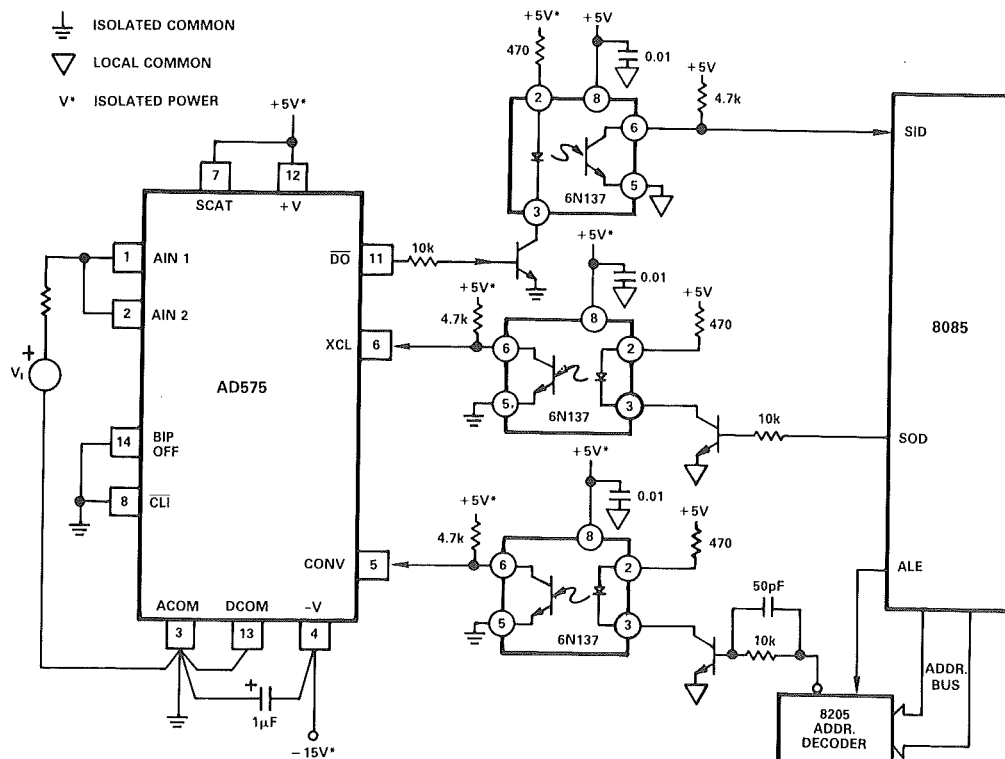
AD575 10-Bit A/D Converter with Serial Output

AD575 FUNCTIONAL BLOCK DIAGRAM



The 8085 has both serial output (SOD) and serial input (SID) capability. Using opto-coupling to establish galvanic isolation between processor and converter a simple three wire interface can be configured.

AD575 TO 8085 ISOLATED INTERFACE



The software routine listed below will read a complete 10-bit data word from the AD575 in 180 μ s using a 3MHz 8085. The software generates the clock for the AD575 in order to synchronize the data output with the 8085 serial read operation.

The DATA procedure loads appropriate constants into the 8085 registers and initiates the conversion. The CONV routine assumes the AD575 clock was in the high state when the CONVERT pulse was generated. A low clock pulse is generated and the data bit is read into the MSB of the accumulator. The data bit is then shifted into the LSB of the temporary register (L), the clock is set high, and the procedure is repeated.

After the loop has executed three times, a logical AND is performed to set the first bit to zero. The result is then placed into the high byte (H) register. The loop counter is reset, and the CONV routine is executed 8 more times. Upon completion of the routine, 10 bits of right-justified data will reside in the HL register pair.

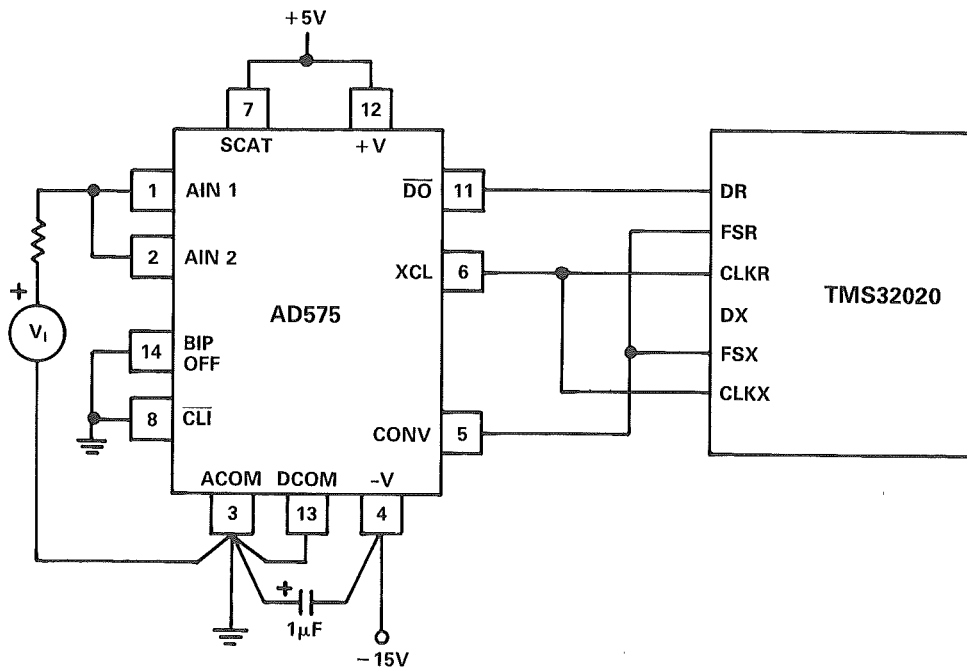
Note that the opto-isolators invert the clock and data lines. If these are not used the constants in the D and E registers must be swapped, a CMA instruction should be inserted after the RIM instruction, and an inverter should be connected between the address decoder and the CONVERT pin. In addition following power up and reset, the results of the first pass through the routine are invalid.

SAMPLE ASSEMBLY CODE FOR AD575 to 8085 ISOLATED INTERFACE

LABEL	MNEMONIC	OPERAND	COMMENT
DATA	MVI	B,03	Set inner loop counter to 3
	MVI	C,02	Set outer loop counter to 2
	MVI	D,CO	Setup register D for clock low
	MVI	E,40	Setup register E for clock high
	MVI	H,10	AD575 address location
	MVI	L,00	Clear temp register
	MOV	M,B	Generate CONVERT pulse
CONV	MOV	A,D	Setup ACC for clock low
	SIM		Output clock low
	RIM		Read AD575 data bit into ACC
	RAL		Shift data bit into Carry
	MOV	A,L	Move temp to ACC
	RAL		Shift data bit from Carry to ACC
	MOV	L,A	Replace temp
	MOV	A,E	Setup ACC for clock high
	SIM		Output clock high
	DCR	B	Decrement inner loop counter
	JNZ	CONV	Repeat CONV until done
	DCR	C	Decrement outer loop counter
	JZ	DONE	Skip to DONE on 2nd pass
	MOV	A,L	Move temp to ACC
	ANI	03	Mask undefined bit
	MOV	H,A	Store temp in H register
	MVI	B,0B	Set inner loop counter to 8
	JMP	CONV	Repeat CONV for 8 LSBs
DONE	RET		10 bits of right-justified data now reside in HL; return

Shown below is a simple serial interface using the AD575 and the TMS32020 Digital Signal Processor. The converter is configured for external clock operation, the clock being provided by the processor's CLKR and CLKX signals. Conversion is initiated on the rising edge of FSR and FSX with the first data bit (the MSB) valid on the falling edge of FSR and FSX. Note that the data format is complementary.

TMS32020 - AD575 SERIAL INTERFACE



ANALOG-TO-DIGITAL CONVERSION USING VOLTAGE-TO-FREQUENCY CONVERTERS

A voltage-to-frequency converter (VFC) is a device which accepts at its input an analog voltage or current signal and provides at its output a train of pulses or square waves at a frequency which is proportional to the input value. A VFC can thus be used as a building block in an A/D conversion system by using the VFC to clock a counter for a certain period of time and reading the output digital word. This digital word will be proportional to the analog input.

There are a number of advantages to using a voltage-to-frequency converter in an A/D conversion scheme. First, unlike converters based on binary-weighted networks, monotonicity is inherent under all supply and temperature conditions. Second, the fact that the signal is converted to an easily-transmitted serial bit stream allows the analog circuitry (the VFC and analog signal conditioning) to be located close to the signal source and the digital circuitry to be located elsewhere. This is especially advantageous when a large number of channels are required; the remote VFCs can be used to provide "converter-per-channel" data acquisition. Finally, since the digital number is accumulated over a large number of cycles, integration of and reduction of unwanted signals is inherent.

The time required to convert an analog signal into a digital word is related to the maximum full-scale frequency of the VFC and the required resolution of the measurement. For example, the Analog Devices AD650 VFC has a full-scale frequency of 1MHz. If this device is used in an application where a resolution of 16 bits, or 1 part in 65,536 is desired, then the time required to convert the analog signal into a 16-bit digital word will be 65.536ms. Resolution of 18 bits, or 1 part in 262,144 will require a count time slightly greater than 0.262 seconds. In general, the required count time for an A/D conversion using a VFC is:

$$T_{\text{COUNT}} = N / \text{FS}_{\text{OUT}}$$

where N is the number of codes for a given resolution and FS_{OUT} is the VFC full-scale output frequency.

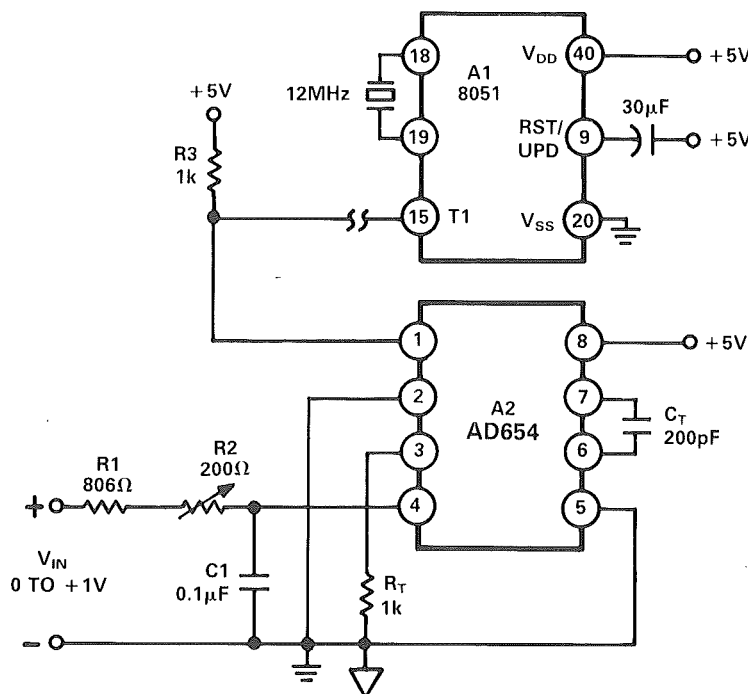
Although VFC-based A/D converters are slower than successive-approximation and flash converters, they are comparable in speed to integrating A/D's. VFC-based A/D converters are thus well suited for low frequency applications such as temperature and strain-gage measurements. The resolution that the VFC can provide in these types of applications offsets the relatively long count time required to acquire the digital word corresponding to the analog input value.

PULSE COUNTING

One way to perform an analog-to-digital conversion using a VFC is to have a single-chip microcomputer count the number of pulses that occur in a fixed time period. The total number of pulses counted during this period is then proportional to the input voltage of the VFC. For example, if a 1V full-scale input produces a 100kHz signal from the VFC and the count period is 100ms, then the total full-scale count will be 10,000. Scaling from this maximum is then used to determine the input voltage, i.e., a count of 5000 corresponds to an input voltage of 0.5V.

The figure below shows the AD654 VFC output connected to the counter input, T1, of the Intel 8051 microcomputer. The AD654 is a low-cost, single-supply, monolithic VFC with a full-scale frequency of up to 500kHz. The 8051 is a member of the Intel MCS-51 family of 8-bit microcomputers whose family members differ mainly in their internal memory capabilities. In the following text, the term "8051" is used to generically refer to all members of the MCS-51 family.

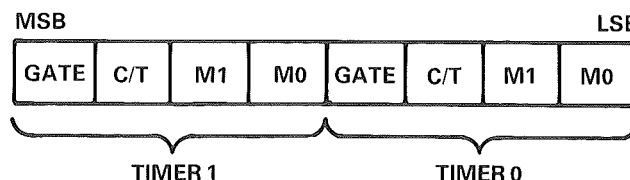
AD654 PULSE COUNTER



The analog input of the AD654 is a 0 to +1V signal. The timing resistor and capacitor, R_t and C_t , are selected such that this 0 to +1V signal seen at Pin 4 results in a 0 to 500kHz output frequency. The pull-up resistor, R_3 , ensures that the AD654 output meets the logic levels required at T1 of the 8051.

The 8051 has two 16-bit timer/event counters on-chip. These counters, Timer 0 and Timer 1, can be programmed independently to operate as 16-bit time-interval or event counters. The use of Timer 0 and Timer 1 is determined by two 8-bit registers, TMOD (timer mode) and TCON (timer control). The TMOD register is shown below:

8051 TMOD REGISTER



M1 and M0 are used to select the mode of each timer. Mode 01 configures the timer as a 16-bit time-interval or event counter. C/T is the timer or counter selector and is cleared for timer operation. In this application, Timer 0 was configured as the timer to provide the fixed time interval and Timer 1 was configured as the counter to count the pulses. Hereafter, the two timers will be referred to as Timer 0 and Counter 1. When running, Timer 0 increments at a rate equal to the external clock divided by twelve. Using a 12MHz crystal, this corresponds to once every microsecond. GATE is the gating control. When this bit is clear, timer/counter X is enabled whenever the TRx control bit found in the TCON register is set. The TRx bit is controlled via software. When the GATE bit is set, timer/counter X is enabled whenever the TRx bit is set and the signal level appearing at the INTx pin is high. Thus, when a GATE bit is clear that timer is controlled by software only, and when it is set that timer is controlled by a combination of software and hardware. In this application, the GATE bits are clear; however, in the next application they will be set.

Table 1 lists the software routine PLSECNT, which counts the number of falling edges that appear at T1 (the Counter 1 input) in a 50ms window. After Counter 1 is cleared, the value 15539 is loaded into Timer 0. Since Timer 0 is a 16-bit timer, the maximum possible count is 65535. With the Timer 0 interrupt enabled, a count of 65536 will cause a jump to the starting address (0BH) of the Timer 0 interrupt service program. With Timer 0 starting at 15539 and incrementing once every microsecond (based on a 12MHz clock), there will be 49997 counts or 49.997 ms before jumping to the service program. The 3 microsecond difference from 50ms is made up in the speed of the interrupt response. The interrupt response latency ranges from 3 to 7 microseconds when using a 12MHz. crystal. During this 50ms count period, control resides with the main program. Thus, the 8051 is not tied up while Counter 1 is counting for 50ms. After the interrupt service is reached, Counter 1 and Timer 0 are stopped and the contents of Counter 1 are moved into RAM, where it may be accessed at the user's convenience. Control is then returned to the main program for which the subroutine was written. With a maximum frequency of 500kHz and a count window of 50ms, the maximum value of Counter 1 will be 25,000. This provides resolution greater than 14 bits. Appropriate scaling from this 1V full-scale reference point may then be performed in software.

8051 PULSE COUNT ROUTINE

```

                                ORG    00H
                                AJMP   MAIN

PLSECNT  ORG    60H              ;PULSE COUNT SUBROUTINE
                                MOV     TMOD, #51H  ;Put Time 0 and Count 1 in Mode 01
                                MOV     TL1, #00H    ;Initialize Counter 1 Register
                                MOV     TH1, #00H
                                MOV     TL0, #0B3H   ;Load Time 0 With 15536 + 3.Will
                                MOV     TH0, #3CH     ;Ovflw After 50ms + 3μs Delay
                                SETB    PT0          ;Prioritize Time 0 Interrupt
                                SETB    ET0          ;Enable Time 0 Interrupt
                                SETB    EA           ;Enable Global Interrupt
                                SETB    TR0          ;Start Timer
                                SETB    TR1          ;Start Counter
                                RET                ;Return to Main Program

                                ORG    0BH            ;TIME 0 INTERRUPT SUBROUTINE
                                CLR     TR1          ;Stop Counter
                                CLR     TR0          ;Stop Timer
                                AJMP   COUNT

COUNT   ORG    40H
                                MOV     50H, TL1     ;Move Counter Contents Into RAM
                                MOV     51H, TH1
                                RETI                ;Return from Interrupt

MAIN     ORG    100H
                                -             ;Main Program for Which PLSECNT
                                                ;Subroutine Was Written

```

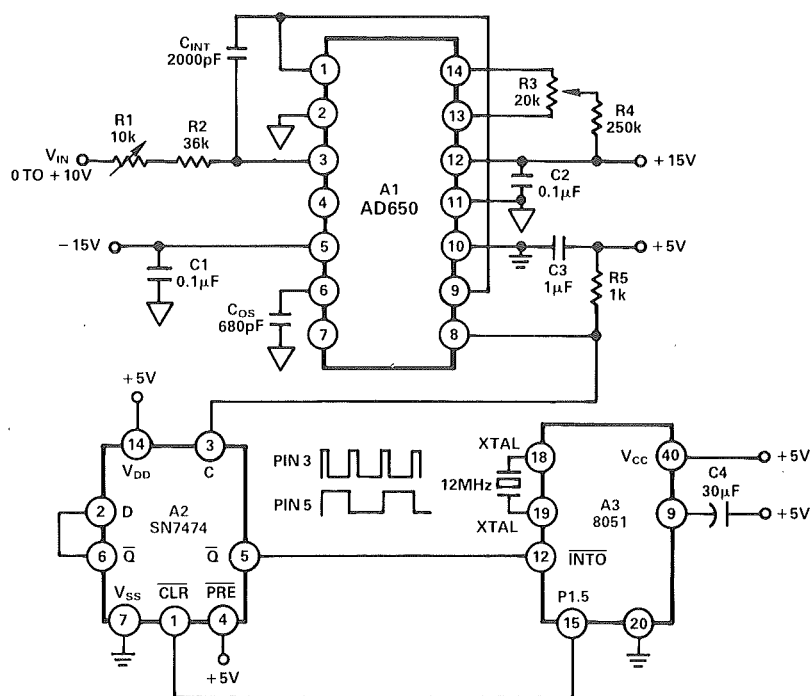
PERIOD TIMING

Another method of performing analog-to-digital conversion using a VFC and a microcomputer is to have the computer time the period of the VFC output frequency. For example, an output frequency of 20kHz has a period of 40 microseconds. If a timer that is incremented once per microsecond is gated to this signal, a total count of 40 will result. An output frequency of 250Hz has a period of 4ms. The same timer gated to this periodic signal will then produce a total count of 4000.

One of the advantages of period timing over pulse counting is that the count window is dependent upon the output frequency of the VFC; in many cases the count window will be shorter for period timing than for pulse counting. This will be especially important in systems where a number of channels are being converted. Recall that in the Pulse Counter application, the count window was 50ms whether the output frequency was 50kHz or 50Hz. With Period Timing, the count window is the inverse of the output frequency. Thus, a 50kHz signal will have a count window of 20 microseconds while a 50Hz signal will have a count window of 20ms. In fact, it's not until the output frequency reaches 20Hz that the Period Timing count window is equal to the 50ms Pulse Counter count window.

The figure below illustrates the circuitry necessary to perform an A/D conversion via period timing using the AD650 VFC. The AD650 has a maximum full-scale frequency of 1MHz with a nonlinearity error of 0.1%. The AD650 is configured such that a 0 to +10V input will result in a 0 to 50kHz output. Because the AD650 output is made up of pulses, the 7474 flip-flop is used to convert these pulses into a square wave. The 7474 Pin 3 and Pin 5 waveforms sketched in the figure show that the width of either the high-level or low-level output appearing at Pin 5 is the same as one period of the AD650 output frequency. Note also that when Pin 1 on the 7474 is held low, Pin 5 is also held low.

AD650 PERIOD TIMING



Recall that the INTO pin on the 8051 is the Timer 0 gate pin. When the GATE bit is set in the TMOD register, Timer 0 will run only when INTO is high and the TR0 bit in the TCON register has been set via software. Thus, connecting the Q output of the 7474 to the INTO pin on the 8051 will ensure the timer runs for one period of the AD650 frequency.

One problem that could arise is the TRO bit in software being set in the middle of a high-level edge at the INTO pin. In this case, Timer 0 would run for a fraction of a period rather than one full period. This is countered by tying the 8051 Port 1 Bit 5 (P1.5) pin to the 7474 CLR pin. When CLR is low and PRE is high, Q is low. When CLR and PRE are both high, Q changes state on every positive edge appearing at the clock pin. Setting P1.5 low, setting TRO (in software) and then setting P1.5 high will thus ensure that Timer 0 runs for one full period.

The software subroutine PCNT increments Timer 0 once per microsecond for one AD650 output frequency period. Note that there are two interrupt service programs, one for INTO and one for Timer 0. The INTO service program is accessed after a negative edge appears at the INTO pin (Pin 12) signifying the end of one period. The timer is then stopped and its contents are loaded into RAM to be later accessed.

PERIOD TIMING ROUTINE

```

                                ORG    00H
                                AJMP   MAIN
PCNT    ORG    90H                ;PERIOD COUNTER SUBROUTINE
                                MOV     TMOD, #05H    ;Put Time 0 in Mode 1, Enable  $\overline{\text{INT0}}$  Pin
                                CLR     P1.5          ;Initially Set INTO Pin Low
                                SETB    IT0           ;Specify Edge Triggered Interrupt
                                MOV     TLO, #00H      ;Initialize Timer
                                MOV     TH0, #00H
                                SETB    EX0           ;Enable  $\overline{\text{INT0}}$  Interrupt
                                SETB    ET0           ;Enable Timer 0 Interrupt
                                SETB    EA            ;Enable All Interrupts
                                SETB    TR0           ;Start Timer
                                SETB    P1.5          ;Enable Gate  $\overline{\text{INT0}}$  Pin
                                RET                  ;Return to Main Program

                                ORG    03H                ; $\overline{\text{INT0}}$  Subroutine Service Program
                                CLR     TR0           ;Stop Timer
                                CLR     EA            ;Disable Interrupts
                                AJMP    COUNT         ;Jump to Count

                                ORG    0BH                ;TIMER 0 SUBROUTINE
                                                SERVICE PROGRAM
                                CLR     TR0           ;Stop Timer
                                CLR     EA            ;Disable Interrupts
                                AJMP    OFLW          ;Jump to OFLW

OFLW    ORG    40H
                                MOV     60H, #FF       ;Load RAM With Overflow
                                MOV     61H, #FF       ;Value
                                CLR     P1.5          ;Set  $\overline{\text{INT0}}$  Pin Low
                                RETI                  ;Return From Subroutine

COUNT  ORG    50H
                                MOV     60H, TH0       ;Load RAM with Counter Contents
                                MOV     61H, TLO
                                CLR     P1.5          ;Set  $\overline{\text{INT0}}$  Pin Low
                                RETI                  ;Return From Subroutine

MAIN    ORG    100H
                                -          -          ;Main Program for Which
                                                ;Subroutine Was Written

```

The Timer 0 service program serves to limit the count window to approximately 65.5ms. It is accessed when the contents of Timer 0 reach 65536. This will occur when the AD650 input voltage is approximately 3.05mV or the output frequency is approximately 15.26Hz. This service program then loads the overflow value 65535 into RAM. After both interrupt subroutines, and after initialization of the PCNT subroutine, control returns to the main program for which the subroutine was written. Thus the 8051 is not tied up while the period timing is occurring.

One possible source of error in this application is jitter, which is the range of variation in the period of the output frequency. This variation in period would result in a variation in the number of pulses counted from one period to the next. The magnitude of this error can be greatly reduced in software by taking the average of a number of period counts and using this average value for calculations.

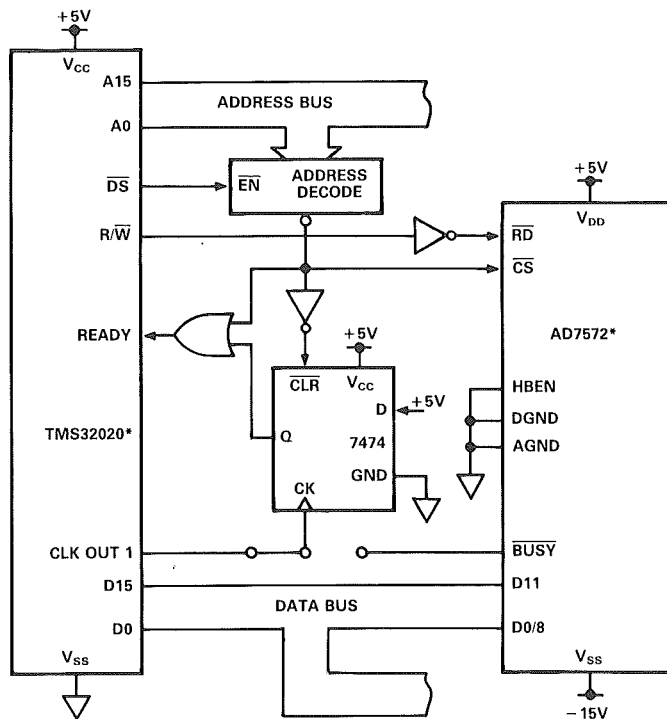
INTERFACING TO HIGH-SPEED DSP PROCESSORS

More and more designers have discovered the tremendous power of digital signal processing on "real world" phenomena. Today, DSP techniques are used in a variety of applications such as telecommunications, secure communications, digital audio, and automotive. Most of these applications involve A/D and D/A converters whereby a signal is digitized, digital filtered, and then reconstructed with the D/A. The maximum input frequency to the A/D and the sampling interval (normally twice the maximum input frequency) will determine the conversion speed of the A/D. But what about the digital interface??? Many of the popular DSP processors such as the TMS320 series from Texas Instruments and the ADSP-2100 from Analog Devices require an extremely fast interface. Older converters are often too slow getting on or off the bus or have minimum pulse widths of several hundred nanoseconds. In these situations, placing the processor in a wait state extends read and write pulse widths as well as increases the time allotment for bus access and float delay.

The fast conversion speed of the AD7572 makes it an ideal candidate when interfacing to a TMS32020. With bus access time of 40ns, the processor must be placed into a wait state when reading data from the AD7572. A 7474 D Flip-flop and an OR gate provide the necessary delay. The following I/O instruction both starts a conversion and reads the previous conversion result into a data memory location:

IN A,PA (PA = PORT ADDRESS)

AD7572/TMS32020 INTERFACE USING WAIT STATES



*ADDITIONAL PINS OMITTED FOR CLARITY

This interface is designed to extend the data read cycle for 1 clock cycle or 200ns for a 20MHz TMS32020. A few simple modifications will extend the A/D read cycle by the AD7572 conversion time of 5 μ s. Connect the BUSY output of the AD7572 to the clock input of the D Flip-flop. Once a conversion is started, the processor's READY input is held LOW during conversion thus forcing the processor into a wait state until BUSY returns HIGH. Although halting the processor is costly in processing time, it does have the advantage of reducing microprocessor noise.

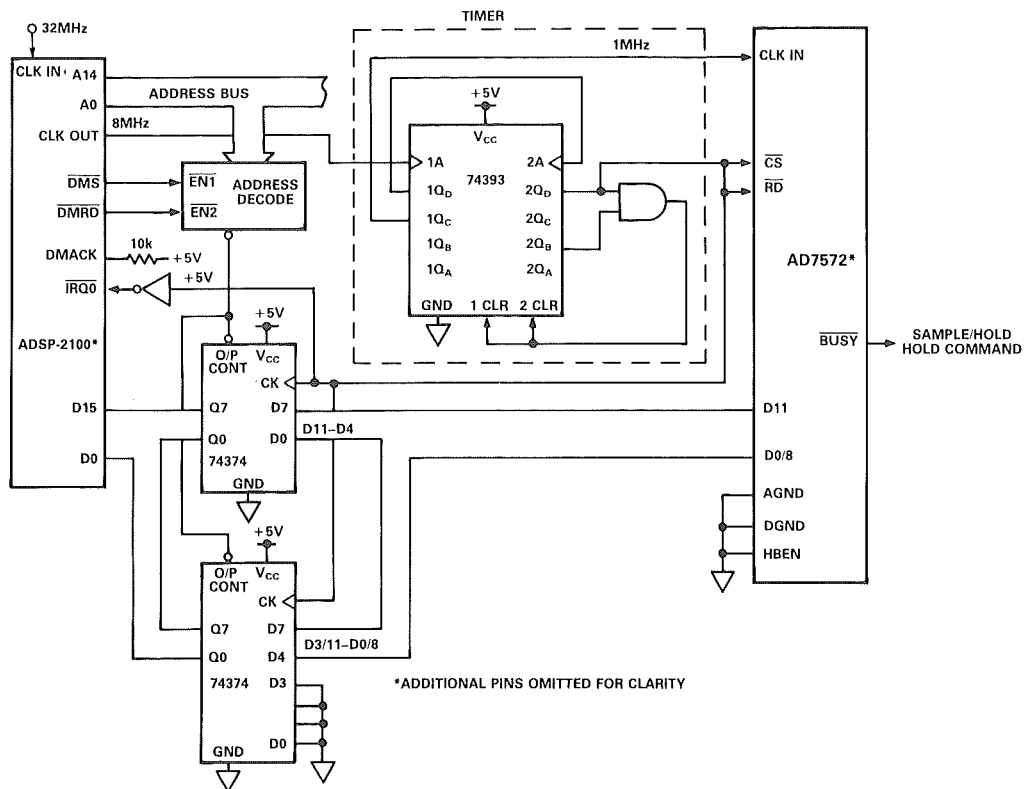
EQUAL SAMPLING INTERVALS

In digital signal processing, sampling the input signal must occur at exactly equal intervals. This minimizes errors due to sampling uncertainty or jitter. Precise timing with a microprocessor involves counting clock cycles and calculating software loop delays. A better solution is to use an external timer such as the software programmable 8254 to control conversion. A 74393 counter is another possibility.

A typical interface to the ADSP-2100 is shown below. The 74393 counter uses the processor's CLK OUT to generate an AD7572 CLK IN providing accurate conversion control. With the processor's CLK OUT at 8MHz, the counter will start conversion every 160 CLK OUT cycles. This results in a sampling rate of 50kHz. The timer pulls $\overline{CS}$ and $\overline{RD}$ LOW to start conversion and $\overline{BUSY}$ goes LOW asserting the HOLD input to the Sample-and-Hold amplifier for the duration of the conversion. $\overline{CS}$ and $\overline{RD}$ remain LOW for 16 μ s. The rising edge of $\overline{CS}$ and $\overline{RD}$ latches the data into the 74374 latches and interrupts the processor. The interrupt input is set to be edge sensitive during program initialization eliminating the need to be reset during the interrupt service routine. The following single load instruction in the interrupt service routine reads the A/D data into the MR0 register

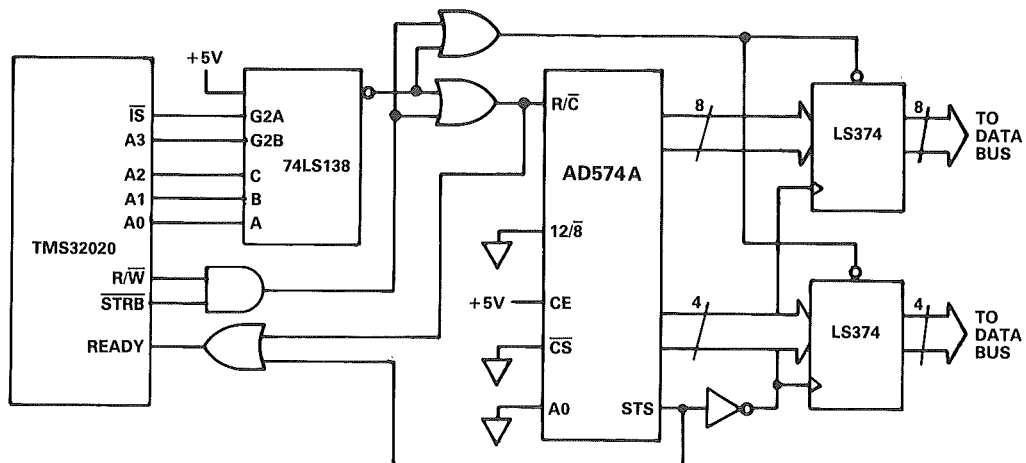
MR0 = DM(A/D ADDRESS)

AD7572/ADSP-2100 INTERFACE



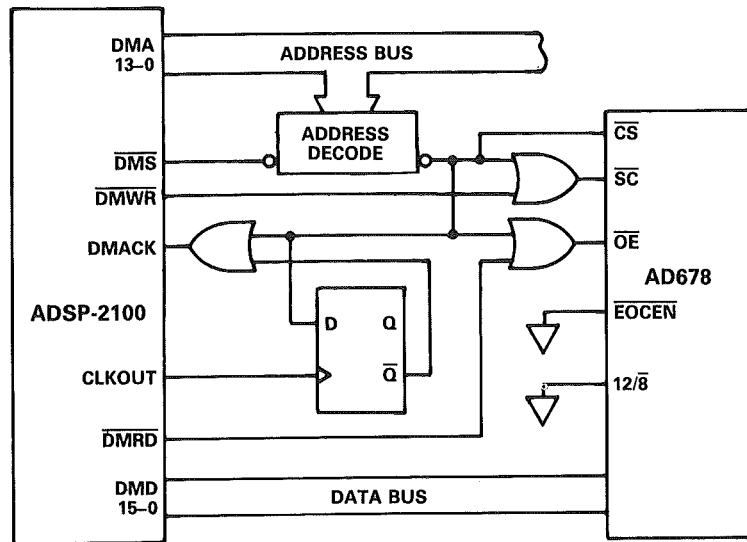
The industry standard AD574A 12-bit A/D converter can be used with the TMS32020. Using the Stand-Alone configuration previously described, this converter will interface to the processor with minimal hardware. The processor's $\overline{IS}$ signal distinguishes the I/O space from the local program/data memory space and is used to enable a 74LS138 address decoder. The decoded port address is then gated with the $R/\overline{W}$ and $\overline{STRB}$ signals to provide the low-going $R/\overline{C}$ pulse to start conversion. Data is latched on the falling edge of STS and can be read by enabling the three states during a read operation.

TMS32020 – AD574A INTERFACE



The AD678 12-bit Sampling A/D is shown interfaced to an ADSP-2100. With a clock frequency of 8MHz, and instruction execution in one 125ns cycle, the processor will support the AD678 data memory interface with a single wait state.

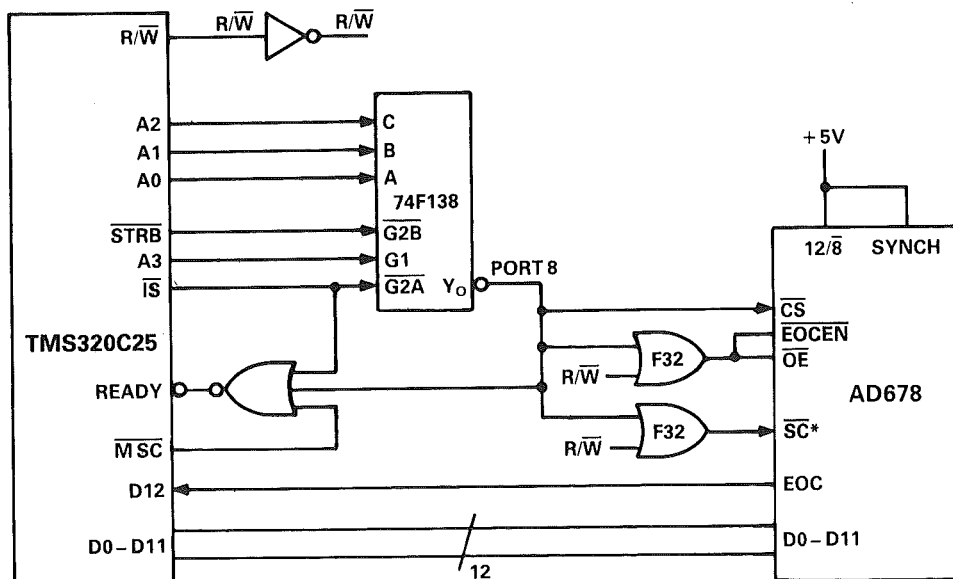
ADSP-2100 – AD678 INTERFACE



At the beginning of the data memory access cycle, the processor provides a 14-bit address on the DMA bus. The $\overline{DMS}$ signal is then asserted enabling a LOW address decode and the AD678 $\overline{CS}$. The processor issues $\overline{DMWR}$ which is gated with the decoded address to start conversion. The LOW decoded address is also OR'ed with the $\overline{Q}$ output of a D flop-flop to pull DMACK LOW. This forces the ADSP-2100 into a wait state for 1 clock cycle. The rising edge of CLKOUT latches $\overline{Q}$ HIGH bringing DMACK HIGH. 5 μ s later, the conversion is complete. The processor can now start a data memory access cycle to read data. For this cycle the $\overline{DMRD}$ and LOW decoded address are OR'ed to generate $\overline{OE}$ for the converter. Once again, a single wait state is inserted allowing data to be read from the bus.

In the following figure, the AD678 is mapped into the TMS320C25 I/O space. An OUT instruction to Port 8 initiates the conversion. EOC status and the conversion result are read with an IN instruction to Port 8. A single wait state is inserted by generating the processor READY input from $\overline{IS}$, Port 8, and $\overline{MSC}$. This configuration supports processor clock speeds of 20MHz and is capable of supporting 40MHz operation if a NOP instruction follows each AD678 read.

AD678 TO TMS320C25 INTERFACE



*FOR ASYNCHRONOUS OPERATION, CONNECT SYNCH TO GROUND AND APPLY EXTERNAL TRIGGER TO SC.

