IN THE DISTRICT COURT OF OKLAHOMA COUNTY
STATE OF OKLAHOMA

Jean Bookout; Charles Schwarz,)
individually and as Personal)
Representative of the Estate of)
Barbara Schwarz, deceased;)
Richard Forrester Brandt, as)
Personal Representative of the)
Estate of Barbara Schwarz,)
deceased,)
)
Plaintiffs,)
)
vs.) Case No. CJ-2008-7969
)
Toyota Motor Corporation; Toyota)
Motor Sales, U.S.A., Inc.;)
Toyota Motor Engineering and)
Manufacturing North America,)
Inc.; Aisan Industry Co., Ltd.,)
)
Defendants.)

* * * * *

TRANSCRIPT OF MORNING TRIAL PROCEEDINGS

HAD ON THE 11TH DAY OF OCTOBER, 2013

BEFORE THE HONORABLE PATRICIA G. PARRISH,

DISTRICT JUDGE

Reported by: Karen Twyford, RPR

***** THIS TRANSCRIPT HAS NOT BEEN PROOFREAD *****

APPEARANCES

For the Plaintiffs:

Mr. Benjamin E. Baker, Jr., Attorney at Law
Mr. R. Graham Esdale, Jr., Attorney at Law
Mr. J. Cole Portis, Attorney at Law
Mr. Jere Beasley, Attorney at Law
Beasley, Allen, Crow, Methvin, Portis & Miles, P.C.
218 Commerce Street
Montgomery, Alabama 36104

Mr. Larry A. Tawwater, Attorney at Law
The Tawwater Law Firm, PLLC
14001 Quail Springs Parkway
Oklahoma City, Oklahoma 73134

For the Defendants:

Mr. J. Randolph Bibb, Jr., Attorney at Law
Mr. Ryan N. Clark, Attorney at Law
Lewis, King, Krieg & Waldrop, P.C.
424 Church Street, Suite 2500
Nashville, TN 37219

Mr. James A. Jennings, Attorney at Law
Mr. J. Derrick Teague, Attorney at Law
Jennings Cook & Teague
204 N. Robinson, Suite 1000
Oklahoma City, Oklahoma 73102

***** THIS TRANSCRIPT HAS NOT BEEN PROOFREAD *****

(Whereupon, the following trial proceedings were had in the morning on the 11th day of October, 2013, to wit:)

THE COURT: We're back on the record in Case No. CJ-2008-7969. We're outside the presence of the jury. Counsel, yesterday evening I did a little research. I'm trying to figure out this whole issue. What I am talking about now are the -- sort of the three different categories of documents that I had reserved. The ones I'm talking about now are the ones that were the e-mails with certain statements attached to it and it is Plaintiffs' Exhibit No. 717 then 730, 731 and 732.

Remind me: Yesterday, did we leave this, Mr. Baker? Did you say you were going to look at something, or was that on Fukushima?

MR. BAKER: I was going to look at all the ones that you had reserved to see if I could redact them to conform to what we kind of talked about, although you haven't made a ruling on them. Specifically, I was looking at the Prius recalls, and then cutting down where we only had in the exhibit the portions of the articles that were actually talked about in the testimony.

MR. CLARK: And that's exactly what I envisioned. I think we will probably be able to work that out; that is 718, 720 and 723.

THE COURT: Wait, I don't even have those are things I had reserved.

MR. BAKER: Those related --

MR. CLARK: Those are the ones that you gave back to Mr. Baker to work on the redactions last night.

THE COURT: And the ones I'm talking about are 717 -- all of these came in through Mr. Lentz's deposition.

MR. BAKER: I hadn't focused on those.

THE COURT: Let me tell you where I'm going with these: With the e-mails, even if they're coming in, for whatever reason, as an exception to the hearsay rule, the cases that I had found in various jurisdictions -- and I think there was one even in Oklahoma -- it didn't necessarily deal with an e-mail, but generally they deal with business records that contain hearsay statements.

And the cases are all consistent in that for the hearsay statement to come in -- and, for instance, the one that I will focus on is the one that had the letter attached to it that went through three or four different people before it got to the person at Toyota that had responded -- the only way those hearsay statements can come in is if you can show me exception at each step of the way. So, for instance, the guy that sent the long letter about his incident.

MR. CLARK: There was a lot of names in that e-mail.

MR. BAKER: That was related to the Fukushima deposition.

MR. CLARK: That is one that is related to Fukushima.

THE COURT: So unless the plaintiffs can show me some sort of additional exception -- and I don't think, because there were -- I mean that cases were all consistent that if it is a business record you cannot contain hearsay from a third party. And, generally, a third-party statement they all reference would be hearsay unless you can show me another exception to the hearsay rule.

MR. BAKER: Okay.

THE COURT: So on all of those, I would be deleting any of the hearsay statements from third parties, including that letter in that one, unless the plaintiff shows me some other exception to bring those in.

MR. CLARK: Do we need to, in view of those thoughts, look again at Mr. Fukushima's testimony? I don't know whether we do. I think the Ito (phonetic) letter is a little bit different from some of the newspaper articles and the like that were discussed with Mr. Lentz because it, as opposed to most of what is in those newspaper articles of Mr. Lentz. Perhaps all of them, it is another incident. So there is a similarity issue on top of it. For that reason, we object to even talking about it in Mr.

Fukushima.

THE COURT: And in Lentz, the reason I let the newspaper statements come in is because he was being asked if he agreed with, comment on certain statements.

MR. CLARK: That's right.

THE COURT: So you're saying that you may need to revisit Fukushima now?

MR. BAKER: I'm not.

THE COURT: You're not. I know that you aren't.

MR. BAKER: We did leave the portion related to the discussion of Mr. Ito's comments, we left that open yesterday. We didn't address that. So that part has been left open.

THE COURT: Okay. And I will --

MR. BAKER: We can do that at a break.

THE COURT: Sorry. Here is the other one that I had the exhibits on Mr. Fukushima, so this also references Plaintiffs' Exhibit 522A. So I will look at that discussion again, and we can --

MR. BAKER: It is at the end of the second day.

THE COURT: Right. So I've got that.

MR. CLARK: Eighty-two is the page. That is the first one, Mr. Baker.

THE COURT: I have page 210 where you are discussing 522 which is the Japanese version.

MR. CLARK: You're right.

THE COURT: So I will look at that. Then the other issue is then on the Fukushima exhibits 718 -- sorry, these weren't Fukushima exhibits, these were Plaintiffs' Exhibits 718, 720 and 723. And my note indicates that you were going to discuss because it had something to do with the Fukushima issue.

MR. BAKER: That's what we just discussed at the beginning about me redacting the Prius recall and portions of the article not discussed.

MR. CLARK: That is what I was going to suggest.

THE COURT: Can I admit those three exhibits subject to the redactions?

MR. CLARK: Yes. Provided we agree to the redactions, and I think we will.

THE COURT: If not, I will make the ruling on redactions.

MR. CLARK: And then reserving other objections that we haven't talked about.

THE COURT: So the court will admit Plaintiffs' Exhibit 718, 720 and 723 subject to the, as redacted, and subject to the court approving those redactions.

MR. CLARK: We have a lot of videos today, so I expect Mr. Baker and I can probably get that done by the end of the day.

THE COURT: These issues about the two congressional statements, let me ask: Mr. Clark, why do you think these are not public documents, statements, the letters, the congressional letters?

MR. CLARK: Let me grab the text.

THE COURT: These are Plaintiffs' Exhibit 716 and 722.

MR. CLARK: Yes. The thing that we can dispose of real easily is the idea that they're business records. Because if they're not admissible as government records, they're not admissible as business records; that is black letter Oklahoma law. As far as government records, there is really not any foundation that has been laid that this is regularly conducted and regularly recorded activity, or it is a matter observed pursuant to a duty imposed by law.

I think as to the second half of the public records exception, that's not true. As to the first half, regularly conducted and recorded activities, it, I suppose, might be possible to lay the foundation that would be necessary there, but I don't think we're there yet.

THE COURT: Let me say: On the public records, there are the three different categories that courts can look at to see if it is a public. The one I was focusing on is whether or not this is a regularly conducted and regularly recorded activity. I don't think it is a matter observed pursuant to a duty imposed by law in which there is a duty to report. I don't think it falls under that second category or the third one, the factual findings from an investigation. So I was focusing sort of on the first of those three.

MR. CLARK: That's where we are too. And I think basically our position is this, your Honor: A congressman or a congresswoman can write a letter that says whatever he wants whenever he wants. If that is not done under some sort of process that would assure reliability, then it doesn't meet the hearsay exception, because that is the point of the hearsay exception, right? This is something that for some reason we say is reliable even though it's hearsay.

THE COURT: Let me say: I don't agree with you that this is just a letter that a congressman wrote. Both of them specifically reference his testimony before the committee, so this is not just a congressman sending a letter on an opinion.

MR. BAKER: That is right. It is our position it is related to an activity that they're supposed to conduct, and it is in relation to his position as chairman of the subcommittee on oversight investigations, and specifically references investigations they're conducting, has been the testimony.

THE COURT: Let me ask: Maybe when our -- I'm wondering is, I don't know that the foundation has been laid at this point that this is the type of document that is a regularly conducted and a regularly recorded activity. I don't know, for instance, could I do an open records request and get these records from the committee on energy and commerce?

MR. BAKER: I don't know the answer. I do know the testimony that we have put it on through Mr. Lentz is they were conducted hearings, and that this was in association with that; that is what congress does.

THE COURT: Let me go back and look at what Mr. Lentz said about that to see if there has been a foundation laid at this point in time then.

MR. CLARK: On that point, I might note that I'm not sure that Mr. Lentz can lay a foundation for what is the regularly conducted, regularly recorded activities of this committee. He's not a member of congress.

THE COURT: I will tell you: The cases that I read, unless the business records where you have to have someone come in from the business and say it is regularly conducted, dah, dah, dah, I don't think that's necessary that someone from congress come in and tell me this is regularly recorded.

MR. BAKER: I have one case, and I don't have it here. I will bring it for your Honor. But as I recall, it stated that was the very purpose of the government exception so you don't have to pull people out of their government jobs come in and tell you; that's exactly what they were doing.

MR. TAWWATER: I want to add one other thing to that. The cases that I looked at all seem to discuss the reliability of the document. And in this case, it's clearly from the committee, clearly signed by the co-chairs. Mr. Lentz testified and said, Yes, this is something that I got from these people in congress. So I think the reliability issue is very well satisfied.

THE COURT: Wasn't this all in reference to Mr. Lentz's testimony and then comments that he made on the Today show or CNN or someplace?

MR. CLARK: One of the letters.

MR. TAWWATER: And his congressional testimony.

MR. CLARK: One of them was specifically in reference to that, and I can't recall, as I stand here, what the other one was. One was with regard to his TV appearances.

THE COURT: Both of these, if I remember, were signed as chair and co-chair of the committee, correct, they weren't just signed as a congressman?

MR. BAKER: Bart Stupak as chairman, and Bart

Stupak, chairman, Henry Waxman as chairman.

THE COURT: Okay. Assuming that they meet the regularly conducted and regularly recorded activity exception, I will go back and see what Lentz says and followup to see what -- how far it has to -- how far you have -- what your burden is to show that. So I'm reserving these as well as the -- I'm trying to remember. Off the record.

(Whereupon, an off-the-record discussion was had.)

THE COURT: On all the videos, we got the Japanese out of at this time now?

MR. CLARK: Yes. There are a few places. And, actually, I talked to both Ms. Allen and Mr. Doyle about it this morning. There are a few places where they are folks talking over other, or there is just so little Japanese that it can't be taken out. But it sounds like we are on the same page on that now.

THE COURT: It will not be like yesterday.

(Whereupon, the jury returns to the courtroom.)

THE COURT: We're on the record in Case No. CJ-2008-7969. Members of the jury are present as well as counsel and their clients. And remind me, Mr. Baker, were we going to start back up with Mr. Fukushima?

MR. BAKER: We will start back with Mr. Ishii, take two.

THE COURT: Tell me again this witness's full name.

MR. BAKER: First name S-A-T-O-S-H-I, Satoshi. Last name, Ishii, I-S-H-I-I.

THE COURT: Okay. Again, this is a deposition where both plaintiff and defendant have designated the testimony from this gentleman?

MR. BAKER: Yes, ma'am. And I believe, with small exceptions, all of the Japanese has been taken out.

THE COURT: All right. You may proceed.

MR. CLARK: Subject to our prior objections.

THE COURT: Exactly.

(Whereupon, the video deposition of Satoshi Ishii was played to the jury. Not on the record.)

THE COURT: Ladies and gentlemen of the jury, we're going to take our morning break at this point. It is 10:15. We're in recess for 15 minutes. I would remind you: During the recess, do not discuss the case, and do not begin to form any opinions about the case.

All rise while the jury exits.

(Whereupon, the jury exits the courtroom.)

THE COURT: Counsel, are there any exhibits that is we can quickly admit into evidence?

MR. BAKER: I don't have them pulled up.

THE COURT: We can do that at lunch.

(Whereupon, a short recess was had.)

THE COURT: We're on the record in Case No. CJ-2008-7969. Members of the jury are present as well as counsel and their clients.

Mr. Portis, you can call plaintiffs' next witness.

MR. PORTIS: Thank you, your Honor. We call Dr. Philip Koopman.

THE COURT: Raise your right hand, please.

(Witness sworn.)

PHILIP KOOPMAN,

called as a witness, after having been first duly sworn, testified as follows:

DIRECT EXAMINATION

BY MR. PORTIS:

Q Dr. Koopman, tell the jury your name, please, sir.

A I'm Philip Koopman.

Q And it looks like a picture of you in a bow tie. And I'm -- one, because I know and, two, because it looks like it on the picture, I will guess that you are a college professor?

A Yes. I'm a professor at Carnegie Mellon University.

Q Tell us a little bit about Carnegie Mellon that.

A That is one of the top five computer engineering programs in the United States, so we are well known for computers. I teach in the electrical and computer engineering department. My specialty is embedded systems and, in particular, safety critical embedded systems. And I do a lot of work on cars, but also railway, airplanes, things of that nature.

Q When you talk about -- I brought this book that you wrote. It is called *Better Embedded System Software*; is that right?

A Yes.

Q And you wrote this book; is that right?

A Yes, I did.

Q I guess the question that we need to understand is what is embedded system software?

A Embedded system is when you have a computer and it is inside some other product. So when you buy something, if you go down to Best Buy and it says DVD player or it says TV set instead of saying computer, there is still a computer in it, but that is an embedded system. And the software is the set of instructions inside it that makes it do what it does.

So maybe there a software ap that takes Netflix and decodes it into -- I watch Netflix too -- and decodes it and shows in on your TV. Well, there is software taking those bits from the Internet and turning them into a picture on your screen. So that would be one instance of embedded software.

Q Well, your book is, obviously, I understand now, embedded system software, it is entitled *Better Embedded System Software*, and it looks like the copyright was copyrighted in 2010; is that right?

A That's correct.

Q And why did you feel the need to write this book?

A I've done a lot of the design reviews; right now about 135 of them. So for most of these, I get on a plane, I go someplace, and I visit people who have written embedded software for real products: compressors, thermostats, petrochemical processing plant equipment. You name it, I've probably seen it for those kind of pieces of equipment.

What I did is I just wrote down all the mistakes they might make, and most teams make one or two mistakes. And I collected them up, and the back of the book has a list of the chapters, and the chapter are just this team made this mistake and here is how you can get it right.

Q Just so I understand, not only do you teach there at Carnegie Mellon, but in addition to that you also do consulting work for other groups; is that right?

A That's right. So these design reviews were all for industry products, some of them you probably have in your house.

Q Now, as part of that, before you became an expert in embedded system software, do you have any expertise in hardware as well?

A Yes. Before I started doing software I was a CPU design for Harris Semiconductor. So I actually laid out the gates on chips, and I've had my own CPUs as a way for the semiconductor for my office. So I built my only CPU and did all the design work on it, so I know both software and hardware.

Q In terms of your background and experience where you came to the knowledge of hardware and software, tell the jury a little bit about your background, educationally and professionally.

A So my undergraduate and master's degree were at Rensselaer Polytechnic where I studied to be a computer engineer. I spent some time driving fast-attack submarines in the Cold War for the U.S. Navy.

Q Driving what?

A Fast-attack submarines. Think *Hunt for Red October*.

Q How did you get involved in the Navy?

A I went through ROTC on a scholarship; that's how I paid back for my college education.

Q So they put you on a submarine?

A They put me on a submarine. I was in charge of all the computer systems, at one point, on my submarine. When I was done with that, I went to a short command where I was helping to put together, build new computers for new submarines. After that mI got at PhD in both hardware and software but computer engineering.

I worked for Harris Semiconductor doing CPU designs, so designing the hardware that goes in the computers, and so chips with gates and wires and all the things on a chip, in a computer chip. I then worked -- went to United Technologies where I worked in our central research center. They own Pratt & Whitney jet engines, they own Carrier air conditioners, Norton sonars, an automotive division, UT automotive. So I got a lot of exposure to all sorts of things there.

Then I went to Carnegie Mellon University. I've done wearable computers, I've done software robustness testing, and I have done a lot of work on embedded system safety.

Q How did the opportunity present itself to go to Carnegie Mellon and the academic world?

A I decided I wanted to about 50 percent applied and 50 percent research, and I enjoy teaching. And I had some contacts there, and the invited me to come work there.

Q Now, as part of -- tell us a little bit, what do you teach?

A I teach three courses. One is for undergraduates, an introduction to embedded system, things like how A/D converters work, which I will get to in a moment. So I teach all of that. Then I teach a first-year graduate level course for master students. And that book is the textbook for that course. It is used in several universities, including ours.

There I concentrate on how to write good software and make sure that things really work. Not almost work, but really work. Then I teach a PhD course which goes through all the theory papers, some of which I cite in my slides about fault tolerance, dependability, safety.

Q I want to go through just a few things on here. I know we talked about computer hardware and your work at Harris Semiconductor and your teaching at Carnegie Mellon. Says you are an expert in computer software, and underneath that you talk about design production, automotive remote keyless entry software.

I think I know what that is, but why don't you talk about that.

A When you take out your car keys and you press the button and it unlocks the car, on the modern ones that is encrypted so no one can eavedrop and play it back to unlock your car when you're not there. And I designed one of the two big algorithms that was in use starting in about 1994, so General Motors and several other companies use that. So that was a production piece of automotive equipment.

I designed that, and I also designed the manufacturing equipment to program them with secret numbers that no one can guess.

Q When we talk about computer safety systems, an expert in computer safety system, what is a computer safety system, and why is it needed?

A When you have a computer that is just sitting on your desktop, it can't do a lot of harm to you. When you give it motors, and you give it the ability to release energy into the environment, that's how safety people think about it. you have the ability to move a piece of equipment, like a robot arm, or drive a vehicle down the road, you have to make sure it's not going the hurt someone.

So computer system safety is going in and making sure that not only does it do what it's supposed to do, but it doesn't do anything dangerous, even though some fault might happen to it. So the research that I do for that is on self-driving vehicles. It is mostly sponsored but the U.S. Department of Defense, but I have industry sponsors as well. We go in and make sure things like self-driving cars are going to be safe and not run people over.

Q Are we about to have self-driving cars?

A I've had a ride in the Google car. I can't say more, but they're coming.

Q All right. And then we talked about your Navy submarine experience and working on computer systems there. You mentioned that you have patents?

A Twenty-six patents from my time in industry, several of them are automotive.

Q And then your bedded industry design reviews; is that primarily your outside work beyond your work there at Carnegie Mellon?

A Right. This is all technical consulting work. I do several reviews a year. As I said, I get on a plane and I find out how people are doing and tell them how to do better if they need it.

Q What industries do you work with?

A So it is -- well, a partial list is there are automotive, trains, chemical processing plants, heating ventilation and cooling, power supplies for computer machine rooms. It just -- the list goes on. Hard to -- it is a big, long list of companies, but that gives the idea. It is embedded systems, it is things where there is a computer hiding inside it, but that's not what you bought it for.

Q In all of those areas, do you deal with computer system safety?

A I would say an increasing number of my reviews lately have been safety. I was doing safety reviews for automotive as early as 2002. Some of them are safety, some aren't. But honestly, if you were making a million of something, you have to get it right even if it's not safe. So the techniques aren't that different, it's pretty much the same stuff.

Q Now, when did you get involved or enthralled in the Toyota litigation.

A I guess it was about last summer. So around I think May or June of last year.

Q So somewhere of 2012 was your involvement. And I know that your book was copyrighted in 2010.

A It was actually written in 2009. It took a while to get it out.

Q Okay. And so in terms of your opinions about better embedded system software, you held those opinions prior to even your involvement in the Toyota litigation?

A Oh, absolutely.

Q Now, what were you asked to do in this case?

A In this case, I was asked to take a look and see whether or not the Toyota ETCS was safe.

Q Does your background help you make those types of determinations?

A Yes. Definitely. I have been working on doing reviews of systems for safety and teaching safety for years.

Q What types of -- what did you do in order to make -- before you gave your opinions in these cases, what did you do? What information did you look at before you offered any opinions?

A I looked at all the information I could get access to; that included the NASA report, which had quite a lot of detail in it. I looked at Toyota highly confidential design documents. I looked at depositions of Toyota and Denso employees. And I looked at the expert reports of Mr. Barr and others who had access to the source code.

Q I want to follow up and define just a couple of things there. The first thing I would like to define is we heard from Mr. Ishii and we heard sort of in the course of the trial about this NASA report. Before we get specific on it later, can you give us some general background on the history of that?

A Sure. I wasn't personally involved, so I'm going by what was written in the report. But what NASA was asked to take a look and see if they could find a fairly narrow source of unintended acceleration. It was fairly narrowly defined. They were given access to some of the materials that were necessary. And they, in particular, on the main CPU. And they went through and the looked through the software, and they looked at the hardware. And they had some things to say that I will be talking about in more detail.

Q And you said they were given some of the information. And I know you were here for Mr. Ishii's testimony just a few minutes ago. Were they provided all the information?

A My understanding is they were not. As Mr. Ishii said, and in looking at the NASA report, I do not think they had access to the software for the monitor CPU, the ESP-B2.

Q The second term that I want us to talk about is source code. What is that?

A Source code is a human readable version of the instructions that go into the computer. So computers are pretty dumb. They do exactly what you tell them; that is a good thing and it's a bad thing. They only do what you tell them. So a source code is a list of instructions, take this number, add one to it, store it someplace. Take this other number, add it to a fourth number, store it someplace else. When you are done, go over here and do some other things.

So the source code specifies that list of instructions, just like if you have a recipe and it says take so much of this and take so much of that. It is a recipe of how to do the computations that the computer needs to do.

Q Just to make sure I understand, the source code itself is provided by human beings; is that right?

A That's right. Human beings write the source code.

Q So the source code itself is only as good as the human being's knowledge in terms of what they're embedding in that source code?

MR. BIBB: Objection. Leading.

THE COURT: Sustained. It was leading. You need to restate it.

Q (By Mr. Portis) Tell us a little bit, then, about the interactions between source code and the human interaction.

A So what happens is sometimes source code is already existing, so it uses some libraries. But at some point, eventually some person had to write this source code down. They had to write the recipe. And when you initially write the recipe, the person writes it, and there are probably some bugs in it because nobody is perfect.

Then you go through a process to make sure there are no bugs there, and we will get into that in more detail as well. I should explain, when I say "bug," I mean a defect. So when a recipe says put 50 cups of flour in, you know, that's probably not right unless you're in an industrial kitchen.

Q Is that the reason why standards are important for those who write those software codes?

A One of the ways that you reduce the number of bugs is by using a standard practice for -- in this case, we're talking about standards for source code, style and source code formating and language use. So there may be things where you say, Okay, instead of using the number 50 or 5, we will spell out. And so in Naval communications they do this, they don't use numbers, they spell them out, because then it is harder to mistake a five for a six and things like that. So you will have style guidelines and language-use guidelines that make it hard to make a mistake, because some of the factors of these languages are really easy to make a mistake, and I have a slide on that.

Q Now, the way that I would like to do this is I want to start off by giving your overall general opinions, and then come back and talk about those general overall opinions. Have you offered opinions in this case?

A Yes, I have.

Q All right. Now, I did a couple of things. And I know they're on your PowerPoint presentation, but I will also have them on a hard board because we may have to refer back and forth to them. So tell us what you say your first opinion in this case is.

A My first main opinion is that Toyota electronic throttle control system, ETCS, design is defective and dangerous.

Q When we're talking about the electronic throttle control system, describe what that is.

A I think we have pictures coming up. But at a really high level, there is a computer that runs the engine. So when you press your foot on the accelerator pedal, what is happening is you're not actually moving any mechanical parts inside the engine. What you're doing is you're sending this computer a signal saying, I want the accelerator pedal to be down, or I want it to be up. So the computer software and hardware runs a program that converts that into a command to where the throttle goes, and the throttle controls air flow that tells your engine how fast to go.

Q From an overall perspective, you have three subpoints. What is the purpose of those?

A Those are supporting reasons why I believe this. The first one is that random hardware and software faults are a fact of life. Random has a special meaning that I will get to, but it means even if you think it is designed perfectly, something always goes wrong anyway.

The defective safety architecture has an obvious single point of failure. A single point of failure is a critical concept in safety critical systems. I will explain an example of where one is and why that is important.

And reading the NASA report, they came to the same conclusion.

Q What is your second opinion overall?

A The second overall opinion is that Toyota's methods to ensure safety were themselves defective. You have to exercise great care when you're doing safety critical software. You can't just wing it. And Toyota exercised some care, but they did not reach the level of accepted practice in how you need to design safety critical systems.

Q And you mentioned, and I know we will talk about this more in a little bit, and we heard a little bit about it from Mr. Ishii, who was played before you. You mentioned something caused MISRA?

A Right. There are two MISRAs, and that can be confusing. There is the thick one and the thin one. Here I'm talking about the thick one.

Q When we are talking about thick?

A That's the thick one.

Q Exhibit 5649, this is MISRA, which stands for what?

A Motor Industry Software Reliability Association.

Q And Exhibit 5649, the MISRA standards. These are standards that automotive manufacturers follow?

A Those are a set of automotive specific safety guidelines that some manufacturers decided to follow. As I explain, there is a bunch of standards to choose from; that is one particularly relevant to automotive.

Q Then in Mr. Ishii's testimony, he mentioned something called MISRA-C. What is the distinction between the two?

A So this is a recipe book for how to build safe cars.

Q This one?

A That one. Right. MISRA-C is a much thinner document, and it is just concerned with how to use the C programming language in a safe way. And so part of the big MISRA thing says that you have to use the programming language in a safe way, and one of the ways to do it is to follow this document.

Q This is Exhibit 3106, which is the MISRA-C?

A Right. And Mr. Ishii was mostly talking about that document, I believe.

Q MISRA-C?

A Yes.

Q All right. Then your second point was design and engineering process, had inadequate rigor and quality. What do you generally mean?

A I mean that if you're designing something that can kill people if it malfunctions, you have to be very careful. In classes, I say you can't be a cowboy, you can't be a cowboy coder, you have to be a methodical, rigorous engineer and pay attention to details; that's what I mean by that.

Q And Toyota was inadequate in their rigor and quality?

A Yes. That is my opinion.

Q Third opinion?

A Third opinion is that the Toyota safety culture is defective. So safety culture is how the organization as a whole treats safety: Do they take it seriously, do they have processes in place to make sure that even if you're having a bad day you will not make a mistake that day, that still things are going to work okay.

And I saw several signs of a defective safety culture. And one example that I will talk about is that when they're investigating an accident, they don't seem to take the possibility that the software can be defective very seriously, they say just say, No, you know, that can't be defective. And I have precise information about that.

Q Let me ask you this: When you're hired in your consulting business to go and travel, do you go through some of this analysis with those companies?

A Sure. Depends on the product, but I spend a lot of time looking. When it's a safety critical thing, I go through these kind of things. I say, Gee, is your safety culture good? Is your process good? Have you followed a good recipe? Have you followed one of the standards for your system safety?

Q And correct me if I'm wrong, but that is to -- when you do that, is that to assist the company to develop good, healthy software that would protect people in some instances?

A It depends on the engagement, but there are several engagements that I've been on where the soul purpose was to make sure that they had a good safety culture and all their processes were good. Yes.

Q And you're telling the company about it?

A I am telling the company. I am an independent person to come in. When you are doing safety, part of a good safety culture is you always have blind spots. So you bring in an outsider to make sure you are getting everything right.

Q In terms of going through a analysis and presenting it to a jury like we have today, is this your first time in trial?

A This is my first time in trial.

Q Now let's go through, you have a fourth opinion; is that right?

A Yes.

Q What is that?

A The fourth opinion is that Toyota should have gone far beyond just vehicle testing. You heard Mr. Ishii talk about that ultimately they test the vehicle. Well, that's a good way to get things mostly right for everyday occurrences; that is completely insufficient to guarantee safety when you have a large fleet of vehicles. And I will go into specifics about that.

Q So when Mr. Ishii talks about some testing that they did, are you saying that is good and profitable, or are you saying that is not enough?

A No. It's good, but not enough.

Q Okay.

A By far not enough.

Q What else do you say here?

A So fault injection is an accepted way to measure fault responses. The big idea there is that if your system is designed to be safe even if something goes wrong, and you never test something going wrong, you don't know if it works.

The next one is that even if you know exactly a problem could happen, if you have a whole vehicle, you may not be able to reproduce that, because it requires changing something or introducing a fault that there is just no way to do except of waiting a really long time for it to happen by itself.

And the last one is that you -- because of these, you have to follow accepted practices. You can't just test a vehicle and know it is safe. You have to do a bunch of other things, the rigorous engineering that I was talking about. So it is both the testing and following a rigorous process.

Q Your next opinion?

A My next opinion is Toyota's source code is of poor quality. And as you know, I haven't seen the source code myself. But what I've done is looked at what NASA said about the source code, I've looked at what Mr. Barr and his associates have said about the source code. Even at a high level, there is some tell-tale signs that you don't need to look at the individual lines of code to know there are some severe problems here.

One of them is 10,000 global variables. If you talk to a safety person, and that number is above 100. Even if it is 100, they will right there say, You know, that's it. There is no way this can be safe.

Q Isn't the actual academic standard there should be zero global variables?

A That academic standard is there should be zero. In fact, I have a chapter in my book called Global Variables Are Evil, and that was written in 2009.

Q And Toyota's system has 10,000 global variables?

A About 10,000. The number depends how you count. We will get to that, but that is the ballpark. Yes.

There is also -- they have poor quality. And Mr. Ishii talked about finding defects with static analysis. And I will explain what that is and show you the numbers. But they have far, far too many bugs. There is academic literature besides the bug chart that we are going to talk about that demonstrates when you have that many warnings there is going to be bugs.

Q I think we saw a little bit in Mr. Ishii's testimony about the bug chart itself.

A Right. And I have some slides. We will be talking about that.

Q Very good.

A And the last one is that you can use analysis tools, you can do design reviews. So all the things that NASA and Mr. Barr and his associates have done are -- that's how people assess code quality. They don't just say, We will take some smart guys and take a look, they also use some tools. Nobody is good enough to find everything, so you use tools to help you find things.

Q You mentioned Mr. Barr. They don't know him. Who is he?

A Mr. Barr is a very well-known embedded system expert who will be testifying in this case. He and his team have had access to the source code and have spent I guess a couple of calendar years at this point looking at it and analyzing it.

Q And he is here today?

A He is here today. Yes.

Q What is your next opinion?

A Toyota's approach to concurrency and timing is defective.

Q What does that mean?

A That means in a car when you're driving a car and the engine is spinning around and the spark is firing to ignite the fuel, it has to happen in a very precise time line. You can't say, When is the computation going to be done? Oh, next Tuesday. It has to happen in a very defined time.

And in a safety critical system, you have to meet deadlines. So they have you have so many tenths or so many hundredths of a second to do it, and it has to be done by that time. If you miss those deadlines, the system is generally considered unsafe.

Q And I don't want us to miss this. Safety critical system. What are you referring to?

A Safety critical system is one in which if there is a defect in the software or a defect in the hardware someone can get hurt or someone can die.

Q Then you have one more page.

A The last main opinion is that the Toyota ETCS is unsafe and unsuitable for use in a safety critical system. In addition to all the things that I talked about, there is a dangerous focus on recovery from UA rather than preventing it in the first place. And it is my opinion that the ETCS, because of its design, can reasonably be expected to produce unintended acceleration.

Q Okay. I'm not sure I understand. There is a focus on UA recovery. What do you mean by recovery?

A What I mean is that a lot of the failsafes are designed so that unintended acceleration happens and then sometime later the failsafes kick in. But in the meantime, it's displaying dangerous behavior.

Q Now, does this system on the Toyota Camry, does it have failsafes in it?

A It has some failsafes; that's what Toyota calls them.

Q Are they adequate?

A They're not adequate.

Q And then what will is that last section there?

A If you have a system like this with single points of failure and poor quality software, it is going to be unsafe. And unsafe is a manifestation of whatever behavior is going to cause a problem. In this case, UA is an unsafe behavior for a throttle control system. So, in other words, bad things are going to happen eventually because that's the way computers are. This system does not adequately protect against them.

Q Let's look at the -- let's get some education done. Let's look a little bit at the electronic throttle control system, and let's try to understand what that is. If you would tell us a little bit about the electronic throttle control system.

A Okay. An electronic throttle control system is a computer that when you put your foot on the accelerator -- I may call it the gas pedal, but the accelerator pedal is the correct term -- it sends an electronic signal up to the engine. So instead of a cable being pulled to open and close something, it is just an electrical voltage.

Then there is a computer that Toyota ETCS-i, the electronic throttle control system -- the "dash i" is intelligent, I usually leave that off when I talk about it -- but that is the full name, an engine control module, it is a piece of software and hardware. It actually has several pieces inside it, we will see on the next slide.

Its job is to look at the accelerator pedal position, also things like whether the air conditioner is on and other loads on the engine and makes sure that it opens and closes the throttle. So in a car engine, the throttle is a valve. So this thing rotates to open and close and air comes up and down here. And so when it is closed there is not a lot of air. And when it is open there is a lot of air, when -- the amount of air is what you use to control how much engine power you have.

It also injects the fuel and does the spark, but the air control, the throttle is what controls engine power, and the fuel injections and spark just sort of keep up with however much air is going through. This is historical in old cars there was a mechanical cable that went from this pedal right to the throttle. First car that I drove just had a mechanical cable, but now there is a computer involved and that can improve fuel economy and help improve emissions, so it gives you better performance.

Q And I guess the ECM itself is a computer, right?

A Right. The ECM has multiple computers inside it. It is an electronic circuit board. If you open up a computer and you see a green circuit board, that's what we're talking about.

Q And you don't have an opinion that computers are wrong to control from th accelerator to the throttle, do you?

A There is nothing wrong with using a safely designed computer to do this.

Q Is that the key?

A That is the key. The key is I don't think this one is safely designed.

Q All right. What is your next point?

A A really important point is that you can do whatever you want with this gas pedal. If the software in here messes up, you're going to get possibly a fully opened throttle. The software and hardware combination can do whatever it wants to that throttle.

Q Let me ask this: Again, I heard Mr. Ishii talk about software and hardware. He was more of a software guy rather than a hardware guy. The ECM up here, is that software?

A I think the next slide sort of addresses this. So the ECM has a bunch of circuits, but the ones we really care about is there is two integrated circuit chips. The computer chips, the black things with all the silver legs on them, there is two of those on the board that matter. And one of them is called the monitor ASIC. ASIC is application specific IC, which means they custom design this chip. And the other one is the main CPU.

THE COURT: Sir, can you slow down just a little bit.

THE WITNESS: I apologize. I'm in my grad student lecturing speed. Sorry.

So this is the monitor ASIC, application specific integrated circuit, and it has two parts. They are really on the same chip. This dotted line is just for illustration. It is all one chip. But it has a CPU. CPU, central processing unit. It is like an Intel pentium or something like that; that is a CPU, so it is a computer chip.

It also has another section that does input processing. So when you press on the accelerator pedal, it sends a pair of two different signals up. That gets converted from an electrical voltage into ones and zeros, bits, which computers only know how to do bits, ones and zeroes.

Q (By Mr. Portis) Let me ask you this, I'm trying to understand it: You showed us the ECU, purple on the previous slide. How does this relate to that?

A This is part of what is inside that.

Q What inside the purple part?

A Inside this purple part, there is a couple of computer chips that implement these functions, but these are not on separate chips, they're all smushed across these two chips.

Q Okay. And --

A So I get to the software part. So this is a CPU. It is like a pentium, okay? It is a much smaller, much less expensive chip, which is appropriate; that's fine. So the hardware are transistors and wires, hundreds of thousands of little transistors and little wires that put together to make a computer.

But that piece of hardware itself doesn't know what to do, there is no recipe. So the source code gets converted into ones and zeros the machine knows how to use to execute the recipe, so that is the software, it is the program image that comes from source code down to binary ones and zeros.

So this CPU, this is the ESPN-2 in this vehicle. It is part of the CPU and also this input conversion. There is some Other things on it as well, but for our purposes, this is the important part. So it has some software and hardware. There is the main CPU that also has hardware. It is a different one, it is a V850 renaissance. It used to be NEC at the time, I believe, and it also has some software. And the software here is primarily responsible for computing the throttle command in our discussion today. And there is some failsafes, there are some other functions that are done on both of these CPUs.

Q So there is a -- and CPU is what?

A The sub CPU I will call the monitor CPU just to keep terminology straight, and the main CPU. So two different CPUs, two different computer chips.

Q Is that a good practice?

A Having two different computers is good practice.

There is some aspects to this that are not good practice that I will talk about.

Q When we talk about -- I see this word to the left, it says accelerator pedal then you have a line up to VPA1 and a line to VPA2. What are those?

A So the physical accelerator pedal has two different sensors for position, and it sends two different voltage signals up here in case one breaks the other one will have a value. So partly that is in case on breaks. More importantly, from a safety point of view, if they don't agree with each other you know something is wrong and you can take action. And some of the failsafes have to do with that.

Q Go to the next slide.

A I will use a definition of unintended acceleration, which is any vehicle acceleration unintended by the driver.

Q And you take that from the NASA report?

A That's right out of the NASA, so I will not split hairs about whether it is speeding up or keeping constant. If the driver releases his foot from the gas pedal, he will expect the engine to slow down. If he puts his foot down, he will expect to speed up. If he keeps it constant, he expects the speed to be relatively constant.

So ETCS-caused UA occurs when the driver loses ability in command throttle position because of a hardware or software fault. In other words, for me UA is when the driver intends a certain thing to happen based on the position of the foot on the accelerator pedal, and that's not what is happening.

Q Is that because bugs are introduced into the systems that are not -- I don't know a better word than this, gotten rid of?

A One possible cause for this is software defects. Another possible cause is hardware faults.

Q Okay. Now, is it vital that you have safe softwares in an automobile that has a computer that is controlling the accelerator to the throttle?

A It's absolutely crucial that your software be of a very high quality and very safe.

Q Why do you say that?

A I say that because unlike in an old car where there was a mechanical wire. The computer has complete control of what is going on with your engine speed. It can do anything it wants with the throttle, so you have to make sure the software gets it right. And you have to make sure that even though faults will occur, faults are going to happen, that it still gets it right despite any fault that is going to happen to it.

Q Now, up here you mention that safe systems -- and that would include this ECM, right?

A The term of art is a safety critical safety.

Q All right. This safe system requires a rigorous approach to design. Then you quote MISRA, which I think we have shown the jury?

A MISRA software is the thick one.

Q And it's says that the higher levels of integrity require more information and more rigorous application of software engineering techniques. Do you agree with that?

MR. BIBB: Objection. Leading.

THE COURT: Overruled. Be care with your leading.

THE WITNESS: I absolutely agree with that.

Q (By Mr. Portis) Can safety -- in the safety systems, can it be an afterthought?

A It cannot be an afterthought. The only way to create a safe system is to start from day one saying we will create a safety critical system. Here is the set of procedures that we will follow, and every step we will follow every step rigorously. If you have a piece of software -- and I have been in this position, I've had companies say, We have this software, can we make it MISRA cell 3? And the answer is only if you start over from scratch. You can't go back and build it in.

Q Did Toyota start over from scratch for the software system built in the Camry in 2005?

A So my understanding, based mostly on reports from

Mr. Barr, is that they over time built up their software. I would defer to him to give more specifics about that.

Q Fair enough. What is this quote that you have here?

A This quote is -- Nancy Leveson wrote a paper about the Therac 25. This is a radiation therapy machine that unfortunately killed some people due to very bad software. And I included the quote because it was really striking some of the things in that article really resinated when I read about all the things that are going on in this case.

But the particular quote I have is that fixing each individual software flaw as it was found didn't solve it. So what happened was they would have an accident and someone would be injured or die and they would say, Okay, we found the but and we fixed it, and then someone else would be injured and die. And they would say, We found the bug and we fixed it.

And the lesson from that, and this is just a case study that documents that really this is what happens, is if you take the point of view I have some software and I am going to debug it by testing and getting rid of bugs and testing and getting rid of bugs, you will never get safe software. You have to do something mone because there is always another bug hiding there. It is not possible to test and find all the bugs.

Q We have talked about source code. We talked about engineering source code. And then at the end here you talk about safety. Can you describe that for us.

A So safety is having some assurance that the result, resultant hardware and software is not going to cause a mishap, so an accident. And to do that, you have to make sure. You don't just look at the source code and say, This source code is safe. If you give me source code and ask me is this source code safe, I am going to say, I need to see the whole engineering process.

Because if I find a bug in a source code, we're sort of done, I know it is not safe. But if I can't find a bug, I still don't know whether the software was developed rigorously or not because no one is smart enough to find all the bugs; that's why you put these processes in place to make sure you have checks, you have balances, and you have tools.

Q How do you determine whether the software was rigorously developed?

A And so I looked for written evidence of following a rigorous process, and there wasn't a lot of that for this code.

Q For Toyota?

A For Toyota. I didn't see a lot of written evidence. And the safety guidelines and standards all say that if you can't go in externally and know that they followed all the steps, then you basically assume they didn't have them. If I get asked to look for safety, I say, Show me the piece of paper that proves you did peer reviews. We don't have the paper. From a safety point of view, it didn't happen. When I do safety reviews, that's how I do it.

Q Now, memory corruption is expected during Toyota ETCS operation. What do you mean by that?

A What I mean is that there is a two types of memory; there is program memory that stores the recipe, but there is also working memory, RAM, R-A-M. In your PC you have RAM that you load Windows into and programs into. But in embedded systems, RAM is just used for the most part to hold working data.

So if you think of a spreadsheet and all the cells in a spreadsheet have numbers in them, so each location and RAM called a variable corresponds to one cell in a spreadsheet so it can hold the number or something like that. What you expect in an embedded system like this is that the spreadsheet cells, individual ones of them, will get corrupted once in a while. It will happen due to hardware problems, it will happen due to software faults, so that is what I will talk about in this section.

Q How does corruption occur?

A One way corruption occurs is by hardware faults. And this sound pretty exotic, but it exactly happens all the time. There are cosmic rays coming from space. They interact with particles in the atmosphere. I know how this sounds, but it happens. And eventually they shoot energized charged particles down into chips and they cause a gate to flip.

Here is a computer chip, and inside it the charged particle hits just in the wrong place. It will change a one to a zero or a zero to a one in that working memory. Here is some data from Chris Constantinescu who worked at Intel at the time saying he looked at some servers over 16 months and found a handful of them that had this happen more than a thousand times in 16 months.

So there is data showing this happens all the time. It has been happening for years. Will it happen on every car every day? No. But on your laptop, it will happen, like, once a year. If you have a million laptops, that is a million times a year. So it happens often enough that on a safety critical system you have to design to mitigate this.

Q Well, and I guess that is my point. In relation to knowing that random hardware faults corrupt memory, how does that relate to Toyota and the rigorousness of their design?

A So in -- even if you have perfect software -- I don't believe that is the case here -- even if you had perfect software, you will still expect these kind of effects to disrupt the software just like it had a bug. And it will give you a wrong answer. It will change a plus to a minus. It will change a throttle angle.

It will change something and the system is going to work incorrectly unless you do something to say, You know, this is going to happen once in a while, and even if it happens, we're still going to guarantee safety.

Q Is that fair to Toyota to guarantee safety knowing that random bits can occur?

A It is absolutely required of a system og this type. It is standard practice to have more than one computer for the purpose or memory error protection. But generally more than one computer specifically for the purpose of counteracting this. On rail systems, on aviation systems on chemical process plant systems, they all use multiple CPUs because they are worried about this, even if they think their software is perfect.

Q So how does a software engineer -- how does it guarantee complete safety?

A We will go into that in a bit, but what you do is you have two copies. If one gets messed up, and the other isn't, you notice they are not the same and you do a safety shutdown.

Q What other issues do we have in this area?

A What does this says? Even Mariani, in a paper from 2003, said these are called soft errors. It is kind of weird because it is a hardware fault but they call them soft errors. Because when you turn the power off and turn the power on, it's gone. It's just -- it messed up a spreadsheet, but when you reload the spreadsheet, it is back to normal. They call them soft errors for that reason.

When you are building drive-by-wire, and this is a throttle-by-wire car -- by-wire means I'm using a computer to tell the throttle where it is -- you have to take these into account. And all the safety standards say this.

Q Not only are there hardware faults, are there also software faults?

A There are also software faults. On a lot of these slides, I will not crawl through the details, but I want you to know that I did the academic research and this is all backed by solid academic research and literature. Software corruption, so this is a software bug that messes up the memory.

So you have a spreadsheet with a formula that puts its answer in the wrong place, or does something weird and messes it up at run time. And software, some people say, Well, anytime that happens, the system is just going to crash and reboot. Well, that is just not true. What they find from industry studies from IBM is that sometimes you get a crash -- IBM is speak for a system crash -- but sometimes you just get an incorrect output and you have no idea that it was incorrect unless you have a second independent system checking it.

Q So we have hardware faults and we have software faults. How often do these random faults happen?

A They happen often enough that when you have a lot of vehicles it's a problem. They don't happen often enough that you will see them in system testing for the most part, and that's what makes them tough.

Q Tell us a little bit about your analysis here.

A For example, hardware faults are about every 10,000 to 100,000 hours per chip. That is just a general number from the literature. And out of those faults, maybe only two percent are dangerous. I've seen numbers a bit higher, but a lot of faults. Okay, the thing crashes, reboots, no big deal. This happens to your PC once in a while, most of you I imagine. It crashes and reboots.

Sometimes it is a software bug, sometimes it is one of these cosmic ray things, and you go on. But sometimes it corrupts something that is critical. And for safety critical systems of this type, Obermaisser was actually studying cars. He said about two percent tend to be dangerous. So this is going to happen, ballpark, one time per million hours. Million hours is a lot of hours.

But if you have a half million vehicles out on the road, and they are driving about an hour a day, that is a pretty typical number, then you will get maybe 31 dangerous faults a year across all 430,000 cars. That is an approximate number, but it is in the ballpark, or maybe 314. So you will see these kind of faults on a regular basis if you deploy enough systems.

The catch with testing is if you test ten vehicles for a year, you just don't have enough hours to see one of these, but they are going to happen in the real fleet.

Q So what I'm hearing is, Listen, these faults are going to happen. Why is it that Toyota should be expected on these numbers that you posted up here to be responsible for those numbers on safety critical systems?

A So on a safety critical system, these are the standard numbers that everyone in the field knows. If you asked me before the trial, I would have said, Oh, about once every 100,000 hours. That is just the way it is. If you're designing a safety critical system, you know this is going to happen, because it happens to everyone that designs these: rail, air, or space, where ever it is. It happens to everyone.

So you're talking about a dangerous fault every week or two, and so you need to do something about it. And what you need to do is you need to use two CPUs that are completely independent so if one fails the other one catches it and makes the system safe.

Q Did Toyota use two CPUs that are independent?

A They used two CPUs, but they're not sufficiently independent.

Q All right. Tell us about some research that you looked at.

A So I did some background research, and so Vinter in 2001, he is at the Chowmers (phonetic) Group, and they have a lot of sponsorship from Volvo, although I don't know if this particular paper was sponsored by Volvo, but I know these guys.

What they did was they put bit-flips into a car engine throttle control. So what they did was said, Let's pretend one of these cosmic rays flips a bit or a software fault corrupts a memory location and see what happens. Sure enough, they found it opened up to full throttle. And so this says in the research community it was well known that bit-flips could result in a wide-open throttle that would be unsafe.

Q When was that information?

A This was in 2001. There is a much older paper by Addy.

Q Who is Addy?

A I don't know that gentleman, but he did an analysis of an industrial realtime control system, so he had a real system, and he found bugs in it. And he found software bugs that a single-bit overwrite could cause a system to be unsafe. And he found memory override bugs.

So the point of this if you are designing a safety critical system, you should expect software bugs will corrupt memory, and you should expect that hardware faults will cause unsafe behavior; therefore, you better do something to prevent it.

Q How do you handle memory corruption?

A So for memory corruption there are two standard techniques. One is you might have two copies of a variable, so you keep the same number in two different spreadsheet cells. So if one gets messed up, you don't know which one is right, but you can compare the two and say, They're not the same, something happened. At least you can detect it. And that gives some protection against both hardware and software corruption.

Another way to do it is to use hardware error detection and correction, EDAC, otherwise known as error correcting codes. You take the value and you put another thing called a check value. Parity (phonetic) is a simple one. The original IBM PCs had parity on them as early as 1982 when I got one. And it is just a couple of extra bits that you just add up all the bits in the pedal position and you say, I see an even number of bits or I see an odd number. And if one of them flip, even changes to odd, and you say, Oh, something is wrong with that. The more sophisticated ones, of course.

Q On point number one, did Toyota do this mirroring? Did they do this software corruption detection mirroring?

A They did mirroring on some variables, but not all variables on the main CPU. And I don't have information that would lead me to believe they did mirroring on the monitor CPU.

Q Is it vital to have mirroring done on all variables not just some variables?

A Given the architecture you would expect to mirror all the variables that can result in an unsafe behavior.

Q Okay. What about number two? Did Toyota do -- perform this on their system?

A Toyota did not have this on the 2005 --

Q All right.

A -- for a ramp. They had it for program memory, but not for the working memory.

Q Tell me what you're describing here.

A So this is what we just went over, that some critical variables are mirrored, but not all of them. The operating system variables are not mirrored. Let me take an aside, because Mr. Ishii was talking about an operating system.

An operating system is a piece of software that runs on the hardware and provides basic services, so think about Windows or a MAC OS. It's not the spreadsheet program, but it schedules different jobs and switches between different tasks and provides basic services.

And the operating system on the main CPU did not mirror its variables either, and that means that if one of those variables is corrupted you can expect it to not run its tasks properly, or something like that.

And so based on all of this, what I do is I conclude because they did not fully protect memory, for that reason alone, you will expect there will be random faults from either hardware or software sources that will corrupt memory and some fraction of them are going to be dangerous.

Q And you described those percentages of what would be dangerous?

A Those are the standard percentages. Yes.

Q Now, we talked about some of your general overview, big broad engines. And your first one was that the Toyota electronic throttle control system design is defective and it is dangerous; is that right?

A That's correct.

Q When we talk about that, we also have, if you will look at point two there, it says that defective safety architecture with an obvious point of failure. What does that mean?

A A single point of failure is one place that if that has a problem the system is unsafe. And just -- this is probably the most important point in safety critical system design. If you have any single point of failure, the system is by definition unsafe. All the safety standards say you cannot have any single point of failure.

Q Since it is so important, I think we need to completely understand single point of failure. Give us an understanding of a single point of failure.

A So a single point of failure is some piece of hardware or software that has complete control over whether the system is safe or not. And so if it fails due to a random hardware event or a software bug, if it fails, then the system is unsafe. And it is kind of tricky because you don't say, Well, I can think of five ways for it to fail, and I protect against all those five; that is not good enough.

It doesn't matter whether you're smart enough to think about how it is going to fail. When you have millions of vehicles on the road, it will find a way to fail you didn't think about. So the rule is simply you cannot have a single point of failure.

Q Did Toyota have a single point of failure on their software?

A They had -- absolutely had a single point of failure in the ETCS, and we have slides that will show exactly where one of them is.

Q Let's talk about those. Go to -- tell us what a fault model is.

A A fault model is how you look at faults. And so in a safety critical system, you say, What faults do I care about, what faults do I not care about. Well, we will not worry about a meteor coming out of the sky and hitting the car; that is outside our fault model. That is not a design problem.

Q What is a fault?

A But a fault is a hardware bit-flip or a software bug, and we are going to worry about those. Not only worry about some of them, we will worry about any one that possibly occur whether we can imagine it or not. Because with a million or more vehicles on the road, it doesn't matter if we are smart enough to think about it, it will find a way to happen.

Q And a commonly accepted fault model, what do you mean by model?

A By fault model, we have a description of the faults we care about. That is our fault model, that is just what people call it.

Q In a commonly accepted fault model, is the arbitrary single point fault, where do you get that from?

A So that is, for example, in the MISRA report two, which is part of the thick MISRA, no single point of failure within the system can lead to a potentially unsafe state, in particular for the higher integrity levels. And some of the other literature makes it clear that there is no restriction on how it can fail, it just fails in the worst way possible.

Q Are you saying within the MISRA documents, in terms of a standard, the standard would be there can be no single point of failure within the system that can lead to a potentially unsafe state in particular for the higher integrity levels?

A That's true. That standard and in every other safety standard I've ever seen.

Q This notes this standard has been in place since 1994?

A That is correct.

Q All right. Turn to the next page here?

A Here is some more. Nancy Leveson, came up with the academic research field of software safety. And in her manifesto, if you want to call it that, she says no single fault can cause a hazardous effect where hazardous means dangerous.

And over here there is another one. You cannot consider the probability of the failure for the single fault. Regardless of how remote that chance is, you have to tolerate every single point failure. And it has broad implications, as I've said.

Q When you talk about any single point of failure, are you tell us that they should be able to mitigate all faults?

A So if you have a picture of the system and you can point to a box and the box is the only place that something happens, and that something affects the safety of the system, if there is only one box, it is unsafe. That is one way to look at it.

Q That seems like a heavy, high standard.

A It is a high standard, but then again you're talking about systems that can kill people, so high standard is warranted. All the systems I reviewed for safety and the people who are getting it right, all meet that standard.

Q Now, when we talk about this fault model, let's compare it to Toyota's fault model.

A So I've had access to fail modes and effects analysis. This is an engineering practice where you ask, Here is A/C, A/D converter, and we're going to talk about that in a minute. And it says it's not a dual system, there is only one. It says, Okay, here is some ways it can go wrong, this bit can get stuck, this bit can get stuck.

And you can see there is only four categories that they enumerate. They say, Okay, we have countermeasures against those or we don't -- in this case, we don't think it's likely to happen. So what you saw before, it doesn't matter how likely you set off the guard, here they are saying, We just don't think it will happen.

MR. BIBB: Objection, your Honor, here is what we're saying. We're interpreting the document. Motion in limine.

THE COURT: Let me just explain to the witness. I don't want you to tell me what you think Toyota meant by anything. You can tell me your interpretation of the documents.

THE WITNESS: I understand, your Honor. So my interpretation of the document is what I said, to clarify.

Q (By Mr. Portis) Now, let me stop you there. Did you -- did you use this word failure mode effects analysis?

A Yes, I did.

Q What is that?

A So that is a technique where you hypothesis all the faults that can happen and see whether or not your system is safe despite them happening.

Q In your analysis, they looked at four areas?

A They looked at a few. They didn't look at, Well, what is the worse that can happen, which is required for a safety analysis.

Q Where is it required?

A Back here where it says any single point of failure, no single fault can cause a hazardous effect. And the documents don't say the ones you can think of, they don't say the ones that are easy to understand, they say any.

Q All right. I want us to spend our remaining time before lunch going through this next slide.

A Absolutely. So this is a picture you have seen before. This is the ETCS. And I'm going to talk about the shared A/D converter. So A standards for analog, D is digital. The real world is analog. You have voltages, 110 volts, five volts, whatever. And computers only know ones and zeros.

In order for an embedded system to see what is going on in the outside world or move a throttle, they have to convert between analog, the real world, and digital, the computer world. So an A/D converter -- and this is actually combined. Some of them are just highs and lows, and some are actually different voltages that vary over time.

Let's take a look at the accelerator pedal. So this is a voltage that changes as you move your foot up and down on the accelerator pedal. This has to be converted from analog on this side to digital, and then it is sent to the monitor CPU, and it is also sent to the main CPU. And that is how the software knows where the accelerator pedal is.

Q How specifically does this affect UA, unintended acceleration?

A So the way it affects UA is the pedal position can -- has, obviously, affects the throttle position because when you press down on the pedal you are supposed to make the engine go faster. What you have is both copies. Now, there is two copies in case one of the sensors is bad, and they cross check them and some other things.

It is going through the same A/D converter. If you look at the detailed documentation for this chip, there is one hardware circuit that does the conversion. And it is going through the same one. That means that in the worse case, if there is a fault in this A/D converter, it could basically lie to the rest of the system about what your foot is doing on the gas pedal. If it has a fault that says, All right, the gas pedal is all the way down, the rest of the system is just going to believe that.

Q Just so I understand this, faults are going to occur?

A Faults happen in every computer system.

Q So you someone mashing an accelerator pedal, right?

A So they're mashing it. Right.

Q And so this is voltage?

A So these are two voltages that indicate accelerator pedal fully depressed. This is not a fault mode right now, we're just talking about normal operation.

Q And both of this information goes to this digital input?

A In this case, it goes to the A/D portion.

Q All right. And it is converted?

A Converted to digital bits that say, Hey, the gas pedal is all the way down.

Q And this information is sent to the sub CPU?

A It is sent to both the sup CPU and the main CPU. And it says, The gas pedal is all the way down. Okay, let's get the throttle more open because the driver wants to speed up.

Q What if there is a single point failure right here?

A If one of these two wires goes bad then you're okay because there are two of them. And this will, if it's working properly, notice they don't match with each other and invoke one of the failsafes.

Q What if there is a failure here?

A If there is a failure here, for some of the failures it will defect that it's failed. For some of the failures, it will result in the voltages not matching. But whether we're not smart enough to think about it or not, there is a single point failure that there is always the possibility that something in here will cause the two voltages to be read as though the gas pedal is all the way down without noticing there is a problem.

Q What is the failsafe involved in this, or is there one?

A I don't know if -- I don't know of a failsafe that will catch all possible, all single point faults in the A/D converter.

Q What is your concern with that?

A My concern with it that makes the system unsafe. For example, there could be a fault that just the A/D converter just decides to say, Do you know what, gas pedal is all the way down, even though it's not.

Q And what is the failsafe design by Toyota into this system?

A So the failsafes are based on this failure mode and effects analysis that basically says we're never going to have a situation in which these two signals come through in a way -- in a way that is wrong but undetectable. They're assuming you can always detect that something is wrong.

Q Why is it wrong to make that assumption by Toyota?

A   Making that assumption limits your fault model to only faults that are detectable, not any possible fault. So that falls short of the requirement of the safety standards.

Q   So, again, how does this -- how could this result in unintended acceleration?

A   It could result in unintended acceleration by, for example, if you have your foot on the throttle and you release it and this keeps shoving out stale data. It just stops updating and keeps doing the old accelerator pedal position that you used to have. It could fail that way, but it can also fail by just spitting out an arbitrary number. It is a single point of failure. And when you look at these, you say, What is the worse thing this could do? Well, the worse thing it can do is probably command wide open throttle. And there is no independent check and balance to stop doing it, and that makes it unsafe.

Q   And that was my next question. Will Toyota's failsafe catch those -- will Toyota's failsafe catch those failures?

A   No, it cannot. Because it is basically trusting that it will be able to detect any difference, and that's a restricted fault model, it is not a general fault model.

Q   So if a -- if one of these fault bits come into play the -- let me start over.

Can this will single point of failure give back information to the monitor and the CPU?

A So it could -- to make it a little more humanlike, it could lie to them, and there would be no way to tell. Sometimes you catch them in a lie, and sometimes not, depends how that particular fault shows it.

Q It will produce bad information?

A It will produce bad information. And some fraction of the time you can detect it. Probably most of the time you can detect it, but once in a while there is going to be a lie that you just can't tell.

Q Now, is this -- let me ask it this way: One of your first year, or one of your undergraduate students, is this something they would recognize?

A If they haven't been through a safety course, maybe, maybe not. But if they had any lecture on safety at all, they're going to say -- so I've actually tried this with some students that have been through my safety course. I say, Here is a picture out of the NASA report. What do you think? And they say, That is a single point of failure.

Q What is the -- is this known to be dangerous?

A Absolutely known to be dangerous. In 1999 there was a paper where they did a study and say, Gee, do you need two CPUs each with independent inputs from the throttle, or can you share them? What they concluded was that -- so there was four different systems they looked at. They had an electronic control unit, so they basically had an ETCS controlling the throttle. Just the picture that you saw for a Toyota ETCS.

What they concluded was that if you only have one throttle input it is dangerous. If you have two independent throttle inputs, it is the same processor, p1, it is also dangerous. However, if you have two processors, two computers, and each computer has its own independent throttle input, then that's safe.

Q Okay. You say that safe dual processors don't share inputs, correct?

A That's correct.

Q And in this particular model that Toyota has that they designed, did they share input?

A They shared inputs. And you saw all of the inputs coming through the same A/D converter on the monitor chip. So that means if there is a fault in the monitor chip it could send bad data over the main CPU. And the main CPU has no independent way to check it.

So instead what you want to do, all the safe systems I have worked with have had independent inputs. They have two CPUs. Each CPU gets its own set of inputs. So in this case, there is already two accelerator pedal inputs. You write one to the first CPU you write one to the second CPU. Then the two computers cross-check and said, I got 10 degrees, what did you got? I got 10 degrees. Okay. I got 10 degrees, but it didn't get 10 degrees, it got 20, but it says 10. And the other guy says, No, no, no, I got 20, something is wrong. But you can't do that if everything comes through the same point of failure because there is no independent check.

Q Based on your analysis of this information, and based upon the standards, and based upon looking at the failsafe, and looking at how the dual processors are unsafe, did Toyota -- is it your analysis, did Toyota know this when the system was designed?

A I can't say what Toyota knew. Toyota should have known it.

Q Should have. That would have been a better question.

A Anyone designing a safety critical system should know this, or they have no business designing one.

Q Go to the next one for us. Tell us what this is?

A So this is a portion of a Toyota document. And it talks about how to understand countermeasures for faults. So it is a long document that says, All right, there is different levels of protection for exactly the kind of faults that we're talking about. Level one protection is you need a redundant input to another CPU because that way you can defect abnormalities of the input circuit, just what I've been talking about.

Level two is inputs to the same CPU, and sometimes you will not detect abnormalities. So that is all the same things I've been saying. So based on this, when I read this, Toyota is telling me that --

MR. BIBB: Objection, your Honor.

THE WITNESS: When I read this, my interpretation of this document is that whoever wrote this document appears to be saying what I've been saying.

Q (By Mr. Portis) And this is -- this is Exhibit 5692, which we will offer later. This is a Toyota document?

A This is a Toyota document. Yes.

Q Okay. Go through two more slides. You're saying that some electronic throttle control system malfunction will go undetected.

A So this is a Toyota document. It is a set of PowerPoint slides, and then there is notes, end notes for the slides. So this is slide five and just the corresponding part of the notes. When I read this document, my impression is whoever edited this document read the document and said, Oh, the document is saying never let a malfunction go undetected.

Q Let me stop you there. Did they?

A No, that's not what they did.

Q Okay.

A And whoever annotated this -- so this annotation was already on the document that I got from Toyota. So I added the yellow box so you can see it, but everything else was already there. So the annotation says redundancy does not exist for everything. Change this sentence to address that issue, which I interpret the word "never" is incorrect.

Q But my question is: Should they have kept this word "never"?

A Well, if they said never it would be incorrect.

Q Okay. What should this language be?

A It should say that some failures will be -- some malfunctions will be detected, and some will be undetected.

Q What is your concern here?

A My concern is on a safety critical system if you have a malfunction that is undetected, then that makes the system unsafe.

Q The fact that redundancy does not exist for everything?

A That's another way of saying that there is a single point of failure.

Q And this is a Toyota -- this is a Toyota document?

A This is a Toyota document. Yes. On the bottom, they say the analog to digital conversion of the pedal/throttle sensor signals, the gas pedal, is only performed by one processor; hence, you should not say never let the malfunction go undetected, which I believe corresponds exactly to what I've been saying.

Q What is your concern then?

A My concern is it's a single point of failure and may make the system unsafe.

Q And Toyota was aware?

A When I look at this document, my impression is that whoever wrote this document understood that it made it unsafe.

Q That is Exhibit 5693. And then finally this last slide here, you say the single point of failure is dangerous.

A Any single point of failure. It doesn't matter how many failsafes you put in. It doesn't matter how much analysis that you do. If there is a single point of failure, by every safety standard I have ever seen, it is by definition unsafe, and no amount of countermeasures, no amount of failsafes will fix that. They will reduce how often it happens, but it won't completely fix it. Because we have millions of vehicles out there, it will find a way to fail that you didn't think of, and it will fail.

Q Is there anything else? I notice at the end here you have an example of a jet aircraft. What did you mean?

What is that example about?

A Okay, so this is an example just trying to put it in a different context. If you're flying on an airplane to Asia or Europe, you probably don't want to fly on an airplane with only one engine, because the engines are very reliable. I know the guys that build these engines. They're very reliable, but they are not perfect, they fail every once in a while.

There is a reason commercial airplanes have two engines, and that is in case one fails. But when you talk to them -- and I used to work at Pratt & Whitney, so I have some incite into this -- when you talk to them, they say that is not good enough. You cannot have a single point of failure anywhere on the aircraft because it will find a way to fail.

So an example is you have two jet engines, but you only have one fuel pump. That one fuel pump is going to fail, and both jet engines go out. You can have two fuel pumps, but if the two fuel pumps aren't configured the right way, it is still going to be a problem. I can even draw a picture of this to make it clear if that is useful.

Q You have two minutes.

A Two minutes. So this will not be to scale. You have a plane, you have a jet engine. And there is a fuel tank. The fuel tank is here underneath the wings. And so what you want to do with a good airplane design is you have a fuel pump here and that pumps fuel here, and you have a fuel pump that pumps fuel here. This is an example of a bad design, there is two fuel pumps, one pumps fuel here, and the other one pumps fuel here. And this CPU -- so there is actually a pair of fault-tolerant CPUs, like I was saying, because that is the right way to do it.

But if these CPUs have control of another fuel pump here, because this just causes the fuel to flow, and these pumps are spitting fuel out into the engine. And then if you built it that the fuel then goes over to this engine like this, and there are fuel pumps here, if this CPU turns off these fuel pumps, that guy is not getting any fuel.

So even though you said, Well, I have two fuel pumps, I have redundancy, if a pump breaks, you're covered. But this software has a thing that turns off one fuel pump, you're fine. But if this software turns off both fuel pumps, this engine is going out too.

Now, this is a little different than a car, because in a car when you turn everything off you don't fall out of the sky. But I'm trying to make the analogy that just because you have two of something doesn't solve the problem. You not only have to have redundancy, you have to do it the right way.

MR. PORTIS: Thank you, your Honor. May we break?

THE COURT: Yes. Ladies and gentlemen, it is noon. We're in recess until 1:15. I would remind you: During the break, do not discuss the case, form no opinions. Again, if you didn't check in at the jury assembly room this morning, please do so during the lunch hour.

All rise while the jury exits.

(Whereupon, the lunch recess was had.)

STATE OF OKLAHOMA)
)
COUNTY OF OKLAHOMA)

C-E-R-T-I-F-I-C-A-T-E

I, Karen Twyford, Certified Shorthand Reporter, in and for the County of Oklahoma, State of Oklahoma, do hereby certify that the foregoing transcript is a true, correct, and complete transcript of my stenographic notes.

I further certify that I am not related to any of the parties herein, nor am I interested in any way in the outcome of these proceedings.

WITNESS my Hand this _____ day of ____________, 2013.

____
KAREN TWYFORD
CERTIFIED SHORTHAND REPORTER
CERTIFICATE NO. 01780