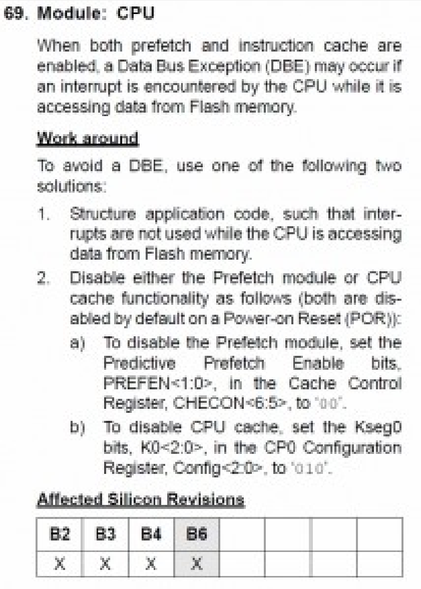
Magnus Errata Part 1

Posted on March 28, 2012

Most of the time, microcontroller errata detail some minor problem with an obscure corner of the chip that only affects a few people. Sometimes though it's much more serious. Recently, Microchip released errata documentation, covering pretty much the entire PIC32 line, with not one but two major CPU issues. The scope of these issues affects the vast majority of PIC32 projects planned, under development and in the field. This is serious stuff. This article covers the first major issue: Reading data from flash while interrupts are in use.



69. Module: CPU

When both prefetch and instruction cache are enabled, a Data Bus Exception (DBE) may occur if an interrupt is encountered by the CPU while it is accessing data from Flash memory.

Work around

To avoid a DBE, use one of the following two solutions:

1. Structure application code, such that interrupts are not used while the CPU is accessing data from Flash memory.
2. Disable either the Prefetch module or CPU cache functionality as follows (both are disabled by default on a Power-on Reset (POR)):
   a) To disable the Prefetch module, set the Predictive Prefetch Enable bits, PREFEN<1:0>, in the Cache Control Register, CHECON<6:5>, to '00'.
   b) To disable CPU cache, set the Kseg0 bits, K0<2:0>, in the CP0 Configuration Register, Config<2:0>, to '010'.

Affected Silicon Revisions

B2	B3	B4	B6				
X	X	X	X				

— Flash Data Read Errata

In a nutshell this issue states that when reading data from flash memory (as opposed to fetching instructions from flash) there is a chance of a serious error (data bus exception) if interrupts occur at inconvenient times. Let's see if this is serious. To have a problem, an application would have to read data from flash and it would have to use interrupts. Without any data to back this up, but based on my years of experience, I would estimate that this issue could affect almost all PIC32 applications!

Given how serious this problem seemed to me, I was perplexed that there was not a lot more being said about it on the discussion forms. So I raised the issue in a posting. This generated a fair bit of interest. For starters, I was NOT the first on the block for this issue. In my defense the original thread was called “PIC32 SPI fail only at PBclk= 40 Silicon Problem??” which is not exactly shouting “memory errata”. I was correct that many others were concerned:

kalpak wrote:

“For a new project the scary and vague errata drove me into the arms of NXP”

bsder wrote:

“…

*Errata 44 (for the 5/6/7 series) lacks a *LOT* of information. Why can’t we just shut the interrupts off? Is the handler the issue? Is the specific mask bit the issue? Is the overall interrupt an issue? The fact that we *don’t* see these every single time says that it’s not *just* an interrupt issue going on here.*

In addition, why should their workaround be any more reliable? Maybe it is, but there is a whole bunch of arbitration issues that come in, and I’ve sometimes seen quirky DMA behavior. The fact that they couldn’t interface SPI to the MIPS core certainly doesn’t give me confidence in their DMA engine.

This one is bad. 🙁*“*

crosland wrote:

“Does anyone form Microchip monitor these forums? What’s the plan to fix the most serious CPU errata? It’s simply not acceptable to have to disable interrupts when reading flash nor use DMA to access peripherals. Imagine the task to go through and sanitise the application libraries…”

Finally after nearly a week, the moderators took notice and made the following statement:

“One explanation why most people do not see DBE errata in their applications is that it can only appear if both cache and prefetch are enabled. By default (on chip reset), both features are disabled, and out of the box start-up code from Microchip doesn’t enable them. So, one has to understand these features and initialize them to enable them (or use provided PLib functions).

When both features are enabled, application performance will increase, as much as double (average execution time will be half, vs the same app without cache and prefetch turned on). However, if only one of them is enabled (as workaround suggests), performance increase will be slightly less (on average 90% instead of 100%).

Very minimum impact.

Microchip understands importance of having these issues fixed and is actively working on it. Please check with your local Microchip representative on availability dates and obtaining early samples (if desired).“

Well I guess that helps, except I have always fully enable all the speed-ups available, so why have I not had any problems? Still, it got me thinking; there are two possible courses of action: A) Disable the Prefetch or B) Disable the Cache. Which choice would do less damage to system performance? To help understand this I wrote a test program that drew 100,000 circles on a graphical display. I then tried four sets of options:

Test	Execution Time	% Slower
None Disabled	79.1	–
Cache Off	84.3	6.5%
Prefetch Off	82.3	4.0%
Both Off	120.2	52.0%

My data seems to support the official line. It looks like performance is slightly less affected by turning off the prefetch, but not by much. Just turn (or leave) both off. The only remaining sticking point (in this case a point shaped 800 pound gorilla) here is my inability to make this issue happen at all! If it is so serious why do I never see it? I still am no closer to an answer. I don’t know what to say. Clearly there are other conditions/setting/configurations that play a role in this issue that are not called out in the errata. Without more complete information, informed choices are hard to come by. However, safe ones are not. Given the slight impact, it’s safe to say that the instruction prefetch can be turned off with only a minor ding in performance, and for now, that is what I shall do.

Due to the dynamic nature of the situation, I have NOT included links to errata files. For up-to-date information, please go to the Microchip website and look up the PIC32 of interest.

As always, comments and observations are welcomed.

Peter Camilleri (aka Squidly Jones)

SHARE THIS:

Print LinkedIn Facebook Twitter

LIKE THIS:

Loading...

This entry was posted in **Embedded Development**, **Errata** by **Peter Camilleri**. Bookmark the **permalink [http://teuthida-technologies.com/?p=562]** .