Impact Case Studies
Real-Time Systems

Robert Davis

Real-Time Systems Research Group, University of York

rob.davis@york.ac.uk
http://www-users.cs.york.ac.uk/~robdavis/

Overview

- Background on Real-Time Systems
- Case studies
  - 1. Guaranteeing the real-time performance of in-vehicle networks
  - 2. The world's smallest automotive real-time operating system
  - 3. How long does your real-time software take to run?
- Questions?

What is a Real-Time System?

- Real-Time System is any system which has to respond to externally generated input stimuli within a specified time
  - Functional correctness – the right computations
  - Timing correctness – completed within predefined time constraints
  - Time constraints typically expressed in terms of deadlines

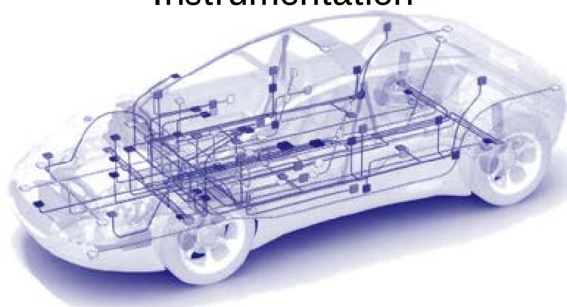
- Hard Real-Time
  - Failure to meet a deadline constitutes a failure of the application (e.g. flight control system)
- Soft Real-Time
  - Latency in excess of the deadline leads to degraded quality of service (e.g. data acquisition, video playback)

Examples of Real-Time Systems

Instrumentation

Robotics and Factory Automation

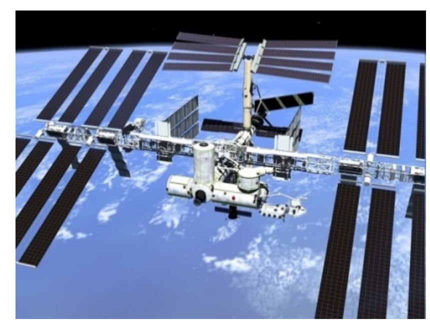
Avionics



Automotive Electronics

Telecommunications

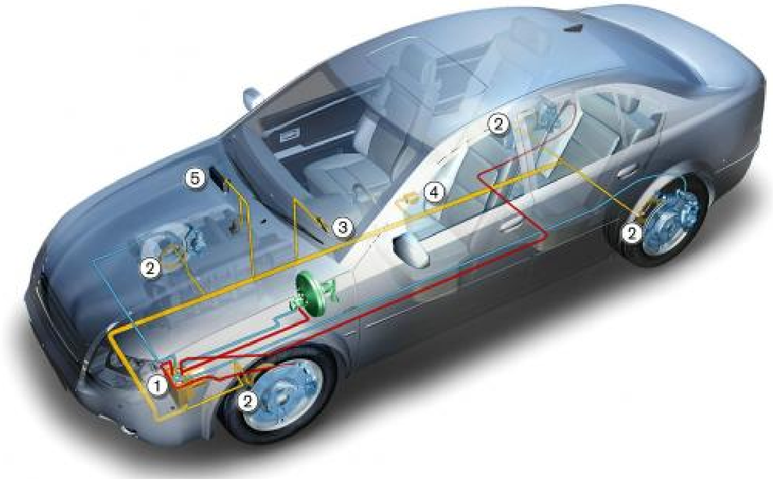
Medical Systems



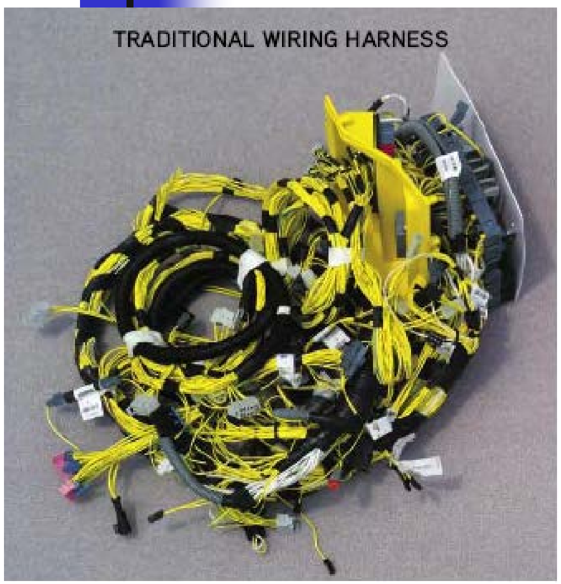
Space

Case Study 1: Guaranteeing the real-time performance of in-vehicle networks

- Controller Area Network (CAN)
  - Simple, robust and efficient, in-vehicle digital communications network
  - Originally developed by BOSCH in the 1980s
  - First used in a production car in the 1991 Mercedes S-Class



Multiplex v. Point-to-point Wiring



- Traditional point-to-point wiring
  - Early 1990s an average luxury car had:
    - 30Kg wiring harness
    - > 1km of copper wire
    - > 300 connectors, 2000 terminals, 1500 wires
  - Expensive to manufacture, install and maintain
    - Example: Door system with 50+ wires

- Multiplex approach (e.g. CAN)
  - Massive reduction in wiring costs
    - Example: Door system reduced to just 4 wires
  - Small added cost of CAN controllers, transceivers etc.
    - Reduced as CAN devices became on-chip peripherals

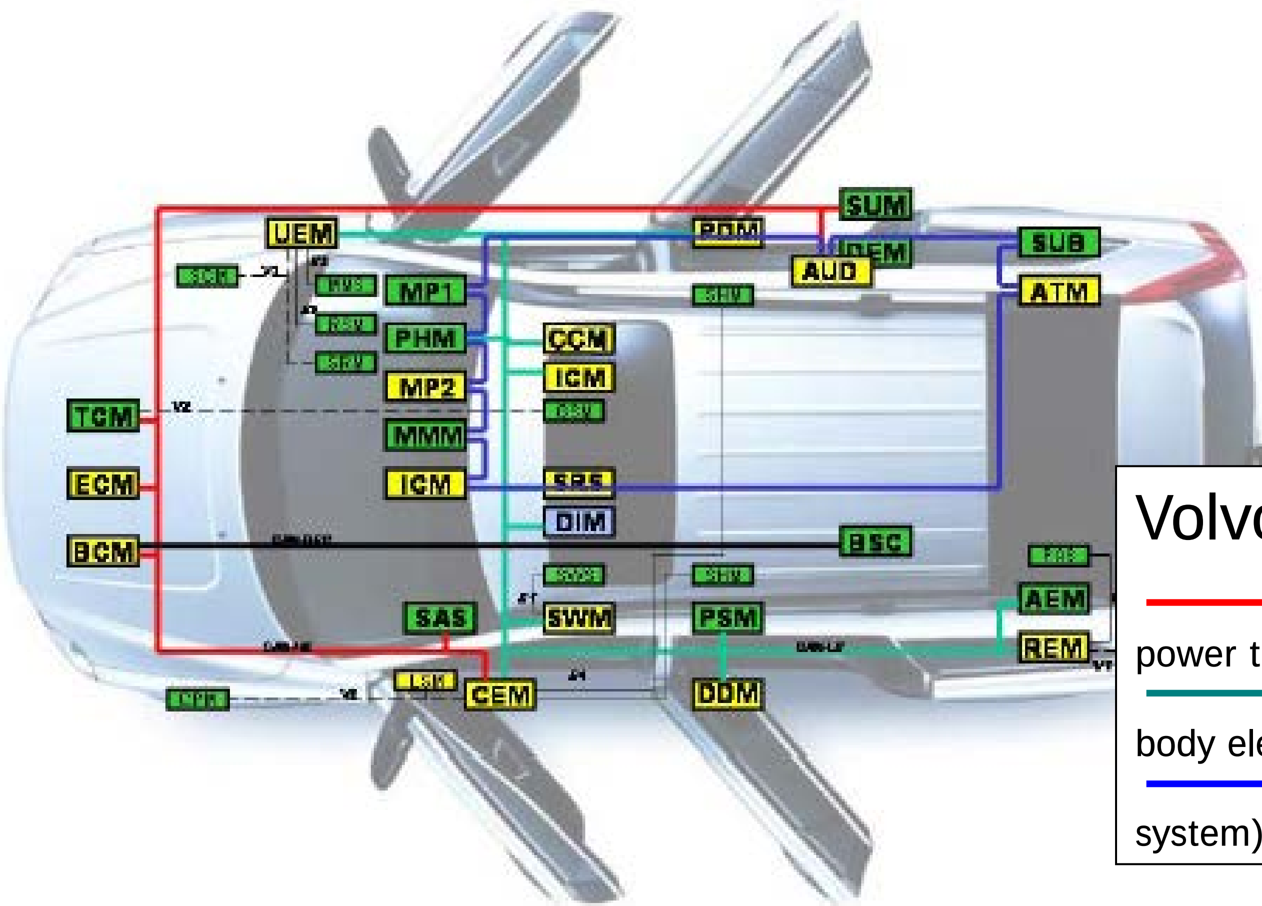
Messages on CAN

- CAN used to communicate *signals* between Electronic Control Units (ECUs)
  - Typically 25-35 ECUs in a modern car
  - Signals include:
    - wheel speeds, oil and water temperature, battery voltage, engine rpm, gear selection, accelerator position, dashboard switch positions, climate control settings, window switch positions, fault codes, diagnostic information etc.
  - > 2,500 signals in a high-end vehicle
  - Multiple signals piggybacked into CAN messages to reduce overhead, but still 100's of CAN messages
- Real-time constraints on signal transmission
  - End-to-end deadlines in the range 10ms – 1sec
  - Example LED brake lights

THE UNIVERSITY of York

RTS York

Volvo XC90 Network Architecture



Volvo XC90

500 Kbit/sec CAN bus for power train

125 Kbit/sec CAN bus for body electronics

MOST (infotainment system)

Schedulability Analysis for CAN

- Research from RTSRG* in 1994
  - Mathematical analysis to compute offline the longest time that each message can take to be transmitted over the network (including time in queues)
  - Used to prove if all messages are guaranteed to meet their deadlines

- Schedulability Analysis
  - Message Length $C_m = \left(g + 8s_m + 13 + \left\lfloor \frac{g + 8s_m - 1}{4} \right\rfloor \right) \tau_{bit}$
  - Queuing delay $w_m^{n+1} = B^{MAX} + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{bit}}{T_k} \right\rceil C_k$
  - Response time $R_m = J_m + w_m + C_m \leq D_m$

*by Ken Tindell, Alan Burns, Andy Wellings

Start-up Company

- Origins
  - In 1994 research presented at a Conference on CAN was picked up by Volvo Car Corporation
- Start-up company **Northern Real-Time Applications Ltd**
  - Founded by Ken Tindell and Rob Davis in 1995
- Products developed
  - Analysis tools “Volcano Network Architect” with Swedish company Kimble AB.
  - CAN device drivers, communications software layer, and configuration tools called “Volcano Target Package”
- Rights transfer
  - Rights to Volcano technology transferred to Swedish company Volcano Communications Technologies AB in 1997
  - Acquired by Mentor Graphics in 2005

Exploitation

- Today Volcano technology is available for more than 30 different microprocessors used in automotive
  - Fujitsu 16LX, FR Series; Hitachi H8S, SH7055, SH7058; Infineon C16x, TC179x, TC176x, XC800, XC2000; Renesas M16C, R32C/M32C; Freescale HC08, HC12, MC683xx, MPC5xx, MAC71xx; S12, S12X, MPC55xx, MPC 56xx; Mitsubishi M32R, MC32C; PowerPC; National CR16; NEC V85x, 78K0; ST Microelectronics ST9, ST10; Texas Instruments TMS470; Toshiba TMP92/TMP94.

Impact

- Volcano Technology
  - First used in Volvo S80 in 1997

  - Subsequently in XC90, S80, S/V/XC70, S60, S40, and V50
  - Approx **4.5 million** Volvo cars since 1997
  - Ford bought Volvo in 1999: Volcano adopted by Jaguar, Land Rover, Aston Martin
  - Used by SAIC since 2007 and Mazda since 2012

Impact

- Benefits for car manufacturers and suppliers
  - Possible to configure networks using 70-80% of bandwidth and still guarantee that all messages meet their deadlines, compares favourably with approx. 30% previously possible
  - Enables more ECUs to be connected to the same network, supporting more functionality at lower cost
  - Fewer wires and connectors, lower network speeds needed, increases reliability
  - Guaranteed performance greatly reduces the time and cost spent testing: No intermittent timing faults due to network reduces warranty costs

- Benefits to wider society
  - More reliable cars with better functionality at lower cost

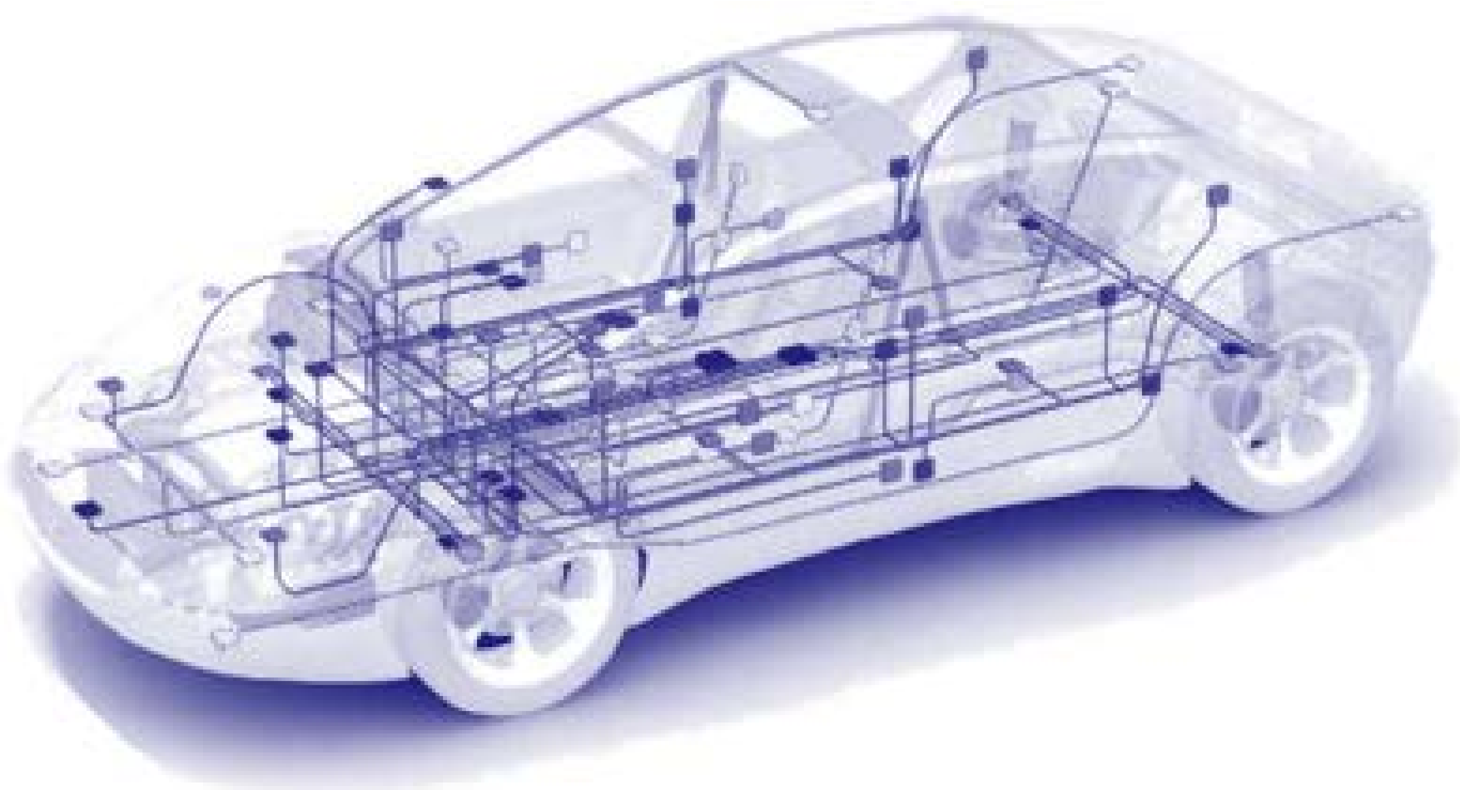
Impact

- Quotes
  - "*The advantages to Volvo of the development and application of Volcano include: Production cost benefits due to high bus efficiency (four times as many signals can be transmitted at half the baud rate). Development cost benefits (in the form of a single, proven implementation which is much cheaper than multiple implementations by suppliers and conformance testing by Volvo). Improved network reliability, resulting in higher product quality. Reduction in Volvo´s test load. Reduction in supplier´s test load.*"

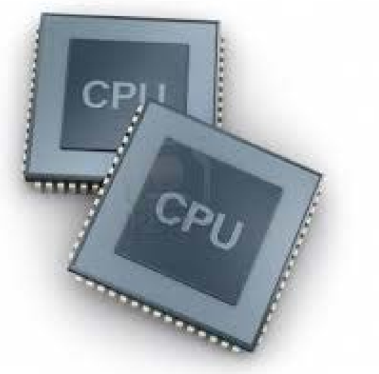
    Volvo 1998
  - "*By using Volcano, network design is made easy and predictable, guaranteeing data communication, which reduces the verification effort to almost zero and eliminates warranty and change costs caused by networking issues.*"

    SAIC 2006

Case Study 2: The world's smallest automotive real-time operating system



Automotive Electronics



- Typical family car (e.g. VW Golf)
  - 25-35 Electronic Control Units connected via two or more communications networks
  - Relatively simple low cost microprocessors (single CPU)

- System functionality
  - Multiple software tasks running on each ECU
  - Time constraints on each task
    - e.g. read and process data, output results by a specified deadline

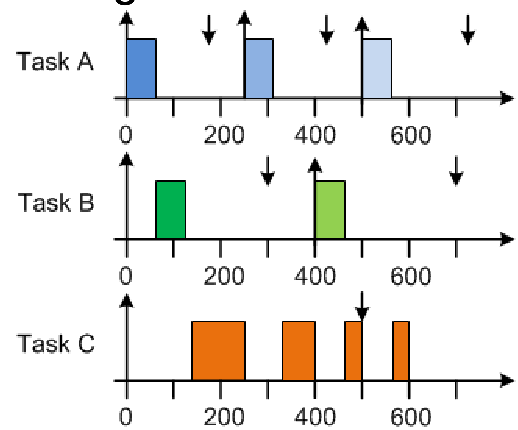
- Real-Time Operating System
  - Needed to schedule when each task can run so that all tasks meet their deadlines
  - Essential that all deadlines are met otherwise the system will suffer intermittent timing faults and poor reliability

Schedulability Analysis for Processors

- Research from RTSRG* in early 1990s
  - Response Time Analysis for fixed priority scheduling
  - Determines longest response time for each task from becoming ready to completing execution
  - Accounts for the complexities of timing behaviour for tasks in automotive systems
  - Accounts for the overheads of a well designed RTOS

$$w_{i,q}^{m+1} = B_i + (q+1)C_i + \sum_{\forall j \in hp(i)} \left\lceil \frac{w_{i,q}^{m} + J_j}{T_j} \right\rceil C_j$$

$$R_i = \max_{\forall q = 0,1,2 \ldots Q_i - 1} (w_{i,q} - qT_i + J_i)$$



*by Neil Audsley, Alan Burns, Ken Tindell, Andy Wellings

Start-up Company

- Origins
  - In 1997, following work for Volvo on Volcano Ken Tindell and Rob Davis founded **Northern Real-Time Applications Ltd**.
  - Purpose: to develop a RTOS for automotive applications

- Aims for the RTOS
  - Systems built using the RTOS must be analysable using schedulability analysis tools
  - RTOS overheads and memory footprint must be much smaller than any other automotive RTOS

Start-up Company

- Development
  - **Real-Time Architect** schedulability analysis tools
  - **RTA-OSEK** real-time operating system
- Standards
  - Influenced OSEK automotive operating system standard to allow single stack execution (enabled low memory use)
- Funding
  - £1M venture capital funding in 1999
  - £9.2M venture capital funding in 2000
- Jobs
  - Grew from <10 to more than 30 employees by 2001

Start-up Company

- Trade Sale
  - ETAS (a subsidiary of Bosch) benchmarked RTA-OSEK against their in-house RTOS and found it was much more efficient – faced with the option to start from scratch or buy the company
  - ETAS bought the company in 2003

Exploitation

- Further development
  - ETAS adapted the operating system to meets the new AUTOSAR standard (RTA-OS)

- RTA-OSEK / RTA-OS Available for over 50microprocessor families including:

  Renesas: V850E, SH2, SH2A, H8S, H8SX, M16C, Xilinx Microblaze, PPC405 Core; Texas Instruments TMS470P, TMS570P; Infineon Tricore TC17x6, C166, XC2000; Freescale Star12, MPC555, MPC55xx, S12X, MPC56x, HC12X16, HC08, HCS12; Fujitsu 16LX; Analog Devices Blackfin, STMicroelectronics ST30, ST7, ST10

Advantages: Low memory usage

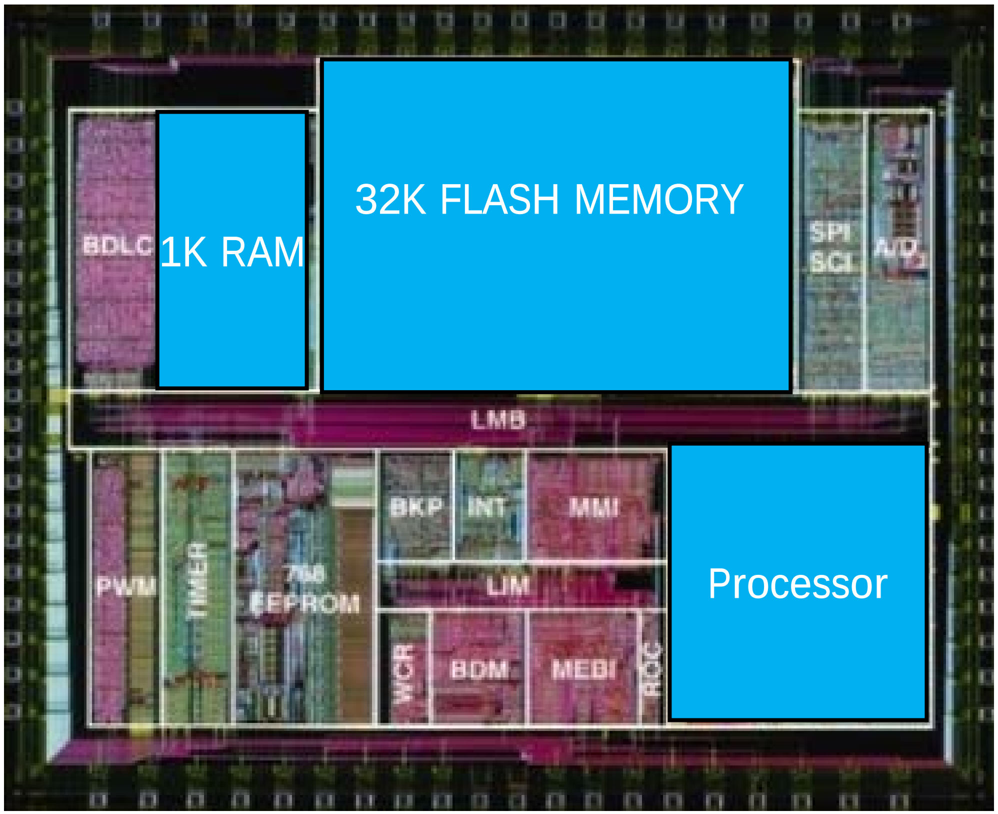
RTA-OS:

World’s smallest and fastest AUTOSAR compliant RTOS

Approx. 1K to 1.5K Bytes (depends on processor)

501025a8 ffe2c04d 09820820 90000010 0f370800 20bbffe2 c04d0000 0340000d 501025c8 04c0000d 0fe2cfcd 0fa6f020 af8ff576
00044019 e8f40f19 0f28f8d4 0fd93f82 501025e8 08dfe8f4 0f590fa6 e8f40f19 11ee0f26 006d800e 0f7fe8f0 0f19e8f4 00198014
50102608 00780f82 46800782 15824482 0a3c0065 af8ff576 0f37ffe2 c04d0148 006d4982 50102628 04c0000d 0fe2cfcd f781ffb7
8fa6f020 c04d4680 09820820 90009202 0f282f82 50102648 ffe2c04d 00000340 000d806a 0f37ffe2 cfcd0fa6 f020af8f f5760f37
080020bb 50102668 0f190f28 f8d40fd9 13da04c0 000d0fe2 0f260004 4019e8f4 0f190008 f65fe8ec 50102688 0282f8d0 065912da
106ee8ec 0f190aee 00780f82 07825000 903b4482 90000f28 501026a8 af8ff576 0f37ffe2 c04d0100 006d4982 04c0000d 0fe2cfcd
f781ffb7 8fa6f020 501026c8 e8ec0919 e8f00819 90009202 0f2812da 000fe8f4 0019fb45 ff6d036e e8ec0f19 501026e8 ff214013
0101f08b f8a54f1b 0f8001b0 e8f00f59 f24c0000 e8ec0f99 e8ec0f59 50102708 0f99ffe2 c04d0000 0340000d fb06ff6d ffb70fa6
4048f020 af8ff576 0f37e8ec 50102728 0fc8e8ec 0f9904c0 000d0fe2 cfcdf781 ffe2c04d 00000340 000d0000 0f2d0000 50102748
cfcd0fa6 f020af8f f5760f37 080020bb 0f19e8ec 0f99fb05 ff6d04c0 000d0fe2 50102768 f8d00f19 e8f40f59 f0e00f0f 1048e8f4
0f590fa6 1048e8f4 0f19f8d0 0f990c6e 50102788 ffa7f83f e8f40f19 f8d00f59 0f82e8f4 ff1d036e e8ec0f19 e8ec0959 e8f00859
501027a8 0340000d 206a0f37 ffe2c04d 9000fab7 af8ff576 0f370800 20bbffe2 c04d0000 501027c8 03d92bda 04c0000d 0fe2cfcd
0fa6f020 424c0004 2259086e 42d4424c 302cf8d4 501027e8 29f6216c e8f00f19 0d3cf202 8010f27f e8f00059 80040f7f e8f00f19
00044019 50102808 e8ec0019 e8e80119 f2200f8b e8e80f19 f5760f37 ffe2c04d 0f68ff40 012b40cc 50102828 000d0fe2 cfcdf781
ffb72fa6 f020af8f f8d40fd9 2fda9000 0282302c 2ada04c0 50102848 90000f28 2eda406c e8ec0f19 0f280282 0f28f8d0 0f59e8ec
0f59f8d5 0fd90f82 50102868 0f590f28 e8f00f59 e8e80f59 f8d60fd9 01b9006d ffa0f8cc 0f59f8c8 0f59e8f4 50102888 0f59285a
e8f802d9 0252006d 39022802 9000fffc 2f240f82 f8cc0f59 395af8c8 501028a8 00000340 000d906a 0f37ffe2 c04d9000 f020af8f
f5760f37 080020bb ffe2c04d 501028c8 0fd9f8d6 0cd904c0 000d0fe2 cfcd0fa6 e8d80fd9 1f828011 08df0428 c814f8d5 501028e8
0f59e8e0 0759e8dc 06590528 0058c02c f5760f37 ffe2c04d c834fe4a ff6de8e4 50102908 000d0fe2 cfcdf781 ffb79fa6 f020af8f
0340000d 806a0f37 ffe2c04d 900004c0 50102928 af8ff576 0f370800 20bbffe2 c04d0000 0cd94fda 04c0000d 0fe2cfcd 0fa6f020
50102948 043c0004 2819046e 42d4424c c02cf8d4 001989f6 0f6840cc 0f82e8f0 0f59214c 50102968 c04dfeaf ff6d8004 0f7fe8f0
0f19e8f4 f781ffb7 8fa6f020 af8ff576 0f37ffe2 50102988 90000282 c02c4eda 04c0000d 0fe2cfcd 146e200c 302cb8b6 02d955da
f8d403d9 501029a8 f8d80019 ffe2c04d 10eeff16 202cffc2 cfcdf781 ffb70fa6 f020af8f f5760f37 501029c8 ffe2c04d 9000302c
54da04c0 000d0fe2 20bbffe2 c04d0000 0340000d 0906a0f3 501029e8 0fe2cfcd 0fa6f020 af8ff576 0f370800 e8f00819 c02cf8d4
0cd957da 04c0000d 50102a08 0f37e8e8 0f99ffe2 c04d126e e8e80f19 cfcdf781 ffb70fa6 1048f020 af8ff576 50102a28 f5760f37
ffe2c04d 2b3c04c0 000d0fe2 04c0000d 0fe2cfcd f781ffb7 f020af8f 50102a48 af8ff576 0f37ffe2 c04d0000 0340000d e8ec0f99
04c0000d 0fe2cfcd 0fa6f020 50102a68 e8f00f19 e8f40019 e8f00f59 f34c0000 ffe2c04d e8f00859 fe2aff6d 80040f7f 50102a88
cfcdf781 ffb79fa6 f020af8f f5760f37 61da9000 0282c02c 56da04c0 000d0fe2 50102aa8 ffe2c04d 00000340 000d0f28 f8d40fd9
cfcd0fa6 f020af8f f5760f37 080020bb 50102ac8 1f8286f6 0808b8b5 0fd904c0 000d0fe2 c04d0000 0340000d 0828f8ec ff6d0f28
50102ae8 04c0000d 0fe2cfcd 0fa6080f f0bbffe2 63dafe3c fd4bff6d 0f28b8b4 0fd90f82 50102b08 0080003b ffe0404d c02c4802
f8d40cd9 ffc1ff6d 7482043e f020af8f f5620f37 50102b28 0340000d ffbbff6d 748294ae ffe0404d af8ff576 0f370800 20bbffe2
c04d0000 50102b48 0fd90282 04c0000d 0fe2cfcd 0fa6f020 f00c0828 0282fe7d ff6d22ee f00cb8b4 50102b68 0f19e8f4 0019f84c
ff6dfe9f ff6d1c6e 0f37ffe2 c04dfda9 ff6d8004 0f7fe8f0 50102b88 000d0fe2 cfcdf781 ffb7f020 af8ff576 73daf8d4 03d99000
1282c02c 62da04c0 50102ba8 0f37ffe2 c04d1dee 200c302c b8b602d9 000d8012 2ffff06a 0f37ffe2 c04d106a 50102bc8 f5760f37
080020bb ffe2c04d 00000340 015904c0 000d0fe2 cfcd0fa6 f020af8f 50102be8 804d9000 302c72da 202c1fc2 200cf8d8 0f8ff864
0f370310 f0372fc2 0082ffe3 50102c08 9000f202 af803000 f00bf020 cf8ff021 9000f4fc 0178f120 50102cf8 f06a0f37 ffe2c04d
13ee200c 302cc8e4 10a6180f f0bb0fe2 c04d0000 0340000d 50102d18 1fc2200c 09240f59 04c0000d 0fe2c0cd f8c80f19 ff63ff6d
9000302c 16da202c 50102d38 0f1bf501 007b9000 f342f8cc 0f19f242 0072006d f501007b 90000fe2 4fcdf020 50102d58 cf1b0f80
59029000 0fe20fcd f0000f1b c04de8e8 0f59e8e8 0819ff20 c413f8a3 50102d78 2048f020 af8ff576 0f37e8e8 0f99ffe2 0f9904c0
000d0fe2 cfcdf781 ffb70fa6 50102d98 000de8e8 08590000 0f2d0000 0fc8e8e8 f5760f37 080020bb ffe2c04d 00000340 50102db8
007b04c0 000d0fe2 cfcd0fa6 f020af8f 0f19e8f4 0019800a 9f3fffff ff1bf004 50102dd8 00000240 000dfc79 ff6d8004 0f7fe8f0
00000340 000d806a 08378fe2 c04d8000 50102df8 f020af8f f5760f37 080020bb ffe2c04d 0428f8d5 0fd904c0 000d0fe2 cfcd0fa6
50102e18 f020af8f f5760f37 ffe2c04d fe47ff6d 900004c0 000d0fe2 cfcdf781 ffb78fa6

Advantages: Low memory usage



Different microprocessor variants available with more / less memory at higher / lower cost

Save a few cents per chip by having less memory

x multiple ECUs per car

x millions of cars

= $$$

Advantages and Benefits

- Low memory footprint
  - Cheaper microprocessor variants reduce unit costs in production
- Low execution time overheads
  - Can include more useful functionality without upgrading to more expensive processors
- Analysable behaviour
  - Can guarantee timing behaviour leading to more reliable systems
  - Reduces time spent debugging intermittent timing problems
- Benefits
  - Reduced development, production and warranty costs for car companies and suppliers
  - Competitive market place hence benefits passed to consumers via less expensive and more reliable cars

Impact

- RTOS Deployment
  - RTOS used by most of the world's leading car companies and suppliers
  - In 2015, the total number of deployed copies of the RTOS was over

1 billion

  - This number is increasing at the quite astonishing rate of between 1 and 2 million new ECUs *per week*

  - Profitable business with substantial revenues
  - Sustained a large number of high tech jobs in York for over 15 years

Case Study 3: How long does your real-time software take to run?

- The Worst-Case Execution Time (WCET) problem
  - Finding the longest time that software components can take to run on a microprocessor is an important issue in embedded systems development
  - Overrunning execution time budgets can cause operational and reliability problems or worse
  - WCET needs to be tightly bounded to avoid the need for overprovisioned hardware

Measurement-based WCET analysis

- Research in RTSRG* from early 2000s
  - Set of hybrid and probabilistic techniques developed for WCET analysis
- RapiTime technology
  - Recognises that the best possible model of a processor is the processor itself – hence uses online testing to obtain measurements
  - Recognises that the best way to determine the overall structure of the code is offline analysis
  - Combines static analysis of the structure of the code and measurements of short sub-paths obtained via testing to obtain tight WCET estimates

*By Guiem Bernat, Antoine Colin, and Alan Burns

Spin-out Company

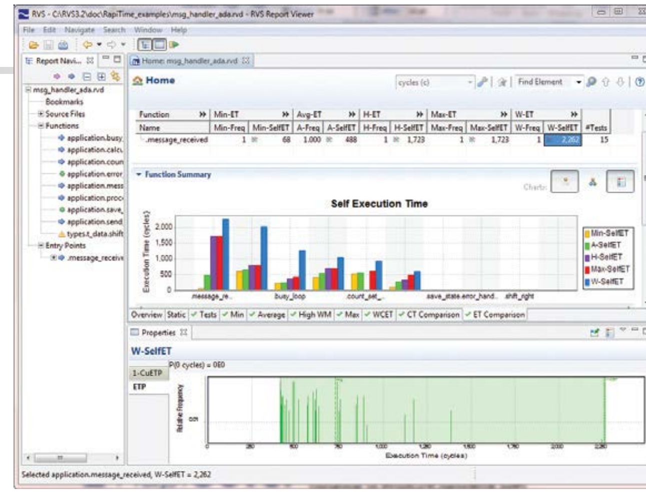
- Origins
  - Guiem Bernat, Ian Broster, Antoine Colin and Rob Davis founded **Rapita Systems Ltd** in 2004.
  - Purpose: to develop **RapiTime** WCET analysis technology for aerospace and automotive applications
- Further developments
  - RapiTime: Extended to support programs written in C++, C, and Ada
  - RapiCover: Code coverage tool
  - RapiTask: Scheduling visualisation
  - Together form RVS (Rapita Verification Suite)
- Funding
  - £200k of investment from Viking Fund + Business Angels in 2005

THE UNIVERSITY of York

RTS York

Exploitation

- Initial prototypes
  - Technology demonstrated on an Audi drive-by-wire system – as part of an EU project



- BAE Systems
  - In 2006 RapiTime used on Hawk Jet Trainer project
  - Identified the 1% of many 100,000s of lines of code that contributed to nearly 1/3rd of the WCET
  - Optimisations to this 1% reduced the WCET by 23%
  - Received BAE Chairman's award for innovation

Impact

- Technology
  - RapiTime Technology deployed on major aerospace and automotive projects in the UK, Europe, Brazil, India, China, and the USA
  - Key customers include leading aerospace companies as well as major automotive suppliers as well as the European Space Agency

- Company
  - Rapita Systems Ltd. is a successful and profitable business that now employs more than 25 people in York

Impact

- Quotes

“*The biggest benefit that RapiTime brought to our development process was just how quickly we could get comprehensive timing measurements from our tests. Not only did we reduce our effort requirements for the testing, but we could use our results in ways that were infeasible before.*”

Engineering Fellow at a major aerospace supplier 2009

“*the main advantage [of using RapiTime] is the possibility to identify software bottlenecks that can be subject to optimisation. Without RapiTime the mandatory code optimisation would have been done without the knowledge of where to concentrate the efforts.*”

Alenia Aermacchi

Summary

- Three Case Studies - one common thread
  - World-class Research from RTSRG
    - Many of the research papers produced are recognised as the seminal ones in the field
    - Some cited more than 500 times
  - Exploitation via a start-up company
  - World-wide impact
    - Products have been adopted and standardised upon by many large companies in the automotive and aerospace industries
    - Created and sustained large numbers of high technology jobs

Questions?

- More info
  - White papers about the Real-Time Systems group Impact Case Studies can be found on my webpage
    http://www-users.cs.york.ac.uk/~robdavis/
  - Videos available soon!