CAN Node using HCS12

Ketan Kulkarni, Siddharth Dakshindas
Electrical and Computer Engineering
Oakland University, Rochester, MI , USA

Dec 02, 2014

Outline

- **Brief Introduction of CAN**
- **CAN in HCS12**
- **Project Setup and Details**
- **Demonstration**

Oakland UNIVERSITY

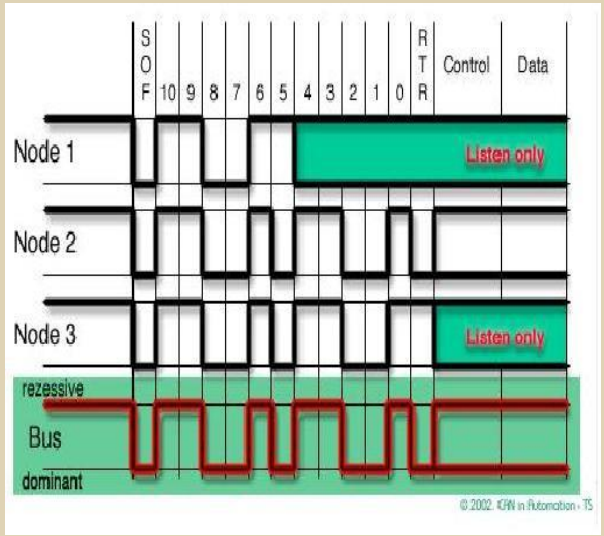
Brief Introduction of CAN

- Message Based Protocol which facilitates communication between various devices
- Originally was designed for automotive applications but is widely used in industrial automation, home appliances, etc.
- CAN Frame Format:

Oakland UNIVERSITY

Brief Introduction of CAN (contd…)

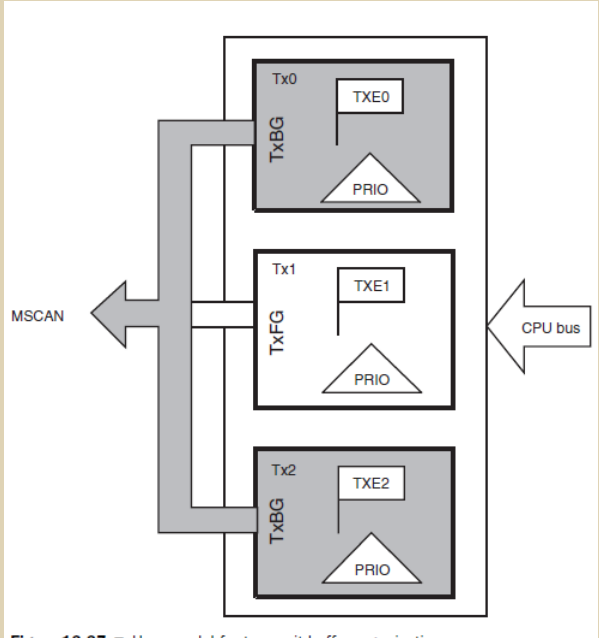
- Messages are distinguished by message identifiers
- The identifier is unique to the network and helps define the priority of the message
- Access conflicts on the bus are resolved by a “wired and” mechanism, where the dominant state overwrites the recessive state



Oakland UNIVERSITY

CAN in HCS12

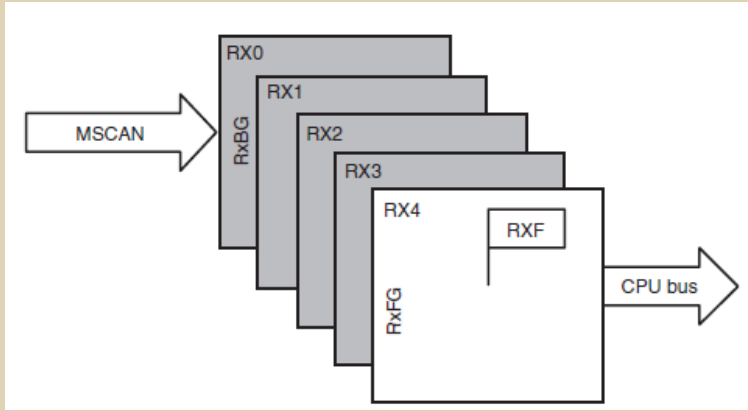
- HCS12's MSCAN12 module supports CAN 2.0A/B. Also supports standard and extended data frames with a programmable bit rate up to 1 Mbps
- The MSCAN12 has a triple transmit buffer scheme which allows multiple messages to be set up in advance and achieve an optimized performance



Oakland UNIVERSITY

CAN in HCS12 (Contd…)

- The received messages are stored in a five stage input FIFO



- The foreground buffer is accessed to read the received message
- The background buffers hold incoming CAN Messages
- The CPU is interrupted to read the message only when the message ID passes the identifier acceptance filter.

Oakland UNIVERSITY

CAN in HCS12 (Contd…)

- **Transmitting Scheme:**

1. Check to see if any of Tx Buffers are empty (by reading CAN***x***TFLG)
2. Select the lowest of the 3 empty buffers for Tx (by writing CAN***x***TFLG to CAN***x***TBSEL)
3. Load the ID, Data, Data Length & Internal Priority in CAN***x***TXIDR0, CAN***x***TXDSR0, CAN***x***TXDLR & CAN***x***TXTBPR respectively
4. Flagging the buffer as ready by clearing the associated TXE flag (by writing a 1 to the buffer register that is free, that is, writing 1 to corresponding buffer of CAN***x***TFLG)

Oakland UNIVERSITY

CAN in HCS12 (Contd…)

- **Receiving Scheme:**
  1. Setup Identifier Acceptance Filter Mode (we have set it to four 16-bit acceptance filters mode by setting CAN0IDAC = 0x10)
  2. Identifier & Mask Registers:

     Mask Registers (CANxIDMR0):

     If 1 then corresponding bit NOT used for ID filtering

     If 0 then corresponding bit IS used for ID filtering

     Identifier Acceptance Registers (CANxIDAR0):

     11-bit identifiers (for standard) setup as B5-B7 in IDAR0 and B0-B7 in IDAR1
  3. Setup interrupt for Rx CAN message processing

Oakland UNIVERSITY

Project Description

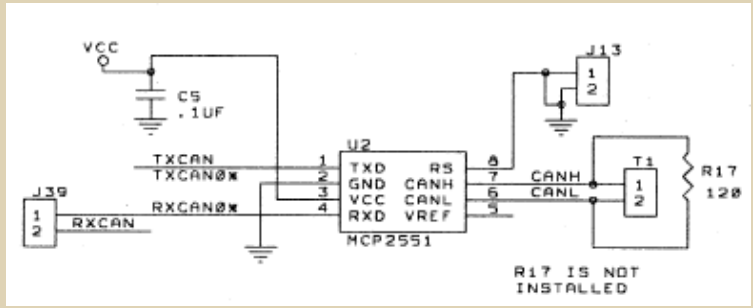
- **Project Objective:** To Demonstrate the creation and working of CAN Node
- The project was carried out in 3 steps:
  1. Setup hardware for CAN
  2. Put CAN module in “loop-back mode”
  3. Setup hardware and software for 2 Dragon-12 boards

Oakland UNIVERSITY

Project Description (contd…)

- **Hardware Setup:**
  - CAN transceiver is basically used to change the logic levels to be able to drive the bus
  - The Dragon-12 lite board does not contain the CAN Transceiver. Hence, a separate CAN Transceiver (MCP2551) was added to the U2 slot on the board.
  - The pins PM0 and PM1 are RxD and TxD pins on the micro

MCP2551



Oakland UNIVERSITY

Project Description (contd…)

- **Loop-back mode Software Setup:**
  - Software debugging is simplified by putting the HCS12 CAN Module in "loop-back" mode. In Loop-back mode, no messages are seen on the bus.
  - The register CAN0CTL1 is set to "0xA0" which enables and sets the CAN module in loop-back mode
  - As per convention, the following times are set:
    - Synchronization Segment = 1 Time Quanta
    - Timing Segment 1 (Prop. + Phase Buffer1) = 11 Time Quanta
    - Timing Segment 2 (Phase Buffer2) = 4 Time Quanta
  - Thus, CAN0BTR1 = 0x3A

Oakland UNIVERSITY

Project Description (contd…)

- **Loop-back mode Software Setup (Contd…):**
  - PreScalar Calculation:
    - Prescalar is given by the formula

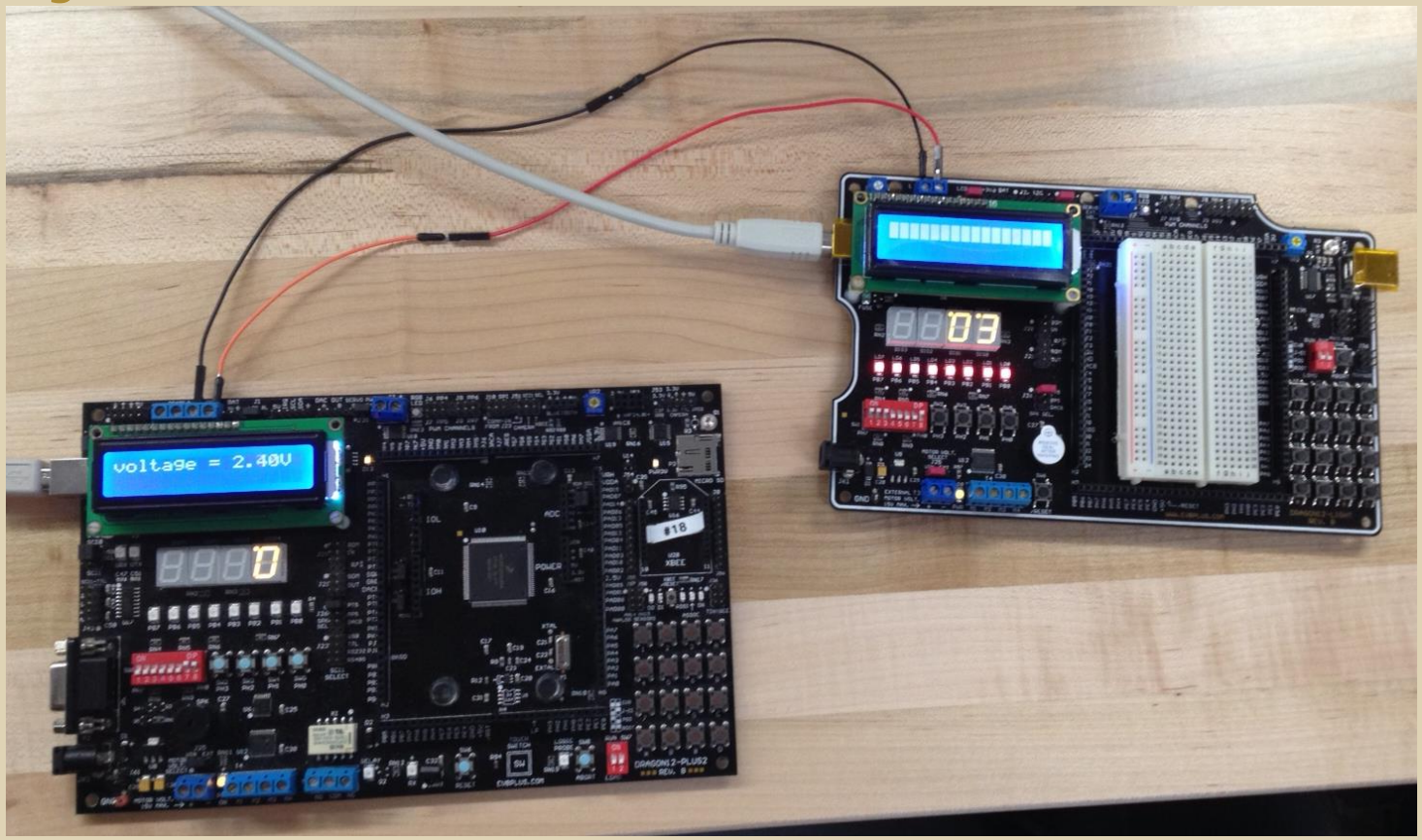
$$\text{Bit Time} = \frac{(\text{Prescale value})}{f_{CANCLK}} \bullet (1 + \text{Time Segment 1} + \text{Time Segment 2})$$

    - We need the baud rate to be 125KHz. Since the Dragon-12 crystal is 8 MHz, fCANCLK = 8M, (1+ Time Segment 1 + Time Segment 2) = 16, Bit Time = (1/125K)
    - Thus Prescalar required = 4
    - For this value of Prescalar, we need to set CAN0BTR0 = 0xC3 (assuming Sync Jump width to be 4 Time Quanta)

Oakland UNIVERSITY

Project Description (contd…)

- **Hardware Setup with 2 Dragon-12:**
  - Once the loop-back test was successful, we connected the 2 Dragon-12 boards



Oakland UNIVERSITY

Project Description (contd…)

- **Software Setup with 2 Dragon-12:**
  - 2 CAN messages with IDs 0x100 and 0x200 are transmitted by CAN Node#1 and CAN Node#2 respectively.
  - The Masked ID and Acceptance filter configuration are as follows:

Message ID	0x100	0x200
Mask Register (CANxIDMR0)	0x00	0x00
Mask Register (CANxIDMR1)	0x07	0x07
Identification Acceptance Register (CANxIDAR0)	0x20	0x40
Identification Acceptance Register (CANxIDAR1)	0x00	0x00

Oakland UNIVERSITY

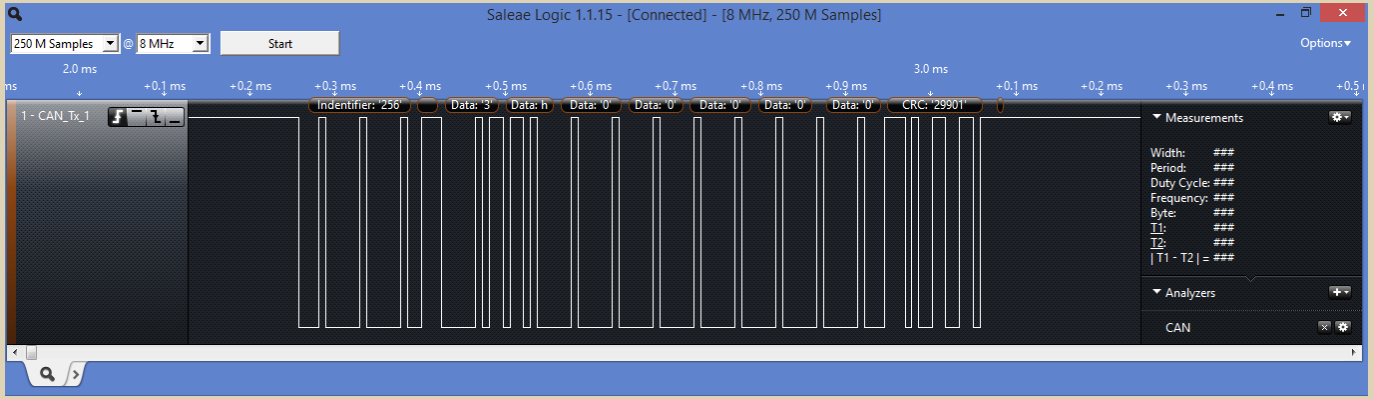
CAN in HCS12 (Contd…)

- **Software Setup with 2 Dragon-12 (Contd…):**
  - 0x100 is transmitted cyclically every 15ms and 0x200 is transmitted cyclically every 10ms. We used a Real Time Interrupt for interrupting every 10ms and 15ms on CAN Node#1 and CAN Node#2 respectively.
  - An CAN receive message interrupt (Interrupt 38) is fired on both nodes when they receive their respective messages.
  - The message ID 0x100 carries the value of the variable resistor from Node#1 and that value is displayed on the LCD on Node#2
  - The message ID 0x200 carries the DIP switch status from Node#2 and its value is displayed on 7-seg display on Node#1
  - Only 1 software is flashed on both boards with only the change for accommodating the ID. This is managed using pre-compile switches ("#ifdef SECOND_NODE" when flashing on CAN Node#2)

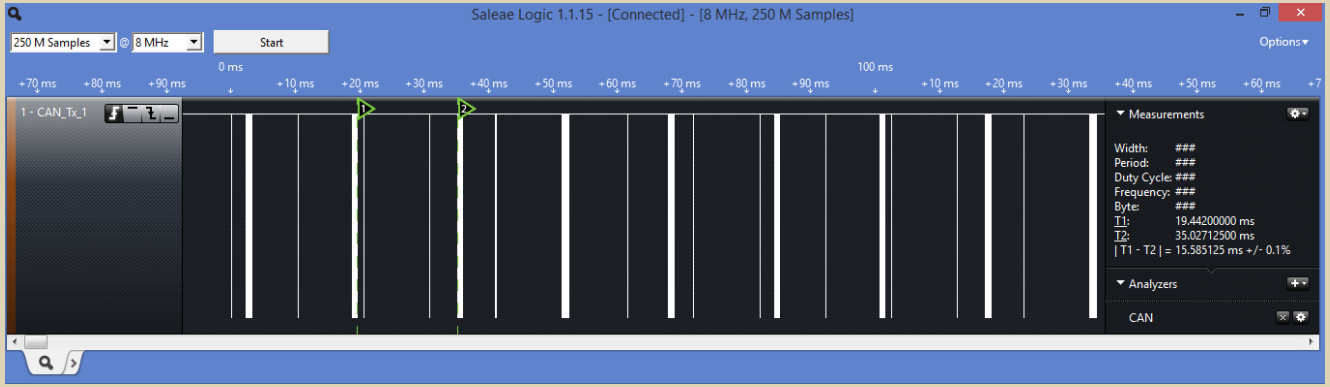
Oakland UNIVERSITY

CAN in HCS12 (Contd…)

- **Software Setup with 2 Dragon-12 (Contd…):**
  - 0x100 Message on Logic Analyzer:



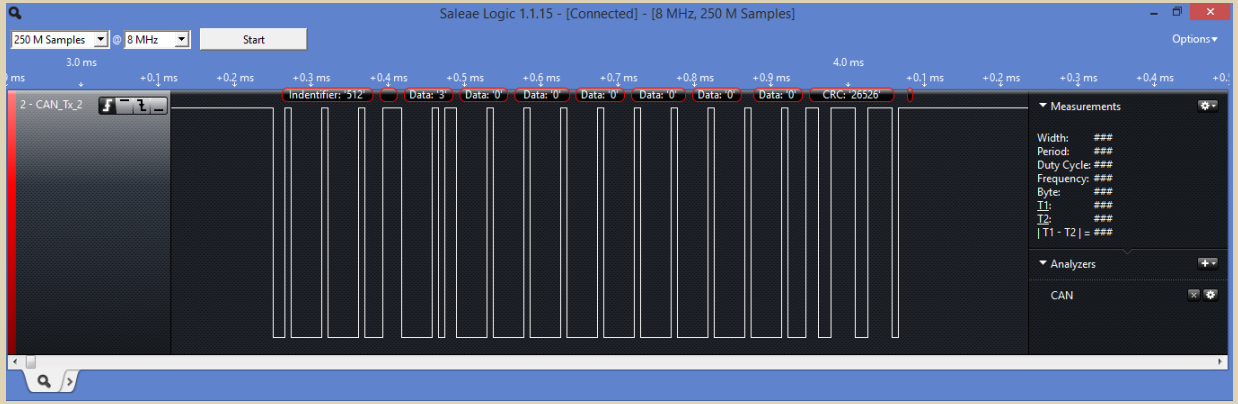
  - Message 0x100 sent cyclically every 15ms:



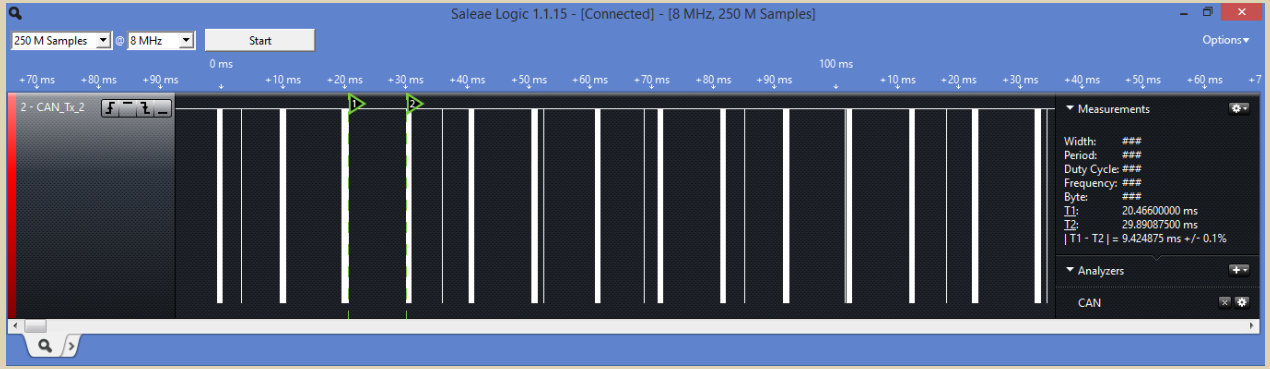
Oakland UNIVERSITY

CAN in HCS12 (Contd…)

- **Software Setup with 2 Dragon-12 (Contd…):**
  - 0x200 Message on Logic Analyzer:



  - 0x200 message sent every 10ms:



Oakland UNIVERSITY

• Demonstration

The Demo video has been uploaded at:
https://www.youtube.com/watch?v=VdNLzp-K8ME&feature=youtu.be

Oakland UNIVERSITY