A Tale of Two Hammers:
A Brief Rowhammer Analysis of AMD vs. Intel

May 2016

Mark Lanteigne, CTO and Founder, Third I/O Inc.

This is the first addendum to Third I/O's March 2016 Rowhammer report and can be found here: www.thirdio.com/rowhammer.pdf

FOREWORD

Rowhammer is a DRAM failure that is observable on silicon that is 32 nm in size or denser. The intention of this paper is to research and determine whether or not AMD's Piledriver architecture has better resilience to this anomaly. For simplicity, we decided to test a known failing DDR3 DIMM in both an AMD and Intel platform. This particular module appears to pass all memory diagnostics with the exception of Rowhammer tests (I.E Rowhammer.js). When this DIMM is hammered, it has shown an unusually high frequency of bit flips.

The original Rowhammer research paper from Yoongu Kim and Carnegie Mellon University (CMU) had a curious finding. In this paper, released in 2014, they showed several x86 systems that were exhibiting Rowhammer bit flips at variable levels while testing one problematic DIMM. One of these systems was an AMD Piledriver which showed 59 bit flips with a performance level of 6.1 MB/s. The CMU researchers also reported on three Intel processors which exhibited between 16,100 and 29,200 bit flips with performance levels ranging from 11.6 to 12.3 MB/s.

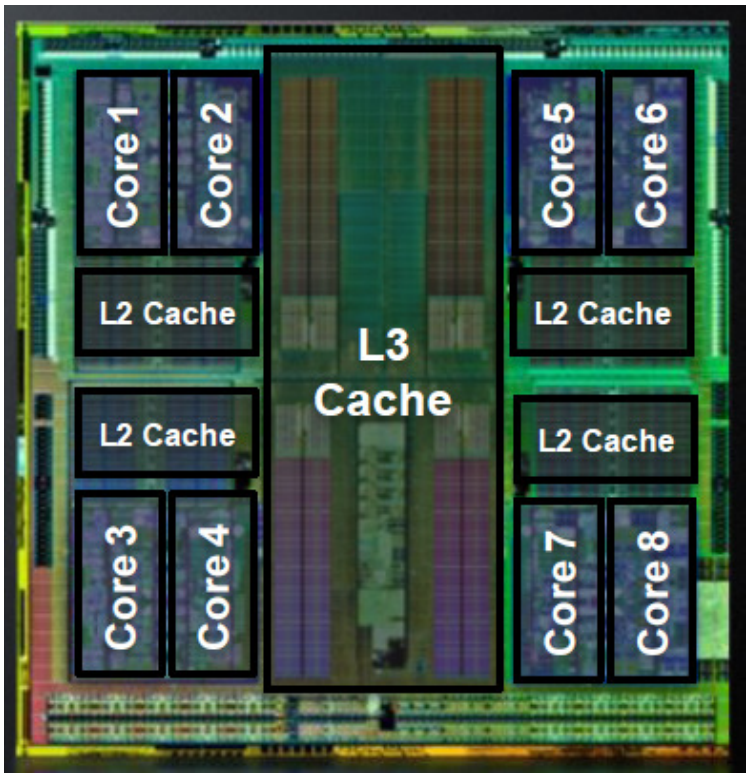
Number of Disturbance Errors

CPU Architecture	Errors	Access-Rate
Intel Haswell (2013)	22.9K	12.3M/sec
Intel Ivy Bridge (2012)	20.7K	11.7M/sec
Intel Sandy Bridge (2011)	16.1K	11.6M/sec
AMD Piledriver (2012)	59	6.1M/sec

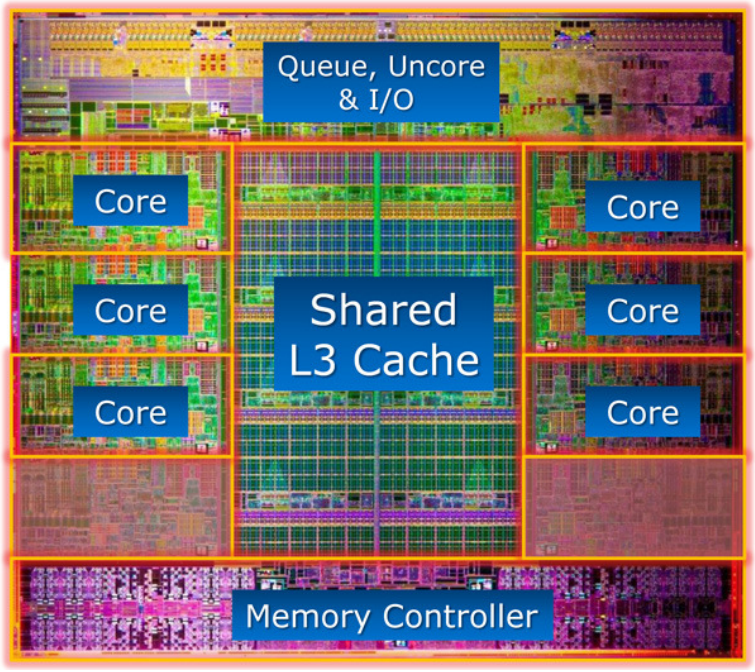
Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors, (Kim et al., ISCA 2014)

From the above chart, it seems as though every generation of Intel processor becomes a naturally more aggressive variable in creating additional Rowhammer bit flips. On the converse, it appeared that AMD's Piledriver was nearly immune to this condition. We can clearly see that Intel shows higher performance by approximately 2X, but it is also showing nearly 400X the number of bit flips.

As Third I/O has primarily been testing Intel based systems, we decided to acquire a modern AMD Piledriver system. We chose the AMD FX-8370 (Vishera) as it was released in 9/2014 and it should represent the latest AMD technology in high core count desktop processing.



AMD's Vishera (FX-8370) CPU is divided into four compute modules. Each module consists of two x86-64 processing cores and share an L2 cache. All cores share a common L3 cache.



In contrast, Intel's 3960x (Sandy Bridge E, released 11/2011) has six distinct cores that each utilizes independent L1 and L2 caches and share a common L3 cache.

OUR TESTING METHODOLOGY

In order to achieve the closest Intel to AMD comparison, we performed the following tasks:

We identified a "worst case" DIMM that was highly sensitive to Rowhammer bit flips. The module below has shown thousands of bit flips per minute when subjected to a regional Rowhammer attack.

The above G.SKILL DIMM fails Rowhammer test cases in default and all XMP modes

From our research, there is no "apples to apples" Intel processor that is comparable to AMD's Vishera. Intel's first native 8-core desktop uses DDR4 memory, whereas AMD uses DDR3. So, to create a reasonably comparable Intel system, we simply took a 6-core Sandy Bridge E, disabled two cores, and enabled Hyperthreading. We also overclocked the Intel to run at 4.0 GHz, the same default speed as the tested AMD. In this manner, we have eight Intel processing cores (4 physical, 4 logical) that mimic AMD in the sense that both systems have eight (OS visible) processing cores that share 4 L2 caches.

We also used the DIMM's native XMP settings to achieve the same base memory settings.

Third I/O's Memesis was written in native C and assembly language. All commands and routines are x86 generic, so our code can be run on any x86 compliant system. We also compile our tool using the generic gcc compiler in Linux, so we do not optimize in favor of any particular vendor's CPU optimizations.

In order to simplify our testing, we created a Rowhammer script that runs for approximately 20 minutes and it investigates the following test cases:

1) Single Sided 2 MB Regional Rowhammer (Reads) 3 Million Hammers Per Address
2) Double Sided 2 MB Regional Rowhammer (Reads) 2 Million Hammers Per Address
3) Quad Sided 2 MB Regional Rowhammer (Reads) 1 Million Hammers Per Address
4) Double Sided 2 MB Regional Rowhammer (Non-temporal writes) 2 Million Hammers Per Address
   (Non-temporal (NT) writes are explained later in this report)

RESULTS

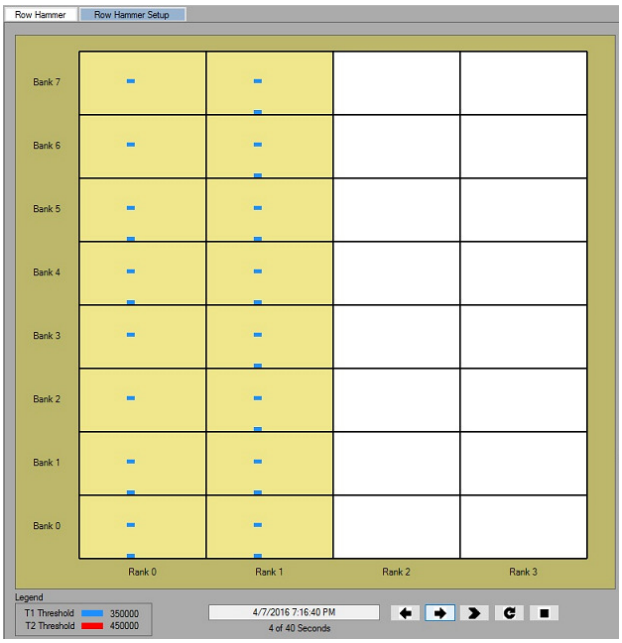
Our first test case was a scan running both all and half of the available cores as seen here:

	Total Bit Flip Count	Single Sided Bit Flips	Double Sided Bit Flips	Quad Sided Bit Flips	NT Double Sided Bit Flips
Intel 8 Cores	10807	9600 at 623 KB/s	570 at 869 KB/s	630 at 1 MB/s	2 at 116 KB/s
Intel 4 Cores Logical	17877	3600 at 343 KB/s	3300 at 526 KB/s	10920 at 1 MB/s	32 at 114 KB/s
AMD 8 Cores	8564	5300 at 114 KB/s	600 at 22 KB/s	110 at 45 KB/s	2500 at 228 KB/s
AMD Cores 2,4,6,8	31468	8300 at 114	5900 at 11 KB/s	840 at 22 KB/s	14600 at 183 KB/s

In our original Rowhammer research, we discussed how multithreading was an effective Rowhammer catalyst. However, we also stated that the best case thread count was from 2 to the maximum core count with each thread pinned to a specific core. As observed in the above results, this is true on both Intel and AMD.

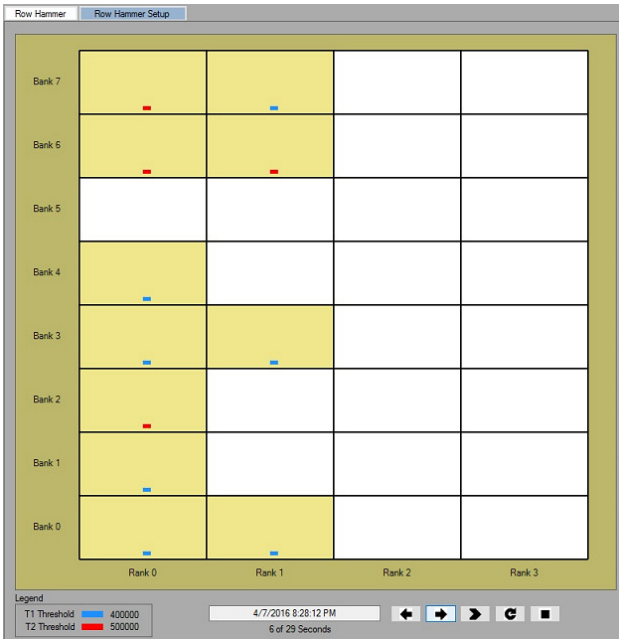
Both Intel and AMD show measurably more bit flips when using only the secondary cores. Total error counts are ~ 65% higher for Intel, but we see a 3.68X increase when we only utilize the secondary 4 cores on AMD. The two most notable results from above are the ~ 11,000 bit flips from Intel in quad sided Rowhammer, but much more interesting is the 14,600 bit flips seen on AMD using non-temporal (NT) writes. In this case AMD shows higher bit flips while also showing lower performance.

We asked Barbara Aichinger of Futureplus Systems to provide real time analysis of these two test cases. Her analysis and screenshots were valuable in determining why these two test cases were catalyst for such high failure counts. Let's begin with AMD's NT results:



FuturePlus' DDR Detective shows 28 unique addresses, the blue dashes, being Rowhammered at over 350,000 ACTIVATEs per 64 ms retention cycle. That's an amazing number of 9.8+ Million DIMM ACTIVATEs. This is a textbook example of a Memesis Regional Rowhammer attack.

Intel's Quad Sided (cached reads) Hammer Results on 4 Hyperthreaded Cores is here:



In contrast, Intel is showing fewer addresses subject to a Rowhammer attack. However, the blue dashes above correspond to 400+k and the red to 500+k ACTIVATEs per retention cycle. This corresponds to 4.8+ Million total ACTIVATEs per retention cycle, which is less than half of what we see from AMD above.

Let's Repeat the Above, But With a "Good" DIMM

As stated earlier, the majority of our early Rowhammer testing was solely based on Intel systems. During our months of research, we began to collect failing memory modules. Some of these DIMMs fail in catastrophic manners, while others might simply allow for a bit flip every few minutes or so. CMU's original research suggested that 85% of DDR3 DIMMs fail in some manner, so a distribution of mild to severe sensitivities to Rowhammer should be expected.

The above Viper Xtreme DIMM fails Rowhammer test cases in default and all XMP modes. Failure rates have generally been 0 to 30 bit flips based on overnight test results on multiple Intel systems.

	Total Bit Flip Count	Single Sided Bit Flips	Double Sided Bit Flips	Quad Sided Bit Flips	NT Double Sided Bit Flips
Intel 8 Cores	2	1 at 640 KB/s	0 at 869 KB/s	1 at 1 MB	0 at 137
Intel 4 Cores Logical	0	0 at 343 KB/s	0 at 526 KB/s	0 at 1 MB	0 at 114 KB/s
AMD 8 Cores	75	65 at 114 KB/s	7 at 22 KB/s	0 at 45 KB/s	3 at 228 KB/s
AMD Cores 2,4,6,8	325	67 at 91 KB/s	84 in 12 KB/s	6 at 22 KB/s	168 at 182 KB/s

In 2014, research would have had us believe that Rowhammer was 400x more likely on Intel versus AMD. But in this one particular test case, we can see that Intel is showing 2 errors versus AMD's 325. Here we see a 163X delta in favor of Intel. The only change was DIMM replacement. Rowhammer is a memory problem and it can change in behavior when analyzed across multiple systems.

WERE THERE FLAWS IN CMU'S RESEARCH?

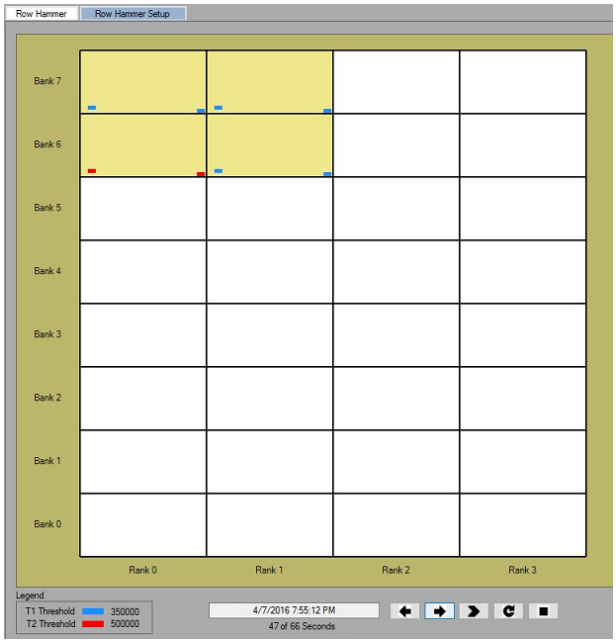
At this point, it's pretty obvious that both Intel and AMD are highly capable Rowhammer aggressors when exposed to Memesis. So, how do we explain the original CMU research and the huge delta in bit flips between AMD and Intel? Simply put, the researchers at CMU had two primary findings:

1) They created an FPGA based DIMM tester that could surgically and precisely perform Rowhammer testing on individual DIMMs. This means that they were performing Rowhammer in an efficient manner. Their best findings in regards to maximum bit flips and aggressive data patterns were performed using an FPGA.
2) Their Intel vs. AMD comparison used a modified Memtest86+ for a proof of concept that simply scratched the surface of Rowhammer vulnerabilities. Since their initial publication, researchers from all over the world have been identifying modern methods for exposing Rowhammer in unique manners.

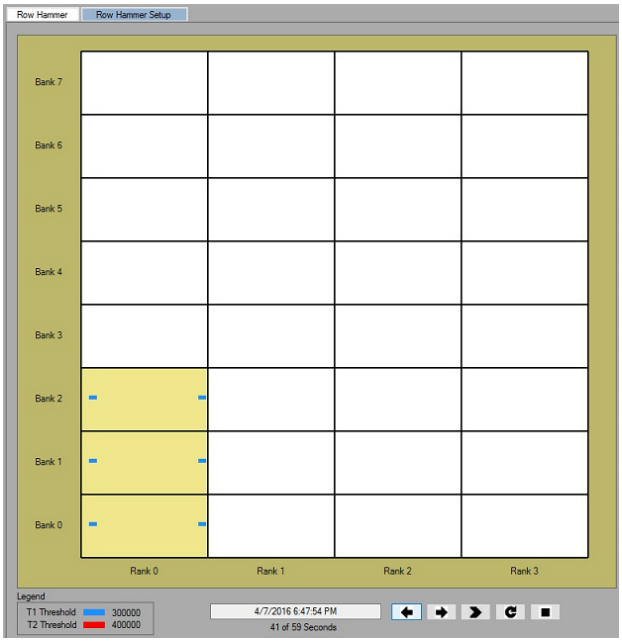
From Third I/O's analysis, we believe that Memesis is now finding more Rowhammer issues in general for the following reasons:

1) Memesis focuses on small regions of memory
2) Memesis uses multithreaded attacks to these memory regions
3) Memesis utilizes data patterns other than all ones or zeroes
4) Memesis has many Rowhammer test cases that focus on memory reads, writes, and uses a number of novel ways to bypass, evict, or flush the processor's caches
5) We have learned a great deal from constant testing and from several papers that have been written on Rowhammer in the past 2 years
6) We have access to DDR protocol analyzers and analysis software that allow us to quickly determine if our test cases are creating Rowhammer activity at the DIMM level

To be clear on this topic, we believe that CMU wrote a paper that was fully accurate and was groundbreaking in content. Their initial x86 diagnostic did exactly what it was intended to do; it exposed Rowhammer bit flips on x86 processors. And when you look at their code running in a real time Rowhammer manner, you can see that it was truly well written for a single threaded test application.



The above graphic was taken from an analysis of the Intel system. This test case showed 174 bit flips on the "bad" DIMM. The precision of CMU's original Rowhammer test is easy to observe using FuturePlus' DDR Detective. You can see that bank #7 has 4 addresses that are being hammered at 1.4+ Million ACTIVATEs. Bank #6 is being stressed even more as it has a couple of hot spots that brings it up to 1.7+ million ACTIVATEs per retention cycle.

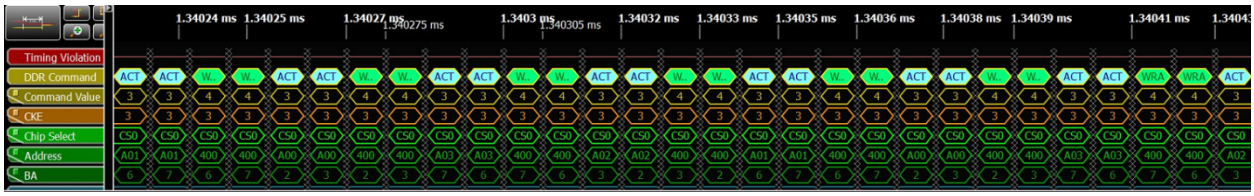


CMU's Hammer does not aggravate the AMD as aggressively. Fewer addresses are being hammered and the ACTIVATE count is greatly reduced. Rowhammer on Intel is showing 3.1+ Million total ACTIVATEs while AMD is only showing an aggregate of 1.8+ Million. From what we see of the above capture, it appears that AMD simply does not generate enough ACTIVATEs to aggravate Rowhammer. This test case showed only one bit flip using the bad DIMM.

THE ACHILLES' HEEL OF AMD

In March of this year, we exposed DDR4 Rowhammer vulnerabilities when tested on the Intel Skylake processor. It is our opinion that the identification of specific "killer" data patterns was a catalyst in exposing bit flips on vulnerable memory.

During our analysis of AMD, we could not identify any killer data patterns that were more effective than random or pseudo-random patterns. However, we think that our use of non-temporal writes expose a profound weakness in AMD's memory controller. Specifically, we found that running solely on either the even or odd cores, we can create a worst-case Rowhammer environment. From trace analysis, it appears to sometimes achieve a nearly theoretical worst-case scenario for a double sided attack. This can best be illustrated by looking at a waveform of this test run on cores 2,4,6,8 and was captured using the Lecroy Kibra 480 analyzer:



The above waveform was captured on the "good" DIMM that showed 168 bit flips during a 5 minute run of a double sided non-temporal write Rowhammer attack. The ACT bars show ACTIVATEs and they represent 52% of the screenshot traffic. Furthermore, there are 7 back-to-back ACTIVATEs to banks 6&7 and 2&3. It is safe to say that such an environment is highly conducive to creating Rowhammer bit flips, even on reasonably stable memory. Our test results back up this statement and screenshot.

About Non-Temporal (NT) Commands

Third I/O first started using NT commands, sometimes referred to as streaming commands, during our early Memesis development. These early tests were initially based on SSE2 instructions and has since been updated to support modern 128 (SSE4.x) and 256 (AVX) bit transfers. We were exploring unique technical methods and manners for bypassing CPU cache while still maintaining high performance. We found that NT commands were highly effective in some scenarios, but that they were not always reliable in creating memory bus activity. Write combining (WC), for example, could prevent tests such as Rowhammer from being performed as back-to-back NT writes to the same address could be cached. After several months of research, we began to realize how to maximize the effectiveness of NT operations and some of our most successful test cases utilize these commands.

NT commands have long been supported on both Intel and AMD processors. And the vast majority of Memesis test cases (both Rowhammer and traditional) employ both cacheable and NT commands. In the case of our Rowhammer NT test cases, we use 100% NT writes for all hammer operations.

Prior to our analysis of AMD systems, we considered the NT tests to be mostly weak and ineffective as they were not the best Rowhammer aggressors on Intel processors—they found fewer bit flips than read

Rowhammer attacks. These recent revelations on AMD systems have renewed our interest in non-temporal Rowhammer tests, so we will begin to explore new methods for integrating these new ideas into our Memesis suite.

Our next area of research is simply investigating the latest public Rowhammer disclosure. In April of 2016, Mark Seaborn of Google and Rui Qiao of Stonybrook released a paper on the usage of non-temporal commands as a manner to induce Rowhammer events. Their research uses a new approach where they follow non-temporal commands with either a cached read or write. They claim that this concept can flush the WC buffers and thereby force ACTIVATEs down to the DRAM.

We suggest that our readers familiarize themselves with this paper as it identifies some of the better real world implications in regards to exposing Rowhammer in the wild. Their paper can be found here:

http://seclab.cs.sunysb.edu/seclab/pubs/host16.pdf

In conclusion, it is Third I/O's opinion that if you have a computer system that contains a CPU, then it can either organically or synthetically be turned into a Rowhammer aggressor. As we have just shown, AMD and Intel share a common instruction set, but they respond in a radically different fashion to our software. Rowhammer is highly reproducible on both vendors' processors, but it is obvious that exposing this issue requires a variety of test cases to identify this issue across different system architectures.

About the Author and Research:

Mark Lanteigne is the CTO and founder of Third I/O Inc. He has been involved in enterprise test and test tool development since 1996. He was previously co-founder and the Director of Test and Test Tool development at Medusa Labs. And prior to Medusa, Mark was the lead test engineer for the Dell Poweredge product test team.

This paper would not have been possible without the technical expertise and assistance of Dr. David Schinke and George Pee, both of Georgetown, Texas. David has always been a brilliant coder; able to transform complex ideas into beautifully clean, functional, and highly effective code. And George has always helped out David and Mark when they lose hope at finding solutions to complex problems. We especially commend David Schinke's NT code which seems to be the best aggravator of AMD's internal memory controller.

And finally, we would like to thank our friends Cindy Stap, Bruce Wagner, Mike Connell, and Jim Marrone. They witnessed our findings before we realized what we had truly discovered. And special thanks to Henry Bruns for his numerous contributions over the years.