Documentation

CAN Bus Simulator

Student: Gergő Papp-Szentannai, gr. 30431

Laboratory assistant: Vlad Miclea

Technical University of Cluj-Napoca, Computer Science, 2017

Structure of Computer Systems

Contents

[Introduction 2](#_Toc484094769)

[Problem 3](#_Toc484094770)

[Proposed solution 3](#_Toc484094771)

[Experimental results 4](#_Toc484094772)

[Conclusions 4](#_Toc484094773)

[References 5](#_Toc484094774)

# Introduction

The aim of this project is to simulate a Controller Area Network (CAN).

"A Controller Area Network (CAN bus) is a vehicle bus standard designed to allow microcontrollers and devices to communicate with each other in applications without a host computer. It is a message-based protocol, designed originally for multiplex electrical wiring within automobiles, but is also used in many other contexts."

*Source: CAN bus -* [*https://en.wikipedia.org*](https://en.wikipedia.org/wiki/special:search/CAN%20bus)

Unlike many other communication networks, this one does not have a controller or a host computer. Each node is connected to the bus and they can send and receive data.

*Source: File:CAN ISO11898-2 Network.png -* [*https://en.wikipedia.org*](https://commons.wikimedia.org/wiki/File:CAN_ISO11898-2_Network.png)

Because this way many nodes could simultaneously write to the bus, data is organized in frames. At any given time, only one node can transmit a frame, while the remaining nodes can simultaneously read the frame. Each node has an arbitration ID. Transmitting any new frame must begin with the arbitration ID. If two or more nodes try to transmit a frame at the same time, the node that has the lowest ID takes control. In other words, if a node transmits the ID, but senses, that a different node is also transmitting having a lower ID, it must stop transmitting.

Simultaneous writes to the bus are possible, because logical “0” is dominant and logical “1” is recessive. If one node writes “1” and another “0”, the value of the bus will remain “0”. In this case, the less important node can sense that it could not write its arbitration ID, so it gives control to the node with a lower ID.

*Source: File:ISO11898-2.svg -* [*https://en.wikipedia.org*](https://commons.wikimedia.org/wiki/File:ISO11898-2.svg)

# Problem

In many cases, engineers might first want to simulate the behavior of a network before implementing it. Although the CAN bus is a simple one, it is good to consider simulating it first.

# Proposed solution

To simulate the behavior of a CAN, we can write a simulator program in a high-level programming language, like Java or C#.

The application would simulate the bus and the interface of a CAN node. Each CAN node would be controlled by a Microcontroller. Some Microcontrollers implementations will be provided, but more implementations can be added later.

The model of the CAN node follows the next scheme:

*Source: File:CAN Node.png -* [*https://en.wikipedia.org*](https://commons.wikimedia.org/wiki/File:CAN_Node.png)

The communication between the nodes (and the bus) is implemented using the observable pattern. At any time, a node can broadcast its message on the bus. This message will then be received by each node. The nodes can decide whether this information is required for them or not.

The full implementation is available on GitHub: <https://github.com/gergo13/CanSim>

# Experimental results

I could send frames to the network, and these frames were intercepted by other nodes. In this sense, the simulation succeeded.

It is however important to mention, that I could not fully simulate a real network, mostly because of the timing. In real networks a baud rate is specified and nodes can synchronize themselves after each frame.

Because this was just a simulation, the nodes could not fully run in parallel. Each node had its own thread, but these were controlled by the operating system, so they could not be extremely precise.

# Conclusions

In conclusion, this project meant a very detailed introduction in the field of automotive communication. I have learned a lot of valuable information.

# References

* <https://www.linkedin.com/pulse/automotive-can-bus-system-explained-kiril-mucevski>
* <https://www.youtube.com/watch?v=RRbrk3SdSKA>
* <https://en.wikipedia.org/wiki/CAN_bus>