"Sample" the 6100

James Downey
5505 Vanderbilt Drive
Mobile AL 36680

Most of the attention of the ever-growing body of microcomputer enthusiasts has been concentrated on a mere handful of processors. One that has generally been overlooked to date is the Intersil/Harris 6100. Why experimenters have looked elsewhere for that ideal chip set is not entirely clear. They may have heard that the 6100 has an antiquated instruction set. Perhaps they were wary of the 12-bit word length or felt uncomfortable with CMOS technology.

I suspect, however, that the greatest reason is simply a lack of company. To put together a system from scratch is a formidable undertaking. Furthermore, by computer industry standards, the financial investment in a microprocessor system may be minuscule, but by most hobby standards it is still enough to make most of us wince. Needless to say, few will embark on such a project unless they are assured of a favorable outcome.

Pro 6100

Why, then, if the 6100 is so poorly represented in the personal computer world, should anyone even consider it? Far and above, the main reason is the PDP-8[1] instruction set. Digital Equipment's sales of these minis over the past decade or so compare favorably to McDonald's volume in hamburger sales.

Due to the immense popularity of that machine in the industry, a great deal of PDP-8 code has been written and is very much available, not at the corner electronic store, however, but primarily through Digital Equipment's software distribution centers, where assemblers, editors and high-level languages are available for very reasonable prices. It's also available through the worldwide Decus Society, where hundreds of PDP-8 programs are available with documentation for only a minimum charge, usually $2 for paper tape.

Another plus not to be overlooked is that you may have had previous PDP-8 experience. Most who have had a close encounter of some kind with one of these creatures find them quite friendly and, at one time or another, usually fantasize about having one for their very own. And, after all, if you are comfortable in one language, why learn another?

Finally, there are those among us who just like to do things a little differently than the next guy. The all-CMOS 6100 is indeed something different and is guaranteed to raise eyebrows when you show up at a club meeting with it.

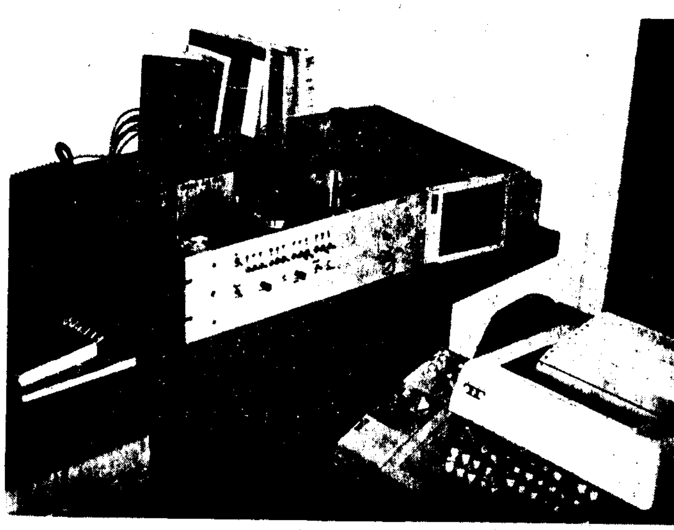
Construction

By far, the best way to scratch-build the 6100 system is to purchase the CMOS sampler package[2] manufactured by Intersil. At $49, it includes a CPU chip, 256 words of CMOS RAM, a ROM containing an excellent keyboard monitor, an interface chip, a UART and a baud rate generator. It includes excellent manuals that are sufficient to get even the greenest of beginners up and running.

Software operation of the ROM-based monitor is covered in detail by the sampler manual, and a commented listing of the ROM's contents is provided. In addition, the sampler manual gives several interface options for the serial terminal and shows how to implement some handy front-panel options such as hardware single step and data displays.

I built my original system with wire-wrap construction on a general-purpose Augat board. I have also built the sampler system on perfboard with glued-in sockets, and the results were the same. For under $40, a high-quality board, the 6960 sampler board, which will accommodate the sampler chip set, is available from Intersil. The board includes: status indicators, hardware single-step control and a serial interface. The perfboard is the cheapest and the Intersil board is the slickest, so take your choice; they all work.

For my system, I gutted a surplus power supply for the 19 inch chassis and salvaged a strip of 30 pin connectors (originally part of a computer backplane), which I use for an expansion bus. A good choice for a bus would be the common 44 pin edge connectors. These accept the 4 1/2 inch × 5 inch cards and are readily available. Due to the 12-bit word length and the common address/data bus, the system does not lend itself well to the popular S-100 configuration. My experience with the system is that it is quite noise tolerant, and about any layout will work.



The author's 6100 system featuring 4K of RAM, two channels of D/A, eight channels of A/D and a real-time clock.

One construction consideration you should observe, however, is to keep heat-generating devices such as regulators, Tristate buffers or non-CMOS

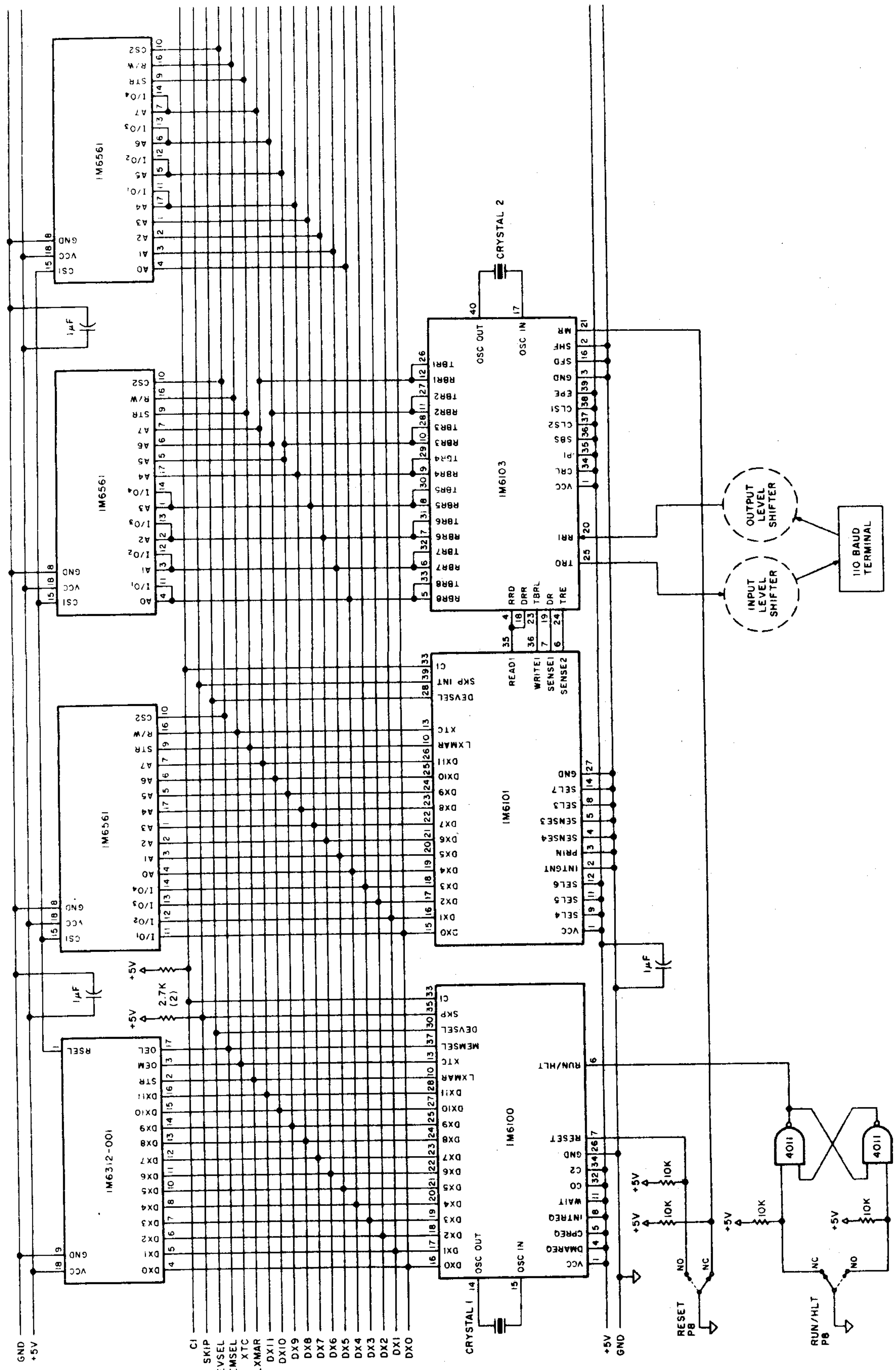


Fig. 1. The system schematic for the 6100 sampler system. Note the single DX bus replacing the traditional data and address buses.

RAM away from the LSI CMOS chips. Unlike other microprocessors, they are cool-running and seem to require it that way.

Fig. 1 shows the sampler system interconnections. Notice that the address and data lines are not separate. Everything is time-multiplexed on one set of bidirectional DX lines. This immediately eliminates 12 lines from the bus as compared to that used on most micros. This is why 30 lines in my bus are more than adequate. The ROM is similar to Motorola's MIKBUG in that it is Teletype-oriented. The ROM includes a binary formatted paper-tape punch and load routine, so an ASR 33 or equivalent is the only sensible way to go with this system.

An inexpensive cassette interface for the 6100 system is available ($25 from The Bit Stop, Box 973, Mobile AL 36680), but its software is currently not available on ROM; you must still use paper tape to initialize the system. The ROM features an emulation of Digital's popular octal debugging technique and is first-rate as a keyboard monitor.

Options

The CMOS sampler manual shows how to enable several panel options, some of which I incorporated on my system but really are not necessary. One option not covered by the manual but highly recommended is the switch register. When the machine executes a 7404_8 instruction (OR the switch register), the settings on the 12 panel switches are logically ORed into the accumulator. This is by far the easiest way to communicate with the machine, since only one instruction is required to bring in a 12-bit word versus a fairly large amount of code required to do this from the keyboard.

More important, much of Digital's software employs this instruction making the switch register a prerequisite for any user who anticipates using those packages. Fig. 2 shows how the data from the switch register is strobed onto the DX lines with Tri-state buffers.

PDP-8	Sampler	Operation
6036*	6160	Read one character
6031	6162	Skip when character received
6046	6161	Transmit one character
6041	6163	Skip when transmit done
NA	6166	Start reader
NA	6167	Stop reader

*6036 also advances the reader one character automatically; 6160 does not.

Table 1. Comparison to Teletype instructions between the 6100 sampler system and Digital Equipment's PDP-8. In general, simply substituting corresponding code from the sampler column for code in the PDP-8 column is all that's needed to configure a PDP-8 program for the 6100. A word search routine is present in the sampler ROM making it easy to find the offending code.

The sampler package includes 256 words of CMOS memory. You will be amazed at what you can do with this limited space, but soon you will get the urge to increase memory size. For my system, I considered Intersil's CMOS memory too expensive—about $85/K—and used the cheaper static RAM as shown in Fig. 3.

I used 2114 chips, which are 1K by 4, running about $36/K. The 1K × 1 21L02s could be used to lower the price to around $24/K, but because of the 12-bit-wide word, this means a lot of wiring. The 2114 layout puts 2K on a board, and these chips will easily fit on a 4 1/2 × 5 experimenter card.

The 12-bit address structure of the 6100 only allows for 4K of addressable memory (this can be expanded to 32K using the Intersil 6102 memory expansion chip), and I have, therefore, limited my system to 4K for the time being.

The fourth K of memory is occupied by the ROM. Though Digital's assembler and editor will run in 3K of memory, FOCAL, BASIC and the 23-bit floating-point package require the full 4K. I therefore put a fourth K of RAM opposite the ROM card and simply put a toggle switch on the chip select line (MEMSEL) to select from one to the other. Working in the fourth K of RAM is a little tricky, since the keyboard monitor must be deactivated, but it's nothing a little creative programming can't get around.

The simplest solution is to load Digital's ODT Low, a Teletype-oriented monitor that occupies about 375 locations in low memory, execute ODT and throw the RAM/ROM switch . . . and "presto," you have a PDP-8 just like the real thing. If your program doesn't use the fourth K, then the ROM ODT is that much handier, since it can't accidentally be wiped out.

By the way, you need not populate the entire 4K all at once. Just one K makes a fine starter system if only part of the memory is populated; however, be sure some memory resides in location $0\text{-}200_8$, since the ROM uses this as its scratchpad.

Other useful options included in the sampler manual include an LED on the run/halt line. This allows the user to tell at a glance if the processor is in a halt state. I also added Tri-state buffers to the lines exiting the CPU as shown in the 6100 manual that accompanies the sampler kit.

If you are sure you will stay with a simple 4K system and an ASCII terminal, you can eliminate the buffers and run the CPU barefoot right into the DX lines. However, let me warn you, the system is extremely simple to interface through the 6101 PIE chip, and it probably won't be long before you hang enough peripherals on it to exceed the capacity of the bus. I, therefore, recommend that the buffers be incorporated early in the design.

On my system, I currently have eight channels of A to D, two channels of D to A, a cassette interface and a real-time clock running. Remember, with memory extension hardware, another 4K of memory

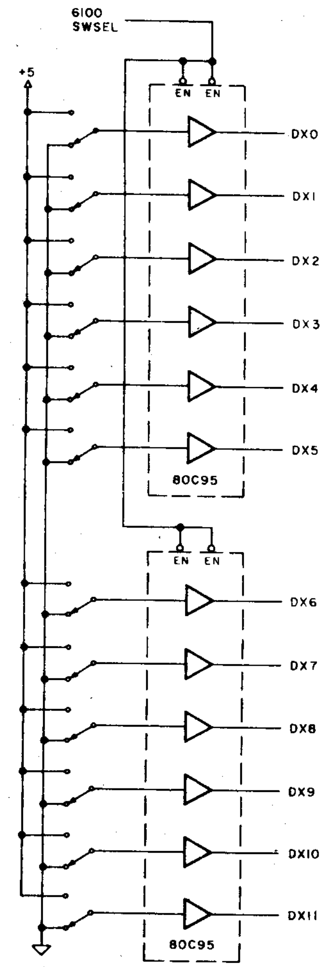


Fig. 2. Hardware required to implement the or-the-switch-register instruction (OSR). Much of Digital Equipment's software uses the switch register. Also, it's the easiest way to put a 12-bit word into the system, since only one instruction is required.

and a floppy disk, the system could accept the capable OS/8 software with two levels of FORTRAN, BASIC and a host of other goodies, so don't think too small at this stage.

For power I have 2.5 Amps at 5 volts. Though the CMOS system draws negligible current, the 4K of RAM will draw about an Amp. With the Tri-state buffers and some LED displays, this adds another half of an Amp or so. A tenth of an Amp of −12 is also needed for the Teletype interface.

The sampler manual claims the commercial grade chips will perform well at a clock frequency of 3.5 megahertz (a color TV crystal); however, I encountered some problems with the UART at this speed, and it

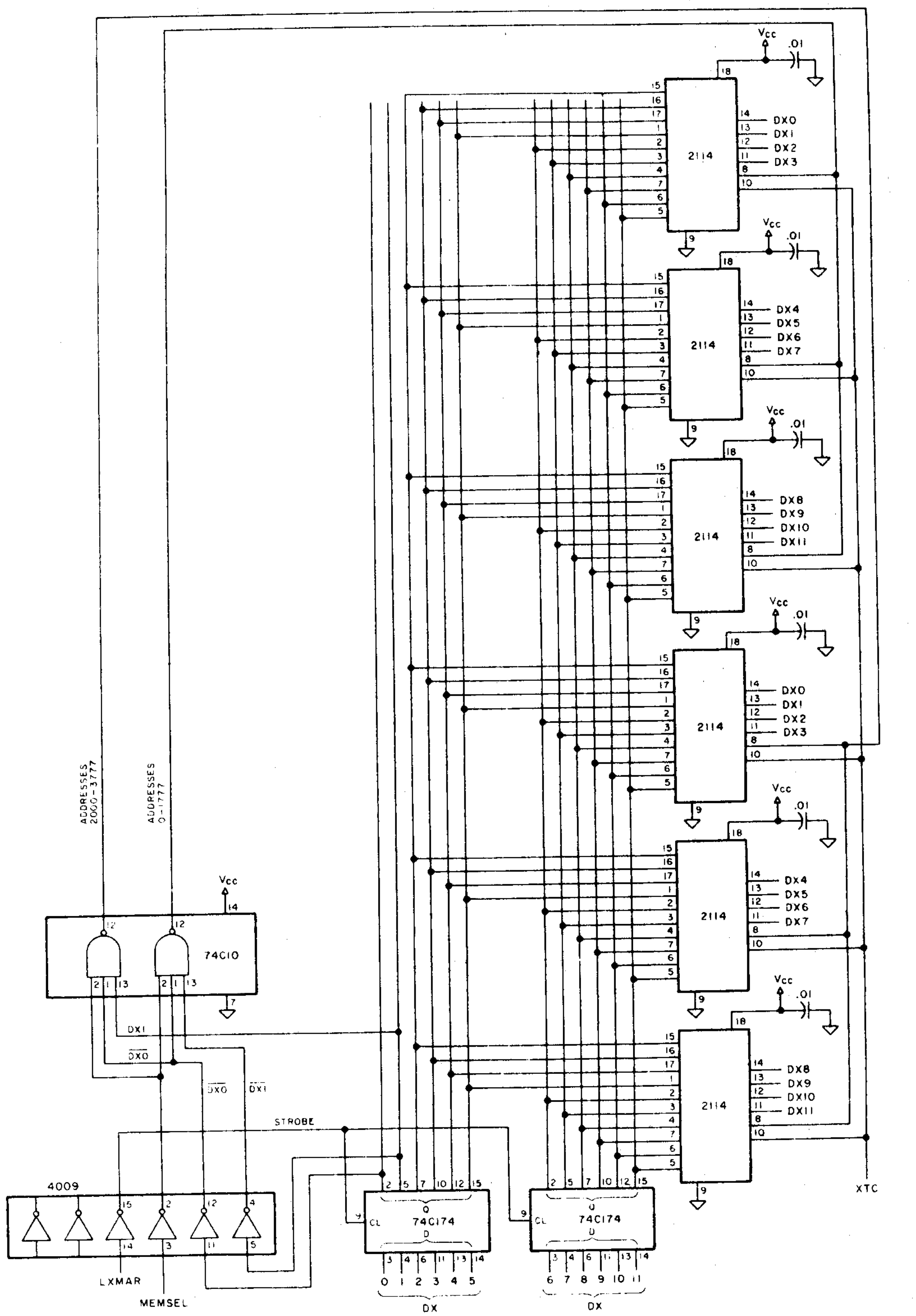


Fig. 3. Static RAMs are easily interfaced to the 6100. Notice how the two most significant bits, 0 and 1, are used to select each K of memory. Addresses are latched off the DX lines using LXMAR as a strobe.

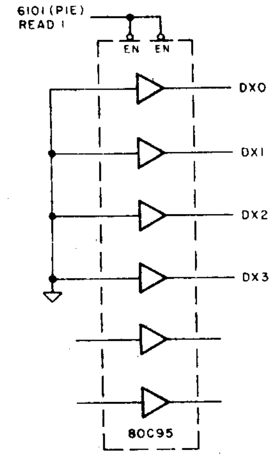


Fig. 4. Hardware to force DX lines 0-3 to a logic zero while reading from the terminal. This modification eliminates the need to mask out these bits in the program, making the system more compatible with Digital Equipment's software.

also puts the 2114 memory chips slightly under their specified access time of 450 ns, though they seemed to work. When I substituted a 2 MHz crystal, all of these problems disappeared; yet the loss of processor speed was not really noticed. Later sampler kits have included a notice to this effect.

Software Compatibility

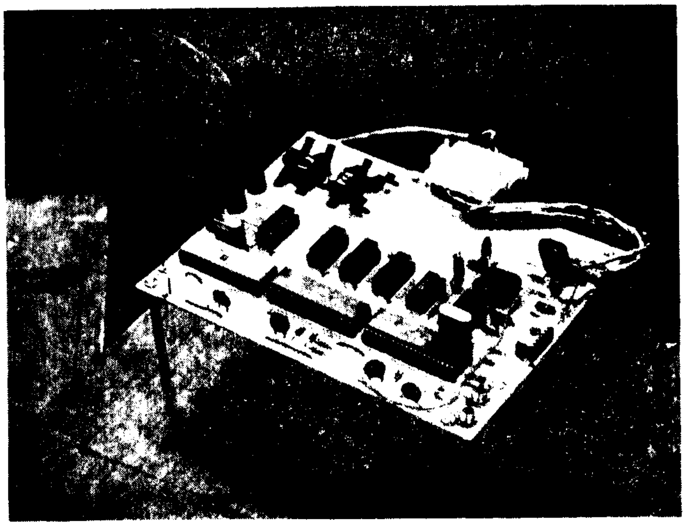
The I/O is very similar to the PDP-8; however, there are some differences, and patches will have to be made to PDP-8 software to make it compatible. A true PDP-8 interface can be built, but it is complex and, in my opinion, it is far easier to simply make the patches. Basically, these differences in I/O are shown in Table 1.

The ROM and ODT both employ a word search feature, so it is easy to locate the offending code and replace it. To enhance software compatibility, it is prudent to make the 6100's I/O as similar to the PDP-8's as possible. Notice in Fig. 1 that data read by the UART is deposited on the DX lines 4-11. Eight lines are used, since the Teletype sends 8-bit ASCII. DX lines 0-3 are unwired and will contain garbage. To get rid of the extraneous garbage, it is necessary to mask out bits 0-3 after each read in the software.

This added maneuver is often hard to squeeze into a tight PDP-8 program. By forcing these lines to zeros, however, with a Tri-state buffer as shown in Fig. 4, the garbage bits and the mask operation are eliminated, making the interfaces almost identical in structure.

Software availability is primarily through the Digital Equipment software distribution center in Maynard MA. The best way to obtain software, however, is to find a PDP-8 in the field and go through its box of goodies. They are widespread and not too hard to find.

Every PDP-8 leaves the factory with a tray of tapes including an editor, an assembler, a great floating-point arithmetic package and a copy of 4K FOCAL. A good place to start looking for an 8 is a local college or university. The best gold mine for software is the Decus Society; however, membership is limited to users of Digital Equipment's machines.



A minimal system configuration for the sampler kit. Built on perf-board, this system can easily be constructed for under $100.

So, when you find a PDP-8 (or a PDP-12 that is language compatible), ask its users about Decus. Also, ask them about licensing agreements, since some of Digital Equipment's software is protected by licenses. A listing of patches and a description of special operating procedures required to run the basic Digital Equipment software kit, including FOCAL, on the 6100 is available for $5 from The Bit Stop, Box 973, Mobile AL.

Finally, an excellent manual, *Introduction to Programming*, tells all about programming the 6100. It is available from Digital Equipment Corporation for $5 and is a must for the serious 6100 user.

In summary, I have found the 6100 system, as supplied in the CMOS sampler kit, an economical and easy system to build. The result is a reliable PDP-8 processor. That Digital is currently using the 6100 system in their new DEC-station version of the PDP-8 attests to this fact. When you add the software availability, the 6100 makes a fine system for the scratch-builder.■

[1]PDP-8 is a registered trademark of Digital Equipment Corporation, 129 Parker Street, Maynard MA 01754.

[2]Intersil part number 6801, available through Schwebber Electronics, Atlanta GA.