82C465MVB Product Update

Revision 1.4 - April 1998

The product updates in this document refer to the OPTi 82C465MVB Single Chip Notebook Solution.

The 82C465MVB part is in full production since Q1 of 1996. OPTi has released an updated Data Book of this part in October 1997 (Document Number: 912-3000-016 Revision 3.0). The following paragraphs discuss limitations of the chipset architecture which are not covered in the updated data book. A thorough understanding of these topics will help ensure a smooth design-in of the 82C465MVB part.

For additional information on any of these topics, please contact OPTi Mobile Applications Engineer, James Castillo at (408) 486-8618. You may also contact Gopa Periyadan at (408) 486-8673.

1. Wrong DMA Address Possible after High Memory Access

The 82C465MVB part incorporates automatic pulldown resistors on the CPU address lines. The chip manufacturing process allows for resistor values between 40k and 60k ohm to be implemented internally. Most of the time, either the CPU or the chipset drives the address lines. The resistors are intended to prevent floating of the address lines when no one is driving, such as during bus hold periods.

During DMA cycles, the DMA controller within the 82C465MVB chipset drives the address on the CPU address bus; the CPU is in hold acknowledge state at this time and does not drive an address. However, the DMA controller does **not** drive address lines above CA23. (Even if it did, the DMA controller still would not be able to control lines CA26-CA30 since they do not connect to the 82C465MVB part.)

If the last CPU access just prior to being put in HOLD was to an address in high memory, one in which any of CA24-CA31 was high, the value on these address lines may remain high when EADS# is generated. If the bus capacitance is high enough, weak internal or external pulldown resistors will not be sufficient to restore the address line value to 0 before the DMA cycle takes place. In this situation, the CPU will see the wrong address when EADS# is asserted, and will not invalidate the proper line in L1 cache.

Hardware Solution

Address lines CA24 to CA31 should be pulled down externally with 2k ohm resistors. This strong pulldown value ensures that the line will return to logic '0' before EADS is asserted.

2. BOFF# Signal in Place of LCLK

The LCLK output signal was originally required on the 82C463MV part because that part required a 2X clock for its internal operation, regardless of whether the CPU clock was 1X or 2X. In the case of a 2X CPU clock, an external 1X clock was still necessary on the VL bus. LCLK was that 1X clock.

On the 82C465MV series parts, the chip uses a 1X clock for its internal operation, regardless of the CPU clock requirement. Because the 1X clock is always brought out externally (on the FBCLKOUT pin), there is always a 1X clock available for the VL bus. Consequently, while the LCLK signal was maintained for backward compatibility with the 82C463MV part, it is not intended for use on new designs. For example, when provisions are made for L2 cache, the pin becomes TAGCS#.

The 82C465MVA chip took the utilization of the LCLK pin a step further and redefines the LCLK pin as BOFF# under a specific configuration:

- No provisions are made for L2 cache (no pulldown resistor on pin 146)

- 82C463MV compatibility mode is not enabled (no pullup resistor on pin 79).

Therefore, the LCLK output is not available at boot time under these circumstances. The final chip design made a provision to work around this setting: setting bit D4h[0]=1 at a later time restores LCLK. Unfortunately, VL-bus devices often require a clock at reset time or else their internal state machines may not reset reliably, preventing them from tri-stating their VL-bus signals. The system can never execute the code necessary to enable the LCLK function under these circumstances.

Hardware Solution

The only reliable solution to this problem is to use FBCLKOUT for the VL-bus clock.

3. Parity Checking

The 82C465MVB part does not support parity checking. Please disregard any sections in the Data Book concerning this support. If not used for L2 cache support, the MP[3:0] pins should be strapped high or low according to the strap-selectable options required.

4. AEN Problem in 16-bit DMA

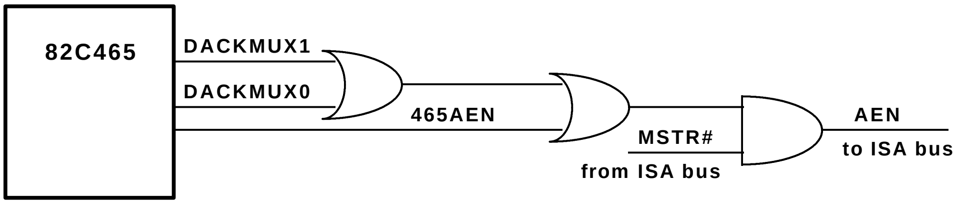
The 82C465 series drops AEN prematurely during 16-bit DMA transfers. The AEN signal should remain high at least for the duration of the ISA command lines (IORD#/IOWR#, MRD#/MWR#) during DMA. But AEN will drop low mid-way through the command, possibly allowing an I/O device to decode the transfer as a standard input/output command.

The 82C465 chip itself can decode this cycle and hang the system under certain circumstances. Generally this failure mode is evident only when writeback CPU operation is enabled.

There are several possible workarounds to this issue.

- If there is no possibility of 16-bit DMA occurring, no workaround is required.
- If no local ISA devices will misinterpret the DMA transfer as an I/O access, it is sufficient to simply set register 43h[2:0]=000 to enable synchronous ATCLK operation. The internal logic of the 82C465 chips will not misinterpret the DMA access as an I/O access when in synchronous mode.
- If full ISA support is expected, the fix shown in the figure below must be implemented using two 7432 gates and a 7408 gate. Note that if ISA masters are not supported, the 7408 gate is not required.

Figure 3. Correcting AEN for 16-bit DMA



5. Unreliable Hidden Refresh Operation

The 82C465MVB chip brought in a new feature, hidden refresh, through register 32h[2]. Unfortunately, enabling this feature causes system hangs with EDO DRAM. For the moment, it is recommended to disable this feature by setting register 32h[2]=1 when using EDO DRAM.

According to DRAM data sheets (Micron, for example), DWE# (high) needs to have setup time twrp and hold time twrh with respect to the RAS falling edge for a CAS before RAS refresh. The 82C465MVB fails to meet that when the chip is in hidden refresh mode.

The reason is as follows. When EDO timings are turned on, a short DWE# pulse comes out at the end of the cycle after BRDY# is deasserted. This is to turn off the memory data bus at the end of the cycle. When normal ISA refresh is selected, this DWE# pulse does not appear since the CPU is on hold during the refresh. But when hidden refresh is turned on, the CPU is no longer in hold during the refresh. This leads to the DWE# appearing anywhere, and could possibly be active at the falling edge of RAS# during refresh.

Not all DRAMs have this restriction on DWE# being inactive on a refresh. So some EDO DRAMs could work fine with hidden refresh enabled.

6. Incorrect Hidden Refresh Description

Incorrect or confusing information is documented at three places in the data book: Under section 4.6.1.4 on page 73, inside the table 4-51 on page 73 and inside the table 5-1 on page 157. Correct Bit 32h[2] description is as follows: Hidden Refresh on AT Bus - 0=Enable, 1=Disable. When Hidden Refresh is enabled, HOLD and HLDA CPU bus request/grant pair will not be used to hold the CPU bus while the DRAM refresh is being performed by the DRAM controller in a transparent manner. Whether Hidden Refresh is enabled or disabled, the REFRESH# signal on the ISA bus will continue to be active for the ISA refresh cycles.

It should also be noted that when EDO is used in the system, Hidden Refresh feature must be disabled by setting 32h[2] = 1.

7. Pulldown Resistor Recommendation

Low-power designs using the AMD 5x86/133 CPU should have pulldown resistors on all CPU data and address lines. The internal pulldown resistors on the 82C465MVB chip are not strong enough to keep the lines low during bus hold states (such as Suspend).