Controller Area Networks: The Basics and Its Applications

Michigan State University

College of Engineering

Electrical and Computer Engineering

Timur Yaprak

ECE 480

A37631375

Design Team 9

Table of Contents

Keywords

Communication Bus – A subsystem that transfers data between electrical components.

Point-to-Point – Communication medium between two nodes.

Half-Duplex – Two electronic devices that can send information/data and receive, but not simultaneously.

Protocol – Communication framework and rules abided by and established between nodes in a communication medium.

ISO – International Standardization Organization

OSI – Open System Interconnection

EMI – Electromagnetic Interference

Introduction and Background

Introduction

Objective

The objective and purpose of this application note is to engage the reader in a tutorial-like overview to instruct the reader how a Controller Area Network (CAN) works, and how it may be implemented in various applications. To accomplish this objective, I researched the background, and technical properties of CAN to apply this knowledge to our design project in developing a robotic transportation vehicle prototype. If the reader is interested in more detailed descriptions please refer to the recommended additional resources page.

Background of CAN

In the past, electronic components in vehicles used point-to-point wiring systems. As manufacturers began integrating more and more electrical components into vehicles, wiring harnesses became more bulky and thus heavy and expensive, promoting an opportunity to create a more efficient, less complex design. This resulted in a communication network developed by Bosch Gmbh and other various professors at research institutions to help resolve this issue. Although its development was intended for automotive applications, Controller Area Networks fulfills the communication needs for a wide range of industrial applications. Some of these applications include: medical devices, aviation, and robotics. It supports messages of up to 8 bytes of length and an operation rate of up to 1Mbps, and possesses excellent error detection mechanisms that cause for a valuable solution for these various applications. This short data length, high speed, and efficient error detection make it a suitable solution for cars, robotics, and other smaller machines.

Open System Interconnection (OSI) 7-Layer Reference Model

In the past, different vendors used different protocols that did not allow computers or other electronics to communicate with each other. In order to address this issue, the International Standardization Organization (ISO) developed a 7-Layer model for describing interconnecting systems. This is referred to the OSI model shown in Figure 1. In this model, each layer performs related subset of functions required to communicate with another system. Each layer relies on the next lower layer to perform more primitive functions and it also provides specific services to the next layer.

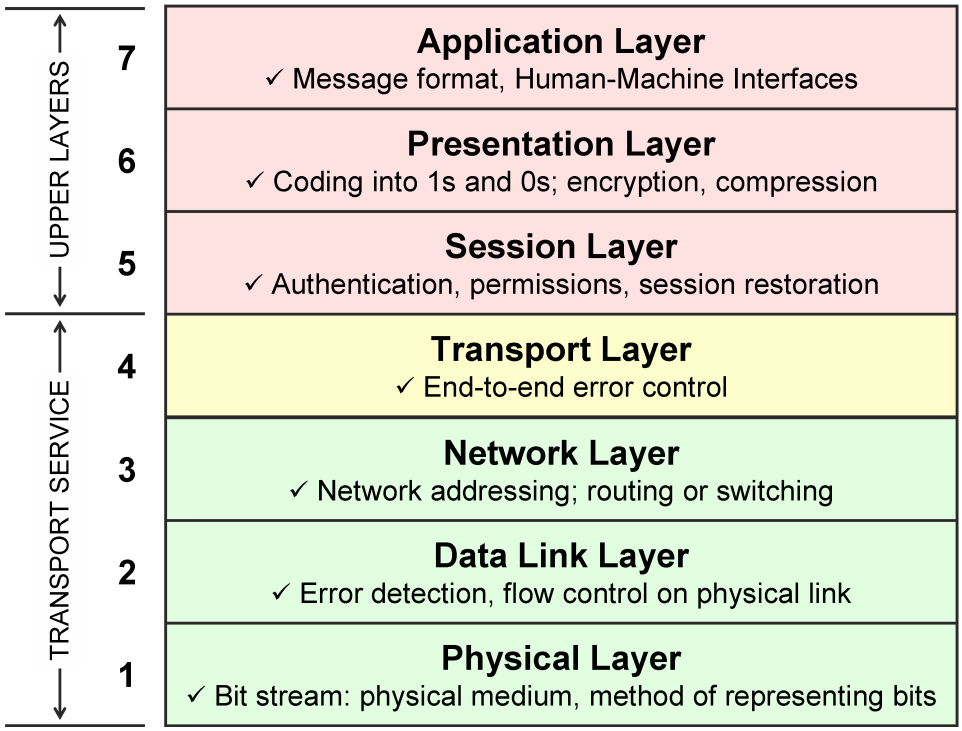

Figure 1 – OSI Model (http://www.telecommunications-tutorials.com/tutorial-OSI-7-layer-model.htm)

The CAN standard only defines the bottom two layers, the physical layer and the data link layer. The physical layer is implemented in the hardware and interprets the electrical characteristics of the bus, such as converting characters into electrical signals for transmitted messages, and converting electrical signals into characters for received messages. The Data Link Layer on the other hand, specifies the communication that defines a few different message types and provides the framework of bus arbitration as well as error detection and correction.

The CAN protocol contains nothing on topics such as how to transport data larger than 8-byte messages, which nodes “talk” to which, when to trigger transmit messages, etc. These sorts of functions are specified in higher layer protocols and implemented in software. Examples of such CAN application layers are CAL (CAN Application Layer), CANopen (an implementation of CAL), and CANKingdom (developed by Kvaser).

Main Characteristics of CAN

Physical Layer

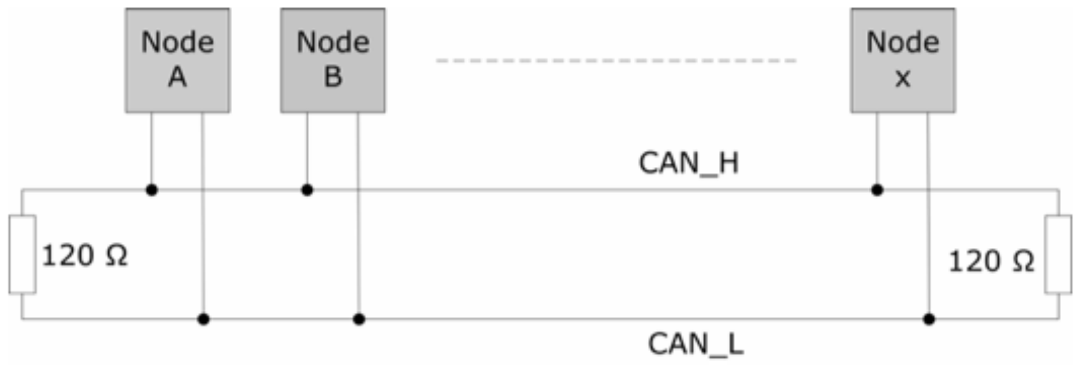

Figure 2 – CAN Bus (A Comprehensive Guide to Controller Area Networks (Voss))

As seen in Figure 2, CAN is constructed as a two-wire, half-duplex serial network system. In this system, these two wires CAN_H and CAN_L, have two voltage levels, either 0V or 5V. The dominant voltage level is 0V and the recessive voltage level is 5V. These two CAN wires are usually in a twisted-pair form to alleviate noise and EMI reception. There are also two 120 ohms resistances at the end of the bus acting as terminating resistances to prevent electrical reflections on the bus. Many nodes can be connected to the bus as long as the bus is not overloaded with frames.

Frames

All messages in the CAN protocol are referred to as frames. There are four different types of messages frames, each having different lengths and purposes. A data frame, contains the corresponding data of the transmitted message and may be accepted by numerous nodes. Data transfer is triggered once there is a remote frame broadcasted.

Message Frame Format

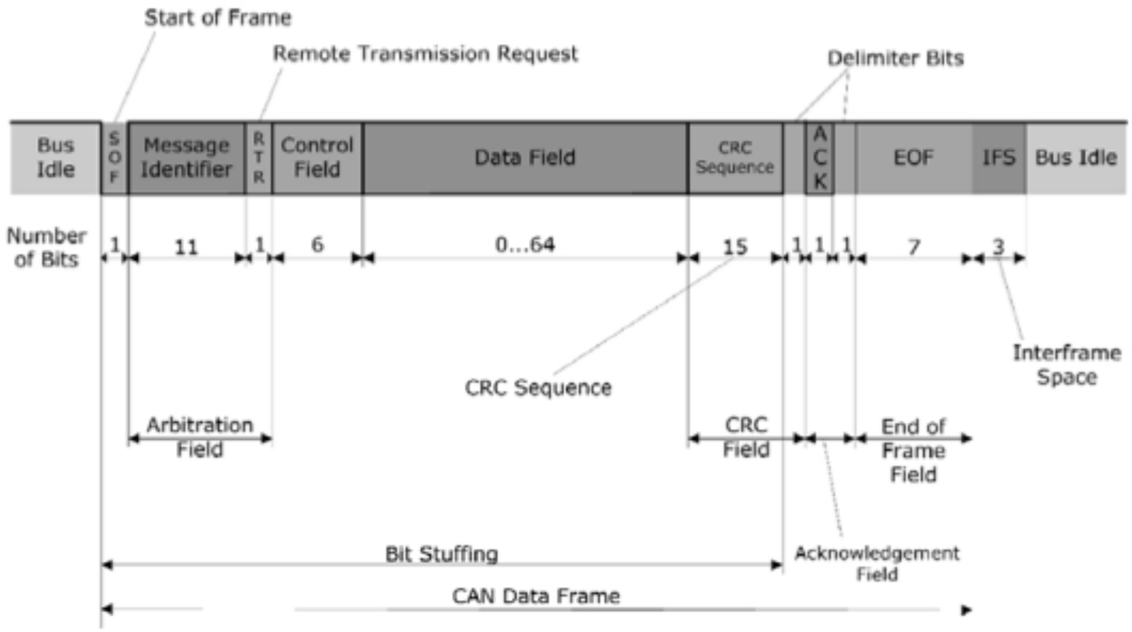

Figure 3 - Message Frame Format (A Comprehensive Guide to Controller Area Networks (Voss))

The Start of Frame (SOF) bit signals the beginning of a message frame. The Remote Transmission Request (RTR) bit separates the data from the remote frame. The Arbitration Field includes the message ID and RTR bit. This field determines the priority of the message when two or more nodes are contending to access the bus.

A remote frame, requests the data from a node and contains the same message ID as the data frame. A remote frame is almost identical to the data frame, except that it doesn't contain data.

An error frame, reports an error if an error detection occurs. And finally, an overload frame reports when a node is overloaded and will request a delay between data and remote frame transmissions. [Voss]

Bus Arbitration

CAN has a unique bus arbitration system and uses a multi-master bus access system, differing itself from other communication systems. Basically, all the nodes on bus share the same physical communication medium and each node on the bus is considered the master when it is transmitting frames to the other nodes. All nodes within the network have the same access to the bus and all "listen" to the bus in order to determine if the frame being transmitted is relevant to them. [Voss]

Through message filtering, only the relevant nodes will accept the frames being transmitted and will receive these messages. If the bus is idle, any node can transmit a frame. If two nodes simultaneously begin to transmit frames, then depending on the message ID, the frame will be sent. The lowest message ID is considered the highest priority frame, therefore that frame will be sent first with the other remaining nodes transmitting after them. [Voss] Here is an example of where this may occur:

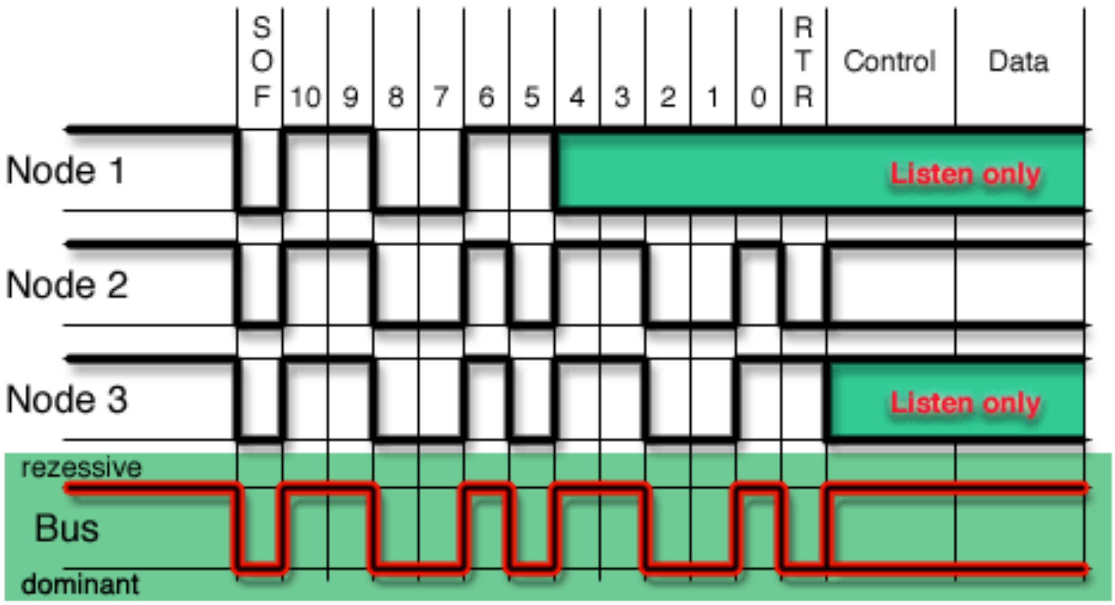

Figure 4 – Bus Arbitration Example (http://www.can-cia.org/index.php?id=systemdesign-can-protocol)

As seen in Figure 3, Nodes 1,2,and 3 all transmit the same dominant value until bit 5, where node 1 transmits a recessive value, and therefore begins "listening" because there are nodes with lower message IDs attempting to transmit. Node 2 is determined the transmitting node after the RTR (Remote Transmission Request) bit, and as illustrated in the diagram above the bus is identical to Node 2's bit values. In hexadecimal values, the message IDs are:

	Hex Value	SOF	11-bit MESSAGE ID	RTR	
Node1	**CC0**	**0**	**1 1 0 0 1 ~~1 0 0 0 0 0~~**	**~~0~~**	**Listening Mode**
Node2	**CB2**	**0**	**1 1 0 0 1 0 1 1 0 0 1**	**0**	**Has Bus Access**
Node3	**CB3**	**0**	**1 1 0 0 1 0 1 1 0 0 1**	**~~1~~**	**Listening Mode**

Therefore, because Node 2 has the smallest message ID, and highest priority, it is subjected to transmit its frame on the bus first. All of the nodes on the bus are connected in a wired-AND fashion creating a network where the dominant level will always override a recessive level to gain bus access.

Error Detection

There are a few different methods CAN uses when detecting errors and resolving them. Some of these methods include bit monitoring, cyclic redundancy checks, frame checks, and acknowledgement checks. The bit monitoring error detection method involves transmitters of the nodes on the bus compare the transmitted level bit by bit with the corresponding level on the bus. [Voss] If there happens to be a difference in the level between the transmitted level and the monitored bus level, a bit error will then be detected.

If a node transmits a certain number of faulty frames or messages, then this node is eliminated from the bus, making the bus more efficient and clear of less defected frames.

Recommended Additional Resources

If the reader would like to gain more insight on this particular subject of communication systems, I would encourage the reader to follow up this material with the following readings:

- A Comprehensive Guide to Controller Area Networks by Wilfried Voss
- Introduction to the Controller Area Network (CAN) – Application Report by Steve Corrigan (http://www.ti.com/lit/an/sloa101a/sloa101a.pdf)
- Controller Area Network by Konrad Etschberger
- Embedded Networking with CAN and CANopen by Olaf Pfeiffer, Andrew Ayre, and Christian Keydel
- Controller Area Network Projects by Dogan Ibrahim

References

1.) http://www.can-cia.org/index.php?id=systemdesign-can-protocol

2.)http://www.telecommunications-tutorials.com/tutorial-OSI-7-layer-model.htm

3.)Voss, Wilfried. *A Comprehensible Guide to Controller Area Network*. Greenfield, MA: Copperhill Technologies Corporation, 2008. Print.