
Serial Memory Interface: The benefits of SPI over I²C and μ Wire

by Carlos Martinez, October 1997

Introduction

In the dynamic semiconductor industry, conventions taken for gospel today are heresy tomorrow. In one of these classic battlegrounds, parallel busses, once undisputed leader in system data transfer, face a serious challenge from new serial interfaces. Recent introductions and improved standards make the serial interface an attractive choice for both memory and peripheral control applications. In the peripheral arena, IEEE1394, gigabit ethernet and Universal Serial Bus (USB) are gaining favor in many systems over parallel busses such as PCI, PCCard, IDE (disk drive), and ISA/EISA.

In the semiconductor memory world, code storage devices almost exclusively use the parallel bus. However, serial access data memories now find increasing acceptance as faster serial busses allow the designer to achieve the benefits without the drawbacks. This application note examines serial memory interface trends and evaluates the three leading interface standards.

Memory Density

As Parallel Flash memories and EEPROMs increase in density, manufacturing lower density devices becomes increasingly less cost effective, so these densities tend to “disappear”. This provides opportunities for serial memories to increase in density to “fill the gap” left by the discontinued parallel types. These serial memory devices prove ideal for many data storage applications.

The Emergence of Serial Memories

Serial memory devices have been around for more than 14 years. The early low density devices found their way into many applications and, because of their low cost, sold in high volumes. Typical early uses included system configuration or device identification. These application generally required very little storage and had few interface speed constraints. In recent years, the use of serial nonvolatile memories has increased at a phenomenal rate into applications ranging from cellular telephones to LAN systems to automobiles to industrial equipment. In many of these applications serial memories are beginning to displace parallel devices.



Figure 1. Trends in Memory Density

Serial memories began to make inroads against parallel devices in response to two related market requirements. First was a rapid growth of portable products, as demonstrated by the proliferation of laptop computers, palmtop computers, pocket cellular phones, pagers, etc. Portability needs drove component vendors to find ways to reduce the size and power of their product. Success in this area fueled more miniaturization and the trend points to continued miniaturization in the foreseeable future.

As devices become more portable, there is a growing desire by consumers to customize the product that they use. Products now are small enough that their owners carry them everywhere. They become part of the person's "life-style". When this happens, the product must adapt to the owners preferences and habits. This leads to an increasing need for user programmable customization.

As devices get smaller, there are different interface media, since a keyboard is not always practical. This leads to voice or pen inputs. To optimize performance the system must adapt to the user or be capable of adjustment for more desirable response.

Customization in a system also includes personal phone numbers, individual schedules, stylized sounds or pictures. To personalize a product the user might include records of credit card numbers, access code lists, medical history, insurance information, etc. As personalization and customization becomes more common, there is a desire for more. This results in an ever growing need for more nonvolatile memory.

One way to provide customized memory is to use a small piece of a large parallel Flash device often seen in many products. The Flash contains the operating code and can be updated in the field. Why not reserve a little of this space for data storage? The emerging consensus is that loss of some personalization data is manageable, but loss of program is catastrophic. Allowing writes of personalization data to the same device that contains the program code increases the likelihood of catastrophic program failures in the field

With each new generation of product, engineers find new ways to reduce the physical size of the unit and to reduce it's power, while at the same time increasing the performance. To do this, the engineer uses device integration (through the use of ASICs and more highly integrated semiconductors), smaller packages, lower operating voltage, lower current and innovative packaging. Serial memories are becoming increasingly important in managing programmability, board space and power. Historical trends show serial memory power requirements (Figure 2) and memory density per square millimeter (Figure 3) decreasing rapidly, with the trend continuing into the near future. Because of their small size, low power characteristics and isolation from program memory areas, serial memories are key in providing new portable solutions.

While serial memories seem to be the ideal solution, as densities increase they are troubled by one major limitation, speed.

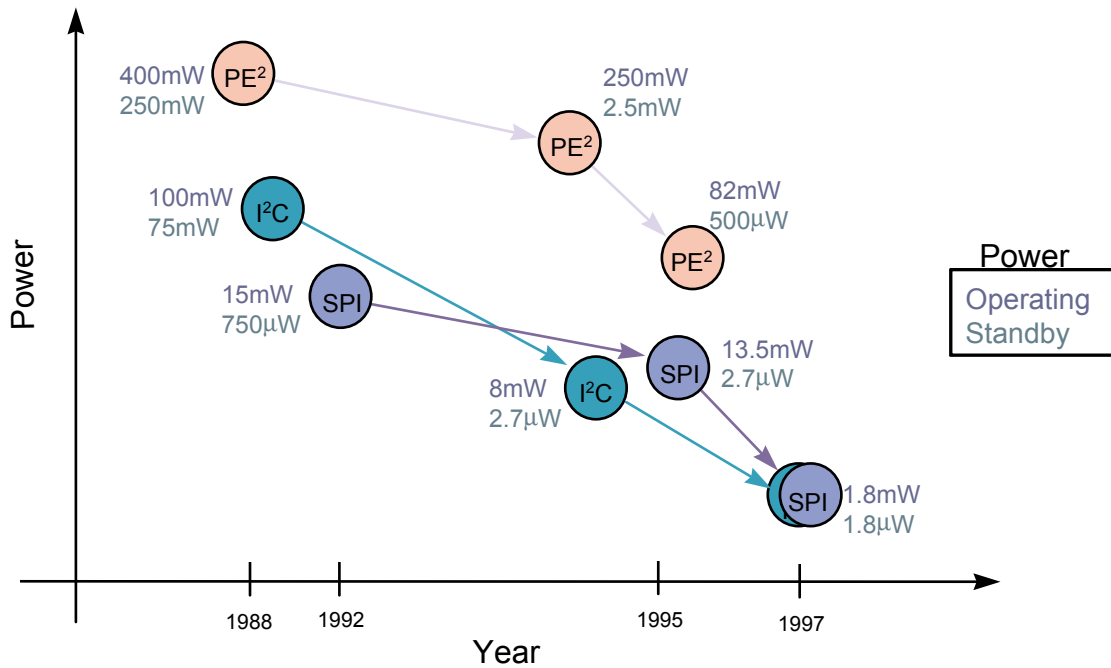


Figure 2. Memory Power Trends - With each new generation of devices power consumption is reduced. Future devices will have zero standby power

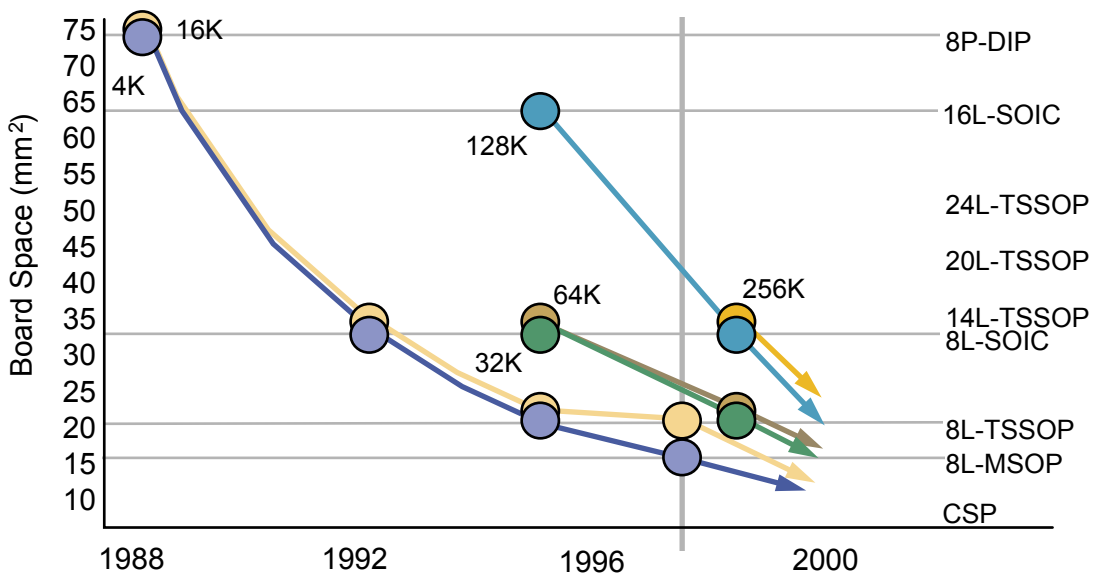


Figure 3. Memory Density/Package Trends - As process technologies shrink device geometries, more memory can fit into smaller packages. The ultimate package is the chip itself. Chip Scale Package technology, in development now, promises to radically reduce future package sizes.

The Importance of Speed

Two new serial peripheral interface standards, the IEEE1394 and USB, are examples of an increased push in the industry toward higher serial bus speeds. Promotions for the 1Gbit IEEE1394 standard indicate that it may be the preferred choice over the parallel PCI bus in the PC because of data throughput. At 1Gb/s, the IEEE1394 serial bus transfers 64-bits of data in less than 65nS. This is reportedly faster than current implementations of the parallel PCI bus, because of PCI protocol overhead. Designers predict the USB will replace the ISA bus and both the PC parallel and serial bus in the PC for low speed peripherals. The USB handles up to 12Mb/s. This is fast enough to handle audio I/O, telephone interfaces, keyboard and mouse control, printers and other standard PC peripherals. The USB is thought to have the inside track over other serial busses like ACCESS.bus (a derivative of I²C), mainly due to its much higher speed.

In serial memory applications speed becomes more important as the density increases. Reading 64kbits, at 100kHz takes over 600ms, while a 10MHz interface takes only 6mS. This can make a big difference in system performance, power consumption and user interaction.

Slow interface speeds impact system performance by adding overhead to the CPU. The need to wait for the next bit strains the ability of the processor to deal with real time activities or the demands of ever more complex software. This overhead can also affect the user of a product. The average person perceives a delay with response times in excess 200 to 300mS. Faster response times do not interrupt thought processes and this gives the user a more solid feel about the product. Slower

response times reflect poorly on the product, leading to dissatisfaction.

Interface speed, coupled with low standby current, can increase the battery life of a product. In many applications, a product operates at full power only a fraction of the time. Most often it is in some standby, low power state. Executing an operation faster reduces the time a product is in full power mode, allowing it to remain in a standby state longer (Figure 4).

Recent high speed serial memory product introductions from Xicor have generated tremendous interest. This is an early indication that the simple serial memory interface speeds will track the serial peripheral bus trends. Recent product introductions from Xicor include 400kHz 2-wire and 5MHz SPI devices. Figure 5 shows a graphical representation of current and projected serial bus speeds. Trends for the SPI bus is based on projections for expected host processor hardware capability and memory I/O design in the near future.

Interface Options

The earliest memory busses were the microwire bus and the I²C bus. The microwire bus began as an inter-processor communication port for the COPs microcontroller. The I²C (Inter-Integrated Controller) bus began as a peripheral bus, meant to interface to many devices with a single set of two wires.

SPI is a Synchronous Peripheral Interface that was created by Motorola. This interface is a more recent introduction that is proliferating as high speed becomes more important and as more products exploit the interface.

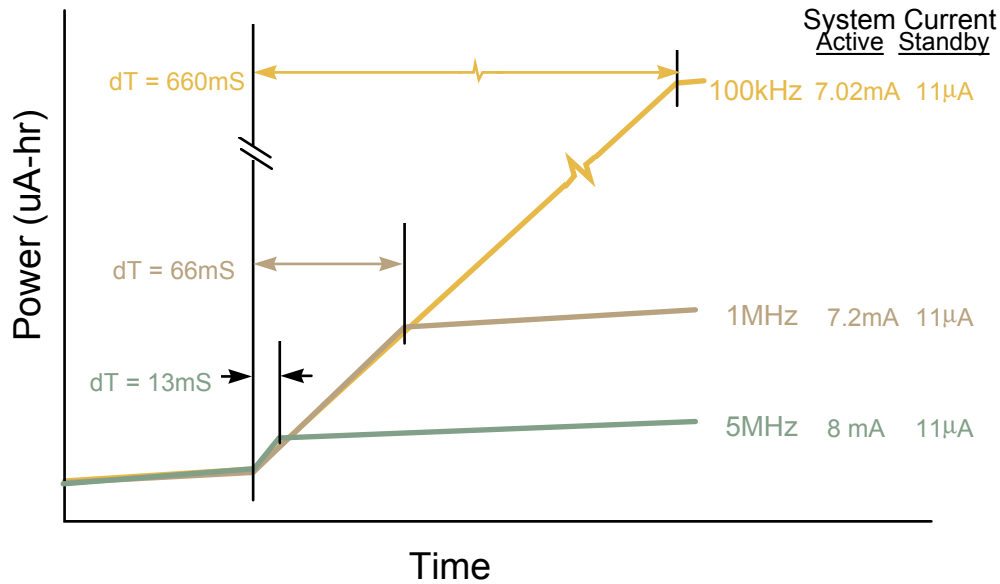


Figure 4. Power vs. Interface speed. This figure shows an example of a processor that draws 7mA active and 10µA in standby. The serial memory current increases linearly as the speed increases. Having the processor active for less time reduces system power, demonstrating one value of a high speed serial device.

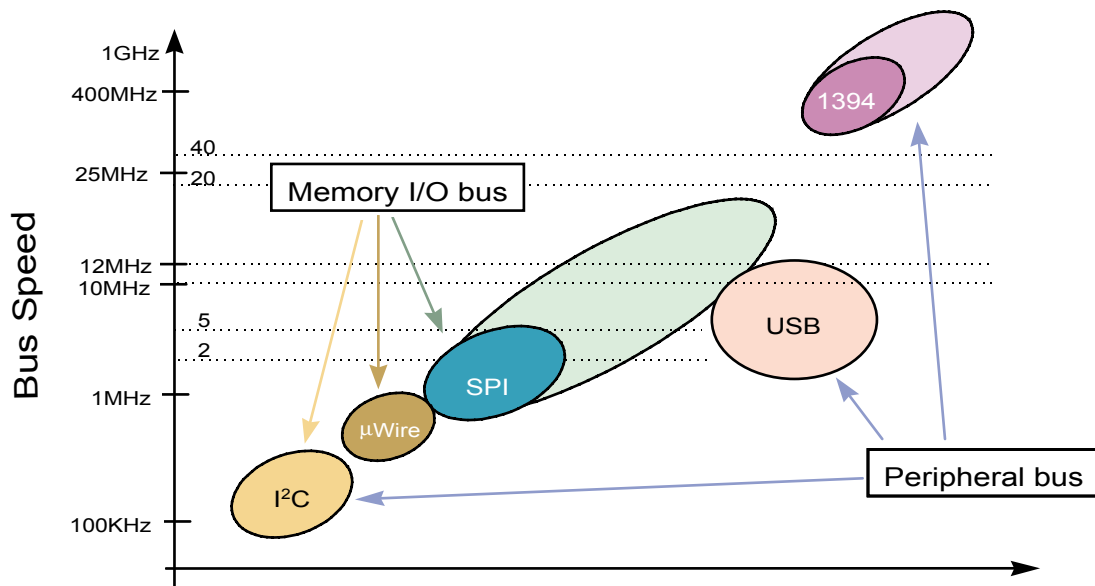


Figure 5. Serial Bus Speed Trends. There are two kinds of serial busses, peripheral and memory. The I²C bus is both. As the need for performance increases, the speed of peripheral and memory busses increase.

The Microwire Bus

As one of the oldest serial busses, the microwire bus has been a popular and high volume solution for a long time. It supports some of the lowest cost serial memories and is fast enough for many applications. However, there are a few drawbacks.

1. The microwire interface does not support as many density options as the other interfaces, with densities topping out at 16kbits. There also have been few new products introduced with this bus. This limits the designer's choices in new designs.
2. The microwire requires more port pins than the I²C bus, so it's use can be more costly to the system designer.
3. The microwire bus is limited in speed and architecture. Microwire devices limit the interface speed to 1MHz and clock data out and in on the same edge. This imposes some limitations on the design. A similar bus (SPI) clocks data in and out on opposite edges of the clock and has higher speed specifications.

So while microwire has some advantages, it will likely be one of the first of today's serial busses to disappear.

The I²C Bus

The I²C bus, developed at about the same time as the microwire bus, will likely be around for some time. Since it was developed as a peripheral bus, it works very well in systems that have few ports available on the host controller and must connect to a number of peripheral devices. Some of the more common I²C devices are A/D converters, LCD displays, Real Time Clocks and memories.

In 1985 Xicor introduced a 2-wire serial memory device that could operate on the I²C bus. Xicor still leads the industry in 2-wire density, interface speed and features and has one of the widest selections of 2-wire memories with densities ranging from 128 bits to 128kbits.

The original I²C bus specified 100kHz maximum speed. Based on industry feedback, this increased to 400kHz, but likely will not increase it further. The I²C bus speed limit results from it's peripheral roots. Potentially long I²C bus lengths and the activity of a number of devices on the bus (including multiple masters) increase the

possibility of noise induced errors as the speed increases. This, coupled with the indeterminate loading and collision detection protocols, makes I²C a noise sensitive interface.

The I²C interface consists of two lines clock (SCL) and data (SDA). The protocol specifies that communication begins with a start bit and ends with a stop bit. Data going HIGH to LOW while the clock is HIGH defines a start bit (Figure 6) Data going LOW to HIGH while the clock is HIGH defines a stop bit. During transmission of data the SDA line cannot change while the clock is high. This protocol makes it difficult to discriminate between a start/stop bit and data on a higher speed, noisy bus.

Noise on the I²C bus can sometimes result in corrupted data. In the I²C protocol, a random read of the array (Figure 7) consists of two parts. First, the host writes the address where the desired read will start. Then the host sends a repeated start bit followed by a current address read instruction. The host then clocks data in from the SDA line. Since the current address read instruction differs from a write instruction by a single bit, noise at a critical time turns a read operation into a write and clocks intended to read data into the host become clocks writing data into the device. The result can be uncontrollable data corruption. This doesn't happen often, but when it does it can be serious.

To limit the probability of this type of error, Xicor introduced 2-wire devices that provided input noise filtering, schmidt triggers and input latching. To give engineers additional control over memory write operations, all new 2-wire devices include Block Lock features. This gives the system designer the ability to lock critical parts of the array, so unexpected and uncontrolled writes due to noise cannot damage critical system data.

In order to allow a number of peripherals to connect to the same set of two wires, the I²C bus requires an open collector with pull-up on the SDA output of any device on the bus. For multi-master systems, the SCL line also needs an open collector and a pull-up. This configuration allows one device to "win" over another when two devices try to send data at the same time (collision).

However, this configuration increases system power, since there is $270\mu\text{A}$ ¹ current flowing through the pull-up resistor for the duration of a “0” bit .

In summary, the I²C bus has the advantages of needing few port pins and supporting a number of peripheral devices. However, the I²C bus has the disadvantages of slow speed, noise susceptibility and higher system current.

The SPI Bus

Motorola created the SPI port in the mid 1980’s for use with their 68HC11 and 68HC05 product families. The SPI port shares similarities with the microwire port, using similar signal names and command protocols. SPI clocks data in and out differently from the microwire and can be clocked at a much higher rate.

Xicor was the first company to introduce an SPI serial memory (the X25C02 in 1991) and led the industry in developing higher density devices. Xicor also pioneered the use of Block Lock™ mechanisms on SPI devices to improve data integrity. In the last few years microcontrollers, DSPs, RISC processors and ASICs all feature built-in SPI ports. These developments prompted Dataquest in 1996 to report SPI as the fastest growing serial interface. Today, as processors increase in performance, the capabilities of the SPI port increase. To keep up with this trend, Xicor recently introduced SPI memories with a 5MHz clock rate and plans future speed increases.

The SPI interface consists of four control lines: $\overline{\text{CS}}$, SCK, SO, and SI. The host controls the $\overline{\text{CS}}$ line to select the serial memory, then uses the clock and data lines to transfer data back and forth. Current implementations of the SPI interface are “half duplex”, meaning that data does not go out on the SO line while writing data on the SI line. So it is possible to connect SI and SO to get a three wire interface. Most SPI implementations can work in a “full duplex” mode. That means that data clocks into the device at the same time data clocks out. This can help increase the data throughput. Future SPI

memories will likely take advantage of this full duplex capability.

The SPI interface is command driven (Figure 8.) To read data from the memory, the host selects the device, sends a READ command, sends the address of the desired data, then clocks the data out of the device through the SO pin. At the end of the transaction, the host deselects the device. There are commands to READ and WRITE the array, read (RDSR) and write (WRSR) to a status register, and set (WREN) and clear (WRDI) a write enable latch.

Before any successful nonvolatile write to an SPI device, the host must select the device, send a WREN command, deselect the device, select the device again, send a WRITE command, send the 16 bits of address (older low density devices had 8 address bits), send a multiple of 8 bits and deselect the device. Any violation of this sequence terminates a write operation. Also, the completion of a nonvolatile write resets the write enable latch, forcing the entire sequence to be repeated for the next write operation. This makes SPI memories insensitive to noise. Block Locking further decreases the probability of inadvertent writes due both to noise and programming glitches.

SPI is a high speed memory port supported by a variety of microcontrollers, DSPs and ASICs. It is growing in popularity as host performance increases and as built in SPI ports become more common. New features as well as higher speeds indicate that the SPI interface will be around for quite a while.

1. Assumes 100pF bus capacitance and 2.7V operation.

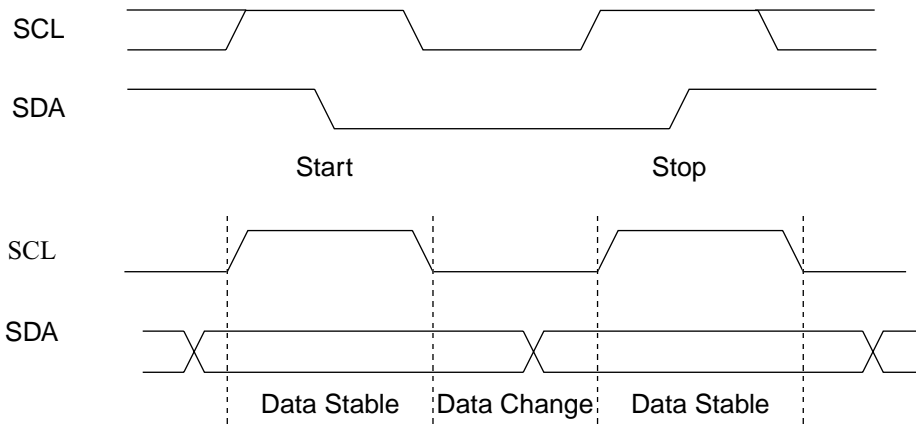


Figure 6. I²C Start, Stop and Data bits

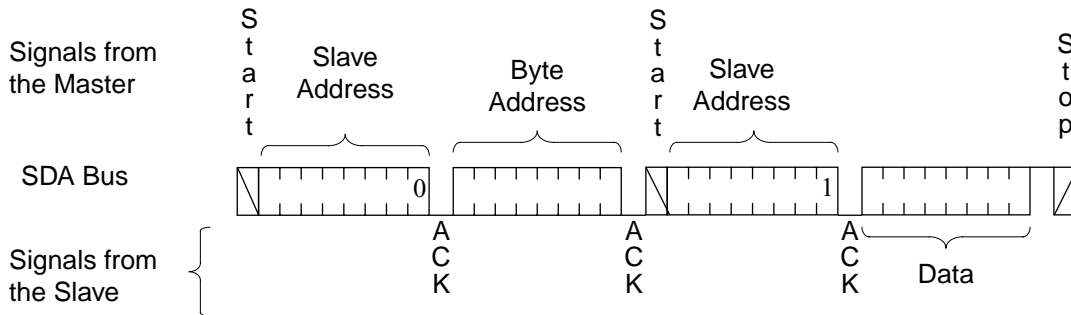


Figure 7. I²C Random Read - This figure shows a random read of one byte.

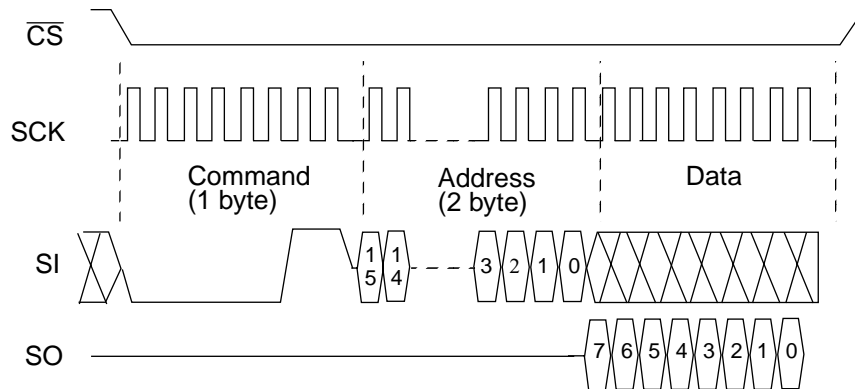


Figure 8. SPI Read Command



Conclusion

Industry-wide, there is a trend to serial interfaces and away from parallel, as demonstrated by several new industry standards. The need for low power, small size and lower cost fuels this conversion and there is no sign that the trend will change in the near future.

In order to provide a communication medium to the serial memory, there are several types of interfaces. The discussion here focused on 2-wire and SPI solutions. The value of each lies in the design trade-off between

how they connect to the host and how fast each transfers data.

In recent years, consumers have demanded more portability, performance and customization from the products they buy. These needs fueled decreases in package size and power and increases in memory density and speed. With state of the art speed and packaging, Xicor 2-wire and SPI serial memories provide designers and consumers with the right solutions for today and for the future.