Plumbers Conference, Vancouver 2018. Revision Nov 15, 2018

High Performance Computing Using RISC V

Christoph Lameter <cl@linux.com> @qant

Is it too early to talk about High Performance?

RISC V has barely escaped from the small embedded market. Operating System support is not mature and the whole infrastructure is going to take a couple of years to mature before actually we would have competitive system.

One instruction set to rule them all…. Embedded, Enterprise, Cloud and HPC.

Key issues of concern

1. Page size. 4k is a problem.

   64k [256k] page table structure mismatch.

2. Large Memory Management

   4TB system.

   Lock contention.

3. Per cpu atomics for statistics

   Execution context (processor id, node id, time,

4. I/O performance. PCIe. 50GT links?

   Coherence / RDMA / PeerDirect

   Offload engines (GPU, FPGA, Storage)

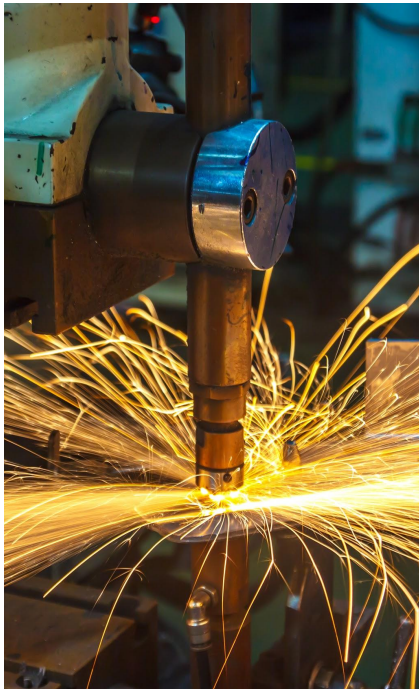
Instruction set issues

Per cpu atomicity

RISC V has SC/LC scheme and atomics. SC/LC cannot be used for percpu atomicity because percpu atomicity requires an instruction that cannot be interrupted. SC/LC require multiple instructions.

Statistics in the Linux kernel require an ability to increment/decrement counters in a light weight way. Counters are per cpu in order to avoid contention.

AMO: Missing cmpxchg. Cmpxchg is used in many places. Also required in a form with percpu atomicity in the SLUB allocator for scaling.



Page size

4k ...

SV39 and SV48 are all based on 4k page sizes (RISC V architecture memory subsystem). Memory of servers is going to be in the 10ths of Terabytes when RISC V becomes able to run general HPC loads.

Cure:
Modify satp register to allow for 64k and 256k page sizes?

Manual mentions TLB reach problem but that is not the key performance issue here. The overheard of processing is.

Note also binary segment alignment to 2M already in x86.

Fast gettimeofday()

Gettimeofday() is the most frequently used system call in HPC applications and it is frequently used concurrently on all cores. Locking would be a scalability problem. Major architectures implement gettimeofday() in user space avoiding kernel syscalls.

RISC V has user space accessible time registers

Thank you.