Automotive embedded systems: some research challenges

Nicolas Navet

ETR’2009
03/09/2009
Electronics is the driving force of innovation

- 90% of new functions use software
- Electronics: 40% of total costs
- Huge complexity! 70 ECUs, 2000 signals, 6 networks, multi-layered run-time environment (AUTOSAR), multi-source software, multi-core CPUs, etc

Strong costs, safety, reliability, time-to-market, reusability, legal constraints!
Many issues in the design of E/E systems are not strictly technical!
Eg. Key issues in architecture development at Volvo in paper ref[2]

- Lack of background in E/E at management level (often mechanical background)
- Influence of E/E architecture wrt to business value? Lacks long term strategy
- Lack of clear strategy between in-house and externalized developments
- Technical parameters are regarded as less important than cost for supplier / components selection
Key issues in architecture development at Volvo in paper ref[2] (2/2)

- How to share architecture/sub-systems between several brands/models with different constraints/objectives?

- Sub-optimal solutions for each component / function

- Architectural decisions often:
  - are made on experience / gut feeling (poor tool support)
  - Lacks well-accepted process
Where to tackle the problem from a technical point of view?

- **Design**: model functional and non-functional features
  - software components, MDD, etc

- **Validation / Analysis**: dependability, (end-to-end) response time, memory consumption (e.g. buffers), deadlocks, etc

- **Synthesis**: remove unused features, mapping of components to runtime objects (ECU/Tasks), setting runtime parameters (priorities, offsets, etc)

- **Runtime system**: OS, protocols, drivers, NM, diagnostics, etc
Validation is a key activity!
Personal view on the developments

Mostly ahead of us!

Talk part 2

Talk part 1

Probabilistic analysis (sub-system)

« Worst-case » deterministic analysis (sub-system)

« Smart » monitoring tools

Simulation tools (co-simulation, HIL)

« Worst-case » deterministic analysis
system level

Probabilistic analysis
system level

« correctness by construct »
and optimal synthesis

© 2009 RealTime-at-Work / INRIA – ETR'09 - 6
Part 1 - probabilistic framework for schedulability analysis: illustration on the aperiodic traffic on the bus
(see paper ref[5])
Probabilistic analysis is needed

- Systems are not designed for the worst-case (provided it is rare enough!)
- Reliability/Safety are naturally expressed and assessed in terms of probability (e.g. < $10^{-9}$ per hour)
- Deterministic assumptions are sometimes unrealistic or too pessimistic, e.g.:
  - Worst-Case Execution Time on modern platforms,
  - Aperiodic activities: ISR, frame reception,
  - ...
- Faults/errors are not deterministic (and better modeled probabilistically)
Accounting for the aperiodic traffic

- Transmission patterns can hardly be characterized: purely aperiodic, mixed periodic/aperiodic, etc
- Aperiodic frames do jeopardize RT constraints
- Few approaches in the litterature:
  - deterministic approaches, such as sporadic, generally lead to unusable results (e.g., $\rho > 1$)
  - Average case probabilistic approach not suited to dependability-constrained systems
  - Probabilistic approaches with safety adjustable level, see paper ref[6] and ref[7]
Approach advocated here

1) Measurements / data cleaning
2) Modeling aperiodic traffic arrival process
3) Deriving aperiodic Work Arrival Process (WAF)
4) Integrating aperiodic WAF into schedulability analysis
Data trace analysis

y: aperiodic interarrival times – x: index of interarrivals

Approximate because what is seen on the bus is not the actual arrival process at ECU level! can be handled.
Question: are interarrival times i.i.d.? 

Use of BDS test for non-linear dependencies 

Periodic frame?
Distribution fitting for aperiodic interarrival: 3 candidates here

MLE adjusted parameters

Kolmo. Smi. and $\chi^2$ tests to confirm visual impression
Captured data trace VS random trace generated with MLE-fitted Weibull

Real data trace

Simulated data trace

© 2009 RealTime-at-Work / INRIA – ETR’09 - 14
Deriving the aperiodic WAF

- \( S(t) \): aperiodic WAF
- \( X(t) \): stochastic process which counts the number of aperiodic frames in time interval \( t \)
- "smallest" \( S(t) \) such that the probability of \( X(t) \) being larger than or equal to \( S(t) \) is lower than a threshold \( \alpha \)

\[
\hat{S}(t) = \min\{S(t) \mid Pr[X(t) \geq S(t)] \leq \alpha\}
\]

By simulation, numerical approximation or analysis (simplest cases such as exp.)

Design choice: e.g., \( 10^{-9} \)
Aperiodic WAF depends on the underlying interarrival distribution.

Same average intensity.
Case-study on a typical body network

- Body network benchmark generated using GPL-licensed Netcarbench
- Characteristics:
  - 125kbps, 16 ECUs, 105 CAN frames with deadlines equal to periods and 1 to 8 bytes of data.
  - Total periodic load is equal to 35.47%
- NETCAR-Analyzer for WCRT computation
- 3% aperiodic traffic
- 7 byte aperiodic frames
- $\alpha = 10^{-4}$
Worst-case response times with/out aperiodic traffic (3%)

13 frames with T=100ms add delays
Conclusion - part 1

- Chosen dependability requirements are met while pessimism kept to minimum:
  - Practical approach
  - Real data are required
  - Can be extended to the non i.i.d. case (not needed here)

- What is needed now is a system level approach that
  - Can handle arbitrary activation processes
  - goes beyond the i.i.d. case (for dependability)
Part 2 – optimized synthesis, the case of frame scheduling on CAN
(see paper ref[8])
Optimizing the use of resources is becoming an industrial requirement

- Reasons for optimizing (i.e., less hardware):
  - Complexity of the architectures (protocols, wiring, ECUs, gateways, etc)
  - Hardware costs / weight, room, fuel consumption, etc
  - Need for incremental design
  - Industrial risk and time to master new technologies
  - Performances (sometimes):
    - Ex1: A 60% loaded CAN network may be more efficient that two 35% networks interconnected by a gateway
    - Ex2: cost of communication in distributed functions
  - Limits of current technologies (CPU frequency w/o fan),
  - Etc ...
Scheduling frames with offsets ?!

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

**Algorithms** to decide offsets are based on arithmetical properties of the periods and size of the frame
System model (1/2)

Performance metric: worst-case response time
System model (2/2)

- The offset of a message stream is the time at which the transmission request of the first frame is issued.

- Complexity: best choosing the offsets is exponential in the task periods → approximate solutions.

- Middleware task imposes a certain granularity.

- Without ECU synchronisation, offsets are local to ECUs.
But task scheduling has to be adapted...

- In addition, avoiding consecutive frame constructions on an ECU allows to reduce latency.
Simple offsets Algorithm (1/3)

- **Ideas:**
  - assign offsets in the order of the transmission frequencies
  - release of the first frame is as far as possible from adjacent frames
  - identify “least loaded interval”
- **Ex:** \( f_1=(T_1=10), f_2=(T_2=20), f_3(T_3=20) \)

<table>
<thead>
<tr>
<th>Time</th>
<th>0</th>
<th>2</th>
<th>4</th>
<th>6</th>
<th>8</th>
<th>10</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>18</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame</td>
<td>( f_{1,1} )</td>
<td>( f_{2,1} )</td>
<td>( f_{1,2} )</td>
<td>( f_{3,1} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Offsets Algorithm applied on a typical body network

![Graph showing WCRT with and without offsets]

- WCRT without offset
- WCRT with offsets (algorithm of the paper)

- 21 ms
- 65 ms

CAN frames sorted by decreasing priority order
Offsets Algorithm – industrial needs

- Low complexity and efficient as is but further improvements possible:
  - add frame(s) / ECU(s) to an existing design
  - user defined criteria: optimize last 10 frames, a specific frame,
  - take priorities on the bus into account
  - optimization algorithms: tabu search, hill climbing, genetic algorithms
- ...
Efficiency of offsets: some insight (1/2)

> Almost a straight line, suggests that the algorithm is near-optimal

Work = time to transmit the CAN frames sent by the stations
Efficiency of offsets: some insight (2/2)

- A larger workload waiting for transmission implies larger response times for the low priority frames.
Computing worst-case response times with offsets
Computing frame worst-case response time with offsets

Requirements:
- handle 100+ frames
- very fast execution times
- ≠ waiting queue policy at the microcontroller level
- limited number of transmission buffers
WCRT : State of the art

- **Scientific literature:**
  - Complexity is exponential
  - No schedulability analysis with offsets in the distributed non-preemptive case
  - Offsets in the preemptive case: not suited for > 10-20 tasks
  - WCRT without offsets: infinite number of Tx buffers and no queue at the microcontroller level

- **RTaW software:** NETCAR-Analyzer
Performance evaluation:

- Experimental Setup
- WCRT of the frames wrt random offsets and lower bound
- WCRT reduction ratio for chassis and body networks
- Load increase: add new ECUs / add more traffic
## Experimental Setup

- **Body and chassis networks**

<table>
<thead>
<tr>
<th>Network</th>
<th>#ECUs</th>
<th>#Messages</th>
<th>Bandwidth</th>
<th>Frame periods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Body</td>
<td>15-20</td>
<td>≈ 70</td>
<td>125Kbit/s</td>
<td>50ms-2s</td>
</tr>
<tr>
<td>Chassis</td>
<td>5-15</td>
<td>≈ 60</td>
<td>500Kbit/s</td>
<td>10ms-1s</td>
</tr>
</tbody>
</table>

With / without load concentration: one ECU generates 30% of the load

- **Set of frames generated with NETCARBENCH**
Offsets in practice: large response time improvements (1/2)

![Graph showing time improvements with different WCRT scenarios]

- WCRT without offset
- WCRT with random offsets (average value)
- WCRT with offsets (algorithm of the paper)
- WCRT lower bound

CAN frames sorted by decreasing priority order

65 ms
32
21
17

© 2009 RealTime-at-Work / INRIA – ETR'09 - 36
WCRT Reduction Ratio

- Results are even better with loaded stations
Offsets allow higher network loads

Typically: WCRT at 60% with offsets \(\approx\) WCRT at 30% without offsets
Partial offset usage

WCRT without offset
WCRT with offsets on the most loaded station
WCRT with offsets on the 4 most loaded stations
WCRT with offsets on all stations

65 ms
42
34
17

WCRT in ms
CAN frames sorted by decreasing priority order
Conclusions on offsets

- Offsets provide a cost-effective short-term solution to postpone multiple CANs and FlexRay.
- Tradeoff between Event and Time Triggered ET CAN CAN with offsets TT-CAN
  
  + Complexity
  + Determinism

- Further large improvements are possible by synchronizing the ECUs...
References
References (1/2)

Automotive Embedded Systems - General


Dependability / probabilistic framework


Scheduling frame with offsets on CAN

Questions / feedback?

Please get in touch at:
nicolas.navet@realtimeatwork.com

http://www.realtimeatwork.com