International Journal of Applied Engineering and Technology ISSN: 2277-212X (Online)
An Online International Journal Available at http://www.cibtech.org/jet.htm
2012 Vol. 2 (1) January-March, pp1-5/Reddy et al.
Research Article

CAN PROTOCOL IMPLEMENTATION FOR INDUSTRIAL PROCESS

D.V.S.Reddy[1], Alia Sultana[2], K.Madhavi Latha[3] and M.Haritha[4]
Dept of ECE, Brindavan Institute of Technology and Science, Kurnool
**Author for Correspondence*

ABSTRACT
To implementation of industrial parameter control through CAN protocol by using 8051 microcontrollers.CAN is a multi-master broadcast serial bus standard for connecting electronic control units (ECUs).Each node is able to send and receive messages, but not simultaneously: a message (consisting primarily of an ID usually chosen to identify the message-type/sender and up to eight message bytes) is transmitted serially onto the bus, one bit after another this signal pattern codes the message (in NRZ) and is sensed by all nodes. The devices that are connected by a CAN network are typically sensors, A CAN message never reaches these devices directly, but instead a host processor and a CAN controller are needed between these devices and the bus. Bit rates up to 1 Mbit/s are possible at network lengths below 40 m. Decreasing the bit rate allows longer network distances (e.g. 125 kbit/s at 500 m). The programming language used for developing the software to the microcontroller is Embedded/Assembly. The KEIL cross compiler is used to edit, compile and debug this program. Micro Flash programmer is used for burning the developed code on Keil in to the microcontroller Chip.

INTRODUCTION
Controller–area network (CAN or CAN-bus) is a vehicle bus standard designed to allow microcontrollers and devices to communicate with each other within a vehicle without a host computer.CAN is a message-based protocol, designed specifically for automotive applications but now also used in other areas such as industrial automation and medical equipment .Development of the CAN-bus started originally in 1983 at Robert Bosch GmbH. The protocol was officially released in 1986 at the Society of Automotive Engineers (SAE) congress in Detroit, Michigan. The first CAN controller chips, produced by Intel and Philips, came on the market in 1987. Bosch published the CAN 2.0 s

DESCRIPTION
CAN is a multi-master broadcast serial bus standard for connecting electronic control units (ECUs).Each node is able to send and receive messages, but not simultaneously. A message consists primarily of an id, which represents the priority of the message, and up to eight data bytes. It is transmitted serially onto the bus. This signal pattern is encoded in non-return-to-zero(NRZ) and is sensed by all nodes.

The devices that are connected by a CAN network are typically sensors, actuators, and other control devices. These devices are not connected directly to the bus, but through a host processor and a CAN controller.

If the bus is free, any node may begin to transmit. If two or more nodes begin sending messages at the same time, the message with the more dominant id (which has more dominant bits, i.e., zeroes) will overwrite other nodes' less dominant id's, so that eventually (after this arbitration on the id.) only the dominant message remains and is received by all nodes. This mechanism is referred to as priority based bus arbitration. Messages with numerically smaller values of id. have higher priority and are transmitted first.

Node Requirements
Host processor
The host processor decides what received messages mean and which messages it wants to transmit itself. Sensors, actuators and control devices can be connected to the host processor.

CAN controller (hardware with a synchronous clock).
Receiving: The CAN controller stores received bits serially from the bus until an entire message is available, which can then be fetched by the host processor (usually after the CAN controller has triggered an interrupt).

Sending: The host processor stores it's transmit messages to a CAN controller, which transmits the bits serially onto the bus.

Bus state with two nodes transmitting

	Dominant	Recessive
Dominant	Dominant	Dominant
Recessive	Dominant	Recessive

Fig(a): Truth table for dominant/recessive

Logical AND

	0	1
0	0	0
1	0	1

Fig(b): Truth table for AND

Logical OR

	1	0
1	1	1
0	1	0

Fig(c):Truth table for OR

Transceiver (possibly integrated into the CAN controller)

Receiving: it adapts signal levels from the bus to levels that the CAN controller expects and has protective circuitry that protects the CAN controller.

Sending: it converts the transmit-bit signal received from the CAN controller into a signal that is sent onto the bus.

DATA TRANSMISSION

CAN feature an automatic arbitration-free transmission. A CAN message that is transmitted with highest priority will succeed and the node transmitting the lower priority message will sense this and back off and wait.

This is achieved by CAN transmitting data through a binary model of "dominant" bits and "recessive" bits where dominant is a logical 0 and recessive is a logical 1. This means open collector, or wired or physical implementation of the bus (but since dominant is 0 this is sometimes referred to as wired and). If one node transmits a dominant bit and another node transmits a recessive bit then the dominant bit "wins" (alogical AND between the two).

So, if you are transmitting a recessive bit, and someone sends a dominant bit, you see a dominant bit, and you know there was a collision. (All other collisions are invisible.) A dominant bit is asserted by creating a voltage across the wires while a recessive bit is simply not asserted on the bus. If any node sets a voltage difference, all nodes will see it. Thus there is no delay to the higher priority messages, and the node transmitting the lower priority message automatically attempts to re-transmit 6 bit clocks after the end of the dominant message.

When used with a differential bus, a carrier sense multiple access/bitwise arbitration (CSMA/BA) scheme is often implemented: if two or more devices start transmitting at the same time, there is a priority based arbitration scheme to decide which one will be granted permission to continue transmitting. The CAN solution to this is prioritized arbitration (and for the dominant message delay free), making CAN very suitable for real time prioritized communications systems.

During arbitration, each transmitting node monitors the bus state and compares the received bit with the transmitted bit. If a dominant bit is received when a recessive bit is transmitted then the node stops transmitting (i.e., it lost arbitration). Arbitration is performed during the transmission of the identifier field. Each node starting to transmit at the same time sends an id. With dominant as binary 0, starting from the high bit. As soon as their id. is a larger number (lower priority) they will be sending 1 (recessive) and see 0 (dominant), so they back off. At the end of id. transmission, all nodes but one have backed off, and the highest priority message gets through unimpeded.

For example, consider an 11-bit id. CAN network, with two nodes with id's of 15 (binary representation, 00000001111) and 16 (binary representation, 00000010000). If these two nodes transmit at the same time, each will transmit the first 6 zeros of their id. with no arbitration decision being made. When the 7th bit is transmitted, the node with the id. of 16 transmit a 1 (recessive) for its id., and the node with the id. of 15 transmits a 0 (dominant) for its id.. When this happens, the node with the id. of 16 will realize that it lost its arbitration, and allow the node with id. of 15 to continue its transmission. This ensures that the node with the lower bit value will always win the arbitration. The id. with the smaller number will win the right to use.

ID ALLOCATION

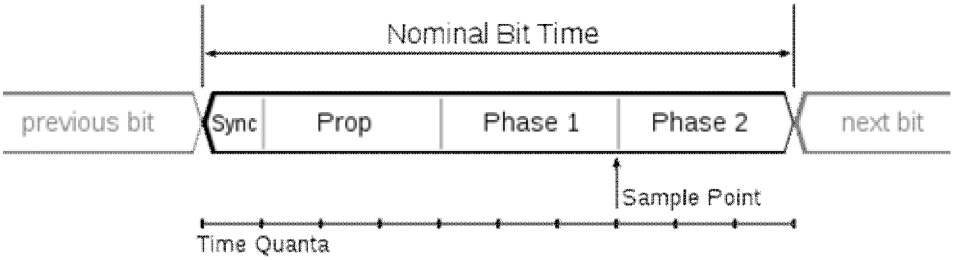
Messages id's must be unique on a single CAN bus, otherwise two nodes would continue transmission beyond the end of the arbitration field (id.) causing an error.

The choice of id's for messages is often done simply on the basis of identifying the type of data and the sending node; however, as the id. is also used as the message priority, this can lead to poor real-time performance. For example, if an urgent message with a short deadline has a numerically high id. (low priority) then its transmission can be delayed by other messages with lower numerical id's (higher priority), even though these messages may have much longer deadlines.

In the early 1990s, CAN messages in automotive systems were typically given id's based only on the type of data and sending node, and this led to the widely held but incorrect belief that a low CAN bus utilization of circa 30% was required to ensure that all messages would meet their deadlines. However, if id's are instead determined based on the deadline of the message, so the shorter the deadline, the lower the numerical id. and hence the higher the message priority, then bus utilizations of 70 to 80% can typically be achieved before any message deadlines are missed.

BIT TIMING

Each node in a CAN network has its own clock, and no clock is sent during data transmission. Synchronization is done by dividing each bit of the frame into a number of segments: Synchronization, Propagation, Phase 1 and Phase 2. The Length of each phase segment can be adjusted based on network and node conditions. The sample point falls between Phase Buffer Segment 1 and Phase Buffer Segment 2, which helps facilitate continuous synchronization. Continuous synchronization in turn enables the receiver to be able to properly read the messages.



CAN Bit Timing

LAYERS

Based on levels of abstraction, the structure of the CAN protocol can be described in terms of the following layers:

- Application Layer
- Object Layer
- Message Filtering
- Message and Status Handling

TRANSFER LAYER

The Transfer Layer represents the kernel of the CAN protocol. It presents messages received to the object layer and accepts messages to be transmitted from the object layer. The transfer layer is responsible for bit

timing and synchronization, message framing, arbitration, acknowledgement, error detection and signaling, and fault confinement. It performs:

- Fault Confinement
- Error Detection
- Message Validation
- Acknowledgement
- Arbitration
- Message Framing
- Transfer Rate and Timing
- Information Routing

PHYSICAL LAYER

CAN bus in its original form was a link layer protocol specification that made reference to the physical layer only in abstract terms (i.e. as a medium that supported multiple access at the individual bit level by providing dominant and recessive states). The mechanical aspects of the physical layer however (connector type, pin-outs etc.) continue to lack formal treatment, as an automotive electronic control unit will typically have a single—often custom—connector that integrates the CAN bus lines, and it is from this background that we see the emergence of several de-facto standards for mechanical implementation with the 9 Way 'D' type CAN-(2), CAN+(7), OV (3), Supply(9) configuration as one of the most common.

The absence of a formal definition for the physical layer had the original advantage of freeing the scope of CAN bus from the constraints of any one particular physical implementation however its legacy has been to leave CAN bus implementations open to inter-operability issues in the physical domain.

The very high noise immunity on ISO11898-2 is achieved by ensuring that the differential impedance of the bus is maintained at a very low level using low value resistors (120 ohms) mounted at each end of the bus. The low impedance draws more current (and power) than other voltage based signaling specifications during the dominant state. On CAN bus systems, balanced line operation, where current in one signal line is exactly balanced by current in the opposite direction in the other signal leg becomes an essential facility to provide an independent stable 0 V reference for the receivers. Best practice determines that CAN bus balanced pair signals be carried in twisted pair form within a shielded cable and it is this precaution that helps to keep RF emissions to a minimum.

Common 'good practice' node design might provide each node with transceivers which are optically isolated from their node host and derive a 5 V linearly regulated supply voltage for the transceivers from the universal supply rail provided by the bus. This configuration generally allows a sufficiently wide operating margin on the supply rail to provide interoperability across a broad spectrum of different node types. Typical values of supply voltage on such networks range between 7 and 30 Volts however lack of standardization means that ultimately it is up to the individual system designer to check out all aspects of supply rail compatibility.

The standard (ISO11898-2) describes the electrical implementation formed from a multi-dropped single-ended balanced line configuration with resistor termination at each end of the bus. In this configuration a dominant state is asserted by 1 or more transmitters switching the CAN- to supply 0 V and (simultaneously) switching CAN+ to the +5 V bus voltage thereby forming a current path through the resistors that terminate the bus. As such the terminating resistors form an essential component of the signaling system and are included not just to limit wave reflection at high frequency. During a recessive state the signal lines and resistor(s) remain in a high impedances state with respect to both rails. Voltages on both CAN+ and CAN- tend (weakly) towards ½ rail voltage. During a dominant state the signal lines and resistor(s) move to a low impedance state with respect to the rails so that current flows through the resistor. CAN+ voltage tends to +5 V and CAN- tends to 0 V. A recessive state is only present on the bus when none of the transmitters on the bus is asserting a dominant state. Irrespective of signal state the signals lines are always in low impedance state with respect to one another by virtue of the terminating resistors at the end of the bus.

This signaling strategy differs significantly from other balanced line transmission technologies such as RS-422/3, RS-485, etc. which employ differential line drivers/ receivers and use a signaling system based on the differential mode voltage of the balanced line crossing a notional 0 V. Multiple access on such systems normally relies on the media supporting 3 states (active high, active low and inactive tri-state)and is dealt with in the time domain. Multiple access on CAN bus is achieved by the electrical logic of the system supporting just 2 states that are conceptually analagous to a 'wired OR' network.

FRAME

A CAN network can be configured to work with two different message (or "frame") formats: the standard or base frame format (or CAN 2.0 A), and the extended frame format (or CAN 2.0 B). The only difference between the two formats is that the "CAN base frame" supports a length of 11 bits for the identifier, and the "CAN extended frame" supports a length of 29 bits for the identifier, made up of the 11-bit identifier ("base identifier") and an 18-bit extension ("identifier extension"). The distinction between CAN base frame format and CAN extended frame format is made by using the IDE bit, which is transmitted as dominant in case of an 11-bit frame, and transmitted as recessive in case of a 29-bit frame. CAN controllers that support extended frame format messages are also able to send and receive messages in CAN base frame format. All frames begin with a start-of-frame (SOF) bit that denotes the start of the frame transmission.

CAN has four frame types:

- Data frame: a frame containing node data for transmission.
- Remote frame: a frame requesting the transmission of a specific identifier.
- Error frame: a frame transmitted by any node detecting an error.
- Overload frame: a frame to inject a delay between data and/or remote frame. Semiconductor.

Advantages:

- Easy to implement and extend.
- Well-suited for temporary or small networks not requiring high speeds (quick setup), resulting in faster networks.
- Less expensive than other topologies (But in recent years has become less important due to devices like a switch)
- Cost effective; only a single cable is used.
- Easy identification of cable faults. The breakdown of a CAN station has no immediate impact on the CAN bus. All the other stations can communicate unconstrained.

CONCLUSION

In this project we have implemented industrial parameter control through CAN protocol by using 8051 microcontroller. By using CAN protocol a vehicle bus standard designed to allow microcontroller and devices to communicate with each other with in a vehicle without a host computer. The sensors, actuators and other control devices that are connected by CAN protocol.

REFERENCES

J.Rauch (2001). *The New old Economy: oil, Computers And The Reinvention of the Earth,* The atlantic .

K.H.Johansson (**2002)**. *Hybrid control systems, UNESCO Encyclopedia of life support systems*. Dept of signals, sensors & Systems, RoYal institute of technology, Sweden.

J. Lygeros (*2004).* *Lecture notes on hybrid systems. Dept of Elect. Comp. Engg. University of Patras, Greece.*

A.K.Ray and Mazidi (2000). *The 8051Micro controllers and embedded systems*.

John B peatman (1998). *Design with PIC Micro controllers*, Pearson education.

Todd D Morton (2001). Embedded Micro controllers. Pearson Education.