U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 5571

Computer
Systems
Laboratory

Operating Principles of MultiKron II Performance Instrumentation for MIMD Computers

Alan Mink

U. S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards and Technology
Gaithersburg, MD 20899

December 1994

CMRF

COMPUTER MEASUREMENT
RESEARCH FACILITY
FOR HIGH PERFORMANCE
PARALLEL COMPUTATION

Partially sponsored by the
Advanced Research Projects Agency

NISTIR 5571

Operating Principles of MultiKron II Performance Instrumentation for MIMD Computers

Alan Mink

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards and Technology
Computer Systems Laboratory
Gaithersburg, MD 20899

Partially sponsored by the
Advanced Research Projects Agency

December 1994

TABLE OF CONTENTS

Operating Principles of MultiKron II Performance Instrumentation for MIMD Computers

Alan Mink

The MultiKron II design is an enhanced version of our earlier MultiKron performance instrumentation chip. They are both 179 pin chips, and although pin compatibility is close they are not 100% compatible. The MultiKron II has a longer TRACE sample, 20 bytes vs 16, to allow larger user data and Timestamp fields. The 16 counters for resource utilization measurements are now writable, to allow for "virtual" use, and their programming interface has changed to allow more options. Although designed for a 64-bit processor bus interface, an internal holding register provides for 32-bit mode operation.

Key words: Computers; hardware support; MIMD; multiprocessor computers; performance characterization; VLSI.

INTRODUCTION

Based on our experience using the MultiKron [MIN92] performance measurement instrumentation and feedback from other researchers, we have designed and fabricated an enhanced version of the MultiKron called the MultiKron II. See Figure 1 for an overall block diagram of the MultiKron II. This evolution of performance measurement instrumentation is part of the parallel processor performance project at NIST [CAR88, CAR89, MIN90, ROB89]. The goal of the project is to characterize the performance of parallel computers as well as uniprocessors. The focus of our instrumentation work is to provide hardware support in obtaining performance measurement data with tolerable perturbation to both the processes executing and the architecture on which they are executing. The body of this report provides a brief background discussion about performance measurement and then goes on to describe the features of the MultiKron II and its processor interface. Appendix-A provides a summary of differences between the MultiKron II and the MultiKron. Appendix-B consists of a group of tables describing addresses and formats of the MultiKron II. Appendix-C provides tables of the MultiKron II pin assignments and descriptions.

BACKGROUND

Tracing events and counting are two basic concepts of performance measurement. The concept of tracing events is to follow different execution threads and to know when various events (the execution of code) occur and to be able to correlate these times. The concept of counting provides the basic mechanism for clocks, stop watches, histograms, etc., and can also tally events at high frequency rates--a must for high speed hardware events. The exploration of instrumentation to provide these basic concepts in various ways while under the constraints of low perturbation, low cost, and similar technology has been the focus of our work. Low perturbation refers to disturbing what you are measuring, also known as the probe affect. If one disturbs the measured program by adding too much extra code, the measured performance could be tangibly different from that of the uninstrumented version of the program. Low cost refers to the size as well as the production cost of the measurement instrumentation. Integrated circuits (chips) are both small and inexpensive, in mass production, when compared to printed circuit boards. Similar technology refers to the circuit technology used in the measurement instrumentation and the architecture being measured. If you need faster circuit technology in your measurement instrumentation than in your measured architecture, then there would be no such technology to use when your computer architecture is already using the fastest circuit technology! The following discussion is centered around how MultiKron II provides these measurement concepts.

Event Tracing

The fundamental measurement function in event tracing is to determine the time of occurrence of various key events in the execution of a computer's tasks with high precision. These times can be used later to calculate elapsed time, duty cycle, access latency, and the like. The measurement data necessary to specify a trace event we call a Trace sample. Passive monitoring of instruction addresses fetched externally to determine event occurrence is insufficient, since internal caches mask the actual instructions executed and in addition external addresses are *physical* addresses while compilers produce code in terms of *virtual* addresses. Due to these problems, an internal view of program activity is necessary to determine when an event occurs so that measurement data for a Trace sample may be collected. For minimal perturbation of the executing program a *hybrid* approach to event tracing is employed in which a small amount of *embedded code* in the software triggers hardware capture of the Trace sample for each event. By designing the data capture hardware as a memory-mapped device, this embedded code could be kept to as little as a single memory write per event. Execution of this embedded code informs the measurement hardware of the occurrence of an event and causes it to capture the **user specified data** of the write as part of the Trace sample. This user data, which is stored as part of the Trace sample, provides the means to communicate information about the internal program view. The user data will include event identification, which identifies *where* in program execution the event occurred, along with any information necessary to qualify the event. It does not identify which process may be using the (possibly-shared) code where the event occurred. Event identification, as an example, can be specified as a single number which may only have meaning to the experimenter, such as "27" which could indicate a location in the code where a message was just received. Qualifying information could be the size or source of that message.

Current time is important data to collect about an *event*. Thus a **Timestamp Counter** is needed with a time resolution on the same order as the instruction execution rate and enough precision to avoid counter wrap-around during any experiment. To allow correlation of samples taken by multiple MultiKron II chips, the Timestamp counters should all be synchronized. One synchronization method uses a system-wide common Timestamp clock and reset signal (which resets the Timestamp counter to zero) for all MultiKron II chips.

Many modern computers are composed of a number of nodes. The coupling between nodes may be fairly loose, but each node may consist of a number of processors which are tightly-coupled together. A single measurement chip can efficiently serve all the tightly-coupled processors at a node. The Trace sample must include the identity of the node, the processor (if a node contains more than one processor), and the process. The MultiKron II provides for a wire per processor, up to eight processors in a tightly-coupled node, to identify the processor requesting the sample. A **Source Address register** is provided which contains the node and process identity packed together and should be updated by the operating system at each context switch. Since a single measurement chip may be used to measure a number of tightly-coupled processors, each working on a different process, process identification must be kept for each processor. This requires a set of Source Address registers; each processor is assigned its own Source Address register, in which node and process identification is kept.

A **Trace sample** consists of the concatenation of the user-written data, the associated Source Address register (for node and process id), the identity of the processor within the node, the Timestamp counter, and the sample type.

Because one may wish to limit the type or amount of events on which Trace samples are actually collected without having to modify the embedded triggering code, a separate filtering facility is provided. This filtering facility allows an experimenter to separate each embedded triggering code into one of sixteen groups. A **Filter Register** on the MultiKron II contains a bit position for each group. If that bit is enabled, the MultiKron II will collect a sample when a trigger occurs for that group. If that bit is disabled, the MultiKron II will not collect a sample when a trigger occurs for that group. The execution time for the embedded triggering code as seen by the processor is the same whether or not the sample is taken. Once instrumented, embedded triggering code can remain undisturbed when different types of measurement are desired (or none at all). Thus, software can be extensively instrumented and yet not burden the data-capture system in routine operation. Neither modification nor recompilation of the software on the measured computer is required to change the measurement set. The simple method we use to indicate trigger subset is to assign a block of addresses for triggering the taking of a data sample. The lowest four bits of the address indicates the group to which the trigger belongs.

Counters

Trace events occur relatively infrequently, typically hundreds or thousands of program instructions apart. Other types of events occur much more frequently, even on every processor clock cycle for short periods. The hybrid approach recommended for event trace support is unsuitable for measuring these very frequent events; they demand a fully-hardware measurement technique. As a compromise between resolution, cost, and data storage requirements, we have chosen to accumulate *counts* as a measure for these events. This amounts to a form of preprocessing of the measurement data, with a great reduction in the cost and data storage requirements. Of course some detail is lost.

Resource Counters can be used for a wide range of measurement if flexible input switching is provided on a per-counter basis. It must be possible to increment each by hardware signals such as cache hits or misses, clock cycles during waits for shared-resource access, or events such as message transmissions. A stop-watch capability can be realized by counting clock cycles between event. If the counters can also be incremented by software, they can be used to keep running tallies of items added to, or removed from, software-managed queues or buffers, average frequency-of-use of code, or other frequent software events. The content of all Resource Counters is optionally concatenated with a Trace sample to form a Resource sample. Resource samples should be collected at key points in program execution to resolve resource utilization to specific activities and to conserve Collection Network bandwidth and storage.

It is expected that all Resource Counters will be disabled prior to a measurement experiment. Then the counting source for each counter will be selected via the MultiKron II Mode Register. During the measurement period, each counter can be selectively enabled and disabled as many times as desired, and can thus accumulate counts during multiple uses of a piece of code, for example.

PROCESSOR INTERFACE

The data path between a processing node and the MultiKron II is 64 bits wide. On a read by a processor, any unused bit positions return a logical "1". On a write from a processor, any unused bit positions are ignored. We divide the physical address issued by a processor into three fields, the byte field, the MultiKron field, and the device field. The byte field is the least-significant two bits of the address, which specifies a byte within a 4 byte word. The MultiKron field is the next seven address bits which is used directly by the MultiKron II. The device field consists of remaining higher order address bits which must be externally decoded to establish the base address of the MultiKron II. The address mapping for the 7 bit MultiKron II address field is shown in Table B4 of APPENDIX- B. The four low order address bits of the MultiKron field for addresses 64-95 are used to specify which of the 16 Resource Counters are being accessed. The four low order address bits of the MultiKron field for addresses 96-127 are used to specify which of the 16 filter levels are being used.

A MultiKron II processor interaction is initiated by asserting (LOW) either the READB or WRITEB signal. See Tables C1 through C3 of Appendix C for pin assignments and descriptions. The processor interface timing is illustrated in Figure 2. The device field address decoder output must be combined with the processor's READ or WRITE signal to produce the corresponding READB or WRITEB and the STARTB signals. After the MultiKron has completed its action, it will assert (LOW) the ACKB signal for the MultiKron II. The READB or WRITEB signals to the MultiKron II must be de-asserted before the MultiKron II asserts its ACKB signal. The STARTB signal to the MultiKron II is intended to augment the READB and WRITEB signals, so that the READB or WRITEB can be placed and left at their asserted logic level for longer than the MultiKron requires and the STARTB can be asseted and de-asserted rather than READB or WRITEB. If STARTB is always asseted, it has no effect -- READB or WRITEB will control the MultiKron II processor interaction. If STARTB is always de-asserted, it will inhibit any MultiKron II processor interaction. Normally the ACKB signal will be asseted for one Node clock cycle. When ACKB is asseted any processor write is completed. When ACKB is de-asserted the output data for any processor read is removed. If the processor requires more time to acquire the output data for a processor read, it should assert (LOW) the HOLDB signal any time before the ACKB signal is asserted. The data and the ACKB will remain active until the HOLDB signal is de-asserted. Actual transfers are at positive-going transitions of the Node clock; A(ddress), D(ata), STARTB, HOLDB, READB, WRITEB, and ACKB signals must have adequate setup and hold times relative to these transitions.

The MultiKron II has two clock inputs, the Node clock and the Timestamp clock. The Node clock is the clock signal used internally by the MultiKron II for all synchronization. The MultiKron II will not function without a Node clock signal. The Timestamp clock is a slower clock (and should not exceed 1/3 of the Node clock frequency) and is used to increment the Timestamp counter and optionally any of the Resource Counters. The MultiKron II design targetted a Node clock frequency of 50 MHz and a Timestamp frequency of 10 MHz. Lower Node clock frequency present no problem. For example a Node clock frequency of 40 MHz and a and a Timestamp frequency of 13.3 MHz, or a Node clock frequency of 20 MHz and a Timestamp frequency of 6.66 MHz are acceptable operating conditions.

MEASUREMENT SUPPORT

It is expected that one MultiKron II chip will be used for each node of a multiprocessor, where a node is a shared-memory, tightly-coupled cluster of one or more processors. The MultiKron II chip (Figure 1) is a memory mapped device requiring a block of addresses; see Table B4 of APPENDIX- B for summary of address mapping. In this discussion, addresses are those of 64-bit locations, and are given as **decimal** numbers. A sample is triggered by a memory write to a specified address within that block of addresses. A Trace sample (Table B2 of Appendix-B) consists of the data from the memory write, along with a timestamp, a source address, and a header byte. The Trace data is assembled and stored in a small internal FIFO until output to the Collection Network. A Resource sample (Table B2 of Appendix-B) contains the same data as a Trace sample but also includes the contents of the Resource Counters. The contents of the Resource Counters are stored in a set of shadow registers until output to the Collection Network. Each sample is sent out over the Collection Network byte-serially, with a parity bit and end-of-message flag bit on each byte.

Software Reset

This pseudoregister (Address=base+0; Write Only, data is ignored) initializes most of the machine state of the MultiKron II. Unlike the hardware reset pin (RESETB), it does not affect the contents of the Timestamp Counter.

Timestamp

This 56-bit counter (Address=base+2; Read Only, 56 bits) tallies the Timestamp clock and is reset to zero by a system reset (RESETB) on power-up. All 56 bits of the counter are included in a data sample. The hardware system reset signal is a low-active asynchronous signal and must be active for a minimum of five Node clock cycles. To synchronize the timestamp of multiple MultiKron IIs in the machine, the occurrence of this reset should be synchronized throughout the machine. Timestamp Synchronization is then maintained by distribution of a common clock to each MultiKron II chip. The Timestamp clock should not exceed 1/3 (one third) of the MultiKron II Node clock frequency. Thus for a node clock of 50 MHz, the timestamp clock should not exceed 16.6 MHz. The Timestamp is readable in full 56-bit precision, but is not writable except in test mode. The 56-bit Timestamp yields a wrap-around epoch of over a 100 years at 10 MHz.

CSR Register

The Control and Status Register (Address=base+1; Read/Write, 16 bits) is used to configure the basic operational modes of the MultiKron II and to allow examination of these settings. A logical "1" in a bit position of the data written to the CSR commands the corresponding action. A logical "0" in a bit position of the data written to the CSR causes no effect on the corresponding action. This allows an experimenter to modify the field of interest, without having to be concerned with previous setting of other fields. All unused bit positions in the control and status register return a logical "0" on read. The register format is shown in Table B1 of Appendix-B.

The number of (Node clock) Wait States (bit 12) is set by the logic level at the package pin during hardware reset. The wait states allow use with processors that would otherwise provide very short times for address decoding by the MultiKron II.

Filter Register

The Filter Register (Address=base+4; Read/Write, 16 bits) controls whether or not a sample is taken when a trigger occurs. Sixteen filter levels (or groups) of measurement data are defined. Any combination of groups can be enabled by the pattern of bits in the Filter Register. Each sample specifies, as the least-significant 4 bits of its MultiKron II address, a 4 bit encoded filter value which is decoded into one of 16 filter levels. If the corresponding bit in the Filter register is enabled (a "1"), the sample is taken. If the corresponding bit in the Filter register is disabled (a "0"), then the sample is discarded. This allows the program to be extensively instrumented with measurement triggers, while only taking data samples of the desired subset of these triggers. Thus the measurement instrumentation code could remain in the program, causing no difference in execution time, while only taking selective measurements or none at all.

Source Address Registers

The eight Source Address Registers (Addresses=base+32 to 39; Read/Write and sample Generation, 32 bits) contain the identity ("node.process" - node number and process identification) of the process executing on the corresponding processor and should be updated at each context switch. The MultiKron II contains a Source Address register for up to eight processors that it can simultaneously support. That is, the identity of the node and process executing on processor 3 (0..n) of this node should be contained in source address register 3. Eight pins on the MultiKron II chip are used to identify the processor, within that node, writing the measurement sample and to select one of the eight registers, causing the corresponding contents to be included in the Trace sample. Thus, at most, only one of the eight lines can be active at any time. Each Source Address register is assigned a unique address, which is used when the contents must be updated or examined by a processor. Any processor in the node can read or write all of the Source Address registers.

High Order 32 Bit Register

The MultiKron II is designed for a 64 bit processor interface, but has a High Order 32 Bit Register (Address=base+7; Read/Write, 32 Bits) that can be used to implement a 32 bit processor interface. By placing the MultiKron II into its 32 bit mode via the Control and Status (CSR) register, input data bits 32-63 will be taken from this High Order 32 bit register on processor writes rather than the input pins. Output data bits 32-63 are always (whether or not in 32 bit mode) placed in that register, as well as on the corresponding output pins, on processor reads. Thus, in 32 bit mode two reads or two writes are required instead of one in 64 bit mode. In 32 bit mode, the High Order 32 bit register should be written to first before the normal write, and read from after the normal read. Note that the MultiKron II makes no provision to keep these two reads or writes indivisible. Interrupts or multiple processors could overwrite data by interleaving their operations.

Wait Error Counter

The Wait Error Counter (Address=base+5; Read or Clear, 32 Bits) tallies the number of Node clock cycles in which the MultiKron II has delayed the acknowledgement (ACKB) signal to the processor, beyond the normal preset number (i.e., invoked additional wait states) because the FIFO or Shadow registers of the Resource Counters are full. This counter is only activated when either the "Wait on sample for available FIFO/Shadow Reg to avoid overrun" or "Wait on processor read-with-copy for available Shadow Reg" options are enabled in the CSR register. In the unlikely event that the number of waits reaches 2^{32}, this counter wraps around to zero and continues counting.

Overrun Error Counter

The Overrun Error Counter (Address=base+6; Read or Clear, 32 Bits) tallies the number of times a sample has failed to be taken because either the Trace sample FIFO or the Shadow registers were already full when the sample required these resources. This counter is only activated when the "Wait on sample for available FIFO/Shadow Reg to avoid overrun" option is disabled in the CSR register. In the unlikely event that the number of overruns reaches 2^32, this counter wraps around to zero and continues counting.

RESOURCE COUNTERS

The sixteen Resource Counters can be used to count clocks, external hardware events, or software-commanded events. The accumulated value of these counts can be sent over the collection network along with Trace event data via a Resource sample.

Resource Counters and their Shadow Registers

The Resource Counters (Addresses=base+64 to 95; Read/Write/Increment and Sample Generation, 32 Bits) accumulate counts of either the Node clock, the Timestamp clock, 1/10 the Node clock, 1/100 the Node clock, external signals (rising edge), or software generated triggers. The external input signals must not exceed the Node clock frequency. Each even/odd address pair of Resource Counters can be configured as a single 64 bit Resource Counter, greatly increasing its maximum value.

When a Resource sample is triggered or one of the Resource Counters is being read, the contents of all Resource Counters are copied into the corresponding shadow registers, while the Resource Counters can continue to count. This ensures that the counter contents used in the Resource measurement sample all correspond to the same instant - the time of the sample. Then the contents of the shadow register are used for data output. There is only one rank of shadow registers, in effect a one-level FIFO. When loaded due to a sample, the shadow registers will remain busy for 84 network clock periods (168 Node clock periods -- the network clock is generated by the MultiKron II and is half the frequency of the Node clock) during the normal output transmission of the data sample. The shadow registers cannot be reused during this period.

The Resource Counters will not count beyond their maximum value (i.e, no wrap around), but will stick at this maximum value until reset. The length of time it takes for a 32 bit counter to reach its maximum value depends on the rate of the signal being tallied. For a 50 MHz signal a 32-bit counter fills in about 85 seconds; for a 5 MHz signal a 32-bit counter fills in about 14 minutes; for a 500 kHz signal a 32-bit counter fills in about 2.3 hours. For a 50 MHz signal a 64-bit counter fills in about 12,000 years; for a 5 MHz signal a 64-bit counter fills in about 120,000 years; for a 500 kHz signal a 64-bit counter fills in about a million years.

Processor Reading and Writing of a Resource Counter. Each Resource Counter has a specific address and may be read or written by a processor. Writing to the address of a Resource Counter (address 64-79) will place the data written into that Resource Counter, the shadow registers are not affected. There are two types of reads for the Resource Counters, one is a read-with-copy (address 64-79) and the other is a read-without-copy (address 80-95). All reads are done from the shadow registers, the copy refers to whether or not the contents of the Resource Counters are copied into the shadow registers before reading. Reading-with-copy any Resource Counter, causes all Resource Counters to be copied into their shadow register. Thus a read-with-copy yields the most current Resource Counter value, while a read-without-copy yields the value last copied (if any). Reading-with-copy all of the Resource Counters consecutively will yield data at 16 different instances. To obtain the data from all of the Resource Counters at the same instant, one read-with-copy should be followed by 15 reads-without-copy. This along with the ability to write the Resource Counters provides a means to implement a virtual set of Resource Counters. A subset of the Resource Counters could be reserved for user processes (vs. the kernel process) and the value of those Resource Counters, and their configuration settings, could be saved and restored at context switches.

Processor Incrementing of a Resource Counter. Writing to a Resource Counter's increment address (address 80-95) will causes the corresponding Resource Counter to be incremented, if the "software increment" option is selected in the control registers. This software increment allows the programmer to selectively use the Resource Counters to accumulate counts of software events.

Overruns and Waits on Resource Counter Use. A problem occurs when an attempt is made to copy the data from the Resource Counters (either by read-with-copy or Resource sample) into their shadow registers when the shadow registers are still busy outputting the data to the Collection Network from the previous Resource sample. Shadow registers are never marked busy from a read-with-copy, only from a Resource sample. The shadow registers act as a one-level-deep FIFO for the Resource Counters. For sixteen Resource Counters, this blockage could last 84 (the length of a Resource sample) network clocks, since only one byte is transferred on each network clock cycle. Upon such a copy request, there are two options, one option is to block the request until the shadow registers are no longer busy and then complete the request. The other option is to discard the request. Such blocking options are offered in the CSR, individually for read-with-copy and for Resource samples. When the shadow registers are not busy, no blockage will occur. Table B3a of Appendix-B summarize the actions for the CSR option related to Resource samples. If blocking is enabled, the next Resource sample will be blocked (but not Trace samples, since Trace samples have a separate, deeper FIFO) while these shadow registers are being output to the Collection Network. Upon completion the pending Resource sample will be taken. If blocking is disabled, the next Resource sample request will be discarded although a completion acknowledgement will be sent to the processor. Table B3b of Appendix-B summarize the actions for the CSR option related to read-with-copy. If blocking is enabled, attempts to read-with-copy a Resource Counter by a processor will be blocked while these shadow registers are being output to the collection network, when done the processor read will complete. If blocking is disabled, the read-with-copy will be discarded although a completion acknowledgement will be sent to the processor along with erroneous data.

No blockage occurs when writing to or incrementing the Resource Counters, since this doesn't affect the shadow registers. A read-without-copy will not be blocked even when the shadow registers are busy. A read-without-copy doesn't affect the shadow registers, but it does share a common data bus with the Collection Network operations. So a read-without-copy will delay the Collection Network for two node clock times, when the shadow registers are busy, to allow the reading of the shadow registers by the processor.

Resource Counter Control Registers

The Resource Counters can be individually controlled via a set of control registers: the Enable; Mode; and Clock Select registers. Each of these control registers is divided into 16 4-bit fields, one for each of the 16 Resource Counters. These control registers are designed so that a zero value written into any field leaves that field unchanged. Only a non-zero value will change a field. This allows subsets of the Resource Counters to be shared between different experimenters or different parts of one experiment, without the need to know the control settings of other fields. It is expected that the counters would be disabled prior to their use via the Enable register, and the counting source selected for each counter via the Mode register. If an internal clock is selected as a counting source, the choice of clocks is selected via the Clock Select register. Resource Counters can then be selectively enabled or disabled as desired via the Enable register to accumulate counts during the experiment. The Resource Counters can be configured to provide a stop-watch feature for either hardware or software events. For software events, configure a counter to count an internal clock. Then via the Enable register, Enable the counter at a start event and disable it at an end event. For hardware events, configure a counter to count an internal clock only when the external signal is active. Once Enabled, via the Enable register, the counter will only count when the external signal is active. The external signal thus acts as an Enable/Disable to the counter.

Mode Register. The Mode Register (Address=base+10; Read/Write, 64 bits) is used to select one of four counting sources for each Resource Counter, as well as determining if the odd/even counter pair should be a single 64 bit counter or separate 32 bit counters, see Table B5 of Appendix-B. The counting sources are an internal clock, a software generated signal, an external signal from a package pin private to each counter, or an internal clock only when the external signal is active (i.e., the external signal is used as an enable signal).

Clock Select Register. The Clock Select Register (Address=base+12; Read/Write, 64 bits) is used only if the Mode Register selects an internal clock as the countering source, otherwise it may be ignored, see Table B6 of Appendix-B. The Clock Select Register selects which internal clock to count. The choices are node clock; 1/10th node clock; 1/100th node clock; or the timestamp clock frequency.

Enable Register. The Enable Register (Address=base+8; Read/Write, 64 bits) is used to enable or disable the Resource Counters, see Table B7 of Appendix-B. There are two choices when enabling a counter. One choice is to reset the counter before enabling it. The other choice is to just enable the counter with its previous contents intact.

SAMPLING

Writing to the Sample pseudoregisters (Address=base+96 to 127; Sample Generation, 64 Bits) causes a sample to be assembled in the 160 bit wide MultiKron II FIFO. The sample is stored in the FIFO until it is output to the Collection Network. Writing to addresses 96 through 111 causes a Trace sample, while writing to addresses 112 through 127 causes a Resource sample. The formats of Trace and Resource samples are shown in Table B2 of Appendix-B. The least significant 4 bits of the MultiKron II address specify a Filter level. If the correspond entry in the Filter Register is enabled, the sample is assembled. If the correspond entry in the Filter Register is disabled, the request is discarded (no sample is assembled) although the processor write is acknowledged normally via the ACKB signal.

A Trace sample is formed by taking the 64 bits of the user data written to these addresses and adding the contents of a Source Address register, the Timestamp, and a header byte. To form a Resource sample, the Resource Counters are appended to the Trace sample data. A Trace sample contains 160 bits (20 bytes). In a Resource sample, each of the sixteen Resource counters adds 32 more bits, for a total of 672 bits (84 bytes).

The selection of which Source Address register to include in a sample is determined by the 8 CPU ID external inputs (C0-C7) to the MultiKron II. Only one of these 8 inputs should be active during a sample to identify which processor within the node is triggering the sample. There are 8 Source Address registers, one for each CPU ID external input. The contents of the Source Address register that corresponds to the active CPU ID external input is included in the sample, and the 3 bit encoding of these 8 CPU ID external inputs is included in the header byte to identify the processor associated with this sample.

Each sample contains a header byte (8 bits: 3 - CPU ID, 2 - sample type, 2 - error flags, 1 - N/A) which is used to identify the processor within the node taking the sample, error flags associated with the sample, and the type of sample. The two error flags are:

1) FIFO overrun - previous sample(s) lost (reset by next successful sample),
2) Shadow register overrun - previous sample(s) lost (reset by next successful sample),

Two bits are allocated for sample type, Trace sample (encoding 11) or Resource sample (encoding 10). The CPU ID is a 3 bit encoding of the 8 processor ID lines.

NETWORK OUTPUT

The data Collection Network output of the MultiKron II chip provides a way for the data samples to reach a collection point without using the computer's normal data paths. Output from the MultiKron II consists of ten-bit elements: eight data bits, an odd-parity bit, and a sample-end flag bit. The MultiKron II delivers a network clock output signal, NETCLK, at one-half the Node clock frequency. Network data output is synchronous with this signal. See Figure 3 for timing specifications of the network output. There are two control signals:

(1) NETRDY - External network ready (input) . The network must assert (high) or de-assert (low) the NETRDY signal at least 16 nanoseconds before the positive-going transition of NETCLK.

(2) FIFODAB - MultiKron II output data available (output). The MultiKron II asserts (low) FIFODAB at least 12 nanoseconds before the positive-going edge of NETCLK if the network must accept the accompanying output data byte. This signal will not be asserted if the NETRDY signal is not asserted.

TEST MODE

The MultiKron II chip can be placed in the test mode only by enabling (LOW) the TESTB pin. When in test mode the Timestamp, the Wait Error Counter, and the OverRun Error Counter become writable and incrementable by the processor. Also the Collection Network may be disabled and the FIFO may be written directly by the processor. These features could conflict with normal operations and so were place in this special test mode.

The FIFO can be read by the CPU, at any time, but only in groups of 32 bits. Thus the 160 bit FIFO is divided into five 32-bit groups (A-E), from the most significant bit to the least significant bit, respectively. But successfully reading data out of the FIFO when the Collection Network is operational is a problem, since the data may be removed from the FIFO at any time. Thus in test mode the Collection Network should be disabled so that the processor is controlling the FIFO data. Note that reading the FIFO by the CPU does not cause the data to be shifted out of the FIFO, as is the case when the Collection Network reads the FIFO data. FIFO shifting is a separate processor command in test mode. The processor can write directly to the FIFO in test mode (via addresses 25-29). The 32 bits of data are duplicated in each 32 bit FIFO group. The processor can always indirectly write to the FIFO by taking a sample.

STATUS

The MultiKron II integrated circuit illustrates that powerful hardware support for multiprocessor computer performance characterization can be incorporated in a single chip. The MultiKron II has been fabricated in 1 micrometer CMOS using a standard cell library. These chips have been packaged in both a 179 PGA (Pin Grid Array -- very nearly compatible with the original MultiKron) and a 208 QFP (Quad Flat Pack). A 60 percent yield was achieved from the first fabrication run, based on 40 MHz testing. Preliminary testing indicates that these chips can successfully operate at 50 MHz.

REFERENCES

[CAR88] Carpenter, R.J. Performance measurement instrumentation for multiprocessor computers. Gelenbe (ed)., High Performance Computer Systems. 1988; Paris; North Holland; pp 81-92; ISBN 0 444 70485 X.

[CAR89] Carpenter, R.J. Performance measurement instrumentation at NBS. Simmons, et al, eds.; Instrumentation for Future Parallel Computing Systems; Santa Fe; Addison-Wesley (1989) pp 159-183; ISBN 0 201 50390 5.

[MIN90] Mink, A.; Carpenter, R.; Nacht, G.; Roberts, J. Multiprocessor performance-measurement instrumentation. IEEE Computer: 63-74; 1990 September.

[MIN92] Mink, A. and Carpenter, R.J. Operating Principles of MultiKron Performance Instrumentation for MIMD Computers. Natl Inst of Standards and Technology, Gaithersburg, MD., NISTIR 4737; 1992 Mar.

[ROB89] Roberts, J.; Antonishek, J.; Mink, A. Hybrid performance measurement instrumentation for loosely-coupled MIMD architectures. Proc. 4th Distributed Memory Computer Conf. (DMCC4); 1989; Monterey, CA; 7 p.

Appendix-A Summary of Differences

The following is a summary of the differences between the MultiKron and the MultiKron II.

Pads and Packages

Both the MultiKron and the MultiKron II have 179 pads, but the MultiKron II uses two of these pads differently. One of these pads was designated for NIST internal testing on the MultiKron, and should, operationally, always be connected to ground. When connected to ground, the new use of this pad will not interfere with normal operations. The function of this new pad, called **STARTB**, is to initiate a processor interaction, along with either the READB or WRITEB pads. Either the READB/WRITEB signals or the STARTB signal must be de-asserted (high) prior to the MultiKron II asserting (low) the ACKB signal. When STARTB is de-asserted, no processor interaction will occur. Only when STARTB is asserted (low) and either READB or WRITEB is asserted will a processor interaction occur. Once a processor interaction has started, the logic value of READB, WRITEB, and STARTB are ignored. The intent of this new signal is to allow a start signal in conjunction with the READB and WRITEB signals, so as to remove the timing constraints on READB/WRITEB signals and alternatively place them on the STARTB signal. Thus the new signal could always be set low and READB and WRITEB will operate exactly as in the MultiKron.

The second different pad is one of the two pads which specified the number of wait states, 0 to 3, to use when decoding an address during a processor interaction. Experience has show that a single pad specifying 0 or 1 wait states is sufficient. The new use of this pad, called **HOLDB**, is to extend the ACKB signal and its associated data for as long as HOLDB is asserted. If HOLDB is de-asserted (high), this signal will be ignored and the ACKB signal will be asserted for 1 node clock. When the ACKB signal is de-asserted the data is no longer valid. The use of HOLDB is mainly targetted for processor read operations.

The MultiKron was only available in a 179 pin PGA (Pin Grid Array) package for through-hole mounting on printed circuit boards. The MultiKron II is available in both a 179 pin PGA package, for compatibility with the MultiKron, and also placed in a **208 QFP (Quad Flat Pack) package** for surface mounting on printed circuit boards. The extra 29 pins of the QFP package are not used.

Processor Data Path

On a read by a processor, any unused bit positions return a logical "1", unlike MultiKron which returned a logical "0". The MultiKron II is designed for a 64 bit processor interface, but has a new **High Order 32 bit register** that can be used to implement a 32 bit processor interface. By placing the MultiKron II into its 32 bit mode via the Control and Status Register, data bits 32-63 will be taken from this High Order 32 bit register on processor writes. Output data bits 32-63 are always (whether or not in 32 bit mode) placed in that register, as well as on the corresponding output pins, on processor reads. Thus, in 32 bit mode two reads or two writes are required instead of one in 64 bit mode. In 32 bit mode the High Order

32 bit register should be written to first before the normal write and read from after the normal read. Note that the MultiKron II makes no provision to keep these two reads or writes indivisible. Interrupts or multiple processors could overwrite data by interleaving their operations.

Sample Size

The MultiKron Trace sample was 16 bytes (1 byte header, 5 bytes Timestamp, 4 bytes Source Address, 6 bytes User Data). The MultiKron II **Trace sample has been expanded to 20 bytes** (1 byte header, 7 bytes Timestamp, 4 bytes Source Address, 8 bytes User Data). This larger sample size allows for a full 64 bits of user data and for the full 56 bits of the Timestamp.

Resource Counters

The are still 16 Resource Counters, each with their own shadow register. The shadow registers are the only means of reading the contents of the Resource Counters. They also act as a 1 level deep FIFO for Resource samples being output to the Collection Network. The Resource Counters on the MultiKron II are both **fully readable and writable**, whereas the Resource Counters on the MultiKron were read only. This allows for "virtual" use of the Resource Counters. The Counters could be allocated to different processes, and on context switches the values of the Counters and their control registers could be saved and restored.

The MultiKron II allows two types of Resource Counter reads, **read-with-copy and read-without-copy**. The copy refers to whether or not the contents of the Resource Counters are copied to the shadow registers before the shadow registers are read. The MultiKron only provided for a read-with-copy. The read-with-copy functions exactly as on the MultiKron, reading any Resource Counter copies all of them to their shadow registers. Thus reading-with-copy all 16 Resource Counters consecutively will yield the Resource Counters' value at 16 different instances. Reading-with-copy one Resource Counter and then reading-without-copy the other 15 will yield the value of all 16 Resource Counters at one instant. Also read-without-copy will not be blocked when the shadow registers are busy outputting to the Collection Network, since their value does not need to be updated for the read to complete.

The control registers for the Resource Counters on the MultiKron II have been revised. There are three control registers, the Enable Register, Mode Register, and the Clock Select Register, see Tables B5 through B7 of Appendix-B for the encoding of these registers. They are all 64 bit registers divided into 16 4-bit fields, one field for each Resource Counter. An all zero field is a NOP (i.e., it doesn't change the current value of that field). This allows controlling selected Resource Counters without having to be concerned with the current settings of the other Resource Counters. The MultiKron Enable, Disable, and Reset registers have been combined into a single **Enable Register** on the MultiKron II, with a separate encoding for enable counter, disable counter, reset and then enable counter. The **Mode Register** of the MultiKron II is similar to the MUX SEL register of the MultiKron, it is used to select the counting source for a counter and also used to specify whether an odd/even pair of Counters should function as two 32 bit Counters or a single 64-bit Counter (i.e., **double precision**). One bit of this field specifies double precision, while the other three bits specify the counting source as either an internal clock, a software signal, an external signal, or an internal clock gated by the external signal. The last source is new and allows an external signal to control the enabling of this counter. When the external signal is active the internal clocks are counted, when the external signal is not active no counts are accumulated. This is useful in determining how long certain external signals are active or for computing duty cycles. If the Mode Register specifies an internal clock is to be counted, The **Clock Select Register** must be set to specify which clock to count. The choices are the Node Clock, 1/10 the Node Clock, 1/100 the Node Clock, or the Timestamp Clock.

Testing

The complicated Test Register and associated Test Instructions of the MultiKron have been removed. A much simpler Test interface was designed for the MultiKron II. **All registers, counters, and the FIFO are read/write in Test Mode.** Test Mode is entered by asserting (low) the TESTB signal of the MultiKron II, just as it was for the MultiKron. All Test functions and the writability of the Error Counters and the Timestamp Counter (see Table B4 of Appendix-B, address map) are implemented as part of the normal MultiKron II addresses (17-21) which do nothing when written to in non-Test mode.

Appendix-B Address and Format Reference Tables

Table B1. Control and Status Register (CSR) Format

Status (CSR read)	Bit Position	Control (CSR write of "1" in this bit position)
Sampling Enabled	0	Enable Sampling
Logical "0"	1	Disable Sampling (default on reset)
Write Wait on Overrun	2	Enable Wait on sample for avail FIFO/Shadow Reg to avoid overrun
Logical "0"	3	Disable Wait on sample due to overrun, discard data (default)
Read Wait for shadow regs.	4	Enable wait on processor read-with-copy if Shadow Regs busy
Logical "0"	5	Disable read wait if Shadow Reg busy (bad data returned) (default)
FIFO Full	6	Read Only, write data ignored.
Resource Shadow Registers Full	7	Read Only, write data ignored.
FIFO Overrun	8	Read Only, write data ignored.
Resource Shadow Register Overrun	9	Read Only, write data ignored.
161st FIFO output bit	10	Read Only, write data ignored.
Logical "0"	11	Read Only, write data ignored.
Number of Wait States	12	Read Only, write data ignored.
Logical "0"	13	Read Only, write data ignored.
32 bit mode enabled	14	High Order Reg is used for upper 32 bits of input data
32 bit mode disabled(64 bit mode)	15	pins 32-63 are used for upper 32 bits of input data (64 bit mode)

Table B2. Sample Formats

TRACE Sample (160 bits / 20 bytes)

Sent first

Header	Timestamp	Source Identification	User-written Data
8 bits	56 bits	32 bits	64 bits

RESOURCE Sample (672 bits / 84 bytes)

Sent first

Header	Timestamp	Source Identification	User-written Data	Resource Cntr 0	...	Resource Cntr 15
8 bits	56 bits	32 bits	64 bits	32 bits		32 bits

1 byte Sample Header Format

most significant bit

CPU ID Encoded	Sample Type 10=Trace 11=Resource	Error Shadow Reg Overflow	Error FIFO Overflow	N/A
3 bits	2 bits	1 bit	1 bit	1 bit

Table B3a. Effects of CSR "Wait on sample for available FIFO/Shadow Reg on overrun" Bit

CSR "Wait on Overrun" Control Bit	Trace or Resource Sample	FIFO status	Shadow Reg status	Overrun Error Bit in Next Sample Header		Operation Performed on Error Cntr		Description of Action
				FIFO	Shadow	Overrun	Wait	
1	T or R	Avail	Avail	0	0	NOP	NOP	Data taken immediately
1	T	Avail	Full	0	0	NOP	NOP	Data taken immediately
1	R	Avail	Full	0	0	NOP	CNT	Wait until Shadow Regs are available, then take data
1	T or R	Full	Avail	0	0	NOP	CNT	Wait until FIFO is available, then take data
1	T	Full	Full	0	0	NOP	CNT	Wait until FIFO is available, then take data
1	R	Full	Full	0	0	NOP	CNT	Wait until both FIFO and Shadow Regs are available, then take data
0	T or R	Avail	Avail	0	0	NOP	NOP	Data taken immediately
0	T	Avail	Full	0	0	NOP	NOP	Data taken immediately
0	R	Avail	Full	0	1	Incr	NOP	Data Discarded
0	T or R	Full	Avail	1	0	Incr	NOP	Data Discarded
0	T	Full	Full	1	0	Incr	NOP	Data Discarded
0	R	Full	Full	1	1	Incr	NOP	Data Discarded

NOP - no operation performed   CNT - count Node clocks during wait   Incr - increment

Table B3b. Effects of CSR "Wait on processor read-with-copy for available Shadow Reg" Bit

CSR "Wait on Read" Control Bit	Read with or without Copy	FIFO status	Shadow Reg status	Operation Performed on Error Cntr		Description of Action
				Overrun	Wait	
1	w	-	Avail	-	NOP	Data copied and then read
1	w	-	Full	-	CNT	Wait until Shadow Reg are available, then copy data and read
1	w/o	-	Avail	-	NOP	Return current contents of Shadow
1	w/o	-	Full	-	NOP	Delay Network output, if necessary, while current contents of Shadow Regs are read
0	w	-	Avail	-	NOP	Data copied and then read
0	w	-	Full	-	NOP	Don't Wait, but return erroneous data
0	w/o	-	Avail	-	NOP	Return current contents of Shadow
0	w/o	-	Full	-	NOP	Delay Network output, if necessary, while current contents of Shadow Regs are read

NOP - no operation performed   CNT - count Node clocks during wait

Table B4. MultiKron II Address Assignments

Address Offsets		Write	Read
decimal	Hex		
0	x0	Reset chip, Timestamp unchanged	N/A
1	x1	CSR Register	CSR Register
2	x2	* Timestamp Register (56 bits)	Timestamp Register (56 bits)
3	x3	-	-
4	x4	Filter Register	Filter Register (16 bits)
5	x5	** Wait Counter to zero	Wait Counter
6	x6	** Overrun Counter to zero	Overrun Counter
7	x7	High Order 32 bit Reg	High Order 32 bit Reg
8	x8	Enable Register (64 bits)	Enable Register (64 bits)
9	x9	-	-
10	xA	Mode Register (64 bits)	Mode Register (64 bits)
11	xB	-	-
12	xC	Clock Select Register (64 bits)	Clock Select Register (64 bits)
13-15	xD-xF	-	-
16	x10	* Incr Timestamp Register	-
17	x11	* Incr Wait Error Cntr	-
18	x12	* Incr OverRun Error Cntr	-
19	x13	* Disable Output Network	-
20	x14	* Enable Output Network	-
21	x15	* Advance FIFO and free Resource Cntr Shadow Reg	
22-24	x16-x18	-	-
25	x19	* FIFO	FIFO group A
26	x1A	* FIFO	FIFO group B
27	x1B	* FIFO	FIFO group C
28	x1C	* FIFO	FIFO group D
29	x1D	* FIFO	FIFO group E
30-31	x1E-x1F	-	-
32-39	x20-x27	Source Addr Reg i (0..7)	Source Addr Reg i
40-63	x28-x3F	-	-
64-79	x40-x4F	Resource Counter j (0..15)	Resource Counter j (with copy)
80-95	x50-x5F	Increment Resource Counter j (0..15)	Resource Counter j (without copy)
96-111	x60-x6F	Trigger a Trace Sample w/o resource data, with filter level 0..15 and Source address index 0..7 from hardware pins.	-
112-127	x70-x7F	Trigger a Resource Sample including Trace data, with filter level 0..15 and Source address index 0..7 from hardware pins.	-

* Available ONLY in TEST mode ** Reset to zero ONLY when NOT in TEST mode

- A Trace sample w or w/o resource data, contains 64 bits of user data.
- The source address index for addresses 16-23 is computed from the 3 least significant address bits.
- The source address index for sampling, addresses 96-127, is the encoded value of the active CPU ID line of the MultiKron II package pins. Only one of the eight CPU ID lines may be active at a time.
- The filter level for sampling, addresses 96-127, is computed from address bits 0-3 (least significant 4 bits).
- The Resource Counter, addresses 64-95, is computed from address bits 0-3 (least significant 4 bits).

Table B5. Mode Register encoding (4 bit field/Resource Cntr)

four Bit Encoding	Description
x000	NOP
x001	CLK (Count internal Clks)
x010	Software (Increment counter by software)
x011	External Enable (Count internal Clks only when the External signal is active high)
** x100	External Signal (Count external signals, low to high)
x101	(same as CLK)
x110	(same as Software)
x111	(same as External Enable)
** 0xxx	Single precision
1xxx	Double precision (This setting applies only to EVEN Resource Counters, ODD Counters use the value from their EVEN pair)

Table B6. Clock Select Register encoding (4 bit field/Resource Cntr)

four Bit Encoding	Description
x000	NOP
x001	NodeClk (chip input Node clk)
x010	NodeClk/10 (NodeClk divided by 10)
x011	NodeClk/100 (NodeClk divided by 100)
** x100	TSclk (chip input timestamp clk)
x101	(same as NodeClk)
x110	(same as NodeClk/10)
x111	(same as NodeClk/100)

Table B7. ENABLE Register encoding (4 bit field/Resource Cntr)

four Bit Encoding	Description
xx00	NOP
** xx01	Disable counter
xx10	Enable counter
xx11	Reset & Enable counter

** default setting

Appendix-C MultiKron II Pad Allocation and Pin Assignments

Table C1. MultiKron II Pad Allocation.
On 179 PGA package all pins are used,
for the 208 QFP package the extra 29 pins are not used.

Number of Pins	Input or Output	Use
64	I/O	CPU data bus
16	I	External inputs for Resource Counters
7	I	Address lines
8	I	CPU ID (NOT encoded)
1	I	ResetB
1	I	Node clock
1	I	Timestamp clock
1	I	ReadB strobe
1	I	WriteB Strobe
1	I	HoldB (ACKB and data valid)
1	I	StartB (used with Read or Write above)
1	I	Number of CPU Wait States, valid only during reset
1	O	read/write ACKB
10	O	Network output (8 data; 1 odd parity; 1 EOM, high active)
1	I	Network Ready
1	O	FIFO Data Available
1	O	Network clock
1	I	TESTB Mode
1	I	OUTDISB (Place all output pins in high impedance mode)
8	O	used for internal testing
127		TOTAL Data pins
52	I	PWR & GND
179		GRAND TOTAL pins

Table C2a. MultiKron II - 179 PGA Pin Assignments (930727).

		45	D29	90	(T2)	135	X15
1	D0	46	(P)	91	D57	136	(P)
2	(P)	47	D31	92	(P)	137	X11
3	D1	48	OUTDISB	93	D58	138	X12
4	(G)	49	D30	94	(G)	139	X13
5	D2	50	(G)	95	D59	140	(G)
6	D4	51	D33	96	G	141	X9
7	D3	52	G	97	D60	142	G
8	G	53	D32	98	P	143	X10
9	D5	54	D35	99	D61	144	X7
10	D7	55	D36	100	D63	145	X6
11	D6	56	P	101	D62	146	P
12	P	57	D34	102	N1	147	X8
13	D8	58	D38	103	N0	148	X4
14	D9	59	D37	104	N2	149	X5
15	D10	60	P	105	N3	150	G
16	D11	61	D39	106	G	151	X3
17	D13	62	D40	107	N5	152	X2
18	(P)CORE	63	(P)CORE	108	(P)	153	(P)
19	(G)CORE	64	(G)CORE	109	(G)	154	(G)
20	D12	65	D41	110	N4	155	X1
21	D14	66	D42	111	N6	156	P
22	(P)	67	D43	112	(T3)	157	Node_clk
23	D15	68	(T1)	113	N7	158	(T5)
24	G	69	D46	114	P	159	RESETB
25	(G)	70	(P)	115	(G)	160	(T6)
26	(T0)	71	(G)	116	(P)	161	(T7)
27	D17	72	G	117	ODD_PARITY	162	X0
28	D16	73	D48	118	FIFODAB	163	WAITSTATE
29	D19	74	D45	119	NETCLK	164	Timestamp_clk
30	P	75	D49	120	EOM	165	HOLDB
31	D21	76	D44	121	C6	166	G
32	D18	77	D51	122	NETRDY	167	A5
33	D24	78	D47	123	C3	168	WRITEB
34	G	79	P CORE	124	G	169	ACKB
35	D23	80	G CORE	125	C4	170	READB
36	D20	81	D53	126	C7	171	A2
37	D26	82	P	127	C1	172	P
38	P	83	D50	128	P	173	A6
39	D25	84	G	129	C2	174	TESTB
40	(G)	85	D55	130	(T4)	175	A1
41	D28	86	(P)	131	C0	176	(G)
42	D22	87	D52	132	C5	177	A3
43	D27	88	D54	133	X14	178	A4
44	(P)	89	D56	134	STARTB	179	A0

Signals names ending with "B" indicate active low, all others active high. Signals in parenthesis are located on inner pins which would not exist on a 144 PGA. The word "CORE" next to a power or ground indicates its use is for the core logic vs. the output pads.

D0-D63	(Bidirectional)	CPU Bus
N0-N7	(Output)	Network Bus
W0-W1	(Input)	Wait States
X0-X15	(Input)	External Signal for Resource Counters
C0-C7	(Input)	CPU id Lines
A0-A7	(Input)	Address Lines
T0-T7	(Output)	NIST Internal Testing Only

Table C2b. MultiKron II - 208 QFP Pin Assignments (940331).

1	-	53	-	105	-	157	-
2	-	54	-	106	-	158	-
3	-	55	-	107	-	159	-
4	-	56	D29	108	(T2)	160	X15
5	D0	57	(P)	109	D57	161	(P)
6	(P)	58	D31	110	(P)	162	X11
7	D1	59	OUTDISB	111	D58	163	X12
8	(G)	60	D30	112	(G)	164	X13
9	D2	61	(G)	113	D59	165	(G)
10	D4	62	D33	114	G	166	X9
11	D3	63	G	115	D60	167	G
12	G	64	D32	116	P	168	X10
13	D5	65	D35	117	D61	169	X7
14	D7	66	D36	118	D63	170	X6
15	D6	67	P	119	D62	171	P
16	P	68	D34	120	N1	172	X8
17	D8	69	D38	121	N0	173	X4
18	D9	70	D37	122	N2	174	X5
19	D10	71	P	123	N3	175	G
20	D11	72	D39	124	G	176	X3
21	D13	73	D40	125	N5	177	X2
22	(P)CORE	74	(P)CORE	126	(P)	178	(P)
23	(G)CORE	75	(G)CORE	127	(G)	179	(G)
24	D12	76	D41	128	N4	180	X1
25	D14	77	D42	129	N6	181	P
26	(P)	78	D43	130	(T3)	182	Node_clk
27	D15	79	(T1)	131	N7	183	(T5)
28	G	80	D46	132	P	184	RESETB
29	(G)	81	(P)	133	(G)	185	(T6)
30	(T0)	82	(G)	134	(P)	186	(T7)
31	D17	83	G	135	ODD_PARITY	187	X0
32	D16	84	D48	136	FIFODAB	188	WAITSTATE
33	D19	85	D45	137	NETCLK	189	Timestamp_clk
34	P	86	D49	138	EOM	190	HOLDB
35	D21	87	D44	139	C6	191	G
36	D18	88	D51	140	NETRDY	192	A5
37	D24	89	D47	141	C3	193	WRITEB
38	G	90	P CORE	142	G	194	ACKB
39	D23	91	G CORE	143	C4	195	READB
40	D20	92	D53	144	C7	196	A2
41	D26	93	P	145	C1	197	P
42	P	94	D50	146	P	198	A6
43	D25	95	G	147	C2	199	TESTB
44	(G)	96	D55	148	(T4)	200	A1
45	D28	97	(P)	149	C0	201	(G)
46	D22	98	D52	150	C5	202	A3
47	D27	99	D54	151	X14	203	A4
48	(P)	100	D56	152	STARTB	204	A0
49	-	101	-	153	-	205	-
50	-	102	-	154	-	206	-
51	-	103	-	155	-	207	-
52	-	104	-	156	-	208	-

Table C3. MultiKron II - description of pin names.

Pin Name	Direction	Description
ACKB	(Output)	This low active signal indicates that the MultiKron II chip has completed a CPU Read or Write cycle.
A0-A7	(Input)	CPU Address Lines to the MultiKron II chip.
C0-C7	(Input)	CPU id Lines, identifies the current CPU Writing samples to the MultiKron II chip. At most only one of these lines may be active during a write.
D0-D63	(Bidirectional)	CPU Data Bus to the MultiKron II chip.
EOM	(Output)	End-of-Message signal for a Network Data Packet.
FIFODAB (WENB)	(Output)	This low active signal indicates that Network data is available for the external FIFO memory.
G	(Input)	GROUND
HOLDB	(Input)	This low active signal is used to extend a processor read by keeping the data and ACKB signal active until it is de-asserted.
NETCLK	(Output)	The Network clock for the external FIFO memory.
NETRDY (FFB)	(Input)	This high active signal indicates that Network external FIFO memory is ready to accept data.
N0-N7	(Output)	Network Data Bus to the MultiKron II chip.
ODD_PARITY	(Output)	Parity (odd) value for the accompanying Network Output Data.
OUTDISB	(Input)	This low active signal disables ALL normal output pins.
P	(Input)	POWER
READB	(Input)	This low active signal from CPU starts a read cycle on the MultiKron II chip.
RESETB	(Input)	This low active asynchronous signal resets the MultiKron II chip to its initial state. This signal MUST be active for a minimum of five node clocks.
STARTB	(Input)	This low active signal is used in conjunction with the READB and WRITEB to initiate a processor interaction, if always kept active this signal has no effect.
TESTB	(Input)	This low active signal places the MultiKron II chip in a general test mode, which disables all counters and allows them to be written and incremented by the CPU. This pin MUST be disabled for normal operation.
T0-T7	(Output)	NIST Internal Testing Only.
W0	(Input)	Wait States to respond to a CPU Read/Write request, only active during a MultiKron II chip RESET.
WRITEB	(Input)	This low active signal from CPU starts a write cycle on the MultiKron II chip.
X0-X15	(Input)	External Signal for Resource Counters, counts on positive edge.
Timestamp_clk	(Input)	The Timestamp clock to the MultiKron II chip. It MUST NOT Exceed 1/3 the Node clock frequency.
Node_clk	(Input)	The Node clock to the MultiKron II chip. It is used to synchronize all internal chip operations.

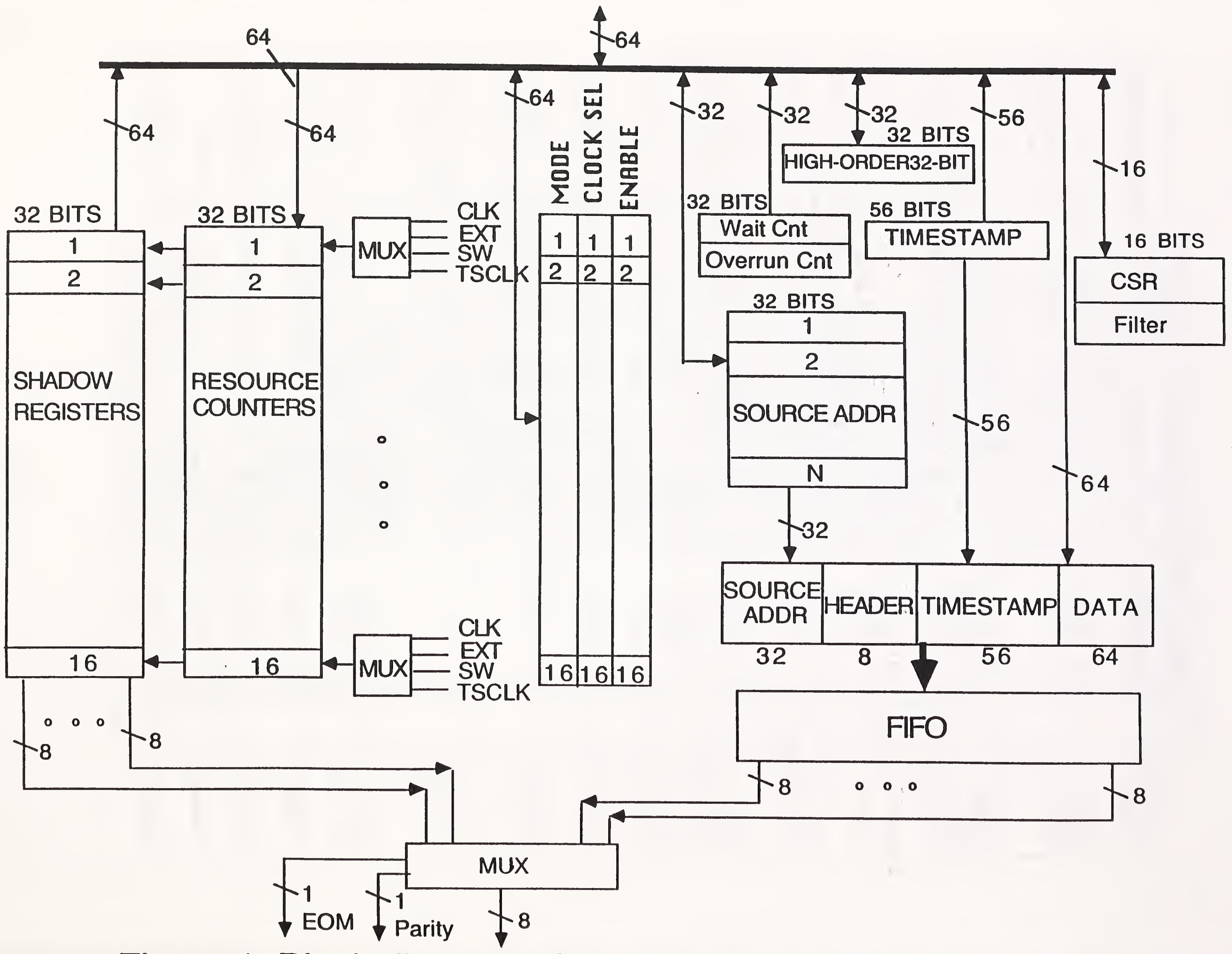

Figure 1. Block diagram of the MultiKron II Chip

940920

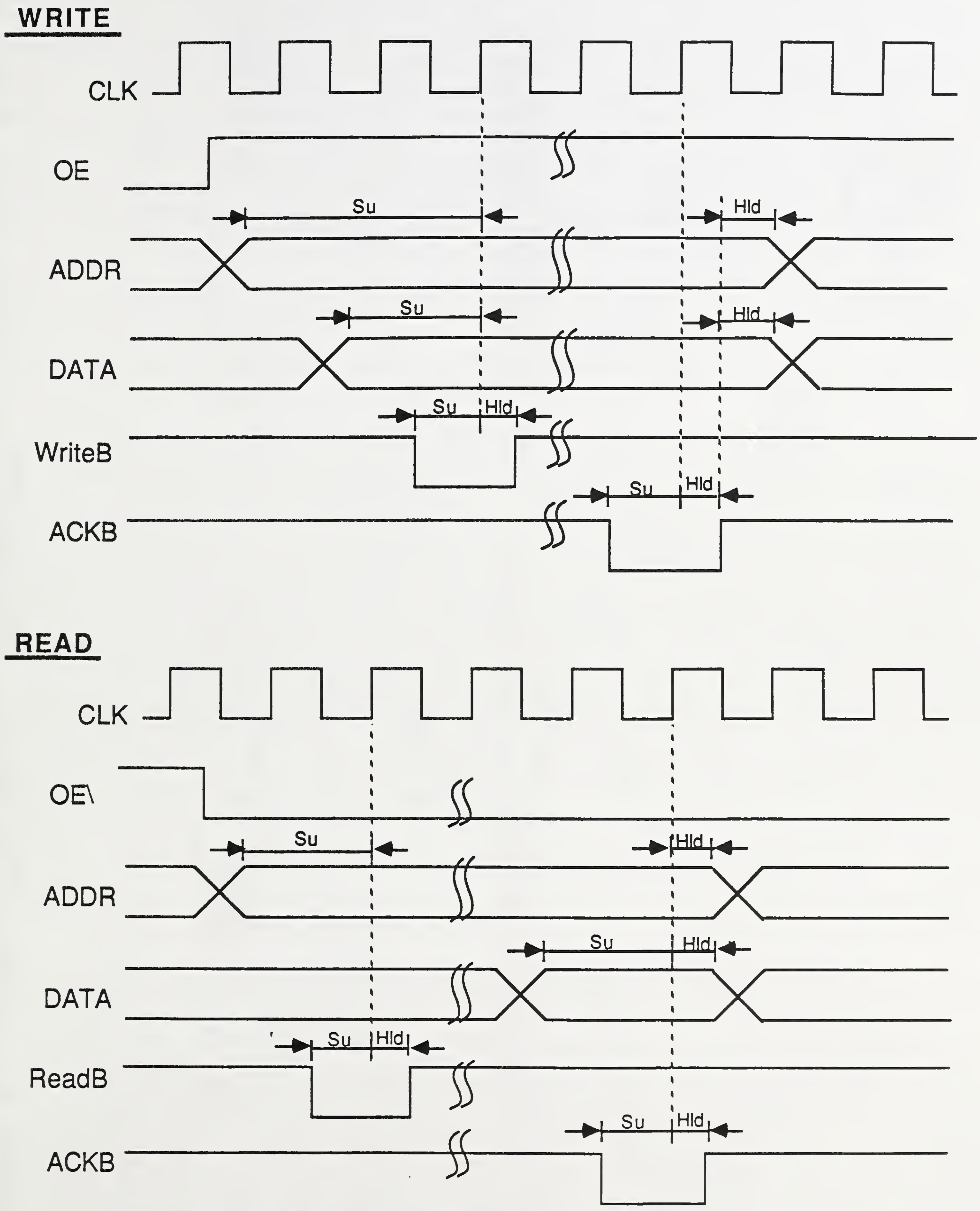

Figure 2. MULTIKRON II I/O Cycle

Tu	10 ns minimum,
Tsu	6 ns minimum
Th	approx 2 ns minimum
Tfsu	10 ns minimum
Tfh	Th minimum

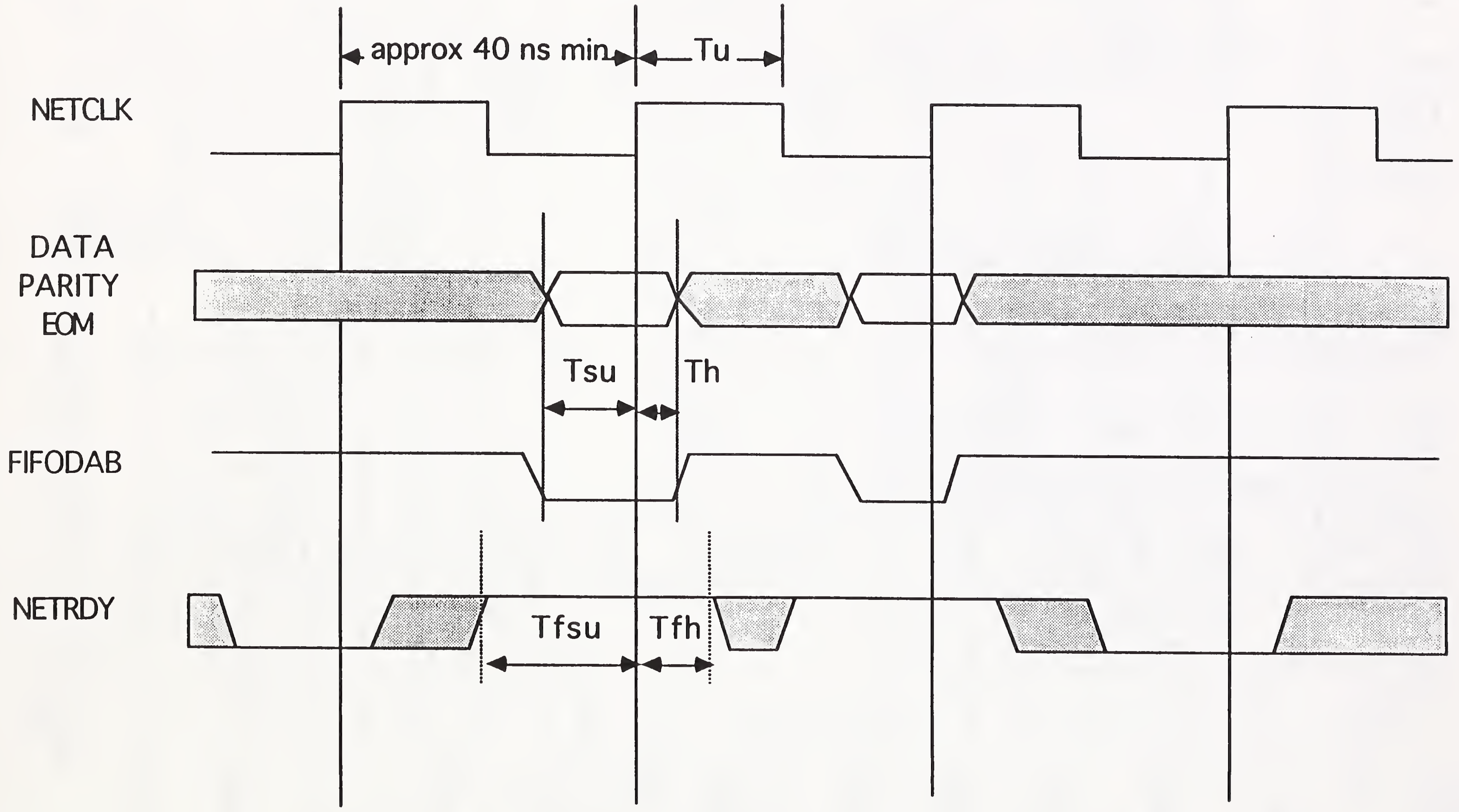

Figure 3. MULTIKRON II NETWORK INTERFACE TIMING