CAN

A brief introduction to

Controller Area Network

Introduction

CAN

- stands for controller area network
- was originally developed for automobiles
- is now used for many applications

Serial Bus

- CAN is a serial bus.
- CAN has robust error-handling.
- The number of nodes is limited only by electrical characteristics.

Physical Layer

- Electrical and mechanical characteristics are not part of the basic CAN specification.
- Single-wire (plus ground) or

  Two-wire differential signaling are popular.
- Several companies make CAN transceiver chips.

Single Channel

- There is a single bit-serial channel for all information. Transmission and reception of data, bit timing, and bus arbitration all take place on what is *logically* a single line.

Structure of a Network

- All nodes are peers.
- Every node receives every message.
- There are no node id’s, so messages do not include source or destination addresses.
- Every message includes a message id.

Message Id

Message id’s serve several purposes and are of supreme importance in CAN.

Messages in a System

- Message id’s are assigned by the system designer.
- Typically, every type of event or command in the system has a unique message id. Parameters are sent in the data field.
- Message id’s should be unique to a single node.

Current Versions of CAN

- CAN 2.0A
- 11-bit message id’s
- “Standard format”

- CAN 2.0B
- 29-bit message id’s
- “Extended format”

Mixing Versions

- Originally, only 11-bit id’s were used.
- Some CAN controllers cannot tolerate 29-bit id’s on the bus.
- “CAN 2.0B passive” means a node can only use 11-bit id’s, but tolerates 29-bit id’s.
- All CAN 2.0B (29-bit) nodes can also handle 11-bit id’s.

Fields in a Message Frame

- Start of frame
- Arbitration field (includes message id)
- Control field
- Data field (0 to 8 bytes)
- CRC field
- ACK field

Message Id

The message id:

- is in the Arbitration field
- is a number
- indicates the meaning of the message
- is the priority of the message

Message Priority

- When more than one node tries to transmit, the highest priority message gets the bus.
- Numerically smaller message id’s have higher priority.

Dominant and Recessive States

- The bus can be in the dominant (0) or recessive (1) state.
- If any node is sending the dominant state, that is the state of the bus.
- A node monitors the state of the bus even when it is transmitting.

Bus Arbitration

- Arbitration for control of the bus takes place while the Arbitration field is being transmitted.
- If a node which is sending a recessive bit sees the bus in the dominant state, it recognizes that another node is transmitting a higher-priority message.

Arbitration Example

Node A	101--------
Node B	10010001100
Node C	100101-----
Bus State	10010001100

Node B wins arbitration.

Implementations

- CAN is implemented by a “CAN Controller” that does almost all the work.
- The CAN controller may be on-chip with the CPU, or a separate part. In either case it is easier to think of it as *logically* separate from the CPU.

Additional Information

- www.keil.com/can contains links to a small selection of very good CAN pages.

- CAN mail list: For information, send an email with "help" as its body to can-request@cichlid.com.