Analysis and Optimization of Message Acceptance Filter Configurations for Controller Area Network (CAN)

Florian Pölzlbauer *
Robert I. Davis
Iain Bate
Outline

- Motivation: Automotive System Design
- Controller Area Network (CAN) Protocol

- CAN Message Acceptance Filtering – how it works
- Measuring Filtering Quality
- Optimal Filter Configuration

- Evaluation
Motivation

System & Responsibilities
- many ECUs with dedicated functionality; developed by Tier1-supplier; (SW+HW)
- data-exchange between ECUs via networks (LIN, CAN, FlexRay, Ethernet)
- network design & system integration by OEM

Challenges
- cost pressure of high-volume ECUs
- constrained HW-resources (CPU clock, memory)
- efficient resource usage needed
**Controller Area Network (CAN)**

- asynchronous, multi-master, broadcast, serial communications bus
- each message uniquely identified by its **ID**, which also determines priority during arbitration phase

<table>
<thead>
<tr>
<th>ID</th>
<th>payload</th>
</tr>
</thead>
</table>

- once idle, priority-based bus arbitration: highest priority (i.e. lowest ID) wins
- non-preemptive transmission of message

**sCHEDULABILITY ANALYSIS (response time analysis)**

- ID: 11 or 29 bits
- payload: 0… 8 bytes (CAN)
  - 0…64 bytes (CAN-FD)

**Message specification:**
- \( s \): payload size
- \( T \): period
- \( D \): deadline
- \( ID \)-format (11 or 29 bit)

\[
R_m = J_m + W_m + C_m
\]
\[
R_m \leq D_m
\]
\[
C^{11}_m = (55 + 10 \cdot s_m) \cdot \tau_{bit}
\]
\[
C^{29}_m = (80 + 10 \cdot s_m) \cdot \tau_{bit}
\]
\[
w_{m+1}^e = B + \sum_{k \in\text{can}(m)} \left[ \frac{w_m^n + J_k + \tau_{bit}}{T_k} \right] \cdot C_k
\]

\[
B = \max \{ C_m \} 
\]
Receiving CAN Messages

- each node receives every message (i.e. broadcast bus)
- node looks at message-ID to determine if message is relevant
- if yes: process message
- if not: discard → undesired message receive interrupts

- goal: minimize undesired message receive interrupt load
- solution: HW-based **acceptance filtering** (widely available)

- question: find an optimal filter-configuration
  (undesired RX interrupts → min.)
Research Question & Contribution

How to measure filtering quality of given filter configuration?
- assessment method & quality metric

How to design an optimal filter configuration?
- design optimization methodology
Message Acceptance Filter: how it works

2 registers (per receive buffer):
- mask: which digits are checked
- tag: which digit-values are accepted

Mask: 111 1111 0011
Tag: 000 1100 0000

ID: 000 0000 0100 (4) blocked
ID: 000 1100 1000 (200) pass

- if message passes HW-filter, it is put into receive buffer, and raises receive interrupt
- if message is blocked, nothing happens
Message Acceptance Filter: how it works

- we use “abstract” notation (0,1,x)
  x … “don’t care”

- filters implemented in HW-logic
  inside CAN-controller → no CPU load

- Most CAN-controllers: mask + tag per buffer
- Some CAN-controllers: “shared mask”
  - tag per buffer
  - mask applied to N buffers

mask : 111 1111 0011
tag : 000 1100 0000
------------------------------------------
filter : 000 1100 xx00

e.g.: National’s CR16
15 buffers
1 mask: buffer[0]
1 mask: buffer[1..14]
How to measure Filtering Quality of given Filter Configuration
Filtering Quality

Given

- $M^{\text{all}}$: set of all broadcasted messages (ID, period)
- $M^{\text{des}}$: subset of messages which shall be received by node
- $F$: filter configuration (f filter-patterns)

Calc.

$$M^{\text{pass}}$$

$$M^{\text{block}} = M^{\text{all}} \setminus M^{\text{pass}}$$

$$M^{UB} = M^{\text{block}} \cap M^{\text{des}}$$  
UB … unintended block

$$M^{UP} = M^{\text{pass}} \setminus M^{\text{des}}$$  
UP … unintended pass
Filtering Quality

**Classification**

- in-feasible \( M^{UB} \neq \{\} \) … some desired messages are blocked
- feasible \( M^{UB} = \{\} \) … all desired messages pass
  - perfect \( M^{UP} = \{\} \) … only desired messages pass

**Quality**

\[
load^{UP} = \frac{M^{UP}}{\text{sec}.} = \sum_{M^{UP}} \frac{1}{T_m} \rightarrow \text{min}
\]
How to Design an Optimal Filter Configuration
Filter Design Problem

Given

- \( \mathcal{M}^{\text{all}} \) set of all broadcasted messages (ID, period)
- \( \mathcal{M}^{\text{des}} \) subset of messages which shall be received by node
- \( f \) number of available filters

Find

- filter-pattern for each of the \( f \) filters
- \( \text{load}^{\text{UP}} \rightarrow \text{min.} \)

3 Cases

- \( f \geq |\mathcal{M}^{\text{des}}| \)
- \( f = 1 \)
- \( 1 < f < |\mathcal{M}^{\text{des}}| \)

Complexity:
- problem is NP-complete
- transformation to SET COVER problem
- proof in paper
Optimal Filter: $f \geq |M^{des}|$

- straightforward
- each filter = one message ID
- perfect filtering

Algorithm 2: Optimal Solution for $f \geq |M^{des}|$ Filters

```
Input: $M^{des}$ /* desired messages */
1 foreach $m_i \in M^{des}$ do
2    $f_i = \text{ID}(m_i)$
3 end
Output: $F$ /* perfect filtering */
```
Optimal Filter: $f = 1$

- most constrained case
- for each filter-digit, derive filter-value from desired message’s ID-digit
- ensures feasible filter

### Algorithm 1: Optimal Solution for $f = 1$ Filter (always feasible)

**Input:** $M^{des}$ /* desired messages */

1. foreach ID-digit do
   2. if $\forall m_i \in M^{des}$ the ID-digit is 0 then
      3. filter-digit = 0
   4. else if $\forall m_i \in M^{des}$ the ID-digit is 1 then
      5. filter-digit = 1
   6. else
      7. filter-digit = x
   8. end
2. end

**Output:** $F$ /* feasible filtering */
simulated annealing (meta-heuristic search)

cost function:
\[ \cos t_1 = \frac{M^{\text{UP}}}{M^{\text{des}}} \]
... avoid infeasible filtering
\[ \cos t_2 = \frac{M^{\text{UP}}}{(M^{\text{all}} \setminus M^{\text{des}}) / \text{sec.}} \]
... avoid “unintended pass” load

problem encoding:
- group desired messages into \( f \) groups
- for each group: derive optimal filter (using algo.1)

neighbour move:
- move desired message \( m \) into another group, and re-calc filters
Optimal Filter: $1 < f < |M^{\text{des}}|$

- heuristic
  - sort desired messages by ID
  - assign $|M^{\text{des}}|/f$ desired messages to each filter
  - derive filter-pattern for each filter (algo.1)

- initial solution for SA
- as reference for evaluation
  (no filter design algorithm in literature)

Idea:
If IDs per filter are similar, then algo.1 should find effective filter-pattern
i.e. few x in filter-pattern
few unintended pass
Evaluation
(synthetic & real-world)
Large scale evaluation

- synthetic message-sets
  - 11-bit message ID (0 … 2047, uniform)
  - 10 … 1000 ms message period (log-uniform)
  - 25 … 100 broadcasted messages
  - 5 … 40 desired messages
  - 1 … 16 available filters
  - only schedulable message-sets considered

- $f=1$ 30 scenarios * 100 example-instances (algo.1)
- $1 < f < |M^{des}|$ 68 scenarios * 100 example-instances (SA, heuristic)
Evaluation: $f = 1$

- Algo.1
- Fig.2 in paper

- $|M^{\text{des}}|$ increases … filter-quality drops
- “good” filtering up to 3..4 des. messages
- $|M^{\text{all}}|$ has almost no effect
Evaluation: $1 < f < |M^{\text{des}}|$ 

- Fig. 7 in paper

- Color: nr. of filters
- Solid = SA, dashed = heuristic
- $|M^{\text{all}}| = 100$

- $|M^{\text{des}}|$ increases ... filter-quality drops
- $|f|$ increased ... better filter-quality
- SA always better than heuristic
Evaluation: $1 < f < |M^{\text{des}}|$ 

- Fig.3 in paper

- color: nr. of broadcast messages $|M^{\text{all}}|$ 
  - solid=SA , dashed=heuristic 
  - $|M^{\text{des}}|=20$

- $|f|$ increases … filter-quality increases
- $|M^{\text{all}}|$ increased … filter-quality lower 
  - but: only minor effect
- SA always better than heuristic
Real-World Application: HVAC-controller of lightweight battery-electric car

- developed in EU-funded project [www.epsilon-project.eu](http://www.epsilon-project.eu)
- our tasks: design electric powertrain, implement HVAC-controller, integrate into CAN-network

- CAN network: 55 messages (1407 messages/sec.)
- HVAC controller:
  - receives 11 messages (227 messages/sec)
  - sends 4 messages
  - AT90CAN128 micro-controller (ATMEL 16MHz)
  - 15 CAN message-buffers
    - perfect filtering is possible (with 11 Rx-filters)

- can we optimize CAN filtering with fewer Rx-filters?
Real-World Application: Results

- find best filter-configuration for different number of filters
- solve each scenario 100 times (for “perfect ratio”)

<table>
<thead>
<tr>
<th>Filters</th>
<th>Desired [msg/sec.]</th>
<th>Unintended [msg/sec.]</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>227 (11)</td>
<td>1180 (44)</td>
</tr>
<tr>
<td>3</td>
<td>227 (11)</td>
<td>48 (13)</td>
</tr>
<tr>
<td>7</td>
<td>227 (11)</td>
<td>0 (0)</td>
</tr>
<tr>
<td>9</td>
<td>227 (11)</td>
<td>0 (0)</td>
</tr>
</tbody>
</table>

- Rx-interrupt handler: 13.75...38.13us (avg. 31.25us)
  - 227 desired/sec: 7.1ms/sec (0.71%)
  - 48 unintended/sec: 1.5ms/sec (0.15%)
  - 1180 unintended/sec: 36.9ms/sec (3.69%)
Filter Details (3 vs. 7 filters)

<table>
<thead>
<tr>
<th>Filter</th>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>000xxx0x0xx</td>
<td>0000001010 (0x00a) UP</td>
<td>ID = 0000001010 (0x00a) UP</td>
</tr>
<tr>
<td></td>
<td>0000100000 (0x020) UP</td>
<td>ID = 0000100000 (0x020) UP</td>
</tr>
<tr>
<td></td>
<td>0001100000 (0x060) UP</td>
<td>ID = 0001100000 (0x060) UP</td>
</tr>
<tr>
<td></td>
<td>0001101010 (0x06a)</td>
<td>ID = 0001101010 (0x06a)</td>
</tr>
<tr>
<td></td>
<td>0010000001 (0x081)</td>
<td>ID = 0010000001 (0x081)</td>
</tr>
<tr>
<td>0x000xx0000</td>
<td>0000010000 (0x020) UP</td>
<td>ID = 0000010000 (0x020) UP</td>
</tr>
<tr>
<td></td>
<td>0100000000 (0x200) UP</td>
<td>ID = 0100000000 (0x200) UP</td>
</tr>
<tr>
<td></td>
<td>0100010000 (0x210)</td>
<td>ID = 0100010000 (0x210)</td>
</tr>
<tr>
<td></td>
<td>0100011000 (0x220)</td>
<td>ID = 0100011000 (0x220)</td>
</tr>
<tr>
<td>11xxxx00xx</td>
<td>1100001000 (0x610) UP</td>
<td>ID = 1100001000 (0x610) UP</td>
</tr>
<tr>
<td></td>
<td>1100001001 (0x611) UP</td>
<td>ID = 1100001001 (0x611) UP</td>
</tr>
<tr>
<td></td>
<td>1100001010 (0x612) UP</td>
<td>ID = 1100001010 (0x612) UP</td>
</tr>
<tr>
<td></td>
<td>1100001100 (0x613)</td>
<td>ID = 1100001100 (0x613)</td>
</tr>
<tr>
<td></td>
<td>1100001110 (0x614)</td>
<td>ID = 1100001110 (0x614)</td>
</tr>
<tr>
<td></td>
<td>1100001000 (0x620)</td>
<td>ID = 1100001000 (0x620)</td>
</tr>
<tr>
<td></td>
<td>1100111000 (0x6f0)</td>
<td>ID = 1100111000 (0x6f0)</td>
</tr>
<tr>
<td></td>
<td>1110000000 (0x700) UP</td>
<td>ID = 1110000000 (0x700) UP</td>
</tr>
<tr>
<td></td>
<td>1110000100 (0x710)</td>
<td>ID = 1110000100 (0x710)</td>
</tr>
<tr>
<td></td>
<td>1110001000 (0x720) UP</td>
<td>ID = 1110001000 (0x720) UP</td>
</tr>
<tr>
<td></td>
<td>1110011000 (0x730)</td>
<td>ID = 1110011000 (0x730)</td>
</tr>
<tr>
<td></td>
<td>1110111000 (0x770)</td>
<td>ID = 1110111000 (0x770)</td>
</tr>
<tr>
<td></td>
<td>1110111100 (0x771) UP</td>
<td>ID = 1110111100 (0x771) UP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Filter</th>
<th>ID</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0001101010</td>
<td>0001101010 (0x06a)</td>
<td>ID = 0001101010 (0x06a)</td>
</tr>
<tr>
<td>0001000001</td>
<td>0001000001 (0x06a)</td>
<td>ID = 0001000001 (0x06a)</td>
</tr>
<tr>
<td>1100010111</td>
<td>1100010111 (0x613)</td>
<td>ID = 1100010111 (0x613)</td>
</tr>
<tr>
<td>0x000010000</td>
<td>0000001000 (0x010)</td>
<td>ID = 0000001000 (0x010)</td>
</tr>
<tr>
<td></td>
<td>0100001000 (0x210)</td>
<td>ID = 0100001000 (0x210)</td>
</tr>
<tr>
<td>1100010000</td>
<td>1100010000 (0x710)</td>
<td>ID = 1100010000 (0x710)</td>
</tr>
<tr>
<td></td>
<td>1100111000 (0x730)</td>
<td>ID = 1100111000 (0x730)</td>
</tr>
<tr>
<td>11xx111000</td>
<td>1101111000 (0x6f0)</td>
<td>ID = 1101111000 (0x6f0)</td>
</tr>
<tr>
<td></td>
<td>1110111000 (0x770)</td>
<td>ID = 1110111000 (0x770)</td>
</tr>
<tr>
<td>x100010000</td>
<td>0100010000 (0x220)</td>
<td>ID = 0100010000 (0x220)</td>
</tr>
<tr>
<td></td>
<td>1100010000 (0x620)</td>
<td>ID = 1100010000 (0x620)</td>
</tr>
</tbody>
</table>
Conclusion & Outlook

Conclusion

- introduced engineering-problem: CAN message acceptance filter
- method for assessing of filtering quality
- method for designing near-optimal filter-configurations
- evaluation shows effectiveness (min. undesired Rx-interrupt load)

Future

- optimize filter quality for “now” and “future extensions” (extensibility)
  - try to block “not yet defined” broadcasted messages
- combine “message ID assignment” and “message filter design”
  - assign message IDs such that messages are schedulable, and efficient acceptance-filters can be designed
- bring methods to industry (engineering tools)
Questions & Discussion
Analysis and Optimization of Message Acceptance Filter Configurations for Controller Area Network (CAN)

25th International Conference on Real-Time Networks and Systems Grenoble, France, 4-6th October 2017

© 2017 Copyright held by the authors. Rights to publish the paper licensed to ACM. ISBN 978-1-4503-5286-4/17/10 DOI: https://doi.org/10.1145/3139258.3139266

Dr. Florian Pölzl Bauer
VIRTUAL VEHICLE Research Center
florian.poelzl@v2c2.at