applications. In most of these cases, the cost requirements of the design will keep you away from high-current solutions anyway.

The second area in which power is a consideration is in battery-operated equipment. In some cases, you must choose a microprocessor with a specific maximum current to match the battery. In other cases, you must pick a microprocessor with a reasonable current requirement and then pick a battery to match. In either case, you need to know the total operational current.

A related issue is sleep current. Many microprocessors have a low-power mode of operation in which the CPU goes to "sleep," turning off internal peripherals to conserve power. Some microcontrollers have very low current in this mode; on others it saves so little power that I've wondered why the manufacturer bothered with it. Either way, you need to get an estimate on the amount of time the system will spend in this mode to have a good handle on battery life.

Environmental Requirements

For the purpose of choosing a microprocessor, environmental requirements typically translate into temperature. If your design must operate over an extended temperature range, such as designs for military or automotive purposes, your choices of available parts are more limited than if you have normal industrial temperature requirements. Note that extended temperature devices are nearly always more expensive, so don't base your cost estimates on the industrial parts if you really need high-temperature parts.

Life Cycle Costs

Are you making a VCR or a piece of industrial equipment? If you are making a VCR, you probably don't need to consider the need to reprogram the unit in the field or worry about long-term availability of replacement parts. VCRs are throwaway consumer items. On the other hand, if you are building some kind of industrial equipment that costs thousands of dollars and will be operating for many years, you have a different set of considerations. You must pick a processor and/or memory architecture that can be upgraded. You probably also want to design in some excess program memory so you will have room for upgrades, and you might make long-term availability of the microprocessor more important than cost.

Life cycle costs are also a factor on the front end of a design. The more widgets you will produce, the more upfront development cost you can stand. If you are selling VCRs, you might pick a very cheap microcontroller and spend a lot of money developing the software, making the software do everything to save on hardware costs. If you are building an expensive piece of industrial equipment, you may sell only a few thousand over the life of the product. In that case, you want to minimize the development cost. In addition, your product cost is not likely to be as sensitive to the electronics cost. In that case, you would probably pick a processor that has

good development tools and other features that will minimize engineering time required to develop the product.

Operator Training/Competence

Operator training/competence has an impact on processor selection because it affects the user interface. If you have a product with a fairly complex set of features and poorly trained operators (such as consumers using VCRs), then you may need a more sophisticated user interface. In some cases, you may need an LCD display and touch screen. This implies more processor horsepower and memory to store the screens and messages.

If the operators of your instrument are well trained, you may be able to use a less sophisticated interface. For example, an electrical engineer using an oscilloscope probably knows what horizontal and vertical resolution knobs are for and doesn't need an explanation for them. If the same instrument has a sophisticated math function with non-obvious controls, you may need the capability to display a menu or even a help screen for the user.

The "Real" Requirements

Sometimes you must look past the request for a specific feature to get at the real requirement. Some years ago, I worked on a product that was going to be designed as a replacement for a current product. In looking at the requirements for the new product, we found some users requesting easier access to change certain chemicals stored in the machine. Looking deeper, we found that the real problem was the capacity of the chemical storage. The only reason that users were requesting easier access was because they had to replace the chemicals too often. Providing additional capacity solved the real problem.

Development Environment

To develop applications on a microprocessor, some basic tools are essential:

- A development system or cross-compiler
- A PROM programmer
- Debug hardware

In the prehistoric days of embedded systems (before the IBM PC), the standard development system consisted of a computer from the company that sold the microprocessor ICs and a PROM programmer. The development systems were expensive, slow, and limited to developing software only for that manufacturer's parts. Some third-party companies had development systems as well. These also were expensive