What We Learned From the Experts

CSC218 Computer Organization

18 December 2017

Contents

Part I

Notable computer engineers

1 Gordon Bell

—Sloane Bartelme

Gordon Bell is a pioneer in the field of Computer Science. He was one of the original DEC employees and helped invent many of their PDP (Programmed Data Processors).

There is a law of computing named for him. Bell's Law states that every ten years, a new class of computer will be developed, and an entirely new industry will appear to compliment the new technology. So far, he has been right. He went on to become Vice President of Engineering at DEC for eleven years (1972–1983) and oversaw the VAX program. He had many endeavors after this, two most notably his time as researcher emeritus at Microsoft Research (1995–2015) and his venture into LifeLogging. Though his resume is massive, it is his venture into LifeLogging that was most interesting to me. Bell believed, at the height of his interest in LifeLogging, that somewhere in the future, humans would be recording every single moment of their lives on their "lifelog," a small computer worn around the neck. Bell has written two books on the subject, both of which focus on the ways LifeLogging could improve people's lives, socially and in the workplace. However, I see a sinister potential for LifeLogging technology that Bell either does not see or willfully ignores. He has since stopped the practice, or at least become less extreme in his recording methods, so perhaps some of the criticism has reached him.

My problem with LifeLogging is that, in our society, there is no way corporations or the government would not exploit LifeLogging in some way. The NSA would use the memories from people's lifelogs to monitor their every move. Imagine receiving targeted ads that were personalized based on your actual, lived experiences. Honestly, cookies are bad enough. I can't imagine the invasive potential of corporate interference with lifelogs. A society under such invasive surveillance would be extremely restrictive, and not just from government and corporate angles. Such obsessive record keeping would mean we surveil ourselves and each other. It would be like comparing FitBit information is now, except you would be monitoring and comparing literally any or every aspect of your entire life. I don't need to say much further to explain why that might be a negative phenomenon.

In conclusion, Gordon Bell is a brilliant computer scientist, and we have him and the other early employees of DEC to thank for modern computers. However, he's a bit of a nut, and I hope LifeLogging never exists in the way he conceptualized it, for the sake of humanity.

2 Maurice Wilkes

—Anthony Delgado

Wilkes is arguable one of the most important figures in computing in the UK. Some of his most important contribution were the EDSAC (Electronic Delayed

Storage Automatic Calculator), which was the first stored-program computer. I presume stored-program means that it could load and run programs previously stored on the computer. He was a Mathematical Physicist and worked on radar technology during the second world war. His curiosity of computers started from his exposure and fascination with the EINAC (Electronic Numerical Integrator and Computer) used during the war for shell trajectory calculations. He attended lectures during 1946 at the University of Pennsylvania that proved pivotal in his understanding of building computers akin to those proposed by Von Neumann. Now being one of the few people in his time to understand how to build a computer, he got to work on plans for the EDSAC.

His goal was not to produce a robust cutting-edge computer with all the bells and whistles, but instead to take a more practical approach and produce something that would make his and his coworkers lives easier. He sought to reduce the burden of heavy calculations for scientists and engineers he worked with. He also wanted to make the computers easier to work with so that other professionals in different fields could benefit from the technology.

Another notable contribution by Wilkes was the concept of microprogramming, which is a type of control unit that uses stored-programs that controls the logic necessary to execute instructions within a processor. It is often juxtaposed with Hardwired control units which as the name suggest are hardwired into the processor via logic gates. One of the many advantages of microprograming is that it simplified logic control in processors and allowed for larger and more complex instructions to be carried out.

His exposure to work being done in digital telephony by the Switzerland Telecommunication firm of Hasler, allowed him to be early on the scene of networking. This was before the internet was a thing. He wanted to connect computers and started the Cambridge Ring project, which used the idea of Rings from topology, where nodes are connected to two other nodes, and are laid out in a circular fashion. This did not last long because the industry agreed on using ethernet as the standard. After retiring, Wilkes went to work for DEC and said he regrets not working in industry earlier since he really enjoyed it.

3 Carver Mead

—Noah Carpenter

Mead graduated from Caltech in 1956, where he is now a professor, with a degree in Electrical Engineering. He is most famous for his Very Large Scale Integration

(VLSI) work. VLSI has been used in cell phones, as well as microwave dishes for communication. He currently holds over 50 patents for design.

Carver Mead is best known for inventing the high frequency transistor, and pioneered the movement toward programmable logic chips in computers. And then in 1999, carver won an award for half a million dollars from MIT for his all of his innovative work. However, in recent years, he has emerged not as an important computer scientist, but as one of the most important physicists of the twenty first century.

Carvers work in computer science is still incredibly influential in todays world. His work in computer chips is still being used in todays computers. Carver recently said that he believes that computer engineers should look for new forms of computing rather than what they are doing today - similar to what he did when he was younger.

One problem that tends to give computers trouble is overheating. The more chips put in computers, and the more powerful they become, they work harder and thus overheat more. This is where Meads work in computer science began. He sought a way to replace the overheating and less-efficient chips. To accomplish this, he used the brains of living things to use as a start-off point for his work. His work applied to engineering as well, as Neuromorphic Engineering (using humans or animals as a model for engineering) has become an important practice for Engineering, as well as computer science.

4 Gordon Moore

—John Hoberg

Moore's Law: Monday November 27th Prediction that set pace for the modern digital revolution. Increase in power with a decrease in cost at an exponential rate. Gordon paved this path for Intel to make transistors faster, smaller and more affordable. Economics: Power/ performance and cost are two key elements to tech development. As transistors become smaller, power is increased and energy efficiency becomes greater all at a cheaper cost to the user. This development increased productivity and enhanced existing industries but also created entirely new ones. Tech: Helped to make this industry much more of an affordable necessity instead of a rare and expensive venture. Just about everything about modern computing as we know it (Social Media, Modern data analytics) sprang from this development. Moore's Law has set the pace for 50 years to help allow digitization and personal electronics.

The Road to 14nm (2014):

- 2010 Had Westmere 32nm
- 2011 had sandy bridge 32nm
- 2012 Ivy Bridge 22nm
- 2013 Haswell 22nm

In 2015 Intel stated that the pace of advancement has slowed from 2012 at 22nm continuing to 14nm. Expresses that moores law is closer to 2.5 years and that late 2017 they hope for 10nm transistors. Shoot for 10nm, 7nm and even 5nm and say they have plans for these but that Moore's law may come to a close and significantly slow down by 2025.

5 Robert Noyce

—Jerome Richards

Robert Noyce was not just a computer guy. He was a renaissance man in the 1900s. He is well known for his work in the computer field, but other than that he was an athlete, pilot, tinkerer, and apparently dabbled in singing. A technologist, he was a co-inventor of the integrated circuit. He also co-founded Fairchild Semiconductor, as well as the more well known and current industry giant Intel Corporation. By co-inventing the integrated circuit, an argument could be made that he is one of the most important figures in created the age of modern computing that we find ourselves in. Other than his intellectual contributions to the industry, which are many, he helped found the Semiconductor Industry Association, which is technically a trade organization, and it lobbies for and represents the semiconductor and microchip industry in the United states. He served on the Presidents Commission on Industrial Competitiveness, and was the first CEO of SEMATECH, which is a not-for-profit R&D consortium focusing on semiconductors and microchips. Robert Noyce over his lifetime acquired 15 U.S. patents, but his greater legacy can be traced through the omnipresence of integrated circuits in the world today. After reading about Robert Noyce, Intel's website has links to the way microchips are built today. It is a massive industrialized process that takes place in a room "thousands of times cleaner than hospital operating rooms." One of the more interesting processes while making a computer chip is the photolithography used on the chip to map out the circuits and transistors to be put onto the wafer. Ordinary light is not

used, as many features on microchips are literally more precise than the wavelength of visible light allows, so shorter wavelength EMW are used to trace the map. Though Noyce took the first steps, the industry is forging on ahead with innovation on each generation of their products.

6 Federico Faggin

—Jason Wang

This video is about this person called Federico Faggin who is in charge of the development of MCS4 back in 1970s. The MCS4 was a four chipsets that was developed for the physicals. And one of the chips which called 4004 is become the world's first microprocessor. Federico Faggin is hired by Intel at that time and he runs into one problem. He run into was by using silicon gate technology, which is a new technology at that time, is still not enough. He need to build complex circuits in different way than metal gate. And this is a total invention that he have to come up with this by his partner and himself. His partner find out a simpler way of building the chips so that they can combine three chips, which was they plan, into one and also faster at the same time.

Then he talks about how people use the microprocessor nowadays and how it had been developed over years. He state that microprocessor really give the computer the ability of thinking like human brain. And it is show the ability inside the use of internet. Microprocessor has been used in all the pc, smartphones and so on. It connect people with each other with the way that we can just talk with each other just like face to face.And then he talks about the new development of computer science that we should not be scared about the technology development but seeing this as pushing the ability also.

7 The Future of Microprocessors

—Kunle Olukotun and Lance Hammand

8 The Future of the Microprocessor Business

—Michael J. Bass and Clayton M. Christensen

- microprocessor is most transformative technology
- likely you have > 100 microprocessors in your household
  - laptop computers, tablet computers, phones
  - entertainment (television, music)
  - appliances
  - toys
  - cameras
  - vehicles (40 or 50 in automobile)
- principal players in industry
  - Intel
  - Motorola
  - Advanced Micro Devices (AMD)
  - IBM
  - Sun Microsystems (now part of Oracle)
  - Hewlett-Packard
- expenditures on development of microprocessors = billions of dollars/year
- 200 millions transistors on 1 square centimeters
- Gordon Moore: co-founder of Intel
- exponential rate of improvement of price/performance ratios
- Moore's Law: double # of transistors on chips every 1.5 years
- Moore's Law: double performance of chips every 1.5 years
- focus on...
  - producing larger wafers
  - producing smaller line widths
  - producing smaller transistors
  - (all to pack more transistors on each chip, maximize performance)

- "this unshakable industry paradigm will change fundamentally"
- **those who strive to keep up with Moore's Law risk losing market share**
- expect Moore's Law to remain valid for at least another 15 years (at least until 2017)
- will physical limits end Moore's Law? — wrong question!
- top chip makers will have not choice but to continue on same path (trying to keep up with Moore's Law)
- many applications will not require highest performance microprocessors
- other factors will be more important than performance...
  - customization
  - rapid development
- microprocessor market: $40B/year
- segments of market
  - top tier—most powerful microprocessors (servers, workstations)
  - PC market—dominated by Intel ($23B in 2001)
  - microcontrollers (for example, to control engines) ($10B)
  - digital signal processors (cell phones, DVD players) ($4B)
- similar patterns seen in developments in many technologies and industries
  - emphasis on improving performance
  - then (when performance exceeds needs of many customers) emphasis is on customization, reliability, ease-of-use, quick delivery)
- customization, upgrades, quick development facilitated by...
  - modularization
  - standard interfaces
- modularization opens opportunities for companies that make just one component
- segment of market that pays the most, wants highest performance will shrink
- largest, oldest firms often miss this kind of shift in markets
- highest performance needed for...
  - editing digital video

  - 3D games
  - speech to text
- highest available performance not needed for many more popular applications...
  - writing reports, memoranda, correspondence, e-mail
  - calendars, scheduling
  - reading on the Internet, retrieving documents
- history
  - Apple—vertically integrated, proprietary architecture—best performance
  - less expensive machines became "good enough" for doing what most people wanted to do
  - IBM PC—modular, open architecture
  - Compaq and other makers of IBM PC clones entered market
  - customers demand for dependability grew—HP and other companies with strong reputations entered
  - functionality and reliability became "good enough"
  - Dell stepped in to provide highly customized computers and quick delivery
- manufacturers are offering lower performance chips, but not yet modular products
- "design gap"—possible to put more transistors on chip than anyone knows what to do with
- what might modularization look like?
  - SoC—System on a Chip
  - reusable IP (intellectual property) blocks
  - subcircuits (described in software or finished hardware design) that can be dropped into the design of a chip
  - for example, memory controller or serial interface
  - choose amount of instruction-level parallelism, bus widths, cache sizes, specify instruction set
  - design in weeks rather than months
- if product life cycles grow shorter, speed to market will be more important
- predictions...

  - market of PCs shrinking
  - market for DSPs growing
  - use modular designs in many different combinations
  - multiple circuit types/process technologies on a single chip
  - specialized designs to meet needs that cannot be met with general purpose processors
  - designs for small markets
  - designs for short-lived markets

9 Microprocessors in 2020

—David A. Patterson

- predicts that one computer in 2020 will have computing power equal to combined power of all computers in Silicon Valley in 1995!
- improvement in microprocessors in first 25 years: 25,000×
- silicon chips have led to countless inventions
  - portable computers
  - fax machines
  - intelligence in automobiles
  - intelligence in wristwatches
- predictions often overstate importance of radical, new technologies
- Patterson instead predicted evolutionary changes
- 2 inventions sparked computer revolution
  - stored program concept in late 1940s (basis of every computer since)
    - processor for crunching numbers
    - memory for storing data, programs
    - advantage: same hardware performs variety of tasks (just change program in memory)
  - transistor also invented in late 1940s
    - much smaller than vacuum tubes
    - made possible smaller, faster electronics

- timeline
  - decade between invention of stored program concept and transistor and first pairing of the two technologies
  - first microprocessor in 1971
    - Intel 4004
    - single silicon chip, size of a child's fingernail
    - first processor that could be made inexpensively in bulk
- manufacture of integrated circuits
  - add chemicals to silicon disk
  - place in oven
  - repeat about 20 times
  - in this way, create complex pattern on silicon
  - transistors, conductors, insulators
  - even a little dust or vibration spoils the product!
- key to cheaper, faster chips
  - larger wafers
  - smaller transistors
- number of transistors on a microprocessor
  - Intel 4004 had 2300 transistors in 1971
  - Intel P6 had 5.5 million transistors in mid-1990s
- trade-offs
  - more chips per wafer $\rightarrow$ lower cost per chip
  - wafers have grown larger but number of transistors per chip has grown $\rightarrow$ fewer chips per wafer
  - larger chips $\rightarrow$ faster chips
  - larger chips $\rightarrow$ more likely to contain defects
- rates of progress
  - 35%/year in mid-1980s
  - 55%/year in mid-1990s
  - actual speed of processors in mid-1990s 3$\times$ greater than speed predicted in early 1980s
- keys to progress

- engineers in industry have borrowed methods from researchers in universities
- quantitative approach in design
- careful experiments
- sensible metrics

10 Microprocessors

Despite claiming that it is "impossible to foresee what inventions will... go on to revolutionize the computer industry," David Patterson is a wizard who accurately predicted the technology of 2017 ($\tilde{2}020$), from virtual reality and voice recognition to cloud data storage. Essentially, his article describes the rapid shrinking of technology.

The passage talks about how the mass production of microprocessors is becoming easier and simpler nowadays. He gives us the example of making the microprocessors as baking a pizza, but the advantage is, we can produce a big amount of microprocessors at the same time while we only can make one pizza. And writing the logic for each microprocessor is like adding the topping of the pizza, that we can just make the pizza and add different toppings for different pizzas based on what people want.

The author also predicted that the microprocessors will become smaller and smaller in the future and to a certain degree that people cannot use eyes or even use light to detect the chips, we have to use a special way, such as X-rays, to build and produce the microprocessors in the future. And eventually one day, the rapid pace of improvement may well slow down.

- pipelining, superscalar designs based on von Neummann architecture
- pipelining, superscalar approaches no longer providing improvements in performance
- microprocessors became smaller, still overheated, required lot of power
- in 1980s—no heat sink needed
- in 1990s—medium-size heat sink needed
- in 2000s—monstrous heat sinks needed
- water cooling impractical, too expensive
- progress (with conventional methods) stopped because of power constraints

- CMPs—fit multiple processors in same space formerly used for one
- processors on CMP share connections to system, thus reducing power consumption
- CMP can achieve same or better throughput at half the clockspeed of single processors
- CMPs are the future
- CMPs do not rely on von Neumann architecture
- Moore's Law—double # of transistors on IC (w/o increase in cost)
- people will not pay premium for performance they do not need
- design efforts should focus on custom features (chips are stronger than needed)
- facilities can fabricate more transistors/year than design teams can use
- in 1997, fab facilities increasing at 60%
- in 1997, design teams increasing at 20%
- cycle seen in kinds of improvements
    1. focus on performance
    2. focus on reliability
    3. focus on convenience
    4. focus on customization
- improvement cycles seen in...
    - mainframe computers
    - personal computers
    - telecommunications
    - banking
    - hospitals
    - steel

11 Statistics

- microprocessor market = $40B/year
- 200 million transistors on 1 cm die in 2002

12 Glossary

Chip Multi-Processors (CMP)

clockspeed

Intel 4004 the very first microprocessor, developed in 1971

latency

microprocessor computer processor which incorporates the functions of a computer's central processing unit on a single integrated circuit, or at most a few integrated circuits

multithreading

pipelining execute instruction #n, decode instruction #(n + 1), and fetch instruction #(n + 2) at the same time

pipelining a method of processing that saves time proportional to the number of stages involved (e.g., one load of laundry is four steps and takes one hour, so 20 loads would take 20 hours. Pipelined laundry would only take 5 hours for 20 loads.)

stored program concept a computer model which prescribes a processor for crunching numbers and a memory for storing both data and programs

superscalar

superscalar and parallel a method that can multi-task at the same time so that we can speed up the whole process. (e.g., same with the laundry example, it will be faster if we can do 3 loads of laundry at the same time for each step, it will be way faster.)

throughput

transistors silicon switches to replace vacuum tubes in circuitry

Part II

Moore's Law

13 Sloane Bartelme

I learned many things from the "Moore's Law is Really Dead" panel. To summarize the panel, Moore's Law states that the number of transistors in a dense

integrated circuit doubles approximately every two years. These panelists were discussing whether or not this "law" is still viable in today's world. A point brought up by Margaret Martonosi, one that I hadn't really considered before, is that Moore's law does depend heavily on capitalism and the viability of computing technology development is directly proportional to how profitable it can be. She posits that, though we are continuing to make more and more efficient computer chips and even venturing into quantum computing, these new innovations won't be in demand and therefore won't generate revenue. However, I wouldn't count quantum computing out. Just like modern computers as they exist today have changed our society to justify their widespread use, so too could the quantum computing industry change our society to create "need" for quantum computers. I'm not sure how this would happen, but I don't doubt that it could.

I also agree with the point that because hardware space has improved so quickly in order to keep up with Moore's law, software engineers became sloppy, inefficient, and lazy in their design. I don't have enough knowledge of software or programming to definitively make a statement on whether or not this is true, but it seems like it could be true. I'm sure there are ways software could optimize its size to apply to different hardware.

14 Noah Carpenter

I found that one of the overall questions of the discussion was whether hardware should be better or if software should be improved in order to make computers better. There werent really any real statistics or graphs or any data shown, so its hard to know.

Based on the answers everyone participating gave, it sounded like both should be improved. Software should be improved, because it cant efficiently navigate the hardware. Hardware has to be improved because the speakers straight up said that the chips are so small that the logic inside them arent as efficient as they should be - as if the designers of the chips only added smaller components rather than fixing inefficient parts year to year.

One statement that did catch my attention was that Moores law was moreso economical rather than referring to the development of computers. This opens up a large range of questions, many of which were asked by the speakers. One in particular being about transistors, and whether or not we really need to improve them by making the components smaller, or if we just need to change the design to something completely different.

Obviously Im no expert, but I think both sound like theyre going to happen.

Theres still some room left to make all of the components smaller, but the industry has to evolve from microchips to something else at some point.

15 Anthony Delgado

John Hennessey prefaces the talk by sort of explaining what is meant by Moore's Law is dead. He details the origins of Moore's Law back in 1965 when Moore first postulated that transistor density would double every year and a half. A law which he revised a decade later, in 1975, to say that transistor density would double every two years. Hennessey claims that in modern times, the doubling of transistors is more or so accomplished every 4 years, quite the departure from Moore's Law. It is this departure, or "slowing down" of Moore's Law that the tittle of the panel alludes to.

Hennessey also bring up a term that is new to me, Dennard Scaling, which says that as devices get smaller, their energy consumption also drops at the same rate. For a while, this worked in tandem with Moore's law to produce an exponential improvement in computing not seen before. However, this only lasted about 15 years, before Dennard's Scaling did not hold. So, now Hennessey says that computing is entering a Dark Age, where traditional ways of trying to improve hardware via Moore's Law is no longer reliable. Instead, we must look elsewhere to solve this slowing in progress.

Dough Burger, one of the panelists said something that intrigued me, he spoke on the free-ride in performance that for so long Moore's Law gave us under pre-existing paradigms, but the end of that free-ride means a necessary shift in computer architecture; some clever new alternative needs to be found. He points to a few areas of improvement in existing systems. One being the inefficient General-Purpose Processors which are less than 1as I mentioned, being the development of new architectures.

Margaret Martonosi has an interesting take on Moore's Law actually is. She claims it was never a law of physics, but rather a law of economics or business. In this sense, the doubling of transistor density will only be pursued if the cost is work it, and the slowing down could be a sign of the cost not being worth it. So, companies and chip manufacturers are less willing to pour resources into maintaining Moores Law if the cost is too high. She also had a great point on updating curriculum for students as the paradigm shift occurs, because it makes no logical sense to prepare students for an antiquated world that no longer exists.

Butler Lampson is quite the character. The only YouTube comment on the video sums him up well, "Butler Lampson is so Sassy." Which I loved. He

challenged the other panelists quite well and kept the conversation going with factual information. Now, he had some great points on where the areas to improve the industry might be. He says there is much room for improvement in the "Top." Essentially, Software, Algorithms and hardware need a lot of work in terms of becoming more efficient. So, this might be where we find out advancement in the field within current paradigms.

Overall, a great talk! I will be looking more into Neural Networks, they sound interesting.

16 John Hoberg

Doug Burger was one of my favorite speakers in the panel. He described 6 new directions after Moore's law. One of the ideas he talked about was the evolution of architectures. He explains the ideas and differences of spatial computing and temporal computing. With temporal computing you have a small working set of data and you stream instructions through those and if you are changing that working set out then you're slow. Spatial computing is the transpose of temporal computing, you fix the instructions down and stream data through. Microsoft made big investments in FPGAs because you could put down these functions or instructions and stream data through at line rate. It's difficult and the languages are not there yet but Doug Burger believes we will see more of spatial computing in the future.

One very wild thing he discussed was the idea of programmable biology. He believes programmable biology needs computer architecture and computer science thoughts and ideas. There is architecture between gene expressions and protein pathways which we do not understand yet. Programmable biology and genetic engineering is extremely interesting. Cells are like tiny computers; they send and receive inputs and outputs accordingly. Biologists have been working to hack cells algorithms in an effort to control their processes. Nature Biotechnology researches have programmed human cells to obey 109 different sets of logical instructions. This could lead to cells being able to respond to environmental cues to fight illness, disease or created important chemicals. A cell could be programed with a NOT logic gate to NOT do something when it receives the trigger. Cells are complex and DNA does not have easy "on" and "off" switches. People are out there trying to debug and hack human cells, progress is slow but in the future we may see more of this computer science application to real human cells with programmable biology and genetic engineering.

17 Jerome Richards

First off I really need to watch more of these podcasts/panels, the only issue is they take so long to watch or listen to. There is so much information going through the panel that it seems to be hard to latch on to something concrete I can form opinions or conclusions on. Something I was interested in before but they only glossed over was quantum computing. Margaret touches on it as one of the possible "silver bullets" moving forward to allow computing power to increase its growth like before. Quantum computing is intriguing, but it doesnt seem that the industry is going to hold its breath and wait for it to be developed enough to become a standard. Its enthusiasts will have to drag it to prominence. One of the more important take aways from this panel for students is the bit about the vertical versus horizontal layering of jobs and responsibilities in the computer industry. Instead of focusing hard on a specific level of the stack like what tends to happen now, Margaret puts forward the idea of gaining specific skillsets in a vertical fashion, learning an area of study rather than an area of skills. (if that makes any sense, I may be butchering it). If anything, this panel highlights how volatile this whole industry still is. Even though Moores law is "dead," competition and innovation are still boiling and blistering into the next stage of development. There is a lot of uncertainty in the specific direction that the industry is taking at the moment, but that is merely a side effect of so many brilliant minds trying to find their own solutions. What I see in this panel is not an industry that is becoming stale, but one that is becoming more interesting and opportunity laden than ever.

18 Jason Wang

The video talks about the moores law is slowly dying and what should we do about it. There are several speakers and every one of them target on a different point of view of this problem. Moore's law is the observation that the number of transistors in a dense integrated circuit doubles approximately every 18 month. It had been mentioned by Gordon Moore, the co-founder of Fairchild Semiconductor and Intel. He gave out this idea at 1965 paper described a doubling every year in the number of components per integrated circuit, and projected this rate of growth would continue for at least another decade. In 1975, looking forward to the next decade, he revised the forecast to doubling every 18 month.

First, John Hennessy, the precedent of Stanford University, talks about from the definition of moores law to the dark silicon and dark era that human might went into. He first talks about the moores law had been slow down from 2000 and when it come to 2005, it become way slower. So he point out one of the

ways that company used nowadays which is multicore, which did not solve the problem of the moores law but still make computer run faster at the same time. This technology had been call dark silicon from him and he thinks human might go into this era that it is become a huge thing. And he thinks this is not what should we do.

Than, he introduce Doug Burger, who is currently working inside Microsoft and I want to keep a focus on him. Doug talks about two big ideas, what is moores law and what are the solutions. He said that Moores Law is a rate that had been correct before. So that we can also correct the moores law so that the new rate can still maintain for a long time. He also talks about we really cannot talks about what is the causing reason because there are too many factors. He used the example of global warming, that we really cannot say what cause global warming. Also, he talks about six different ways trying to save the situation. Most of them goes over different varieties.

Personally, I think that we should not just thinking about the moores law, if there are designs that can make computer to reach a faster speed, we should just adopt that. Such as the software nowadays that they talks about, most of the people claim that software become one of the defect of nowadays hardware designing. I think that we can just just trying to improve the development of the software and when we reach the limit of coming out new software algorithm that can make computer run faster, we can come back to hardware development again. At that time, we will have way better software to simulate and latest algorithms on software that might apply or give ideas on the hardware. So the development of hardware and software are rotated with each other but the development of computer science would never stop.

Part III

Lessons from the inventors of RISC

19 Sloane Bartelme

19.1 MIPS: Risking It All on RISC

- Speakers, all co-founders or colleagues at MIPS:

Skip Stritter from Boston, Computer Science PhD from Stanford. First job out of Stanford was designing chips at Motorola.

Bob Miller from Long Island, Computer Science PhD from Stanford. Worked at IBM with microprocessors for 15 years post grad.

John Hennessy also from Long Island, Computer Science PhD from Stony Brook. Taught at Stanford as an assistant professor.

Joe DiNucci from Pittsburgh, MBA from Duquesne University. Worked for DEC for 17 years.

- Terms:

  Backwards compatibility new hardware supporting old software. Backwards compatibility is tricky because all the design flaws or glitches present in the old hardware or software persist.

  microcode a layer of hardware-level instructions that implement higher-level machine code instructions in a computer.

  NMOS/CMOS Two different methods of semiconductor technology. CMOS allows for more gates per integrated circuit than NMOS.

  ARM

RISC Aptly named

(7:45) Stritter says that, at the time, there were other features and concerns with designing chips in 1984, but that RISC was all about making a faster machine hardware-wise to make software design as simple and easy as possible.

(9:04) Hennessy mentions that, at one time, competing companies had instruction set count wars where they would brag about how many instruction sets their machines had, even if no one was ever executing those instructions. The MIPS team designed their machines with the compiler as their focus. This allowed them to emphasize and prioritize efficiency of compilers in a way that none of their contemporaries were even attempting.

"Rather than building microcode up to the compiler, we would take the compiler down to the level of hardware." (Hennessy)

These ideas were revolutionary at the time because they completely flipped all methods of computing entirely on their heads and created something new.

(16:05) Stritter says they had an idea of what their ideal architecture and software would look like/how they would interact, but they had no chip that was viable, so they designed a software simulator to start creating a chip that would work.

- Creating MIPS and moving forward with RISC was already a gamble, and even so, the panelists agree that if they had moved forward sooner and taken bigger risks the payoff would have been greater.

- Basically, these guys set out to create a completely new technology with no money, no help, and no idea if what they were trying would even work. Lots of tech innovators and creators in the past fifty years have made gambles like this on tech. This echoes the message from some of our earlier readings (Microprocessors in 2020, etc) that theres really no reliable way to predict what creations will become successful in the tech world, and success doesnt even guarantee longevity, as was the case with DEC.

20 Noah Carpenter

20.1 MIPS - Risking it All on RISC

(Watch from 34:58–39:14)

20.1.1 Speakers

Dave House Mediator

Skip Stritter From Boston, went to Dartmouth, Stanford for PHD, worked at Motorola, returned to Silicon valley

Bob Miller From Long island, went to Bucknell University, Stanford worked at IBM for microprocessors. Went to Data General (thats what I heard.

John Hennesy from New York, went to Villanova, Stony Brook, worked a Stanford as an assistant professor, worked at Silicon Graphics

Jared DiNucci Went to Carnegie Tech, worked for digital

Dave House worked at Intel

20.2 Segment Summary

The question asked was covering the topic of software issues and incompatibility, giving way to ACE, and was answered by Bob Miller. Bob Miller answered by stating that Digital and Microsoft's 'software guys' were needed an API in order to make a common API. After asking software companies to help, they would agree if given a substantial amount of money.

In response to this, IBM, with the help of silicon graphics, made the API and found many applications for the newly created API. Bob Miller says that nowadays, similar situations are going on with a 'cloud,' such as google's.

John Hennesy then began to explain that people figured that creating a higher level language would solve the problem. As a result, software companies came out with 'shrink-wrapped software,' so if you wanted to utilize the software, you'd have to hire someone new, and you would also have to pay the software company to update their software. John and David were in agreement, so we didnt get to see any fights or arguments.

To finish up, Bob Miller explained that giving the application developers a consistent interface would make their lives easier, and benefits everyone.

ACE Advanced Computing

Indian (Couldnt find an answer - google thought I meant the nationality)

UNIX an interface, such as Windows or MAC OS, designed to be effective most specifically for human interaction. - Computersciencedegreehub.com

API Application Program Interface is a set of commands, function, and protocols that programmers can use to create software, by utilizing standard commands for common operations so that they dont have to start a whole new code.

Shrink-wrapped software Software sold to companies, rather than being developed by a companies own programmers; usually required hiring a new employee with expertise in the software - Businessdictionary.com

21 Anthony Delgado

21.1 Instruction Sets Want To Be Free: A Case for RISC-V

David Patterson is a notable American Computer Scientist. He has been a professor at UC Berkeley since 1976. His most important contribution to Computer Science has been leading the Berkley RISC project which lead to the creation of RISC (Reduced Instruction Set Computers) architecture. (Source: https://en.wikipedia.org/wiki/David_Patterson_(computer_scientist)

Patterson speaks on the importance of making Instruction Set Architectures open source. (19:30)

- For performance ISA don't matter as much as other factors (algorithms, compiler, OS, etc), but for MONEY it matters a lot! (20:20). Most of the cost in a new chip is developing the software for it.

- So why no open source ISAs until now? Business! Companies will legally come after you to keep their ISAs a secret because of fear of losing money. (21:30)

- "Proprietary Instruction Sets are tied to the success of the companies who owns them. . . ," when companies go under, the Instruction Sets disappear like the case with VAX and DEC. (23:30)

- Some benefits: lower cost due to the reusability of code, and much shorter time to market.

- Fewer errors because of more eyeballs if made open source. More eyeballs refers to more people proof reading and checking the code. (24:00)

- ISA open source: would make it HARDER for governments to make backdoors in your devices because of transparency. (24:10)

- **Backdoors**. In an article titled "Hacker Lexicon: What Is a Backdoor?" publish on wired.com by Kim Zetter, backdoors are essentially undocumented secret portals for hackers or intelligence agencies to gain unfettered illicit access to some software or hardware. (https://www.wired.com/2014/12/hacker-lexicon-backdoor/)

- Last year WikiLeaks published the Vault 7 leaks detailing CIA surveillance practices. One of the mentioned tactics involves backdoors in many popular android and apple devices that allow collection of audio and message traffic before any encryption is applied. Which circumvents messaging encryption services such as Signal, which is an encrypted messaging service. See more: https://en.wikipedia.org/wiki/Vault_7.

22 John Hoberg

- RISC Architectures: MIPS vs. ARM discussion

- MIPS: Risking it all on RISC

- Panel consists of Skip Stritter, Bob Miller, John Hennessy, and Joe Dinucci

- For the section I choose the speaker is primarily Bob Miller

22.1 Background on Bob Miller

- Bob Miller: Grew up in Long Island, New York
- He went to school at Bucknell University where he got degree in engineering.
- Went to graduate school at Stanford then worked at IBM for 15 years and was responsible for microprocessors strategy at IBM
- Working at Data General, he was senior executive and later recruited to MIPS by Skip and John
- Was the CEO of MIPS

22.2 RISC in General

- Reduced Instruction Set Computer is one with an ISA (Instruction Set Architecture) that allows for lower cycles per instruction than CISC (Complex Instruction Set Computer)

22.3 Question From Moderator

- 50:15- 53:47 begins to discuss two models of RISC architecture that exist, MIPS and ARM. What would MIPS have needed to do in order to have the numbers that ARM has today. This video was uploaded in 2011
- MIPS made a mistake and didn't realize how important the cell phone market was.
- Believe one could not make money in it and ARM took hold of it (investing) and got the leverage associated.
- In this industry as Bob Miller says, there is a huge change every 10 years. ARM won the flip phones and now it is smartphones. Smartphones are becoming the device where everything will be done. Bob hopes that due to the open nature of the MIPS license they will prevail in the end. MIPS licensing model allows semiconductor manufacturer or smartphone or any manufacture to add "value." It was a design that said if you paid us your royalty you got to it. It was not a license that said here is a design and you have to use it this way
- Open means a lot of people can contribute. Similar to the idea of open source software. It allows this collaborative nature of feedback and collective efforts

22.4 MIPS

- Microprocessor without Interlocked Pipeline Stages (Since 1980), Many market applications
- Open Source Implementations available
- Simple coding syntax, ease of use, used for educational purpose due to simplicity
- Cost of devices using MIPS is less than other RISC architectures, Long term Graphics and general support
- OVERALL: Low cost, high functionality and performance

22.5 ARM

- Advanced RISC Machines (since 1985), Support 64bit architecture
- Development of architecture depending on the market (ARM-v8-A for high performance markets like mobile devices and ARM-v8-R for embedded devices like automotive, industrial control
- Variety of units according to demand. Some include Virtual platforms, code generation tools and debug tools, Every device configured to consume low power
- OVERALL: Low cost, adaptability, variety of low power-high performance devices gives it a large market.

22.6 MIPS vs. ARM: APPLE

- The phone industry is basically ARM dominated, android, apple and windows phones all use ARM Architecture.
- Custom apple processor core use ARM instruction set, pay small royalty to ARM holdings. Apple wants own implementations to run code faster/ more efficiently than ARM-based chips
- Apple may be interested in acquiring MIPS processor assets which are being sold in hopes of cutting ARM holdings out.
- This is likely to not happen because transitioning their app ecosystem to MIPS64 instruction set would not be easy and app developers would need to rebuild entire apps to be compatible.

- Processor teams would need to move to MIPS64 when very well acquainted with ARMv8. Would do it if gains were significant which they seem to not be.
- MIPS engineers at Imagination will probably be looking for new opportunities after and prior to the sell and maybe apple will buy their MIPS division and bring them to apples processor teams to have great chip engineers.
- Imagination better get to selling the assets as the potential outflow of talent would diminish the value of their assets to buyers.

22.7 Sources

- MIPS, ARM, and SPARC—An Architecture Comparison, by Sarah El Kady, Mai Khater, and Merihan Alhafnawi
- Comparsion between RISC architectures: MIPS, ARM, and SPARC, slides by Apurv Nerlekar
- Reduced Instruction Set Computer (Wikipedia article)
- course handout from Professor Eric Roberts (Stanford University)
- Here's Why Apple, Inc. Won't Dump ARM for MIPS, by Ashraf Eassa on *The Motley Fool*

23 Jerome Richards

- URL takes you to the video and time that prompted this
- Dave Patterson is a "Computer Pioneer" who was a professor at Berkeley for 40 years, recently retiring. He help create RISC design, and is currently at the RISC-V foundation.
- Why open source ISA is needed instruction set architecture
  - RISC-V specifically
- No standards or open implementations
- Choice of ISA doesnt have much impact on system performance or energy consumption.
  - So there is no reason to have closed ISAs other than historical business ideas.

- Without standards ISA life is tied to the companies life-using them has risks
- This among other reasons, there is no good technical argument for the lack of free and open ISAs.
  - If an open source ISA is made standard, there will be more players making the core industry stronger
  - A hotbutton issue is privacy. With an open source ISA and open-source cores make it harder to add backdoors secretly.
- Why NOW
  - Dave patterson says that with the ending of moores law the only way to replicate such dramatic increases in technology is with architectural improvements
- RISC-V — plans to become an industry wide standard architecture. Dave believes that if enough individuals jump on board, companies will follow and open source ISA and cores will become viable. (very much like Linux)
- The RISC-V Foundation today is a non-profit organization driven by its members and overseen by a board of directors. All members have access to and participate in the development of the RISC-V ISA. Members must pay a fee with different rates for varying levels of membership
  - lowest level is an individual membership at $99 a year. Going up to the highest level is 25,000 a year
- RISC-V even today is still a new thing and only time will tell if the semiconductor industry will accept RISC-Vs rise to a leading role.(information taken from the RISC-V website and an online article written by one Prakash Mohapatra)

24 Jason Wang

1985 logic is more expensive than ROM/TAM, ROM ¿ RAM

1978 RAM = ROM on speed, moores law thinks control store could grow, allow complicated instruction sets (CISC) VAX ISA, VAX microcode bugs → field repair

CISC - RISC PC ERA sells > 350M / year and X86 ISA (RISC) dominate servers as well as desktops

Nowadays post PC era: Client/Cloud, IP in SoC vs. MPU, value die area that energy as much as performance

1990 VLIW compiler responsibilities, no interlocks, solution to make it easy is change loop from $i++$ to $i+=4$.

VLIW failure because unpredictable branches, unpredictable cache misses and code size explosion

2000 how should we build scalable multiprocessors

What do we want 1.

- Shared memory with "non Uniform Memory Access"NUMA time for loads and stores — DASH/FLASH projects at Stanford 1992-2000.
- Message passing cluster with separate address space preprocessor using RPC(or MPI) Idea is that from there is a switch from different cpu to cpu

NUMA advantages 1. Easy to programming when patterns are complex or vary dynamically during execution. 2. Ability to develop apps using familiar SMP model. 3. Because did it in hardware, lower communication overhead, have local memory to grab data that use the most.

Non-uniform memory access (NUMA) is a computer memory design used in multiprocessing, where the memory access time depends on the memory location relative to the processor. Under NUMA, a processor can access its own local memory faster than non-local memory (memory local to another processor or memory shared between processors). The benefits of NUMA are limited to particular workloads, notably on servers where the data is often associated strongly with certain tasks or users.

Very long instruction word (VLIW) refers to instruction set architectures designed to exploit instruction level parallelism (ILP). Whereas conventional central processing units (CPU, processor) mostly allow programs to specify instructions to execute in sequence only, a VLIW processor allows programs to explicitly specify instructions to execute at the same time, concurrently, in parallel. This design is intended to allow higher performance without the complexity inherent in some other designs.