A protocol for Mixed-Criticality management in Switched Ethernet networks

O. Cros, X. Li, L. George

December 1, 2015
What’s the matter?

Mixed-Criticality (MC) in network context
A protocol to manage MC
Delay computation
Simulation results
Plan and objectives

MC in network context
A protocol to manage MC
Delay computation
Simulation results
## Context

### Domains

- Public transport (CAN) (Volvo, Renault, ...)
- Avionics (AFDX) (Airbus)
- Home automation
- Defense
## Context

### Domains

- Public transport (CAN) (Volvo, Renault, ...)
- Avionics (AFDX) (Airbus)
- Home automation
- Defense

### Principles

- Message routing and scheduling
- Classifying messages by importance
## Context

### Domains
- Public transport (CAN) (Volvo, Renault, ...)
- Avionics (AFDX) (Airbus)
- Home automation
- Defense

### Principles
- Message routing and scheduling
- Classifying messages by importance

### MC
- Privileging messages in critical situation
  - Critical for the vehicle, for the mission, for the users
- Overload context
- Assuring critical messages transmission
Why Mixed-Criticality ?

<table>
<thead>
<tr>
<th>Today</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 network per group of functions (mechanical, comfort, gps tracking, ...)</td>
</tr>
<tr>
<td>Increasing of financial costs, weight, fuel and energy consumption</td>
</tr>
<tr>
<td>Example: 3/4 different antennas per public bus</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mixed-criticality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mixing all the functions in the same network</td>
</tr>
<tr>
<td>Each function associated to a criticality level</td>
</tr>
<tr>
<td>MC management protocol to guarantee critical messages transmission</td>
</tr>
</tbody>
</table>
#### Related work

- Mono/Multicore context
- 2-levels of criticality
- QoS

#### Synchronization protocol

- Ethernet IEEE-1588
- PTPv2
Mixed-criticality

Problems

How to manage messages scheduling inside a embedded network?
How to assure critical messages transmission?
## Mixed-criticality

### Problems

How to manage messages scheduling inside a embedded network?
How to assure critical messages transmission?

### Solution

Providing MC management in embedded networks
Period-oriented or WCTT-oriented
Static-defined path
Example

<table>
<thead>
<tr>
<th>Flow</th>
<th>$T_i^{LO}$ ($\mu s$)</th>
<th>$T_i^{HI}$ ($\mu s$)</th>
<th>$C_i$ ($\mu s$)</th>
<th>$u_i^{LO}$</th>
<th>$u_i^{HI}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$v_1$</td>
<td>500</td>
<td>250</td>
<td>100</td>
<td>0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>$v_2$</td>
<td>500</td>
<td>250</td>
<td>100</td>
<td>0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>$v_3$</td>
<td>300</td>
<td>-</td>
<td>100</td>
<td>0.33</td>
<td>-</td>
</tr>
</tbody>
</table>
Topology

Centralized topology

Automotive Ethernet, AFDX targets
One central node to store criticality information

Example
Plan and objectives

MC in network context

A protocol to manage MC

Delay computation

Simulation results
A two-phase protocol

First phase
Switch-criticality call
Transmitting switch order to a central node
Centralized topology

Second phase
Multicast the switch criticality order (reliable multicast)
Sending new criticality info to all nodes
Reliable (deterministic) multicast
A two-phase protocol: The call phase

<table>
<thead>
<tr>
<th>Principle</th>
</tr>
</thead>
</table>
| Triggering a criticality switch when:
| Longer period detected Or shorter inter-arrival time detected |
| Fixed WCTT $C_c$ (static known size) |
| Ethernet protocol |

<table>
<thead>
<tr>
<th>Transmission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dedicated VLAN, highest priority</td>
</tr>
<tr>
<td>PTP messages: clock accuracy (PTP-ETE)</td>
</tr>
</tbody>
</table>
MC management protocol
A two-phase protocol: the multicast phase

**Multicast**
Each node can get the switch criticality order at a different instant
Clock precision and clock synchronization (PTP - IEEE1588)
Total order: coherency in the network
At each instant, all nodes have the same criticality information (precision $\epsilon$)

**Reliability**
Each single physical link is bounded
Clock accuracy $\epsilon$
Worst-case delay computation

**Switch-criticality order**
All nodes switch at the same time
Last reception instant: $\max_{n \in \mathcal{N}} (d_n * (C_c + s_l) + \epsilon_n)$
Plan and objectives

MC in network context
A protocol to manage MC
Delay computation
Simulation results
Flows

Representing the network as a set of flows
Each flow $v_i$ emits messages
$v_i = \{P_i, C_i, \vec{T}_i\}$

Criticality management

One period per criticality level
$\vec{T}_i = \{T_i^{LO}, T_i^{HI}\}$
Call phase delay

Principle

Sending criticality switch call to central node
Emitting a call when a message exceeds its period or LO-WCTT
Highest priority VLAN

Delay

\[ I^n_{\text{delay}} = F_{\text{PTP}} \times \sum_{j \in \text{hpc}} \left( S^{\text{first}_{c,j}}_{\text{max}_c} - M^{\text{first}_{c,j}}_c + A_{c,j} \right) \]

\[ + \sum_{h \in \mathcal{P}_c} \delta^h_c + (|\mathcal{P}_n| - 1) \times (sI + 2 \times C_c) \]
Multicast phase delay

**Principle**

Sending the criticality level information to all nodes
Depending on the size of the network

**Delay**

\[ M^n_{delay} = d_n \times (C + sI) + \epsilon_n \]
Total delay

Phases delay

\[ S_{\text{delay}} = \max_{n \in \mathcal{N}} (I^n_{\text{delay}} + M^n_{\text{delay}}) \]

Final expression

\[
S_{\text{delay}} = F_{PTP} \times \sum_{j \in hpc \atop \mathcal{P}_c \cap \mathcal{P}_j \neq \emptyset} \left( S_{\text{max}_c}^{\text{first}_{c,j}} - M_c^{\text{first}_{c,j}} + A_{c,j} \right) \\
+ \sum_{h \in \mathcal{P}_c} \delta^h_c \\
+ (2 \times \max_{n \in \mathcal{N}}(d_n) - 1) \times (C_c + sl) + C_c(\max_{n \in \mathcal{N}}(d_n) - 1) + \epsilon
\]
Plan and objectives

MC in network context
A protocol to manage MC
Delay computation
Simulation results
<table>
<thead>
<tr>
<th>Load</th>
<th>Delay (µs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3</td>
<td>120</td>
</tr>
<tr>
<td>0.4</td>
<td>130</td>
</tr>
<tr>
<td>0.5</td>
<td>140</td>
</tr>
<tr>
<td>0.6</td>
<td>150</td>
</tr>
<tr>
<td>0.7</td>
<td>160</td>
</tr>
</tbody>
</table>

Criticality switch delay

![Graph showing criticality switch delay](image)

- WCTT_MAX: 20 µs
- WCTT_MAX: 30 µs
- WCTT_MAX: 40 µs
- WCTT_MAX: 115.8 µs
Criticality switch delay

Highest priority (except PTP)
Non-preemptive effect
Switch criticality delay stays constant
Criticality messages transmission is guaranteed in a bounded time
Conclusion

MC management protocol
Reliable multicast
Independent from the load

Perspectives

Delay computation on switch-criticality delay
Delay computation to return to low-criticality mode
Uncentralized MC management
Conclusion

Thanks

Thanks to all the authors of this presentation and publication
Thanks to our respective labs
Thank you for your attention!

Questions? Feel free to ask!