# Reconfiguring Issue Logic for Microprocessor Power/Performance Throttling

Edwin Olson, Dave Maze, Andrew Menard

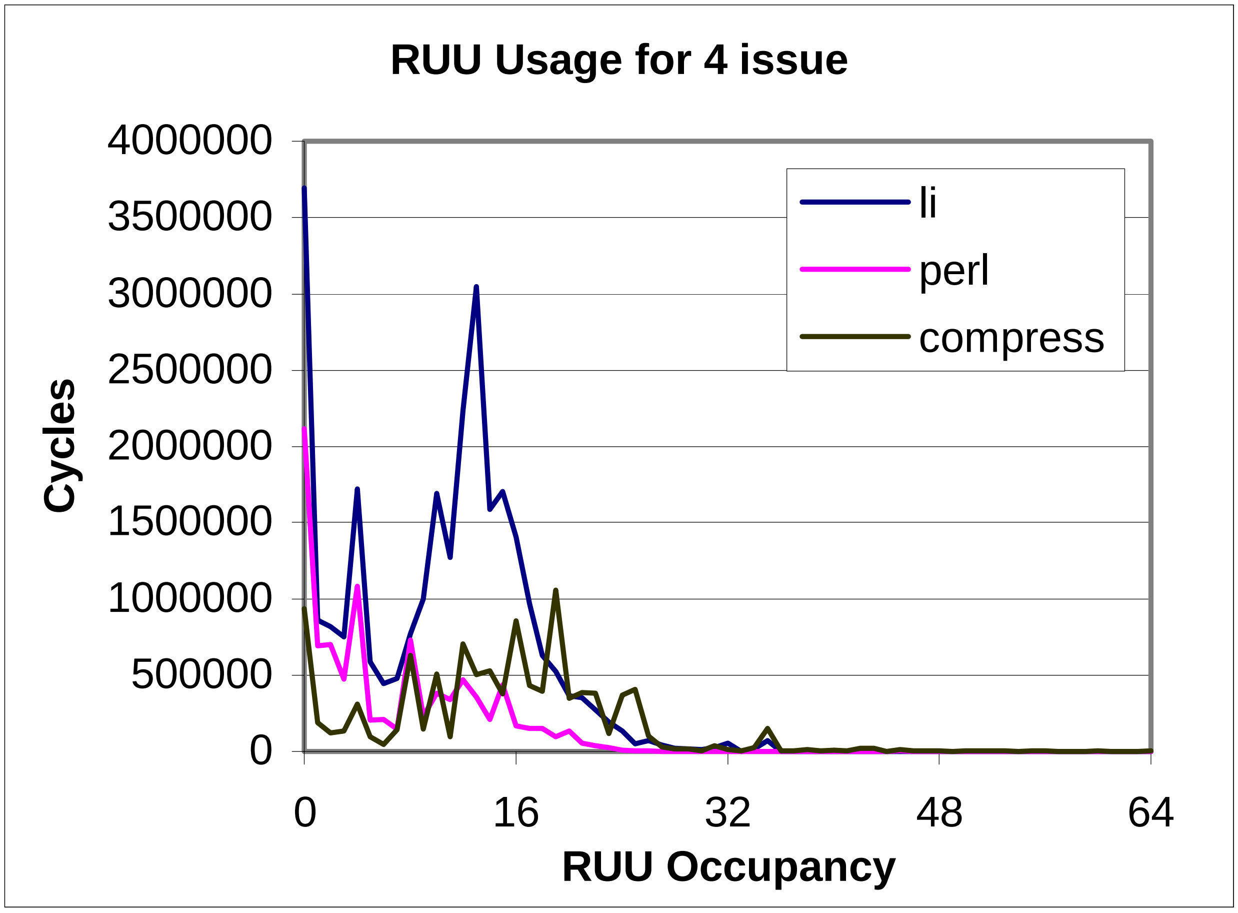
Problem:
High performance processors use too much energy.  Some applications need high performance part of the time, but want minimum power consumption the rest of the time (laptops, PDAs, etc.).

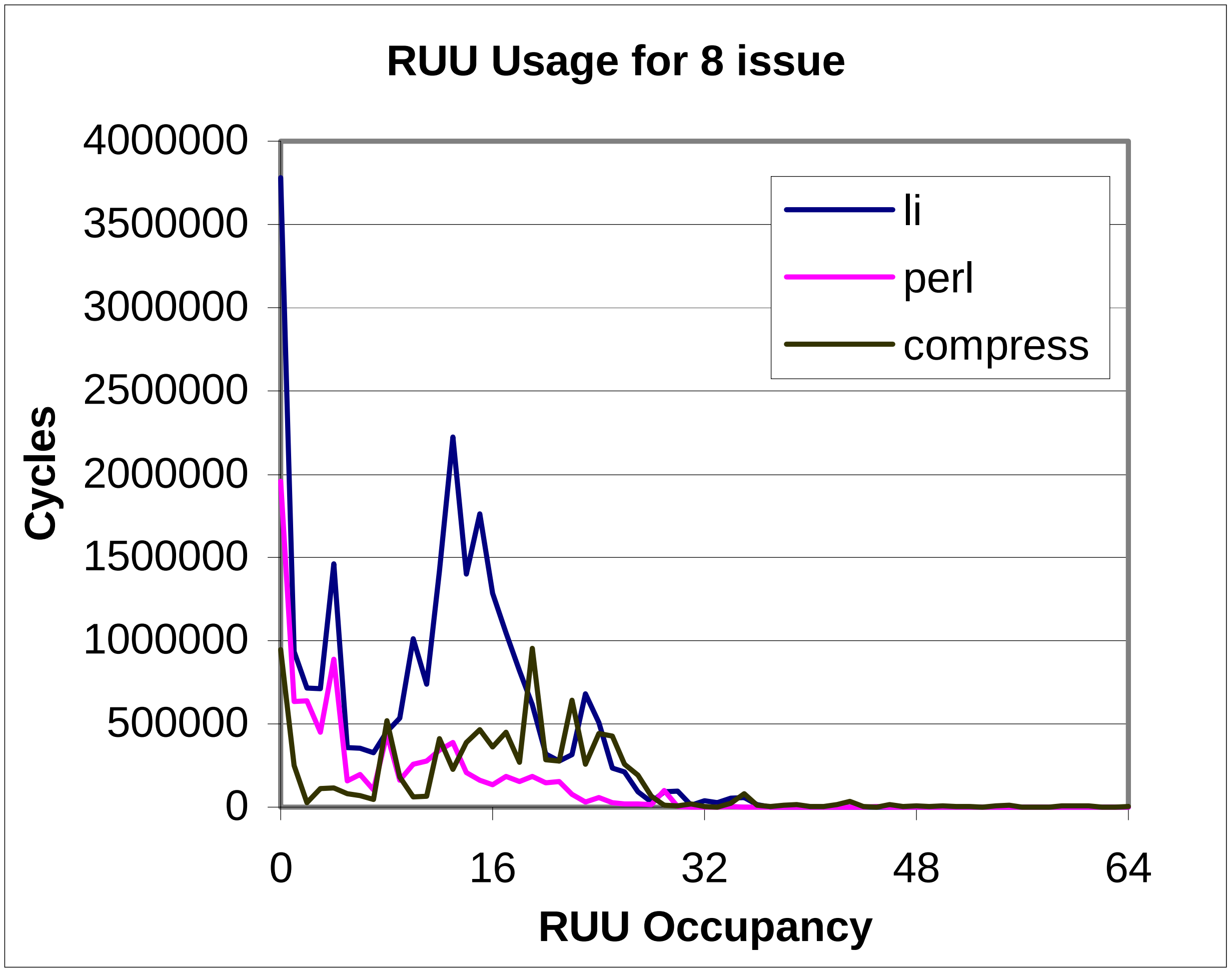
Two proposed solutions:
Reduce issue window size when peak performance is not required; large issue windows take up a lot of power (18-46% on the alpha), so reducing its size should lower power usage.

Have a high-performance core and a lower-performance core, and let the user switch between them as required.

Issue Window Size:

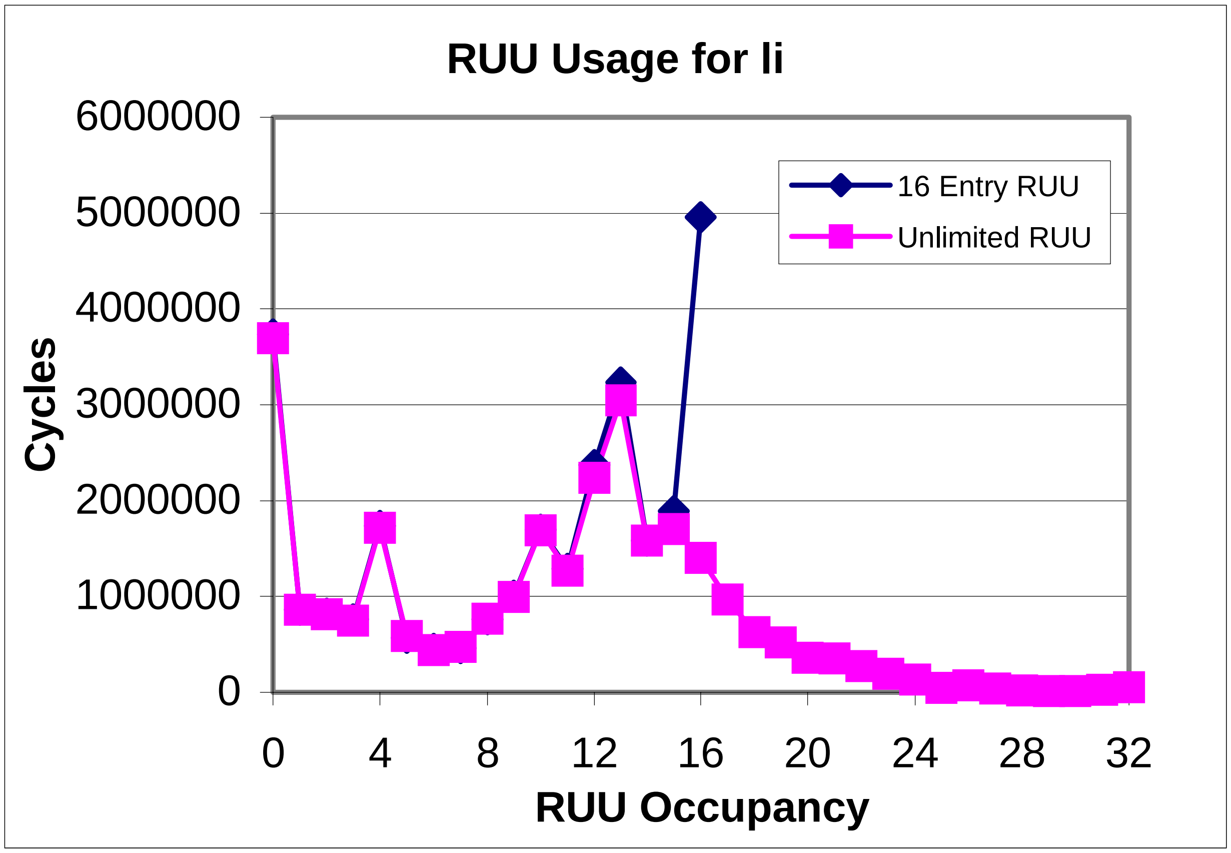
We varied the size of the issue window on a 4-issue and an 8-issue machine, modeled in SimpleScalar, with power numbers provided by Wattch. Benchmarks used were a subset of SpecInt95: reduced size versions of Li, Perl, and compress95.
How much of register update unit is used?





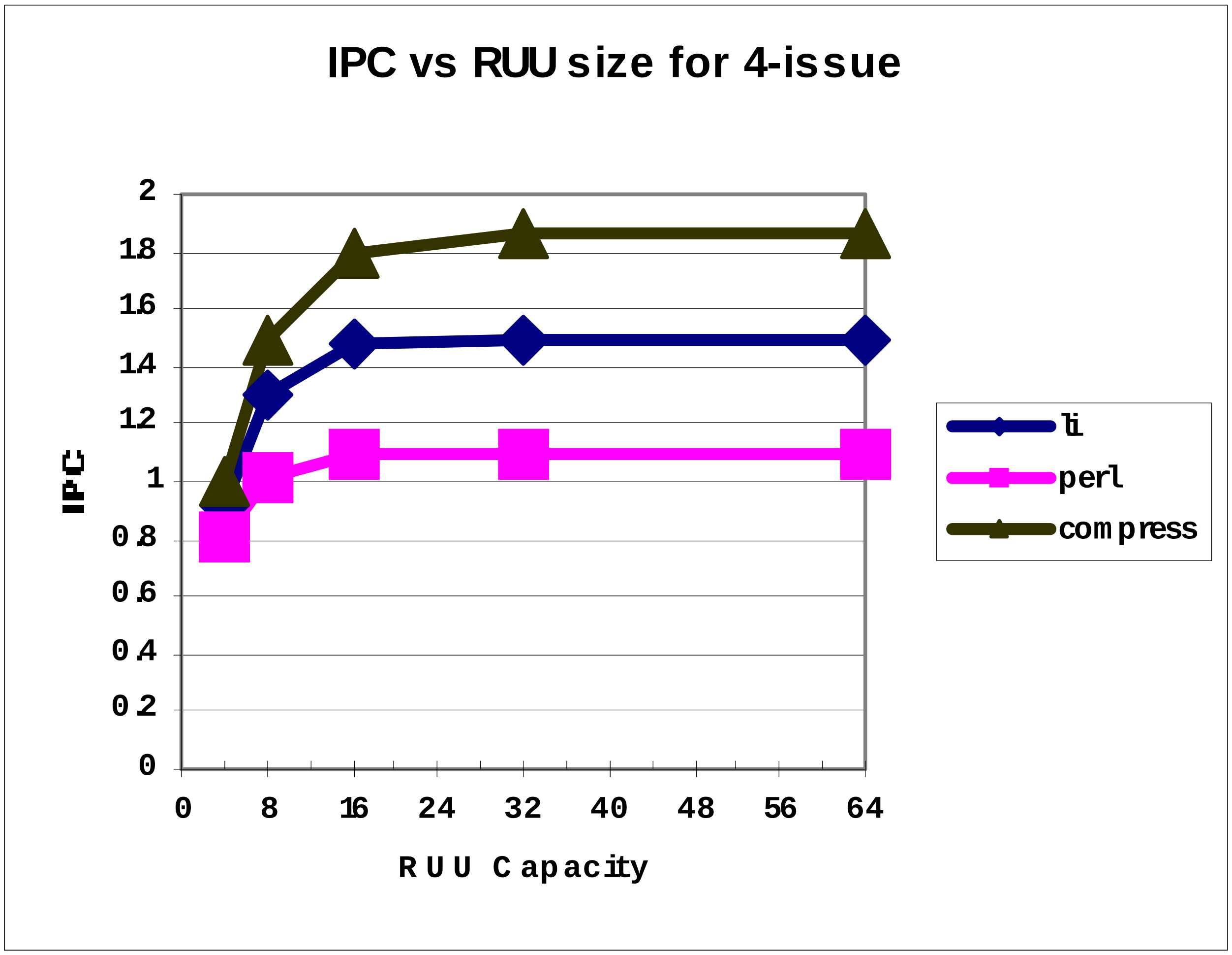
As expected, the 8-issue version used more of the RUU; but in both cases 32 entries was almost always sufficient, and 16 was usually sufficient

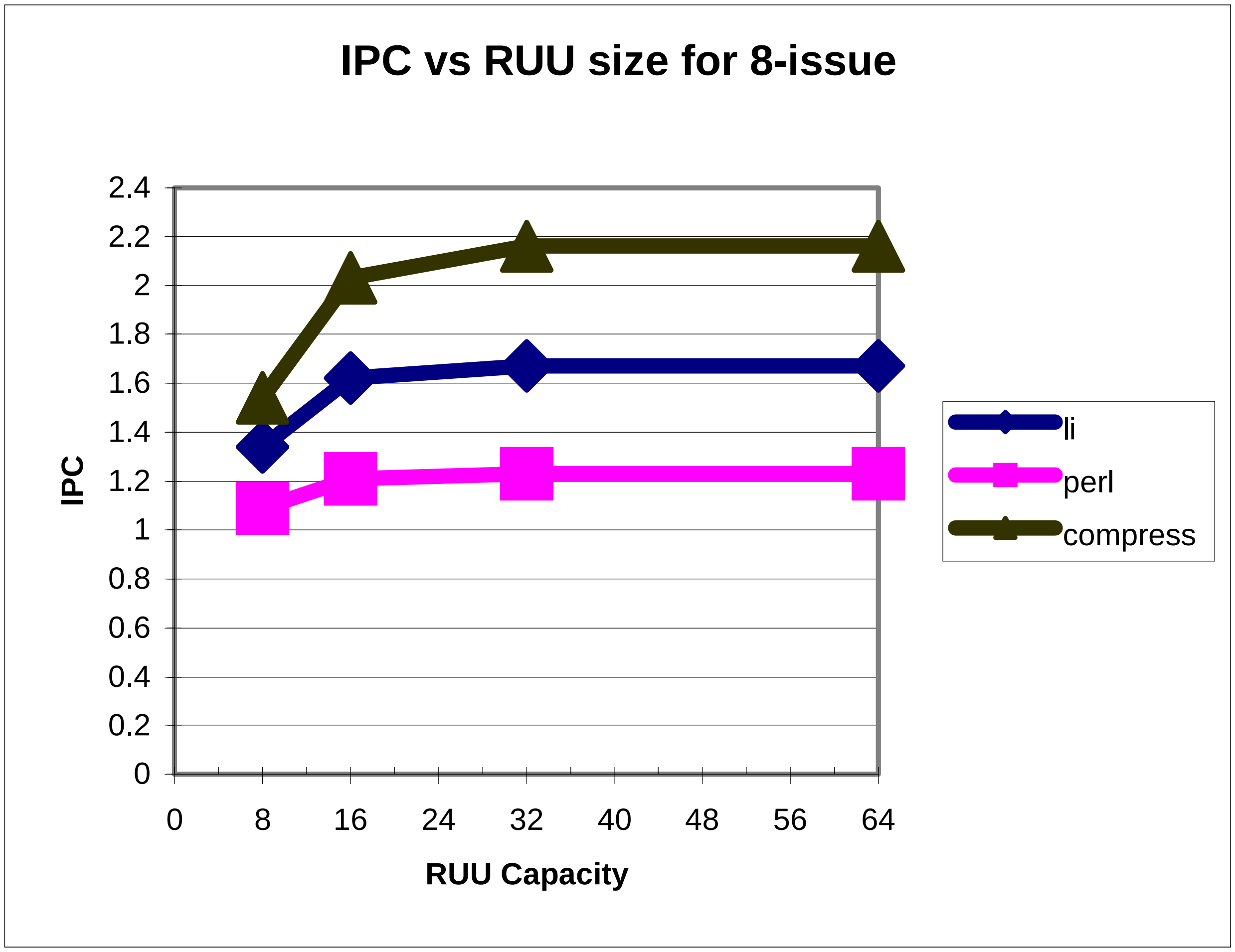
Next Experiment: how much does performance drop as RUU shrinks below 32 entries?



The 16-entry RUU effectively saturates at 16, but how does this affect the number of instructions actually executed? Are 16 entries enough to find all of the ILP present?

Next we charted average IPC versus RUU size for both the 4 and 8 issue machines.





No loss of IPC going from 64 to 32 entries.

Very little loss from 32 to 16, but sharp drop below 16.

What about energy?

| **Structure** | **4x4** | **4x8** | **4x16** | **4x32** | **4x64** |
|---|---|---|---|---|---|
| **Energy/Inst (li)** | 15.8 | 13.0 | **11.8** | 12.8 | 14.1 |
| **Energy/Inst (perl)** | 16.5 | 14.3 | **13.6** | 14.7 | 16.1 |
| **Energy/inst (compress)** | 14.4 | 11.5 | **10.6** | 11.3 | 12.5 |

| **Structure** | **8x8** | **8x16** | **8x32** | **8x64** |
|---|---|---|---|---|
| **Energy/Inst (li)** | 13.8 | **12.5** | 13.4 | 14.9 |
| **Energy/Inst (perl)** | 15.1 | **14.7** | 15.8 | 17.6 |
| **Energy/inst (compress)** | 12.4 | **11.4** | 11.9 | 13.3 |

In all cases, the minimum is at 16-entry RUU!

Conclusion:  Dynamically scaling the RUU to smaller than 16 entries is a bad idea; voltage scaling and other techniques will do better.  Scaling above 16 uses a lot more power for very little performance result.

Alternate plan: Have a high-performance core and a low-power core on the same chip; that way the low-power version could be a single issue machine, avoiding the complex issue logic completely.

We investigated commercial processor families, comparing the energy per instruction between a high-performance and a lower-performance processor in the same family, using the same process.

PowerPC: 440 vs. 405
440 is a dual issue, out of order, 7 stage pipeline, high-performance machine.
405 is a single issue, 5 stage pipeline, low-power machine implemented in the same technology.

The 405 is approximately 40% of the speed of the 440, but consumes 50% of the power, so the power per instruction is worse.

Intel has demonstrated with the PIII that voltage scaling applied to modern processors can result in a much more dramatic improvement; the mobile PIII has a low power mode that consumes 46% of the power of the performance mode, with 70% of the performance.  Similar results can be had with the Transmeta Crusoe chip, which scales from 1.6V to 1.2V and consumes only 25% of the energy at the lower level for half the performance, or from the Intel Xscale chip, which scales from 800MHz to 150Mhz, consuming less than 10% of the energy at the slower level.

Moreover, all modern chips aimed at mobile computing have a sleep mode, and can achieve a linear power-performance tradeoff by toggling into and out of sleep mode…

Thus, the dual core approach is also less useful than voltage scaling.

Where do we go from here:

Run some additional benchmarks to verify our numbers.

Extend survey of commercially available processors to verify those numbers.