Professor Dr Steve Furber

Interviewed by

Richard Sharpe

15 June 2018

At the

BCS London Office,

5 Southampton Street, London, WC2E 7HA

Kindly provided by BCS – The Chartered Institute for IT

Welcome to the Archives of Information Technology, where we aim to capture the *past and inspire the future. It is Friday the 15th of June 2018, and we are in the HQ of the British Computer Society in London. I'm Richard Sharpe, and since the early 1970s have been researching on and writing about IT.*

You will all have made a mobile phone call, and used the ever-growing range of applications for mobile devices. So you owe something, a lot in fact, to our entrant to the archives today, who is Professor Dr Stephen Furber, CBE, who co-designed the chip that is in over a billion mobile devices.

[00:47]
Professor Furber, you were born in Manchester in 1953. Your father was a nuclear scientist?

My father was a mechanical engineer, who spent his early career at UMIST in Manchester as a PhD student and lecturer, but in the mid-Fifties, so shortly after I was born, he moved to the Nuclear Power Group and he spent the rest of his career in the nuclear industry.

Right. And, after bringing you up, your mother, who had trained as a physiotherapist, took up the work of a maths teacher, is that right?

That's correct, yes.

And did that influence you in any way?

Well, I was well through school at the time. I was educated at the Manchester Grammar School, and, it was clear from an early age that, that mathematics was my forte, so, clearly both of my parents are of a mathematical bent, but whether my mother's actually becoming a maths teacher was an influence, I'm, I'm less sure.

Here's a philosophical question. Is mathematics an imposition by humans on the world, or is mathematics actually in the world itself?

Oh mathematics is, is a natural system of pure thought. And therefore, it exists both in, in the world as it were, as a useful tool for describing how the world behaves, but it also exists as a, as an abstract formulation of, of the properties of numbers.

[02:26]
What drew you to mathematics to begin with?

I think, because I, I found it easy, I found I was good at it. So I, I was a natural mathematician from quite an early age.

You passed your Eleven Plus after going to your primary school. You must have passed your Eleven Plus, because you went to a grammar school.

Well actually, I passed my Eleven Plus *after* I went to the grammar school.

OK.

I went to Manchester Grammar at the age of ten, and I didn't take the Eleven Plus until toward the end of my first year there.

Would you consider yourself to be a child prodigy?

No. I, I was… I was fairly bright, but I wouldn't put myself in the prodigy class.

But you had, in Hungary I believe in 1970, you were seventeen?

Yes.

And you got a prize at the Olympiad for mathematics. Is that right?

Yes. MGS put a lot of its mathematicians in for the National Mathematics Competition, and this competition was used to select members of the British team to go to the International Mathematical Olympiad, and I was fortunate to be chosen, in 1970.

So, and Hungary was behind the Iron Curtain then.

Yes it was. I mean the, the International Mathematical Olympiad was, predominantly European, I don't think it was exclusively European, but, but the East Europeans had a very strong emphasis on mathematics education, and they were very good at it.

[03:59]
What were your mathematics teachers like at the grammar school?

We had a range of, of maths teachers in my early years. It was, a schoolteacher called Maugham, I think, I think he was known as Killer Maugham by the pupils, and I can't remember why. He was very friendly. In the sixth form I had three mathematics teachers. I used to describe this as, let me get this right, I think, Mr Wilkinson taught us pure maths, Mr Schofield taught us applied maths, and Mr Copley, taught us whatever he felt like. And… And he was very interesting. He had all sorts of mathematical stories to tell. He was involved in the development of the resonant cavity magnetron, which was developed in the Second World War as the first system that could generate radio waves that were sufficiently short that they could be used to detect submarines. And of course, after being used for hunting submarines in the Second World War, they then turned up in everybody's kitchen, in their microwave oven, which was an interesting story.

Yes. And that was the type of story that you relished.

Oh yes. I loved that, yes.

Right.

Yes.

Right. And he was, you think, had most influence on you?

No, I, I don't think so. I think, I think they were all, they were all good teachers. I think, you know, MGS, then and now, was, was a pretty academic school, and, and attracted the sorts of teachers who enjoy dealing with the more academic pupils, and, and they were the sorts of people I enjoyed interacting with.

[05:48]
So you did, you flew through your O Levels, maths, English language, physics and chemistry, French, German and Latin.

Yes. A classical education. [laughs]

What?

A proper classical education.

Yes indeed. No, art there, as in painting or sculpture or…?

No. And in fact, by today's standards it's quite a small set of O Levels, if you look at the sort of ten, eleven, twelve GCSEs that my daughters took at school. This was a minimal set, and, I think this was a, a conscious choice by the school. You see history isn't in there. We took history classes. But the history staff did not like the O Level syllabus, so we didn't take it. Likewise geography. We, we took geography at school. I can't remember quite what we did by way of art. There must have been something.

[06:43]
And then two years after that you just romped through your A Levels, maths A, further maths A, and physics A.

Mhm.

And, S Levels, you got maths and further maths.

Yes.

So you were really mathed up all the time.

Well yes. And of course, you know, physics is not very different. I mean, much of physics is maths, is more maths too. So, yes, it was a fairly solid set of mathematical exams.

And, naturally you applied for Cambridge. Had anybody in your family been to university?

My father was, was a, a graduate of UMIST, and in fact he had a PhD from UMIST. So… And before me, my elder sister had gone to York, to study maths. So, you see a trend here.

[laughs] Somewhat in the genes was it?

[laughs] Yes. My younger sister actually became a doctor, so she didn't pursue the maths line, but, the rest of us were fairly maths-orientated.

[07:45]
Right. Right. And you applied to Cambridge. That was a, a natural selection for you, or, why Cambridge?

It was a very natural selection. I mean MGS was, was keen to send as many people as possible to Cambridge. It was partly reputational, but partly because it, for mathematicians, it felt it was the right place to go. In fact, I think then, the university application form was called a UCAS form, it's now an UCCA form, or is the other way round? No, it was UCCA then and UCAS now. Anyway, you could fill six universities in, but on the school's advice I just put Cambridge on the first line and left the rest blank. So… And because I was a year young, I was, I was sixteen when I took A Levels, and the school's advice was, if you don't get in this year, try again next year.

Right. OK. And you got in that year?

I got in that year. In fact I got a, a Baylis scholarship to St John's College.

Did thy interview you?

No.

They just said yes?

Yes. There was no interview. The first time I saw Cambridge was, was the day, one fairly dark October evening, dark and wet, and my parents had dropped me off at St John's, and that was my first encounter with the place.

Had you lived away from home until then?

I had, because, I finished school eighteen months before starting at Cambridge. I started in Cambridge in '71. I finished school, well it must have been Easter 1970. So I had eighteen months, and, the first nine months of that I actually spent working at, with my father's employer, on a temporary job at the Nuclear Power Group, and then the second nine months I went to North America. I spent a term at McGill in Montreal, sort of, attending any courses I felt like. And then, worked on a summer camp in Meadville, Pennsylvania, over the summer.

What struck you about McGill?

Well it was my first encounter with the university of course. And, and… I picked courses naturally of a mathematical leaning. And I… I remember some of the things we were told about, such as the concept of, the simplest possible computer I think, the simplest concept I had come across, where you had a set of labelled holes with stones in, with instructions which involved taking a stone out of a hole and putting a stone in another hole, and, and a conditional execution, if you tried to take a stone out of a hole and there wasn't one there then you did one thing, and if there was one there you did another thing. And this is a Turing complete model, so you can basically do anything with it. But I, I found that very interesting. And, and the other thing I think was… I had a little bit of programming at school. Neil Sheldon had organised computing classes right at the end of my sixth form where we physically pushed the little squares out of pre-punched 80-column cards. I mean, they only pre-punched 40 columns. So, they were 80-column cards with half the columns pre-punched and pushed the little squares out. And then posted it to Imperial College, and two weeks later, the printout came back. So there was a two-week debug cycle, which taught you to be very precise. But then at McGill, I had access, direct access, to computers, and I, I'm pretty sure it was there that I wrote my first program to, to play Conway's Life Game.

[11:15]
What was this computer, do you remember?

I don't, I don't remember what the computer was at McGill, no.

Right.

I, I think I was probably programming in FORTRAN, but I'm not precisely sure.

How many languages can you program in?

Well, I'm not a programmer in, in the normal sense of the word. I've not done a lot of serious coding, as my, my career has been to do with the hardware rather than the software. But I, I spent my three undergraduate vacations working at IBM's sale office near Manchester, and the first year I was there, I think it was PL/1, and the second year it was COBOL, and the third year it was FORTRAN, or maybe… So each year I went there, there was, 'Here's a language manual, go home and come back on Monday and start programming.' I had also learnt Modular One at Cambridge. BASIC of course, with the BBC Micro. I did a little bit LISP on the BBC Micro. And, these days, my main programming, when I do it, which is not very often, is, is either Python or Jobe or C.

I would have thought you might have programmed in APL. That would have perhaps been your forte, being a mathematician.

I don't think I've come across APL. And of course I've used MATLAB as well.

Right. Right. Right. What do you think of C as a programming language?

Well, I, I think, C is a thinly disguised assembly code. It's a… It, it's quite a nice language if you need to keep very close to the machine. So, on the SpiNNaker project, which I guess we'll get round to at some point, the run time code is all written in C, because it, it's real-time code, it has to be very efficient. You know, you can measure the number of, the amount of time available for a given routine in, in two figures of clock cycles. And, a lot of languages don't allow you to know when you write the code how many clock cycles it will take to execute. The control stack, SpiNNaker is already in Python.

I would have thought you would like that, getting really close to the hardware, to see what...

Oh I do, yes. And in fact, you know, I, my team often looks under the compiler and sees what assembly code it produces, and groans, and, improves it by hand there.

Yes. I once had to try and rewrite a compiler on a commercial project because, the compiler, which was for COBAL, was producing such garbage. [laughs]

Yes. Well the, the particular problem we have with, with the GNU C compiler for ARM, is, we needed a fixed point library for some of the stuff that runs on SpiNNaker, and the fixed point library is relatively recent and relatively clunky.

[14:23]
You got a First in your BA in Maths from Cambridge. And you went on to do Maths Part III.

Yes.

What does that mean in Cambridge talk?

Well, in Cambridge, you, you take a Tripos, OK, and a Tripos, as the name suggests, comes in three parts. And, the standard undergraduate degree, the first year is Part IA, and the second year is Part IB, and the third year is Part II. So, you've done your three years and you get your Bachelor's degree, but you've not completed the Tripos. And so there's a, a third year – a fourth year, Part III of the Maths Tripos. Which, which a relatively small proportion of people go on to take, but if you, for example, want to carry on to take a PhD in Maths, then you have to do Part III Maths. At that time, it was not a, you didn't get a qualification at the end of it. You just, did Part III. 35 years later the university decided that Part III was worth a master's, so, so I got my MMath from Cambridge. I took the course in two thousand and… no, in 1974/5, and I got the degree in 2010.

A nice piece of paper to have.

Yes. And of course, the, the convention at Cambridge is that, when you receive a degree, you wear the gown of your highest previous degree, which in my case was a PhD, so I wore my PhD gown to receive my Master's degree, which was…

It's not as if you needed the piece of paper, because by then you're a CBE, a Fellow of the Royal Society, a fellow of, a fellow of this, et cetera et cetera. But, nice to have.

Yeah, it's, it was interesting to go back and… And of course the maths department had moved out into completely new buildings, and it was very nice, looking around. Their old buildings were fairly depressing, the old Cambridge University Press buildings on Silver Street.

Right.

They have much nicer premises now.

[16:27]
And then you did your PhD on aerodynamics.

Yes.

What drew you into that?

Well, I, I… In the course of Part III Maths, I did… I had some interesting teachers, lecturers. I had to do a project in Part III Maths, and the project I chose was on the intermittent flight of birds. And my supervisor for that was Sir James Lighthill. And so, the idea here was, if you observe small birds, you see they don't fly continuously, they fly in little bursts, and fold their wings up and, follow a, a projectile trajectory between flapping the wings. And, I did some analysis to show why this was beneficial in energetic terms. But, one of the courses that was normally offered in Part III, but was not offered the year I took it, was a course on aeroacoustics, given by Professor Ffowcs-Williams from the engineering department. But I thought this sounded closest to my interests. I had always had an amateur interest in aeroplanes. You know, I flew model aeroplanes as a teenager. And… And so I contacted Ffowcs-Williams and, and asked him about PhD opportunities. And he was happy to take me. So I, I changed from maths to the engineering department for my PhD.

But with this solid grounding in maths, so, that really didn't faze you.

No. I was a fairly theoretical engineer. Although I did end up doing some real experiments, but the basis of my PhD was theoretical.

Right.

And, and it was… It had strong links with Lighthill, who I had worked with n Part III, because my PhD is entitled, 'Is the Weis-Fogh principle exploitable in turbomachines?' And Torkel Weis-Fogh was a biologist who had studied the flying motions of a whole range of animals, and he discovered this tiny wasp, the chalcid wasp, which is important in the control of aphids in greenhouses. Used a different flying mechanism from most other insects, and, it basically, instead of just moving its wings back and forth, it'll actually clap them together at one extreme of the motion. And Lighthill had done a theoretical analysis of how this clapping was actually beneficial, given the Reynolds number at which this fly operated. And it was that idea, Weis-Fogh's observations, Lighthill's explanation, that I then took for my PhD as the starting point, and tried to think whether there were similar mechanisms that you could use to improve the efficiency of a jet engine.

[19:15]
And that was three years' work.

Yes.

Thereabouts. So that was awarded in 1980 eventually.

Yes. I mean, in those days, the three-year PhD, the three years was treated fairly flexibly. I must have submitted it in, '79 I think.

Right. And you were in Emmanuel College as a Rolls Royce Research Fellow.

Yes.

So they were willing to, Rolls Royce were willing to put money in to academics like you, to see what you came up with.

Yes. Well, of course they didn't, Rolls Royce did not choose me. Ffowcs-Williams was very well-connected with Rolls Royce, he had a lot of, a lot to do with sort of, the work on trying to make Concorde quieter. And, and he persuaded Rolls Royce to fund a research fellowship at Emmanuel College, which was then advertised, and so I applied for it and I was fortunate to be successful in that application. So, yes, I got the Rolls Royce Research Fellowship. It didn't come with a company car. [laughs]

Oh. Or a jet engine.

Or a jet engine. Well of course, the Rolls Royce car company and the aero engine company are separate.

It was separated by then.

Yes. [laughs]

[20:29]
And, that was until '81. And, you are in the milieu of Cambridge.

Mm.

The Fen. And lots of people are playing about with microprocessors and electronics. And you go to a club, don't you, a microprocessor club?

Yes. I, I…

Over a pub.

I heard about the formation of the Cambridge University Processor Group, CUPG. I was not a founder, but I went to the formation meeting, so I, I… I had heard about it. And I was interested, because, as I've mentioned, I was, I was very interested in aeroplanes, and, and I had sort of, turned this interest round to this idea that, you know, maybe it would be fun to build a flight simulator. What do you need to build a flight simulator? Well obviously, you need to start with some kind of computer. So I thought I'd go along to this society and, and see what was going on. And, it was a very interesting society. It was…

So who was there?

It was founded by people who build computers for fun. The real men used discrete logic, TTL chips as they were then, to build their computers, and the wimps like me bought these new-fangled microprocessor things. And we were doing very scary things, ordering chips mail order from California, using credit cards. And this was all very new-fanged in those days. And then, hand-assembling them into machines, and then, going around to each other's houses, being impressed by what each other could build, and finding bugs and so on.

And, well I should think one of the delights would be to find a bug in your colleague's machine, yes?

That's right, yes. I was, I was not especially good at that, but, that, it was in the Processor Group that I first met Sophie Wilson, who took great pleasure in finding bugs in my computer. [laughs]

Right. Was Hermann Hauser there?

No. Hermann, at that time, was a, a postdoc at the Cavendish Lab. I think his science interest was extremely high speed photography. But he had started talking to Chris Curry about forming a company in the microprocessor area, because microprocessors were clearly going to be important. Chris Curry of course was working for Clive Sinclair at the time, but, getting a bit frustrated by Clive's reluctance to let Chris go and play with his computer game, and so, he and Hermann cooked up this idea of starting a consultancy. And, and as the idea formulated, they clearly thought that the Processor Group was a good place to go and look for potential staff, people who could do the technical work. So, so Hermann found me through the Processor Group, although he wasn't himself an active participant.

[23:23]
So Hermann is, as ever, putting these things together, these different elements, and seeing what happens to them.

Yes.

I interviewed him just recently, and it seemed that that was one of his fortes. He had seen something here, something here, and somebody there, and, just put them together and see what would happen.

Yes, well of course Acorn was probably Hermann's first such attempt at assembling external factors. I say Acorn, of course it was originally called the Cambridge processor Unit Limited. Acorn was, came in as a trading name initially before it took over the whole company.

And it was a very small company.

Well it started from nothing, as, as do, as do all companies. Chris and Hermann decided to set up. Thy looked for people who could do some of the technical work. I was involved from the outset. In fact I think the first meeting, where they talked to somebody else about the company, was in the Fort St George on Midsummer Common. And I think it was Chris, Hermann, Chris Turner, and myself. And at that time Chris Turner was working I think for Pye, on the other side of Cambridge. I was in the university. And they were beginning to sort of, put these ideas together.

And, you moonlighted with them.

Yes. I was, I was initially a student funded from SERC, and, and then I was a Research Fellow, so, both circumstances, you can't arbitrarily undertake additional employment, but I found what they were doing interesting. So we had a kind of deal whereby I would design bits and pieces, and hand the designs to them, and then they'd give me more bits and pieces to play with. So… So I was effectively funded in kind rather than employed.

[25:24]
And Acorn was one of just a slew of little companies with, some of them strange names, and all of them with their own particular approach to microcomputers, many of them not compatible with anything else...

Yes.

…apart from, from their own things. Some of them not even workable.

Yes.

And the public probably was a bit confused by then about what this was about.

Yes. I mean as I say, Acorn started this CPU Limited, who did a bit of consultancy work. I think, we designed a, a controller for a one-arm bandit for a Welsh company initially. But then, Chris Curry still had links with Sinclair, and they together formed Science of Cambridge, who produced this MK14, which is a, a kit I think which you could assemble, and it had a hexadecimal keypad and seven segment display, and you could put little programs in and… And Sophie Wilson looked at this and, as she said many other times, 'Er, I can do better than that.' And went off and designed what became the Acorn System 1. And at that point, Hermann and Chris's thoughts turned to selling computer products rather than running a consultancy. And the Acorn System 1 was, was marketed as a kit, and sold in quite reasonable numbers, and, and the whole business moved over to that side.

Did you make it yourselves? Did they make it? Or was it made for them?

Well it was a kit. So, so…

Yes, I mean…

They bought the parts…

The just bought the parts, did they, and…?

And, and put them in a box and…

Right, oh OK.

And the users then soldered them together.

Right.

And tried to make them work.

Right.

And that model was followed for the Acord Atom, which was the first sort of single-box machine. Well, it had the computer and the keyboard in; you still needed an external screen and an audio cassette or something to save your programs and data on. But the Atom… I was not at all closely involved in the Atom, that was rather a Chris Curry project, and Nick Toop did a lot of the design work. Sophie was by then an employee. I wasn't. And Sophie wrote BASIC for the Atom. But the Atom was the first sort of, package thing, still sold as a kit, until we realised that actually, the market for these things was significantly bigger than the number of people who knew which end to pick up a soldering iron. And we got some very strange returns with the kits. So we transitioned from selling kits to selling a finished product in the course of the Atom. But the Atom was really the foundation for, for what came next. But you're right, the market was incredibly diverse.

How much was the Atom?

I think it sold for £120, something in that region.

That was a bit of a price point then wasn't it?

Around 100 was, was a key point, yes, anything beyond 100 was, a lot of money.

[28:39]
Mhm Was this a robust machine?

The Atom was pretty solid, yes. You could make it less robust by adding things to it, and, and people added a lot of things to it. It was the first machine that, that could have an Ethernet module attached and do some basic local area networking for instance. Which in '79 was pretty advanced. But you could also put a ROM extender card in, so you could have more different bits of software. And, if you put enough things in, the power was regulated by a little regulator, and the printed circuit board was kind of upside-down, so the keyboard was on the back, and all the main components were underneath. And this poor little 7805 regulator would get so hot supporting all these added components that it would literally melt the solder, and drop out and rattle around in the box. [laughs]

[29:36]
One of the philosophical statements of this period was, it should be as open as possible, that you should be able, you the user who had bought this computer, should be able to plug whatever you like into it.

Mm.

And that was one of the great things that I think expanded this market relatively quickly, because other people then built peripherals for them other[?], the various devices for them. But Apple actually stopped that, didn't it, with the Mac. Are you an open systems person or a Mac person?

Oh, well we, the Mac is going quite a long way forward in time.

Oh yeah, yeah, I realise that. Yeah.

And there was, the other issue that of course Acorn discovered when it tried to take the BBC Micro into the North American market is that, at that stage, the Americans had much tighter rules on radio emissions than we had here in and in Europe. And, the FCC regulations in America were a real impediment to building the hardware in an open way. Because, in order to pass the test, you had to plug a metre of cable into every available socket, with nothing on the other end, and then they measured the radio interference. And, so the more sockets you had, the harder it was to pass the emissions tests.

Do you think that was deliberate?

No. No, I think… I, I think it was a sensible rule, because radio emissions from electronic equipment are a significant, has a, they do need controlling. And, and the controls are now universal. But then, you know, the BBC Micro was designed to be extremely open, it had lots of connectors. But to make these pass FCC was very challenging.

[31:27]

So here is the mighty BBC, Aunty, meant to be a, a rather slow moving organisation, but some people at the BBC obviously see something here that is a new phenomenon, and they want to educate and play a role in education. And so they look around for someone to build them a microcomputer to go in parallel with their programme, on, on the TV

Yes.

So they choose you.

Well, they had been working with another company for quite a long time. I think they had been working with Newbury. I don't know the details. But the story I heard was that they, they had become frustrated by the slow rate of progress of the NewBrain or whatever it was they were planning to build the programs around. And, and in the end they became so frustrated they decided they ought to throw the issue open.

Well I remember our correspondent at the time who knew this market tremendously well, and, I remember him putting the phone down and turning to me and saying, 'The boys in Cambridge have got it. Hermann Hauser and Chris Curry have got it,' the BBC contract.

Really? Yes.

Yes.

OK, well, you know, I, I don't know too much about that side of it. What I do know from the Acorn side is that, Chris Curry was able to attract the BBC's attention to at least get them to come and look. And, and then there was the famous weekend when Hermann played Sophie and me off against each other saying, 'The BBC are coming on Friday. Can we show them a prototype?' We hadn't even got a design at this point, let alone starting to build anything. We, we had a sketch for a thing that we called the Proton, which was the successor to the Atom which was a dual processor. And I had got as far as sketching a preliminary circuit diagram. And, by chicanery he persuaded us to go for a prototype and, and… And by the Friday, just, it was working. And in parallel, he had got Allen Boothroyd, who had done some of the industrial design for the Atom, to come and design, mock up a case for the BBC Micro. So five days later when the BBC came, and, I think, the visitors included Richard Russell, and I'm not sure who else. Over here we had a prototype, running code, and over here we had a case. This is what it'll work like. And the case, this is what it'll look like. And, and clearly, the rate of, the fact we had done that in a week, was a factor in persuading them to back the Acorn horse.

[34:09]
There's a common theme it seems to be with a lot of your work, and I call it, sort of um, pinpoint development. You don't necessarily like big teams, you don't necessarily think that big teams work. If you've got good enough people, a few of you can do some really remarkable things. Am I right?

Yes. I, I… I'm not particular negative about big teams. I've just never been involved in one. That… I do think, if you, if you have a, a key bunch of people to do the job, then getting a complex job done with a small number of people is, is probably easier than getting it done with a vast number of people. But, my career is characterised by the fact that I've, I've never been in a position to have a large team. Acorn was a small company. When we designed the BBC Micro it was very small, perhaps, 30 people I suppose. And when we designed the ARM, Acorn was a few hundred people, but of course the team to design the ARM was, was still, in the order of ten. If you designed a processor in industry today, you have teams of 100. I have no experience to know how to work that kind of process. Because I'm in a university, research grants are, are finite, and so we still have to work out how to do things with, with really quite small teams.

[35:41]
How many Micros did the BBC expect and plan to sell?

Well their initial discussion with Acorn, they were expressing confidence that 12,000 machines would be sold on the back of their programmes.

And how many did they sell?

About one and a half million. So two orders of magnitude was the sort of, degree of error in the early estimates. Nobody anticipated the public appetite for computing equipment at the time that, you know, the discussions were taking place. I mean the BBC I think quite rightly felt their duty to offer education to the wider public, but they had no sense the extent to which the public would lap this up, and actually spend real money to back up their interest.

And they did, spend real money, large amounts of it. And you went on the road, did you not?

Yes.

And had a series of seminars which were crammed with people.

Well, we… Yes, we, we took the show on the road, and, Chris Turner, Sophie and myself initially went to all of these. The first one was set up at the IEEE, it was then the IET, in Savoy Place. And they have a lecture theatre which seats seven or eight hundred I think in the middle of the building. And the first talk we set up there was oversubscribed, I think three times the number of people they could allow in turned up, and there were people who booked coaches from Birmingham. As I say, the, the degree of public interest was just phenomenal in this. It was the first time the computer was taken out of the hands of the man in the white coat and put in the hands of, of the general public. So, retrospectively, you can see why it was such a, a turning point in the development of computers. But, then it was very unexpected.

[37:38]
And they would be coming consumer devices.

Yes. Yes, a lot of them found their way into homes.

Yes. Yes. But I was totally puzzled by this, because, I entered the computer industry in the early Seventies, and, and I couldn't see what anyone would want a computer in their house for, but I was obviously very badly wrong.

Well a lot of people shared your views I think. This was, this was a common view, particularly of the more experienced professionals, who saw computers as work tools, and, and, perhaps… Well, why did they do in homes? Initially, it was just satisfying curiosity. People just wanted to familiarise themselves and see what these things would do. Very rapidly of course, other factors came into play, and one of the big ones was computer games.

Yes, one of the things that I remember is, constantly being asked to get down on the floor and play with a Sinclair, and plug it together for people, because they thought I knew about these microcomputers. I knew virtually nothing about them.

Mm.

But you were quite happy and quite confident and, comfortable with that transition into these domestic machines.

Well, it was really my entry into the business. I mean I, I had had some exposure to professional computing before then. But this was the first point at which I was professionally involved. And, and, you know, we, we started off dealing with the BBC. Then it became clear that the Government was going to give significant backing for its computer literacy project, and so we were going to end up in schools in quite a big way. And then we were going into homes. But, the thing basically just developed its own momentum. And there was no sense in which we were pushing. We were, we were more sort of standing back and watching, and, trying to anticipate what the next move would be, so that we would be ready for that. But, we made a lot of commitment to the BBC, and we had to develop quite a lot of ancillary devices. So I, I personally was responsible for the 6502 second processor, which you could plug onto your Beeb and make it effectively two or three times as fast and powerful. But there were other second processors. There was the magic Prestel box, OK, for downloading tele software over the airwaves. There were variants with more commercial computing and so on. So the BBC had a lot of interfaces, and it spawned a lot of activity. Yes, some of it was third party, but there was a lot of add-on activity at Acorn as well.

[40:29]
How were the people who had started by being enthusiasts, meeting over pubs and so on, able to run now an organisation of perhaps four, five hundred people?

Well the organisation… The organisation was run by, by Chris and Hermann. And, they scaled to that job quite well. They also of course employed professional managers to support them, so there were, there were people with experience. So it wasn't, it wasn't just members of the Processor Group running the whole company. People were brought in from outside, with relevant manufacturing and financial and other, management experience, to, to run the company as a serious going concern. But I think, you know, it, it's clear that, that Hermann has a, an innate ability to scale to jobs of any scale. I mean, Hermann was never fazed by anybody. He was quite interesting. He's very un-British in this respect, right. We tend to sort of, as Brits we tend to be a bit respectful of people who are, who are senior or who are royal or who are politicians, and, Herman just treated everybody as another human being. So he could scale into these problems very effectively.

[41:53]
Why Cambridge? Why the Cambridge phenomenon? Here's you, here's Sinclair, here's Pye just down the road. There's a multitude of different companies spinning out of Cambridge.

But it was, it was a slow start.

OK.

So if, if you… What we now call the Cambridge phenomenon, Acorn was really right at the very beginning of that. And, if you listened to the voices on the wind around Cambridge University, and, and you, you heard, you know, comparisons with Stanford, you know, they would always say, 'Well you've got this great university, but where's your billion dollar company? And there wasn't one. There was nothing at all to show for the very great academic strength of the university, in, in commercial terms. And, and, even Acorn… I don't think Acorn ever grew to be a billion-dollar company. I think it grew and, it waxed and waned before it reached that kind of scale. But of course, it then begat the likes of ARM, which scaled their wavy arms, and several other companies that followed. So the Cambridge phenomenon, a lot of its roots can be traced back to Acorn.

[43:13]
So the BBC Micro comes out in 1981. It's a big success for Acorn, big success for the BBC. Quite a bit of money made in it I imagine. And people, you included, are looking round in Acorn thinking, OK, what next?

Mhm.

And so you start looking at microprocessors. And you're not very impressed by what you see, are you?

No. The, the obvious next step from the BBC Micro, which used an 8-bit microprocessor, the 6502, the same processor that was used in the Apple II and the Commodore PET and, and several other products of the day. The next step was obviously 16-bit. And, this was driven by what was becoming known as Moore's Law. The number of transistors you could put on a chip was doubling every eighteen months to two years. And so the amount of functionality you could put on there was, was growing at that rate, and going from eight to sixteen was, was a natural progression. But we looked at all the 16-bit processors that we could buy off the shelf, and we didn't like any of them.

Of which there were quite a few.

Yes. There were, processors such as, I think, where was Intel at that stage? Probably 80186, probably coming on to 286. Motorola had the 68000.

National Semiconductor had them.

National Semiconductor had, a design that was initially called the 16032, but was later, became the 32016, and they changed the name without changing the internals much. And there were several others around at the time. And all of these had microarchitectures which were to some degree based on the very successful minicomputers of the 1970s. So the minicomputers had shown how you scale a machine up from, from quite small, to, to significant computing power, epitomised by the digital VAX-11/780, which was the original one-MIP computer, one million constructions per second. And, and the 16-bit microprocessors were trying to emulate that direction of progress. But this had, in our eyes, two significant drawbacks, the first on of which was, their real-time performance was very poor. And real-time means, if you generate an interrupt, how long does it take the processor to respond? We used interrupts very extensively on the BBC Micro; the 6502 had a good response. These other processors were much slower to respond, because they adopted these very complex instruction sets, styled on the minicomputers. And, the random fact I remember is that the National Semiconductor 32016 had a memory to memory divide instruction which took 360 clock cycles to complete. And it was running with a six megahertz clock, so that was 60 microseconds during which that processor was not interruptible. Now, a standard floppy disk delivers a byte every 64 microseconds, with an interrupt, and a double density floppy disk delivers it in half that time. So you couldn’t even handle a double density floppy disk with interrupts on the first row of sixteen. Of course there was a solution. They would sell you a DMA controller or something. So you could pay more to get more hardware to solve the problem. But we thought that was the wrong answer. The second thing, that was equally important, was, we had done a number of tests on a number of microprocessors, that drove us to the conclusion that the thing that determined the performance of a computer more than anything else was the computer’s ability to access memory bandwidth. And, as far as we could tell, performance scaled with this and not much else. So, if you had got a 6502 with as much bandwidth as a 32016, it was just as fast. Even though the 32016 had a, quotes, ‘nice instruction set’, and the 6502 had a rather simple 8-bit one, that made no difference to performance. And, memory was the most expensive part of a personal computer, and these 16-bit processors did not make the optimum use of that memory. They were slower than the memory. And this was clearly the wrong answer. So poor real-time response, and poor use of memory.

[47:44]
And we were scratching our heads about this problem when, I think it was Hermann dropped some papers from, particularly from Berkeley, but also from Stanford, on our desks, where Dave Ditzel and Dave Patterson were expounding the virtues of what they called the reduced instruction set computer. And the thesis they were expounding was that, the minicomputer style architecture was the wrong answer. If, if the goal was to put the whole processor on a chip. The number of transistors that it took to implement these complex instruction sets did not leave enough transistors to do some more important things.

And when you say complex, scale that for me. What would a RISC chip have as an instruction set, how many instructions?

Well, that's a hard question to answer, exactly as you posed it, but, we're talking about complex instruction set computers.

Yes.

The CISCs. They would have instructions which were something like, enter subroutine, or, or save the set of registers on the stack, or, or do this other complicated thing. The RISC processor would do something much more, much simpler. The way you would enter a subroutine would be to do a branch, but you would have to save return address. So you'd have branch and link. But that's all this instruction would do. If you then needed to save some registers to get some space to do some work in the subroutine, that would be another instruction, or, maybe several instructions, to push a number of registers onto the stack. So, so the subroutine entry, instead of being a single high level complex instruction, would be broken down into a sequence of small simple instructions. And it, it sounds like it's less efficient. I mean the, the complex processors were doing the thing that was the flavour of the day, which was called reducing the semantic gap between the instruction set and the high level language, and RISC processors were going the other way. They were increasing the semantic gap. But the argument was, if you kept it simple, then, you could use your transistors for things that were much more important than a complex instruction set, basic things such a pipelining the execution. And, and the thing on RISC that had the biggest impact on throughput was pipelining.

Pipelining meaning…?

Pipelining means that instead of executing the instructions by doing the fetch, and then you decode the instruction, then you execute it, then you store the results, then you fetch the next instruction. With a pipeline, you fetch an instruction, then while you are decoding it, you fetch the next one. And then while you are executing the first, you are decoding the second and fetching a third. So effectively, all these resources, which are usually separate bits of hardware on the processor, they're all being used at once on different instructions. Instead of using them sequentially, they're used concurrently. And this very easily gives you a factor three, maybe four, increase in throughput, with the same basic resource. But you need to control that. And…

[51:00]
So in 1983 you decided, you had better build, design, no, not build, sorry, design, your own microprocessor around the RISC concepts.

Yes. Fed with these published papers, Sophie started playing with the instruction set designs. And we were encouraged. Hermann, Hermann saw this as an opportunity. I mean, Hermann was… He, he's not deeply technical in the sense of being a computer architect, but he saw a good idea, he recognised a good idea when he saw one. And, and I think there's also the factor that he knew us quite well. So if we thought it was an interesting way to go, he kind of trusted us on that and backed us. So…

And it worked.

Yes. Sophie did the instruction set architecture design; I did the microarchitecture. There were other factors. I mean, Acorn had decided strategically, I think this was advice from Andy Hopper at the Computer Lab, to invest in silicon design tools, and silicon designers. So we had identified what we thought was the best silicon design software from VLSI Technology in California. We employed some experienced chip designers. But we didn't really have anything for them to do. There were some chips, there was the second generation of the electron chip, and, and one or two other things, but, basically, we had this capability, this capacity that was under-utilised. And so when, when I sketched a microarchitecture, they were there waiting, and they took the microarchitecture specs and, and did the silicon implementation. And, and within eighteen months we had the first working ARM chip.

[52:24]
April the 26th 1985…

That is the correct day, yes.

…you got your first chips back from VLSI.

Yes.

This of course presupposes the existence of, you have become then what is now called a fabulous chip company. You don't make anything.

Yes.

You don't make the chips yourselves. And it presupposes therefore that there are fabrication plants. And what you have done in this business model, consciously or unconsciously, is walked away from that horrible, horrible problem of, where do I get n, a billion dollars, to build a fab, and once I've got it, what do I do with it?

Yes. Yes, there was… Of course Acorn was not a, a classic fabulous semi, because a fabulous semi company designs a chip, gets it fabricated in a foundry and then sells the chip. Acorn was building desktop computer products. So we were developing these chips to go in our products, not as products in their own right. But yes, there was never any question of Acorn building its own fab. The foundry business was, was becoming reasonably established then. So the model was not new.

No, not entirely new, but…

And Acorn had… We, we had two semi-custom chips in the BBC Micro, which were initially fabricated by Ferranti in Oldham as uncommitted logic arrays. We had one or two problems with those, and so we looked for a second source, and that was our first contact with VLSI Technology. We then followed a similar path, the electron chip. So we had done chip design. But the ARM was our first attempt at designing the processor itself.

[54:40]
And the wheels came off the wagon somewhat, and Acorn was taken over by Olivetti.

Concurrently with the development of the ARM, yes, Olivetti came on the scene. The wheels came off because, Acorn had overextended itself in attempting to break into the US market, and it had also, the electron timing didn't work quite right. So the first, you know, Christmas was very important for electron light products. The first Christmas we couldn't make enough because of problems with the ULA, yield on ULA. And the second Christmas, when we could make huge numbers, suddenly the markets had gone.

The moment had, had passed.

So both of those things cost the company a lot of money, each. I think they were roughly equivalent in terms of financial penalty, and, the company was then effectively bust.

[55:39]
Did Olivetti have any idea what they had in this jewel of a RISC processor in the middle of this company?

No. In fact, in the negotiations with Olivetti, we were not allowed to talk to them about it. It was kept secret. When they finally agreed the rescue, to buy the company, we told them about it, and they had no idea what it meant, what its significance was. They, they were apparently quite happy to allow Acorn to continue doing this bizarre thing with their own processor and put it in their own products, and, and we sold the,

the ARM base products, starting with Archimedes, which I think was first shipped in '87. And then, developments of that, moving on to RISC BC. I think probably, that was in the Nineties by now. So, Olivetti sat at a distance and says, they've got their processor, they're putting in their own product. Don't know what you do. I mean, neither is standard, OK.

No.

We make Intel-based PCs, that's what everybody wants.

Well they were Europe's biggest weren't hey.

That's what everybody wants.

Yep. Yep.

Don't know what Acorn's doing, just let them get on with it.

[laughs] Sorry, I'm not, I'm not accusing you of not telling them. Or, or of, slipping this out the door quick and without them seeing. But you did get it out, into a separate company.

Ah. Now that's, that's later. So we, we went out in Acorn's products.

Yes.

From '87. And in fact, in 1990… I mean Acorn's business had grown exponentially in the early, in the first half of the Eighties, but then it went very flat. And it was, it was kind of, pretty much, you know, each year the Government declared its budget for computers in schools, and that was Acorn's business. They didn't get all of that, but it, it set a cap on Acorn's business. So it stopped growing. It was becoming increasingly hard to maintain the processor development, which was getting more expensive, to keep it competitive.
[57:50]

So, in 1990 I decided to leave the company and I, I moved to Manchester. And I took up my, the ICL chair in Manchester on the 1st of August 1990. And, sometime very soon after that, Apple came knocking on the door at Acorn, and they said, they were interested in using the ARM in their Newton products. Now, ARM had gone into Apple stuff through a Radius graphics accelerator sometime earlier. VLSI Technology had licensed the processor design from Acorn and had sold it to third parties. So there was some contact between ARM and Apple. But at this point…

Newton was a bit of a failure.

Well, not yet.

OK.

No no, Newton was, was Apple's big idea.

Yes, I, I think..

I think it was Sculley's big idea, John Sculley.

Yes.

But they had been playing with AT&T in the AT&T Hobbit as a processor for some time, but for some reason that deal was turning sour and they wanted to switch to ARM. Of course interestingly, Hermann started the Active Book Company with ARM, and then switched to EO with Hobbit. He went the other way. But, that was, a separate story. Apple wanted ARM in Newton, Apple was a very big name then, as, as now. But they were not comfortable with ARM being controlled by a competitor, albeit a rather puny one, in the form of Acorn. And so they approached Acorn with the proposition to set up a joint venture. And, they were all pushing on open door, in fact my last two years at Acorn, I had spent quite a lot of that time trying to find ways to set up the processor activity as a separate business. So, lots of the groundwork had been done. And we just couldn't work, we couldn't find a model that made business sense. But Apple came. They wanted the joint venture. Pushing on open door. And, by November that year ARM Limited was set up in an old barn in Swaffham Bulbeck, and, the processor was now in independent hands, Robin Saxby was brought in. He introduced this business model that made it all work. He found the trick that we failed to find when, you know, I had been thinking about it with, with the technical director at Acorn. And the trick of course was, the standard licensing businesses is a royalty business, but royalties come very slowly, and downstream. Saxby introduced the 'join the club' model, where you pay your licence fee upfront, quite a big licence fee. And of course, that's brilliant for cash flow. Royalties are terrible for cash flow, until they've really built up. The upfront licence fee is, is perfect for cash flow. It's a big chunk of money, very early in the negotiation – in the engagement.

And as he has told the Archives, this is Sir Robin Saxby, 'Over my dead body will we make those chips. We're only going to design them.'

Yes. ARM, I don't think, ARM Limited, ever thought about manufacture again. Actually, it's almost economically impossible to contemplate setting up manufacture in the UK, because the investment is so huge.

[1:01:23]
So you moved back to Manchester.

Yes.

And that gives you therefore a pretty well unique perspective. The two holy grails of computing, of IT, computing history, in the UK, of Cambridge and Manchester.

Yes. [laughs]

Manchester with the Atlas machine, Manchester with the, with the Baby machine. Manchester where Turing had gone. Cambridge of course with the Mathematical Laboratory. Cambridge with all that was going on in the Silicon Fen. Can you, very briefly, compare and contrast Manchester and Cambridge?

I can. I mean I, I have allegiances to both. I was born and brought up in Manchester, and I've worked there for, more than quarter of a century. Spent 20 years in Cambridge, at the university and then at Acorn. And, and I think, when I came to Manchester, in fact, that was really the first time that I sensed this historic tension. There was a bit of competition, and, and some ill-feeling I would say, going back to the very early history of computing. And, and some of that boils down the, you know, who really built the first stored-program computer? And, and you know, the historic answer is, the first operational stored-program computer was in Manchester, but it was rather small and prototype-y and not very useful, and needed expanding a bit. And the first usable stored-program computer was the one Maurice Wilkes built, the EDSAC, in Cambridge. Of course, Williams and Kilburn were not actually trying to build the first stored-program computer at all. They had this idea for memory based on cathode ray tube storage, which they had used during the war for, in analogue form for radar, and their question was, can we now turn this to digital purposes? And they built a very simple computer around it, as their idea of the easiest way to test this concept. But Manchester can certainly claim the first operational stored-program computer, and Cambridge can clearly claim the first stored-program computer that could support a sensible user service. And both of those are important.
[1:03:40]
Now, since then of course, Manchester has a continuous tradition of building big machines. And that isn't evident in Cambridge. I mean by the time I got to Cambridge, EDSAC was long gone, the Computing Service was using TITAN, which was actually basically a Manchester Atlas machine, and they were in the process of buying an IBM computer. So, so their dabbling in building machines was really a one-off with the EDSAC, and that was the first and last big computer. Roger Needham did lead a project to build a capability machine, I think in the Eighties. But Cambridge's contributions were, were very diverse. You know, Andy Hopper's work with, with high speed networking, it was highly influential. Manchester kept building machines. And…

And you had Ferranti beside you as well to help you.

Yes. Of course, you say 'we'. I was not at Manchester during a lot of this, this key period.

Yes.

But Manchester had Ferranti's, and then ICL. You know, the British computer industry all, through a series of takeovers and mergers, ended up as this one company, ICL, who had a big base in West Gorton in Manchester. But, the Manchester machines, Baby was the first operational stored-program machine, first transistor computer, then Atlas in the Sixties, probably the most impressive Manchester machine actually. If you want a real Manchester flagship, it was Atlas. It was a transistor machine, it was the world's first supercomputer, aimed to execute a million instructions a second, right? Had hardware floating-point. It was, virtual memory was invented as part of the development of Atlas, OK, virtual memory which is everywhere now, it's on your, on your mobile phone, on your computer. Manchester held the patents. The Manchester carry chain came up there. I mean, previous to Atlas, Manchester encoding was invented, which later became the basis of the Ethernet. So there was a lot of Manchester influence. And then in the Seventies, MU5 was the prototype for the ICL 2900 mainframe. And, the UK's, well, local authorities, Inland Revenue, all ran on ICL mainframes, pretty much through to the end of the twentieth century. So, so these big machines were very influential.
[1:06:12]
And, Cambridge's influence, you can feel in all sorts of other dimensions. Manchester really owns the big machine story. And it didn't end with MU5. I mean in the Eighties it was dataflow, which, which turned out to be a bit of a dead end, but was very, there was a lot of interest around the world. MIT was also in dataflow at the same time, and… And then, I came in the Nineties, and we, we went from designing big machines to chips, but now with SpiNNaker we're back in big machines again.

[1:06:45]
Two things that you've really contributed a lot from your move into Manchester was the AMULET microprocessor series.

Yes.

Can you tell us about that?

Yes. AMULET was a series of research projects, principally funded from the ECFEG programme. It originally was the open microprocessor systems initiative, which Acorn and Inmos were instrumental in encouraging the EC to set up. The AMULET processors were on compatible microprocessors, so we didn't redesign the micro, the instruction set architecture; we stayed compatible with that, so that all the software tools would work for us. But their fundamental difference from the commercial ARM processors is that they were clockless. So, generally you think of a processor and you say, you know, my PC has a two gigahertz clock. Well, the AMULET processors had no clock. They were designed on a principle where they timed themselves, they did thing as fast as they could. And all parts of the circuit interacted. There was no global synchronisation. And asynchronous technology is very interesting. In some sense, the clock, you know, the role of a clock in a chip is to slow the fast bits down so that the slow bits can keep up, right? It's… Everything has to run at the same rate. And in asynchronous design, you move away from that straightjacket. You can potentially do things much more energy-efficiently, because you only use activity where you need it, rather than having a clock buzzing away everywhere all the time. And, particularly significant is, they have very good radio interferers advantages. A clock, by synchronising everything, causes all the little current spikes to line up, add up and generate as much interference as possible by desynchronising them and actually, not locking them to a particularly frequency, you get something which is much lower interference and very broad spectrum. So there were lots of advantages. And we demonstrated this fairly convincingly. There's a, there's a global asynchronous design community, we weren't the only people by any means in this game.
[1:09:10]
By the end of the Nineties, it was clear that processors had moved from being sort of, manually designed, hard bits of design, to being synthesised so to speak. ARM started off marketing the ARM as hard macros. By the end of the Nineties they were moving entirely to synthesised cores. And, the state of tools for synthesising asynchronous circuits was much less developed than the state of tools for clock circuits. So at that point, we shifted the emphasis a bit from designing the cores by

hand to designing tools. And then it proved very hard to compete with industry, because there were huge resources going into conventional tool flows.
[1:09:57]
So, so the AMULET processors kind of occupied us nicely for the Nineties, and we did some nice work there. But nobody really managed to make a commercial success out of asynchronous design.

No they didn't, did they.

[1:1:10:10]
There have been some interesting recurrences recently. So, we're now in the neuromorphic space, and the two commercial competitors in that space are IBM with TrueNorth, and Intel with their Loihi chip. And both of those are fully asynchronous digital designs. In fact Loihi, one of the key designers was one of the people we were working with in the Nineties, he set up a spinout from CaltTech called Fulcrum that built very high-speed network switches using fully asynchronous techniques. They were bought by Intel, and now he's popped up, [laughs] close to our current space, because of the asynchronous nature of, of the neuromorphic system.

[1:10:52]
And SpiNNaker.

Yes.

From 200 onwards. Let's get lots of ARM chips [laughs] in the same room, and have them communicate, and, really flood the scene with…

What can you do with a million processes? Yes. The motivation for SpiNNaker is, is… You know, I've been designing conventional processors for several decades. They became a thousand times faster, but they still struggle to do things which we humans find easy, even when we're babies. So what's the difference between a computer and, and the brain? Which is how we do it. So I, I got interested by that question. And I got a particular grant from the EPSRC to, look at associative memories, which I've always liked in digital systems. But I wanted to make them less

brittle, softer. And every way I looked at it trying to make a sort of softer associative memory, I found I was just reinventing neural networks. So, at the end of that I threw in the towel and said, I must be interested in neural networks then. And, what can we do as computer engineers to contribute, you know, to the world progress in, in neuroscience, and in particular in computational neuroscience? And that led to this, this idea of, what kind of machine could we build that will make a difference? Well, brains are big. They're very complex. We each have 100 billion neurons inside our heads with ten to the fifteen connections. And all our memories somehow are stored in how these connections form over our lives. So it's big. So, so clearly, the machine has to be scalable. Well how do we make sure we make it scalable? Well let's set a big target. And the target was, what can we do with a million ARM processors? The real-time brain modelling in a single machine. And a million was an arbitrary number. It was clearly from the outset that with a million you, you barely get to one per cent of the scale of the human brain. Although it's, you had passed a mouse brain by then. The mouse brain is, very conveniently, quite like the human brain but a thousand times smaller.
[1:13:00]
And, and so that was the motivation, what can we do with a million processors? And, how do you do the interconnect, is the big question, because that's the question that people have wrestled with in the past when they've tried to build this kind of machine. And we came up with what I still think is the best answer available to, to the, how do you do the connectivity. And all the competing neuromorphic platforms that we see out there, even from IBM and Intel, I think are still lacking on the connectivity question. To which we found a very effective answer. Which is the heart of SpiNNaker. So if you look at a SpiNNaker machine, we put eighteen processors on a chip, 48 chips on a board, that's 864 processors on a board, and put 1200 boards in a machine, and that's a million cores. You know, that's, that's how you do it. But the innovation in SpiNNaker is, how these processors talk to each other, and that's the interconnect network.

[1:13:59]
You're still excited by this aren't you?

Oh yes. Well we, we're currently working…

I don't mean SpiNNaker. I mean just, the whole thing.

Oh. OK. Yes. The, the whole thing. I mean, I think… I think, computer technology, it doesn't just advance, it changes nature. It's a roughly every decade you get a qualitative change in, in the role of computers in the world. And, although the technology is hitting some fundamental limits, you know, Moore's Law is, is now slowing down. Gordon Moore in '65 reckoned it was good for ten years; it's lasted 50. That's not bad. But it's not going to last another 50. And, and so there's major innovations happening in technology. To continue delivering more functionality we've got to find different answers from simply making the processor go faster, because that doesn't work any more.

[1:14:58]
What's the biggest mistakes you've made in your life, your professional life?

[laughs] I wonder, I wondered where we were.. [both laugh] [pause] Well I… I don't know. I mean one answer to that is a very technical answer, which was with the 6502 second processor. I designed it initially in such a way that, that the system was prone to synchronisation failure. I say, this is a very deep technical issue which I now understand very thoroughly, partly due to that experience. And fortunately that one was found before the product went out to customers. I suppose my, my grandest mistake is, BBC Micro disk error 14. And the story there is, I, I built, when I was doing my little bit of moonlighting I built the first hard disk controller, using the Intel 8271 floppy disk controller – I didn't mean hard disk, I meant floppy disk. And I got this going at home. And I, you know, read the datasheet very carefully, and set up all the parameters, and got it to work. And took it into Acorn, and said, 'Look, I've got a floppy disk controller here.' And they took it over. And this thing found its way into Acorn products, and then into the BBC Micro. And five years later, with a million and a half BBC Micros in the field, there were strange reports of disk error 14 happening rather more than we'd liked. So it was investigated. And it turned out to be my fault.

Oh dear.

And, the reason is, because when I had read the datasheet and set up the parameters, I got one of them wrong. Nobody had checked in the five years in the one and a half million BBC Micros later, nobody had checked these numbers. So I don't feel that guilty about it. I was just knocking something up in my back room to, to prototype. But the consequence of this parameter being wrong is, I got all the… Each track on the floppy disk is divided into sectors, and between the sectors there's a gap, and I had made that gap slightly too big. Which sounds fine, except of course when you get back around to the beginning, that gap between the last sector and the first sector is then too small. And, if your floppy disk is, is… You know… There's a tolerance on the rotational speed.

Yes.

And if it's being formatted on one floppy disk drive with one speed, and then you write it on a different floppy disk drive at a different speed, and you write that last sector, and it's going slower or faster, I can't remember which, if that's, that gap is too small, you overwrite the header of the first sector, and that was disk error 14.

Suitably technical.

[laughs] Horribly technical. I'm not sure.

[1:17:55]
You have been used by Government. I don't mean used and abused. You are the specialist adviser to the House of Lords Science and Technology Committee inquiring into microprocessor technology. You are Editor-in-Chief of the BCS's Computer Journal.

Mhm.

So, you are spread wide. And I'm very interested in your appreciation of the future, because in this latest edition of the Computer Journal, of which you are the Editor-in-

Chief, it focuses on computer security issues. Seven of the twelve papers are from China.

Mhm.

Does that worry you?

We are concerned with the *Computer Journal* in general that… We're not concerned that we get a lot of submissions from China, and, also from India. What we are concerned about is that we don't get as many submissions as we'd like from the UK and from Europe. And some of this is, is, you know, the, the biblical statement that a prophet is never recognised in his own country, that, the *Computer Journal* is not held in, in as high esteem at home as it is abroad. Which is an interesting observation. But I think your question was, was perhaps pointing particularly at the computer security issue.

Not only the computer security issue, but also, isn't the future in the East rather than the West, or here? ARM has been bought by a Japanese company. The Chinese are ploughing ahead with computer security issues.

Mhm. Well the Chinese are very active. In fact if you read a different report, the, a report of the Council of Advisors on Science and Technology to the US President, they wrote a report back in January 2017, it was a different US President then, but, a very strong characteristic of that report is, is paranoia about Chinese progress in computing. And, and, there's no doubt the Chines are making spectacular progress. If you look at the top 500 supercomputers, number two, Tianhe-2 is in China. That uses American technology, and it was due an upgrade that, the Americans imposed an export ban on the upgrade components. Number one in the list uses entirely Chinese components. And it's 30 times faster and three times more efficient and so on.

And it's not being made or designed in Manchester.

No. No no, that, that, that is not, that's entirely Chinese.

Is there not an irony in that you have made a massive contribution to computer science in this country, but you are the ICL Professor of Competitor Science, and there isn't such a thing as ICL any longer.

No, ICL, as far as I'm aware, exists only in two forms. One is my chair, and the second is a pension fund. [both laugh] Of course, what was ICL is, is now owned by Fujitsu, was bought by Fujitsu, and is still operational in this country, still quite a significant business, but the name is no longer there. And… Yes, I'd always quite fancied sitting in an ARM chair, but that's… [laughs] That's an old joke. [laughs]

But a good one. Thank you very much Professor Stephen Furber, thank you.

You're welcome.

[End of Interview]