BPU Affect on Tigerlake, Icelake, and Skylake

- See testing conditions described in previous writeup
- Tigerlake CPU
- Icelake CPU
- Skylake CPU

Comparison All Three

We can essentially see the exact same repeated trend for all three processors here:

Results for `sz` in `[0, 32]` implementation on Skylake, Icelake, and Tigerlake:

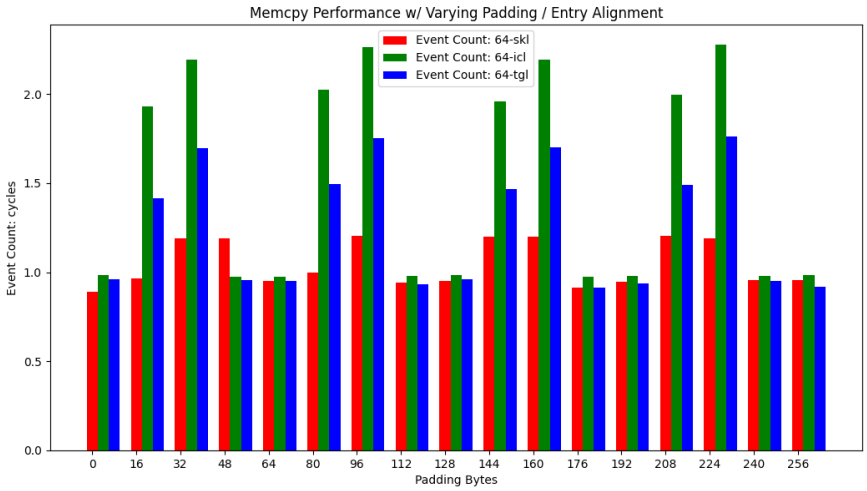

Figure 1: $> python3 graph-results.py -a -f all-results-32.txt –levents cycles –alignments 64 –pmod64 16,32,48,64 –pmod2 0 –pmax 256 –save –cpu skl,icl,tgl –norm

The above values are normalized to median of all runs on the processor and we can see that while Skylake appears to be affected by the same pattern, it is by far affected the least.

All three essentially follow the same conditions for “slow” vs “fast” mode:

The `ALIGN_ENTRY` and `address_of(L(less_vec)) % 64` are as follows:

ALIGN_ENTRY	0	16	32	48
0	SLOWER	SLOWEST	FAST	FAST
16	SLOWER	SLOWEST	FAST	FAST
32	FAST	FAST	SLOWER	SLOWEST

ALIGN_ENTRY	0	16	32	48
48	FAST	FAST	SLOWER	SLOWEST

Which again boils down to the following formula: `32 & (address_of(L(less_vec)) - address_of(memcpy))`

Although note a distinction in Skylake from Icelake and Tigerlake is that **Skylake does not have a clear distinction between Slower and Slowest mode**.

As well while it appears **branch misses are the root cause on all three CPUs**, though **on Skylake the pattern is significant different**.

As you can see the results on Skylake are much less binary. **On skylake the change in branch-misses is within a factor of 2**. Whereas **On Tigerlake and Icelake the change in branch-misses is factor of 10^4**.

Tigerlake & Icelake

**Observed nearly identical pattern of performance vs `ALIGN_ENTRY` and `PADDING % 64` on *Tigerlake* and *Icelake*.

Results for `sz` in `[0, 32]` implementation on Tigerlake:

Results for `sz` in `[0, 32]` implementation on Icelake:

Final **the reason for the variance on both Tigerlake and Icelake appears to be branch misses**:

Note: Icelake appears to experience a proportionately more significant slowdown and greater increase in branch misses:

Skylake

The Skylake versions appear to suffer from the same issue although to a signficantly reduced magnitude:

Results for `sz` in `[0, 32]` implementation on Skylake:

Note that Skylake appears to have two different trend, one with NOP padding in range `[0, 96]` and another with NOP padding in range `[112, ...]`. The reason for this is that in the `[0, 96]` range the `jb L(less_vec)` can use the 2-byte encoding. After `NOP padding == 112`. This end up adding an additional `16 bytes` to the padding.

And again it appears for **Skylake to be related to branch misses**:

Although again the change in branch misses from “slow” to “fast” mode is not nearly as severe as on Icelake or Tigerlake.

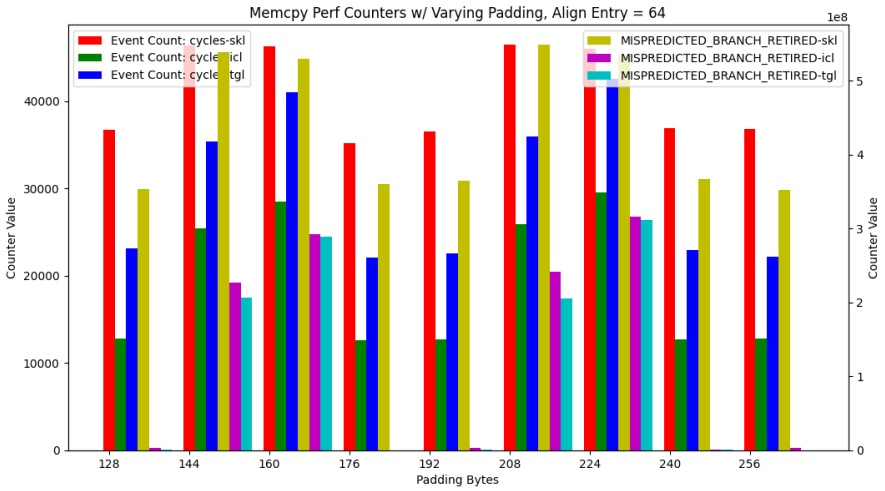

Figure 2: $> python3 graph-results.py -e -f all-results-32.txt –levents cycles –revents MISPREDICTED_BRANCH_RETIRED –alignments 64 –pmod64 16,32,48,64 –pmod2 0 –pmax 257 –save –cpu skl,tgl,icl –pmin 128

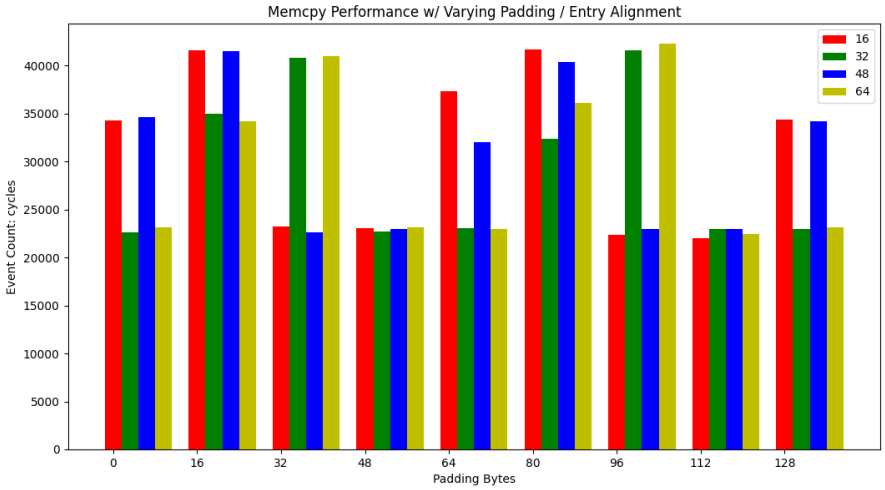

Figure 3: $> python3 graph-results.py -a -f all-results-32.txt –levents cycles –alignments 16,32,48,64 –pmod64 16,32,48,64 –pmod2 0 –pmax 129 –save –cpu tgl

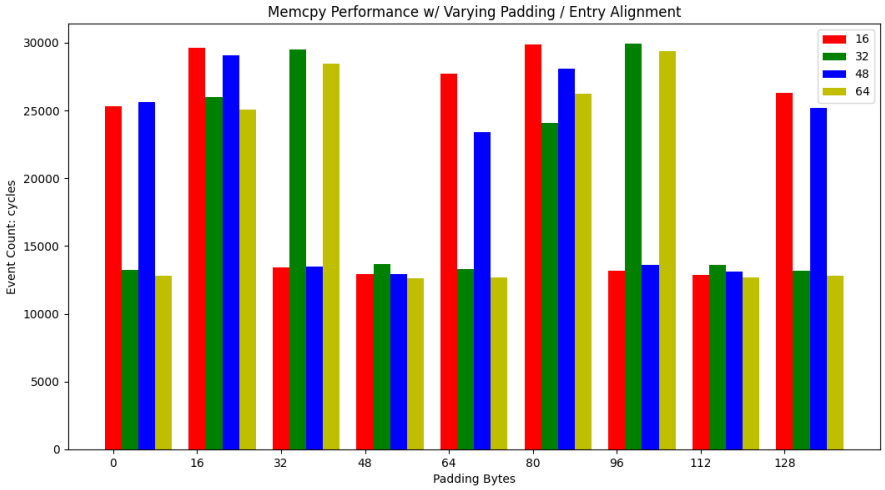

Figure 4: $> python3 graph-results.py -a -f all-results-32.txt –levents cycles –alignments 16,32,48,64 –pmod64 16,32,48,64 –pmod2 0 –pmax 129 –save –cpu icl

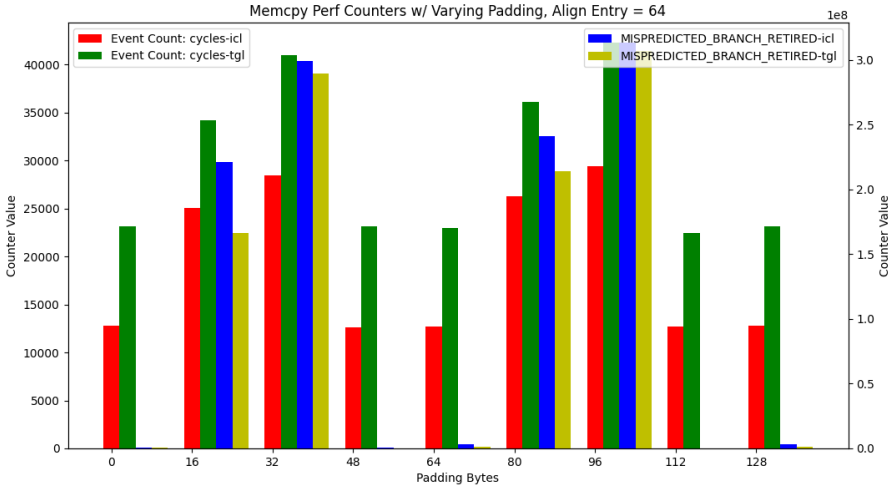

Figure 5: $> python3 graph-results.py -e -f all-results-32.txt –levents cycles –revents MISPREDICTED_BRANCH_RETIRED –alignments 64 –pmod64 16,32,48,64 –pmod2 0 –pmax 129 –save –cpu tgl,icl

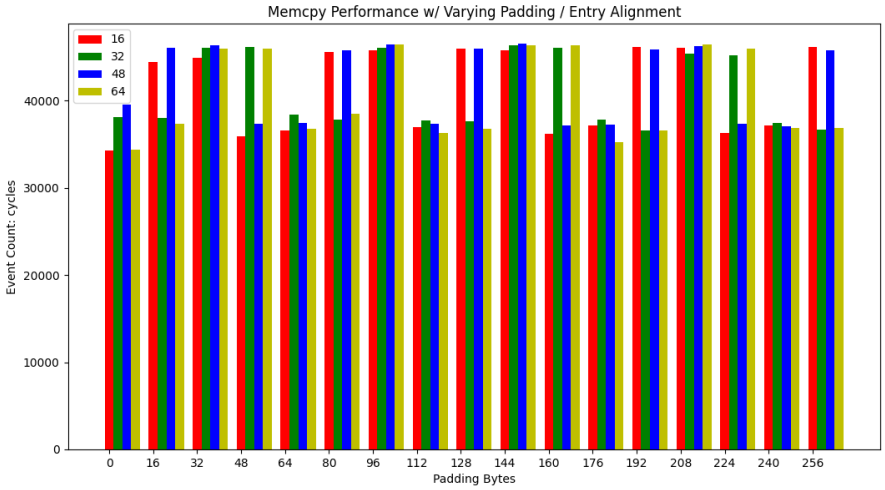

Figure 6: $> python3 graph-results.py -a -f all-results-32.txt –levents cycles –alignments 16,32,48,64 –pmod64 16,32,48,64 –pmod2 0 –pmax 256 –save –cpu skl

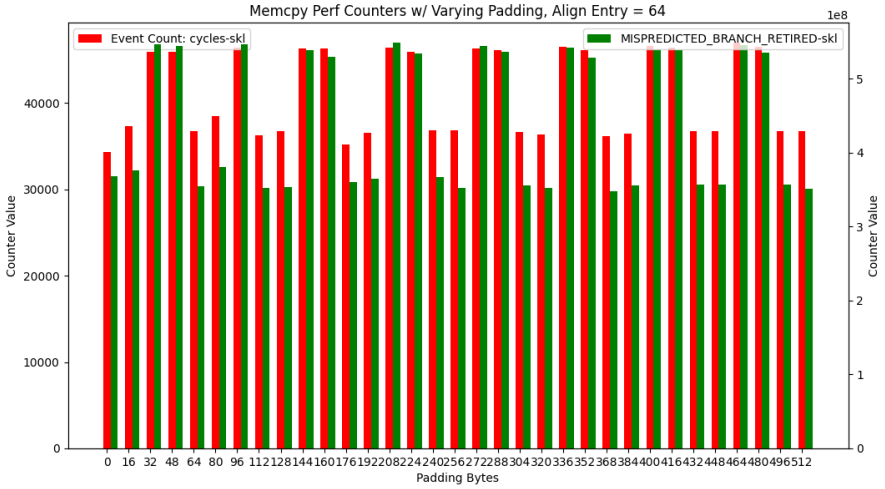

Figure 7: $> python3 graph-results.py -e -f all-results-32.txt –levents cycles –revents MISPREDICTED_BRANCH_RETIRED –alignments 64 –pmod64 16,32,48,64 –pmod2 0 –pmax 513 –save –cpu skl

And I think we can rule out to Uop Cache as `IDQ_DSB_UOPS` seems to scale 1-1 with `UOPS_ISSUED_ANY` which is indicative of repeated uops due to branch misses.

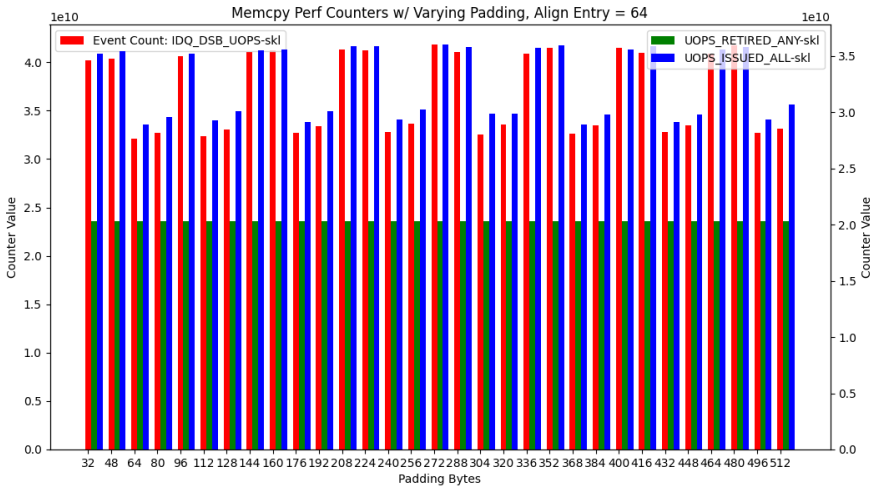

Figure 8: $> python3 graph-results.py -e -f all-results-32.txt –levents IDQ_DSB_UOPS –revents UOPS_RETIRED_ANY,UOPS_ISSUED_ALL –alignments 64 –pmod64 16,32,48,64 –pmin 32 –pmod2 0 –pmax 513 –save –cpu skl