

# IRM with Crate Utility Board

## *Logic analysis of system code*

Tue, May 22, 2001

IRMs are not usually configured with a Crate Utility Board. With the need of support for a "little console" in a 162-based system, the system logic in this area needs some analysis. This note is an attempt to explain the present logic in order to help decide how to change it to better support a little console on a 162-based system.

During system initialization, the code determines whether a Crate Utility Board is present, and if so, it sets the `VRAMPTR` system global variable to nonzero, its value being the low word of the VME I/O address of the video ram memory on the CUB. From a search of all references to `VRAMPTR`, we find the following:

1. `Abort.a` Following an abort, if `VRAM`, then reset system by setting the auto-reset control line high; otherwise, if we have a 162 board, set a bit in the `VMEchip2` to effect the system reset.

A better plan may be: if a 162 board, reset via the `VMEchip2`; else, if `VRAM`, use the auto-reset control line method.

2. `BeamInh.a` When operating the beam inhibit control line, if `VRAM`, do it via the CUB; else, do it via relays on the Digital IP board.

A better approach here may be to operate the relays on the Digital IP board, if any, and also to activate the CUB control line, if any.

3. `DspInp.a` If `VRAM`, support video ram accesses for the little console; else, skip such accesses, only interacting with the image buffer. Ok.

4. `IntsFP.a` If `VRAM`, set `CYCLEREF` to reading of the 2000 Hz decrementing counter on the CUB; else, use the `CNT2000` byte that is updated at 2000 Hz by a high priority interrupt in the 162-based systems.

It may be better to check for a 162 board first, using the `CNT2000` byte; else, access the 2000 Hz timer register on the CUB.

5. `IntsFP.a` If `VRAM`, restart 40 ms delay timeout using a CUB timer; else, if a 162 system, restart a 162 board timer for this purpose. If instead a 162 always used its own timer for this, even when a CUB exists, then the 40 ms timer in such a CUB should be disabled. Only one 40 ms timer interrupt should be used in a node.

6. `IntsFP.a` The `Get2000` logic accesses the 2000 Hz timer on the CUB or the `CNT2000` global byte based upon the `VRAMPTR`. This is similar to point 4 above.

7. `IntsFP.a` In the `TICKLE` routine, if `VRAM`, pulse the auto-reset circuit on the CUB; else, if a 162 board, reset the watchdog timer by setting bit 21 in the `VMECTRL` register of `VMEchip2`. A better plan here is to activate one or both circuits, depending on which is available.

8. `IntsFP.a` When updating the `OPTIONS` global variable, if `VRAM`, access the switches via the `CUB`; else, read the switch byte from the Digital IP board. Also, access the appropriate 2000 Hz timer byte to get length of current cycle. If a 162, use the emulate countdown global counter; else, use the `CUB` timer register.

A more accurate value for the length of a cycle would be obtained if the  $\mu$ sec counter could be used. The difference would then be converted into  $\mu$ sec units by dividing by 1000. To do this, use a 32-bit equivalent for `CYCLEREF`. One could expect that the result would show a solid `0x85` value rather than alternating with `0x85-0x86`. This plan may be implemented later, if desired. For now, just pick the appropriate byte.

9. `IntsFP.a` If `VRAM`, initiate little console serial access via the `CUB`; else, there is no little console, so do nothing. Ok.

10. `InzSys.a` In addition to determining whether a `CUB` is present by accessing the address of the option switches byte on the `CUB`--at address `0xFFFFA300`--in the case that a `CUB` is present, the low 7 bits of the `SECNOSW` byte of the `CUB` is taken as the low byte of the local node#. This logic may be removed. The complete local node number should be installed in the word at `TRING + 0x046`.

11. `LocApp1.a` When about to initialize a local application whose enable bit has just been set, the auto-reset logic is tickled if `VRAM` present. See point 7.

12. `RdAD.a` When measuring the timing of Data Access Table instructions, if `VRAM`, use the `CUB` 2000 Hz counter; else use the 162 board 1MHz timer.

This can be more accurately done in 162 systems by using the  $\mu$ sec counter instead of the `CNT2000` byte counter.

13. `SResets.a` When performing a system reset, if a 162 board, use the `VMCTRL` register in the `VMEchip2`; else if a 133 board and if `VRAM`, clear the data direction register bit that will ultimately cause a system reset. Ok.

The general approach here is for a 162-based node to use the features it inherently has, even if the `CUB` also has them. Anything relating to the little console display can only be supported when a `CUB` is present. Eventually, 133's will be retired.