* DMU Disk Controller Register Operation *

and

* State Sequencing *

Burrell Smith

1Ø October 1984

-------- CONFIDENTIAL ---------- CONFIDENTIAL --------

OVERVIEW

The DMU has two DMA channels, Channel Ø, which supports an internal 2Ø Megabyte Nisha Winchester disk drive, and Channel 1, which supports a general purpose high speed serial DMA link. Each channel has enough bandwidth so that the internal Winchester and another peripheral of slightly less than 8 Mhz transfer rate each appear to have concurrent access to the DMU RAM. Double these values when calculating the peak transfer rates available for a one Megabyte memory configuration, since the memory data bus in systems employing the DMU support a 32 bit memory bus.

The Winchester controller (Channel Ø) supports consecutive sector DMA for reads, the common case, without requiring CPU intervention. Writes, which must be performed at two-to-one interleave because of the Winchester's long recovery time following a write (this prohibits reading the header on the following sector) may only be written a sector at a time. This requires the internal AMU registers to be reloaded every 1.2 ms for writes (design change pending to eliminate this and allow sector blocks for writes). The Winchester controller also supports variable-size sectors at single byte increments, up to just under 4K bytes in size, and up to 64 sectors per track.

The General Purpose DMA channel transfers data in a block of up to 32K bytes per transfer, and restricts the transfer to a contiguous block. The General Purpose channel supports laser printing, and other external peripherals. This document presents detailed information only on the Winchester Disk Controller, Channel Ø.

The DMA steals cycles from the CPU for each transfer. Dedicated bandwidth assigned to the CPU guarantees a minimum of approximately 25% of full bandwidth for the 68ØØØ, eliminating processor lockout because of a faulty peripheral.

Although almost all the registers needed for a DMA transaction reside in the DMU, two important registers do not. These are the DMA Memory Pointers, which point to a physical location in RAM as the source or destination for DMA

transfer. Following each transfer, the Memory Pointers increment by one 16-bit word. In 32 bit configurations, the Memory Pointers increment by two 16-bit words. Consult the Turbo Memory Map document for details about how to address these registers. In normal operation, the Memory Pointers will be changed only at the beginning of a transfer. They are write only registers.

CHANNEL Ø REGISTERS

1. Register Addressing

The DMU appears as two word locations: the Address Register and the Data Register. The Address Register selects one of the Data Registers for reading or writing. All registers are read-only or write-only, and the direction of transfer, to or from the Data Register is implied in the register pointed to by the Address Register. Details of how to communicate with these registers is located in the DMU Memory Map document.

2. Registers, Channel Ø (Winchester controller)

A. Sector Length Register (SLR) * write-only *

This register holds the ones complement of the sector length, which may vary from the current 532 bytes up to 4K bytes. Normally, the SLR is set to the number giving 532-byte sector length, which is ***.

B. Sector Number Register (SNR) * read-write, using separate addresses *

This register is the ones complement of the number of sectors to be transferred. The read address allows the progress of the transfer to be monitored by the processor. The sectors desired must be consecutive. When reading the Sector Number Register, it should be read twice to insure a valid reading.

C. Sector Header Latch Ø,1 (SHL Ø,1) * write-only *

These two eight bit latches correspond to the first two bytes of the three-byte sector header format. When formatting the disk, these must be a constant value for all sectors on a given track. The third byte, as detailed below, is really a six bit counter, and increments each sector.

D. Sector Header Counter (SHC) * write-only *

When addressed by the CPU, this six bit counter broadside loads. Following each sector, the counter increments by one, for reads. Assuming that when formatting the disk the sector header counter is loaded with the correct values, then consecutive sectors may be read in as a block. Hardware in the DMU compares the combined three bytes formed by the two sector header latches, and the six bits of the Sector Header Counter (zeroes are appended to the high-order two bits of the Sector Header Counter) with the three bytes of the header as it comes off of the disk. The ECC eight byte data

block which follows the sector data during a read is thrown away except when Read ID is envoked. This requires the sector to be re-read in the case of an ECC error, specifying Read ID instead of Read.

E. Servo Transmit Register (STR) * write-only *

This register is the asynchronous communications link from the DMU to the Winchester disk, and transmits at 38,4ØØ baud with eight data bits and no parity. An interrupt is issued following the transmission of the data byte and two stop bits.

F. Servo Receive Register (SRR) * read-only *

This register is the asynchronous communications link from the Winchester disk to the DMU, and receives at 38,4ØØ baud an eight bit byte without checking parity. An interrupt is issued when the data byte is assembled. The interrupt cannot be masked. * Should it be??? *

G. Interrupt Status Register (ISR) * read-only * (common to both channels)

When a DMU interrupt is issued, a level 1 interrupt is presented to the CPU. Since this level is shared by the SCC, a single DMU register displays status bits for all internal DMU interrupting devices, so a single check of the interrupt register is all that is required to determine whether or not a DMU or SCC interrupt occurred. Interrupt bits relevant to Channel Ø include the Servo Transmit and Receive Register, End Of Transfer (EOT*) and ECCERR* interrupts. For formatting purposes, the Index bit may be read by the software, but currently does not interrupt.

H. Master Command Register (MCR) * write-only *

This register contains three bits which control the operation of the Winchester Disk Controller. Two are CMDØ, CMD1; these set the mode of operation for the controller.

CMD1	CMDØ	: Operation
Ø	Ø	Format: rewrites the entire sector including the sector header.
Ø	1	Write: checks the sector header; and if it matches with the internal sector registers, rewrites the data.
1	Ø	Read ID/ECC: reads the entire sector, including the sector header and the ECC bytes following the sector data. Normally used only to verify the result of a formatting operation or in the case of a hard error, to correct the error.
1	1	Read: checks the header; and if it matches, reads the sector data, stripping away the ECC data.

The third bit in the Master Command Register is the START* strobe. This bit resets the internal DMAIP* (DMA In Progress) bit, which is normally the final

enable for a DMA transfer. At the end of a transfer, the DMAIP* bit is set. The setting of DMAIP* causes an interrupt.

DMU State Sequence: Channel Ø (Winchester Controller)

Both DMU channels have buffering in both the transmit and receive directions. The size of the buffer appears as the natural width of the memory bus, either 16 or 32 bits, depending on the memory configuration of the machine. A soft switch, PRIME*, allows a single DMA transaction to take place in the case of writes. An internal latch is loaded with the memory data from that transaction; this data is then available to the disk drive with essentially no latency. For reads, the read buffer behaves in a similar manner, except that no PRIME* signal is needed, since the data is "pushed" into the read buffer, as opposed to being "pulled" from the write buffer.

All of the logic in Channel Ø operates off of the disk's read/write clock with the exception of the register file and the DMA prioritizers.

Connected to each read and write buffer memory is a shift register of the same length as the width of the read and write buffer memories. This Data Shift Register has tristate bus lines, which are driven in the case of RAM writes, and remain high impedance in the case of RAM reads. The signal for enabling the Data SR bus is therefore CMD1 in the case of Channel Ø.

In parallel with the Data SR is a shift register of equal length, called the Control SR, such that for every bit in the Data SR there is a corresponding bit in the Control SR. These bits amount to "valid" bits; that is, when a bit is set in the Control SR, the corresponding bit in the Data SR is tagged as having valid data. This is conceptually the same as a counter, but with higher performance.

Taps from the Control SR are used by control logic to determine when to perform a DMA transaction. This makes the bus sizing of 16 or 32 bits a matter of selecting alternate tap points from the Control SR. The state of the tap points is used to generate three timing signals: STOREDATA*, which is used during writes to synchronously enable the output buffer to accept data from the Data SR, and generates a DMA request; LDDATASR*, which causes the Data SR to accept data from the input buffer, and also causes a DMA request; and EVERY8BITS*, which is used to clock the Sector Length Counter (SLC).

The Sector Length Counter is used to time the intervals between state transitions. Three actions occur to the Sector Length Counter during the sector transfer. The first action occurs when START* and the sector pulse are valid. This enables the search for the first instance of a one in the data stream. When the one is received, the Sector Length Counter is reset.

Two decodes off of the SLC are used to assist an internal state machine with the state sequencing. These are ENDCOUNT* and HEADERENDCOUNT*. For all but formatting operations, the first three bytes of the six byte header (ID) are shifted into the Data SR and compared with the value of the sector header latches and counters. Then, the next three bytes of the ID are shifted into the Data SR and compared to be exactly unequal to the Sector Header Latches and

Counters. This determines whether or not the current sector can be read or written, or if the current sector should be skipped (header match fail).

For reads, the state machine ignores the write splice following the header, and re-synchs off of the low-to-high transition just preceding the data. At this time, the SLC broadside loads from the SLR. This allows a variable amount of data to be contained in the sector, but will nominally be 532 data bytes. Data passes through the ECC logic, which monitors the state of the state machine, expressed on YØ-Y3. Normally only the data itself is DMAed into the system DRAM, although state "A" of the state machine causes the ECC logic to accept or emit the six byte ECC syndrome, depending on whether we are reading or writing.

In the case of reads, at the end of the transfer, the Sector Header Counter increments, as does the Sector Count Register. In the event of an ECC error or the final state of the Sector Count Register, the DMA terminates.

When writing, the byte following the ID is skipped, and RDGT negated during this time. WTGT is then issued, causing the Ø1ØØ preamble, and then the data, ECC, and trailing ØØ byte to be written.

When writing, the transfer terminates at the end of the sector.

State transitions for the disk occur on the rising edge of the DMACLK. ENDCOUNT* changes at byte boundaries; therefore all state transitions which are based on the time interval since bit sync have a resolution of 8 bits. Start of sector bit sync requires START*, SECTOR, and the 0-1 transition of RDDATA. This makes reading the header of the sector possible.

Read ID/TAG/ECC

This routine is a straight read of the entire sector. It includes reading the six byte header, the tag data and the six byte ECC block.

Event 0: Initial state = 0

We wait for SECTORSTART*.

Event 1: Initial state = F.

SECTORSTART* differentiated resets the SLC, sets us to state F. We delay 12 bytes.

Event 2: Initial state = 7

We now look for the 0-1 transition in the bitstream. The transition again resets the SLC, puts us into state 9.

Event 3: Initial state = 9

We shift in four bytes: a "0" byte, and the first three bytes of the six byte header. The =M* signal will be true if the Data SR contents matches the Sector Header Counter and Latch. This routine ignores the =M signal.

Event 4: Initial state = 9

We shift in three more bytes, the complement of the first three header bytes.

The <>M signal should be true if the three last header bytes are exactly unequal to the Sector Header Counter and Latch. The value of <>M is ignored in this routine.