Controller Area Network (CAN): Response Time Analysis with Offsets

Patrick Meumeu Yomsi, Dominique Bertrand, Nicolas Navet, Robert I. Davis

IEEE WFCS 2012
Outline

1. Design of (automotive) CAN networks today
2. Worst-case response time analysis with offsets
3. Performance evaluation
### Automotive CAN: the early days (1/2)

<table>
<thead>
<tr>
<th>Priority</th>
<th>Sender node</th>
<th>DLC</th>
<th>Period (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Engine Controller</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>Wheel angle sensor</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>Engine Controller</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>AGB</td>
<td>2</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>ABS</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>6</td>
<td>ABS</td>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>7</td>
<td>ABS</td>
<td>4</td>
<td>15</td>
</tr>
<tr>
<td>8</td>
<td>Body gateway</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>9</td>
<td>undisclosed</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td>Engine Controller</td>
<td>7</td>
<td>100</td>
</tr>
<tr>
<td>11</td>
<td>AGB</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>12</td>
<td>ABS</td>
<td>1</td>
<td>100</td>
</tr>
</tbody>
</table>

Early CAN project at PSA (1996, see [1])

250kbit/s
Automotive CAN: the early days (2/2)

Worst-case latencies (=response times) are less than 5.5 ms
NETCAR-Analyzer screenshot
Proliferation of ECUs and buses

# ECUs and buses in some PSA projects between 2000 and 2010 [2]

Up to 5 CAN interconnected by gateways
Today’s set of messages

- **Size**: Up to 20 nodes and > 100 frames
- **Bus-rate**: 250 or 500kbits
- **Load**: > 50%, sometimes 60% or more …
- **Max latencies**: 5ms or less
- **Gateways**: CAN/CAN or CAN/FLEXRAY induce delays and bursty traffic.
- **Complex traffic model**: aperiodic (w/wo exclusion time), Autosar mixed transmission mode, segmented messages, download session, etc …

NETCARBENCH is a GPL licensed software to generate “realistic” and non confidential CAN message sets according to a set of user-defined parameters. Available at www.netcarbench.org

“easy” integration for the OEM till 35-40% - precise performance evaluation needed beyond [4]
Higher load level calls for

1. More constraining specifications to the suppliers / or conservative assumptions → a single node can jeopardize the whole system

2. Thorough use of Validation & Verification techniques:
   - simulation, worst-case analysis and trace inspection
   - none of them alone is sufficient!

Know-how, embedded software, verification techniques, and tool support have progressed to a point where **highly loaded CAN networks - yet safe** are possible.
Simulation
fault-injection

RTaW-Sim & Netcar-Analyzer freely downloadable

“Exploratory” set of messages
“Project” set of messages
Communication traces

Trace analysis for:
- Error model
- Aperiodic traffic model
- Real periods, offsets, clock drifts, functioning modes, bit-stuffing, etc
- Communication stack quality
- ...

Worst-case analysis
Offset optimization

Tools & techniques complementarities the case at RTaW [4]

RTaW-Sim
RTaW-TraceInspector
NETCAR-Analyzer
Different sets of messages along the development process: our view

**“Exploratory” sets of message**
- Virtual sets of messages generated from real sets of messages
- Architecture design
- Technological choices
- “Coarse grained” verification
- Incremental design possibility
- GPL tool Netcarbench

**“Project” sets of messages**
- Configuration: offsets, ID, etc.
- “Fine grained” verification
- Evolutions: adding frames, ECUs

**Communication traces**
- Verifying specification respect
- Impact of non-conformance
2

Optimizing CAN networks
What levers do we have?
Automotive CAN communication stack: a simplified view

ECU

Middleware

Frame-packing task

5ms

Waiting queue:
- FIFO
- Highest Priority First
- OEM specific

CAN Controller

9 6 8

buffer Tx

CAN Bus
Optimizing CAN: meeting performance and robustness constraints at higher load

An industrial requirement

- Reduce architecture complexity, HW costs & weight, consumption and emission
- Avoid industrial risks and costs of new technologies
- Incremental design / better performances

How?

1. Keep amount of data transmitted minimum! → better identification and traceability of timing constraints
2. Synchronize producing tasks with communication tasks
3. Desynchronize frames by using offsets [3,4]
4. Assign priorities according to deadlines
5. Re-consider frame packing [12]
6. Optimize communication stacks so as to remove all “departure” from the ideal CAN behavior
**Scheduling frames with offsets?**

**Principle:** desynchronize transmissions to avoid load peaks

**Offsets algorithms:** DOA [14], least-loaded [3], SOA [RTaW], local optimization (GA), etc.
Offsets algorithm applied on a typical body network [3]

Worst-case latencies on a 125 kbit/s body network

WCRT without offset
WCRT with random offsets (average value)
WCRT with offsets least-loaded algorithm [3]
WCRT lower bound

Worst-case latencies on a 125 kbit/s body network
Analyses for safety critical systems: simple, peer-reviewed and documented

✓ Flawed analyses are dangerous in safety critical systems but (fine-grained) analyses are complex and error prone. Remember “CAN analysis refuted, revisited, etc” [6] ?!

✓ Implemented analysis have to make simplifications esp. in a heterogeneous systems (and tools do not document that well)

✓ Solutions?
  • peer-review of the WCRT analyses is needed
  • coarse-grained / conservative but simple as far as possible: e.g., [5,6] vs [9]
  • no black-box software: documentation of implemented analyses and underlying hypotheses
  • cross-validation between tools on benchmarks
3

Response time analysis with offsets
Contribution: exact response time analysis with offsets

- Adaptation of Palencia & Harbour work to non-preemptive scheduling [15]
- Periodic and sporadic frames / with and without jitters / arbitrary jitters and deadlines
- Complexity is exponential but usable for medium-size systems with typical automotive characteristic (e.g., non-arbitrary periods)
- Performance evaluation with jitters shows that offsets bring major performance boost
- Sound basis for optimization and non-ideal CAN behavior
System model

- Stations are not synchronized (no global clock)
- Ideal CAN nodes
- Each CAN message has a:
  - Unique sending node
  - Unique priority $m$
  - Maximum transmission time $C_m$
  - Minimum inter-arrival time or period $T_m$
  - An offset $\Phi_m$ : first activation time after origin (multiple of the communication task period if any)
  - Arbitrary deadline $D_m$
  - Arbitrary max. queuing jitter $J_m$
1. On each station, model the outgoing traffic as the minimum number of transactions.

2. Identify the smallest set of scenarios that must contain to the worst-case response time for a specific CAN ID.

3. Compute the max. response time for the current ID on each identified scenario.
Step 1: from frames to transactions

A transaction captures all the periodic traffic sent by a node.

$T_1 = 4, J_1 = 2, \Phi_1 = 0$

$T_2 = 6, J_2 = 7, \Phi_1 = 1$

All periodic frames $a$ of node form a single transaction, each sporadic frame needs its own transaction.
Step 2: set of scenarios to examine

What is the worst scenario for of priority $p$?

Theorem (adapted from [15]): the worst-case scenario for the frame of priority $p$ belongs to the set of scenarios in which one frame with a priority higher than or equal to $p$ in each transaction is released simultaneously after having experienced its maximum jitter.

Simple optimization: reconstruct smaller transactions that only include frames with priority higher than or equal to $p$. 
Step 3: response time in a specific scenario – source of interferences

1. Lower priority frame that has started transmission

2. Instances of the same priority

3. Higher priority frames: due to jitters, they might be several instances released simultaneously at the start of the busy period $t_c$

$$K^k_i(t) \overset{\text{def}}{=} \left\lfloor \frac{J^k_i + \varphi^k_i(\phi^i)}{T^k_i} \right\rfloor + \left\lfloor \frac{t - \varphi^k_i(\phi^i)}{T^k_i} \right\rfloor + 1$$

- max number of instances that can accumulate at $t_c$
- max number of instances arriving after $t_c$
4

Experimental results
Setup: medium-size automotive body networks

- Generated using NETCARBENCH
- 8-12 ECUs - 250Kbit/s - load from 38 to 42%
- One station transmits 20% of the traffic
- Message periods: 20, 50, 100, 200 or 500 ms
- Deadlines equal to periods
- Queuing jitter: with (below period for 10% of the frames) and without
- Offset algorithm: DOA [14]
- Priorities: transmission deadline monotonic order (TDMPO) i.e. D-J order
Focus on a single (typical) configuration – WCRT and max from simulation

- Gain with offsets: 68% for the lowest priority
- Difference WCRT vs max on random simulations:
  - avg: 25%
  - max: 45%
Results over 100 random configurations – average WCRT gain over all frames

Gain with offsets

without jitter
Avg: 47%
Max: 57%

with jitter (10%)
Avg: 42%
Max: 52%
Results over 100 random configurations – average WCRT gain over the 20% lowest priority frames

- Gain with offsets
  - without jitter
    - Avg: 65%
    - Max: 74%
  - with jitter (10%)
    - Avg: 59%
    - Max: 70%

![Graph showing gain in % vs experiment index with and without offsets and jitter.](image)
Future work

- [ongoing] Optimization so as to make exact analysis usable on arbitrary large CAN networks
- [ongoing] Extension to heterogeneous networks with non-ideal CAN behavior
- Extension to segmented messages
References
References

(1/2)


References (2/2)


