CMOS CONVERTERS AS I/O DEVICES

Memory-Managed I/O Architectures Using ADI Microprocessor-Compatible IC Converters

by Ivar Wold

Elsewhere in this issue, you have read about a complete analog-to-microcomputer I/O interface, the RTI-1200. We have described its ability to interface as memory and have mentioned its "card-select" feature, which permits a number of such devices to occupy the same block of memory interchangeably when addressed (Figure 1). These features, "vertical" and "horizontal" memory-managed (or memory-mapped) I/O, as mentioned, offer significant advantages and are becoming well-recognized in the industry.

The many inherent features of the RTI-1200, including its clean analog design, thorough software documentation, on-board PROM capability, and total compatibility with the Intel SBC-80/10 microcomputer, establish its architecture as the definitive general-purpose data-acquisition-system interface for systems involving 8080-type microprocessors. We expect it to gain widespread acceptance–as a system building block to accompany the SBC-80/10–among busy system-designers who value their time above all else.

Despite its flexibility, however, there is a price paid in some applications: some users might find that they need only a few of its features, others might have wished for a different architecture (for example, a converter-per-channel), yet others must save money on parts cost and, in any event, "would rather-do-it-myself."

The good news, as evidenced in these pages[1,2,3] on several recent occasions, is that low-cost IC converters, readily available from Analog Device, can be used with little external circuitry to provide conversion and interfacing to microcomputers. They can provide memory-managed I/O with but little external logic. We shall discuss the devices and the principles involved in interfacing them to microprocessors. Since the details of analog application circuitry and performance have been covered earlier and are available on data sheets, we shall confine this essay to their connections in the digital domain and software.

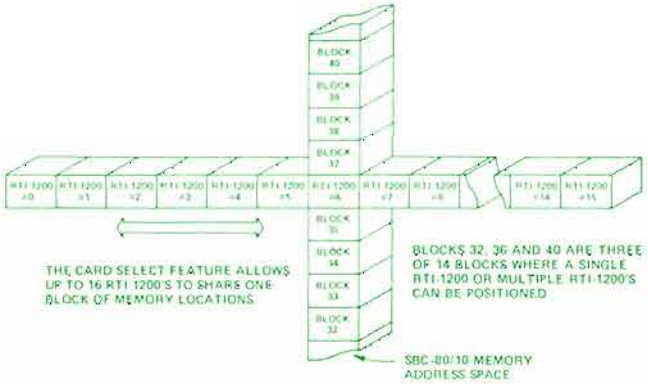

Figure 1. Vertical and horizontal memory-managed I/O as applied to the RTI-1200.

[1] *Analog Dialogue* 9-2, cover story (AD7570)
[2] *Analog Dialogue* 9-3, page 10 (AD7522)
[3] *Analog Dialogue* 10-1, cover story (AD7550)

The (first-generation) products to be discussed are:
- AD7522 10-bit multiplying d/a converter
- AD7550 13-bit 2's-complement quad-slop a/d converter
- AD7570 10-bit successive-approximation a/d converter

The digital data terminals of these devices employ 3-state logic and are byte-addressable. This means that no peripheral interface circuitry is required, and the digital lines can be connected directly to the bidirectional data-bus of an 8-bit microprocessor with clock speed compatible with the device *enable* time. The user provides only the address decoding appropriate to the application for the control and data-read/write functions.

WHAT IS A MICROPROCESSOR?

For our purposes, it is useful to consider that, like all stored-program digital computers, a microprocessor is basically a memory controller. Its primary function is to fetch instructions from memory, read data from memory, and write data back into memory (a satisfying internal game, but unfortunately, this cozy little world must communicate with the outside world, which consists of peripheral I/O devices –Teletype-writers, CRT terminals, and printers–for communicating with human beings, and converters for dealing with the transducers that measure and control real-world variables).

Figure 2 shows an idealized microprocessor, which will be used throughout this article to describe the interface techniques favored by the author (and in use for some time at Analog Devices). While most microprocessors provide special I/O signals and I/O instructions to deal with external devices, it is easy to conclude that it is simpler and more elegant to make these input/output devices and their associated registers appear to the microprocessor as memory locations. To review the advantages:

- Since different microprocessors communicate with memory in essentially the same way, it becomes simpler (for both user and vendor) for a company like Analog Devices to furnish memory-managed-I/O integrated circuits that will communicate with all available microprocessors.

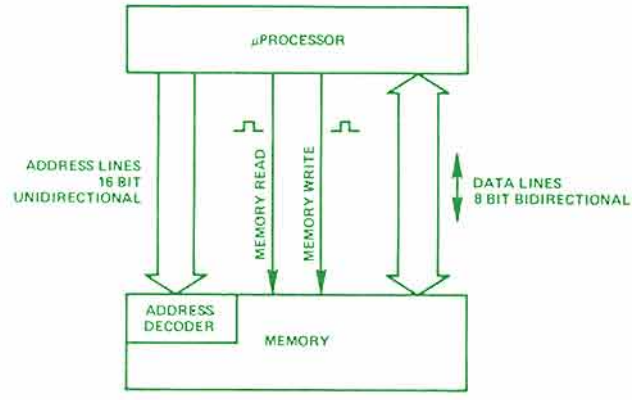

Figure 2. A micro-processor is a memory-controller.

•Most microprocessors offer a larger repertoire of memory-reference instructions than of I/O instructions, and , in particular, many processors–such as the 8080–provide a double-precision 16-bit memory load-and-store instruction, which becomes directly available for input/output control.

•A very-large number of input/output ports become available, with a much-wider range of choice over their disposition. Essentially the entire memory space is accessible to I/O devices.

•The use of memory ports for I/O permits the use of "horizontal" memory-managed I/O (*card select* in the RTI-1200). This technique allows the design of highly structured operating-system architectures, which can sidestep the difficulties posed by lack of relative- and index-addressing modes in the 8080. Horizontal memory-managed I/O is particularly applicable to general-purpose systems that must retain a high degree of flexibility of I/O device configuration, while the simpler vertical memory-managed I/O is adequate for special-purpose applications, where circuit configurations are fixed.

CMOS CONVERTER ARCHITECTURE

Several years ago, when we decided to design a range of microprocessor-compatible a/d and d/a converters, the first dilemma that faced the designers was the complete lack of standards for microprocessor interfaces. It would clearly be desirable to have integrated circuits that were compatible with all the μP's in the marketplace–with 4-bit, 8-bit, and even 16-bit processors. Also, it would be useful to have IC's that would fit easily into higher-level system structures, such as operating systems. Given the advantages of memory-managed I/O, we decided to adopt that concept as a standard approach to be used in present and future IC converter families designed for direct microprocessor interfacing.

The devices to be discussed here are unique in that, while they offer the ability to communicate on an 8-bit data bus, they can also be applied conventionally in the full-parallel mode.

A good example of the approach is the AD7522* 10-bit (4-quadrant multiplying) digital-to-analog converter (Figure 3). A double-buffered converter, it has two sets of input registers. The first set consists of an 8-bit register and a two-bit register; they can be separately loaded. The outputs of these two registers can be strobed in parallel into the DAC register for full-parallel monotonic update of 10-bit analog-output data (500ns§ access time for LDAC, HBS, LBS).

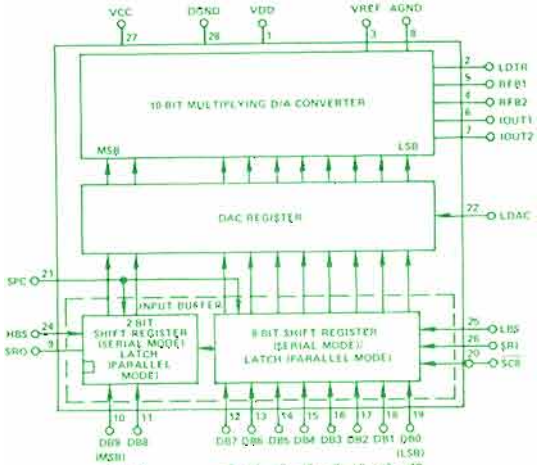

Figure 3. AD7522–double buffered multiplying digital to analog converter. Relevant control inputs are: HBS–High-Byte Strobe, LBS–Low-Byte Strobe, LDAC–Load DAC.

*For technical data on any of the products mentioned here, use the reply card.

§These specs appear on all data sheets issued since January, 1976.

INTERFACING THE AD7522 D/A CONVERTER

It is easy to see that it is a simple matter for a microprocessor to first load the 8 least-significant bits into the 8-bit register, then load the two most-significant bits into the 2-bit register, and finally, to strobe 10-bit data into the output register, for the d/a converter. Note that the DAC register can also accept data in the form of a stream of 10 serial bits–and shift them out as well as in.

Figure 4 shows how the AD7522 is connected into an 8-bit data bus. The bus is wired directly to the 8 least-significant bits;

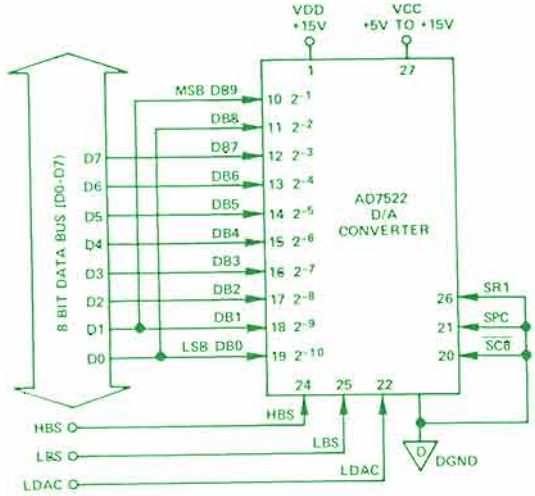

Figure 4. AD7522 – Connection for 2 byte operation.

and the two most-significant bits of the converter are wired to the two least-significant bits of the bus.†

Figure 5 shows two AD7522's (configured as in Figure 4), interfaced to our "ideal" microprocessor. Since the AD7522 was designed as a compromise for both parallel and byte-serial operation, the external address-decoding logic is necessary as shown. Nevertheless, the interface is extremely simple and can allow either simultaneous or non-simultaneous update of the two d/a converters. The *or* gates allow a single memory address to update the output registers of both d/a converters simultaneously. It is worth noting that many μP's (the 8080 included) incorporate 16-bit data instructions, which would allow the processor to output the data to both converters with a single *memory write* instruction.

†The reason for this is that most μP's use formats that are "right-justified" for ready interpretation as integers, i.e., the LSB (Data-Bit 0) of a one- or two-byte n-bit word corresponds to an integer with a weight of 1; the MSB has a designation of n-1 and a weight of 2^{n-1}. Converter-users, on the other hand, are accustomed to a left-justified fractional format: the LSB (Bit n) has a weight of 2^{-n}, and the MSB (Bit 1) always has a weight of 2^{-1}. The difference in format, for a 10-bit word, can be seen in this comparison:

μP: XXXXXX10 11101011 vs. 10111010 11XXXXXX

(For better or for worse,) the devices discussed here are formatted for ease-of-use with popular μP's rather for either historical continuity or highest analog accuracy in the single-byte 8-bit mode.

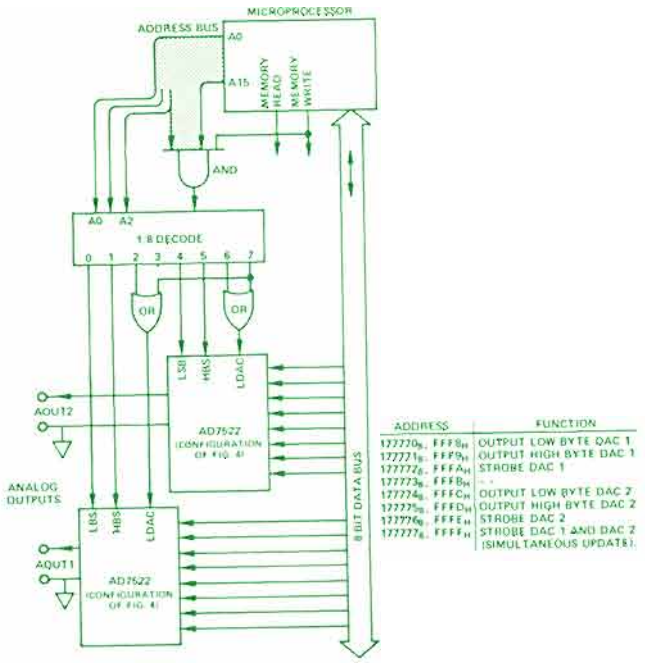

Figure 5. Interfacing multiple AD7522's to a microprocessor.

INTERFACING A/D CONVERTERS

The AD7550 is a high-precision essentially self-contained (except for the reference) 13-bit integrating a/d converter capable of maintaining zero-and gain-stability better than 1ppm/°C (Figure 6). The AD7570 is a 10-bit successive-approximations a/d converter that is essentially complete with an external comparator and reference. We will use the AD7550 as the context for our ADC interfacing example; the AD7570 is somewhat similar, but has 650ns *enable* time.

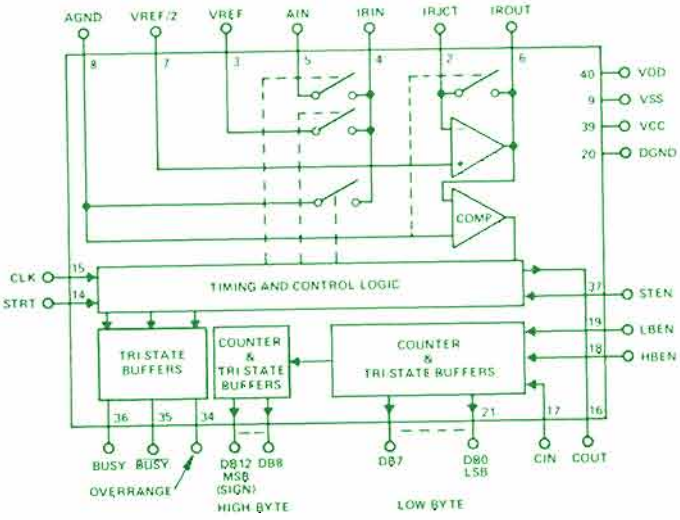

Figure 6. AD7550—13 bit analog-digital converter. Relevant control inputs are STRT—Start Conversion, STEN—Status Enable, LBEN—Low-Byte Enable, HBEN—High-Byte Enable; Relevant control outputs are BUSY, $\overline{\text{BUSY}}$—Status of conversion. Reference, voltage divider, and integrating RC are external.

Figure 7 shows how the AD7550 may be connected to an 8-bit data bus. The outputs of the AD7550 are arranged in three groups, each having 3-state outputs and independent *select* lines. The three groups are:

- Low 8 bits of the data word
- High 5 bits of the data word and the overrange bit
- The "status" bit (busy signal)

When wired as shown, each of the three groups can be selected and enabled (500ns access time, max) for transmission on the data bus. A fourth signal, START, is required to initiate conversion.

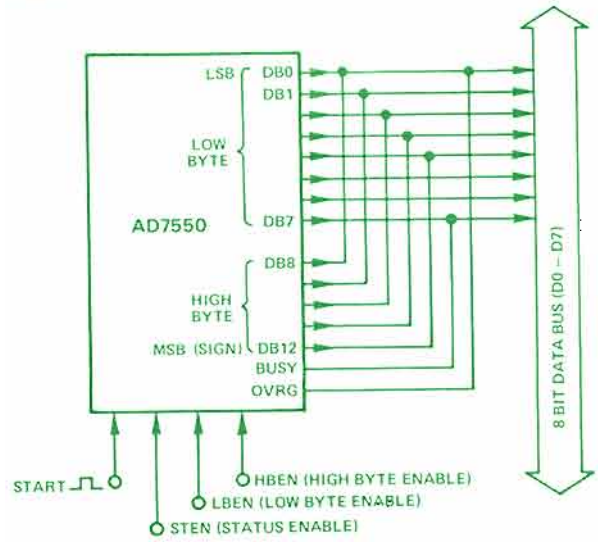

Figure 7. Interfacing the AD7550 to an 8 bit data path. Coding of AD7550 is 2's complement.

Figure 8 shows how two AD7550's, connected as described above, would interface to the "ideal" microprocessor. As in the case of the AD7522, since the AD7550 was also designed for conventional, full-parallel –as well as byte-serial– operation, a few external logic components are necessary. The resulting interface is nevertheless quite simple.

Figure 8 is similar to Figure 5, except that the circuit is wired to perform MEMORY READ operations. Note that one of the addresses for each AD7550 is decoded for the purpose of obtaining a *start* conversion command; the microprocessor must issue a dummy *read* command to start a conversion. Following

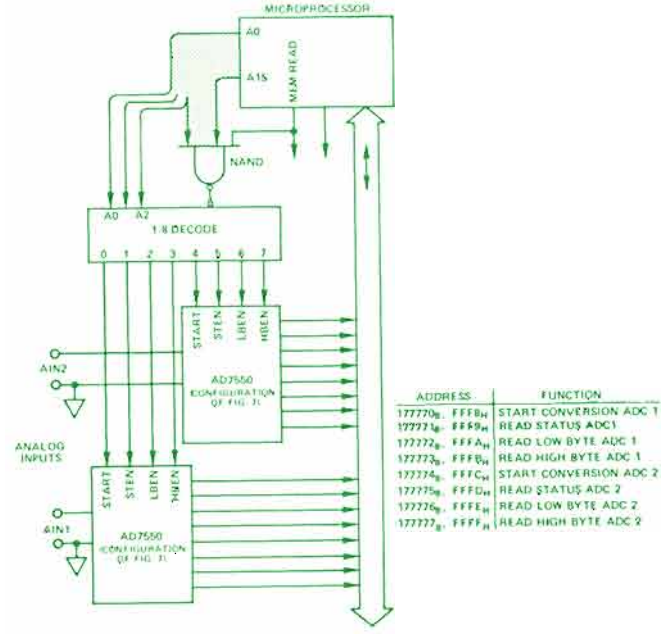

Figure 8. Interfacing multiple AD7550's to a microprocessor.

the *start* command, the microprocessor can examine the status of conversion at will by reading data from the *status* addresses. When this datum indicates that the conversion process has been completed, a single 16-bit MEMORY READ instruction will provide the 13-bit digital value. If it is desired that the end of conversion produce an interrupt request, the *status* line could be wired to an interrupt request line and enable at the start of conversion.

This scheme can be extended to embrace any number of ADC's, or indeed, any, number of combinations of ADC's and DAC's for an analog in/out subsystem.

HORIZONTAL MEMORY-MAPPED I/O

Figure 9 shows an architecture for horizontal memory-mapped I/O. In this example, all devices are mapped into the upper 1024 (1k) memory locations. The *and*-gate G1 provides a *map* signal whenever the upper 1k of memory is addressed.

All devices contain a *select register* (SR), which is loaded with data whenever the highest (177777_8 or $FFFF_H$) memory location is addressed (detected by *and*-gate G2). To select one of the devices, the μP outputs a device number to location $FFFF_H$, which causes the SR's of all devices to be loaded with the device number. All devices compare this number to their own (different) device numbers, and the one which finds itself selected *enables* its device circuits (ROM, RAM, registers, etc.) to respond whenever the μP references the top 1k of memory.

Using this configuration, a given device-access routine or device parameter can be assigned a fixed entry location within the *map* area, directly analogous to the displacements within a device driver based on indexed addressing. So, clearly, this memory-mapped I/O configuration allows the structuring of a *device driver* for general-purpose μP-based data-acquisition systems, even for μP's which lack proper indexed-addressing modes.

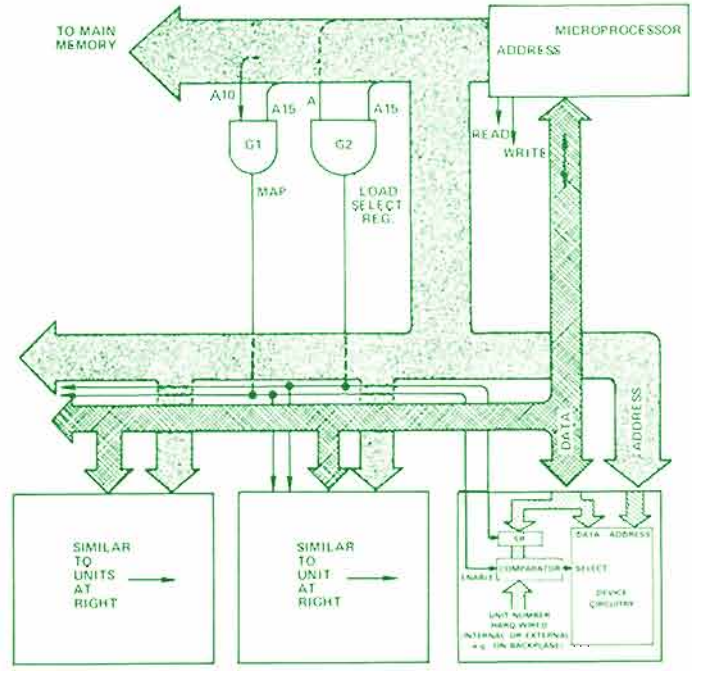

Figure 9. Horizontal Memory-Mapped I/O.