An I/O System on a Chip

The heart of the I/O subsystem for the HP 9000 Model 712 workstation is a custom VLSI chip that is optimized to minimize the manufacturing cost of the system while maintaining functional compatibility and comparable performance with existing members of the Series 700 family.

by Thomas V. Spencer, Frank J. Lettang, Curtis R. McAllister, Anthony L. Riccio, Joseph F. Orth, and Brian K. Arnold

The HP 9000 Model 712 design is based on three custom pieces of VLSI that provide much of the system's functionality: CPU, graphics, and I/O. These chips communicate via a high-performance local bus referred to as GSC (general system connect). This paper will focus primarily on the I/O chip.

A major goal of the Model 712 I/O subsystem was to provide a superset of the I/O performance and functionality available from other family members at a significantly reduced manufacturing cost. This goal was bounded by the reality of a finite amount of engineering resources, and it was obvious from the start that integrating several of the I/O functions onto a single piece of silicon could greatly reduce the total I/O subsystem manufacturing cost. Each function of the I/O subsystem was examined individually as a candidate for integration. The value of maintaining exact driver-level software compatibility was also evaluated with respect to the advantages of minimizing the hardware cost for each of the I/O functions.

The investigation indicated that the optimal solution for the Model 712 was an I/O subsystem that centered around a single piece of custom VLSI. The chip that resulted from this investigation directly implements many of the required I/O functions and provides a glueless interface between the GSC bus and other common industry I/O devices. This chip was named LASI, which is an acronym that refers to the two major pieces of functionality in the chip, LAN and SCSI. The LASI chip also provides several miscellaneous system functions that further reduce the amount of discrete logic required in the system.

Chip Overview

The LASI chip was designed in a 0.8-μm CMOS process and is 13.2 mm by 12.0 mm in size (including I/O pads). It contains 520,000 FETs and is packaged in a 240-pin MQUAD package. LASI dissipates approximately three watts when operating at the maximum GSC frequency (40 MHz). LASI was designed primarily using standard-cell design methodologies although several areas required full custom design.

A functional block diagram of LASI is shown in Fig. 1. The majority of circuitry in LASI is consumed by only two functions, LAN and SCSI. Both of theses designs were purchased from outside companies and ported to HP's design process. The SCSI functionality is exactly identical to the NCR 53C710 SCSI controller, and the LAN functionality is exactly identical to an Intel 82C596 LAN controller.

Other I/O functionality that is completely implemented on LASI with HP internal designs includes: RS-232, Centronics parallel interface, a battery-backed real-time clock, and two PS/2-style keyboard and mouse ports. In addition, LASI provides a very simple way of connecting the WD37C65C flexible disk controller chip to the GSC bus. The system boot ROMs are also directly controlled by the LASI chip. The Model 712 provides 16-bit CD-quality audio and optionally supports two telephone lines. LASI provides the GSC interface and clock generation (using digital phase-locked loops) for both of these audio functions. Fig. 2 shows an approximate floor plan of the LASI chip. The floor plan shows the general layout and relative size of each block.

LASI contains several system functions that help to minimize the miscellaneous logic required in the system. This includes GSC arbitration and reset control. LASI also serves as the GSC interrupt controller.

It is possible to use up to four LASI chips on the same GSC bus. LASI can be programmed at reset to reside in one of four different address locations. The arbitration circuit supports chaining, and LASI can be programmed to either drive or receive reset.

System Support Blocks

The following sections give a brief overview of each of LASI's major functional blocks that provide system support functionality in the Model 712, but do not directly support or implement any I/O function.

GSC Interface. The GSC (general system connect) bus connects the major VLSI components in the Model 712. It is a 32-bit bus with multiplexed address and data. The bus consists of 47 signals for devices capable of being a bus master. The GSC bus is defined to run at up to 40 MHz giving a peak transfer rate of 160 Mbytes/s.

The GSC interface block in LASI provides the connectivity between the GSC bus and the wide variety of internal bus blocks, many of which have different logical and timing requirements. This block converts the GSC bus to a less complex internal LASI bus. The LASI internal bus is very similar to the GSC bus, but it is not as heavily multiplexed and is more flexible than the GSC bus in that it easily accommodates the simpler interface for the general-purpose I/O blocks in LASI. The GSC interface block handles bus errors and keeps track of parity information for other internal

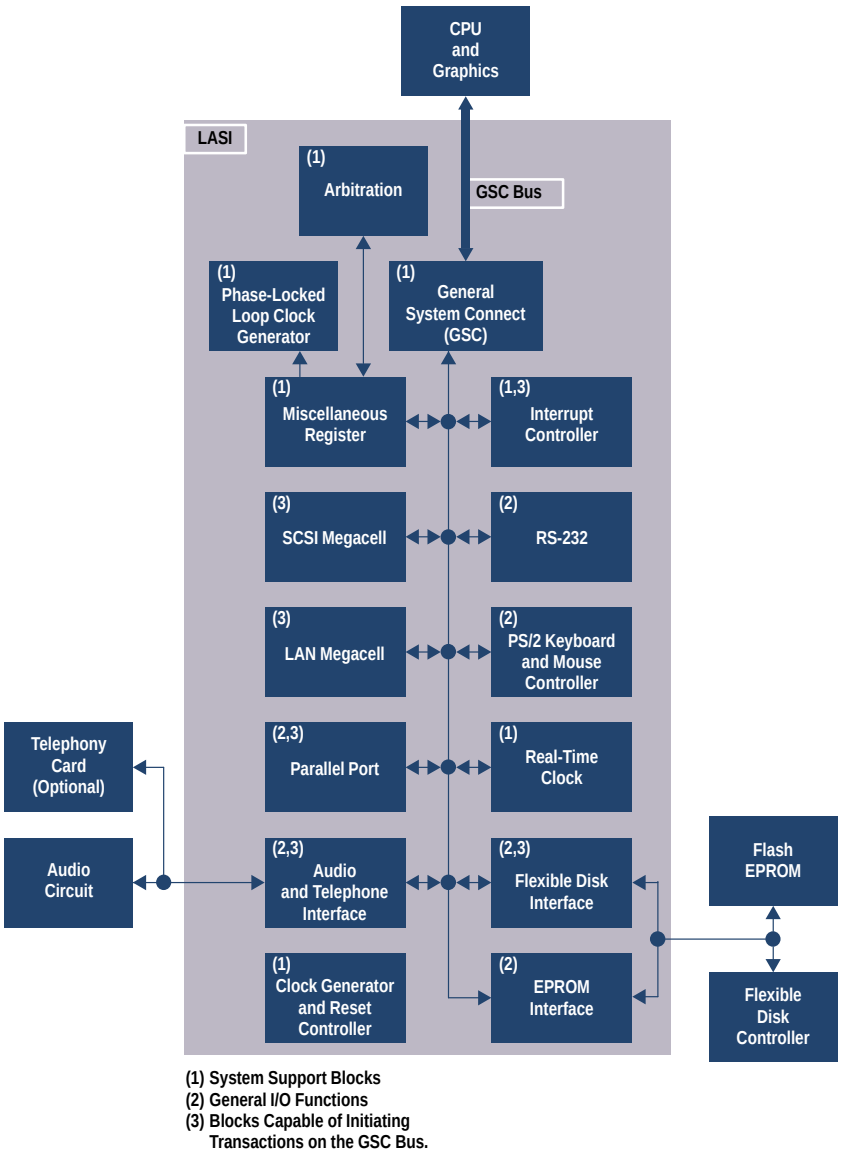

Fig. 1. LASI chip block diagram.

blocks, removing the associated complexity from these controllers. Both master and slave devices reside on the LASI internal bus.

LASI is a slave whenever the CPU initiates data transfer. As a slave, LASI supports only subword and word write, and subword, word, and double-word reads.* Internal slave devices only need to support a subset of these transactions. There are five different protocol behaviors for slave devices in LASI: unpaced byte wide, paced byte wide, packed byte wide, unpaced word wide, and paced word wide.

Unpaced devices, such as the real-time clock, don't use a handshake with the GSC interface, making their protocol very simple. When a device requires a variable length of time to transfer data it is called paced. The SCSI interface is an example of a paced device. A packed device is one that sends a sequence of bytes to make up a word or double word. The boot ROM interface is an example of a packed device.

* In PA-RISC a subword is typically one byte, a word is 32 bits, a double word is 64 bits, and a quad word is 128 bits.

A simple strobe signal is asserted while internal data and address buses are valid. Internal devices have no direct interaction with bus errors.

As a bus master, LASI is capable of initiating subword, word, double-word, and quad-word transactions on the GSC bus. Once one of LASI's internal bus masters owns the bus, it can signify the start of a transaction by asserting the master_valid signal (see Fig. 3). The device must then simultaneously drive its DMA address (master_address), transaction type, and byte enables onto the bus. On a read, the first available data word will appear on the internal bus when the master_acknowledge signal is asserted by the GSC interface. The GSC interface will not accept another master_valid until all the read data has been transferred.

If a timeout error, address parity error, or data parity error is encountered on the GSC bus, the GSC interface will always do a normal handshake for the transaction by asserting the master_acknowledge signal. The transaction will complete as usual except that an error is logged, disabling arbitration for the device so it cannot be a bus master again. This means

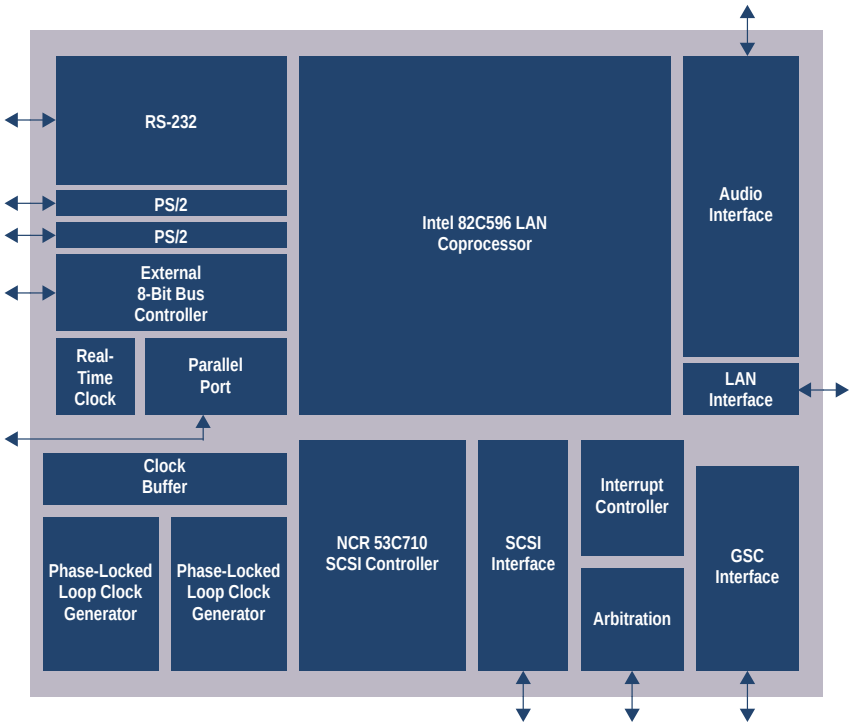

Fig. 2. LASI floorplan.

that internal masters, at the hardware level, never need to respond directly to bus errors. When the GSC interface block sees a timeout error it will, from the perspective of its internal bus blocks, complete a transaction normally. In this way the GSC's error signaling mechanism can correctly terminate an errant transaction without adding complexity to LASI's internal blocks.

Parity is generated in the GSC interface whenever LASI sources data or an address on the bus. Parity is checked whenever LASI is a data sink. LASI does not respond to address parity errors on the GSC bus, which result in a timeout error.

Arbitration. LASI contains six different blocks capable of initiating a transaction on the GSC bus (see Fig. 1). To initiate a transaction, a block must first own (or gain control of) the GSC bus. Deciding which potential master owns the bus is the job of LASI's arbitration block. The arbitration circuit in LASI provides internal bus arbitration for all six internal devices and provides external GSC arbitration signals for the CPU and an expansion slot. This capability allows LASI to function as the central arbiter for the GSC bus in low-end systems. The arbitration circuit can also be pin-programmed at reset to behave as a secondary arbitration device that is controlled by another arbiter. This feature allows LASI to be used in larger systems that provide their own arbitration circuit. A second LASI can also be used for I/O expansion in low-end systems in which the first LASI is providing the central arbitration. Support for multiple LASI's on the same GSC bus makes the speedy development of multifunction I/O expansion boards a relatively simple task.

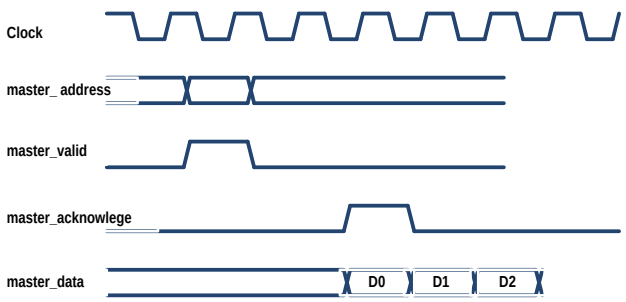

Fig. 3. Master read timing diagram.

The LASI design was simplified by requiring that the LASI arbitration circuit gain control of the GSC bus before granting the internal bus to potential bus masters. This saved a significant amount of complexity in the GSC interface block as well as greatly reducing the number of cases that needed to be tested during the verification effort. This simplification does create a couple of wasted GSC cycles for each transaction initiated by LASI. However, this inefficiency has a negligible impact on system performance.

The LASI arbitration circuit provides a simple round-robin scheme that provides roughly equal access to all devices. The arbitration circuitry keeps track of the identity of the last device granted the bus and all currently outstanding requests. (A simple truth table makes sure the GSC resource is handed out fairly.) If no devices are requesting the bus, LASI will default to granting the bus to the CPU. This has a small positive impact on performance, given that the CPU is the most likely device to initiate the next transaction. This arbitration scheme helps simplify the arbitration circuit by not requiring it to monitor bus activity. Each bus master is responsible for being "well-behaved" with respect to bus use.

The arbitration circuit plays a key role in the error handling strategy for LASI. If an error occurs on the GSC bus while LASI is the bus master, the arbitration circuit will not grant the bus to additional internal devices until the CPU clears the error by clearing a bit in the arbitration circuit. This simplifies the design of other devices within LASI by not requiring

them to use the error signal as an input to their state machines. When an error is detected, the transaction will terminate normally, but no additional transactions will be allowed until the situation is rectified by software.

Interrupt Controller. A total of 13 different interrupt sources exist on the LASI chip. Each interrupt source drives a single signal to the interrupt controller block. When the interrupt signal is asserted, the interrupt controller block will master the bus and issue a word write to the I/O external interrupt register (IO_EIR), which is physically located in the CPU. The data transferred to the IO_EIR contains a value that indicates the source of the interrupt. The address of the IO_EIR and the interrupt source value can be programmed by writing to the interrupt address register located in LASI's interrupt controller block. Individual interrupt sources can be masked by setting bits in the interrupt mask register.

LASI's interrupt controller is designed to provide a variety of interrupt approaches. The Model 712 uses only one of these alternatives. Asserting an interrupt causes a write to the IO_EIR to be mastered on the GSC bus. Upon receiving an interrupt from LASI (via IO_EIR), the CPU will read the interrupt request register located in LASI's interrupt controller block. One bit in the interrupt request register is designated for each potential interrupt source in LASI. The interrupt request register is cleared automatically after it is read by the CPU.

Real-Time Clock. The Model 712 needs to keep track of time when the system power is off. To this end, LASI provides a battery-backed real-time clock. The real-time clock is logically very simple and consists of a custom oscillator circuit and a 32-bit counter that can be read and written to by software. The 32-bit counter is used to keep track of the number of seconds that have elapsed from some reference time.

The oscillator unit operates at 32.768 kHz and typically uses less than 10 μA of current when operating on battery backup. It uses a minimum of external circuitry (consisting of two capacitors, a crystal, and a resistor) to accomplish its task.

Inside the LASI real-time clock, the 32-kHz signal is reduced to a 1-Hz signal by a 15-bit precounter. The 1-Hz signal is then used to increment the main 32-bit counter. Both the counter and the precounter are implemented using simple ripple counters. The 15-bit precounter is always cleared when software writes to the 32-bit counter.

Phase-Locked Loop Clock Generators. The goal for the LASI clock subsystem was to generate all the I/O subsystem clocks from one crystal oscillator over a wide range of system frequencies. The LASI clock block generates five different clock frequencies required for the wide variety of I/O interfaces. Three of these clocks are subharmonics of the processor clock, and are generated using simple digital state machines. However, the 40-MHz clock and the audio sample clock are fixed-frequency clocks. The 40-MHz clock is used for the SCSI back end and RS-232 baud rate generator, and the audio sample clock is used for the external CODEC chip. The frequency of this clock (16.9344 MHz to 24.576 MHz) is selectable on the fly by the audio and telephone interface.

Two digital phase-locked loop circuits are provided in LASI to generate the two fixed-frequency clocks from the CPU

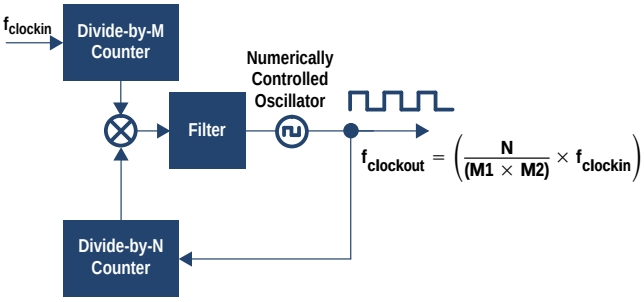

Fig. 4. Phase-locked loop clock controllers.

clock. These digital phase-locked loops implement the equation: $f_{clockout} = (f_{clockin} \times N)/(M1 \times M2)$, where N, M1, and M2 are digital coefficients stored in LASI control registers. $f_{clockin}$ comes from the main system reference clock. The $f_{clockout}$ from one of the phase-locked loops is used for the audio clock, and the $f_{clockout}$ from the other phase-locked loop is used for SCSI, RS-232, and other I/O functions. At power-on, the processor initialization code (stored in the flash EPROMs) loads the coefficients corresponding to the processor clock for the particular product. The audio sample clock has two sets of coefficient control registers, which are selected by a multiplexer based on a signal from the audio interface. Fig. 4 shows one of the phase-locked loop circuits.

The phase-locked loop circuits are completely digitally controlled, including a digitally controlled oscillator, digital phase detector, counters, and scan test hardware. This design eliminates analog control voltages which are susceptible to noise and integration errors. The digitally controlled oscillator is a ring oscillator with a digitally programmable delay element. This design is capable of generating frequencies of up to 135 MHz. A combination of custom and standard-cell design techniques are used in this design. Each phase-locked loop cell measures 1500 μm by 890 μm.

General I/O Functions

The blocks shown in Fig. 1 that make up the general I/O functions include the parallel port, audio and telephone interface, RS-232 port, and flexible disk and boot ROM interface. These I/O functions originate from HP internal standard-cell designs that were originally designed using Verilog RTL models and then synthesized into a standard-cell design using Synopsys. Some blocks were designed specifically for LASI while others were leveraged from previous HP ASIC designs.

Parallel Port. The parallel port is designed to be software compatible with previous generations of HP 9000 Series 700 I/O subsystems while minimizing overall complexity and chip area. This port allows interfacing to printers and other peripherals supporting the industry-standard Centronics parallel interface. The parallel port signals are driven directly from the LASI chip without additional buffering.

DMA was supported on previous workstation controllers and therefore needed to be provided on LASI's controller. However, since no central DMA controller exists, all DMA hardware is contained within the parallel I/O block. Since parallel port bandwidth requirements are fairly modest

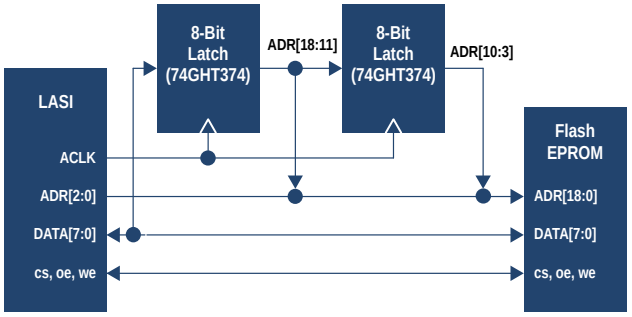

Fig. 5. Address latching logic, and the data and control lines associated with the external 8-bit bus.

(about 400 kbytes/s), DMA is done by reading one 32-bit word of data, releasing the bus, transferring one to four bytes of data over the interface, and then requesting the bus again. This approach keeps the DMA controller quite simple while easily accommodating byte unpacking.

Keyboard and Mouse Controller. LASI provides support for two IBM PS/2-style keyboard and mouse devices, making the keyboard and mouse ports just like those used on a standard IBM personal computer. These interfaces are new to the Series 700 family so there were no software compatibility issues, allowing us to optimize the design for low manufacturing cost. The interface provides only a minimal amount of hardware and relies on the driver to do most of the work. The interface also performs the serial-to-parallel and parallel-to-serial conversion and does a small amount of buffering. An interrupt is generated for every byte of data received from the PS/2 device. The software overhead is not a performance issue because of the extremely low data rate of the interface.

Flexible Disk and Boot ROM Interface. LASI supports an external 8-bit bus that provides the capability to connect discrete flash EPROM devices and a flexible disk controller with very little additional logic. Fig. 5 shows a simple schematic of a flash EPROM and the required address latching logic on the 8-bit bus. It was not cost-effective to integrate these devices into the LASI chip. The 8-bit bus is also capable of supporting other types of 8-bit devices, giving some degree of flexibility to the I/O system.

The 8-bit bus supports 1M bytes of address space (the first half of the LASI address space). All transactions to this address space on the 8-bit bus begin with two address cycles. These cycles transfer bits 18:3 of the address to two 74GHT374-type 8-bit latches wired in series and controlled by LASI. Multiplexing the address on the data lines saves 15 pins on LASI.

LASI is capable of supporting byte, word, and double-word reads and byte writes to devices on the 8-bit bus. Word and double-word reads are accomplished by doing multiple accesses to devices on the 8-bit bus and packing the bytes into words before returning them on the GSC bus. Word and double-word accesses require the address to be latched only once since LASI drives the lower three address bits directly. This greatly reduces the word and double-word access time. Double-word reads take approximately 75 GSC cycles to complete because eight accesses are required on the 8-bit bus. During each of the eight accesses a new address is presented to the flash EPROM which results in valid data being driven to the 8-bit bus by this flash device. Byte accesses are also relatively slow (12 GSC cycles) to support very slow devices on the 8-bit bus. It is important to note that the 8-bit bus is not electrically connected to the GSC.

LASI is designed specifically to support the WD37C65C flexible disk controller on the 8-bit bus. The Model 712 uses a personal computer style flexible disk controller instead of a SCSI-based flexible disk controller because of the significantly lower cost of the drive mechanism. The flexible disk controller was not integrated into the LASI chip because of the low cost of the WD37C65C chip and the potential for SCSI drives to come down in cost in the future. The WD37C65C shares the data bus and two control lines with other devices on the 8-bit bus, but does not consume any of the 1M bytes of allocated address space. Supporting the WD37C65C requires six dedicated signals and no external glue logic. LASI supports the WD37C65C running in DMA mode and provides the capability to move data directly between main memory and the WD37C65C without processor intervention.

RS-232. The RS-232 block in LASI is an HP internal standard-cell design that emulates the behavior of the National Semiconductor NS16550A. The Verilog HDL description for this design was leveraged from previous HP ASIC designs used in other members of the HP 9000 Series 700 workstation family.

One difference between this block and the NS16550A is that its baud clock is derived from a 40-MHz signal. This allows the block to share the phase-locked-loop-generated 40-MHz clock with the back end of the SCSI block and eliminates the need to support an external crystal or dedicated phase-locked loop for baud clock generation.

Audio Interface. The Model 712 supports built-in CD-quality audio and an optional telephony card.[1] The telephony card is DSP-based and provides simultaneous access to two telephone lines both capable of supporting voice, fax, or data modems. LASI provides the interface between the GSC bus and the audio and telephony circuitry.

An objective for the Model 712 audio subsystem was to maintain complete software compatibility with previous discrete designs. As a result, a good deal of the audio interface circuitry on LASI is dedicated to supporting this compatibility and is not optimized for minimal manufacturing cost.

The audio interface in LASI has two DMA channels that support the input and output audio streams. Each channel has two 4K-byte pages of main memory continually reserved for transferring data to and from the CS4215 CODEC. The buffering in the interface is sufficient to guarantee isochronous audio operation, given worst-case GSC bus latencies in the Model 712. A wide range of audio formats is supported including 8-bit or 16-bit words sampled in either linear, μ-law, or A-law format at a variety of sample rates from 8 kHz to 48 kHz.[1] The clock that determines the sample rate in the CODEC is generated in one of LASI's programmable phase-locked loop circuits. Communication between LASI and the CODEC is accomplished via a full-duplex, serial bit stream.

The high-speed serial bus over which LASI communicates with the daughter card is similar to a concentrated highway bus developed by AT&T but has several modifications. The core pinout is the same using the signals data transmit (DX), data receive (DR), and frame synchronization (FS), but the definition of the bus has been extended to incorporate control of external buffers and bus reset.

Communication with the telephony card is accomplished via two TTY channels internal to LASI. The serial concentrated highway bus data is multiplexed onto the high-speed serial stream and sent to the CODEC and the telephony card. Since TTY devices are used, the driver for the telephone system is a highly leveraged version of the existing TTY drivers. The audio interface and HP Teleshare[2] have a common digital interface which resides in LASI. HP Teleshare is described in more detail in the article on page 69.

Megacell I/O Functions

LASI contains two megacells whose designs were purchased by HP from external vendors. The decision to do this was based on maintaining software compatibility with past HP 9000 Series 700 workstations and the availability of engineering resources in HP. In both cases, an important goal was to maintain the integrity of the megacell as much as possible. A definite boundary was drawn between functionality leveraged from external vendors and new design work. This boundary proved vital to functional verification and production testing.

LAN Megacell. IEEE 802.3 LAN support is provided by a megacell derived from the Intel 82C596 LAN coprocessor. To understand the integration, two key areas should be considered. First, importing the megacell at the artwork level solved some problems and imposed others. Second, in the area of interfacing, the integrated megacell eliminated a substantial number of chip pins but raised some protocol issues that had to be overcome.

The LAN megacell was imported into our IC design flow at the artwork level. Because the original Intel design was done in a custom fashion, a netlist translation would have required a significantly longer design time and a much larger manpower deployment than the artwork translation. Even at the artwork level, several modifications were made because of differences between the original CMOS process design rules and those of our target process.

One challenge in importing the megacell at the artwork level was developing a verification strategy that allowed concurrent simulation of the megacell and the rest of the chip. Because the megacell vendor used proprietary simulators running in a mainframe environment, the vendor simulation models couldn't be used in our Verilog-based environment. Hardware modeling was explored, but characteristics of the part made this solution impractical. Converting either functional representations or transistor-based representations to Verilog HDL raised too many concerns about modeling accuracy. In view of these roadblocks, an unconventional approach to simulation modeling was employed. First, FET-level model was extracted from the artwork. This model was turned on and verified using Intel's production test vectors and a proprietary in-house simulator. Second, the in-house simulator was compiled and linked into the Verilog simulator using a procedural-level interface. Third, a Verilog HDL interface module was written that defined synchronization events for data transfer between the two simulators, and the model was reverified using production vectors. Finally, tests were run that were specifically designed to test the interface between the megacell and the internal bus.

Integrating the LAN megacell did provide a clear win by improving the ratio of I/O to core area. When sold as a separate device, the Intel 82C596 has 89 signal pins devoted to the host interface. Once the megacell was integrated, all of these signals remained on-chip. In addition, 77 of the removed signal pins had output drivers, so the associated power and ground pins were eliminated.

The megacell did require a small amount of circuitry to interface the 82C596 bus to the LASI internal bus. The primary difficulty in this area was burst transactions. The system bus wanted to know at the start of the transaction how many words were to be bursted. In contrast, the 82C596 burst protocol would only indicate whether or not it had one more word to burst. To minimize complexity and avoid the area associated with a FIFO buffer, the decision was made to support only two-word bursts. This logical intersection of the two bursting protocols provided a bandwidth utilization improvement over nonbursted transactions while minimizing chip area and development time.

SCSI Megacell. To provide SCSI-2 support, LASI uses an NCR 53C710 megacell. This megacell was imported into our design methodology as a netlist port. The design was translated from NCR's standard-cell library to HP's cell library. A few unique components were added to HP's library specifically to support the SCSI megacell. While this created some challenges, doing a schematic port allowed more flexibility to optimize the aspect ratio of the megacell for a more efficient floorplan. This technique also masked differences between NCR's process and HP's process. Verilog models for this schematic port were simulated in the conventional way.

The programming and SCSI bus model for the 53C710 megacell is completely compatible with the industry-standard component marketed by NCR. However, the host side interface of the megacell is modified to eliminate the pads and replace them with standard-cell components. These components connect directly to internal megacell signals, providing an interface to the chip's internal bus.

The 53C710 can be a master and a slave device on the GSC bus. LASI's internal bus protocol for slave transactions requires only combinational logic between the megacell and the internal bus. As a slave, byte and word transactions are supported in the megacell. If SCSI is a bus master, the interface supports all the transaction types needed by the megacell, with the help of a small state machine located in the SCSI interface block shown in Fig. 2. SCSI data is typically transferred using four-word read and write transactions on the GSC bus.

Test Support

The primary objective for LASI testing was to provide an extremely high level of coverage with a limited amount of test development resources. Test support was complicated by the diverse nature of the circuits on LASI.

The non-megacell functionality is tested by a combination of parallel pin vectors in conjunction with automatically generated scan vectors. LASI has an enhanced JTAG (IEEE 1149.1) test block, 25 distributed internal I/O device scan chains, and embedded test functionality in the I/O pads. The JTAG test block contains a test access port and boundary-scan architecture defined in IEEE standard 1149.1-1990 and private instructions used for clock control, full-chip step control, and specific scan-chain functions.

To maximize test coverage for the two megacells and to minimize the required test development resources, the vectors used for production testing by Intel and NCR are used on LASI. Doing this requires multiplexing all megacell signals to pads to create what looks to the chip tester like an Intel 82C596 or an NCR 53C710, depending on the test mode.

This technique provides important verification and test coverage, but complicated the design. Each output pad includes a three-input multiplexer, and each input pad drives signals to three destinations on-chip, significantly increasing the loading. The additional routing complexity required devoting more space for routing channels, and the larger pads reduced placement flexibility.

Conclusions

Integrating multiple I/O functionality onto a single VLSI chip can significantly reduce the cost of the I/O subsystem. However, many system dependent factors and each candidate functionality need to be examined carefully in the system context before deciding to integrate. Some important system considerations are software compatibility, the cost of discrete alternatives, the cost of printed circuit board area, customer connect rate, available IC fabrication capacity, available engineering development resource, and so on. The LASI chip definition is the result of a detailed investigation into optimizing an I/O system for HP's low-end workstations.

Acknowledgments

Many people who made significant contributions to the development of LASI did not have the opportunity to contribute directly to this paper. Some of those individuals include Tom Baker, Greg Burroughs, Kin Chan, Larry Chesler, Marc Clarke, Keith Erskine, Mercedes Gil, Jeff Hargis, Rob Horning, Peter Meier, Eileen Murray, Cheryl Ranson, Charlie Spillman, and Chuck Summers.

References

1. Monish S. Shah, "Overview of A-law and μ-law Data Formats," *Hewlett-Packard Journal,* Vol. 45, no. 2, April 1994, p. 65.
2. Gary P. Rose, et al, "Development of a Multimedia Product for HP Workstations," *Hewlett-Packard Journal,* Vol. 45, no. 2, April 1994, p. 9.