SECTION IV

INTERRUPT HANDLING AND WRITE DEFERRAL

The 60Hz (50Hz for European version) non-maskable interrupt (NMI) in the ColecoVision system has a wide variety of applications such as providing a fixed time base for the timing software, and a natural debounce interval for the controller interface. However, interrupts can cause problems if not handled properly.

Let us say, for example, that the system is in the midst of a call to PUTOBJ and is, in fact, writing to VRAM when the interrupt occurs. If the interrupt service routine calls for transferring data to another area of VRAM by setting up the VDP address register (auto-increment) to a different value, the pending VRAM operation cannot resume properly after the interrupt is serviced.

The OS contains software which allows graphics operations on the object level to be protected against damage by asynchronous interrupts. It should be stressed that the OS protects ONLY the object level. Routines on the table and chip driver levels could be deferred against interrupt by using the application library routine, DEF_INT, suggested in ColecoVision Bulletin No. 0010 (Appendix D).

In order to implement this protection for graphics operations, the application program must allocate space for a deferral queue. The size of this queue depends on the number of graphics operations that are expected to be performed between NMIs, but usually fifteen entries of three bytes each will prove sufficient. The address of the queue should be passed on to the OS using a routine called INIT_WRITER which also empties the queue and prepares it for operation. Thereafter, whenever the flag byte DEFER_WRITES is set to true (1), calls to PUTOBJ are deferred by placing them on the queue where they may be performed using a single OS call from the interrupt service routine.

In addition to the buffer in which the queue resides, the deferral routines use several defined storage areas in the course of their operation. These are: QUEUE_SIZE, QUEUE_HEAD, QUEUE_TAIL, HEAD_ADDRESS, TAIL_ADDRESS and BUFFER. They are all related to the state of the queue.

4.1 INIT_WRITER

Calling Sequence:

```
        LD    A, SIZE
        LD    HL, LOCATION
        CALL  INIT_WRITER
```

Description:

INIT_WRITER initializes the queue. It does not, in fact, do anything to the "physical" queue in RAM. Instead, it merely sets up its description by setting QUEUE_SIZE to SIZE, HEAD_ADDRESS and TAIL_ADDRESS to the beginning of the buffer and QUEUE_HEAD and QUEUE_TAIL to 0.

Parameters:

SIZE	The size in entries of the queue. SIZE should be equal to the amount

	of space allocated for the queue divided by three. Range for SIZE is 1 to 255.
LOCATION	The location of CRAM area allocated for the queue.

Side Effects:

- Destroys AF.

4.2 WRITER

Calling Sequence:

```
CALL  WRITER
```

Description:

WRITER performs any deferred PUTOBJ operations that may be on the queue emptying the queue as it goes. WRITER should be called by the interrupt service routine.

WRITER uses a "back door" into the PUTOBJ software without ever making an explicit call to PUTOBJ.

Side Effects:

Destroys AF, BC, DE, HL, IX, IY, BC', DE' and HL'.