John Cocke

Research by G00342591/Matthew Bartley and G00348436/John Shields

Basic Info:

Who is John Cocke?
Turing Award Winner
Born May 30th 1925 - Died July 16 2002
Received a BSE in Mechanical Engineering 1945 - Duke University
Received a PHD in Mathematics 1956 - Duke University
Joined International Business Machines Corporations or IBM in 1956 where he worked for the majority of his life

Awards and Achievements

Received the 1987 Turing reward from the association for Computing Machinery-the group's highest honour for technical contributions in computing and just four years later in 1991 he received the 1991 National Medal of Technology for "his development and implementation of Reduced Instruction Set Computer architecture that significantly increased the speed and efficiency of computers, thereby enhancing US technological competitiveness."
In 2002, he was made a Fellow of the Computer History Museum "for his development and implementation of reduced instruction set computer architecture and program optimization technology."

https://computerhistory.org/profile/john-cocke/
https://www.uspto.gov/learning-and-resources/ip-programs-and-awards/national-medal-technology-and-innovation/recipients/1991

Innovate and Abstract

Often considered the father of RISC, he was renowned for his knowledge and ability to look at things differently and had a deep understanding of both computer hardware and software and the way they interacted.
“After I had talked for about 5 minutes on the topic-digital signal processing—he went to the blackboard and more or less wrote out a major part of my thesis. It was a rude awakening." - Abraham Peled
“If I see an opportunity, I drop all the rules, even when doing so is probably a mistake.”
— JOHN COCKE
His work on the ACS and IBM 7030(Stretch) helped him develop RISC.

http://www.iment.com/maida/tv/computer/johncocketranscript.htm

Introduction

In the early 1970s, John Cocke transformed computing by simplifying the set of instructions that tell computers which functions to perform. Cocke was the turing award winner of 1987. Cocke won this award for his significant contributions in the design and theory of compilers, the architecture of large systems and the development of Reduced Instruction Set Computers (RISC): for discovering and systematizing many fundamental transformations now used in optimizing compilers including the reduction of:

- Operator Strength
- Elimination of Common Subexpressions
- Register Allocation
- Constrant Propagation
- Dead Code Elimination
- RISC Architecture
- Reduced Instruction Set Computers.

The main idea behind this architecture was to make simpler hardware by using a reduced instruction set composed by a few basic step for loading evaluating and storing operations.

Although this is a reduced instruction set this approach tries to increase the CPU performance.

Reduce the cycles per instruction at the cost of the number of instructions per program.

Characteristic of RISC

- Relatively few instructions
- One word instructions.
- Instruction take a single clock cycle to get executed.
- More number of general purpose register.
- Simple Addressing Modes.
- Less Data types.
- Pipeline can be achieved.

Pipelining

- Pipelining, a standard feature in RISC processors, is much like an assembly line. Because the processor works on different steps of the instruction at the same time, more instructions can be executed in a shorter period of time.

- Pipelining is a technique of decomposing a sequentail process into sub-processes, with each sub-process being executed in a special dedicated segment that operates concurrently withh all other segments

- Any operation that cn be cecomposed into a sequence of a sub-operation of about the same compleity can be implemented by a pipeline processor

RISC Projects

John Cocke was involved in three RISC projects:

- IBM 801 machine (1974)
- Berkeley’s RISC-I and RISC-II processors (1980)
- Stanford’s MIPS processor (1981)

IBM 801 machine

- IBM designed an experimental minicomputer named the 801.
- The name 801 comes from the building the project was housed in.
- IBM used the resulting architecture in various roles out through to the 1980s.
- In the early-mid 1970s, IBM were interested in the possibility of constructing a telephone switch to handle a million cal per hour.
- Their estimation was that this would require at least a 6 MIPS processor.
- John Cocke and his group who were working on the project at the Thomas J. Watson Research Center.

Berkeley’s RISC-I and RISC-II processors

RISC I

- Originally known as Gold
- The first attempt to implement the RISC concept was originally known as Gold
- VLSI design course
- Work on the design started in 1980 as part of a VLSI design course, but the then-complicated design crashed almost all existing design tools. The team had to spend considerable amounts of time improving or re-writing the tools, and even with these new tools it took just under an hour to extract the design on a VAX-11/780.
- ACM ISCA
- 44,500 transistors
- 31 instructions
- 78 32-bit registers
- The final design, known as RISC I, was published in ACM ISCA in 1981.
- It had 44,500 transistors implementing 31 instructions and a register file containing 78 32-bit registers

RISC II

- While the RISC I design ran into delays, work at Berkeley had already turned to the new Blue design.
- Work on Blue progressed slower than Gold, due both to the lack of a pressing need now that Gold was going to fab, as well as changeovers in the classes and students staffing the effort. This pace also allowed them to add in several new features that would end up improving the design considerably.
- 138 registers - broken into
- 8 windows - of
- 16 registers each - with another
- 10 globals
- RISC instruction set with only 39,000 transistors

Stanford’s MIPS processor

- Strong background in compilers
- Develop a processor
- whose architecture would represent the lowering of the compiler to the hardware level, as opposed to the raising of hardware to the software level, which had been a long running design philosophy in the hardware industry.
- MIPS processor implemented a smaller, simpler instruction set
- 32 registers - each
- 32 bits wide

CISC and RISC:

Before RISC the Complex Instruction Set Computer(CISC) was the normal(it wasn’t known as CISC until years later).
CISC architecture was created to complete tasks in as few lines of assembly code as possible
While the CISC approach is to minimize the number of
instructions per program which in turn sacrifices the number of cycles per instruction.
RISC does the opposite, reducing the cycles per instruction at the cost of the
number of instructions per program.
He researched and discovered that 20% of the instructions did 80% of the work
This 80/20 rule helped created the concept of RISC.
By removing seldom used instructions they create a simple set of instructions
that would be executed in a single cycle thus increasing the efficiency
This would mean more instructions when performing a task but executed in one cycle. This was a contrast to CISC which would cram
them into single instructions which causes more cycles.
Computer power is measured by the speed it can process things and RISC is much more beneficial for that.

RISC Concept:

"We knew we wanted a computer with a simple architecture and a set of simple instructions that could be
executed in a single machine cycle—making the resulting machine significantly more efficient than possible with other,
more complex computer designs," - John Cocke in 1987.
https://www.ibm.com/ibm/history/ibm100/us/en/icons/risc/

RISC vs CISC:

Something to note that in modern times the comparison is very minimal as
both have adopted each other’s features making them both similar
CISC has more emphasis on hardware as opposed to RISC with software
RISC is built for single cycle instructions while CISC can take several cycles
CISC has a hardware centric design while RISC is software centric
RISC uses a lot more RAM which can cause bottleneck(system does not have sufficient RAM)
while CISC is more efficient
RISC has only one layer of instructions while CISC can support microcode
https://www.microcontrollertips.com/risc-vs-cisc-architectures-one-better/

Modern day RISC

While RISC and CISC are both still used both architecture types have adopted features of each other
such as RISC architectures often containing variable length instructions(ARM architectures often use this) and several contain microcode capabilities and multi cycle instructions
Used in high end applications such as video/image processing and telecommunications.
A13 Bionic(iPhone 11 model) and the A12X Bixonic(iPad Pro) systems use ARM architecture which is a family of RISC architectures.
Summit Supercomputer which was created by IBM used in Oak Ridge National Laboratory uses
IBM Power Instruction Set Architecture which is based upon RISC
Super computer Fugaku(worlds fastest supercomputer)
The NASA path finder utilises a Radiation Hardened IBM Risc 6000 Single Chip (Rad6000 SC) CPU
https://mars.nasa.gov/MPF/mpf/faqs_general.html
https://medium.com/swlh/what-does-risc-and-cisc-mean-in-2020-7b4d42c9a9de

Stretch

The IBM 7030 System. Provided key concepts that lead to the success of RISC, ACS and IBM future projects.
A supercomputer created in 1961 that was considered the fastest computer at its time.
It’s technical goal was to be 100 times faster than the IBM 704 however it did not reach this requirement. It reached around 30 times its speed.
While considered a technical failure, it’s work with partitioning, lookahead and pipelining paved the way for IBM and contributed towards many of their future works and the project is considered a major success in hindsight.

References

[1] A.M. Turing

[2] John Cocke | Lemelson

[3] RISC Architecture

[4] Britannica Article by William L. Hosch

[5] Computer Pioneers

[6] Computer Organization | RISC and CISC

[7] IBM 801

[8] RISC I & RISC II

[9] MIPS

[10] RISC Pipeline

[11] Stretch

[12] RISC Concept

[13] CISC

[14] RISC and CISC

[15] RISC Concept

[16] Achievements