MICROCHIP

AN713

Controller Area Network (CAN) Basics

Author: Keith Pazul
Microchip Technology Inc.

INTRODUCTION

Controller Area Network (CAN) was initially created by German automotive system supplier Robert Bosch in the mid-1980s for automotive applications as a method for enabling robust serial communication. The goal was to make automobiles more reliable, safe and fuel-efficient while decreasing wiring harness weight and complexity. Since its inception, the CAN protocol has gained widespread popularity in industrial automation and automotive/truck applications. Other markets where networked solutions can bring attractive benefits like medical equipment, test equipment and mobile machines are also starting to utilize the benefits of CAN. The goal of this application note is to explain some of the basics of CAN and show the benefits of choosing CAN for embedded systems networked applications.

CAN OVERVIEW

Most network applications follow a layered approach to system implementation. This systematic approach enables interoperability between products from different manufacturers. A standard was created by the International Standards Organization (ISO) as a template to follow for this layered approach. It is called the ISO Open Systems Interconnection (OSI) Network Layering Reference Model and is shown in Figure 1 for reference.

The CAN protocol itself implements most of the lower two layers of this reference model. The communication medium portion of the model was purposely left out of the Bosch CAN specification to enable system designers to adapt and optimize the communication protocol on multiple media for maximum flexibility (twisted pair, single wire, optically isolated, RF, IR, etc.). With this flexibility, however, comes the possibility of interoperability concerns.

To ease some of these concerns, the International Standards Organization and Society of Automotive Engineers (SAE) have defined some protocols based on CAN that include the Media Dependant Interface definition such that all of the lower two layers are specified. ISO11898 is a standard for high-speed applications, ISO11519 is a standard for low-speed applications, and J1939 (from SAE) is targeted for truck and bus applications. All three of these protocols specify a 5V differential electrical bus as the physical interface.

The rest of the layers of the ISO/OSI protocol stack are left to be implemented by the system software developer. Higher Layer Protocols (HLPs) are generally used to implement the upper five layers of the OSI Reference Model.

HLPs are used to:

1) standardize startup procedures including bit rates used,
2) distribute addresses among participating nodes or types of messages,
3) determine the structure of the messages, and
4) provide system-level error handling routines.

This is by no means a full list of the functions HLPs perform, however it does describe some of their basic functionality.

CAN PROTOCOL BASICS

Carrier Sense Multiple Access with Collision Detection (CSMA/CD)

The CAN communication protocol is a CSMA/CD protocol. The CSMA stands for Carrier Sense Multiple Access. What this means is that every node on the network must monitor the bus for a period of no activity before trying to send a message on the bus (Carrier Sense). Also, once this period of no activity occurs, every node on the bus has an equal opportunity to transmit a message (Multiple Access). The CD stands for Collision Detection. If two nodes on the network start transmitting at the same time, the nodes will detect the ‘collision’ and take the appropriate action. In CAN protocol, a non-destructive bitwise arbitration method is utilized. This means that messages remain intact after arbitration is completed even if collisions are detected. All of this arbitration takes place without corruption or delay of the higher priority message.

There are a couple of things that are required to support non-destructive bitwise arbitration. First, logic states need to be defined as dominant or recessive. Second, the transmitting node must monitor the state of the bus to see if the logic state it is trying to send actually appears on the bus. CAN defines a logic bit 0 as a dominant bit and a logic bit 1 as a recessive bit.

A dominant bit state will always win arbitration over a recessive bit state, therefore the lower the value in the Message Identifier (the field used in the message arbitration process), the higher the priority of the message. As an example, suppose two nodes are trying to transmit a message at the same time. Each node will monitor the bus to make sure the bit that it is trying to send actually appears on the bus. The lower priority message will at some point try to send a recessive bit and the monitored state on the bus will be a dominant. At that point this node loses arbitration and immediately stops transmitting. The higher priority message will continue until completion and the node that lost arbitration will wait for the next period of no activity on the bus and try to transmit its message again.

Message-Based Communication

CAN protocol is a message-based protocol, not an address based protocol. This means that messages are not transmitted from one node to another node based on addresses. Embedded in the CAN message itself is the priority and the contents of the data being transmitted. All nodes in the system receive every message transmitted on the bus (and will acknowledge if the message was properly received). It is up to each node in the system to decide whether the message received should be immediately discarded or kept to be processed. A single message can be destined for one particular node to receive, or many nodes based on the way the network and system are designed.

For example, an automotive airbag sensor can be connected via CAN to a safety system router node only. This router node takes in other safety system information and routes it to all other nodes on the safety system network. Then all the other nodes on the safety system network can receive the latest airbag sensor information from the router at the same time, acknowledge if the message was received properly, and decide whether to utilize this information or discard it.

Another useful feature built into the CAN protocol is the ability for a node to request information from other nodes. This is called a Remote Transmit Request (RTR). This is different from the example in the previous paragraph because instead of waiting for information to be sent by a particular node, this node specifically requests data to be sent to it.

For example, a safety system in a car gets frequent updates from critical sensors like the airbags, but it may not receive frequent updates from other sensors like the oil pressure sensor or the low battery sensor to make sure they are functioning properly. Periodically, the safety system can request data from these other sensors and perform a thorough safety system check. The system designer can utilize this feature to minimize network traffic while still maintaining the integrity of the network.

One additional benefit of this message-based protocol is that additional nodes can be added to the system without the necessity to reprogram all other nodes to recognize this addition. This new node will start receiving messages from the network and, based on the message ID, decide whether to process or discard the received information.

CAN Message Frame Description

CAN protocol defines four different types of messages (or Frames). The first and most common type of frame is a Data Frame. This is used when a node transmits information to any or all other nodes in the system. Second is a Remote Frame, which is basically a Data Frame with the RTR bit set to signify it is a Remote Transmit Request (see Figure 2 and Figure 3 for details on Data Frames). The other two frame types are for handling errors. One is called an Error Frame and one is called an Overload Frame. Error Frames are generated by nodes that detect any one of the many protocol errors defined by CAN. Overload errors are generated by nodes that require more time to process messages already received.

Data Frames consist of fields that provide additional information about the message as defined by the CAN specification. Embedded in the Data Frames are Arbitration Fields, Control Fields, Data Fields, CRC Fields, a 2-bit Acknowledge Field and an End of Frame.

The Arbitration Field is used to prioritize messages on the bus. Since the CAN protocol defines a logical 0 as the dominant state, the lower the number in the arbitration field, the higher priority the message has on the bus. The arbitration field consists of 12-bits (11 identifier bits and one RTR bit) or 32-bits (29 identifier bits, 1-bit to define the message as an extended data frame, an SRR bit which is unused, and an RTR bit), depending on whether Standard Frames or Extended Frames are being utilized. The current version of the CAN specification, version 2.0B, defines 29-bit identifiers and calls them Extended Frames. Previous versions of the CAN specification defined 11-bit identifiers which are called Standard Frames.

As described in the preceding section, the Remote Transmit Request (RTR) is used by a node when it requires information to be sent to it from another node. To accomplish an RTR, a Remote Frame is sent with the identifier of the required Data Frame. The RTR bit in the Arbitration Field is utilized to differentiate between a Remote Frame and a Data Frame. If the RTR bit is recessive, then the message is a Remote Frame. If the RTR bit is dominant, the message is a Data Frame.

The Control Field consists of six bits. The MSB is the IDE bit (signifies Extended Frame) which should be dominant for Standard Data Frames. This bit determines if the message is a Standard or Extended Frame. In Extended Frames, this bit is RB1 and it is reserved. The next bit is RB0 and it is also reserved. The four LSBs are the Data Length Code (DLC) bits. The Data Length Code bits determine how many data bytes are included in the message. It should be noted that a Remote Frame has no data field, regardless of the value of the DLC bits.

The Data Field consists of the number of data bytes described in the Data Length Code of the Control Field.

The CRC Field consists of a 15-bit CRC field and a CRC delimiter, and is used by receiving nodes to determine if transmission errors have occurred.

The Acknowledge Field is utilized to indicate if the message was received correctly. Any node that has correctly received the message, regardless of whether the node processes or discards the data, puts a dominant bit on the bus in the ACK Slot bit time (see Figure 2 or Figure 3 for the location of the ACK Slot bit time).

The last two message types are Error Frames and Overload Frames. When a node detects one of the many types of errors defined by the CAN protocol, an Error Frame occurs. Overload Frames tell the network that the node sending the Overload Frame is not ready to receive additional messages at this time, or that intermission has been violated. These errors will be discussed in more detail in the next section.

Fast, Robust Communication

Because CAN was initially designed for use in automobiles, a protocol that efficiently handled errors was critical if it was to gain market acceptance. With the release of version 2.0B of the CAN specification, the maximum communication rate was increased 8x over the version 1.0 specification to 1Mbit/sec. At this rate, even the most time-critical parameters can be transmitted serially without latency concerns. In addition to this, the CAN protocol has a comprehensive list of errors it can detect that ensures the integrity of messages.

CAN nodes have the ability to determine fault conditions and transition to different modes based on the severity of problems being encountered. They also have the ability to detect short disturbances from permanent failures and modify their functionality accordingly. CAN nodes can transition from functioning like a normal node (being able to transmit and receive messages normally), to shutting down completely (bus-off) based on the severity of the errors detected. This feature is called Fault Confinement. No faulty CAN node or nodes will be able to monopolize all of the bandwidth on the network because faults will be confined to the faulty nodes and these faulty nodes will shut off before bringing the network down. This is very powerful because Fault Confinement guarantees bandwidth for critical system information.

As discussed previously, there are five error conditions that are defined in the CAN protocol and three error states that a node can be in, based upon the type and number of error conditions detected. The following section describes each one in more detail.

Errors Detected

CRC Error

A 15-bit Cyclic Redundancy Check (CRC) value is calculated by the transmitting node and this 15-bit value is transmitted in the CRC field. All nodes on the network receive this message, calculate a CRC and verify that the CRC values match. If the values do not match, a CRC error occurs and an Error Frame is generated. Since at least one node did not properly receive the message, it is then resent after a proper intermission time.

Acknowledge Error

In the Acknowledge Field of a message, the transmitting node checks if the Acknowledge Slot (which it has sent as a recessive bit) contains a dominant bit. This dominant bit would acknowledge that at least one node correctly received the message. If this bit is recessive, then no node received the message properly. An Acknowledge Error has occurred. An Error Frame is then generated and the original message will be repeated after a proper intermission time.

Form Error

If any node detects a dominant bit in one of the following four segments of the message: End of Frame, Interframe Space, Acknowledge Delimiter or CRC Delimiter, the CAN protocol defines this to be a form violation and a Form Error is generated. The original message is then resent after a proper intermission time. (see Figure 2 and/or Figure 3 for where these segments lie in a CAN message).

Bit Error

A Bit Error occurs if a transmitter sends a dominant bit and detects a recessive bit, or if it sends a recessive bit and detects a dominant bit when monitoring the actual bus level and comparing it to the bit that it has just sent. In the case where the transmitter sends a recessive bit and a dominant bit is detected during the Arbitration Field or Acknowledge Slot, no Bit Error is generated because normal arbitration or acknowledgment is occurring. If a Bit Error is detected, an Error Frame is generated and the original message is resent after a proper intermission time.

Stuff Error

CAN protocol uses a Non-Return–to-Zero (NRZ) transmission method. This means that the bit level is placed on the bus for the entire bit time. CAN is also asynchronous, and bit stuffing is used to allow receiving nodes to synchronize by recovering clock information from the data stream. Receiving nodes synchronize on recessive to dominant transitions. If there are more than five bits of the same polarity in a row, CAN will automatically stuff an opposite polarity bit in the data stream. The receiving node(s) will use it for synchronization, but will ignore the stuff bit for data purposes. If, between the Start of Frame and the CRC Delimiter, six consecutive bits with the same polarity are detected, then the bit stuffing rule has been violated. A Stuff Error then occurs, an Error Frame is sent, and the message is repeated.

<u>Error States</u>

Detected errors are made public to all other nodes via Error Frames or Error Flags. The transmission of an erroneous message is aborted and the frame is repeated as soon as the message can again win arbitration on the network. Also, each node is in one of three error states, Error-Active, Error-Passive or Bus-Off.

<u>Error-Active</u>

An Error-Active node can actively take part in bus communication, including sending an active error flag, which consists of six consecutive dominant bits. The Error Flag actively violates the bit stuffing rule and causes all other nodes to send an Error Flag, called the Error Echo Flag, in response. An Active Error Flag, and the subsequent Error Echo Flag may cause as many as twelve consecutive dominant bits on the bus; six from the Active Error Flag, and zero up to six more from the Error Echo Flag depending upon when each node detects an error on the bus. A node is Error-Active when both the Transmit Error Counter (TEC) and the Receive Error Counter (REC) are below 128. Error-Active is the normal operational mode, allowing the node to transmit and receive without restrictions.

<u>Error-Passive</u>

A node becomes Error-Passive when either the Transmit Error Counter or Receive Error Counter exceeds 127. Error-Passive nodes are not permitted to transmit Active Error Flags on the bus, but instead, transmit Passive Error Flags which consist of six recessive bits. If the Error-Passive node is currently the only transmitter on the bus then the passive error flag will violate the bit stuffing rule and the receiving node(s) will respond with Error Flags of their own (either active or passive depending upon their own error state). If the Error-Passive node in question is not the only transmitter (i.e. during arbitration) or is a receiver, then the Passive Error Flag will have no effect on the bus due to the recessive nature of the error flag. When an Error-Passive node transmits a Passive Error Flag and detects a dominant bit, it must see the bus as being idle for eight additional bit times after an intermission before recognizing the bus as available. After this time, it will attempt to retransmit.

<u>Bus-Off</u>

A node goes into the Bus-Off state when the Transmit Error Counter is greater than 255 (receive errors can not cause a node to go Bus-Off). In this mode, the node can not send or receive messages, acknowledge messages, or transmit Error Frames of any kind. This is how Fault Confinement is achieved. There is a bus recovery sequence that is defined by the CAN protocol that allows a node that is Bus-Off to recover, return to Error-Active, and begin transmitting again if the fault condition is removed.

CONCLUSION

The CAN protocol was optimized for systems that need to transmit and receive relatively small amounts of information (as compared to Ethernet or USB, which are designed to move much larger blocks of data) reliably to any or all other nodes on the network. CSMA/CD allows every node to have an equal chance to gain access to the bus, and allows for smooth handling of collisions.

Since the protocol is message-based, not address based, all messages on the bus receive every message and acknowledge every message, regardless of whether in needs the data or not. This allows the bus to operate in node-to-node or multicast messaging formats without having to send different types of messages.

Fast, robust message transmission with fault confinement is also a big plus for CAN because faulty nodes will automatically drop off the bus not allowing any one node from bringing a network down. This effectively guarantees that bandwidth will always be available for critical messages to be transmitted. With all of these benefits built into the CAN protocol and its momentum in the automotive world, other markets will begin to see and implement CAN into their systems.

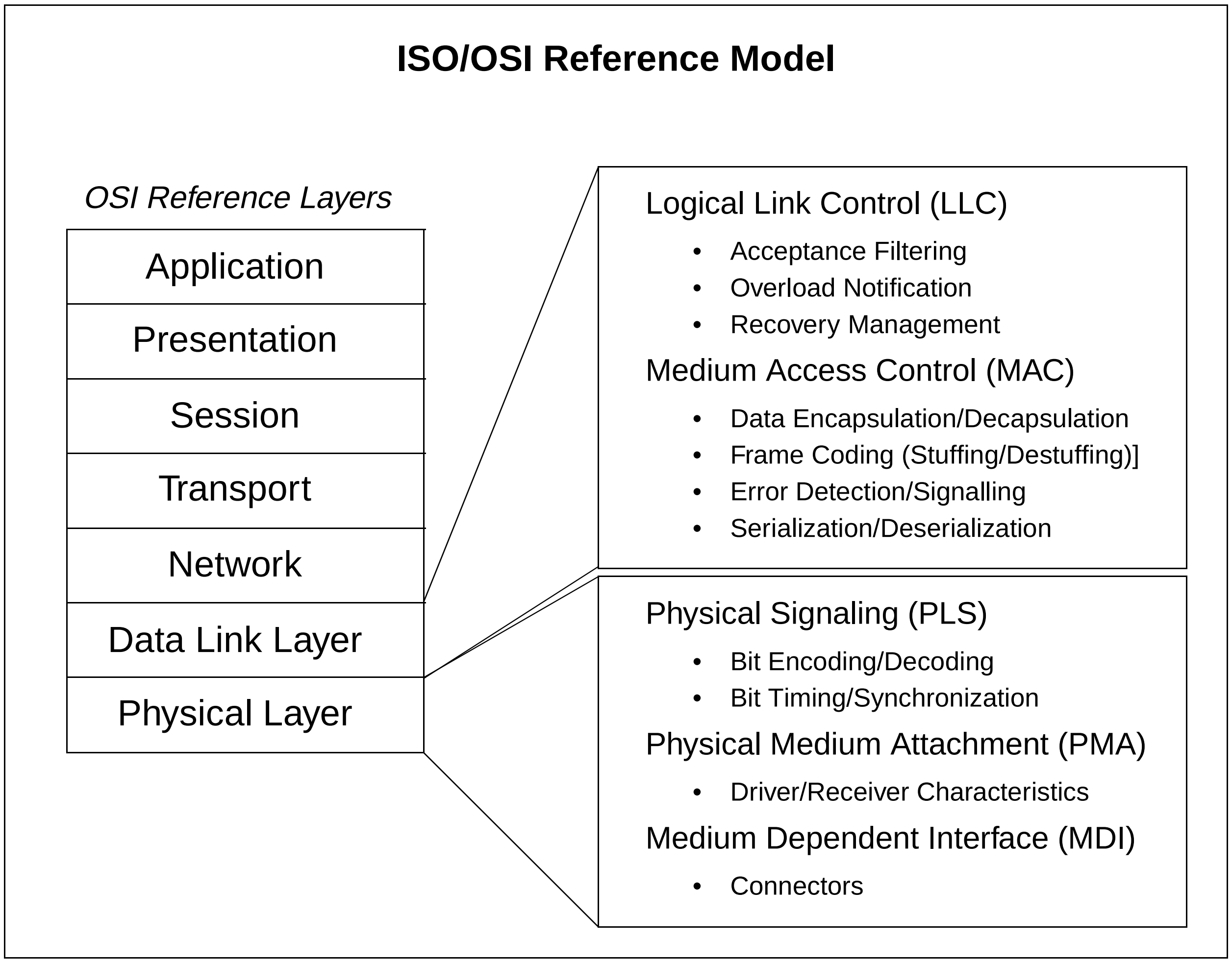
ISO/OSI Reference Model
OSI Reference Layers
Application
Presentation
Session
Transport
Network
Data Link Layer
Physical Layer
Logical Link Control (LLC)
• Acceptance Filtering
• Overload Notification
• Recovery Management
Medium Access Control (MAC)
• Data Encapsulation/Decapsulation
• Frame Coding (Stuffing/Destuffing)]
• Error Detection/Signalling
• Serialization/Deserialization
Physical Signaling (PLS)
• Bit Encoding/Decoding
• Bit Timing/Synchronization
Physical Medium Attachment (PMA)
• Driver/Receiver Characteristics
Medium Dependent Interface (MDI)
• Connectors

FIGURE 1: *ISO/OSI Reference Model*

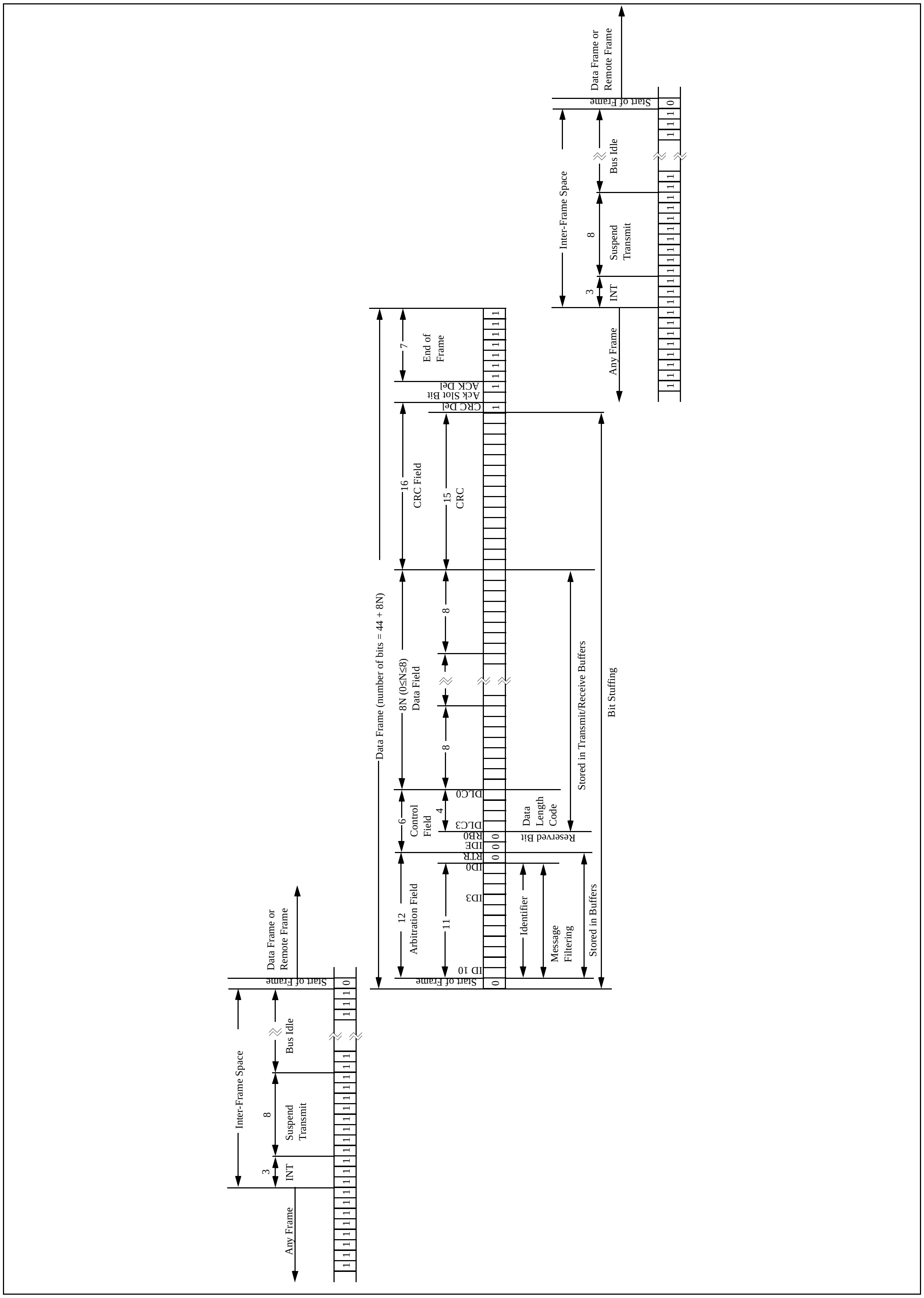

FIGURE 2: *Standard Data Frame*

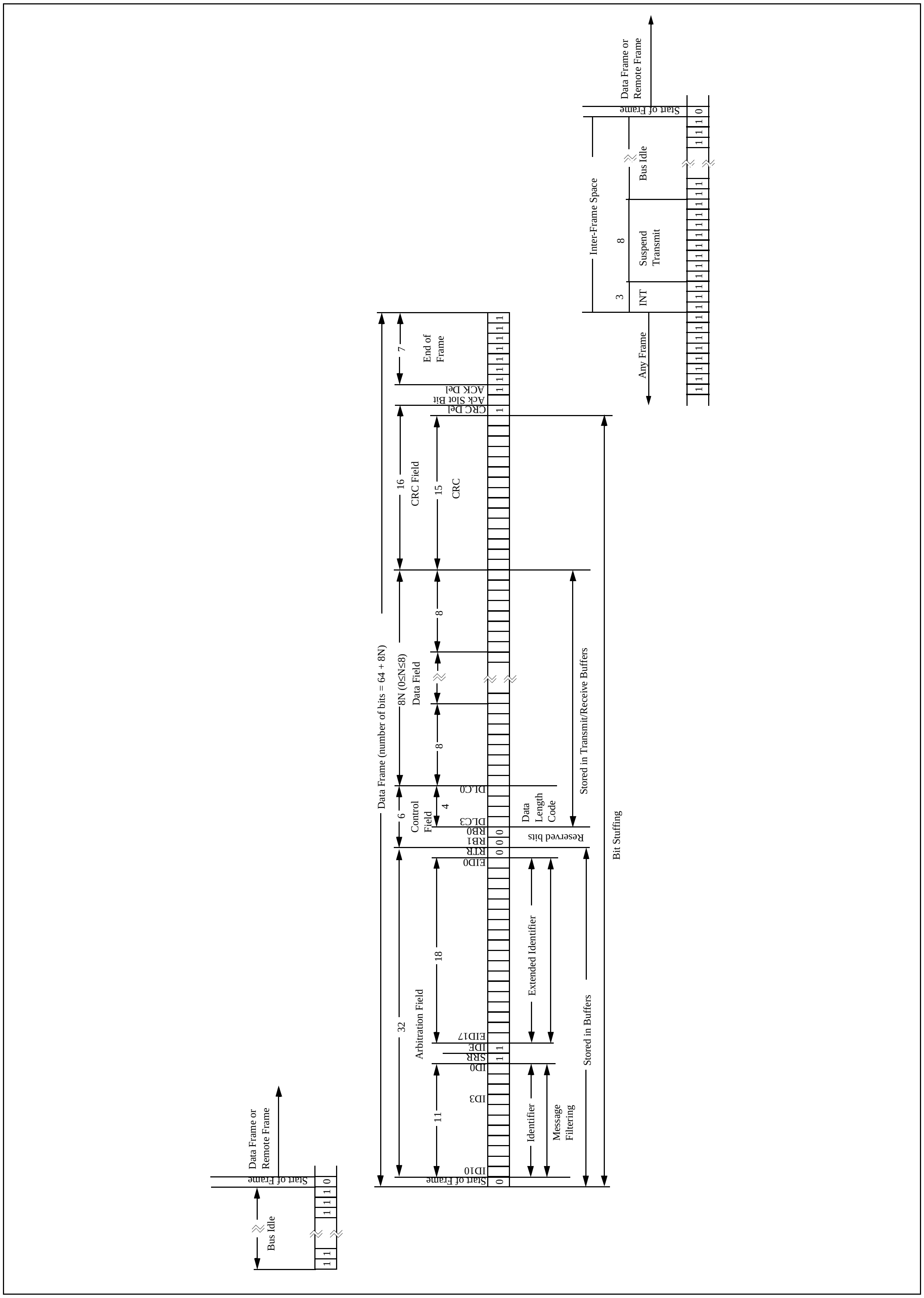

FIGURE 3: *Extended Data Frame*

MICROCHIP

Worldwide Sales and Service

AMERICAS

Corporate Office
Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-786-7200 Fax: 480-786-7277
Technical Support: 480-786-7627
Web: http://www.microchip.com

Atlanta
Microchip Technology Inc.
500 Sugar Mill Road, Suite 200B
Atlanta, GA 30350
Tel: 770-640-0034 Fax: 770-640-0307

Boston
Microchip Technology Inc.
5 Mount Royal Avenue
Marlborough, MA 01752
Tel: 508-480-9990 Fax: 508-480-8575

Chicago
Microchip Technology Inc.
333 Pierce Road, Suite 180
Itasca, IL 60143
Tel: 630-285-0071 Fax: 630-285-0075

Dallas
Microchip Technology Inc.
14651 Dallas Parkway, Suite 816
Dallas, TX 75240-8809
Tel: 972-991-7177 Fax: 972-991-8588

Dayton
Microchip Technology Inc.
Two Prestige Place, Suite 150
Miamisburg, OH 45342
Tel: 937-291-1654 Fax: 937-291-9175

Detroit
Microchip Technology Inc.
42705 Grand River, Suite 201
Novi, MI 48375-1727
Tel: 248-374-1888 Fax: 248-374-2874

Los Angeles
Microchip Technology Inc.
18201 Von Karman, Suite 1090
Irvine, CA 92612
Tel: 949-263-1888 Fax: 949-263-1338

New York
Microchip Technology Inc.
150 Motor Parkway, Suite 202
Hauppauge, NY 11788
Tel: 516-273-5305 Fax: 516-273-5335

AMERICAS (continued)

San Jose
Microchip Technology Inc.
2107 North First Street, Suite 590
San Jose, CA 95131
Tel: 408-436-7950 Fax: 408-436-7955

Toronto
Microchip Technology Inc.
5925 Airport Road, Suite 200
Mississauga, Ontario L4V 1W1, Canada
Tel: 905-405-6279 Fax: 905-405-6253

ASIA/PACIFIC

Hong Kong
Microchip Asia Pacific
Unit 2101, Tower 2
Metroplaza
Kwai Fong, N.T., Hong Kong
Tel: 852-2-401-1200 Fax: 852-2-401-3431

India
Microchip Technology Inc.
India Liaison Office
No. 6, Legacy, Convent Road
Bangalore 560 025, India
Tel: 91-80-229-0061 Fax: 91-80-229-0062

Japan
Microchip Technology Intl. Inc.
Benex S-1 6F
3-18-20, Shinyokohama
Kohoku-Ku, Yokohama-shi
Kanagawa 222-0033 Japan
Tel: 81-45-471- 6166 Fax: 81-45-471-6122

Korea
Microchip Technology Korea
168-1, Youngbo Bldg. 3 Floor
Samsung-Dong, Kangnam-Ku
Seoul, Korea
Tel: 82-2-554-7200 Fax: 82-2-558-5934

Shanghai
Microchip Technology
RM 406 Shanghai Golden Bridge Bldg.
2077 Yan'an Road West, Hong Qiao District
Shanghai, PRC 200335
Tel: 86-21-6275-5700 Fax: 86 21-6275-5060

ASIA/PACIFIC (continued)

Singapore
Microchip Technology Singapore Pte Ltd.
200 Middle Road
#07-02 Prime Centre
Singapore 188980
Tel: 65-334-8870 Fax: 65-334-8850

Taiwan, R.O.C
Microchip Technology Taiwan
10F-1C 207
Tung Hua North Road
Taipei, Taiwan, ROC
Tel: 886-2-2717-7175 Fax: 886-2-2545-0139

EUROPE

United Kingdom
Arizona Microchip Technology Ltd.
505 Eskdale Road
Winnersh Triangle
Wokingham
Berkshire, England RG41 5TU
Tel: 44 118 921 5858 Fax: 44-118 921-5835

France
Arizona Microchip Technology SARL
Zone Industrielle de la Bonde
2 Rue du Buisson aux Fraises
91300 Massy, France
Tel: 33-1-69-53-63-20 Fax: 33-1-69-30-90-79

Germany
Arizona Microchip Technology GmbH
Gustav-Heinemann-Ring 125
D-81739 München, Germany
Tel: 49-89-627-144 0 Fax: 49-89-627-144-44

Italy
Arizona Microchip Technology SRL
Centro Direzionale Colleoni
Palazzo Taurus 1 V. Le Colleoni 1
20041 Agrate Brianza
Milan, Italy
Tel: 39-39-6899939 Fax: 39-39-6899883

04/05/99