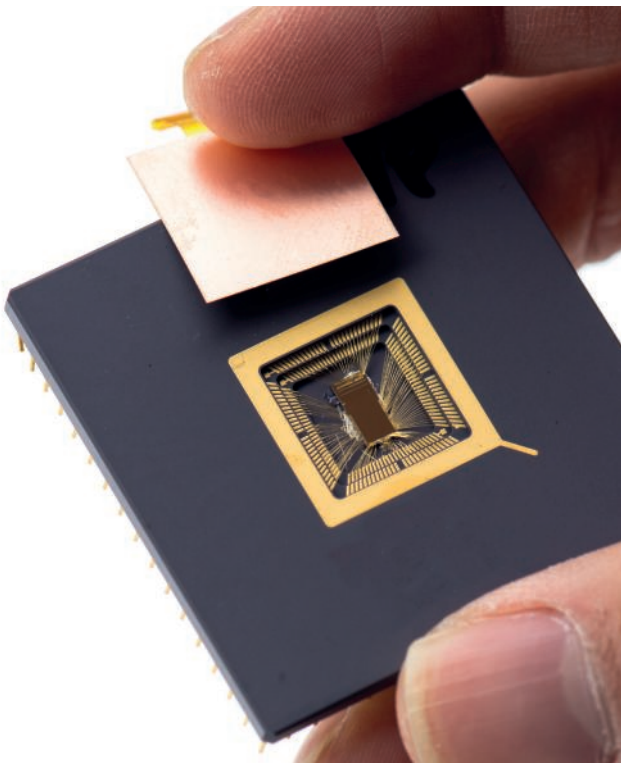
Back to the future for RISC

As ARM is acquired by SoftBank, the prospects for the RISC-V architecture appear to have improved. By **Chris Edwards**.



A prototype RISC-V processor from early 2013

The surprise announcement that SoftBank was to buy ARM sent ripples through the electronics industry as companies tried to determine the consequences of the deal. Though there is no immediate threat to users of the ARM architecture, the potential for upheaval has led some companies to consider the risks of changes in their relationship with the processor-IP supplier.

One option is to look more closely at direct competitors to ARM, such as Imagination Technologies, which acquired rights to the MIPS architecture in 2013. Another is to look at an architecture with no commercial ties.

In their 2014 technical report for the University of California at Berkeley, Krste Asanovic and Professor David Patterson – who developed the concepts behind the SPARC architecture in the 1980s – argued that, while there are good commercial reasons for processor makers to maintain proprietary instruction sets, there is no good technical reason for users to adopt them. Their proposal was for an instruction set architecture (ISA) offered on a similar basis to open source software that could stimulate an ecosystem like that built around Linux.

As their proposal represented the fifth generation of RISC processor design, it was called RISC-V.

The resulting ecosystem would not run the risk of one vendor deciding to close access or suddenly raising prices. If one implementation becomes prohibitively expensive – or is removed from the market – other providers can notionally step in.

RISC-V is not alone in being open source. Sun Microsystems, ahead of its acquisition by Oracle, provided open source versions of the SPARC architecture. Before that, Sun decided to provide the specification to the IEEE and so make the SPARC ISA a standard. The European Space Agency took advantage of that, adopting SPARCv8 as the basis for its Leon processor.

The Berkeley group behind RISC-V has attempted to make the architecture more attractive for commercial users by adopting the Berkeley Software Distribution (BSD) open-source licence, rather than the Gnu Public Licence used for Leon. The BSD licence does not demand users feed improvements and changes back to the public pool: RISC-V implementations can be made open source or kept proprietary.

Limited commercial adoption of RISC-V so far means there is little in the way of tool support, other than open source packages developed primarily by the academic community. But changes in the computing landscape, as well as acquisitions, have made RISC-V more prominent.

Graphics chipmaker nVidia says it intends to replace the proprietary Falcon onchip processor that manages its rendering arrays with a version based on RISC-V. In recent months, several IP vendors – including Andes Technology and Cortus – have joined RISC-V. Codasip, LowRisc and SiFive, formed by members of the UC Berkeley group, are offering processor cores for custom silicon. Lattice Semiconductor has a version suitable for use in its FPGAs.

A growing interest in compute accelerators amongst server-farm operators – many of whom will incorporate a mixture of custom SoCs and FPGAs – has rekindled interest in an open, extensible ISA. Some Google engineers, for example, have worked on a way of loading firmware into RISC-V that uses the same scheme as Android. The question is the degree of openness required to push RISC-V into the mainstream.

ARC and Tensilica were both built around the idea of customisable processors, while MIPS has incorporated 'hooks' for custom accelerators. IBM is trying to encourage wider use of the Power architecture in data centres through its OpenPower programme and the

upcoming Power 9 processor, which has been designed to work with closely coupled accelerators. Some see RISC-V as a more flexible option.

The RISC-V group took extensibility into account when defining the ISA. At its core, RISC-V has a similar structure to the DLX architecture used in the book '*Computer Architecture - A Quantitative Approach*', written by Prof Patterson with Professor John Hennessy of Stanford University. Sharing similarities with the original MIPS ISA, RISC-V is a pure load-store machine with 32 integer registers.

Unlike most classic RISC architectures, RISC-V was designed to be modular and allow for expansion through a variable-length instruction-encoding scheme. Whereas conventional RISC architectures use single 32bit word instructions to minimise the complexity of the decoder stage and allow predictable pipelining, a variable-length scheme makes it easier to add new instructions at the cost of one bit of addressing range on short loop and jump instructions. The most common instructions use a single-word encoding to maintain throughput. And, like ARM's Thumb, there is a compressed-instruction version that uses 16bit instruction words.

Although the presence of a scheme for adding instructions provides a temptation to add operations seen in processors such as the x86, the team has used an idea called macro-op fusion to try to head off instruction-set bloat (see fig 1). The approach takes into account the way that many processors decompose single instructions into multiple micro-operations at the instruction decode stage. What looks like one, potentially quite long, instruction to the compiler results in multiple hardware-level microinstructions.

In one benchmark, UC Berkeley researchers compared the code density and pipeline throughput of RISC-V to x86 and ARM, finding that indexed loads perform better on the older architectures because they are handled as single instructions. Indexed loads are common in code that manipulates arrays and other complex data structures. The RISC-V answer was, in effect, to tell the decode logic to treat certain combinations of instruction as though they were one unit.

With an indexed load, rather than computing the offset and storing it in a register for the next instruction, the internal hardware can compute the offset and use it directly for the actual load command, in effect knocking out the pipeline stages associated with the additional instructions. In the case of the indexed load, the compressed instruction format condenses the two operations into a single 32bit word.

A processor that does not implement the hardware for certain fused instructions will simply run the original commands as is. Compiler writers may choose to add flags that determine whether it is a good idea or not to split the operations that make up fused instructions depending on the target processor.

Some of the RISC-V ISA remains a moving target. One decision yet to be fixed, as of early September 2016, is how to handle processing in supervised mode, compared to user mode. The current proposal, likely to be adopted, is for one mandatory level and three optional privilege levels. The machine level – which RISC-V supports today – is intended for simple embedded processors with no protection between kernel and applications. To provide hardware-based security, implementors can add hypervisor, supervisor and user levels. With this in place, RISC-V should be in a position to scale from tiny embedded cores to processors suitable for running full Linux-style operating systems within virtual machines.

As the pieces come together, the RISC-V architecture has the chance to become much more than an academic exercise. But it will take time for an ecosystem to develop that can push it into the mainstream. And that assumes SoftBank's actions push design teams into the arms of an open-source alternative.

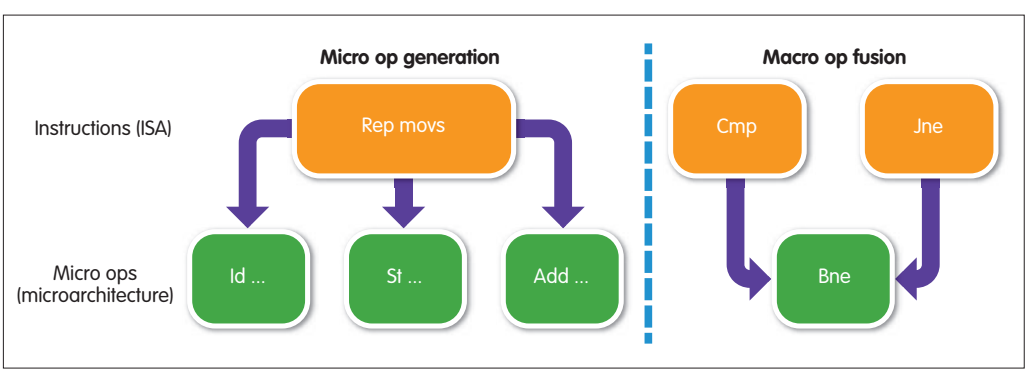


Figure 1: The conceptual difference between micro-op decomposition and macro-op fusion