

FPGA Based Controller Area Network

Abdolhamid Shorabi*

* Department of Electronic Engineering, Bushehr Branch Islamic Azad University, Bushehr, Iran

Article Info

Article history:

Received Dec 20, 2014

Revised Mar 28, 2015

Accepted Apr 20, 2015

Keyword:

Controller area network

FIFO

FPGA

NDBA

Non return to zero

ABSTRACT

In this paper the Controller Area Network (CAN) Controller is presented. CAN is an advance serial bus communication protocol that efficiently supports distributed, broadcast real-time control and fault tolerance features for automobile industries to provide congestion free networking. The CAN Controller is designed for scheduling of messages, consist of the Transmitter Controller, FIFO buffer, CRC generator and bit stuffer. Scheduling messages on CAN corresponds to assigning identifiers (IDs) to message according to their priorities. Non Return to Zero (NRZ) coding and Non Destructive Bitwise Arbitration (NDBA) is used. The data is taken from the buffer FIFO, bit stuffed and then transmitted after CRC is performed. The whole design is captured entirely in VHDL language using bottom up design and verification methodology. The proposed controller was designed for applications needing high level data integrity and data rates upto 1Mbps. The applications of CAN are factory automation, machine control, automobile, avionics and aerospace, building automation.

Copyright © 2015 Institute of Advanced Engineering and Science.

All rights reserved.

Corresponding Author:

Abdolhamid Shorabi,
Department of Electronic Engineering,
Bushehr Branch Islamic Azad University, Bushehr, Iran.
Email: hamid.sohrabi844@gmail.com

1. INTRODUCTION

Network between industrial equipment, established through CONTROLLER AREA NETWORK is similar to computer network. The protocol used is CAN protocol. A computer network is a communication system that allows computer to exchange information with each other in a meaningful way. A protocol is a formalized set of procedural rules for the exchange of information and for the interactions of the networks interconnected nodes. ISO (7498) defines a communication standard known as the open systems interconnection (OSI) model. The OSI model defines seven independent layers of a protocol stack. They are Application, Presentation, Session, Transport, network, data-link and physical layer. The CAN specification (ISO 11898) discuss only the physical and data-link layers for a CAN network. CAN is internationally standardized by International Standard Organization (ISO) and Society Of Automotive Engineers (SAE). The German Company Robert Bosch GmbH, for the automobile industry originally developed the Controller Area Network during the late 1980's. CAN bus is designed for communication between microcontrollers in an automotive environment. It is used to exchange information between on-board Electronic Control Units (ECUs) such as the engine management system, gearbox, instrument packs and body electronics. In 1993 CAN become the standards ISO 11898 (for high speed applications) and ISO 11519 (for low speed applications). CAN is a multi-master serial communication bus for high speed, high noise-immunity and error detection features. CAN is a high-integrity serial data communications bus for real-time applications. Operates at data rates of up to 1 Megabits per second .Has excellent error detection and confinement capabilities It is now being used in many other industrial automation and control applications. The paper is organized as follows: In sections 2 and 3 a brief description of the existing and proposed system features are presented. Controller Area Network shown in section 4, and the simulation results and paper conclusion are given in section 5.

2. EXISTING SYSTEM

The Existing system is the system before CAN specifications and the standard chip CAN is shown in (Figure 1).

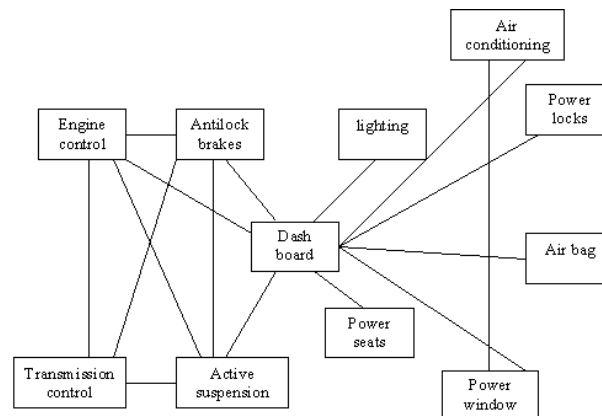


Figure 1. Before CAN

The connections are point to point connection from each and every node in the existing system. The standard chips are already available but the main disadvantages of using the chips are as follows. They are not easy to integrate and need large memories, software for basic operation, maximal amount of support, not ready to use and fault detection is very difficult. They are not used for the standalone applications as well as in powerful communication modules. They will not guarantee long-term availability of products. They have limitations such as size and power consumption. The frequency required long-term availability makes its usage impossible or risky.

3. PROPOSED SYSTEM

The proposed system is design of CAN Controller using VLSI architecture. Field Programmable Gate Array (FPGA) is used for hardware implementation. CAN on FPGA enable to see bit by bit, what happens on CAN bus. In addition to valid CAN messages, bit error cases, error counters, internal status, frame position etc can be seen online.

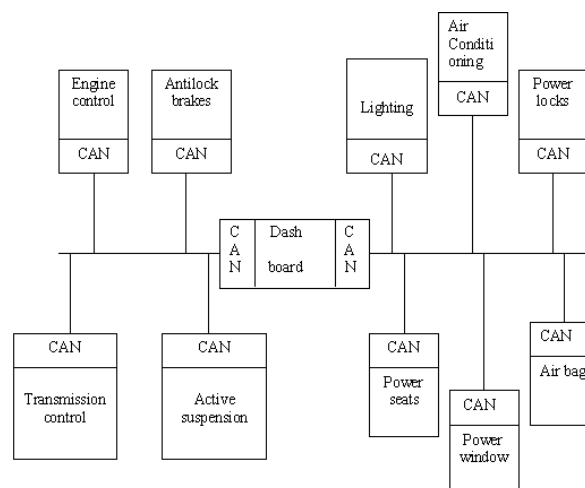


Figure 2. With CAN

The above all can be done without using a complete pc based CAN Analyzer tool. The high flexibility of these solutions positions FPGA CAN nodes together with software based solution (CPU+CAN).

The normally higher price per part of the FPGA is compensated for by the ease of use where no programming knows how and software maintenance is needed. CAN solutions on FPGA donot depend on a particular technology or version of a chip, since the circuit description is easily portable to any type of FPGA. The additional features can be added. The circuit is optimally protected against the copying of the particular project. FPGA prototyping is important area where complex ASIC developments can be prototyped in hardware. Real system wide network simulations are done with FPGA based CAN nodes at low cost.

This offers technology independent description of a long term availability and better performance through intelligent hardware support of higher layer functions. Combination of DSP technology and peripherals all onto one chip can be performed. The message routing, segmentation and reassembly of long frames, automatic configuration of each node etc is supported by proposed system. This proposed system even works standalone.

The higher cost of the application specific parts are more than compensated for by better performance, product exclusivity and long term savings. Integration technologies may change but the data base remains the same. It is efficient and can be done very fast.

4. CONTROLLER AREA NETWORK

CAN Controller Area Network) is a reliable high-performance protocol, which was designed for linking control units in a heavily distributed environment. It was developed by Robert Bosch GmbH for the automobile industry, where a reliable bus system was needed to reduce the abundant amount of wires in a vehicle. CAN protocol is a bus protocol with many desirable properties for embedded and real –time systems, CAN is inexpensive and widely used in factory automation and in vehicles. CAN bus is an on a line-topology. CAN is a serial communication bus designed for broadcasting short real-time control messages. Today, it is used in many other areas, too, where secure communication between control units is needed, e.g.in the automation industry. Remarkable features are its minimal latency, fail-safe behavior due to error correction precautions which have been implemented and low connection costs for subscriber circuits.

The protocol was raised an international standard in ISO-DIS 11898 and ISO-DIS 11519-1.several chips are already commercially available according to specification 2,0A or 2,0B (Extended CAN) . These devices are either microprocessors with integrated CAN interface, that can be directly connected to the CAN bus, or stand-alone CAN controllers, that can act as an interface between the host processors and the CAN bus. Another type are so called SLIOs (Serial Linked Input Output), which are input/output devices with integrated CAN interfaces these intelligent sensors or actuators provide the means for realization of distributed systems.

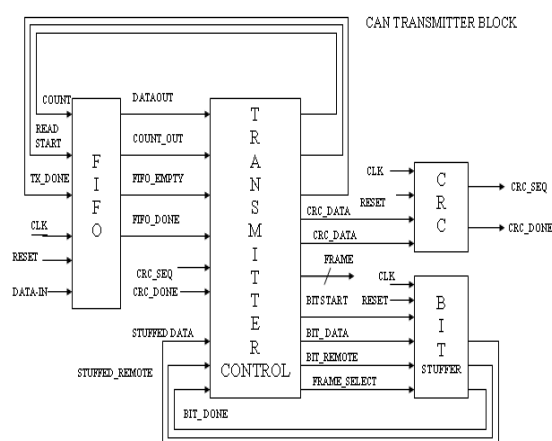


Figure 3. CAN Transmitter

4.1. Technical Data

Because of the usage in highly distributed environments, the transmission media should either be an optical fibre or a shielded cable by two wires, where the signal is determined by the voltage difference. The

transmission rate was chosen between 10 kbit/s and 1mbit/s, with a capacity of 0 to 8 byte of data per message.

4.2. Broadcasting

The CAN protocol knows no source or destination address. A message is broadcasted to all nodes simultaneously. Each message contains an identifier of 11 bit or 29 bit (extended CAN) describing its contents. By an acceptance filtering mechanism each node can distinguish the relevant messages from the irrelevant ones.

4.3. Multimaster Network Management

Each node is allowed to start transmission, if the bus is idle (carrier sense). A message on the bus must not be interrupted (non-destructive), unless an error is detected. If two nodes start transmitting simultaneously, the message with the lower value as the identifier will have the higher priority, and the other messages will be aborted (arbitration without time consumption). Therefore CAN is a CSMA/CA protocol (Carrier Sense Multiple Access/Collision Avoidance). The CAN can be used in real-time environment, as a message with the highest priority will never take longer than 259 bit times for transmission.

4.4. Error Handling

The following mechanisms are used in the CAN protocol for error detection:

- A transmitter checks whether sent out bits appear on the bus.
- Cyclic redundancy check (CRC)
- Bit stuffing
- Check of the frame formats
- Acknowledge bit

According to the five simultaneous errors are definitely detected. This corresponds to a hamming distance of 6.

In order to guaranteed, the system wide data consistency an incorrect message is destroyed by the sender itself or by a receiving node as soon as the error is detected. The sending node will then repeat the transmission of this frame.

If a permanent failure occurs a node will disconnect itself logically from the network. Thus the functionality of the network is guaranteed, even if certain nodes fail to operate correctly.

With all these qualities the Controller Area Network is very suitable in an environment, where short messages have to be exchanged between peer stations. With a very small overhead secure communication can be established. Additionally the small overhead also results in high baud rates and even real-time requirements can be met. For these reasons more and more applications are using the Controller Area Network for communication between electronic control units in a distributed system [2].

4.5. Message Format

SOF	Identifier	Control	Data	CRC	Ack	EOF
-----	------------	---------	------	-----	-----	-----

Figure 4. Message format

The meaning of the bit fields are:

1. SOF—The single dominant start of frame (SOF) bit marks the start of a message, and is used to synchronize the nodes on a bus after being idle.
2. Identifier—The Standard CAN 11-bit identifier establishes the priority of the message. The lower the binary value, the higher its priority.
3. RTR—The single remote transmission request (RTR) bit is dominant when information is required from another node. All nodes receive the request, but the identifier determines the specified node. The responding data is also received by all nodes and used by any node interested. In this way all data being used in a system is uniform.

4. IDE—A dominant single identifier extension (IDE) bit means that a standard CAN identifier with no extension is being transmitted.
5. r0—Reserved bit (for possible use by future standard amendment).
6. DLC—The 4-bit data length code (DLC) contains the number of bytes of data being transmitted.
7. Data—Up to 64 bits of application data may be transmitted.
8. CRC—The 16-bit (15 bits plus delimiter) cyclic redundancy check (CRC) contains the checksum (number of bits transmitted) of the preceding application data for error detection.
9. ACK—Every node receiving an accurate message overwrites this recessive bit in the original message with a dominate bit, indicating an error-free message has been sent. Should a receiving node detect an error and leave this bit recessive, it discards the message and the sending node repeats the message after re arbitration. In this way each node acknowledges (ACK) the integrity of its data. ACK is 2 bits, one is the acknowledgement bit and the second is a delimiter.
10. EOF—This end-of-frame (EOF) 7-bit field marks the end of a CAN frame (message) and disables bit-stuffing, indicating a stuffing error when dominant. When 5 bits of the same logic level occur in succession during normal operation, a bit of the opposite logic level is stuffed into the data.

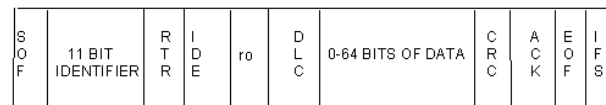


Figure 5. Standard format

11. IFS—This 7-bit inter-frame space (IFS) contains the amount of time required by the controller to move a correctly received frame to its proper position in a message buffer area.
1. SRR—The substitute remote request (SRR) bit replaces the RTR bit in the standard message location as a placeholder in the extended format.
2. IDE—A recessive bit in the identifier extension (IDE) indicates that there are more identifier bits to follow. The 18-bit extension follows IDE.

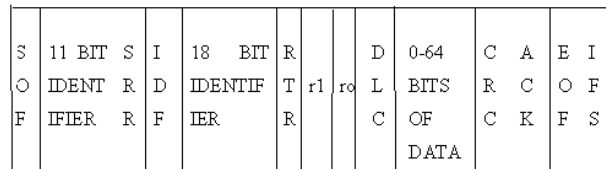


Figure 6. Extended format

3. r1—Following the RTR and r0 bits, an additional reserve bit has been included ahead of the DLC bit.

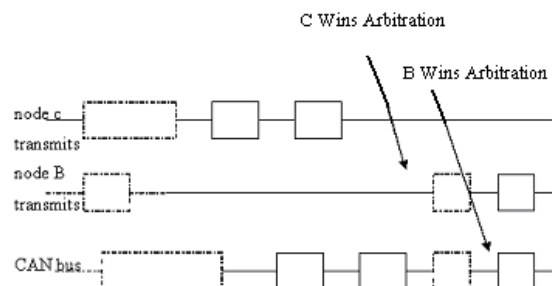


Figure 7. Arbitration on a CAN Bus

4.6. Message Types

There are four different message types, or frames that can be transmitted on a CAN bus: the data frame, the remote frame, the error frame, and the overload frame. A message is considered to be error free when the last bit of the ending EOF field of a message is received in the error-free recessive state. A dominant bit in the EOF field causes the transmitter to repeat a transmission.

4.6.1. The Data Frame

The data frame is the most common message type, and is made up by the arbitration field, the data field, the CRC field, and the acknowledgement field. The arbitration field determines the priority of a message when two or more nodes are contending for the bus. The arbitration field contains an 11-bit identifier for CAN 2.0A and the RTR bit, which is dominant for data frames. For CAN 2.0B it contains the 29-bit identifier and the RTR bit. Next is the data field which contains zero to eight bytes of data, and the CRC field which contains the 16-bit checksum used for error detection. Lastly, there is the acknowledgement field. The CAN controller that is able to correctly receive a message sends a dominant ACK bit that overwrites the transmitted recessive bit at the end of correct message transmission. The transmitter checks for the presence of the dominant ACK bit and retransmits the message if no acknowledge is detected.

4.6.2. The Remote Frame

The intended purpose of the remote frame is to solicit the transmission of data from another node. The remote frame is similar to the data frame, with two important differences. First, this type of message is explicitly marked as a remote frame by a recessive RTR bit in the arbitration field, and secondly, there is no data.

4.6.3. The Error Frame

The error frame is a special message that violates the formatting rules of a CAN message. It is transmitted when a node detects an error in a message, and causes all other nodes in the network to send an error frame as well. The original transmitter then automatically retransmits the message. There is an elaborate system of error counters in the CAN controller which ensures that a node cannot tie up a bus by repeatedly transmitting error frames.

4.6.4. The Overload Frame

The overload frame is mentioned here for completeness. It is similar to the error frame with regard to the format, and it is transmitted by a node that becomes too busy. It is primarily used to provide for an extra delay between messages.

4.7. Error Checking and Fault Confinement

The CAN protocol incorporates five methods of error checking: three at the message level and two at the bit level. If a message fails with any one of these error detection methods, it is not accepted and an error frame is generated from the receiving nodes, causing the transmitting node to re send the message until it is received correctly. However, if a faulty node hangs up a bus by continuously repeating an error, its transmit capability is removed by its controller after an error limit is reached. At the message level are the CRC and the ACK slots displayed. The 16-bit CRC contains the checksum of the preceding application data for error detection with a 15-bit checksum and 1-bit delimiter.

The ACK field is two bits long and consists of the acknowledge bit and an acknowledge delimiter bit. Finally, at the message level there is a form check. This check looks for fields in the message which must always be recessive bits. If a dominant bit is detected, an error is generated. The bits checked are the SOF, EOF, ACK delimiter, and the CRC delimiter bits. At the bit level each bit transmitted is monitored by the transmitter of the message. If a data bit (not arbitration bit) is written onto the bus and its opposite is read, an error is generated. The only exceptions to this are with the message identifier field which is used for arbitration, and the acknowledge slot which requires a recessive bit to be overwritten by a dominant bit.

The final method of error detection is with the bit stuffing rule where after five consecutive bits of the same logic level, if the next bit is not a complement, an error is generated. Stuffing ensures rising edges available for on-going synchronization of the network, and that a stream of recessive bits are not mistaken for an error frame, or the seven-bit interframe space that signifies the end of a message. Stuffed bits are removed by a receiving node's controller before the data is forwarded to the application.

5. SIMULATION RESULT AND CONCLUSION

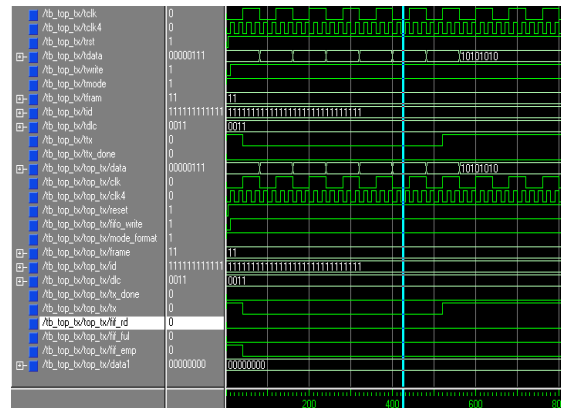


Figure 8. Simulation Result

In this paper we have paper presented the fundamental characteristics of CAN and designed the CAN controller. This CAN bus is broadcast bus with a multi-master architecture aims the transaction of messages in a small scale distributed environments. Because of its real time and fault tolerance capabilities, CAN controller has gained a wide acceptance in a large number of application areas: automotive, astronomy, agriculture, biochemical, medical systems, robotics, building and industrial automation

REFERENCES

- [1] Robert Bosch Gmbh, CAN specification Version 2.0, September 1991.
- [2] Hanser-Verlag, Munchen, Wien, "CAN Controller Area Network", 1994.
- [3] E. Barke, IMS Hanover: Breadboard zum Simulator und zuruck, slides for a workshop about complex systems verification, Munich, 1994.
- [4] D. Behrens, E.Kiel: "Logikemulation mit FPGAs- Der Weg aus der Verifikationskrise?", GI/ITG Workshop, Anwenderprogrammier-bare Schaltungen, Kalsruhe, 1994.
- [5] A. Winter, D. Bitruf, Y. Tanurhan, K.D. Muller-Glaser, "Rapid Prototyping of a Communication Controller for the CAN Bus", Proceedings of the 7th IEEE International workshop on rapid system prototyping, 1996.
- [6] Ian Broster, Guillem Bernat and Alan Burns, "Weakly Hard Real-time Constraints on Controller Area Network", Proceedings of the 14th Euromicro Conference on Real-Time Systems, 2002, IEEE.
- [7] K. Zuberi and K. Shin, "Non-preemptive scheduling of messages on controller area networks for real-time control applications", In proceedings of the IEEE Real-Time Technology and Application Symposium, pages 240-249, Chicago, Illinois, USA, May 1995.