Controller Area Network (CAN) Schedulability Analysis: Refuted, Revisited and Revised

Robert Davis
Real-Time Systems Research Group
University of York

Research by:
Robert Davis, Alan Burns (*University of York*)
Reinder Bril, Johan Lukkien (*Technische Universiteit Eindhoven)*

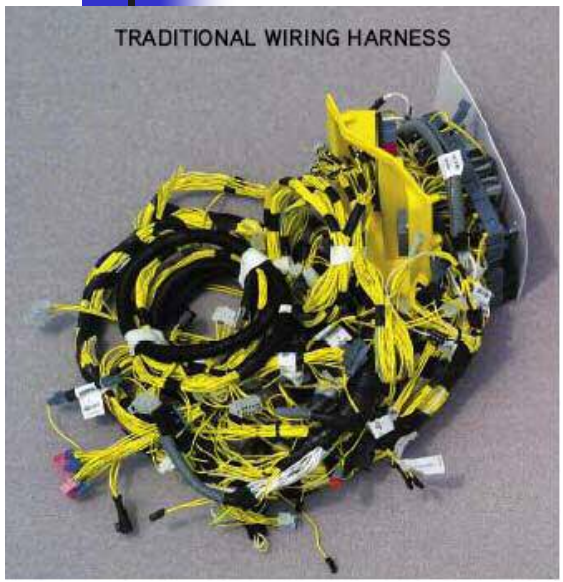
Roadmap

- Controller Area Network (CAN)
  - History, usage
  - Basic protocol
- Schedulability Analysis
  - Highlight a serious flaw in previous analysis of CAN which results in optimistic message response times
  - Revised schedulability analysis addressing the problem
  - Look in detail at circumstances under which the previous analysis fails
- Impact on Deployed Systems
  - Should we expect to see failures?

CAN History

- Controller Area Network (CAN)
  - Simple, robust and efficient serial communications bus for in-vehicle networks
- Developed by BOSCH
  - Starting in 1983 presented at SAE in 1986
  - Standardised by ISO in 1993 (11898)
- First CAN controller chips
  - Intel (82526) and Philips (82C200) in 1987
- First production car using CAN
  - 1991 Mercedes S-class (W140)

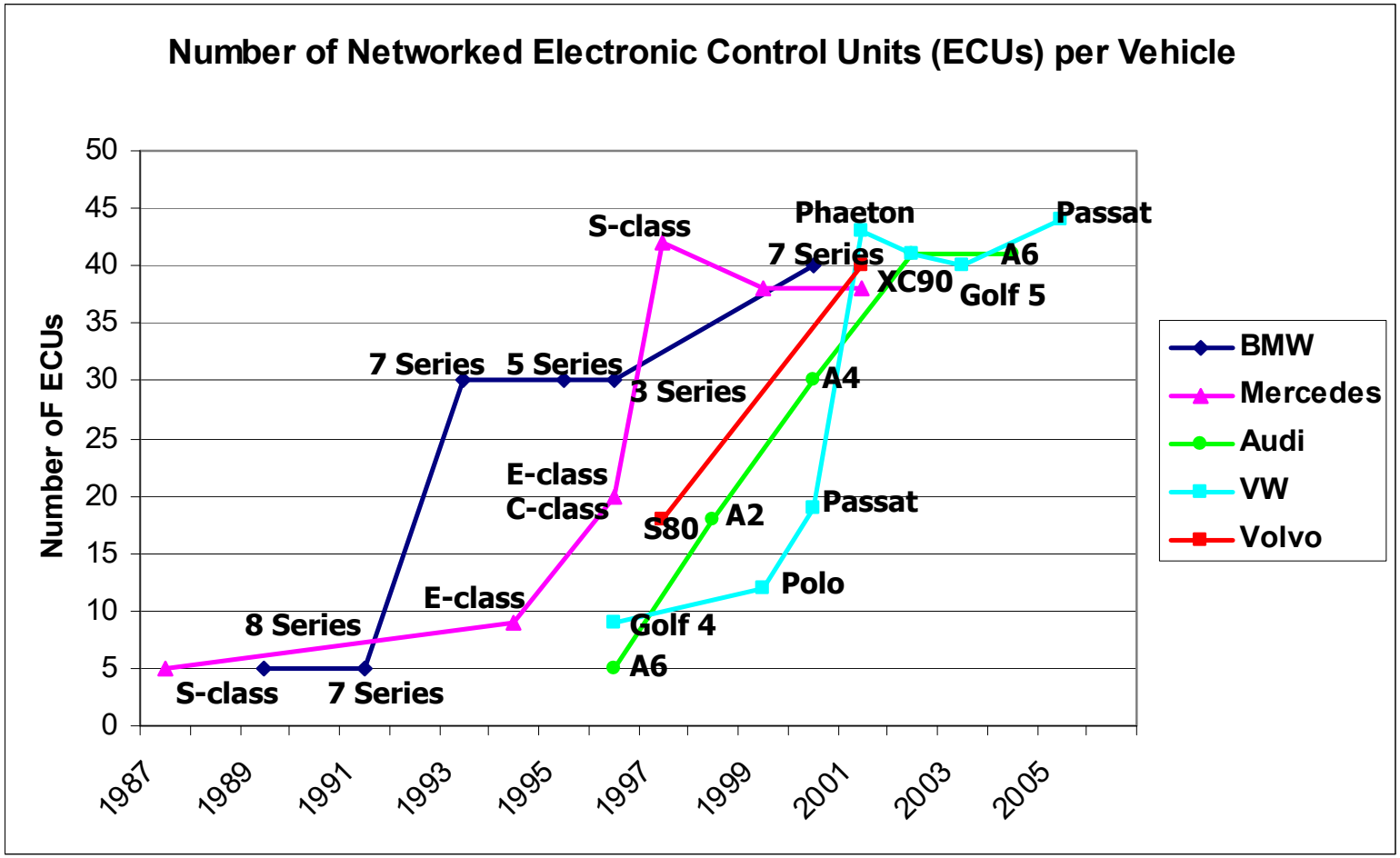
Multiplex v. Point-to-point Wiring



- 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

Increase in Complexity of Automotive Electronics

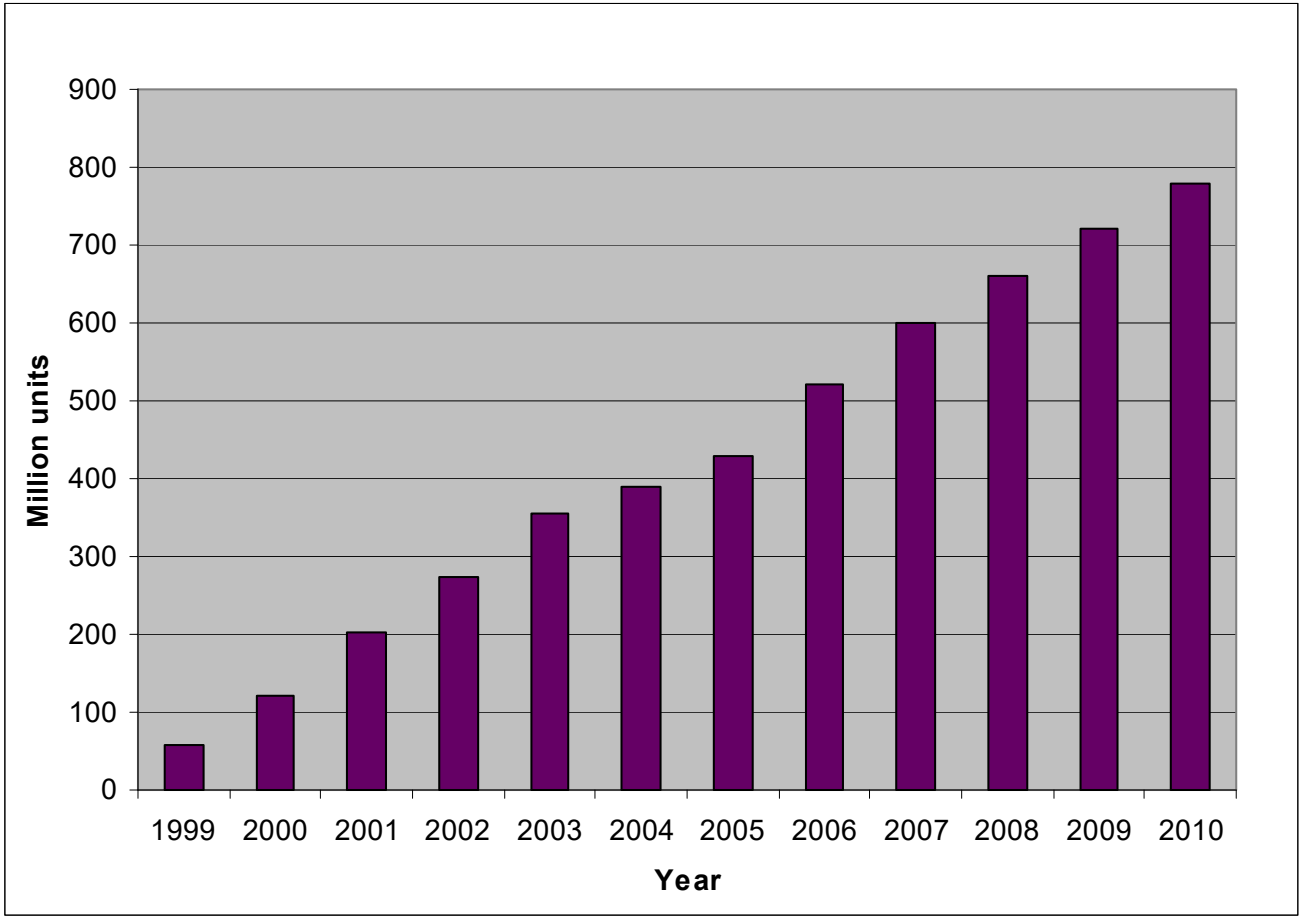


Widespread use of CAN in Automotive

- Other European manufacturers quickly followed Mercedes lead in using CAN

- By 2004
  - 15 different silicon vendors manufacturing over 50 different microprocessor families with on chip CAN capability
    - Analogue Devices, Atmel, Cygnal, Fujitsu, Infineon, Maxim formally Dallas, Microchip, Mitsubishi, Motorola, NEC, Phillips, Renesas, Siemens, Silicon Laboratories, and STMicroelectronics
- In 2008
  - EPA rules for On Board Diagnostics made CAN mandatory for cars and light trucks sold in the US
- Today
  - Almost every new car sold in Europe has at least one CAN bus

CAN Node Sales

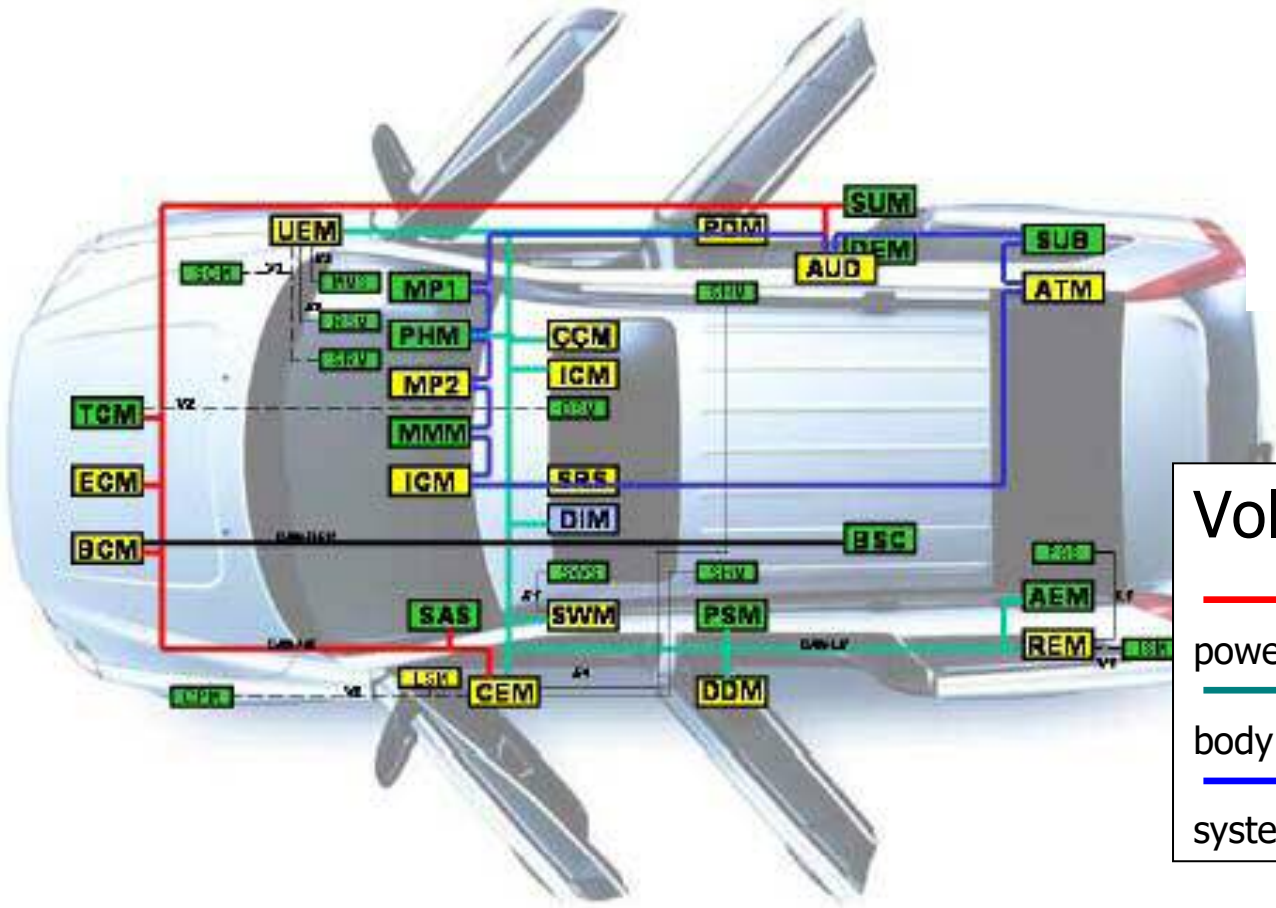


Sales of microprocessors with on chip CAN capability increased from under 50 million in 1999 to over 750 million in 2010

CAN in Automotive

- CAN typically used to provide
  - "High speed" (500 Kbit/sec) network connecting chassis and power train ECUs
    - E.g. transmission control, engine management, ABS etc.
  - Low speed (100-125 Kbit/sec) network(s) connecting body and comfort electronics
    - E.g. door modules, seat modules, climate control etc.
  - Data required by ECUs on different networks
    - typically "gatewayed" between them via a powerful microprocessor connected to both

Volvo XC90 Network Architecture



Volvo XC90 (2001)

500 Kbit/sec CAN bus for power train

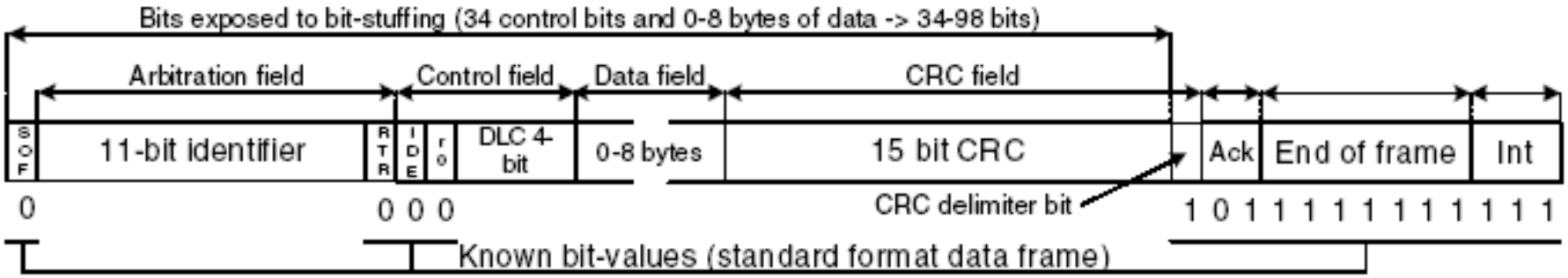
125 Kbit/sec CAN bus for body electronics

MOST (infotainment system)

Information on CAN

- CAN used to communicate *signals* between ECUs
  - Signals typically range from 1 to 16-bits of information
  - 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

CAN Protocol: Data Frame Format



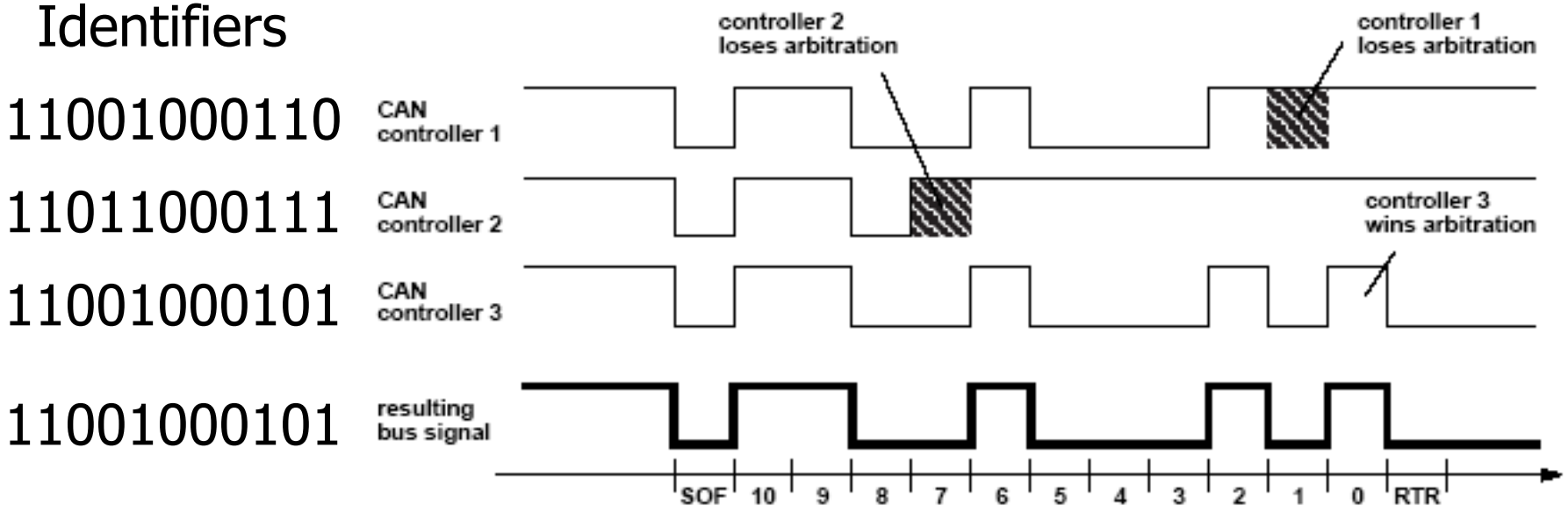
- Start of frame (synchronisation)
- Identifier determines priority for access to bus (11-bit or 29-bit)
- Control field (Data length code)
- 0-8 bytes useful data
- 15-bit CRC
- Acknowledgement field
- End of frame marker
- Inter-frame space (3 bits)

CAN Protocol

- CAN is a multi-master CSMA/CR serial bus
  - Collision resolution is based on priority
  - CAN physical layer supports two states: “0” dominant, “1” recessive
- Message transmission
  - CAN nodes wait for “bus idle” before starting transmission
  - Synchronise on the SOF bit (“0”)
  - Each node starts to transmit the identifier for its highest priority (lowest identifier value) ready message
  - If a node transmits “1” and sees “0” on the bus, then it stops transmitting (lost arbitration)
  - Node that completes transmission of its identifier continues with remainder of its message (wins arbitration)
  - Unique identifiers ensure all other nodes have backed off

CAN Protocol: Message Arbitration

- Message arbitration based on priority



Schedulability Analysis

- CAN resembles single processor fixed priority non-pre-emptive scheduling
  - Messages compete for access to the bus based on priority
  - Effectively a global queue with transmission in priority order
  - Once a message starts transmission it cannot be pre-empted

- Schedulability Analysis for CAN
  - First derived by Ken Tindell in 1994 from earlier work on fixed priority pre-emptive scheduling
    - Calculates worst-case response times of all CAN messages
    - Used to check if all CAN messages meet their deadlines in the worst-case
    - Possible to engineer CAN based systems for timing correctness, rather than "test and hope"

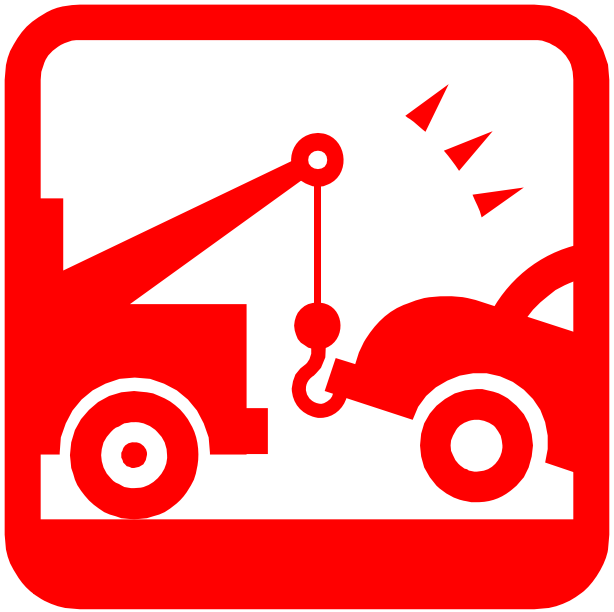
Schedulability Analysis

- Schedulability analysis for CAN
  - Seminal research, appeared in conference proceedings, journal papers, used in teaching...
  - Referenced in over 400 subsequent research papers
  - Lead to 2 PhD Theses
  - In 1995 recognised by Volvo Car Corporation
  - Used in the development of the Volvo S80 (P23)
  - Formed basis of commercial CAN analysis products

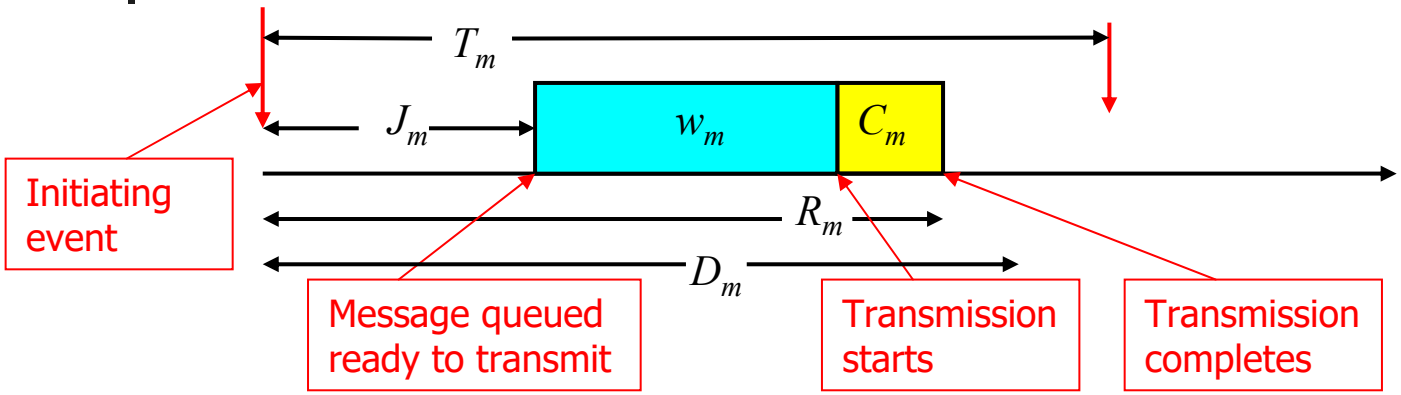
  - Used by many Automotive manufacturers who have built millions of cars with networks analysed using these techniques
  - Enabled increases in network utilisation from 30-40% to typically 70-80%

Unfortunately…

- The original schedulability analysis for CAN is seriously flawed…



Schedulability Analysis: Model



- Each CAN message has a:
  - Unique priority m (identifier)
  - Maximum transmission time C_m
  - Minimum inter-arrival time or period T_m
  - Deadline $D_m <= T_m$
  - Maximum queuing jitter J_m
- Compute:
  - Worst-case queuing delay w_m
  - Worst-case response time
    $$R_m = J_m + w_m + C_m$$
  - Compare with deadline

Schedulability Analysis: TX Time

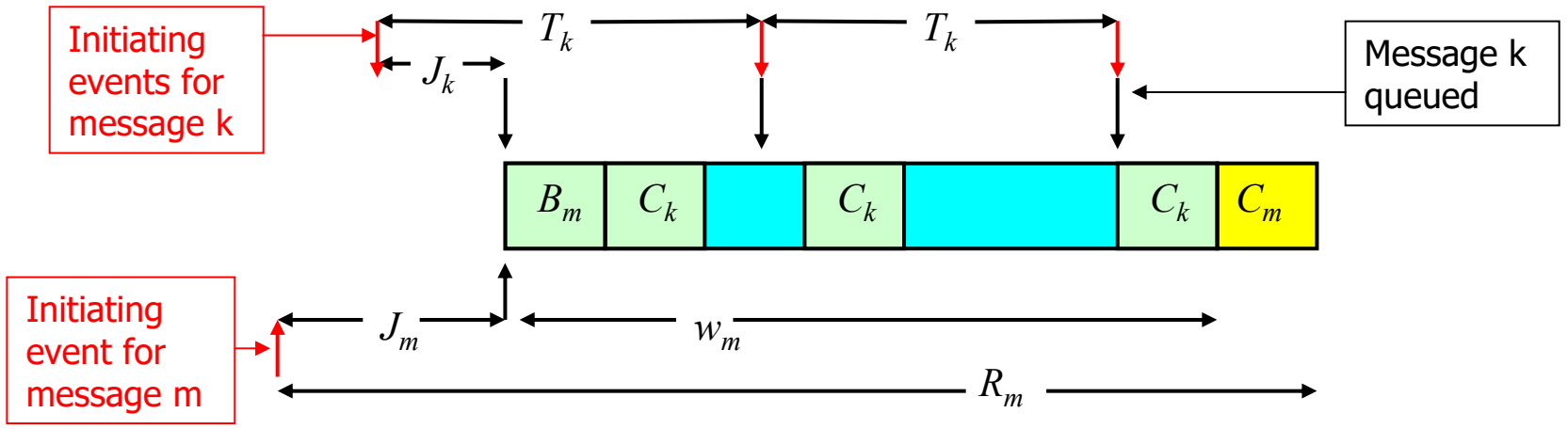
- Maximum transmission time
  - Bit stuffing
    - Bit patterns "000000" and "111111" used to signal errors
    - Transmitter insert 0s and 1s to avoid 6 consecutive bits of same polarity in messages
  - Increases transmission time of message

11-bit identifiers: $C_m = (55 + 10s_m)\tau_{bit}$

29-bit identifiers: $C_m = (80 + 10s_m)\tau_{bit}$

Schedulability Analysis: Equations

- Blocking $B_m = \max_{k \in lp(m)} (C_k)$
- Queuing delay $w_m^{n+1} = B_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{bit}}{T_k} \right\rceil C_k$
- Response time $R_m = J_m + w_m + C_m$

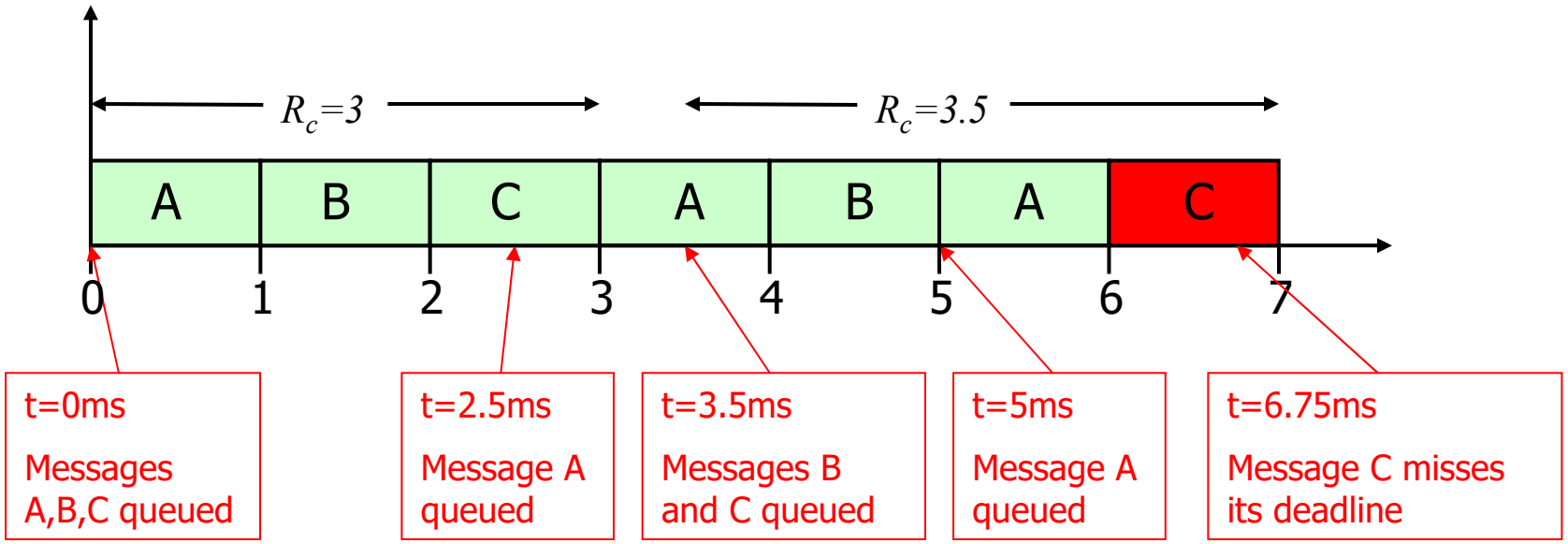


Schedulability Analysis: Example

- 125 Kbit/s bus
- 11-bit identifiers
- 3 messages with 7 data bytes each, max. 125 bits including bit stuffing

Message	Priority	Period	Deadline	TX Time	R
A	1	2.5ms	2.5ms	1ms	2ms
B	2	3.5ms	3.25ms	1ms	3ms
C	3	3.5ms	3.25ms	1ms	3ms

Response time of message C



The original schedulability analysis gives an optimistic response time for message C: 3ms v. 3.5ms

2nd instance of message C misses its deadline

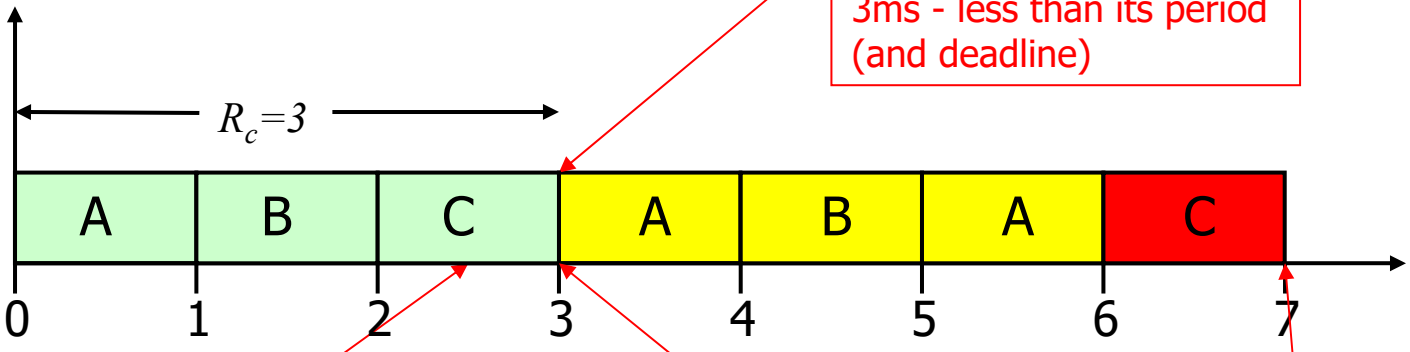
Schedulability Analysis: Example

Message	Priority	Period	Deadline	TX Time	R
A	1	2.5ms	2.5ms	1ms	2ms ✓
B	2	3.5ms	3.25ms	1ms	3ms ✓
C	3	3.5ms	3.25ms	1ms	3ms ✘

- If the periods of messages B and C were also 3.25ms …

  The original analysis would result in the same response times implying a schedulable system with a total bus utilisation of 102%!

What is the flaw in the analysis?



Response time of 1st instance of message C is 3ms - less than its period (and deadline)

BUT transmission of message C is non-pre-emptive and blocks message A, pushing extra interference into next period of C

Busy period at priority of message C does NOT end with transmission of message C

Busy period ends here. Must examine all instances of message C in the busy period to find WCRT

Revised Schedulability Analysis

- Find length of longest busy period for message *m.*
  - *(Busy period includes all instances of message m and higher priority messages queued strictly before the end of the busy period)*

$$t_m^{n+1} = B_m + \sum_{\forall k \in hp(m) \cup m} \left\lceil \frac{t_m^n + J_k}{T_k} \right\rceil C_k$$

  - Starts with $t_m^0 = C_m$
- Number of instances of message *m* ready before end of busy period

$$Q_m = \left\lceil \frac{t_m + J_m}{T_m} \right\rceil$$

Revised Schedulability Analysis

- For each instance q ($q = 0$ *to* $Q_m - 1$) of message *m* in the busy period, compute the longest time from the start of the busy period to that instance starting transmission:

$$w_m^{n+1}(q) = B_m + qC_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{bit}}{T_k} \right\rceil C_k$$

- Response time of instance q of message *m*:

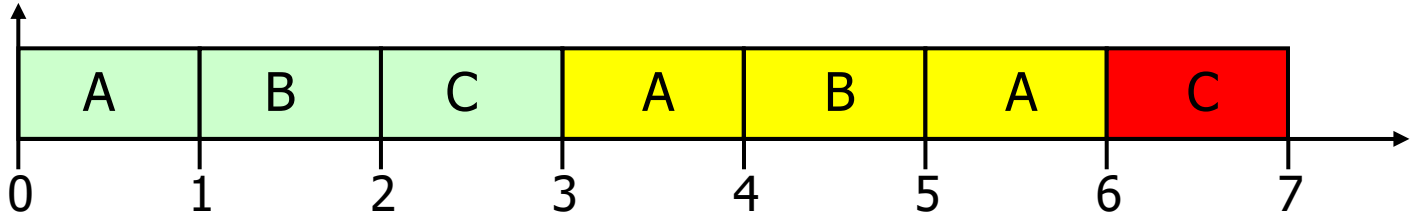
$$R_m(q) = J_m + w_m(q) - qT_m + C_m$$

- Worst-case response time of message *m*:

$$R_m = \max_{q=0..Q_m-1} (R_m(q))$$

Example Revisited

Message	Priority	Period	Deadline	TX Time
A	1	2.5ms	2.5ms	1ms
B	2	3.5ms	3.25ms	1ms
C	3	3.5ms	3.25ms	1ms



Message	Busy period	Q	R(0)	R(1)	R max
A	2ms	1	2ms	-	2ms ✓
B	5ms	2	3ms	1.5ms	3ms ✓
C	7ms	2	3ms	3.5ms	3.5ms ✓

Sufficient Schedulability Test #1

- 1st invocation of message *m:*

$$w_m^{n+1} = B_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{bit}}{T_k} \right\rceil C_k$$

- For messages with $D_m <= T_m$ and schedulable 1st instance, then a pessimistic view of 2nd and subsequent instances is a *critical instant* with indirect or push-through blocking of C_m from the previous instance of message *m*

$$w_m^{n+1} = C_m + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{bit}}{T_k} \right\rceil C_k$$

- Combined:

$$w_m^{n+1} = \max(B_m, C_m) + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{bit}}{T_k} \right\rceil C_k$$

Sufficient Schedulability Test #2

- Let maximum possible transmission time of the longest possible message on the network be: C^{MAX}
- Always assume this as the blocking factor

$$B^{MAX} = C^{MAX}$$

- As $B^{MAX} \geq \max(B_m, C_m)$
- Simple sufficient schedulability test

$$w_m^{n+1} = B^{MAX} + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{bit}}{T_k} \right\rceil C_k$$

When does existing analysis fail?

- Can the original analysis give faulty guarantees to messages of any priority?

  1st and 2nd highest priority messages ok

  3rd and lower can get faulty guarantees

- If the bus utilization is low, can the original analysis still result in optimistic response times?

  Yes

Number of messages	Breakdown Utilisation
5	21.4%
10	9.2%
25	3.4%
100	0.82%

When does existing analysis fail?

- Do error models give sufficient margin for error to account for flaws in the analysis?
- Yes: If a system is deemed schedulable by the existing analysis, including a reasonable error model, then it is actually schedulable when there are no errors on the bus
- Does the omission of maximum length diagnostic messages during normal operation mean that the deadlines of the remaining messages will be met?
- Yes: other messages will meet their deadlines. In normal operation, with no diagnostic messages, there can be no problem due to the flawed analysis

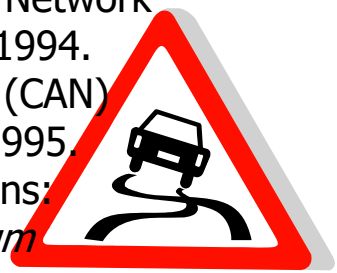
When does existing analysis fail?

- Which message guarantees can we be sure are not at risk?
- Messages are not at risk if there is at least one lower priority message with the same or longer transmission time
- If all messages are the same length, then only the lowest priority message is at risk

Implications and Recommendations

- CAN schedulability analysis tools
  - Need to be checked. Is the analysis implemented correct?
  - Sufficient schedulability tests provide a simple fix
- Research
  - Authors who have cited the original CAN schedulability analysis papers are encouraged to check the implications on their own work

[1] K.W. Tindell and A. Burns. "Guaranteeing message latencies on Controller Area Network (CAN)", In *Proceedings of 1st International CAN Conference*, pp. 1-11, September 1994.
[2] K.W. Tindell, A. Burns, and A. J. Wellings. "Calculating Controller Area Network (CAN) message response times". *Control Engineering Practice*, 3(8): 1163-1169, August 1995.
[3] K.W. Tindell, H. Hansson, and A.J. Wellings. "Analysing real-time communications: Controller Area Network (CAN)". In *Proceedings 15th Real-Time Systems Symposium (RTSS'94)*, pp. 259-263. IEEE Computer Society Press, 1994.



Impact on deployed CAN systems

- Will your car still work?
  - Typical systems have 8 data byte diagnostic messages:
    no problems in normal operation
  - Analysis used allows for errors:
    no issues when errors not present
  - Typically all messages have 8 data bytes:
    only lowest priority message could be affected
  - Deadline failures require worst-case phasing, worst-case bit stuffing and errors on the bus:
    very low probability of occurrence
  - Systems designed to be resilient to some messages missing their deadlines and simpler problems such as intermittent wiring faults

Commercial CAN Analysis Tools

- Volcano Network Architect
  - Commercial CAN schedulability analysis product
  - Uses a simple sufficient schedulability test, assuming maximum blocking factor irrespective of message priorities / number of data bytes

$$w_m^{n+1} = B^{MAX} + \sum_{\forall k \in hp(m)} \left\lceil \frac{w_m^n + J_k + \tau_{bit}}{T_k} \right\rceil C_k$$

  - Pessimistic but correct upper bound on message worst-case response times
  - Used to analyse CAN systems for Volvo S80, S/V/XC 70, S40, V50, XC90 and many other cars from other manufacturers
  - **By 2005 over 20 million cars with an average 20 ECUs each developed using Volcano technology**

Journal paper

R.I.Davis, A. Burns, R.J. Bril, and J.J. Lukkien,
"Controller Area Network (CAN) Schedulability Analysis: Refuted, Revisited and Revised". *Real-Time Systems*, Volume 35, Number 3, pp. 239-272, April 2007.
DOI: 10.1007/s11241-007-9012-7

- Open access – freely available
- 138 citations (~36 per year)
- End of 2010 it was the most downloaded paper from the journal, *Real-Time Systems*