To the Graduate Council:

I am submitting herewith a thesis written by Russell Briscoe Rochelle entitled "A Programmable High-Speed Data Acquisition Controller." I have examined the final copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Electrical Engineering.

Donald W. Bouldin

Donald W. Bouldin, Major Professor

We have read this thesis
and recommend its acceptance:

Robert Bozenheimer

M. J. Robert

Accepted for the Council:

Vice Provost
and Dean of The Graduate School

STATEMENT OF PERMISSION TO USE

In presenting this thesis in partial fulfillment of the requirements for a Master's degree at The University of Tennessee, Knoxville, I agree that the Library shall make it available to borrowers under rules of the Library. Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of the source is made.

Permission for extensive quotation from or reproduction of this thesis may be granted by my major professor, or in his absence, by the Head of Interlibrary Services when, in the opinion of either, the proposed use of the material is for scholarly purposes. Any copying or use of the material in this thesis for financial gain shall not be allowed without my written permission.

Signature Russell B. Rochell

Date 4/6/89

A PROGRAMMABLE HIGH-SPEED DATA ACQUISITION CONTROLLER

A Thesis Presented for the Master of Science Degree

The University of Tennessee, Knoxville

Russell Briscoe Rochelle

May 1989

ACKNOWLEDGMENTS

The subject of this thesis was included in a large data acquisition and control system developed by a team of engineers at Oak Ridge National Laboratory, Oak Ridge, Tennessee. I wish to endow J. M. Jansen, Jr., of Oak Ridge National Laboratory with my greatest appreciation for his superb guidance and leadership in the design and implementation of this project. Further gratitude is expressed to D. A. Clayton for his technical assistance, R. G. Maples for his mechanical design expertise, and all other participants in this systems integration.

I thank the members of my committee, Drs. R. E. Bodenheimer and M. J. Roberts, and especially my major advisor, Dr. D. W. Bouldin, for their advice and suggestions.

Most of all I thank my wife and daughter for their patience and support during the writing of this thesis. My Father and Mother are to be commended for their raising of me towards this and higher goals.

ABSTRACT

Multiple analog-to-digital converters (ADCs) operating at sampling rates up to 1 MHz yield data rates that surpass the throughput bandwidths of digital computers. Long-term acquisition of these data requires specialized magnetic tape recorders for data storage and later playback and analysis.

A programmable high-speed data acquisition subsystem has been designed and implemented to provide the properly conditioned data for the high-speed tape recorders. The subsystem controls the order and rate at which 16-bit data words are read from as many as four of five ADCs, a time code generator, and an internal 64-kbyte tape header memory. The data are multiplexed, formatted, and converted into 24-bit words for synchronous parallel transfer to a high-speed digital magnetic tape recorder at rates of up to 7.3 Mbytes/s. Control words and time stamps are placed periodically in the data stream to facilitate playback synchronization and demultiplexing. A DEC PDP-11/73 minicomputer provides supervisory control. ADC initialization, operating parameters, and channel multiplexing sequence lists are transferred by the computer, using direct memory access (DMA), and multiplexed by the subsystem to a selected ADC. The subsystem provides each ADC with sampling trigger signals ranging from 10.24 to 900 kHz derived from three internal programmable crystal clock oscillators and frequency-divider circuitry.

Computer-aided testing examines the operational features of the subsystem. This software-based diagnostic program, which runs on the host, determines and calls attention to hardware malfunctions.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

In test and analysis systems it is often desirable to compare and correlate many simultaneously generated analog signals for long durations. When digitized, these signals yield generous amounts of data at high speed. Real-time signal processing of these data requires significant data processing bandwidths and input/output (I/O) throughput, and strains the capacities of conventional storage media (as well as the pocketbook).

Figure 1.1 depicts a high-level concept of the data flow desired. Phenomena of interest, for example, pressure, flow, vibration, sound, and voltage, occur concurrently. These signals are multiplexed, quantized, and stored for later repetitive analysis. A controller is required to further combine multiple digital data streams and provide synchronization markers to assist in decommutation at playback. Memory buffers the resultant high bit-rate for the slower processing rates of the analyzer.

Significant technological advances regarding conversion rates and dynamic ranges of analog-to-digital converters (ADCs), coupled with high signal-to-noise ratios and bandwidths of digital magnetic tape recorders, now permit in-depth analysis of data from a large number of sensors and transducers at data rates as high as 8 Mbytes/s. Recording of these digitized data in real time for later playback and analysis is a practical technique for multichannel signal-processing systems. The entire dynamic range of the ADC is used, and it is fully utilized when compared to direct analog recording that produces considerable noise. Up to 2 Gbytes of data are archived on tape, and time-wise repetitive passes may be made on

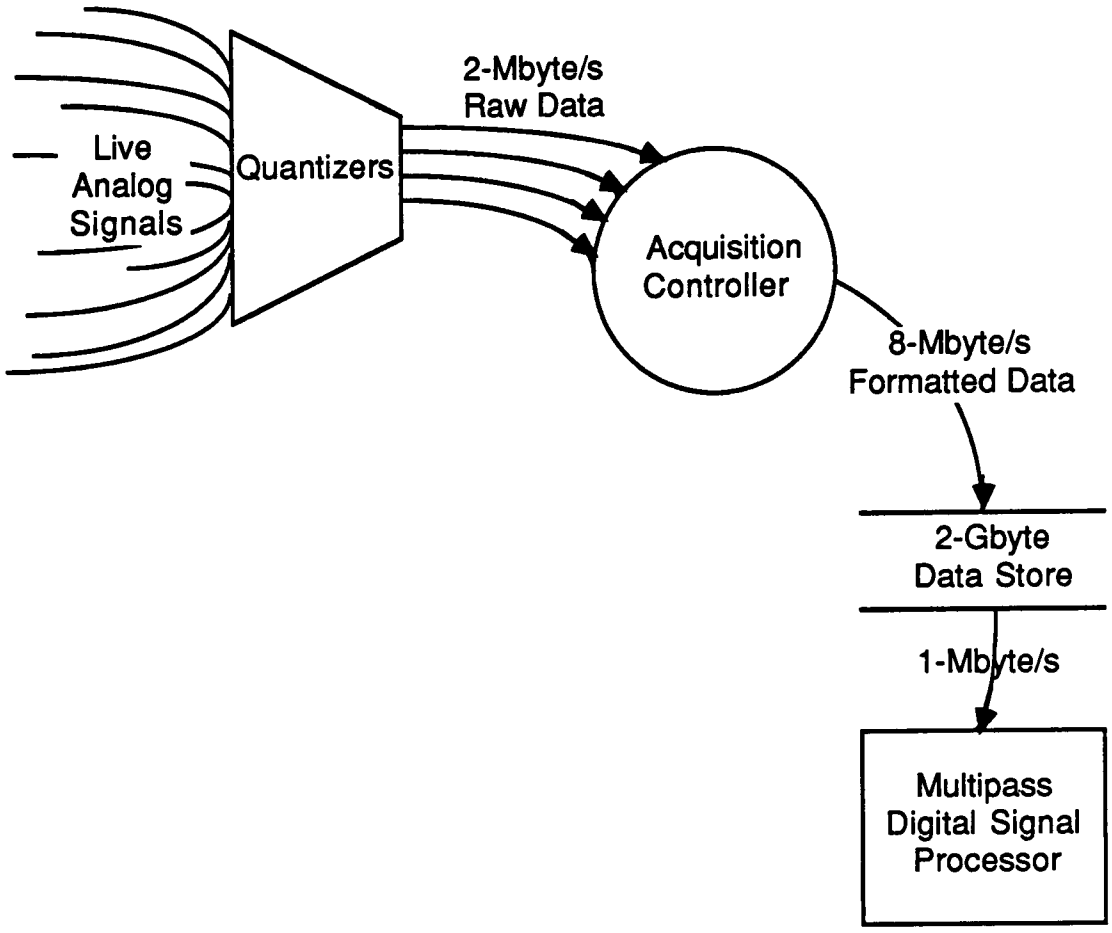

Figure 1.1. Conceptual Data Flow Diagram

the data, without noticeable tape degradation, at a reduced playback ratio that matches the I/O and processing bandwidths of the computer and peripheral equipment.

Analog multiplexers built into commercially available multichannel ADCs allow as many as 128 channels to be combined into a single data stream. This capability far surpasses that of commonly available 28-track analog magnetic tape recorders that directly record one signal per track. Additionally, data from multiple ADCs may be interleaved to offer even

greater channel quantities and thus increase the aggregate data transfer rate.

Test and analysis systems often contain signals generated with various information bandwidths. With the advent of high-speed ADCs, an assortment of bandlimited analog signals is multiplexed, sampled, and converted into 15-bit digital words. High-bandwidth signals are time-multiplexed with low bandwidth signals. The digitized signals are then stored in sequence on a digital magnetic tape recorder.

By digitizing the signals prior to recording, the recorder bandwidth can be optimized at a maximum level. The designer need not consider the phase and amplitude response of the recorder, only that of the analog portion. The 40-dB signal-to-noise ratio (SNR) typical of analog tape and recorder electronics is replaced by a 90-dB SNR and bit error rates on the order of 10^{-8} with digital recording techniques. High-bandwidth signals are time-multiplexed with low-bandwidth signals on the same tracks. The aggregate sampling rate must not exceed the maximum bit-rate of the recorder.

Multiplexing sequences can be formulated that permit multiple ADCs to operate at a constant overall sampling rate while sampling individual channels at varying rates.[1] The constant conversion rate of the ADCs provides a constant data transfer rate that is written synchronously to a digital tape recorder. The tape speed of the recorder is proportional to the synchronous transfer of 24-bit wide parallel data, and is therefore

---

[1]Eason, R. O. Development and Comparison of Algorithms for Generating a Scan Sequence for a Random Access Scanner, ORNL/TM-7504. Oak Ridge: Oak Ridge National Laboratory, 1980.

controlled by the aggregate of the ADC sampling rates. The data rate that ensues must be less than the maximum bit-rate of the recorder.

This thesis describes a subsystem that enables digitized data from up to four of five ADCs to be written synchronously to a 28-bit digital magnetic tape recorder. Time stamps from an externally connected time code generator and control words generated internally are periodically placed in the data stream by the subsystem to aid in synchronization and to confirm the integrity of the data upon reproduction. A DEC PDP-11/73 minicomputer provides supervisory control.

Section 2 is an overview of the subsystem and includes block diagrams. Section 3 details the interface and internal circuitry. Section 4 discusses command and control information programmed by the host computer. Section 5 describes software for computer-aided testing and diagnostics that may also be of interest as a programming guide or example. Finally, Section 6 summarizes the project and suggests some improvements and extensions.

2. SUBSYSTEM OVERVIEW

The subsystem connects to five ADCs, as shown in the block diagram of Figure 2.1, through separate but similar interfaces. Single digitized 16-bit sign extended words are read sequentially from up to four of these ADCs and written synchronously to the high bit-rate recorder at two-thirds of the aggregate sampling rate. To aid in playback synchronization and decommutation of the multiplexed data, markers or control words are placed in the data stream. These control words format the data into blocks whose sizes are the least common multiple of all the ADC scan table sizes for a particular data run. When detected during playback, they define scan table boundaries and uniquely tag each data block.

The PDP-11/73 computer configures the data acquisition controller for ADC sampling frequency, ADC data read sequence, data block length, control word spacing, tape header memory contents, and various operating modes. The data acquisition controller, as depicted in block diagrams on Figure 2.2, controls the order and rate at which 16-bit data words from the five ADCs, the time code generator/translator (TCG), and the 64-kbyte tape header memory are multiplexed and converted into 24-bit words and written synchronously to the high bit-rate recorder (HBRR).

A block diagram of the internal clock generators and data sequencer are shown in Figure 2.3. The ADC sample trigger rate and tape data clock are derived, under software control, from one of three internal crystal clock oscillators/frequency-divider networks (10.485760, 14.680064, and

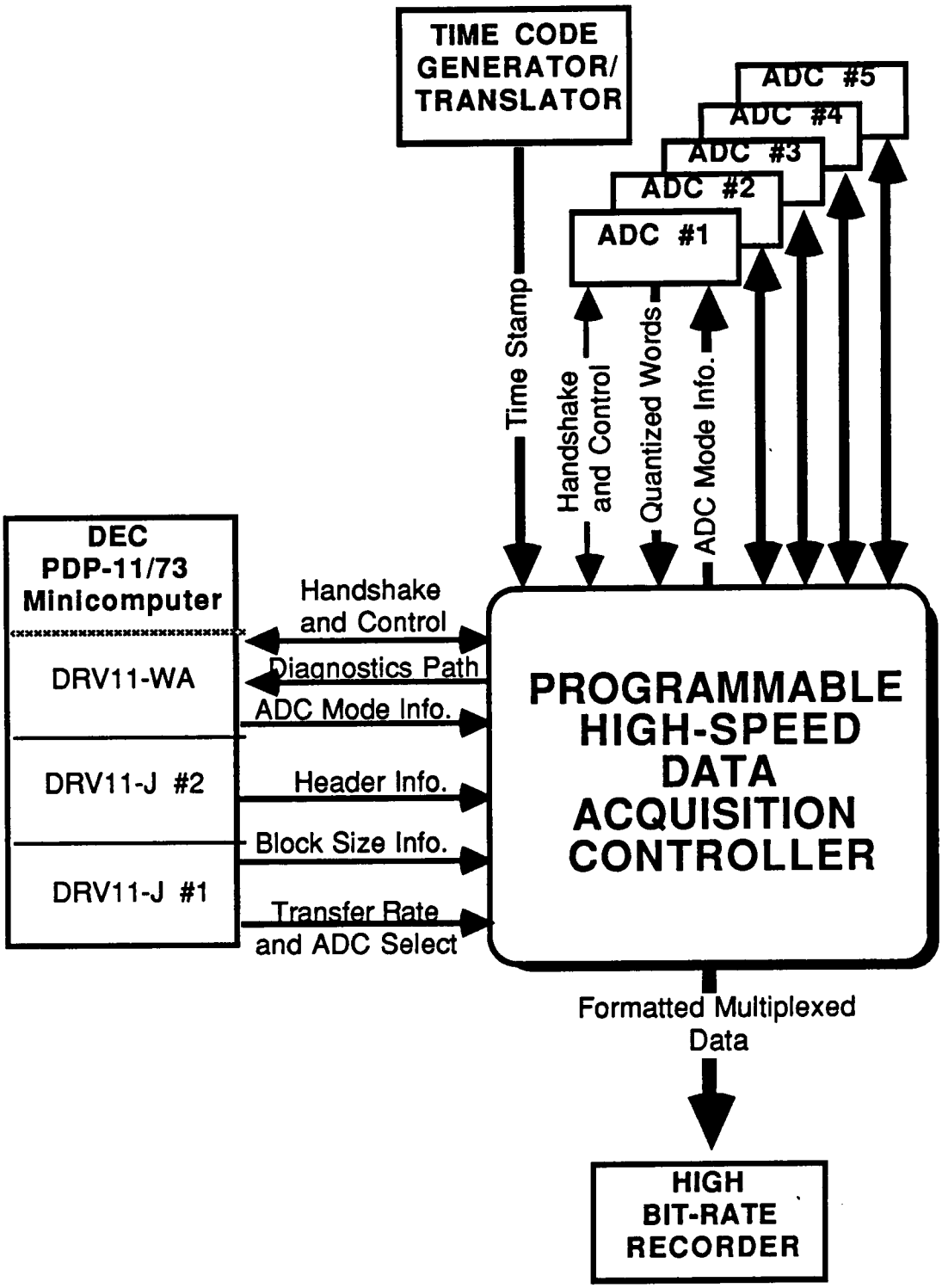

Figure 2.1. System Block Diagram

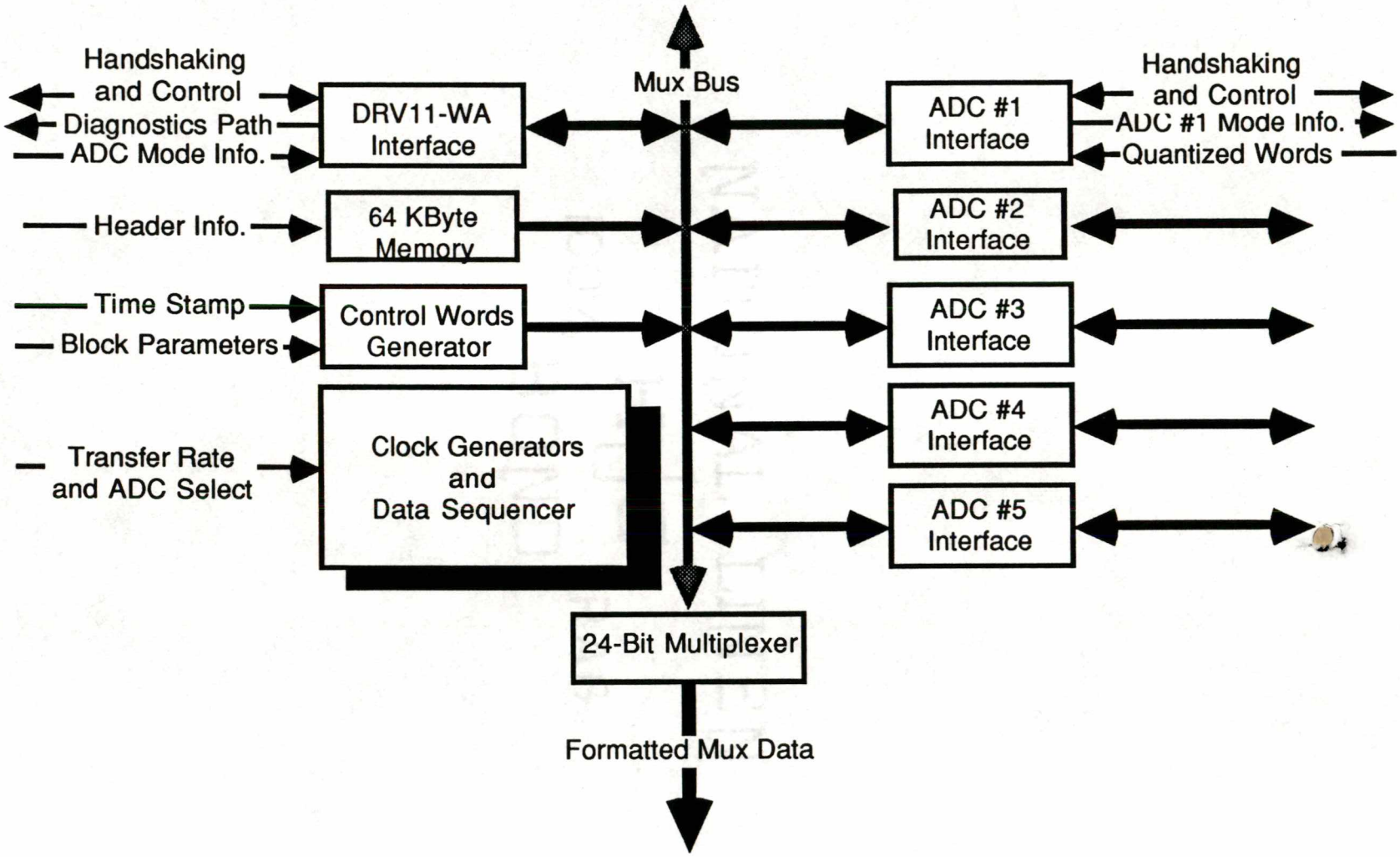

Figure 2.2. Controller Block Diagram

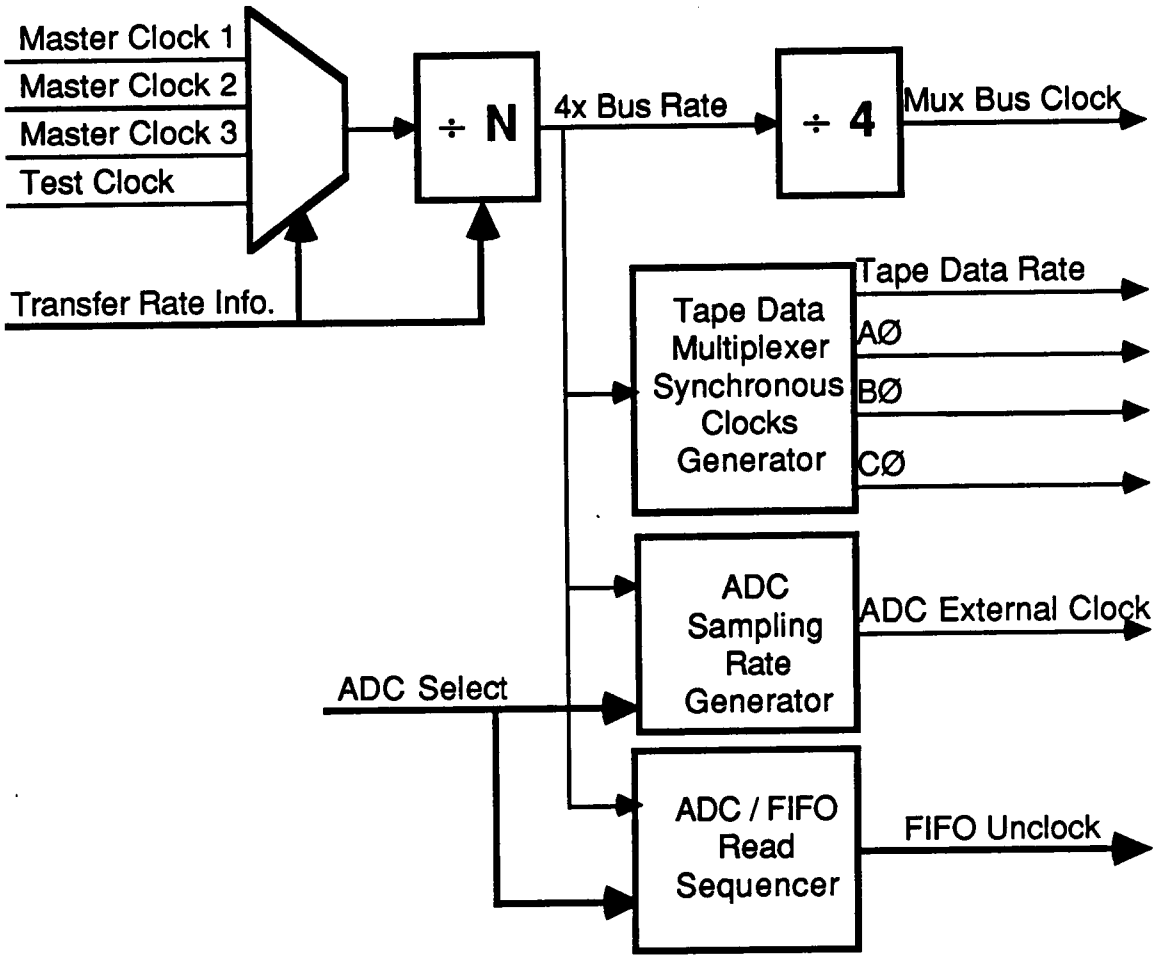

Figure 2.3. Clock Generators and Data Sequencer

25.165824 MHz). This selected master clock is inherently divided by four. A programmed 8-bit divisor (1 to 256) then determines the 16-bit word record rate. Further division by the number of ADCs to be read (1 to 4) yields each ADC sampling frequency. This provision allows a maximum individual ADC sample rate of 917.504 kHz, a minimum of 10.240 kHz (with one ADC and 2.56 kHz with each of four ADCs), and a maximum aggregate sample rate of 3.670016 MHz using four ADCs. Two-thirds of the 16-bit word rate is the tape data clock rate, since two-thirds of 24 bits is 16 bits.

The 64-kbyte tape header memory, depicted in block diagram form in Figure 2.4, contains run-specific data originating from the PDP-11/73 computer and formulated by the user. Upon computer command, the entire contents of the memory are written to tape as the header. Gap lengths between subsequent recordings of the memory or data are host software-controlled, during which zeroes are written to tape by the controller.

ADC initialization, setup parameters, scan tables, and run mode data are transferred via DMA from the computer and multiplexed by the data acquisition controller to a selected ADC. A block diagram of the host computer DRV11-WA DMA interface is shown in Figure 2.5. An ADC interface is shown in Figure 2.6. When commanded by the computer, the data acquisition controller provides the ADCs with a sampling clock and a start signal to begin their data conversion.

The controller reads the resultant ADC data words in a predesignated repetitive sequence. These words are grouped into uniform data blocks and, until commanded to stop, written to tape with the format shown in Figure 2.7. The data block length is equal to the number of active ADCs multiplied by the least common multiple of the individual ADC scan table lengths. An integer block length ranging from 2 to 8192 words is programmed with the computer prior to the start of a run.

The data acquisition controller overwrites six ADC data words in each block of data with equally spaced control words by, appropriately, the control words generator shown in Figure 2.8. Control word spacing, software-controlled prior to the start of a run, is an integer ranging from 1 to 1024 words. The six control words consist of two synchronization words,

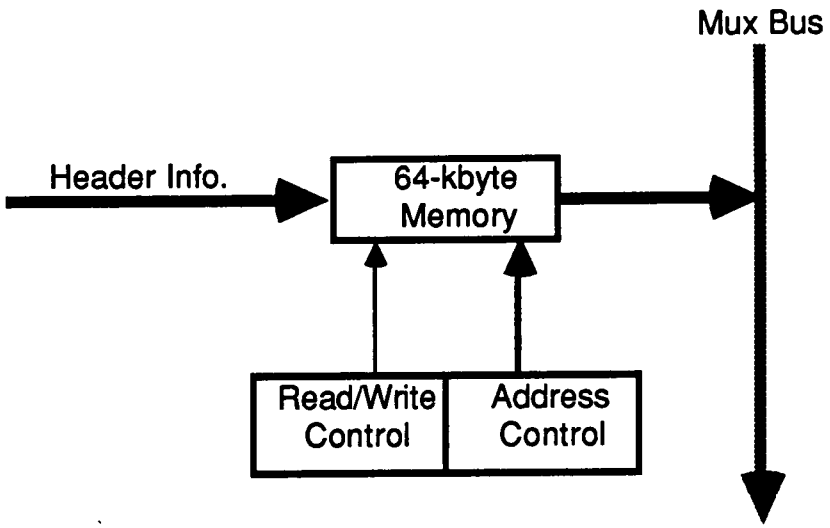

Figure 2.4. 64-kbyte Memory Block Diagram

a block counter, and three time words read from the TCG that are used for playback synchronization, data decommutation, error detection and time stamp information.

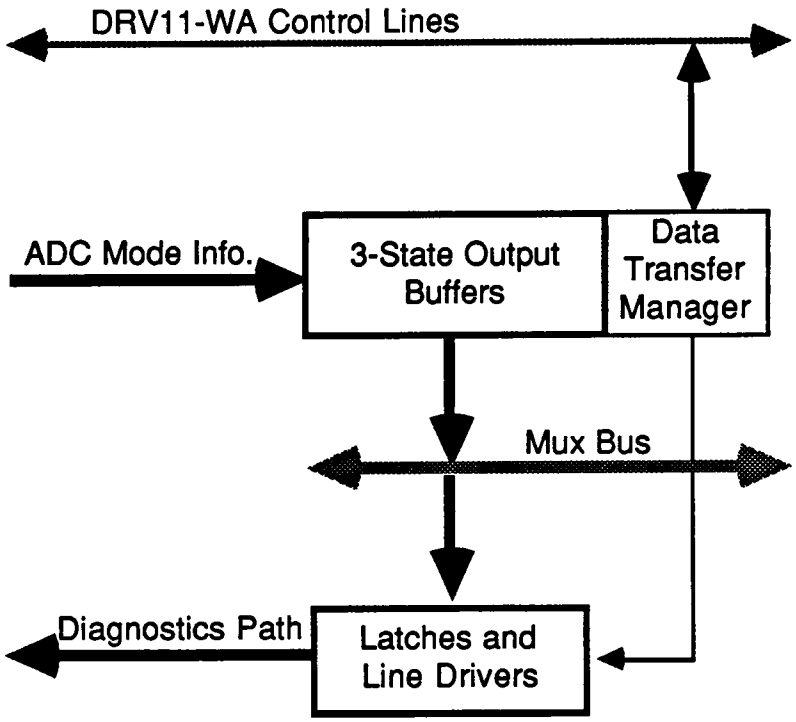

Figure 2.5. DRV11-WA Interface

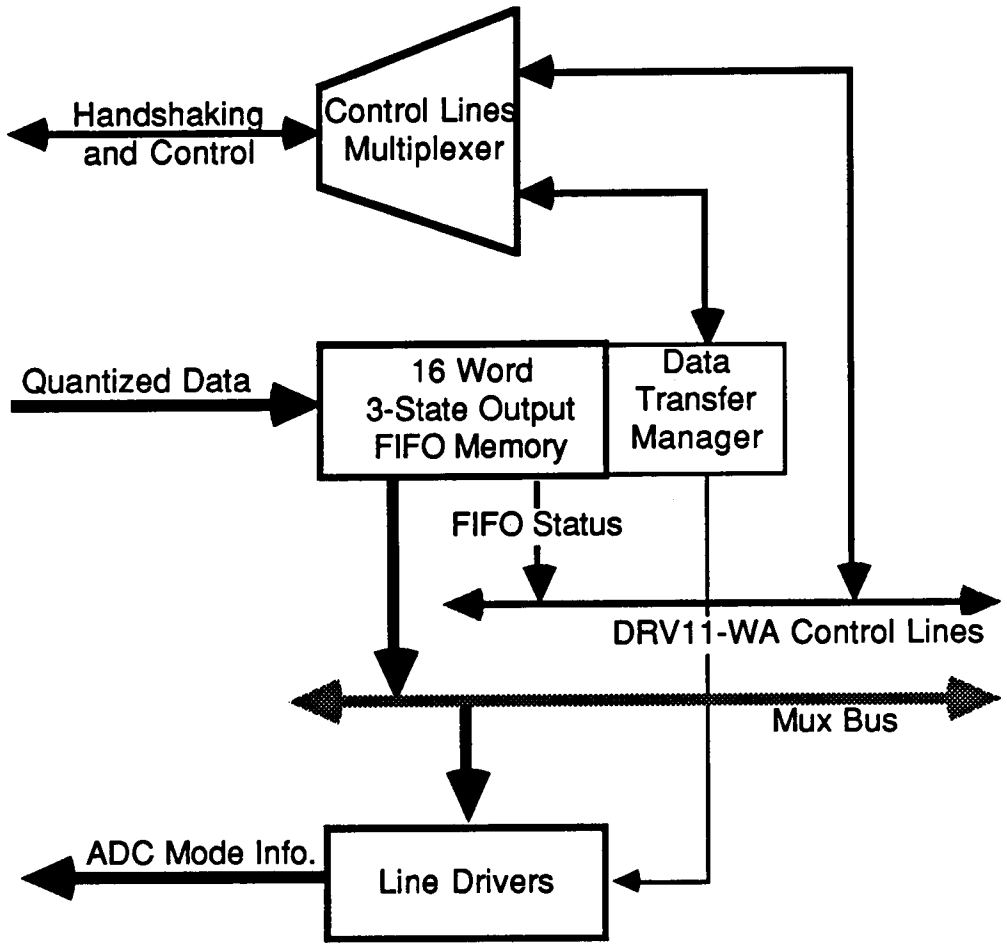

Figure 2.6. ADC Interface

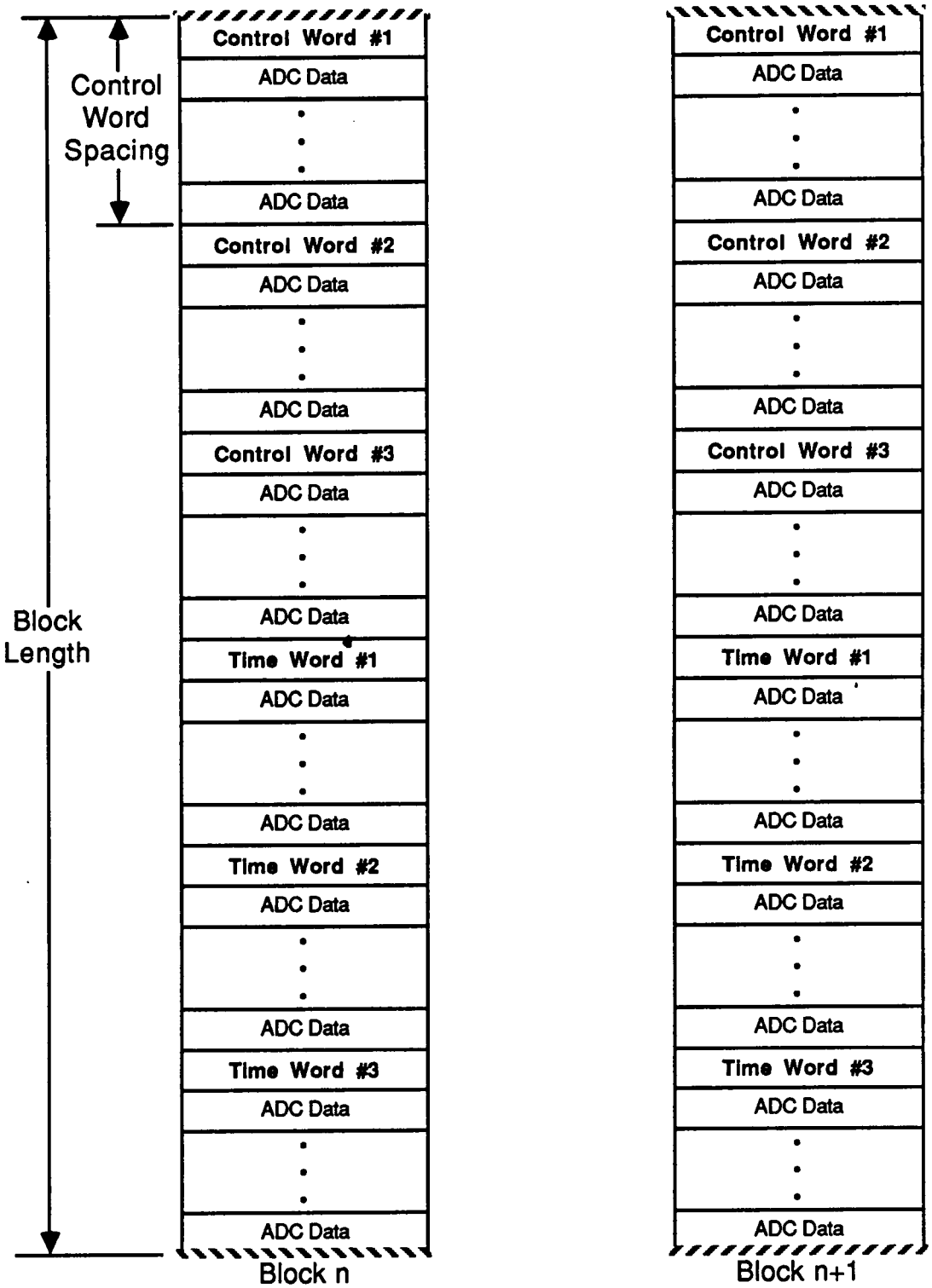

Figure 2.7. Tape Data Block Format

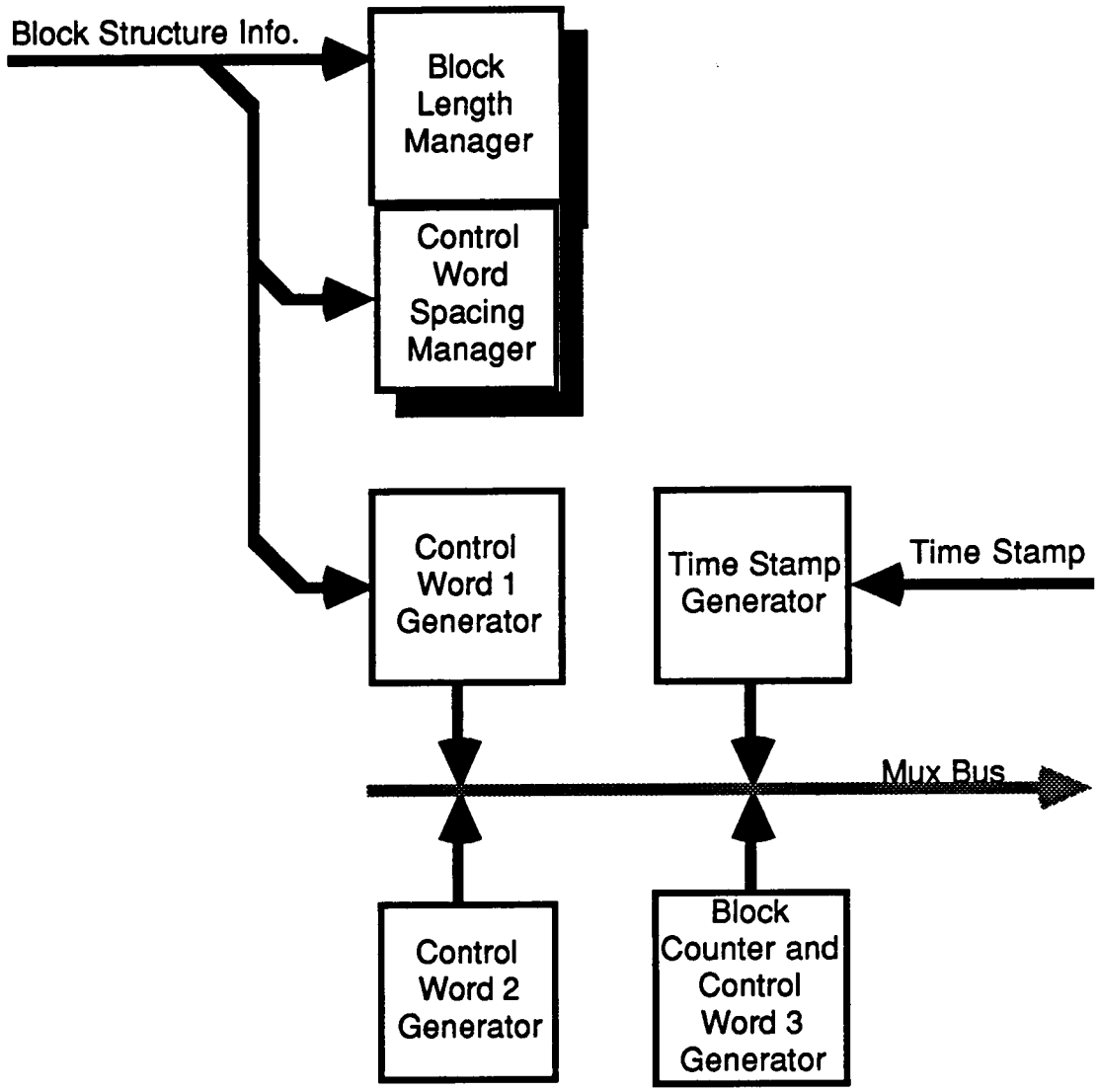

Figure 2.8. Control Words Generator

3. HARDWARE DESIGN

3.1. Electrical Interfaces

3.1.1. DEC PDP-11/73 Minicomputer

Two DRV11-J digital I/O boards and a DRV11-WA general purpose DMA board are utilized to interface the controller to the Digital Equipment Company PDP-11/73. The DRV11-J digital I/O board is composed of four 16-bit I/O ports, configured for this application as output only, with synchronous transfer. Four ports from one board and two ports from the other are used. Two ports of a DRV11-J select one of three crystal clock oscillators as the master clock, its 8-bit divisor, the number of ADCs from which to gather data, and their multiplexing sequence. Block length and control word spacing within the blocks are selected with two ports of a DRV11-J. Two more DRV11-J ports are used to select subsystem operating modes and to write the contents of the 64-kbyte memory.

A DRV11-WA general purpose DMA board is used in the minicomputer to transfer initialization and setup parameters, analog channel scan tables, and run mode data to each ADC individually via the controller system. In the data collection mode, it is used to retrieve status information, and DMA acquired and formatted data from the MUX.BUS at rates below 175 kHz or data rates less than 350 kbytes/s.

DMA transfer direction is controlled by DRV11-WA CSR function bit 1 (DRV11W.FNCT1), which is inverted and sent back to the DRV11-WA. With this function bit high, DMA out of PDP-11/73 memory is enabled, and with it low, DMA into PDP-11/73 memory is enabled.

The DRV11-WA is allowed write access to the MUX.BUS when DRV11W.FNCT1 is low and when neither recording a header or collecting data. These data are then bussed to a selected ADC in conjunction with the DRV11-WA handshaking, status, and function signals. DEC-specified protocol is used for data transfers with this device.

INTERFACE.SEL.<2:0>,[2] equal to six, enables handshaking, status, and function signals, while the data bus controls the minimum rate. DMA data transfers from the MUX.BUS are permitted at the MUX.CLK rate. While DRV11-WA.RDY(L) is asserted, the start of a header or data block asserts the first DRV11-WA CYCREQ on the leading edge of MUX.CLK. The assertion of DRV11-WA.BUSY(L) negates DRV11-WA CYCREQ. Negating DRV11-WA BUSY(L) lets the handshaking cycle repeat with the next MUX.CLK, causing DRV11-WA CYCREQ to assert.

DRV11-WA STATUS bit A is generated from a MISSEDDATA signal when either CYCREQ or BUSY(L) is asserted during a MUX.CLK period, thus showing a communication error. This error will occur when the data transfer rate, that is, the MUX.CLK, is greater than 175 kHz as the PDP-11/73 DRV11-WA cannot keep up. DMA from the MUX.BUS is generally for diagnostic purposes.

3.1.2. Five 15-bit 1-MHz ADCs

The system interfaces to five analog-to digital converters, each capable of sampling and converting at rates of up to 1 MHz. Analog multiplexers internal to the ADCs allow time division multiplexing at various rates dependent upon the sampling period, the aggregate sampling rate being

---

[2]<n:m> indicates bits "n" (more significant bit) through "m" (less significant bit).

held constant and controlled by the data acquisition controller for each data run. Each ADC is intended to interface to a single DEC DRV11-WA direct memory-access module. In this application, the DRV11-WA control and data lines are multiplexed from each of five ADCs to a single DRV11-WA module because of throughput limitations and to limit the amount of additional hardware. ADC setup information emanating from the host computer is transmitted via the DRV11-WA to the MUX.BUS. The data acquisition controller is in a transparent mode when operating in this manner. Control lines and setup data are multiplexed and passed directly to individually selected ADCs. The ADC and DRV11-WA acknowledge the data transfer through the use of DRV11-WA handshaking protocols. (Refer to the DEC DRV11-WA manual for more information.) The ADC is accessed by an address output by one of the DRV11-J general purpose digital I/O modules.

When the command to start data collection is given, quantized words from the ADCs are temporarily stored in first-in first-out (FIFO) memories until read in a round-robin fashion by the data acquisition controller for output to tape. Each converted word is latched into the FIFO while handshaking is maintained by the FIFO manager at each of the controller ADC interfaces. The DRV11-WA STATUS bits are used to check FIFO, ADC, and controller status in this mode.

Start commands and sampling clocks are buffered and output simultaneously to each ADC. Synchronous transfer to tape is used so that for each quantized sample provided by the ADC, another one will be output to tape, thereby ensuring that the long-term transfer rates in and out of the FIFOs are equal. FIFOs are aids during startup, since each ADC has a

worst case uncertainty of three sample clock cycles before output of the first quantized word. Each sample thereafter is output at the sample clock rate. In addition, upon startup a delay may be incurred after each FIFO contains at least one word of data, while the internal controller clocks are situated such that the first word written to tape will be the most significant 16 bits.

I/O data transfers and associated communications handshaking between the controller and an ADC are performed by ADC interface circuitry. Handshaking and DMA data from the PDP 11/73 are multiplexed to an accessed ADC. Converted ADC digital data are buffered by FIFO memory, whereby the controller reads the data and writes them to tape. Receive handshaking, for the later process, is handled by the controller without the use of the DRV11-WA.

There are five of these interfaces in the controller subsystem, one per ADC, all functionally equivalent. The motherboard slot into which an interface board is plugged determines its intended ADC number and equivalent address, hence geographical addressing establishes the ADC number.

One ADC is selected for a DMA of setup parameters by the host PDP-11/73 DRV11-WA general purpose DMA interface module connected to the controller DRV11-WA interface board. INTERFACE.SEL.<2:0> signals, also output by the host and bussed to all the ADC boards, designate the ADC to access. Unique address codes are hardwired at each of the ADC interface boards' motherboard connectors. Upon an address match, a comparator enables a state-machine to transmit and receive the DRV11-WA and ADC handshaking, status, and function signals. DMA data to the ADC are transferred from the MUX.BUS and buffered by line-drivers.

Additional line-drivers output a zero word while COLLECT.DATA is asserted by the host to prevent the ADC from accepting its master reset word [FFFF (hex)].

After a DMA to an ADC is complete, DRV11-WA should be programmed low. This enables the assertion of ADC.RDY(L) by the controller when the host later asserts COLLECT.DATA and begins the ADC conversion process. FNCT1 is inverted and latched when the ADC is deselected or COLLECT.DATA is asserted.

Four 4-bit FIFOs latch the ADC converted data on the leading edge of FIFO.LDCK, asserting FIFO.*.RDY.[3] FIFO.LDCK is generated from ADC's CYCREQ through a multiplexer while COLLECT.DATA is asserted. One handshake cycle is completed by inverting ADC.CYCREQ through the handshake source multiplexer and transmitting this signal back to the ADC as BUSY(L).

FIFO.*.RDY asserted then enables the open collector output DATA.RDY. DATA.RDY allows the writing of control words and ADC data to the HBRR when all FIFOs associated with the ADCs used in a data run contain at least one word of ADC data. This provision compensates for any timing uncertainties between ADCs. Negating ADC.USED neglects the associated FIFO.RDY(L) before writing data blocks and assumes that the FIFO contains data. ADC.USED should be programmed asserted on all ADC/FIFO boards associated with an active ADC and negated on all others.

FIFO.UNCK, from ADC sequencing demultiplexer, places the next available word in the FIFO output buffer. The negation of FIFO.UNCK

---

[3]* = 1, 2, 3, 4, or 5, depending on the slot into which the board is plugged.

enables this buffer to the MUX.BUS. If a control word is being enabled, the FIFO is unclocked but its output is not enabled, hence the ADC data word is discarded or overwritten by the control word. Writing the first data block to tape is prevented until the FIFO has a word of ADC converted data.

FIFO EMPTY and FULL signal levels are latched during the data conversion process whenever a FIFO is unclocked with no data or whenever it contains 16 words respectively. Normally the FIFO contains only a few words and is needed for timing uncertainties between ADCs and to allow for proper phasing of the tape clocks. These status bits, plus ADC.MISSED.DATA (generated and latched by the ADC), may be monitored in the DRV11-WA CSR (status bits B, C, and A respectively) by first accessing the ADC of interest with INTERFACE.SEL.<2:0>. They are reset with the assertion of RESET(L) or DRV11-WA FNCT2.

Additional buffer circuitry provides the ADC with its analog sampling clock and the command to start converting analog data. The ADC external clock signal is buffered for connection to each ADC. The ADC start signal is asserted on the leading edge of the clock after asserting COLLECT.DATA.

3.1.3. Time Code Generator

One of the 28 tracks of the high bit-rate recorder is used to record an IRIG (interrange instrumentation group) Format B modulated time code signal. This signal, produced by the time code generator (TCG), is used during playback analysis to locate a desired segment of tape with a resolution of 0.1 ms. Additionally, the time code generator outputs a binary coded-decimal format time code which is read once during each data block and written to tape as three control words.

Time data are held stable by the TCG when a HOLD signal from the controller is asserted at the start of a data block. The HOLD signal is then negated after the last time word is written in each block, allowing the TCG to update time values.

Time code generator outputs are tristate buffered to the MUX.BUS. These buffers are enabled by the control words generator circuitry. The HOLD signal prevents time updates until all six control words have been written.

3.1.4. High Bit-Rate Magnetic Tape Recorder

The high bit-rate recorder is a 28-track digital tape recorder. One analog track is dedicated to the modulated time code signal. Twenty-four digital tracks are used for recording output by the data acquisition controller. Another three tracks are used for parity and error correction code circuitry internal to the tape recorder in order to decrease its bit-error rate.

One and a half 16-bit words (24 bits) are synchronously output by the controller at two-thirds the aggregate ADC sampling clock rate. The synchronous clock signal is also provided by the controller. RS-422 type differential transmitters are used for both clock and data output drivers.

Sixteen bit words from the MUX.BUS are multiplexed into 24-bit words for recording by the HBRR. A tape data clock, HBRR.DATA.CLK, output to the HBRR by the controller is generated synchronously with the 24-bit data. All outputs to the HBRR are RS-422A differential signal levels.

While header data or ADC data are to be recorded, 16 bits from the MUX.BUS are latched at the MUX.CLK rate. Otherwise, these latches are

reset so that zeroes will be recorded. The data are then multiplexed, a byte at a time, filling up a 3-byte register.

All phases of the three-phase clock cycle at the tape data rate and are one-half of the MUX.CLK period out of phase with one another. Two bytes from one word and a byte from another, latched into the 3-byte register by these clocks, are then latched again at the HBRR.DATA.CLK rate, providing stable data for output to tape. Differential line drivers then convert the 24-bit single-ended TTL signals to RS-422A-compatible differential signals.

3.2. Internal Circuitry

3.2.1. Clock Generators

All data are produced and recorded in synchronization with the selected master clock. Three master clocks are provided at 10.48576, 14.680064, and 25.165824 MHz, allowing for sampling rates of each analog channel that are integer powers of 2 while scanning multiples of two, three, five and seven samples. The selected master clock is the time base for the controller's internal 16-bit data bus clock, a three-phase 24-bit tape data clock and the ADC sample trigger.

Data selectors choose one of the crystal oscillators for dividing to generate the synchronization clock, which is four times the 16-bit word record rate. Data latched from a PDP-11/73 DRV11-J address a digital multiplexer that selects one of the crystal oscillators as the master clock. A latched divisor provides the number to reload an 8-bit recycle counter. The counter outputs a half-cycle pulse at the end of each count-up cycle at the

rate of four times the 16-bit record rate. This generates all internal timing for the controller.

After the RESET(L) command is negated under program control by a DRV11-J, a synchronous 3-bit Johnson counter generates a three-phase clock from the timing clock for use by the HBRR interface circuitry. One phase is the tape data clock, providing the synchronous clock to the 24-bit tape recorder words. A 2-bit Johnson counter divides the timing clock by four to generate the 16-bit word rate clock – MUX.CLK.

A 4-bit counter divides the synchronization clock by four, and then divides it again by the number of ADCs to provide the ADC sampling trigger clock. This is the ADC sampling rate that determines the time between sequential samples within the ADC.

After a PDP-11/73 DRV11-J asserts COLLECT.DATA(H) and all FIFOs designated for use contain ADC converted data, the first 16-bit word of a data block (Control Word 1) is written to the most significant digital channels of the HBRR.

To control the order in which converted data words are read from each ADC and written to tape, ADC sequence order data is latched from the DRV11-J. This is the contents of a 4-element by 3-bit memory (SCAN.ADDR0.<2:0> thru SCAN.ADDR3.<2:0>) containing addresses of ADCs to be read. A 4-bit up counter addresses the memory's multiplexer to select an ADC. The designated ADC data will be enabled to the MUX.BUS and unclocked from their respective FIFOs via a one-of-eight selector.

Periodically, one of the six control words is enabled for recording in place of an ADC word. A series of counters and registers and a demultiplexer arrange data and place control words in the data stream.

The twos complement of the block length (BLK.LNGTH.<13:00>) latched from a DRV11-J is input to a 16-bit repetitive up-counter. This circuitry keeps track of the data block boundaries by counting the number of 16-bit words written to tape. Overflow of the block length counter increments sets the block count and restarts the sequencing of control words synchronous with MUX.CLK.

The latched twos complement of the control word spacing (CNTRL.WRD.SPC.<9:0>) determines the distance between successive data block control words. This spacing is maintained by the control word spacing counter until all six control words have been enabled for recording.

At the start of each data block, Control Word 1 is enabled to the 16-bit MUX.BUS for one MUX.CLK cycle. The overflow of control word spacing counter allows an up-counter to be incremented. This counter addresses a demultiplexer that enables the next control word to the MUX.BUS. When a control word is enabled, the oldest word in the FIFO of an ADC/FIFO interface is selected and unclocked, but it is not enabled to the MUX.BUS and therefore is lost. The enabling of TIME WORD 3 disables control words generation until the start of another data block, whereby the process is repeated.

Time data from the time code generator/translator are prevented from being updated during the writing of control words with the HOLD.TIME signal asserted from the beginning of a block until after TIME WORD 3 is enabled.

3.2.2. ADC Multiplex Sequence Controller

Analog signals within each ADC are time division multiplexed, quantized to 16-bits, and written to the controller. These data again are

multiplexed as each ADC FIFO is read in rotation. Rotational order is defined by the multiplex sequence controller circuitry, a 4-element by 3-bit memory programmed via a DRV11-J port. Each element contains the address of an ADC from which to gather data for a particular run. The quantity of the enabled ADCs determines the depth of the ADC sequence memory such that only one memory location is used when the data from only one ADC are collected; two locations are used when the data from two ADCs are collected, and so forth. Memory address generation, clocked at the 16-bit data bus rate, is provided by a 2-bit recycle counter, thereby pointing to the address of a FIFO containing ADC data to be read. The counter is clocked at the 16-bit data bus rate and recycles each time the quantity of ADCs has been realized.

3.2.3. Control Words Generator

To locate desired segments of data from tape during playback, control words are installed. Because tape errors and dropouts can cause loss of data (noticed upon playback), it is imperative to place some known words regularly among the digitized signals. These control words keep a check on data integrity, time stamp and date it, and can help "rebuild" it upon playback. Examining these control words upon playback provides a close watch on the integrity of the data. Control words also partition the data into regular and uniform blocks. Control Words 1 and 2 are constants for the duration of any data run. Control Word 1 may change from run to run, as it is a number derived from the block length and the stride between control words. Control Word 2 is a hardwired constant often referred to as the Clayton Constant in honor of its originator. Control Word 3 – a Mod 2^{14} counter – is reset to zero at the start of each run and subsequently

incremented each data block. Control Words 4, 5, and 6 are composed of decoded time read from the TCG in BCD format.

3.2.4. 64-kbyte Header Memory

To keep an entire run self-contained on a tape, all descriptive conditions concerning the run are in the 64-kbyte header memory, the contents of which are determined by the user and downloaded via a dedicated DRV11-J port. The memory is sequentially dumped to the recorder prior to ADC data collection and upon command of the host. Upon playback, the header is read and deciphered for proper setup of the playback system.

This header memory contains run-specific data and control words for playback synchronization. It is configured as 32768 x 16-bit words written by the PDP-11/73 through a DRV11-J interface. The entire contents are dumped to the MUX.BUS at the MUX.CLK rate with a computer command.

Computer control signals are also input to this board to select modes of operation and an ADC for DRV11-WA communications. Power-on reset is generated here to set the controller in a known quiescent state. Outputs of the mode latch are reset on power-up. Further latching is then controlled by the DRV11-J on the leading edge of DRV11-J RPLY(L).

RESET(L) stops all internal clocking and resets all interfaces of the controller. RESET.ADDR(L) clears the address pointer of the 64-kbyte memory to point to its first memory location. COLLECT.DATA starts the ADC analog conversion process and its subsequent recording.

MUX.DATA.SYNC(L) controls whether synchronization with the start of a data block or header occurs when reading data from the MUX.BUS through the DRV11-WA interface. When asserted, the first word DMA'ed

will be the start of a data block or header depending on which occurs after the DRV11-WA.READY(L) is asserted. When MUX.DATA.SYNC(L) is negated, whatever is on the bus will be output. All DMA to the DRV11-WA is at the MUX.CLK rate.

INTERFACE.SEL.<2:0> selects an interface for DRV11-WA access. An address of 1, 2, 3, 4, or 5 selects ADC 1, 2, 3, 4, or 5 respectively. An address of 6 selects the MUX.BUS.

Setting the direction bit to output in the DRV11-J CSR associated with memory control and then asserting and negating RESET.ADDR(L) addresses the first header memory location. Subsequently, each output data transfer of the DRV11-J writes to the addressed location on assertion of DRV11-J.RPLY(L). The negation of DRV11-J.RPLY(L) incrementally sets a 15-bit address counter to the next consecutive memory location.

Resetting the direction bit for input in the DRV11-J CSR places the header memory in a read mode for data transfer to tape, not to the DRV11-J. Asserting and negating RESET.ADDR(L) under program control, followed by proper phasing of the MUX.CLK with the tape data clock managed by the controller, begins an entire read of the 64-kbyte header memory at the MUX.CLK rate. Memory data are enabled to the MUX.BUS while reading. After 32768 words have been written or read, the address counter overflows, disabling the counter and placing the memory chips into a low-power mode.

4. OPERATIONAL DESCRIPTION AND PROGRAMMING CONSIDERATIONS

The controller internal registers are initialized and configured just prior to a data acquisition exercise. ADC setup parameters are then programmed. At the desired commencement, the tape recorder is started and enough time allowed for it to attain operating speed, which is synchronized with the controller transfer clock. Commands are then given by the host to the controller to record the 64-kbyte memory contents and formatted data, after which the host computer need only monitor controller and ADC status until the end of the run to determine the occurrence of errors. This section describes the programmable parameters in the general order in which they are programmed.

A zero word written out DRV11-J 2 port C I/O <6:0> resets the controller. Bit significance is shown in Figure 4.1, and a description of port functions is given in Table 4.1. This zero word halts all data transfers with external equipment and stops internal clocking.

One of the three crystal oscillators is chosen by 4 clock select signals, CLK.SEL<3:0>, shown in Table 4.2. Its twos complement 8-bit divisor, CLK.DIV<7:0>, is written out of DRV11-J 1 port A I/O <15:8,3:0>, which controls the clock generators. Figure 4.2 shows the bit assignments and Table 4.3 describes their function.

The sequential order in which the ADCs will be written to tape, SCAN.ADDR<3:0>.<2:0>, and the twos complement of the quantity of these

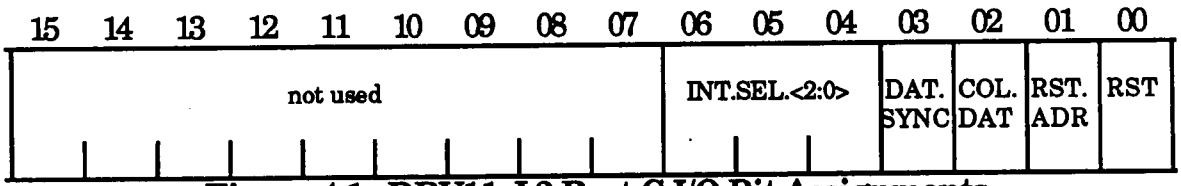

Figure 4.1. DRV11-J 2 Port C I/O Bit Assignments

Table 4.1. DRV11-J 2 Port C I/O Bit Functions and Descriptions

MNEMONIC	DESCRIPTION
RESET (L)	RESET. This active low-level bit controls the power-up state of the controller. A low level stops all internal data clocks and the HBRR data clock and disables DRV11-WA communication. A high level synchronizes and activates these clocks and enables DRV11-WA communication.
RESET.ADDR (L)	RESET 32K MEMORY ADDRESS. This active low-level bit resets the 64-kbyte memory address counter to zero. A high level enables the address to be incrementally set after writing or recording a word of memory data.
COLLECT.DATA	COLLECT ADC CONVERTED DATA (H). This active high-level bit is the ADC EXTERNAL START command. It directs all selected ADCs to begin the analog-to-digital conversion process whereby digital data may then be written to the HBRR.
DATA.SYNC (L)	SYNC W/HEADER OR DATA BLOCK. This active-low level allows data written to the DRV11-WA to be synchronized with the start of a header or data block, depending upon which is to be recorded after the DRV11-WA RDY line is asserted. Data transfer is synchronous with the MUX.CLK (16-bit word rate).
IFACE.SEL.<2:0>	ADC/DRV11-WA INTERFACE SELECT. These 3 bits are used for multiplexing DRV11-WA I/O. They determine which ADC interface is accessible for programming or status monitoring by the DRV11-WA. A value of 1, 2, 3, 4, or 5 selects ADC 1, 2, 3, 4, or 5 respectively. If a value of 6 is entered, then data may be read from the MUX.BUS at the MUX.CLK rate.

Table 4.2. ADC Read Sequences

ADC.QUANT<1:0>	READ ORDER	ADC NO. SELECTED
1	1	contents of SCAN.ADDR3.<2:0>
2	1	" SCAN.ADDR2.<2:0>
	2	" SCAN.ADDR3.<2:0>
3	1	" SCAN.ADDR1.<2:0>
	2	" SCAN.ADDR2.<2:0>
	3	" SCAN.ADDR3.<2:0>
4	1	" SCAN.ADDR0.<2:0>
	2	" SCAN.ADDR1.<2:0>
	3	" SCAN.ADDR2.<2:0>
	4	" SCAN.ADDR3.<2:0>

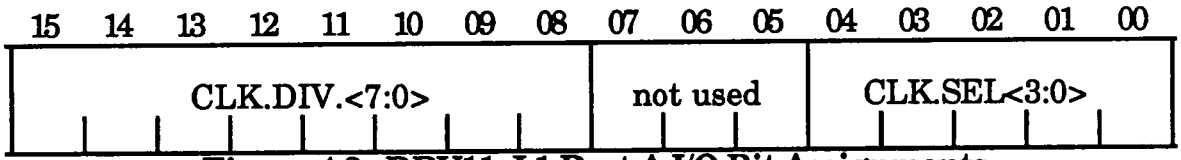

Figure 4.2. DRV11-J 1 Port A I/O Bit Assignments

Table 4.3. DRV11-J 1 Port A I/O Bit Functions and Descriptions

MNEMONIC	DESCRIPTION
CLK.SEL<3:0>	MASTER CLOCK SELECT BITS. According to Table 4.3, these 5 bits choose one of three clock oscillators or a test frequency input to a test point, and whether the oscillator will be divided by one or an integer defined by CLK.DIV<7:0>. There is no provision for division by one when the 25.165824-MHz clock oscillator is selected, as that rate is too fast for the ADCs to convert.
CLK.DIV<7:0>	MASTER CLOCK DIVISOR BITS. This byte is the twos complement of the integer divisor of the selected master clock that yields four times the 16-bit word rate of data to be written to tape. Division by one is accomplished with MASTER CLOCK SELECT bits.

ADCs, ADC.QUANT<1:0>, are written out of DRV11-J 1 port B I/O <14:3,1:0>. Table 4.4 describes the port bit assignments as represented by Figure 4.3.

The twos complement of the spacing between control words in a data block, CNTRL.WRD.SPC.<9:0>, is written out of DRV11-J 1 port C I/O, as shown in Figure 4.4 and Table 4.5. The twos complement of the number of words in a data block, BLK.LNGTH.<12:0>, is written out of port D I/O <12:0>, as presented in Figure 4.5 and Table 4.6.

Writing a one to RESET (L), DRV11-J 2 port C I/O bit 0, starts MUX.CLK, ADC sampling clock, and HBRR data clocks and allows the ADCs to be programmed. The HBRR data clock is the synchronous 24-bit data rate and is the reference for the tape speed.

Setting the DRV11-J 2 port D CSR data direction bit to one for output allows data written out of the DRV11-J 2 port D I/O <15:0> to be written into the header memory. Figure 4.6 depicts port D and Table 4.7 is its description. The memory address counter is then reset to zero by taking RESET.ADDR (L) at DRV11-J 2 port C I/O bit 1 from a low level to a high level. Each word written into this memory incrementally sets its address counter. When all 32768 words have been written successively, the controller ignores any further writes to memory unless the write process is repeated.

Record status of the header memory may be monitored with DRV11-WA CSR STATUS B and INTERFACE.SEL.<2:0> set to 110 (binary), as described in Table 4.8. A zero indicates it is in the process of being either read or written. A one indicates memory is idle and not being accessed.

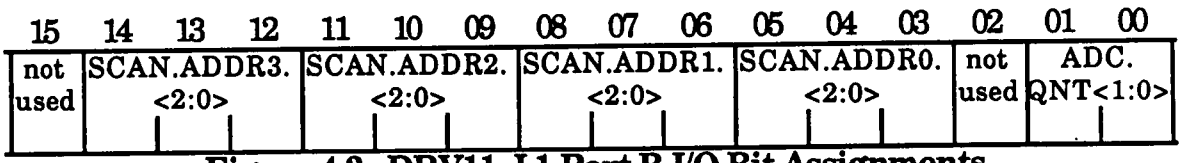

Figure 4.3. DRV11-J 1 Port B I/O Bit Assignments

Table 4.4. DRV11-J 1 Port B I/O Bit Functions and Descriptions

MNEMONIC	DESCRIPTION
ADC.QUANT<1:0>	ADC QUANTITY. These 2 bits signify the twos complement of the number of ADCs to be used for the present run or mode. A maximum of four ADCs may be specified.
SCAN.ADR<3:0>.<2:0>	ADC READ SEQUENCE TABLE. These 12 bits are organized as four 3-bit words. Each word contains the address of an ADC (1, 2, 3, 4, or 5 for ADC 1, 2, 3, 4, or 5 respectively) in the desired scan order or sequence. The scan order is repeated until the end of a run and is determined by Table 1.

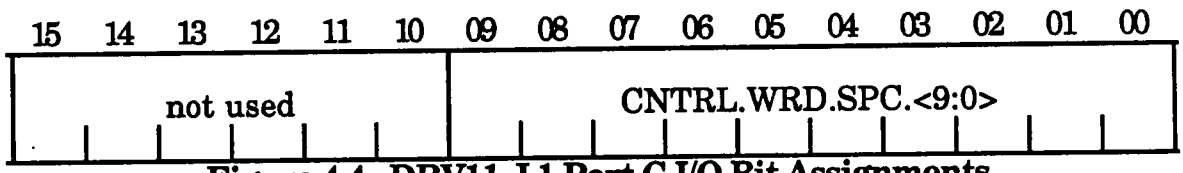

Figure 4.4. DRV11-J 1 Port C I/O Bit Assignments

Table 4.5. DRV11-J 1 Port C I/O Bit Functions and Descriptions

MNEMONIC	DESCRIPTION
CNTRL.WRD.SPC.<9:0>	CONTROL WORD SPACING. These 10 bits indicate the twos complement of the number of 16-bit word to record until the next control word is recorded. The six control words are recorded only once per block. Control Word 1 is the first word recorded in each block. Twos complement of 1, for CONTROL WORD SPACING, records all six control words sequentially with no other words between.

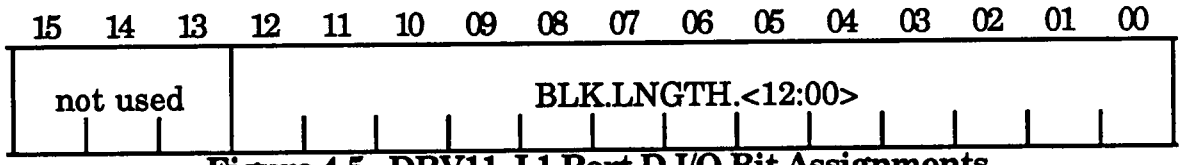

Figure 4.5. DRV11-J 1 Port D I/O Bit Assignments

Table 4.6. DRV11-J 1 Port D I/O Bit Functions and Descriptions

MNEMONIC	DESCRIPTION
BLK.LNGTH.<12:00>	BLOCK LENGTH. These 13 bits indicate the twos complement of the number of 16-bit words in a data block that are written to tape.

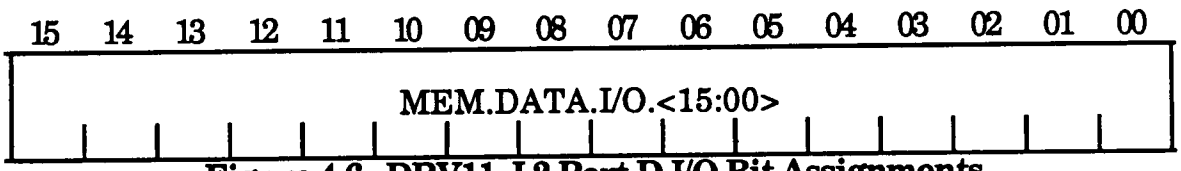

Figure 4.6. DRV11-J 2 Port D I/O Bit Assignments

Table 4.7. DRV11-J 2 Port D I/O Bit Functions and Descriptions

MNEMONIC	DESCRIPTION
MEM.DATA.I/0.<15:00>	64K MEMORY INPUT/OUTPUT DATA. These 16 bits are the data written to the 64-kbyte memory when the DRV11-J is programmed for output data transfer and RESET.ADDR (L) is high. The entire 64-kbytes of memory is output to tape at the MUX.CLK rate each time the DRV11-J is programmed for input data transfer and RESET.ADDR (L) moves from low to high. Words are read from or written to memory sequentially as each memory access increases the memory address by one.
DRV11J.2D.RDY	DRV11-J 2 PORT D DATA DIRECTION. This control line reflects the state of the DRV11-J CSR data direction bit and signals that data will be written to the 64-kbyte memory or read from it and written to tape.

Table 4.8. DRV11-WA Bit Functions and Descriptions

MNEMONIC	DESCRIPTION
DRV11W.IN.<15:00>	DRV11-WA INPUT DATA. 16 bits of data are input to the DRV11-WA at the MUX.CLK rate when INTERFACE.SEL.<2:0> equals six (110 binary).
DRV11W.FNCT1	DRV11-WA FUNCTION 1. This bit, written to the DRV11-WA CSR, determines the DRV11-WA data direction as it is inverted in the controller and sent back to the DRV11-WA as C1. High is for write, low is for read. This bit is also latched, after programming an ADC, for use when COLLECT.DATA is asserted. A low level means data will be read from the associated FIFO.
DRV11W.FNCT2	DRV11-WA FUNCTION 2. This active high bit written to the DRV11-WA CSR resets the control signals of the DRV11-WA interface accessed with INTERFACE.SEL.<2:0>.
DRV11W.STAT.A	DRV11-WA STATUS A. This bit, read in the DRV11-WA CSR, is the selected ports' MISSED.DATA signal. A high level signifies a communication error in that data were lost.
DRV11W.STAT.B	DRV11-WA STATUS B. When INTERFACE.SEL.<2:0> addresses an ADC, a high level in this bit read in the DRV11-WA CSR denotes an error condition, which signifies that the corresponding FIFO memory was empty when data were needed. When INTERFACE.SEL.<2:0> equals six, a high level in this bit signifies the 64-kbyte memory is inactive and does not accept read or write commands.
DRV11W.STAT.C	DRV11-WA STATUS C. When INTERFACE.SEL addresses an ADC, a high level in this bit read in the DRV11-WA CSR denotes an error condition which signifies the corresponding FIFO memory was full during the analog conversion process. When INTERFACE.SEL.<2:0> equals six, a high level in this bit signifies ADC-converted data are being recorded.
DRV11W.RDY	DRV11-WA READY SIGNAL. Indicates initiation of DMA to/from PDP-11/73 computer. Refer to DEC DRV11-WA manual for more information.

Table 4.8. (continued)

MNEMONIC	DESCRIPTION
DRV11W.OUT.<15:00>	DRV11-WA OUTPUT DATA. 16-bits of data are output from the DRV11-WA for use as control words by the selected ADC when it is recording neither header nor converted data.

ADC setup parameters are programmed by entering the address of the ADC to be accessed into INTERFACE.SEL.<2:0> and then transmitting the data from the computer DRV11-WA. Setting the DRV11-WA CSR FUNCTION 2 to one resets the ADC interface. Setting FUNCTION 1 indicates the DRV11-WA will send data. The reader is referred to the Preston ADC Interface Manual for more ADC programming information.

After each ADC is programmed, FUNCTION 1 must be reset to zero. This indicates to the respective ADC/FIFO interface that data will be collected from this ADC. There must be at least one word of ADC data in each of the active FIFOs before the controller will start writing data blocks to tape. Such is the case when all desired ADCs have converted at least one word of data and passed it to the controller.

When all ADCs have been programmed for a particular mode and the tape is up to speed, the 64-kbyte memory may be written to tape. The data direction bit in DRV11-J 2 port D CSR is reset to zero for input, thus indicating a memory read is desired. RESET.ADDR (L) is then programmed zero and then one, whereby the entire 64 kbytes of memory will be written once to tape.

After a 2-s gap controlled by the host, COLLECT.DATA may be programmed from a zero to a one. This starts the ADC conversion process

and begins the transfer of ADC data and control words to the HBRR. Programming a zero into the COLLECT.DATA bit stops data from being written to tape. As a check, setting INTERFACE.SEL.<2:0> to 110 (binary) and reading zero in DRV11-WA CSR Status C indicates the controller is writing data blocks to tape. Zeroes are written to tape whenever COLLECT.DATA is low and header is not being recorded.

While COLLECT.DATA is asserted, ADC status may be checked by setting INTERFACE.SEL.<2:0> to an ADC address (1, 2, 3, 4, or 5). Inspecting DRV11-WA CSR statuses A, B, and C then indicates the selected ADC missed-data, FIFO-empty, and FIFO-full signals, respectively. These latched signals should normally read zero during a data run; otherwise they indicate an error and possible loss of data.

5. TESTING AND DIAGNOSTICS

5.1. Computer-Aided Diagnostics

By computer-aided diagnostics, systems may be checked out by the host for proper operation and, in the event of a failure, the search for faulty components facilitated. This section describes a test that when passed gives the user a high degree of confidence that the controller will operate properly when integrated with the other systems.

The controller system may be tested by the PDP-11/73 computer when it executes the diagnostic software listed in the Appendix. Test setup requires specially configured connectors, constructed entirely of passive devices, that provide known digital inputs to the controller.

The software exercises the tape header memory, data collection, and sequencing capabilities of the controller. The high bit-rate recorder interface differential outputs, time code generator/translator HOLD(L), and ADC external start signals need to be tested with a logic analyzer and/or oscilloscope.

DRV11-Js configure the controller to select the master clock and its divisor, the sequence for ADC data output tape, test data for the tape header memory, and operating modes. To read the memory and data blocks, the DRV11-WA transfers data via DMA from the internal data bus. DRV11-WA DMA transfers to an ADC output data port test the handshaking facilities of the latter. Data presented with the test connectors are complemented manually with external toggle switches for stuck-bit testing.

Checks are made and errors reported to the operator on at the terminal to aid in troubleshooting.

5.2 Diagnostic Description

The diagnostic software, listed in the Appendix, assumes the DRV11-WA base address (WCR) is assigned 172410 (octal), and extended addressing is selected. DRV11-J 1 base address (CSR) is assigned 164160 (octal) and is connected to the controller clock generator board. DRV11-J 2, whose base address (CSR) is assigned 164140 (octal), is connected to the controller 64-kbyte memory board.

Special test connectors are required to simulate ADC and time code data. They allow these data to be complemented by toggling a switch on each connector. The test connectors attach to the controller ADC and time code generator interfaces to present static data.

A Nassi-Schneidermann chart of program structure is shown in Figure 5.1. After declaring constants, the program CSIDIA.SAV sets the data direction for the DRV11-J ports. The controller is reset, then test data are written to the 64-kbyte header memory whereby the three master clocks are tested by reading back with a 32-kword DRV11-WA DMA. Error messages are displayed; upon the occurrence of a timeout of the DMA, a handshaking error, data read does not match data written, or controller status bits are not correct.

Upon following the "Toggle all test connector switches towards CSI" message, the handshaking facilities of each ADC data port are checked. Data presented by the test connectors are scanned, and two data blocks are

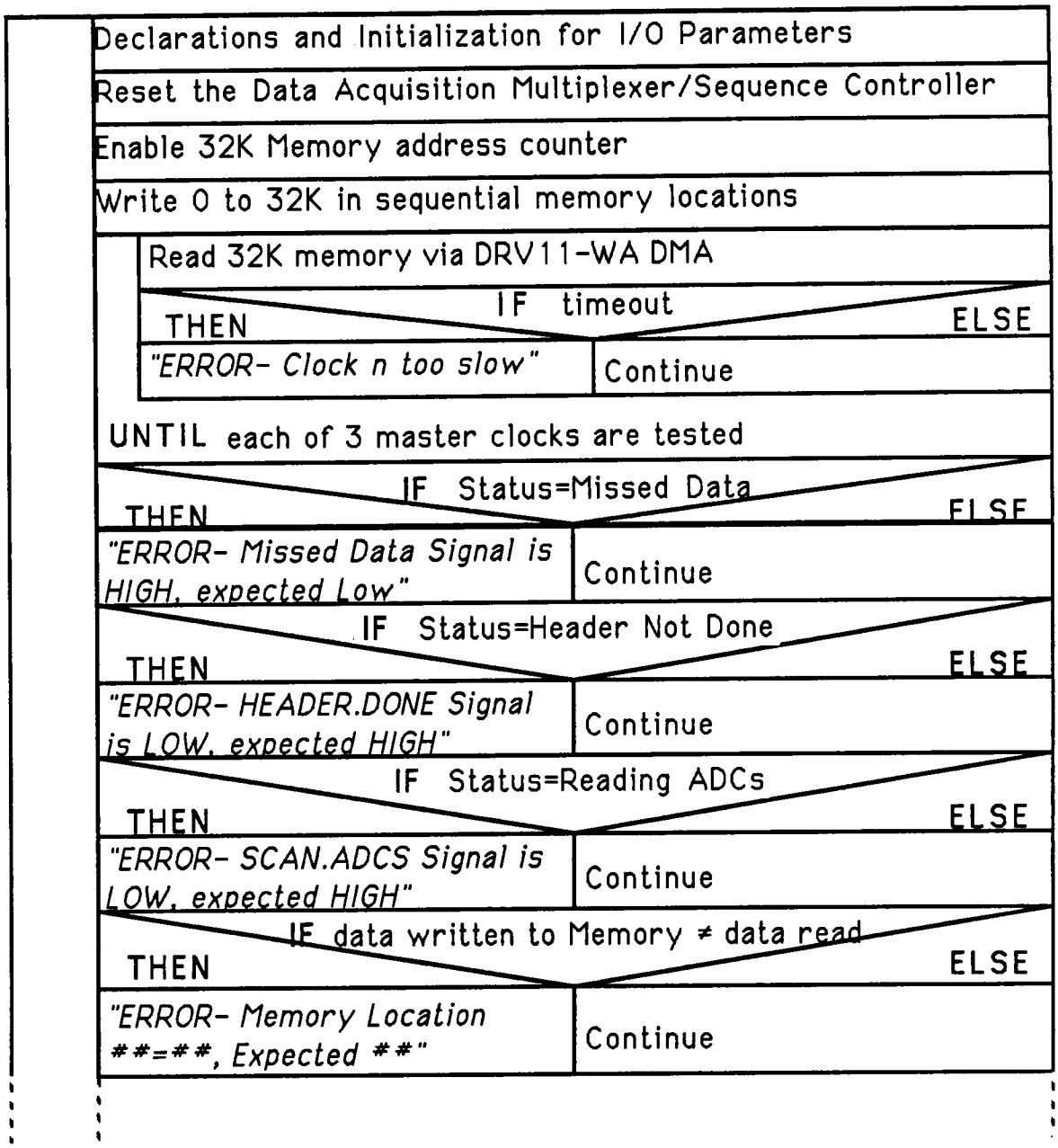

Figure 5.1. Diagnostic NS Chart

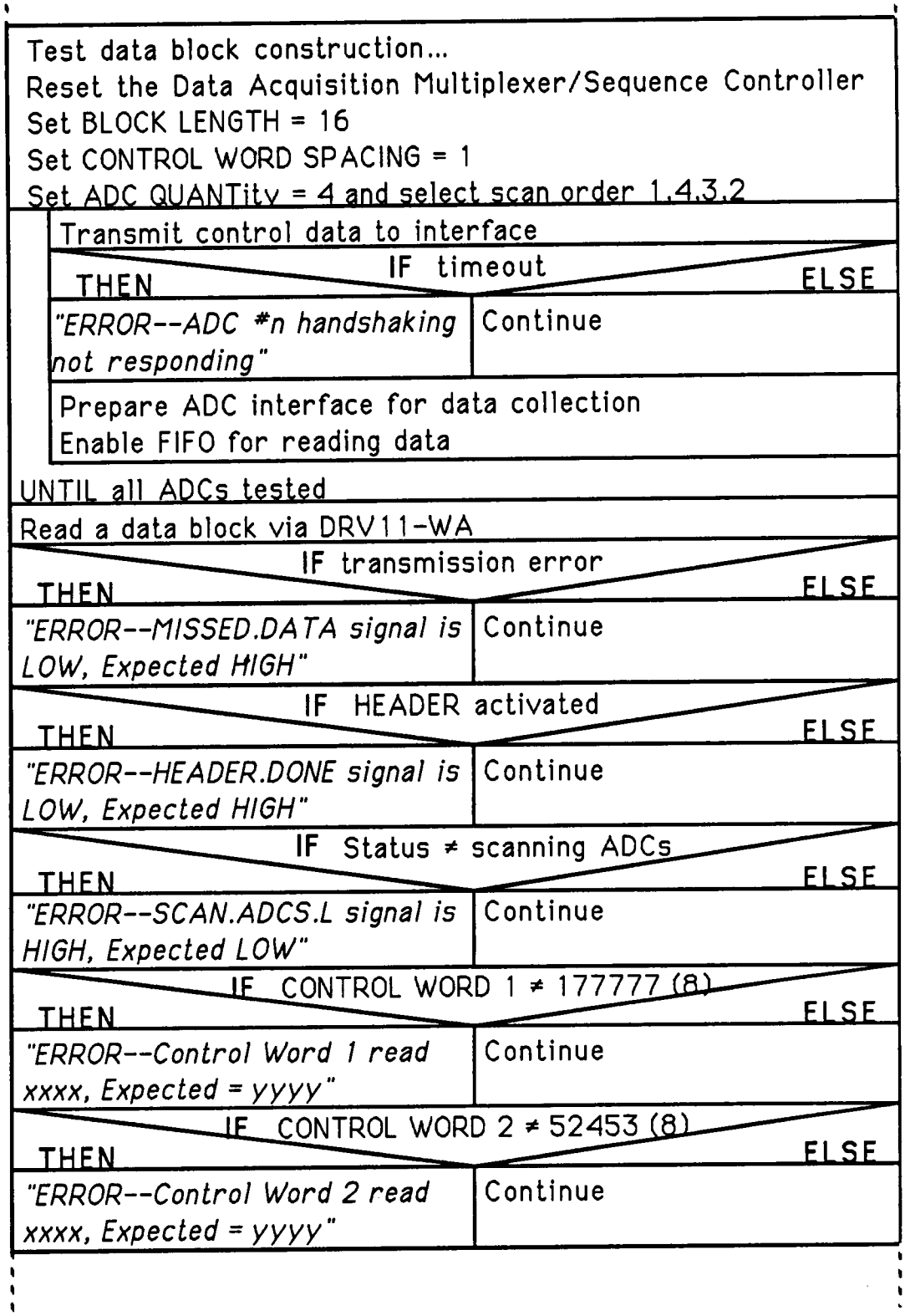

Figure 5.1 (continued)

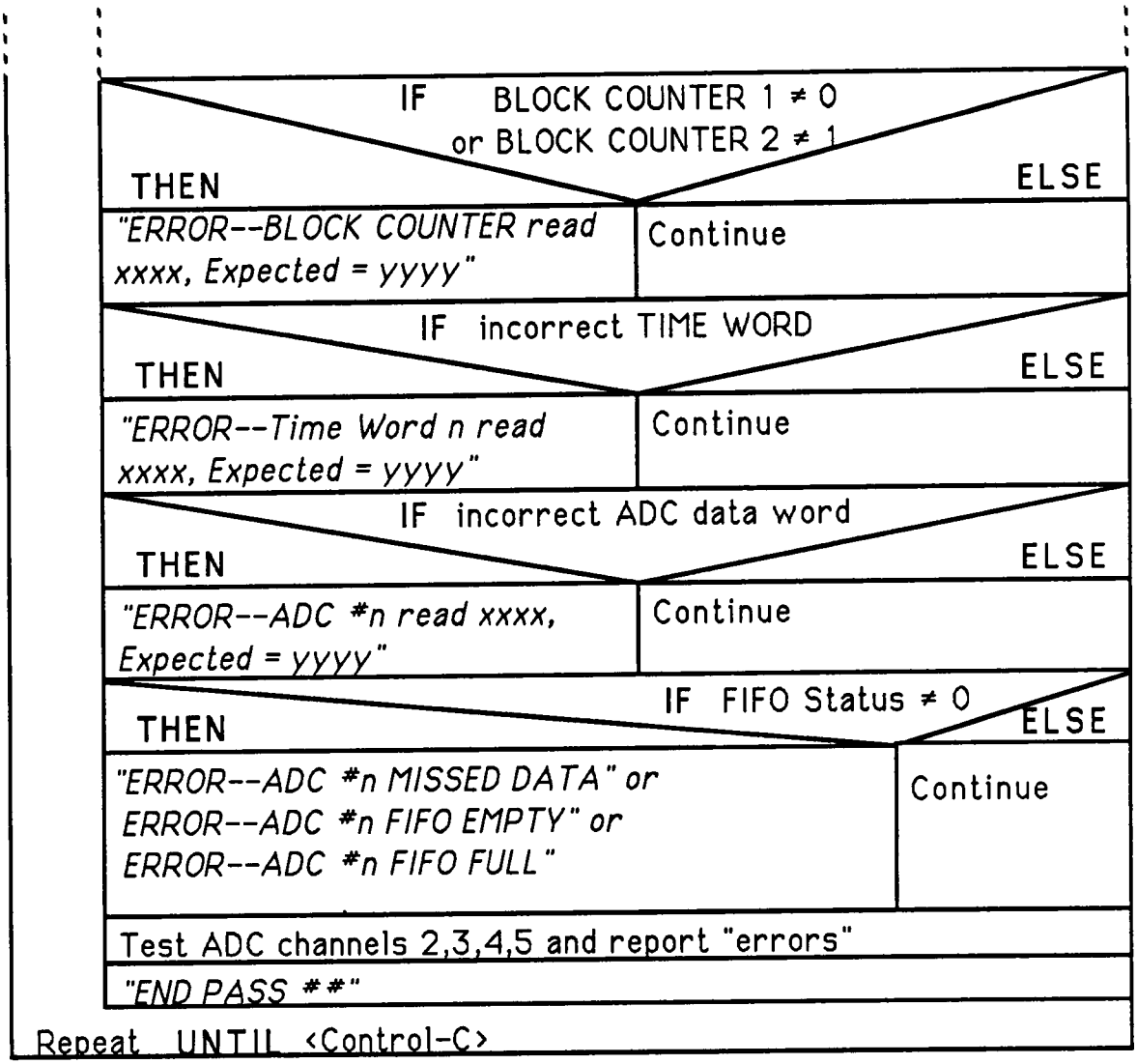

Figure 5.1 (continued)

transferred via DMA to the computer and checked to see that control words and test data are written correctly. Discrepancies are noted with error messages.

The previous tests are repeated with test connector data complemented after obeying the displayed message, "Toggle all test connector switches away from CSI." An END PASS <number> message is displayed when all

tests have been completed, then the entire diagnostic procedure is repeated until aborted with ^C^C (control-c).

6. CONCLUSION

The programmable high-speed data acquisition controller system is operational and performing as required. Because of the high speeds involved, circuitry is implemented with a combination of advanced Schottky (74AS series) TTL, advanced low-power Schottky (74ALS series) TTL, and complementary metal oxide semiconductor (CMOS) integrated circuits. Over 185 electronic components populate 11 double-sided etched-wire circuit boards that plug into a custom motherboard. Five circuit boards that implement the ADC interface are physically and functionally equivalent. No internal signal cabling is used, as the 36 connectors for all external interface equipment mount directly onto the circuit boards. The controller subsystem mounts in a standard 19-in. RETMA relay rack, occupying 8.75 in. of panel space to a depth of 16 in.

To increase reliability, the number of DRV11-J ports and connectors used could be reduced by more thorough use of the DRV11-WA interface and its functions. Controller setup parameters could then be multiplexed via this interface instead of by dedicated ports. The additional circuitry necessary to implement such an improvement would be offset by deleting DRV11-J boards in the host, therefore freeing valuable host backplane space.

During a data acquisition exercise, no data are transferred to the host because of the high data rates. It would be a considerable advantage to know that the equipment is operating correctly and thus be able to verify data integrity during a run. A lookup table could be designed into the DRV11-WA interface that, when properly programmed, would allow the

transfer of decommutated data points at a reduced data rate that the host could handle. Analysis of these data, coincident with the run, would allow inspection and verification in real time of a limited number of channels.

Use of programmable logic devices for the state machines, clock generators, and associated logic would reduce chip count and increase reliability. The circuit board layout would also be simplified. During design development, simulation and testing of each circuit could also be performed, as these capabilities are included in most popular computer-aided design packages.

Design and implementation of the high-speed data acquisition controller have demonstrated features that multiplex and prepare for recording large quantities of digitized data from five ADCs and a time code generator. The internal circuitry formats the data at high speed prior to recording to enable proper decommutation and synchronization upon playback. Analysis of these data can then be performed later at lower transfer rates, conforming to the capabilities of a postrun processor. This controller design could also be expanded to include more ADCs or other digital output devices from a multitude of vendors.

BIBLIOGRAPHY

BIBLIOGRAPHY

DRV11-WA General Purpose DMA Interface User's Guide. Maynard, Massachusetts: Digital Equipment Corporation, 1980.

DRV11-J Parallel Line Interface User's Guide. Maynard, Massachusetts: Digital Equipment Corporation, 1980.

GM Series Analog-to-Digital Conversion Systems Interface Manual. Anaheim: Preston Scientific, 1985.

ALS/AS Logic Data Book. Dallas: Texas Instruments, Inc., 1986.

Nagle, H. Troy; B. D. Carroll and J. David Irwin. An Introduction to Computer Logic. Englewood Cliffs, New Jersey: Prentice-Hall, 1975.

APPENDIX

```
FORTRAN IV    V02.6      Thu 09-Jul-87 12:46:29          PAGE 001

      C12345678901234567890123456789012345678901234567890123456789012345678901234567890
      C
      C  This program tests the ADC data collection capabilities and the 32K
      C  Special Control Memory of the CSI.
      C  A DRV11-WA, 2 DRV11-J boards and CSI test connectors are necessary.
      C  This program is to be used only as a diagnostic.
      C
      C
      C  PROGRAM:  CSIDIA.FOR
      C  AUTHOR: Rusty Rochelle
      C  DATE  : Jan. 16,1987
      C
      C              Version X01.01  9-JUL-87  12:39  -  9-JUL-87  12:39
      C       Modified by: R. B. Rochelle
      C       Added name of program to the comments above.
      C
      C*********************************************************************
0001        IMPLICIT INTEGER*2 ( A-Z )
0002        DIMENSION  ADC(5), CLK(3),TMWRD(3),ADCDAT(5),NOTTM(3),NOTADC(5)
0003        VIRTUAL VARRAY(32767)
0004        COMMON /CSIPAR/ CSIBUS,RESET,VADDR,FUNCT1,FUNCT2,DRWCSR,DRWWCR,
           1 DRWBAR,DRWBAE,DBR1B,DBR1C,DBR1D,DBR2C,GO,STATA,STATB,STATC,
           2 CYCLE,READY
      C
      C  This is address where the virtual array VARRAY begins
0005        VADDR= "160000
      C
      C  ADC Data block test words for test switches towards CSI
0006        CNTRL1= "177777
0007        CNTRL2= "52453
0008        BLKCNT= "40000
0009        TMWRD(1)= "66666
0010        TMWRD(2)= "77777
0011        TMWRD(3)= "66666
0012        ADCDAT(5)= "155555
0013        ADCDAT(4)= "44444
0014        ADCDAT(3)= "133333
0015        ADCDAT(2)= "22222
0016        ADCDAT(1)= "111111
      C
      C  Clear Pass Counter
0017        PASCNT=0
      C
      C  CSI Master Clocks, 25.165824 MHz, 14.680064 MHz, 10.485760 MHz respectively
0018        CLK(1) = "17
0019        CLK(2) = "16
0020        CLK(3) = "15
      C  CSI Control Register Bits, DRV11-J 2C
0021        RESET= "01
0022        RSTADR= "02
0023        COLECT= "04
      C
      C  DRV11-WA Interface Addresses
0024        ADC(1)= "20
```

```
FORTRAN IV    V02.6      Thu 09-Jul-87 12:46:29          PAGE 002

0025          ADC(2)= "40
0026          ADC(3)= "60
0027          ADC(4)= "100
0028          ADC(5)= "120
0029          CSIBUS= "140
      C
      C  READY bit for DRV11-WA
0030          READY = "200
      C
      C  CYCLE bit
0031          CYCLE = "400
      C GO bit
0032          GO = 1
      C
      C  Function 1 is 1 for write, 0 for read
      C  Function 1 bit assignment
0033          FUNCT1 = 2
      C
      C  Function 2 is 1 for reset selected interface, 0 otherwise
      C  function 2 bit assignment
0034          FUNCT2 = 4
      C
      C  Function 3 NOT USED
      C  Function 3 bit assignment
0035          FUNCT3 = 8
      C
      C  DRV11-WA Status Bits
0036          STATA= "4000
0037          STATB= "2000
0038          STATC= "1000
      C
      C  DRV11-WA Word Count Register
0039          DRWWCR= "172410
      C
      C  DRV11-WA BAR
0040          DRWBAR = DRWWCR + 2
      C
      C  DRV11-WA BAE
0041          DRWBAE = DRWBAR
      C
      C  DRV11-WA CSR
0042          DRWCSR = DRWWCR + 4
      C
      C  Base CSR address for DRV11-J #1
0043          DRVJ1 = "164160
      C
      C  Base CSR address for DRV11-J #2
0044          DRVJ2 = "164140
      C
      C  Addresses for DRV11-J #1 Ports
      C
      C  CSR address for Port A
0045          CSR1A = DRVJ1 + 0
      C
```

```
FORTRAN IV     V02.6       Thu 09-Jul-87 12:46:29          PAGE 003

      C  DBR address for Port A
0046        DBR1A = DRVJ1 + 2
      C
      C  CSR address for Port B
0047        CSR1B = DRVJ1 + 4
      C
      C  DATA Buffer for Port B
0048        DBR1B = DRVJ1 + 6
      C
      C  CSR address for Port C
0049        CSR1C = DRVJ1 + 8
      C
      C  Data Buffer for Port C
0050        DBR1C = DRVJ1 + 10
      C
      C  CSR address for Port D
0051        CSR1D = DRVJ1 + 12
      C
      C  Data Buffer for Port D
0052        DBR1D = DRVJ1 + 14
      C
      C  Set addresses for DRV11-J #2 Ports
      C
      C  CSR address for Port 2 C
0053        CSR2C = DRVJ2 + 8
      C
      C  Data Buffer address for Port 2C
0054        DBR2C = DRVJ2 + 10
      C
      C  CSR address for Port 2D
0055        CSR2D = DRVJ2 + 12
      C
      C  Data Buffer address for Port 2D
0056        DBR2D = DRVJ2 + 14
      C
      C
      C  SET DIROUT TO EQUAL THE BIT PATTERN TO COMMAND THE DRV11-J TO SET THE
      C  PORT TO OUTPUT DATA
      C
0057        DIROUT = 256
      C
      C  Set DRV11-J #1 Ports data direction to output
      C
      C  SET PORT 1A TO OUTPUT DATA
      C
0058        CALL IPOKE(CSR1A, DIROUT)
      C
      C  SET PORT 1B TO OUTPUT DATA
      C
0059        CALL IPOKE(CSR1B, DIROUT)
      C
      C  SET PORT 1C TO OUTPUT DATA
      C
0060        CALL IPOKE(CSR1C, DIROUT)
```

```
FORTRAN IV    V02.6      Thu 09-Jul-87 12:46:29          PAGE 004

      C
      C  SET PORT 1D TO OUTPUT DATA
      C
0061        CALL IPOKE(CSR1D, DIROUT)
      C
      C  Set DRV11-J #2 Ports data direction to output
      C  SET PORT C TO OUTPUT DATA
      C
0062        CALL IPOKE(CSR2C, DIROUT)
      C
      C  SET PORT D TO OUTPUT DATA
      C
0063        CALL IPOKE(CSR2D, DIROUT)
      C
      C  Increment diagnostic pass count
      C  Diagnostic repeats here
      C
0064  1000  PASCNT=PASCNT+1
      C
      C
      C
      C  RESET THE CSI AND THE SPECIAL CONTROL MEMORY
      C
0065        CALL IPOKE (DBR2C, 0)
      C
      C  Enable 32K Memory addressing
0066        CALL IPOKE (DBR2C, RSTADR )
      C
      C  SEND PORT D THE DATA TO BE WRITTEN TO THE SPECIAL CONTROL MEMORY
      C
0067        DO 100 I = -10, 32757
0068          CALL IPOKE(DBR2D, I+10)
0069  100   CONTINUE
      C
      C
      C  Test the 3 Master Clocks for operation
      C
0070        DO 108 CLKNUM=1,3
      C  Reset CSI
0071        CALL IPOKE(DBR2C,0)
      C  Select Master Clock
0072        CALL IPOKE (DBR1A, CLK(CLKNUM))
      C
      C  Clear CSR2D data direction in, to prepare to read Special Control Memory
0073        CALL IPOKE(CSR2D,0)
      C
      C  Set PDP 11/73 memory for DMA
      C
0074        DO 102 I=-10,32757
0075         VARRAY(I+11)="177777
0076  102   CONTINUE
      C
      C  DMA 32K Special Control Memory, "100000= -32768
      C
```

```
FORTRAN IV    V02.6      Thu 09-Jul-87 12:46:29          PAGE 005

0077          CALL DMA( "100000,RSTADR)
      C
      C
      C  Wait here until DRV11-WA DMA is done (READY bit is high)
      C
0078          DO 106 N=1,10
0079           DO 105 I=1,32766
0080             IF ( (IPEEK( DRWCSR) .AND. READY) .NE. 0) GO TO 108
0082  105      CONTINUE
0083  106     CONTINUE
0084          IF (CLKNUM .EQ. 1)
             1     TYPE *,'ERROR--25.165824 MHz CLOCK too slow'
0086          IF (CLKNUM .EQ. 2)
             1     TYPE *,'ERROR--14.680064 MHz CLOCK too slow'
0088          IF (CLKNUM .EQ. 3)
             1     TYPE *,'ERROR--10.485760 MHz CLOCK too slow'
0090  108     CONTINUE
      C
      C  If CSI.MISSED.DATA is set then transmission error occurred.
      C  DRV11-WA didn't get data fast enough.
0091          IF( ( IPEEK(DRWCSR) .AND. STATA) .NE. 0)
             1     TYPE *,'ERROR--CSI.MISSED.DATA signal is HIGH, Expected LOW'
      C
      C  Check HEADER.DONE, should be high
0093          IF( (IPEEK(DRWCSR) .AND. STATB) .EQ. 0)
             1     TYPE *, 'ERROR--HEADER.DONE signal is LOW, Expected HIGH'
      C
      C  Check SCAN.ADCS (L), should be HIGH
0095          IF ( (IPEEK(DRWCSR) .AND. STATC) .EQ. 0)
             1     TYPE *, 'ERROR--SCAN.ADCS signal is LOW, Expected HIGH'
      C
      C
      C  Check VARRAY, Data read from CSI 32K Special Control Memory
0097          DO 400 I=-10,32757
0098           IF( VARRAY(I+11) .NE. I+10 )
             1      WRITE(5,600) I+10,VARRAY(I+11),I+10
0100  600       FORMAT(' ERROR--Memory Loc.',O6,'=',O6,' Expected=',O6)
0101  400     CONTINUE
      C
      C test CSI data block
      C
0102          PAUSE 'Toggle all test connector switches towards CSI,
             1 then hit "RETURN"'
      C
0103          CALL SCNTST(CNTRL1,CNTRL2,BLKCNT,TMWRD,ADCDAT,
             1 N,VARRAY,ADC,COLECT)
      C
      C  Test complement of Time Words and ADC data
      C
0104          PAUSE 'Toggle test connector switches away from CSI,
             1 then hit "RETURN"'
      C
      C  Define expected data
      C
```

```
FORTRAN IV     V02.6       Thu 09-Jul-87 12:46:29          PAGE 006

0105          DO 700 I=1,3
0106           NOTTM(I)= (.NOT. TMWRD(I)) .AND. "77777
0107  700     CONTINUE
0108          DO 800 I=1,5
0109           NOTADC(I)= .NOT. ADCDAT(I)
0110  800     CONTINUE
      C
      C  Define Control Word 3, complement 2nd Most Sig. bit
      C
0111          CNTRL3=BLKCNT .AND. "137777
      C
      C  test data
      C
0112          CALL SCNTST(CNTRL1,CNTRL2,CNTRL3,NOTTM,NOTADC,
             1 N,VARRAY,ADC,COLECT)
      C
0113          WRITE(5,15000) PASCNT
0114  15000 FORMAT(' END PASS ',I4/ )
      C
      C  Do it again
      C
0115          GO TO 1000
0116          END
```

```
FORTRAN IV     Storage Map for Program Unit .MAIN.

Local Variables, .PSECT $DATA, Size = 000160 (   56. words)

Name   Type  Offset     Name   Type  Offset     Name   Type  Offset

BLKCNT I*2   000076     CLKNUM I*2   000140     CNTRL1 I*2   000072

CNTRL2 I*2   000074     CNTRL3 I*2   000144     COLECT I*2   000104

CSR1A  I*2   000114     CSR1B  I*2   000120     CSR1C  I*2   000122

CSR1D  I*2   000124     CSR2C  I*2   000126     CSR2D  I*2   000130

DBR1A  I*2   000116     DBR2D  I*2   000132     DIROUT I*2   000134

DRVJ1  I*2   000110     DRVJ2  I*2   000112     FUNCT3 I*2   000106

I      I*2   000136     N      I*2   000142     PASCNT I*2   000100

RSTADR I*2   000102

COMMON Block /CSIPAR/, Size = 000046 (   19. words)

Name   Type  Offset     Name   Type  Offset     Name   Type  Offset

CSIBUS I*2   000000     RESET  I*2   000002     VADDR  I*2   000004

FUNCT1 I*2   000006     FUNCT2 I*2   000010     DRWCSR I*2   000012

DRWWCR I*2   000014     DRWBAR I*2   000016     DRWBAE I*2   000020

DBR1B  I*2   000022     DBR1C  I*2   000024     DBR1D  I*2   000026

DBR2C  I*2   000030     GO     I*2   000032     STATA  I*2   000034

STATB  I*2   000036     STATC  I*2   000040     CYCLE  I*2   000042

READY  I*2   000044

Local and COMMON Arrays:

Name     Type   Section Offset  ------Size------ Dimensions
ADC      I*2    $DATA   000000  000012 (    5.) (5)
ADCDAT   I*2    $DATA   000026  000012 (    5.) (5)
CLK      I*2    $DATA   000012  000006 (    3.) (3)
NOTADC   I*2    $DATA   000046  000012 (    5.) (5)
NOTTM    I*2    $DATA   000040  000006 (    3.) (3)
TMWRD    I*2    $DATA   000020  000006 (    3.) (3)

VIRTUAL Arrays, Total Size = 00200000 (  32768. words)

Name     Type    Offset    -------Size-------- Dimensions
VARRAY   I*2     00000000  00177776 (  32767.) (32767)

Subroutines, Functions, Statement and Processor-Defined Functions:

Name   Type   Name   Type   Name   Type   Name   Type   Name   Type

DMA    I*2    IPEEK  I*2    IPOKE  I*2    SCNTST I*2
```

```
FORTRAN IV    V02.6      Thu 09-Jul-87 12:46:41          PAGE 001

      C
      C
      C
0001        SUBROUTINE SCNTST(CNTRL1,CNTRL2,BLKCNT,TMWRD,ADCDAT,
           1 N,VARRAY,ADC,COLECT)
      C
      C*************************************************************************
      C
      C  Subroutine SCNTST tests the ADC data collection capabilities of the CSI
      C
      C*************************************************************************
      C
      C
0002        IMPLICIT INTEGER*2 ( A-Z )
0003        VIRTUAL VARRAY(32767)
0004        DIMENSION  ADC(5),TMWRD(3),ADCDAT(5)
0005        COMMON /CSIPAR/ CSIBUS,RESET,VADDR,FUNCT1,FUNCT2,DRWCSR,DRWWCR,
           1 DRWBAR,DRWBAE,DBR1B,DBR1C,DBR1D,DBR2C,GO,STATA,STATB,STATC,
           2 CYCLE,READY
      C
      C  Reset the CSI
0006        CALL IPOKE (DBR2C, 0)
      C
      C  Set BLOCK LENGTH = 16
0007        CALL IPOKE( DBR1D, -16)
      C
      C  Set CONTROL WORD SPACING = 1
0008        CALL IPOKE( DBR1C, -1)
      C
      C  Set ADC QUANTity=4 and select ADC scan order 1,4,3,2
0009        CALL IPOKE( DBR1B, -4 .AND. "03 .OR. "23410)
      C
      C  Setup ADC 1,2,3,4 for data collection.
0010        DO 130 ADCNUM=1,4
0011         CALL SETUP(ADCNUM,ADC)
0012  130   CONTINUE
      C
      C
      C  DMA 22 data block words
      C
0013        CALL DMA(-22,COLECT)
      C
      C  Wait here until DRV11-WA DMA is done (READY bit is high) or timesout
      C
0014        DO 150 I=1,100
0015  150   IF ( (IPEEK(DRWCSR) .AND. READY) .EQ. 0) CONTINUE
      C
      C  Check received data
      C
      C  Point N to ADC #4 data
      C
0017        N=10
0018        CALL SCNCHK(CNTRL1,CNTRL2,BLKCNT,TMWRD,ADCDAT,N,VARRAY,ADC)
      C
```

```
FORTRAN IV     V02.6       Thu 09-Jul-87 12:46:41          PAGE 002

      C  Check ADC/FIFO #1,2,3,4 Status bits
      C
0019        DO 155 ADCNUM=1,4
0020         CALL ADSTAT(ADCNUM,ADC)
0021  155   CONTINUE
      C
      C
      C  Repeat test for ADC #2,3,4,5
      C
      C
      C  RESET THE CSI AND THE SPECIAL CONTROL MEMORY
      C
0022        CALL IPOKE (DBR2C, 0)
      C
      C  Set ADC QUANTity=4 and select ADC scan order 5,4,3,2
0023        CALL IPOKE( DBR1B, -4 .AND. "03 .OR. "23450)
      C
      C  Setup ADC #2,3,4,5 for data collection
      C
0024        DO 160 ADCNUM=2,5
0025         CALL SETUP(ADCNUM,ADC)
0026  160   CONTINUE
      C
      C  DMA 22 words of data block,
      C  6 control words + 10 ADC data words + 6 control words
      C
0027        CALL DMA(-22,COLECT)
      C
      C  Wait here until DRV11-WA DMA is done (READY bit is high) or timesout
      C
0028        DO 4200 I=1,100
0029  4200  IF ( (IPEEK( DRWCSR) .AND. READY) .NE. 0) CONTINUE
      C
      C  Check received data and
      C  Point N to position of ADC #5 in data block
      C
0031        N=9
0032        CALL SCNCHK(CNTRL1,CNTRL2,BLKCNT,TMWRD,ADCDAT,N,VARRAY,ADC)
      C
      C  Check ADC/FIFO #2,3,4,5 Status Bits
      C
0033        DO 4500 ADCNUM=2,5
0034         CALL ADSTAT(ADCNUM,ADC)
0035  4500  CONTINUE
      C
0036        RETURN
0037        END
```

```
FORTRAN IV     Storage Map for Program Unit SCNTST

Local Variables, .PSECT $DATA, Size = 000032 (   13. words)

Name    Type  Offset      Name    Type  Offset      Name    Type  Offset

ADCNUM  I*2   000022      BLKCNT  I*2 @ 000004      CNTRL1  I*2 @ 000000

CNTRL2  I*2 @ 000002      COLECT  I*2 @ 000020      I       I*2   000024

N       I*2 @ 000012

COMMON Block /CSIPAR/, Size = 000046 (   19. words)

Name    Type  Offset      Name    Type  Offset      Name    Type  Offset

CSIBUS  I*2   000000      RESET   I*2   000002      VADDR   I*2   000004

FUNCT1  I*2   000006      FUNCT2  I*2   000010      DRWCSR  I*2   000012

DRWWCR  I*2   000014      DRWBAR  I*2   000016      DRWBAE  I*2   000020

DBR1B   I*2   000022      DBR1C   I*2   000024      DBR1D   I*2   000026

DBR2C   I*2   000030      GO      I*2   000032      STATA   I*2   000034

STATB   I*2   000036      STATC   I*2   000040      CYCLE   I*2   000042

READY   I*2   000044

Local and COMMON Arrays:

Name      Type    Section Offset  ------Size----- Dimensions
ADC       I*2     @ $DATA   000016  000012 (    5.) (5)
ADCDAT    I*2     @ $DATA   000010  000012 (    5.) (5)
TMWRD     I*2     @ $DATA   000006  000006 (    3.) (3)

VIRTUAL Arrays, Total Size = 00000000 (      0. words)

Name      Type     Offset    -------Size-------- Dimensions
VARRAY    I*2      @   000014  00177776 (  32767.) (32767)

Subroutines, Functions, Statement and Processor-Defined Functions:

Name    Type    Name    Type    Name    Type    Name    Type    Name    Type

ADSTAT  I*2     DMA     I*2     IPEEK   I*2     IPOKE   I*2     SCNCHK  I*2

SETUP   I*2
```

```
FORTRAN IV    V02.6      Thu 09-Jul-87 12:46:49          PAGE 001

      C
      C
0001        SUBROUTINE SETUP(ADCNUM,ADC)
      C
      C*****************************************************************
      C
      C  Subroutine SETUP prepares an ADC for data collection.
      C  Reset interface with DRV11-WA FUNCTION 2 (H) bit and enable
      C  FIFO with FUNCTION 1 (L).
      C  Test control word output to ADCs handshake capabilities.
      C
      C*****************************************************************
      C
0002        IMPLICIT INTEGER*2 ( A-Z )
0003        DIMENSION  ADC(5)
0004        COMMON /CSIPAR/ CSIBUS,RESET,VADDR,FUNCT1,FUNCT2,DRWCSR,DRWWCR,
           1 DRWBAR,DRWBAE,DBR1B,DBR1C,DBR1D,DBR2C,GO,STATA,STATB,STATC,
           2 CYCLE,READY
      C
0005         CALL IPOKE( DBR2C, (ADC(ADCNUM) .OR. RESET))
0006         CALL IPOKE( DRWCSR, FUNCT2)
      C
      C  Prepare DRV11-WA to output dummy data
0007         CALL IPOKE( DRWCSR, FUNCT1)
0008        DUMMY=IPEEK(DRWWCR)
0009         CALL IPOKE( DRWBAR, 0)
0010         CALL IPOKE( DRWBAE, 0)
0011         CALL IPOKE( DRWWCR, -16)
      C
      C  DMA some dummy words to ADC board output data port
0012         CALL IPOKE( DRWCSR, CYCLE .OR. FUNCT1 .OR. GO)
      C
      C  Wait here until READY is high or timeout
0013        DO 125 J=1,1000
0014  125     IF((IPEEK(DRWCSR) .AND. READY) .NE. 0) GO TO 128
0016        WRITE(5,126) ADCNUM
0017  126   FORMAT(' ERROR--ADC #',I1,' handshaking not responding')
      C
      C  Enable FIFO for ADC data read
0018  128   CALL IPOKE( DRWCSR, 0)
      C
0019        RETURN
      C
0020        END
```

```
FORTRAN IV     Storage Map for Program Unit SETUP

Local Variables, .PSECT $DATA, Size = 000014 (     6. words)

Name    Type  Offset      Name    Type  Offset      Name    Type  Offset

ADCNUM  I*2 @ 000000      DUMMY   I*2   000004      J       I*2   000006

COMMON Block /CSIPAR/, Size = 000046 (    19. words)

Name    Type  Offset      Name    Type  Offset      Name    Type  Offset

CSIBUS  I*2   000000      RESET   I*2   000002      VADDR   I*2   000004

FUNCT1  I*2   000006      FUNCT2  I*2   000010      DRWCSR  I*2   000012

DRWWCR  I*2   000014      DRWBAR  I*2   000016      DRWBAE  I*2   000020

DBR1B   I*2   000022      DBR1C   I*2   000024      DBR1D   I*2   000026

DBR2C   I*2   000030      GO      I*2   000032      STATA   I*2   000034

STATB   I*2   000036      STATC   I*2   000040      CYCLE   I*2   000042

READY   I*2   000044

Local and COMMON Arrays:

Name      Type    Section Offset  ------Size------ Dimensions
ADC       I*2     @ $DATA   000002  000012 (     5.) (5)

Subroutines, Functions, Statement and Processor-Defined Functions:

Name    Type    Name    Type    Name    Type    Name    Type    Name    Type

IPEEK   I*2     IPOKE   I*2
```

```
FORTRAN IV     V02.6       Thu 09-Jul-87 12:46:54          PAGE 001

      C
      C
0001        SUBROUTINE SCNCHK(CNTRL1,CNTRL2,BLKCNT,TMWRD,ADCDAT,
           1 N,VARRAY,ADC)
      C
      C*********************************************************************
      C
      C  Subroutine SCNCHK checks an ADC data block, collected by CSI and
      C  DMA'ed to PDP 11/73 VARRAY.
      C  Error messages are posted.
      C
      C*********************************************************************
      C
0002        IMPLICIT INTEGER*2 ( A-Z )
0003        VIRTUAL VARRAY(32767)
0004        DIMENSION  ADC(5),TMWRD(3),ADCDAT(5)
0005        COMMON /CSIPAR/ CSIBUS,RESET,VADDR,FUNCT1,FUNCT2,DRWCSR,DRWWCR,
           1 DRWBAR,DRWBAE,DBR1B,DBR1C,DBR1D,DBR2C,GO,STATA,STATB,STATC,
           2 CYCLE,READY
      C
      C  If CSI.MISSED.DATA is set then transmission error occurred.
      C  DRV11-WA didn't get data fast enough.
0006        IF( ( IPEEK(DRWCSR) .AND. STATA) .NE. 0)
           1     TYPE *,'ERROR--CSI.MISSED.DATA signal is HIGH, Expected LOW'
      C
      C  Check HEADER.DONE, should be high
0008        IF( (IPEEK(DRWCSR) .AND. STATB) .EQ. 0)
           1     TYPE *, 'ERROR--HEADER.DONE signal is LOW, Expected HIGH'
      C
      C  Check SCAN.ADCS (L), should be low
0010        IF ( (IPEEK(DRWCSR) .AND. STATC) .NE. 0)
           1     TYPE *, 'ERROR--SCAN.ADCS signal is HIGH, Expected LOW'
      C
      C
      C  Check Data
      C    Check Control Word 1, depends on block length and control word spacing
      C
0012        IF (VARRAY(1) .NE. CNTRL1) WRITE(5,210) VARRAY(1),CNTRL1
0014  210   FORMAT(' ERROR--Control Word 1 read',O6,' Expected=',O6)
      C
      C  Check Control Word 2, hardwired on CSI Control/Mux board
      C
0015        IF (VARRAY(2) .NE. CNTRL2) WRITE(5,212) VARRAY(2),CNTRL2
0017  212   FORMAT(' ERROR--Control Word 2 read',O6,' Expected=',O6)
      C
      C  Check Control Word 3 (MSbit is 0, next MSbit is from time code
      C   and could be 0 or 1 depending on state of test conn. sw., 13 LSbits
      C   are Block Counter)
      C
0018        IF(VARRAY(3) .NE. BLKCNT)
           1    WRITE(5,214) VARRAY(3),BLKCNT
0020  214   FORMAT(' ERROR--Control Word 3 read',O6,' Expected=',O6)
      C
      C  Check Block Counter of next data block
```

```
FORTRAN IV     V02.6       Thu 09-Jul-87 12:46:54          PAGE 002

      C
0021          IF(VARRAY(19) .NE. (BLKCNT +1))
             1    WRITE(5,230) VARRAY(19), BLKCNT +1
0023  230     FORMAT(' ERROR--2nd Block Counter read',O6,' Expected=',O6)
      C
      C
      C  Check Time Words
      C
0024          DO 220 I=1,3
0025           IF(VARRAY(I+3) .NE. TMWRD(I))
             1      WRITE(5,216) I,VARRAY(I+3),TMWRD(I)
0027  216     FORMAT(' ERROR--Time Word ',I1,' read ',O6,' Expected=',O6)
0028  220     CONTINUE
      C
      C
      C  Check four ADC data words
      C  Since scan order is ADC#1,4,3,2 (or 5,4,3,2,1) and repeats,
      C  when I=10 ADC #4 is read, and when I=13 (or 9) ADC#1 (or 5) is read.
      C  N points to location of highest order ADC data word in a data block.
      C
0029          DO 228 I=N,N+3
0030           IF(VARRAY(I) .NE. ADCDAT(14-I))
             1      WRITE(5,222) 14-I,VARRAY(I),ADCDAT(14-I)
0032  222     FORMAT(' ERROR--ADC #',I1,' data read ',O6,' Expected=',O6)
0033  228     CONTINUE
      C
0034          RETURN
0035          END
```

```
FORTRAN IV     Storage Map for Program Unit SCNCHK

Local Variables, .PSECT $DATA, Size = 000056 (   23. words)

Name   Type  Offset     Name   Type  Offset     Name   Type  Offset

BLKCNT I*2 @ 000004     CNTRL1 I*2 @ 000000     CNTRL2 I*2 @ 000002

I      I*2   000022     N      I*2 @ 000012

COMMON Block /CSIPAR/, Size = 000046 (   19. words)

Name   Type  Offset     Name   Type  Offset     Name   Type  Offset

CSIBUS I*2   000000     RESET  I*2   000002     VADDR  I*2   000004

FUNCT1 I*2   000006     FUNCT2 I*2   000010     DRWCSR I*2   000012

DRWWCR I*2   000014     DRWBAR I*2   000016     DRWBAE I*2   000020

DBR1B  I*2   000022     DBR1C  I*2   000024     DBR1D  I*2   000026

DBR2C  I*2   000030     GO     I*2   000032     STATA  I*2   000034

STATB  I*2   000036     STATC  I*2   000040     CYCLE  I*2   000042

READY  I*2   000044

Local and COMMON Arrays:

Name      Type   Section Offset  ------Size------ Dimensions
ADC       I*2    @ $DATA  000016  000012 (    5.) (5)
ADCDAT    I*2    @ $DATA  000010  000012 (    5.) (5)
TMWRD     I*2    @ $DATA  000006  000006 (    3.) (3)

VIRTUAL Arrays, Total Size = 00000000 (      0. words)

Name      Type    Offset   -------Size-------- Dimensions
VARRAY    I*2     @   000014  00177776 ( 32767.) (32767)

Subroutines, Functions, Statement and Processor-Defined Functions:

Name   Type   Name   Type   Name   Type   Name   Type   Name   Type

IPEEK  I*2
```

```
FORTRAN IV     V02.6       Thu 09-Jul-87 12:47:01          PAGE 001

      C
      C
0001        SUBROUTINE ADSTAT(ADCNUM,ADC)
      C
      C*********************************************************************
      C
      C Subroutine ADSTAT checks ADC/FIFO Status Bits and reports errors.
      C
      C*********************************************************************
      C
0002        IMPLICIT INTEGER*2 ( A-Z )
0003        DIMENSION  ADC(5)
0004        COMMON /CSIPAR/ CSIBUS,RESET,VADDR,FUNCT1,FUNCT2,DRWCSR,DRWWCR,
           1 DRWBAR,DRWBAE,DBR1B,DBR1C,DBR1D,DBR2C,GO,STATA,STATB,STATC,
           2 CYCLE,READY
      C
0005         CALL IPOKE(DBR2C,(ADC(ADCNUM) .OR. COLECT .OR. RESET) )
0006         IF (( IPEEK(DRWCSR) .AND. STATA) .NE. 0) WRITE(5,4000) ADCNUM
0008  4000     FORMAT(' ERROR--ADC #',I1,' MISSED.DATA')
0009         IF (( IPEEK(DRWCSR) .AND. STATB) .NE. 0) WRITE(5,4005) ADCNUM
0011  4005     FORMAT(' ERROR--ADC #',I1,' FIFO.EMPTY')
0012         IF (( IPEEK(DRWCSR) .AND. STATC) .NE. 0) WRITE(5,4010) ADCNUM
0014  4010     FORMAT(' ERROR--ADC #',I1,' FIFO.FULL')
      C
0015        RETURN
0016        END
```

```
FORTRAN IV     Storage Map for Program Unit ADSTAT

Local Variables, .PSECT $DATA, Size = 000010 (     4. words)

Name    Type  Offset      Name    Type  Offset      Name    Type  Offset

ADCNUM  I*2 @ 000000      COLECT  I*2   000004

COMMON Block /CSIPAR/, Size = 000046 (   19. words)

Name    Type  Offset      Name    Type  Offset      Name    Type  Offset

CSIBUS  I*2   000000      RESET   I*2   000002      VADDR   I*2   000004

FUNCT1  I*2   000006      FUNCT2  I*2   000010      DRWCSR  I*2   000012

DRWWCR  I*2   000014      DRWBAR  I*2   000016      DRWBAE  I*2   000020

DBR1B   I*2   000022      DBR1C   I*2   000024      DBR1D   I*2   000026

DBR2C   I*2   000030      GO      I*2   000032      STATA   I*2   000034

STATB   I*2   000036      STATC   I*2   000040      CYCLE   I*2   000042

READY   I*2   000044

Local and COMMON Arrays:

Name      Type    Section Offset  ------Size------ Dimensions
ADC       I*2     @ $DATA   000002  000012 (     5.) (5)

Subroutines, Functions, Statement and Processor-Defined Functions:

Name    Type    Name    Type    Name    Type    Name    Type    Name    Type

IPEEK    I*2    IPOKE    I*2
```

```
FORTRAN IV     V02.6       Thu 09-Jul-87 12:47:05          PAGE 001

      C
      C
0001        SUBROUTINE DMA(WRDCNT,WHAT)
      C
      C*********************************************************************
      C
      C  Subroutine DMA sets up DRV11-WA to receive WRDCNT number of words,
      C  either memory or adc data (WHAT), at CSI rate at address of VADDR.
      C
      C*********************************************************************
      C
0002        IMPLICIT INTEGER*2 ( A-Z )
0003        COMMON /CSIPAR/ CSIBUS,RESET,VADDR,FUNCT1,FUNCT2,DRWCSR,DRWWCR,
           1 DRWBAR,DRWBAE,DBR1B,DBR1C,DBR1D,DBR2C,GO,STATA,STATB,STATC,
           2 CYCLE,READY
      C
      C  Set DRV11-WA Interface Select to CSI DATA BUS
0004        CALL IPOKE( DBR2C, (CSIBUS .OR. RESET) )
      C
      C  Reset CSI DATA BUS DRV11-WA Interface
0005        CALL IPOKE( DRWCSR, FUNCT2 )
      C
      C  Delay for Reset to propagate
0006        DO 40 I=1,32000
0007        IF( (IPEEK(DRWCSR) .AND. STATA) .EQ. 0) GO TO 60
0009  40    CONTINUE
0010        TYPE *,' ERROR--could not reset CSI/DRV11-WA Interface'
0011  60    CONTINUE
      C
0012        CALL IPOKE( DRWCSR, 0 )
      C
      C  Load DRV11-WA Word Count Register
0013        CALL IPOKE( DRWWCR, WRDCNT )
      C
      C  Point BAR and BAE to buffer.  Look at WCR to clear BAE flag.
0014        I=IPEEK(DRWWCR)
0015        CALL IPOKE( DRWBAR, VADDR )
0016        CALL IPOKE( DRWBAE, 0 )
      C
      C  Allow DMA Read operation
0017        CALL IPOKE( DRWCSR, GO )
      C
      C  Set DRV11-WA Interface Select to CSI DATA BUS and
      C   set COLLECT.DATA or RSTADR
0018        CALL IPOKE( DBR2C, (CSIBUS .OR. RESET .OR. WHAT) )
      C
0019        RETURN
0020        END
```

FORTRAN IV Storage Map for Program Unit DMA

Local Variables, .PSECT $DATA, Size = 000012 (5. words)

Name	Type	Offset	Name	Type	Offset	Name	Type	Offset
I	I*2	000004	WHAT	I*2 @	000002	WRDCNT	I*2 @	000000

COMMON Block /CSIPAR/, Size = 000046 (19. words)

Name	Type	Offset	Name	Type	Offset	Name	Type	Offset
CSIBUS	I*2	000000	RESET	I*2	000002	VADDR	I*2	000004
FUNCT1	I*2	000006	FUNCT2	I*2	000010	DRWCSR	I*2	000012
DRWWCR	I*2	000014	DRWBAR	I*2	000016	DRWBAE	I*2	000020
DBR1B	I*2	000022	DBR1C	I*2	000024	DBR1D	I*2	000026
DBR2C	I*2	000030	GO	I*2	000032	STATA	I*2	000034
STATB	I*2	000036	STATC	I*2	000040	CYCLE	I*2	000042
READY	I*2	000044						

Subroutines, Functions, Statement and Processor-Defined Functions:

Name	Type	Name	Type	Name	Type	Name	Type	Name	Type
IPEEK	I*2	IPOKE	I*2						

VITA

Russell Briscoe Rochelle was born December 24, 1954, in Washington, D. C. He grew up in the suburbs of that city and graduated from Springbrook High School, Silver Spring, Maryland, in 1972.

As an Engineer Trainee for NASA/Goddard Space Flight Center, Greenbelt, Maryland, in the cooperative work experience program with UTK, he manned an integrated circuit fabrication process line for the development of charge coupled devices and became intimately acquainted with microprocessors. He received a Bachelor of Science in Electrical Engineering Degree upon his graduation in 1978 from UTK with High Honors and grand hoopla, whereupon, he headed West to live out his lifelong ambition of applying classical physics theory to practical applications and became a ski instructor at the Steamboat Ski Corporation, Steamboat Springs, Colorado. After two seasons, and in keeping with the ski industry, he transferred to the Lift Maintenance Department and spent five years as a Sparky, engineering uphill transportation equipment.

Soon tiring of noisy engine rooms and long underwear he sought out high-speed frontiers in data acquisition and signal processing. Carting his wife, Alison Haag, and newborn daughter, Britney Deas, he trekked back across the nation to a land of fewer diversions and enrolled in the UTK graduate program, with this thesis being the culmination of that activity.

In May 1988 he was hired at his present position as a Development Associate in the Instrumentation and Controls Division of Oak Ridge National Laboratory.