What matters?
What is the general state of tools?
Architecture quirks?

- Overall tools stack, what is ported, what remains?
- Performance tools? Debugging tools?
- https://gitlab.com/arm-hpc/packages/wikis/home
- ARM HPC google group
- stat - wisconsin & LLNL pacakges are working
- Talking about ARM64
- Job launch (SLURM) is missing.
- Needs include: Dyninst analysis, symtabapi, elfutils, libdwarf, binutils, PAPI, POSIX kernel for signals, C/C++11 atomic operations, perf inerface, libunwind, most are (partially) working.
- AR64 drops backwards compatibility with ptrace interface. ptrace is a moving target, moving to a new standard.
- Tools started at Livermore are not on gitlab list
- developer.arm.com/hpc/hpc-software
- Where are the opportunities?
-
- Ideas about tools and a sense of what's there/
- Tag along -- other sessions start later... who's going to build large ARM systems and run HPC problems on them.
- Gauge interest in ARM clusters outside of Japan & Barcelona, what do people care about wrt binary instrumentation? Want to prioritize.
- Score-P is interested in unwinding the stack, libunwind does it pretty well already if used within a context (outside of a signal context).
- Collect feedback for colleagues on interest and how to use ARM tools and clusters.
- ARM stack unwinding, works with Fujitsu on their ARM cluster, wants to support his tools on ARM.
- Interested in knowing what others were expecting. What analysis would be useful?
- Working on Macau, interested in performance bottlenecks specific to the ARM architecture, and performance analysis to reduce bottlenecks, and what is the maturity of compilers, especially compared to Intel.
- What performance tools do people have now, what might be needed to support them. Any information about data analytics on ARM would also be useful.
-
- State of the tools:
- Source level tools are pretty much there
- Binary tools are a mixed bag, no binary intstrumenters working on ARM yet
- Maqao, performance analysis tool framework, similar to Dyninst in that it works at the binary level to parse binary code and analyze it at multiple levels. http://www.maqao.org, porting from x86, uses performance counters, looks at binary code to identify what the compiler did and what it might have missed. Gives advice to the application developer about compiler flags and loop-llevel transformations. Instrumentation already works on x86 but not yet on ARM.
- Possible side discussions between Maqao and HPCToolkit and TAU developers about the pitfalls of ARM uncovered so far.

- Are PAPI and perf events extracting sufficient information from x86 architectures? Maqao is currently using native perf for both x86 and ARM.
- What sort of binary rewriting applications are relevant in our tools? What might be interesting is identifying loops and functions, placing tags around the loops for later identification.
- There is a module in Maqao that does value identifying.
- Ubuntu and SLES are both supported, so standard serial tools (gdb, gprof, gcc, etc.) are supported, as are TotalView, ddt and MAP
- http://montblanc-project.eu/developer-tools
-