GREEN TECH // ADVANCED CARS

FEATURE

This Car Runs on Code

It takes dozens of microprocessors running 100 million lines of code to get a premium car out of the driveway, and this software is only going to get more complex

Image: Daimler

BY ROBERT N. CHARETTE // FEBRUARY 2009

The avionics system in the F-22 Raptor, the current U.S. Air Force frontline jet fighter, consists of about 1.7 million lines of software code. The F-35 Joint Strike Fighter, scheduled to become operational in 2010, will require about 5.7 million lines of code to operate its onboard systems. And Boeing's new 787 Dreamliner, scheduled to be delivered to customers in 2010, requires about 6.5 million lines of software code to operate its avionics and onboard support systems.

These are impressive amounts of software, yet if you bought a premium-class automobile recently, "it probably contains close to 100 million lines of software code," says Manfred Broy, a professor of informatics at Technical University, Munich, and a leading expert on software in cars. All that software executes on 70 to 100 microprocessor-based electronic control units (ECUs) networked throughout the body of your car.

Alfred Katzenbach, the director of information technology management at Daimler, has reportedly said that the radio and navigation system in the current S-class Mercedes-Benz requires over 20 million lines of code alone and that the car contains nearly as many ECUs as the new Airbus A380 (excluding the plane's in-flight entertainment system). Software in cars is only going to grow in both amount and complexity. Late last year, the business research firm Frost & Sullivan estimated that cars will require 200 million to 300 million lines of software code in the near future.

Even low-end cars now have 30 to 50 ECUs embedded in the body, doors, dash, roof, trunk, seats, and just about anywhere else the car's designers can think to put them. That means that most new cars are executing tens of million of lines of software code, controlling everything from your brakes to the volume of your radio [see table, "How and Where Is Software Used in Cars? "].

"Automobiles are no longer a battery, a distributor or alternator, and a carburetor; they are hugely modern in their complexity," says Thomas Little, an electrical engineering professor at Boston University in Massachusetts, who is involved in developing intelligent transportation systems. "The goals to save energy, reduce [emissions], and improve safety have driven the specialization and adoption of electronics in particular."

I have experienced that complexity myself recently. Last year I bought a new car and was staggered to discover a 500-page manual explaining its operations, along with a 200-page companion manual for the GPS and radio systems. One of the new features touted was the much larger glove compartment, but the size was probably dictated by that of the required manuals.

My new car also comes with front and side passenger air bags. The car's air bag electronic controller—along with the dozen or so sensors that provide it with data—have to be able to operate for years in temperatures ranging from the dead of a freezing Minnesota winter to the blazing heat of an Arizona summer sun.

Most of the time the air bag system is just monitoring the car's condition, but if the air bags are triggered by, say, a multiple vehicle collision, the software in the ECU controlling their deployment has 15 to 40 milliseconds to determine "which air bags are activated and in which order," says Broy.

In the near future, Broy says, air-bag control systems will use more than just crash impact information. For example, BMW has just announced that many 2009 models will be equipped with its BMW Assist system, which features a "risk of severe injury" calculation based on information gathered from the car's air-bag controller and its other ECUs, which will inform accident response teams not only where the accident took place, but the likelihood of passengers being severely injured.

The current amount of software in cars is pretty amazing, given that the first production automotive microcomputer ECU was a single-function controller used for electronic spark timing in the 1977 General Motors Oldsmobile Toronado. In 1978, GM offered as an option its Cadillac Trip Computer on the Cadillac Seville. The computer, a modified Motorola 6802 microprocessor chip, displayed speed, fuel, trip, and engine information. However, the chip served another function: It was used by GM to test how well a microprocessor could control multiple functions such as port fuel injection, electronic spark timing, and cruise control.

By 1981, GM was using microprocessor-based engine controls executing about 50 000 lines of code across its entire domestic passenger car production. Other car companies quickly followed suit.

Jonas Bereisa, a GM engineer, wrote in a 1983 article in *IEEE Transactions on Industrial Electronics* that "software development will become the single most important consideration in new product development engineering." How right he was. Broy estimates that more than 80 percent of car innovations come from computer systems and that software has become the major contributor of value (as well as sticker price) in cars. The cost of electronics as a percent of vehicle costs climbed from around 5 percent in the late 1970s to 15 percent by 2005 (excluding final assembly costs). For hybrids, where the amount of software needed for engine control alone is nearly twice as great as that for a standard car, the cost of electronics as a percent of vehicle costs is closer to 45 percent. Within 10 years, some experts predict that the percentages relating to the cost of electronics as a percent of vehicle cost are expected to rise to 50 percent for conventional vehicles and 80 percent for hybrids.

For today's premium cars, "the cost of software and electronics can reach 35 to 40 percent of the cost of a car," states Broy, with software development contributing about 13 to 15 percent of that cost. He says that if it costs US $10 a line for developed software—a cost he says is low—for a premium car, its software alone represents about a billion dollars' worth of investment.

John Voelcker, *IEEE Spectrum* 's automotive editor, wrote in April 2007 about the GMC Yukon hybrid automobile and its Two-Mode Hybrid automatic transmission. Voelcker told me that "of all the staff hours in the entire program to build the Two-Mode Hybrid transmission…some 70 percent…were devoted to developing the control software."

As Voelcker pointed out in his story, that control software logic analyzes hundreds of inputs every 10 milliseconds, including vehicle load, engine operations, battery parameters, and the temperatures in the high-voltage electric components.

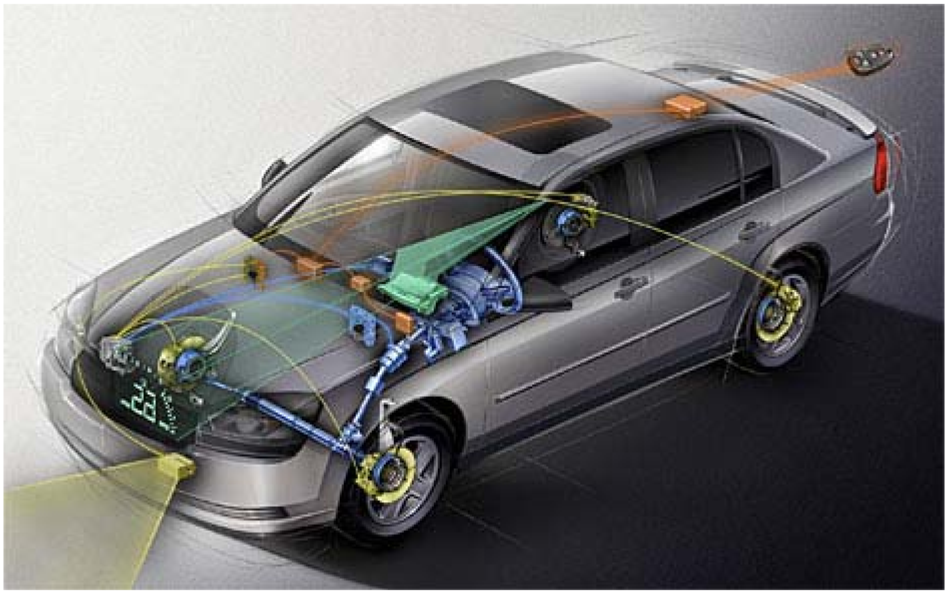
Image: General Motors

Such complexity brings with it reliability issues. IBM claims that approximately 50 percent of car warranty costs are now related to electronics and their embedded software, costing automakers in the United States around $350 and European automakers 250 per vehicle in 2005.

In 2005, Toyota voluntarily recalled 160 000 of its 2004 and some early 2005 model year Prius hybrids because of a software problem that caused the car to suddenly stall or shut down. The time needed to repair the software was estimated at about 90 minutes per vehicle, or about 240 000 person-hours. Even at cost, that is a lot of money.

Last year alone, there were several automotive recall notices related to software problems. For example, in May 2008, Chrysler recalled 24 535 of its 2006 Jeep Commanders because of a problem in the automatic-transmission software. Then in June, Volkswagen recalled about 4000 of its 2008 Passats and Passat Wagons and about 2500 Tiguans for a problem in the engine-control-module software that could cause an unexpected increase in engine revolutions per minute when the air-conditioning is turned on. In November, GM recalled 12 662 of its 2009 Cadillac CTS vehicles for a software problem within the passenger-sensing system that could disable the front passenger air bag when it should be enabled or enable it when it should be disabled. It is a tribute to the automotive software developers, though, that there aren't many more recalls, given all the software in cars.

The increased use of software has not only affected car warranty costs but has also made cars harder to repair—so much so that insurance companies increasingly find it cheaper to declare cars damaged in accidents total losses than it is to fix them.

It is not hard to understand why. "In a premium car you have 2000 to 3000 singular functions that are related to software," Broy says. These are then combined into the 250 to 300 functions used by the driver and passengers to operate the car's systems.

And unlike most commercial aircraft, which have strict firewalls between critical avionic systems and the in-flight entertainment systems, there is more commingling of information between the electronic systems used to operate the car and those for entertaining the driver and passengers. According to a Wharton Business School article entitled "Car Trouble: Should We Recall the U.S. Auto Industry?," a few years ago, some Mercedes drivers found that their seats moved if they pushed a certain button; the problem was that the button was supposed to operate the navigation system.

Roughly one-third of all the software in cars is devoted just to diagnostics, according to a former automotive engineer I spoke to. But even with all that diagnostic information produced, car mechanics often cannot determine the exact cause of the trouble.

Broy told me that more than 50 percent of the ECUs that mechanics replace in cars are technically error free: They exhibit neither a hardware nor a software problem. Mechanics replace the ECUs simply because they don't have a better way to fix them, he says.

"The garages and the maintenance people are really at a point where repairing a car is too complex and demanding [for them]," says Broy. Remote diagnostics and repair are likely to render mechanics obsolete for many tasks.

In the not-so-distant future, says Broy, when you have a problem with the computer system in your car, you will go to your garage, where your car will be connected to a network so that off-site OEM specialists can download data, do the analysis, and then upload a software correction.

Voelcker says he wouldn't be surprised to see onboard systems like BMW's Assist, Ford's Sync, and GM's OnStar soon begin routinely feeding operating data parameters back to centralized systems run by the car manufacturers that will analyze the data for parts drifting out of spec or for software that needs updating and automatically inform the driver that the car needs to be brought in for repair.

Besides monitoring their own internal health, cars are beginning to analyze the world around them. "We're getting into this era where in addition to sensing what's going on inside the car, we are using things like radars to detect the presence of external objects, lasers to measure distance for cruise control, and video and ultrasonics to detect objects behind you," says Little. "The trend will be to extract information external to your vehicle about other vehicles and then exploit this information" to improve safety.For example, cars in front of you will let your car know whether there is ice on the highway or an accident.

Says Little, "We are giving up little pieces of control in exchange for safety. The interesting question is, at what point will you and I be willing to say, 'Okay. I am not going to drive the car; it is going to drive me.' "

About the Author

Robert N. Charette, an IEEE Spectrum contributing editor, is a self-described "risk ecologist" who investigates the impact of the changing concept of risk on technology and societal development. Charette also writes IEEE Spectrum Online's The Risk Factor.

To Probe Further

John Voelcker reviewed the Top 10 Tech Cars in the April 2008 edition of IEEE Spectrum.

Manfred Broy and his colleagues wrote a comprehensive article for the February 2007 issue of *Proceedings of the IEEE* titled " Engineering Automotive Software", which is probably one of the best overviews of how software is used in—and developed for—cars.

For a good early historical perspective on software use in cars, see the article by Jonas Bereisa in the May 1983 issue of *IEEE Transactions on Industrial Electronics* titled " Applications of Microprocessors in Automotive Electronics." It provides an interesting chronology of many of the microcomputer applications that were used in cars from 1977 to 1982.

most recent comments

don mccallum 02.25.2010

One last shot. It's a sure fire bet that there is no 6502 running a million lines of code. If those guys were stuck using those old 8 bit uP workhorses, they'd be writing in Assembly and LOC would equal Machine Code. And no more of that than 16 bits of address could access. Oh yes, they could bloat their code by using bank switching, but that's more work..

DON MCCALLUM 02.25.2010

That's 100 million opportunities for failure per vehicle. And I seriously doubt that any of it is tested nearly as well as that in the F22. Military software development teams have to jump through some pretty high hurdles in terms of reliability. Terms like redundancy and fault tolerance haunt the lives of those engineers. With the current donnybrook spinning around Toyata, It would be interesting to have a critical look at the software engineering validation test suites used to move this stuff into production. I would cynically guess it is way too inadequate. With IBM claiming that 50% of car warranty costs now related to electronics and software I would ask what happens when that all too short warranty ends. Then the consumers are stuck with the costs for keeping all this stuff working. Perhaps one way of improving the reliability of this stuff would be to require manufacturers to warranty the electronic/software content of their vehicles for 25 years or the junking and subsequent destruction of the vehicle. That should put a damper on their enthusiasm for bloating their cars software out past 100 million lines of code. What the heck are they doing with all that code anyway. Sounds like they're using a Microsoft OS. Oh God! Not that! I think I'll pass on buying any new cars and just keep my old clunkers going. With a little extra TLC they should last till I die, and then I won't have to worry about it anymore..

ALAN T 02.22.2010

I think this article is conflating two computing terms. "Machine Code" and "Code". The first means that there is - for one example used in the article - 20 megabytes worth of machine code used for controlling the radio/Sat nav - and yes, that's quite believable. The 2nd term, "Code" usually means human-readable high level language software, written in - for example - BASIC or C++. I think it highly unlikely that there are 20 million lines of such code for this function. There may be 1,000 lines of high level language code which, when compiled, produces 20 megabytes of "machine code" I think the writer has assumed that the terms "code" and "Machine code" are interchangeable, which they are not. Or, possibly the interviewees wanted to inflate their claims over rivals by deliberately confusing a non-technical journalist. Seems to have worked, if that was their intent!.

YBGREENE 02.18.2010
"Such complexity brings with it reliability issues. IBM claims that approximately 50 percent of car warranty costs are now related to electronics and their embedded software, costing automakers in the United States around $350 and European automakers 250 per vehicle in 2005". The complexity is definitely an economic concern. I recently had a warranty repair to my Dodge because the lights and wiper could not be turned off save pulling the fuses. Fix?.... Replace: main Instrument Cluster Assembly, Wiper motor, and Power Distribution Center. I got an oil change while it was in the shop...glad it was under warranty. But somebody has to pay eventually..

KENNETH ACASON 02.18.2010
The article is interesting, there is however a difference between marketing features of automobiles and extending the safety of operational aircraft. Cars used to manage quiet well without navigation, it does not impact the safety of operation. Whilst premium cars are wonderfully equipped, I have rolled in a car 450 degrees without the equipped airbags going off. Fine timing in the order of 20ms for passenger safety may help in a few cases, not all. My business is based on control and communications, yet I do not claim (count) the code that underlays the open standards that make a system reliable and extendable. Sadly the prognosis for 50% of vehicle costs being sunk in software may have some truth. The USA could fly to the moon in 1969, but not today! Are we making more out of the development process than the end result requires?.

ABDULLAH PERVAIZ 02.18.2010
handy information provided above about the Controllers of cars and the planes. so much large codes are designed i-e millions lines of code. very hard thing to develop. thats why the price of new models of cars is so high..

ONSCREENCARS.COM 02.18.2010
It's amazing what information technology does in today's automotive industry. We take our IT and IS guys for granted!.

DAVE FROM OTTAWA 02.17.2010
The notion of car runs on codes does not give me good feeling, au contraire, especially when one relates to the massive recalls of Toyota on the so-called gas pedal defectives. I think it would mask the real problem behind: a glitch in the Electronic Throttle Control codes! Need I say more....

TALLEY TRANS 02.12.2010
Superb Robert. This is the best retrospective & forecast that I've read on automotive computer controls. Can you imagine an Audi transmission control computer boiling in transmission fluid. Perhaps it's one of the CVT's, but no longer external. Michael Talley CEO Talley Trans, Inc. www.talleytransinc.com 1-888-413-1875 .

KOSALA UDUGAHAPATTUWA 02.12.2010
know obout your car in future.

PASSINGBY 02.05.2010
have your computer ever crash? now imagine computer in your car crash suddenly! what will happen? ask BMW thailand, they have done a good job on removing the news and content related to the software and electronic failure, but not in the thai language keyword. i use to live there and and BMW lead on fire cause by electronic and or car software failure in that country. google for BMW ¿¿¿¿¿¿ and see it yourself. or ask toyota today! it is on the news everywhere.
.

RICHARD TUHRO 10.23.2009
another advantage code intensive car parts gives the manufacturers is control over replacement parts and service. you won't be able to go to a salvage yard or an after market vendor and get, for example, a throttle body, bolt it on your engine and have it work without the network management of the car recognizing it as an "approved" (read fee collected by the car manufacturer) replacement. True, this could prevent some safety issues with some modules, but I can't help but think it will be used to control the after market. How long will it be before it is used to set the service life, to prevent "unsafe operation" ? Like your copier shuts down after the photoreceptor has 20,000 cycles to prevent unsatisfactory performance..

STUART JOBBINS 02.25.2009

Having been fortunate enough to work in both the automotive and aerospace industries as a software manager on things like powertrain and engine control, and also worked on working groups in at least some part of AUTOSAR, I can readily believe that premium cars are moving towards this level of complexity. Take a look for yourself at the publicly available material. (www.autosar.org).

PEEDS 02.16.2009
100 million lines of code sounds way too large. I don't think a modern car, with all its computing sophistication, would need such an enormous amount of software..

JOHN MALVASO 02.06.2009
This article brings me back....

DANIEL 02.06.2009
Do you think one day manufacturers will put code updates on their website so everyone will be able to update the firmware of their cars?.

CHRISTOPHER DAUERER 02.05.2009
I believe the first microprocessor controler used in an automotive application was actually in 1967. Bosch D-Jetronic (D representing DrÃ¿Â¼ck or pressure) was introduced in late 1967 on the 1968 model year Volkswagen type 3 model, also known as the VW Fastback or Squareback (wagon). .

MILES 02.05.2009
As a student of Automotive Technology this article interests me greatly..

MARTIN GRAVENSTEIN 02.05.2009
Measuring or estimating lines of code is a traditional and yet controversial way of determing the software aspect of a systems development cost. However, for products that are complete and in production there is a much easier method. Development cost is recovered in the products cost over life sales volume. Simplifing things emensely the cost of the product is cost of manufacture + cost of developement + profit. So, quick back of the napkin sanity check for comparing airplane and automotive NRE. Very crude assumptions: Designing, building and profit parameters for the Airplane industry and the automotive industry are similar. Argue that all you want but to a rough order I think this is a reasonable assumption. Cost of a 787 is ~$175M, Weight is 242K lb, volume is ~1000, cost per lb is ~$7. Cost of a Ford Taurus is ~$28K and weighs about 3.7K lb. Sales volume is 6.7M for four generations. Make a conservative assumption that each generation is a new development so divide by 4 and you get cost per lb is about ~$7. The only major difference I see between the two companies financial equations is time. The life cycle for a car is much shorter than for the airplane so Boeing has a different time factor in their ROI than Ford. They have to wait longer recoup their cost and that means they need to recoup more to cover fincancing costs. The back of the napkin says there is not 2 orders of magnitude difference in NRE..

ZACH 02.05.2009
I would venture to say that the LOC number is composed of every piece of the software "suite" running in the machine (car, fighter jet, etc...). This would include the operating system, the libraries, and the actual programs used for all of the various functions inside the car. Just a guess, though..

JOHN WATSON 02.05.2009
I am highly suspicious of the 100 million number and the scale factors involved. Is this to say that the union of unique programs running on the car adds up to 100 million lines?.

JOHN WATSON 02.05.2009
Lines of Code is a rough approximation indeed..

ZACH 02.05.2009
Software complexity has traditionally been measured in lines of code. It is essentially one of the best ways to show the complexity, although as you've pointed out it certainly isn't perfect. It touches both ends of the spectrum because you could have a very complex 100 line program, but at the same time it would be hard to call a 1 million line program "simple"..

RAMIN HONARY 02.05.2009
I'm still new to computer engineering, so this may be a naive question, but I've always been a bit curious about why software is sometimes measured in lines of code. I can imagine it has something to do with giving a round-about figure for the magnitude of the amount of effort gone into designing the system. But still, getting the system to work according to specs with fewer lines of code requires more effort..

GIL 02.05.2009
Isn't there a (wrong) scale factor involved in stating cars run 100 milion lines of code?.