Controller Area Network (CAN) Tutorial

A Controller Area Network (CAN) bus is a high-integrity serial bus system for networking intelligent devices. CAN busses and devices are common components in automotive and industrial systems. Using a CAN interface device, you can write LabVIEW applications to communicate with a CAN network.

Topics

A. What You Need to Get Started

Tutorial Materials

Before you use this tutorial, ensure you have all the following items:

- ❑ Windows 2000 or later installed on your computer. The tutorial is optimized for Windows XP.
- ❑ Multifunction DAQ device configured as `device 1` using Measurement & Automation Explorer (MAX)
- ❑ LabVIEW Full or Professional Development System 2009 or later
- ❑ A CAN Interface
- ❑ NI-CAN 2.6.3 or later
- ❑ Exercises folder for saving VIs created during the tutorial and for completing certain tutorial exercises
- ❑ Solutions folder containing the solutions to all the tutorial exercises

B. History of CAN

In the past few decades, the need for improvements in automotive technology caused increased usage of electronic control systems for functions such as engine timing, anti-lock brake systems, and distributorless ignition.

Originally, point-to-point wiring systems connected electronic devices in vehicles. More and more electronics in vehicles resulted in bulky wire harnesses that were heavy and expensive. To eliminate point-to-point wiring, automotive manufacturers replaced dedicated wiring with in-vehicle networks, which reduced wiring cost, complexity, and weight. In 1985, Bosch developed the Controller Area Network (CAN), which has emerged as the standard in-vehicle network.

CAN provides a cheap, durable network that allows the devices to speak through the Electronic Control Unit (ECU). CAN allows the ECU to have one CAN interface rather than analog inputs to every device in the system. This decreases overall cost and weight in automobiles. Each of the devices on the network has a CAN controller chip and is therefore intelligent. All transmitted messages are seen by all devices on the network. Each device can decide if the message is relevant or if it can be filtered.

As CAN implementations increased in the automotive industry, CAN (high speed) was standardized internationally as ISO 11898. Later, low-speed CAN was introduced for car body electronics. Finally, single-wire CAN was introduced for some body and comfort devices. Major semiconductor manufacturers such as Intel, Motorola, and Philips developed CAN chips.

By the mid-1990s, CAN was the basis of many industrial device networking protocols, including DeviceNet and CANOpen.

Automotive Applications

Examples of CAN devices include engine controller (ECU), transmission, ABS, lights, power windows, power steering, instrument panel, and so on.

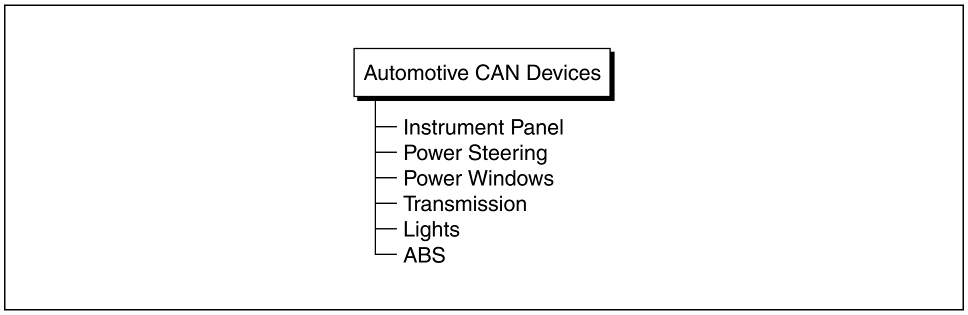


Figure 1-1. Automotive CAN Devices

Other CAN Markets

Comparing the requirements for automotive and industrial device networks showed the following similarities:

- the transition away from dedicated signal lines
- low cost
- resistance to harsh environments
- real-time capabilities

These similarities led to the use of CAN in industrial applications such as textile machinery, packaging machines, and production line equipment such as photoelectric sensors and motion.

With its growing popularity in automotive and industrial applications, CAN has been increasingly used in a wide variety of applications. Usage in systems such as agricultural equipment, nautical machinery, medical apparatus, semiconductor manufacturing equipment, avionics, and machine tools validate the versatility of CAN.

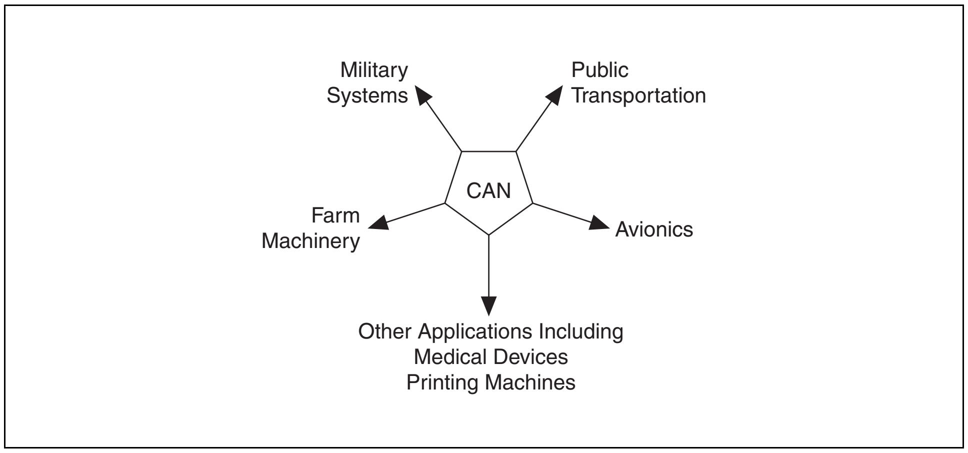

Military Systems
Public Transportation
CAN
Farm Machinery
Avionics
Other Applications Including Medical Devices Printing Machines

Figure 1-2. Other CAN Markets

C. CAN Basics

Benefits of CAN

- Lower cost from reduced wiring compared to two wire, point-to-point wiring
- Highly robust protocol
  - Built-in determinism
  - Fault tolerance
  - Reliable—More than a decade of use in the automotive industry

CAN Specifications

- CAN data (up to 8 bytes in a frame)
- Maximum 1 Mbaud/s
- 40 Meters at 1 Mbaud/s
- 6 km at 10 kbaud/s
- Theoretical maximum of 2,032 nodes per bus
  - Practical limit is approximately 100 nodes due to transceiver
  - Most buses use 3–10 nodes
- Data fields—Arbitration ID (11 bit or 29 bit)
  - Indicates message priority

Types of CAN

- High Speed CAN
  - Up to 1 M bits/s transmission rate

- Low Speed/Fault-Tolerant CAN
  - Up to 125 kbaud/s transmission rate
  - Fault tolerant
- Single Wire
  - Up to 83.3 kbaud/s
  - High-voltage pulse to wakeup sleeping devices

NI CAN Interface Devices

National Instruments provides four types of CAN interface devices that use the NI-CAN API described in this appendix. Each category of devices supports multiple form factors and is available in one or two port varieties.

High-speed CAN

- 1 and 2 ports
- Maximum baud rate of 1Mb/s

Low-speed CAN

- 1 and 2 ports
- Maximum baud rate of 125 kbaud/s

Software Selectable CAN

- 1 and 2 ports (each port can be used as high-speed, low-speed, or single-wire CAN)

Single Wire CAN

- 1 and 2 ports
- Maximum baud rate of 83.3 kbaud/s

CAN Frame

CAN devices send data across the CAN Network on packets called frames. A typical CAN frame contains an arbitration ID, a data field, a remote frame, an error frame, and an overload frame.

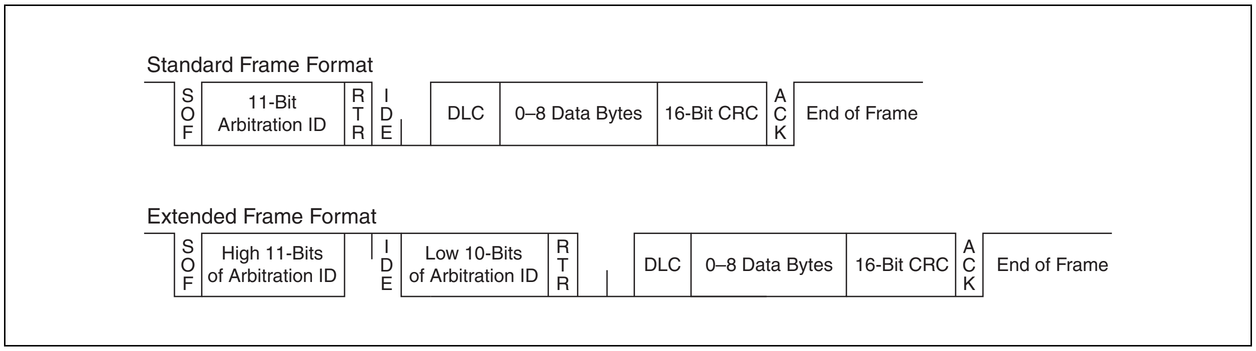


Figure 1-3. Standard and Extended CAN Frames

Arbitration ID

The arbitration ID determines the priority of the messages on the bus. If multiple nodes try to transmit a message onto the CAN bus at the same time, the node with the highest priority (lowest arbitration ID) automatically gets bus access. Nodes with a lower priority must wait until the bus becomes available before trying to transmit again. The waiting devices wait until the end of the frame section is detected.

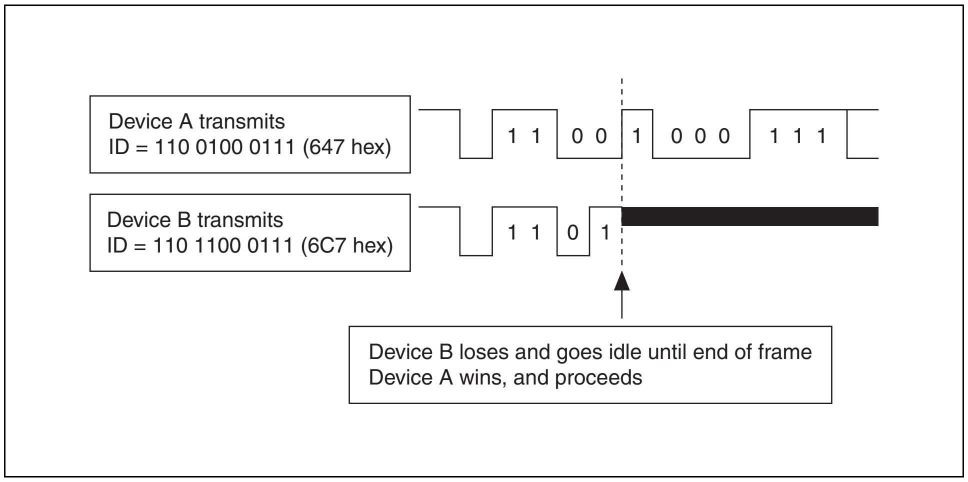

Device A transmits
ID = 110 0100 0111 (647 hex)
1 1 0 0 1 0 0 0 1 1 1
Device B transmits
ID = 110 1100 0111 (6C7 hex)
1 1 0 1
Device B loses and goes idle until end of frame
Device A wins, and proceeds

Figure 1-4. CAN Arbitration

Data Field

The data field contains the actual data being transmitted. The CAN protocol supports two data field formats as defined in the Bosch Version 2.0 specifications, the essential difference being in the length of the arbitration ID. In the standard frame format (also known as 2.0A), the length of the ID is 11 bits. In the extended frame format (also known as 2.0B), the length of the ID is 29 bits.

In NI-CAN terminology, a data field for which the individual fields are described is called a message. A message can contain one or more channels, which contain the data and other relevant information. In Figure 1-5, the ABS message contains two channels—Temperature and Torque.

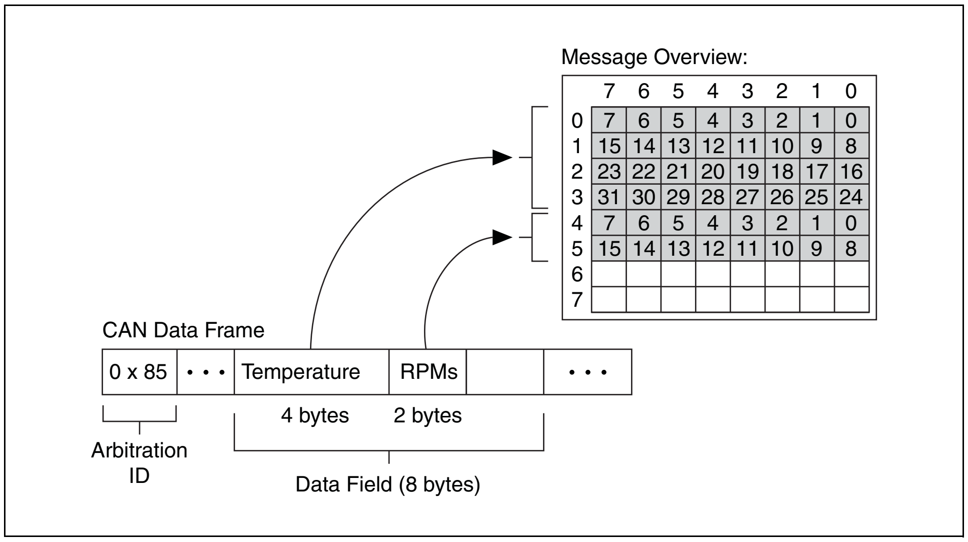


Figure 1-5. CAN Message with Two Channels

D. Channel Configuration

You can configure Channels in MAX to enable LabVIEW to interpret your data fields. MAX allows you to load a set of channels from a database file or configure channels manually.

CAN Databases

To translate the data field into usable data, a CAN device comes with a database that describes the channels contained in the message. A CAN Database File (CANdb File) is a text file that contains this information. It allows you to find the data in a frame and convert it to engineering units. For each channel, CAN databases store the following data:

```
Channel name
Location (Start bit) and size (number of bits) of the
Channel within a given Message
Byte Order (Intel/Motorola)
Data type (signed, unsigned, and IEEE float)
Scaling and units string
Range
Default Value
Comment
```

With many software programs, you must manually convert the data field to usable data using the information contained in the database file. However, using National Instruments software, this conversion is done for you. You can load the channel configuration from the file into MAX, and then use it in your application through the NI-CAN driver.

Loading Channels From a Database

To import channel configurations from a Vector CANdb file into MAX, right-click the **CAN Channels** heading and select **Import from CANdb File**. <Shift>-click to select multiple channels, then select **Import**. If you need to select another set, you can select the channels and then import them again. When you finish importing, click **Done** to return to MAX.

Note You also can access a CAN database directly from the LabVIEW CAN API. However, loading the channels into MAX prevents you from having to specify a path in your LabVIEW programs and allows you to read and write from the channels using the MAX test panels.

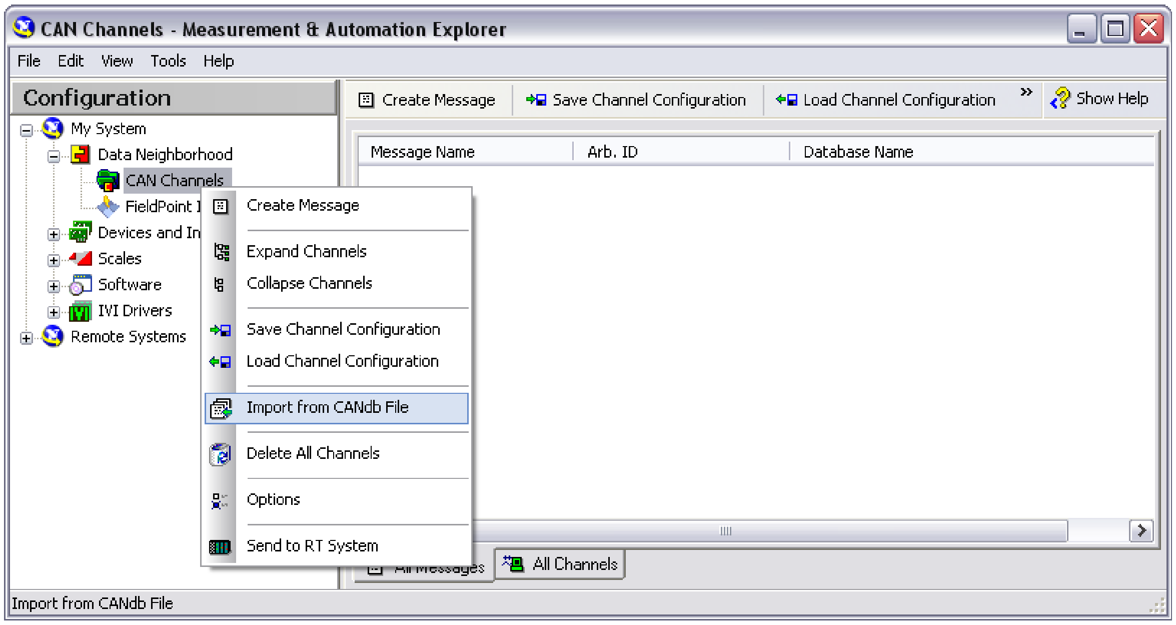


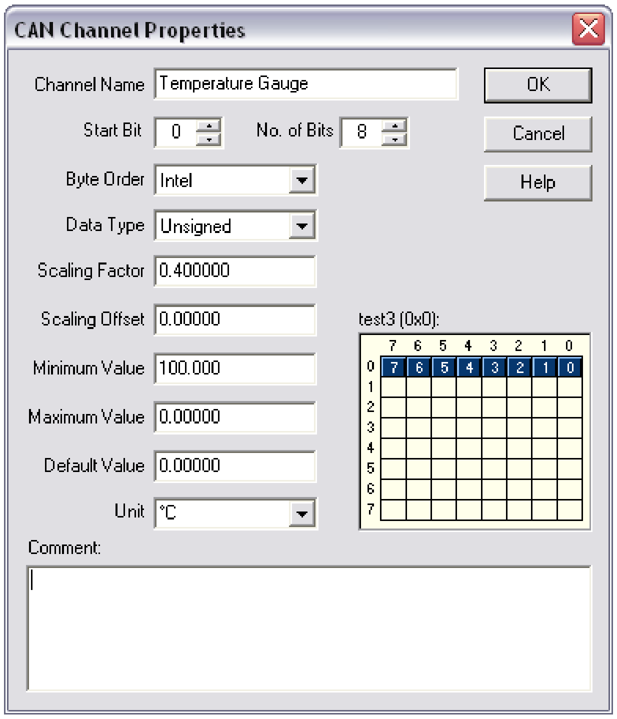
Figure 1-6. Importing a CAN Database File In MAX

Explicitly Creating a Channel

If no database file exists, you can create channels in MAX. Complete the following steps to create channels within MAX:

1. Right-click the **CAN Channels** heading under **Data Neighborhood** and select **Create Message**.
2. Enter the message properties and click **OK**. After the message is created, it appears under CAN Channels.
3. Right-click the message name and select **Create Channel**. A configuration box appears similar to the one shown in Figure 1-7. As you enter information about Start Bit and No. of Bits, the matrix is populated with dimmed boxes to indicate the bits that have already been

used in channel definitions for that message. Blue boxes indicate the bits that are currently being defined.


CAN Channel Properties

Channel Name: Temperature Gauge
Start Bit: 0
No. of Bits: 8
Byte Order: Intel
Data Type: Unsigned
Scaling Factor: 0.400000
Scaling Offset: 0.00000
Minimum Value: 100.000
Maximum Value: 0.00000
Default Value: 0.00000
Unit: °C
Comment:

test3 (0x0):

Figure 1-7. Explicit Channel Configuration in MAX

4. Enter the channel properties and click the **OK** button.
5. Right-click and select **Create Channel** again for each channel contained in the message.
6. To save channel configurations to a file, right-click the **CAN Channels** heading and select **Save Channel Configuration**.

Saving the channel configuration creates a custom database file for your device. The resulting NI-CAN database file uses file extension `.ncd`. You can access the NI-CAN database just like any other CAN database. By installing the NI-CAN database file along with your application, you can deploy your application to a variety of users.

You can also test explicitly created channels using the Channel Test Panel.

E. CAN APIs

There are two APIs that you can use with NI-CAN hardware: Channel and Frame.

- Channel API
  - High level

  - Easy-to-use physical units
  - Easy CAN/DAQ synchronization
  - Not compatible with NI-CAN 1.6 or lower
- Frame API
  - Lower-level, advanced API
  - Command/response protocol
  - Advanced CAN/DAQ synchronization
  - Compatible with all versions of NI-CAN

For a single NI-CAN port such as CAN0, you can use only one API at a time. For example, if you have one application that uses the Channel API and another application that uses the Frame API, you cannot use CAN0 with both at the same time. If you have a 2-port CAN card, you can connect CAN0 and CAN1 to the same CAN network, then use CAN0 with one application and CAN1 with the other. Alternately, you can use CAN0 for both applications, but run each application at a different time. In most cases, you should use the Channel API, because it simplifies programming and reduces development time. However, there are some situations in which the Frame API is necessary; for example:

- You are maintaining an application developed with NI-CAN version 1.6 or earlier. The Frame API is compatible with code developed in early CAN versions.
- You need to implement a command/response protocol in which you send a command to the device, and then the device replies by sending a response. Command/response protocols typically use a fixed pair of IDs for each device, and the ID does not determine the meaning of the data bytes.
- Your devices require use of remote frames. The Channel API does not provide support for remote frames, but the Frame API has extensive features to transmit and receive remote frames.
- You are synchronizing the CAN communication with data acquisition from a DAQ card. The Frame API provides lower-level RTSI features than those provided with the Channel API, and therefore is better for advanced synchronization.
- You are using one of the NI USB-847x CAN Interfaces. The USB-CAN products do not support Channel API or CAN Objects. You can still use Frame API but there are some limitations on the functions available for USB-CAN. For a list of these functions, see the Related Links section below.

Note This tutorial covers only the Channel API.

F. CAN Programming in LabVIEW (Channel API)

The basic NI-CAN program consists of an initialization and start, a read or write data, and a clear. A Get Names VI is also frequently used to access the list of channels names in a database.

The link between the functions is the task reference. A CAN task is a collection of CAN channels that have identical timing and the same communication direction—read or write. A task can encompass several messages, but they must all be on the same interface or port.

CAN Init Start VI

The CAN Init Start VI initializes a task for the specified channel list and starts communication. The mode input determines whether the task is configured to read or write.

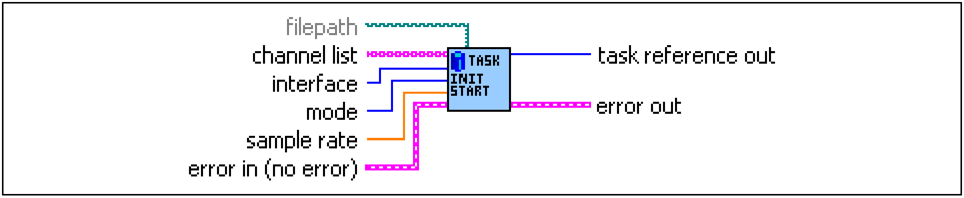


Figure 1-8. CAN Init Start VI

CAN Get Names VI

The VI gets an array of CAN channel names or message names from MAX or a CAN database file. If you leave the **file path** input unwired, the channel names are retrieved from MAX. Otherwise, they are retrieved from the database file you specify. The **mode** input determines whether you are accessing channel names or message names.

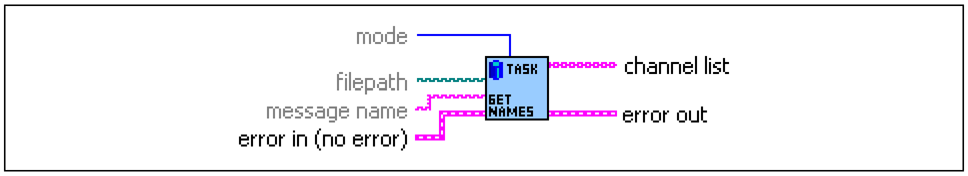


Figure 1-9. CAN Get Names VI

There are three ways to access channels in your application.

- specify a channel name that has been imported into MAX
- specify a database file and channel name for channels not in MAX
- use the CAN Get Names VI to access all channels in a database file

To directly access a CAN channel from a CAN database, specify the channel name with the database path as a prefix. For example, if you use a channel named `Switch0` in the `C:\CAN Demo Box.DBC` CAN Database, pass `C:\CAN Demo Box.DBC::Switch0` to the CAN Init Start VI, as shown in Figure 1-10. This figure also demonstrates how to read a channel available in MAX, and how to extract all channels from a database file.

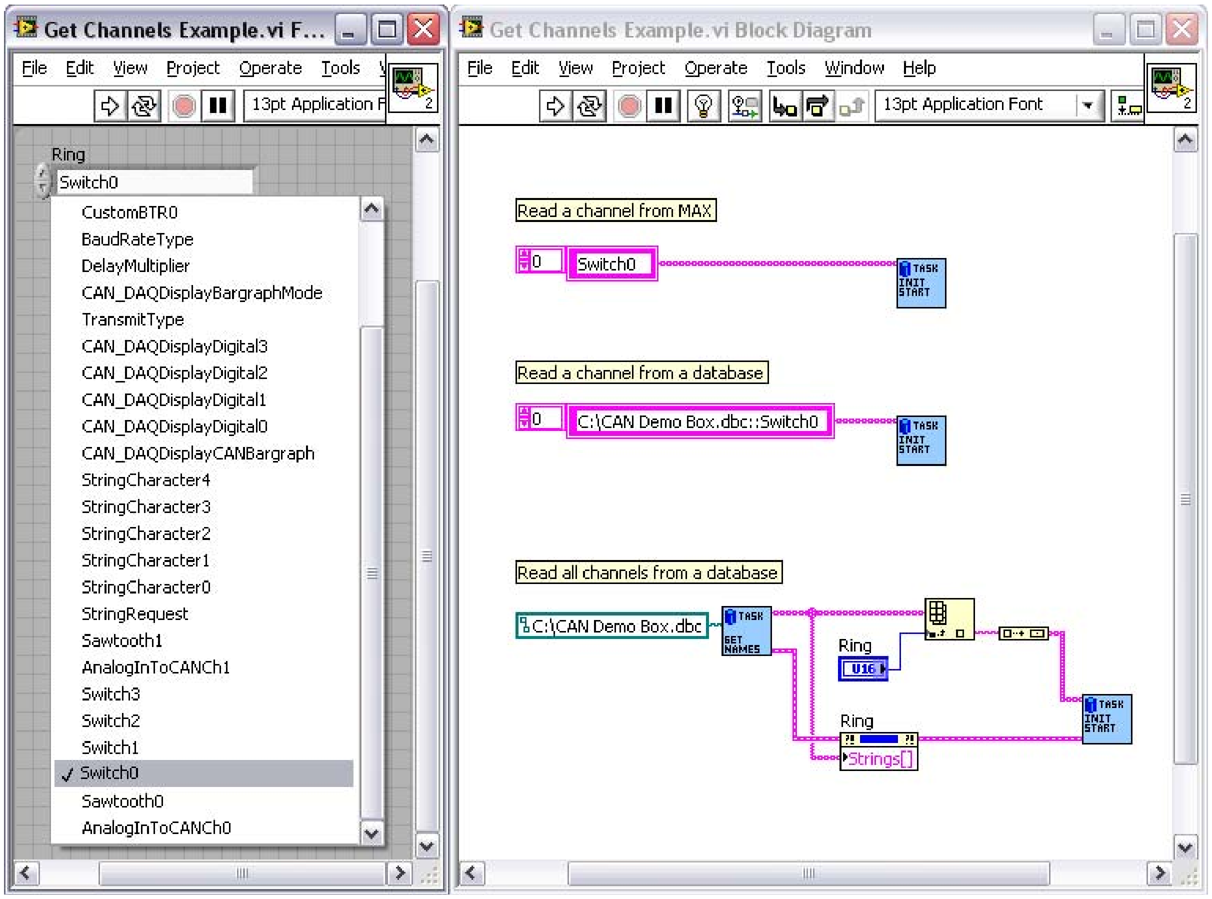


Figure 1-10. Specifying Channels in LabVIEW

CAN Read

The CAN Read VI reads samples from an input CAN task. Samples are obtained from received CAN messages. Right-click the icon and choose **Select Type** from the shortcut menu to select the input data type.

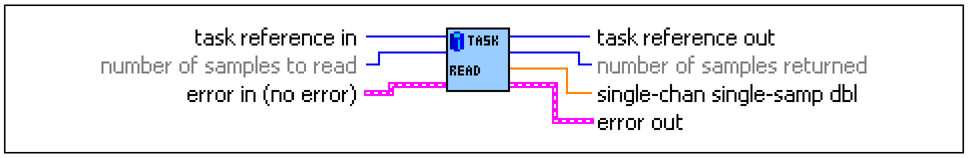

task reference in
number of samples to read
error in (no error)
TASK
READ
task reference out
number of samples returned
single-chan single-samp dbl
error out

Figure 1-11. CAN Read VI

CAN Write

The CAN Write VI writes samples to an output CAN task. Samples are placed into transmitted CAN messages. Right-click the icon and select **Select Type** from the shortcut menu to choose the write data type.

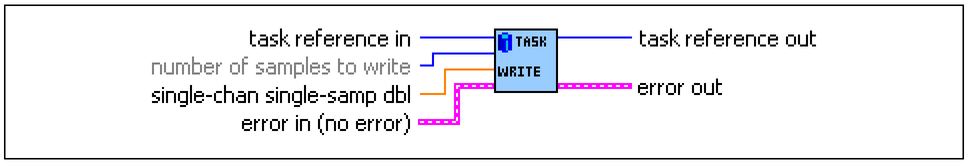


Figure 1-12. CAN Write VI

CAN Clear

The CAN Clear VI stops communication for the task and clears the configuration.

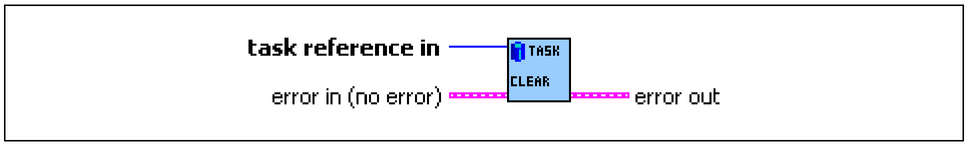


Figure 1-13. CAN Clear VI

Self Review: Quiz

1. Which of the following is the maximum transmission rate of Low Speed or Fault-tolerant CAN?
   a. 1 Mbaud/s
   b. 83.3 kbaud/s
   c. 256 kbaud/s
   d. 125 kbaud/s

2. NI-CAN channels are used for which one of the following reasons?
   a. They allow associating bits/bytes of a CAN message/frame with meaningful names and scaling information.
   b. They allow access to the CAN message/frame in its entirety
   c. They allow access to the CAN physical bus channel
   d. They allow associating different types of CAN frames with user defined names

3. Which of the following are ways to access CAN channels in your CAN application?
   a. Specify a channel name that has been imported into MAX
   b. Specify a database file and channel name for channels not in MAX
   c. Use the CAN Get Names VI to access all channels in a database file

Self Review: Quiz Answers

1. Which of the following is the maximum transmission rate of Low Speed or Fault-tolerant CAN?
   a. 1 Mbaud/s
   b. 83.3 kbaud/s
   c. 256 kbaud/s
   d. 125 kbaud/s

2. NI-CAN channels are used for which one of the following reasons?
   a. They allow associating bits/bytes of a CAN message/frame with meaningful names and scaling information
   b. They allow access to the CAN message/frame in its entirety
   c. They allow access to the CAN physical bus channel
   d. They allow associating different types of CAN frames with user defined names

3. Which of the following are ways to access CAN channels in your CAN application?
   a. Specify a channel name that has been imported into MAX
   b. Specify a database file and channel name for channels not in MAX
   c. Use the CAN Get Names VI to access all channels in a database file

Notes

Exercises

Exercise 1-1 Concept: CAN Device Setup

Goal

Connect CAN hardware and use MAX to view and test a CAN device.

Description

The exercises in this tutorial use a 2-port CAN interface device and a CAN Demo Box. In this exercise, you set up your hardware, observe the configuration of the CAN device in MAX, and run a self-test to verify that the device functions correctly.

1. Connect the CAN hardware.

   ❑ Connect **Port 1** on your CAN interface to the **CAN** input on the CAN Demo Box using a CAN cable.

   ❑ Connect the cable from your DAQ device to the 68-pin terminal on the CAN Demo Box.

   ❑ Use a wire to connect the following terminals on the CAN Demo Box.

   – **Gen** terminal of the **Function Generator**

   – **Ch 0** terminal of the **Analog In to CAN**

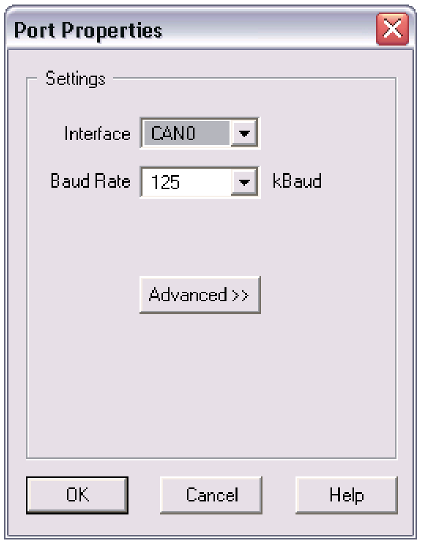
2. Explore the CAN device and ports in MAX.

   ❑ Launch MAX.

   ❑ Expand **Devices and Interfaces**. The PCI-CAN/2 device is your CAN Interface Device.

   ❑ Expand **PCI-CAN/2**. CAN0 and CAN1 are physical ports on your CAN device. A single port interface has only one item.

   ❑ Select **CAN0** and click the **Properties** button to display the **Port Properties** dialog box.



Interface CAN0

Baud Rate 125 kBaud

Figure 1-14. Port Properties Dialog Box

Note The **Port Properties** dialog box allows you to change the baud rate and other properties of the CAN Interface. The settings are already correct for the current configuration. You also can access the **Port Properties** dialog box by right-clicking a port and selecting **Properties** from the shortcut menu.

❑ Click **OK** to close the dialog box.

3. Test the CAN Interface Device.

❑ Select **PCI-CAN/2** and observe the value of the **Test Status** property. The Test Status property displays **Untested** until you execute a self-test.

❑ Click the **Self-test** button located above the list of properties.

❑ Observe the value of the Test Status property after the test completes. Also notice that a small blue circle with a white check appears above the bottom right corner of the PCI-CAN/2 icon in the Configuration tree, indicating that the device has passed the self-test.

End of Exercise 1-1

Exercise 1-2 Channel Configuration

Goal

Learn how to load and test channels in MAX.

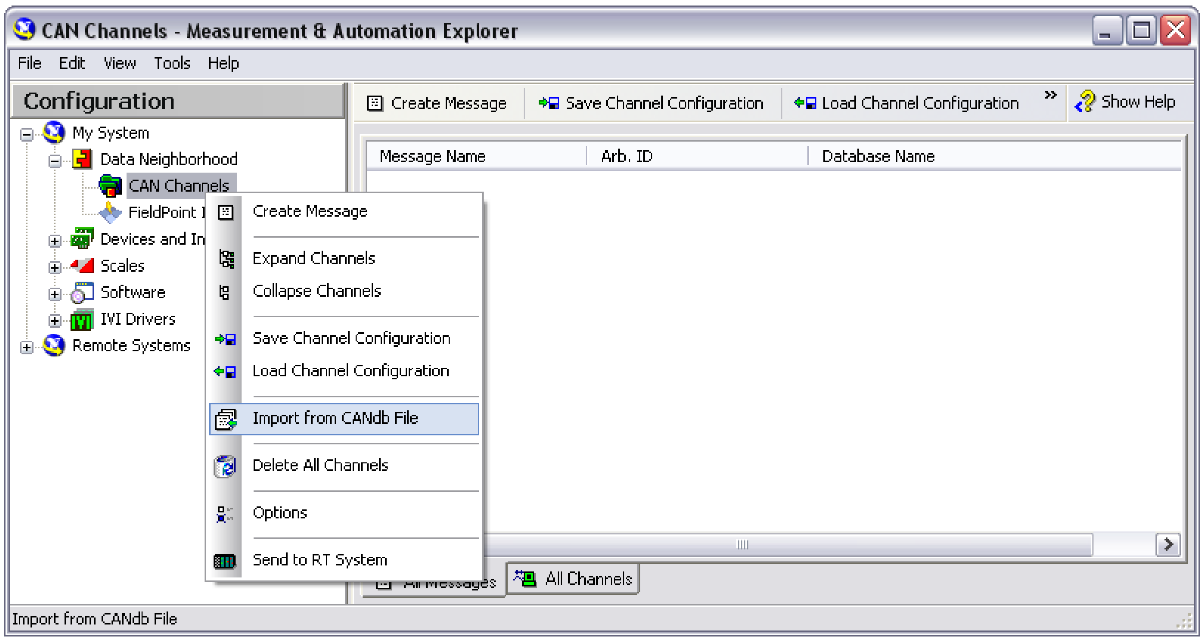
Description

Load channels from either an NI CAN database file or a Vector CAN Database file into MAX. Investigate the properties of messages and channels. Read data on a specific channel using MAX Test Panels. Finally, use the built-in CAN Bus Monitor feature in MAX to monitor the frame data being sent along the CAN bus.

1. Launch MAX and expand **Data Neighborhood**.

2. Load the database file for the CAN Demo Box.

   ❑ Right-click **CAN Channels** and select **Import from CANdb File**.



   ❑ Navigate to `<Exercises>\CAN\CAN Demo Box.dbc`.

Tip You could choose to import the `.ndc` database files instead. To do so, select **Load Channel Configuration** instead of **Import from CAN db File**.

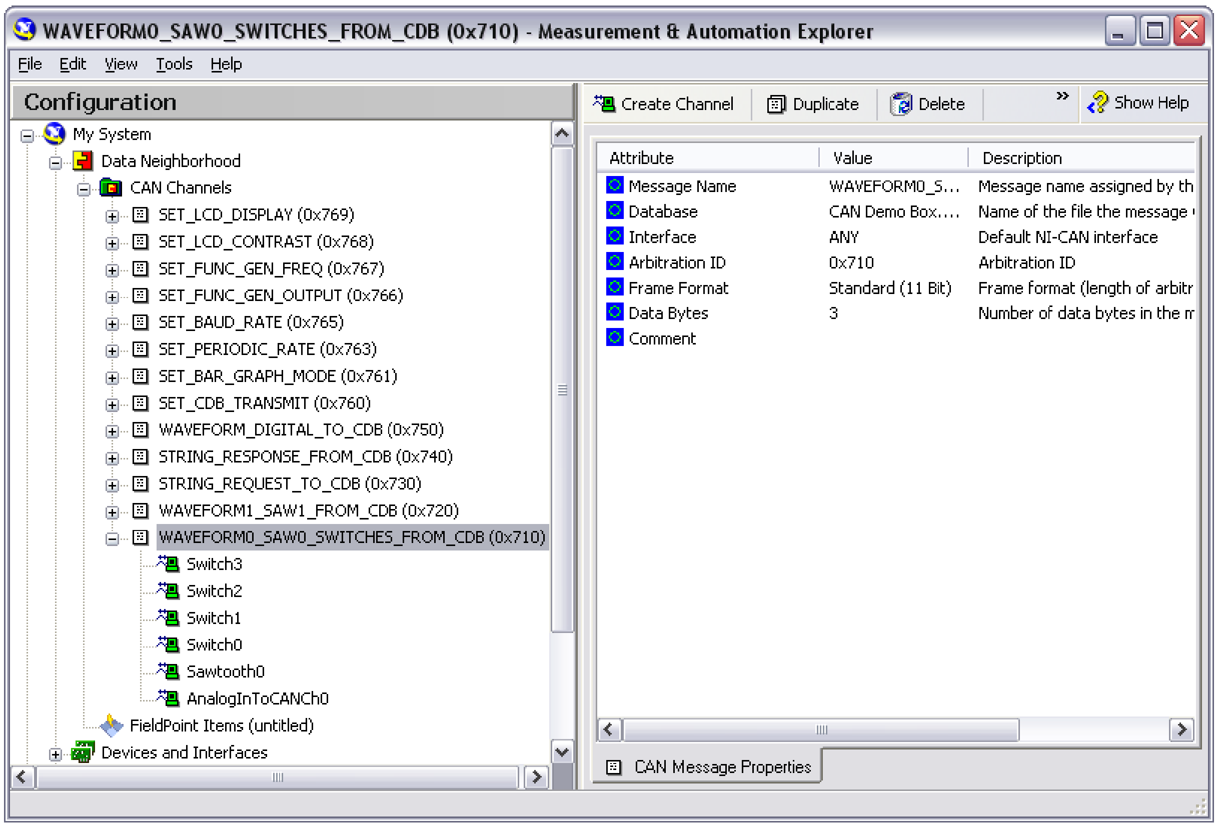
   ❑ Select **Add All Messages and Channels**.

   ❑ Click **Import**.

- ❑ Click **Done**.

3. Test a channel in MAX

- ❑ Expand **CAN Channels**. A list of messages with predefined channels appears. Use these channels to communicate with your CAN Demo Box through your CAN Interface.
- ❑ Expand the message **WAVEFORM0_SAW0_SWITCHES_FROM_CDB (0x710)** to see the channels in that message, as shown in Figure 1-15.



WAVEFORM0_SAW0_SWITCHES_FROM_CDB (0x710) - Measurement & Automation Explorer

File Edit View Tools Help

Configuration

- My System
  - Data Neighborhood
    - CAN Channels
      - SET_LCD_DISPLAY (0x769)
      - SET_LCD_CONTRAST (0x768)
      - SET_FUNC_GEN_FREQ (0x767)
      - SET_FUNC_GEN_OUTPUT (0x766)
      - SET_BAUD_RATE (0x765)
      - SET_PERIODIC_RATE (0x763)
      - SET_BAR_GRAPH_MODE (0x761)
      - SET_CDB_TRANSMIT (0x760)
      - WAVEFORM_DIGITAL_TO_CDB (0x750)
      - STRING_RESPONSE_FROM_CDB (0x740)
      - STRING_REQUEST_TO_CDB (0x730)
      - WAVEFORM1_SAW1_FROM_CDB (0x720)
      - WAVEFORM0_SAW0_SWITCHES_FROM_CDB (0x710)
        - Switch3
        - Switch2
        - Switch1
        - Switch0
        - Sawtooth0
        - AnalogInToCANCh0
    - FieldPoint Items (untitled)
  - Devices and Interfaces

Create Channel | Duplicate | Delete | Show Help

Attribute	Value	Description
Message Name	WAVEFORM0_S...	Message name assigned by th
Database	CAN Demo Box....	Name of the file the message
Interface	ANY	Default NI-CAN interface
Arbitration ID	0x710	Arbitration ID
Frame Format	Standard (11 Bit)	Frame format (length of arbitr
Data Bytes	3	Number of data bytes in the m
Comment		

CAN Message Properties

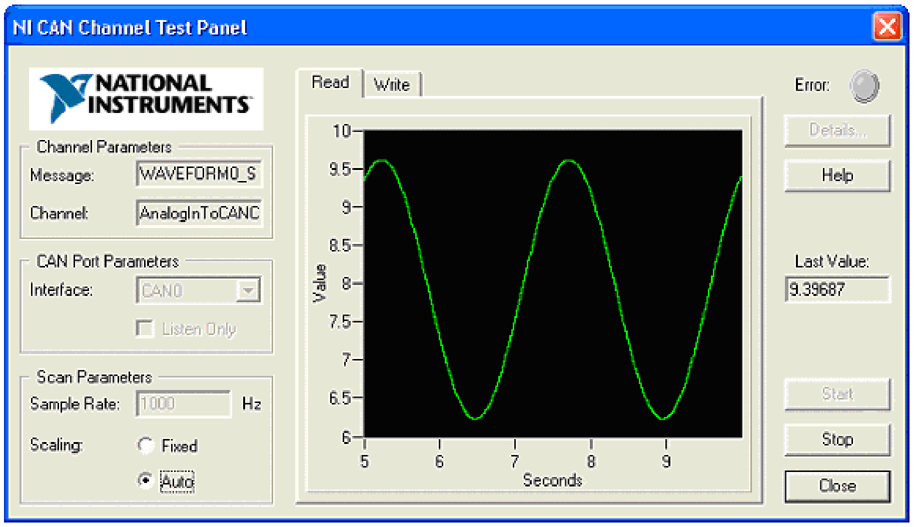
Figure 1-15. WAVEFORM0_SAW0_SWITCHES_FROM_CDB (0x710) Message

- ❑ Select the **AnalogInToCANCh0** channel.
- ❑ Click the **Test Panel** button.

Tip You also can right-click the channel name and select **Test Panel** from the shortcut menu.

- ❑ Click the **Read** tab if it is not already selected.

❑ Click the **Start** button to begin reading continuously. If the test panel is already reading when you open it, the start button is disabled and you do not need to click it.



Channel Parameters
Message: WAVEFORM0_S
Channel: AnalogInToCANC

CAN Port Parameters
Interface: CAN0
Listen Only

Scan Parameters
Sample Rate: 1000 Hz
Scaling: Fixed [✓ Auto]

Last Value: 9.39687

Tip A periodic signal should appear. You can see the signal better by changing the scaling to Auto.

❑ Experiment with changing the settings on the demo box itself. Use the **Menu Select** button to scroll through the different options.

– Click the **Menu Select** button until you see Func Gen Output.

– Click the **<+>** and **<->** buttons to change which type of signal the Function Generator outputs. Select from the following choices—Sine, Square, and Triangle. The screen displays which type is currently being output.

– Click the **Menu Select** button again until you see Func Gen Freq.

– Click the **<+>** and **<->** buttons to increase and decrease the frequency of the signal the Function Generator outputs.

❑ Click the **Stop** button in the MAX test panel to finish reading.

❑ Click the **Close** button to exit the Test Panel.

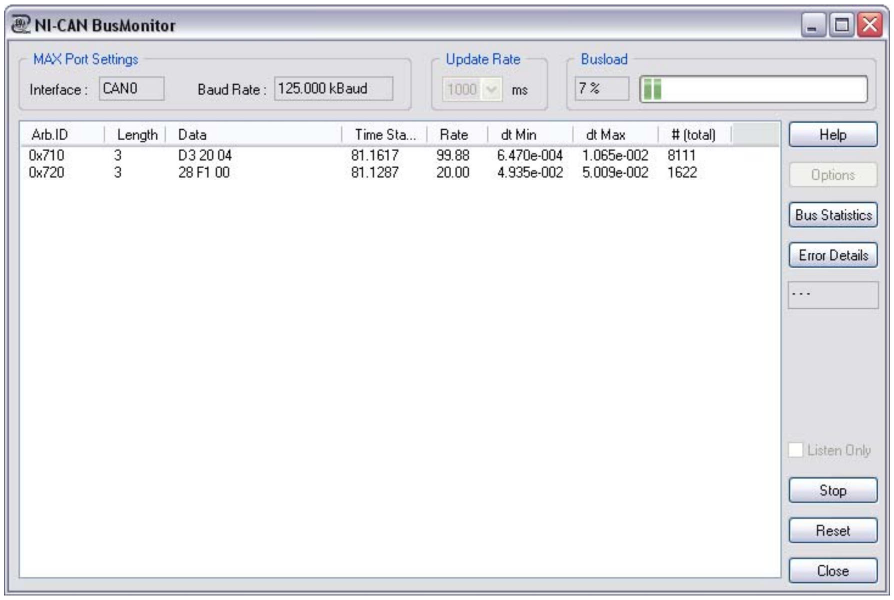
4. Monitor the CAN bus

❑ Expand **My System»Devices and Interfaces»PCI-CAN/2**. When selected, the device's properties display in the window to the right.

❑ Expand **PCI-CAN/2** and select **CAN0**. On the right, notice a list of attributes.

❑ Click the **Bus Monitor** button located above the attributes.

❑ Click the **Start** button to begin monitoring the CAN bus.
If monitoring is already active, a **Stop** button appears in place of the **Start** button and you need not click a button.


NI-CAN BusMonitor

MAX Port Settings — Interface : CAN0 — Baud Rate : 125.000 kBaud

Update Rate: 1000 ms

Busload: 7 %

Arb.ID	Length	Data	Time Sta...	Rate	dt Min	dt Max	# (total)
0x710	3	D3 20 04	81.1617	99.88	6.470e-004	1.065e-002	8111
0x720	3	28 F1 00	81.1287	20.00	4.935e-002	5.009e-002	1622

Figure 1-16. NI-CAN Bus Monitor

❑ Unplug the connector between the CAN Interface and your demo box. The monitor ceases to show any transmissions because there is nothing being sent to or from the bus.

❑ Plug in the connector between the CAN Interface and your demo box. Notice periodic transmissions because the demo box is constantly transmitting to and receiving from the CAN Interface.

❑ Click the **Stop** button.

❑ Click the **Reset** button.

5. Save the bus monitor activity to file.

❑ Click the **Options** button.

❑ Enable the **Stream To Disk** checkbox.

❑ Leave the remaining options at their default values.

❑ To specify where to stream the data, click the **File Name** button in the dialog box and navigate to a file path.

- ❑ Click the **OK** button.
- ❑ Click the **Start** button to begin monitoring and logging.
- ❑ After a few moments, click the **Stop** button.
- ❑ Click the **Close** button to close the bus monitor.

6. Open the log file that you specified in the previous steps and view its contents.

End of Exercise 1-2

Exercise 1-3 Read and Write CAN Channels

Goal

Read and write channels from the CAN device using LabVIEW.

Description

Complete a VI that reads a single value at a time from the CAN Interface and graphs it on a chart. The channel being read is Analog Input on Channel 0 of the CAN Demo box.

Note Before beginning this exercise, load the `CAN Demo Box.ncd` or `CAN Demo Box.dbc` database into MAX. Also, wire the **Function Generator Gen** output to the **Analog In To CAN Ch0** input on the CAN demo box. You completed these steps in a previous exercise.

Implementation

Read a Single Channel

1. Open **Read CAN Channels VI**, located in the `<Exercises>\CAN` directory.

In the following steps, you complete the block diagram shown in Figure 1-17.

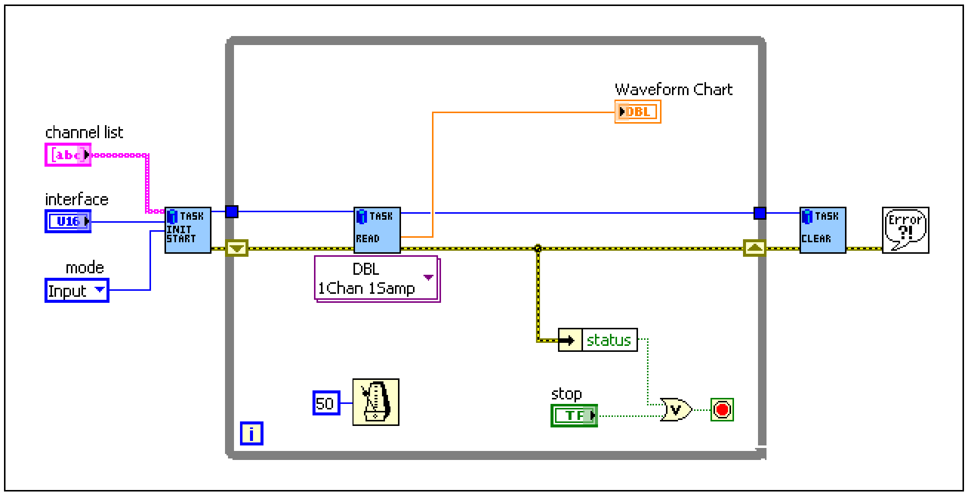


Figure 1-17. Read CAN Channels VI Block Diagram

2. Add the CAN Channel VIs to the block diagram.

   ❑ Add the CAN Init Start VI to the block diagram to the left of the While Loop.

❑ Right-click the **channel list** input of the CAN Init Start VI and select **Create»Control**.

❑ Right-click the **interface** input of the CAN Init Start VI and select **Create»Control**.

❑ Right-click the **mode** input of the CAN Init Start VI, select **Create» Constant**.

❑ Select **Input** from the mode enum constant.

❑ Add the CAN Read VI to the block diagram inside the While Loop.

❑ Select **Single Channel»Single Sample»DBL** from the pull-down menu on the CAN Read VI.

Tip You also can accomplish this by right-clicking the VI and selecting **Select Type» Single Channel»Single Sample»DBL** from the shortcut menu.

❑ Add the CAN Clear VI to the block diagram to the right of the While Loop.

3. Wire the block diagram as shown in Figure 1-17.
4. Switch to the front panel.
5. Test the VI.

❑ Set the interface control to **CAN0**.

❑ Enter `AnalogInToCANCh0` into the first array element of the channel list control.

Tip You also can drag the name of the channel into the array element from MAX. The channel is found in the message WAVEFORM0_SAW0_FROM_CDB (0x710).

❑ Run the VI.

❑ Experiment with CAN Demo Box. Change the signal time and frequency. Notice the changes taking effect on your chart.

Your front panel should resemble Figure 1-18.

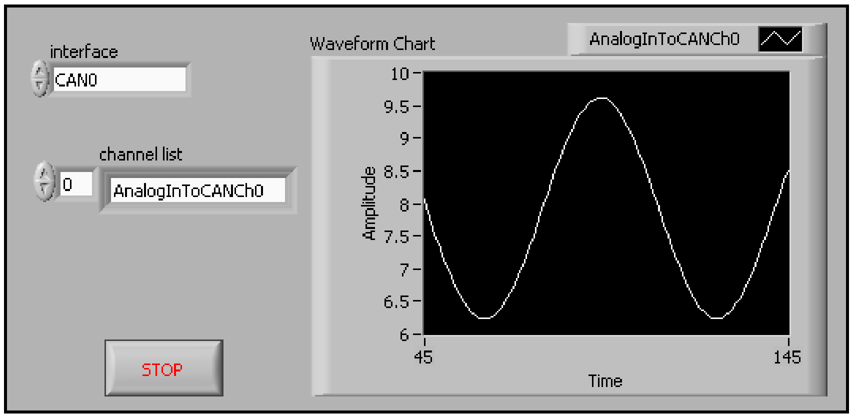


Figure 1-18. Read CAN Channels VI Front Panel

- ❑ Click the **Stop** button to terminate the program.

6. Save the VI as `Read CAN Channels.vi` in the `<Exercises>\CAN` directory.

Read Two Channels

Modify the VI to read a second channel, `Switch0`.

1. Add a second channel to the front panel of the Read CAN Channels VI.

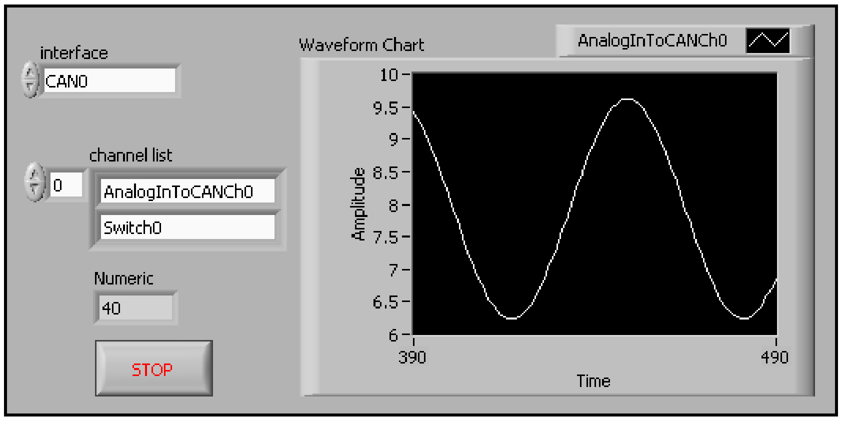


Figure 1-19. Read CAN Channels (Multiple) VI Front Panel

- ❑ Expand the number of array elements visible on the channel list control on the front panel of the Read CAN Channels VI.

- ❑ Enter `Switch0` into the second array element of the channel list control.

- ❑ Add a numeric indicator to the front panel and rename it as `Switch0`.

2. Modify the block diagram as shown in Figure 1-20 to display the second channel.

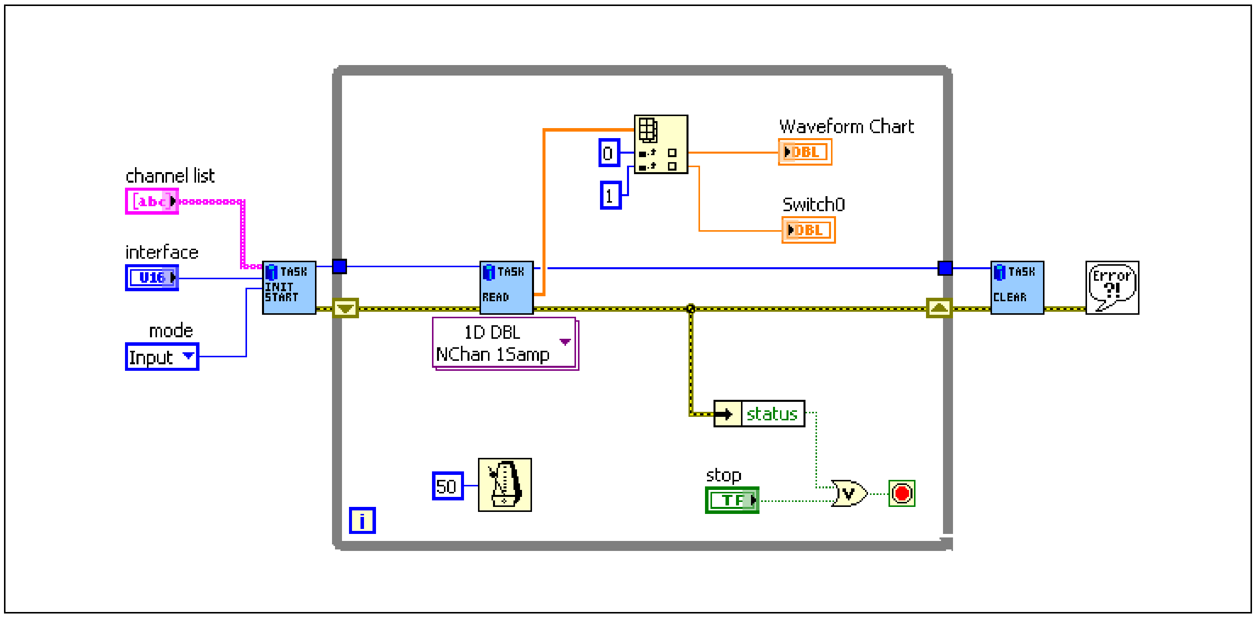


Figure 1-20. Read CAN Channels (Multiple) VI Block Diagram

- ❑ Switch to the block diagram.
- ❑ Delete the wire between the output of the CAN Read VI and the waveform chart.
- ❑ Select **Multiple Channels»Single Sample»1D DBL** from the pull-down menu that appears below the CAN Read VI (the polymorphic VI selector).

  This specifies that the VI should read an array of doubles, one from each channel, every time that it executes.
- ❑ Add an Index Array function to the block diagram.
- ❑ Using the cursor, drag the border of the Index Array function to expand its size so that it returns two array elements instead of one.
- ❑ Wire the **multi-chan single-samp 1D dbl** output array from the CAN Read VI to the **array** input of the Index Array function.
- ❑ Right-click the **index0** input of the Index Array function and select **Create»Constant**.
- ❑ Right-click the **index1** input of the Index Array function and select **Create»Constant**.

❑ Enter `1` in the second constant.

❑ Connect the first output element of the Index Array function to the **waveform chart indicator** terminal and connect the second output element to the **Switch0** indicator terminal.

3. Save the VI as `Read CAN Channels (Multiple).vi` in the `<Exercises>\CAN` directory.

4. Switch to the front panel.

Test

1. Test the VI.

   ❑ Run the VI. While it is running, experiment with changing the properties of the function generated by the CAN Demo Box.

   ❑ Experiment with changing the state of Switch0 as well. As you toggle the switch (Digital Input 0) on the CAN Demo Box, notice that the value of the numeric indicator changes.

2. Use another VI to modify the frequency of the waveform generated by the demo box.

   ❑ Open the Write CAN Channels VI, located in the `<Exercises>/CAN` directory.

   ❑ Ensure that the channel list has one element with the channel name `FunctionGeneratorFrequency`.

   ❑ Set the interface control to `CAN0`.

   ❑ Run the CAN Read Channels VI and the CAN Write Channels VI.

   ❑ Experiment with changing the Frequency Value control on the CAN Write Channels VI and see how it affects the chart for the CAN Read Channels VI.

3. Terminate the execution of both VIs by clicking their **Stop** buttons.

End of Exercise 1-3

Exercise 1-4 Synchronize CAN & DAQ

Goal

Synchronize CAN and DAQ inputs in LabVIEW.

Description

There are two CAN and DAQ acquisitions inputting data simultaneously. Both the acquisitions are buffered, meaning that the timing is handled on the interface devices. Explore a VI that synchronizes the inputs by routing the clock signal from one of the devices to the other using a Real-Time Serial Interface (RTSI) cable to connect the two.

1. Open the Sync CAN & DAQ VI located in the `<Exercises>\CAN` directory.
2. Switch to the block diagram.
3. Explore the block diagram shown in Figure 1-21. Notice that this VI reads from both the CAN and DAQ interfaces and uses RTSI to synchronize the timing.

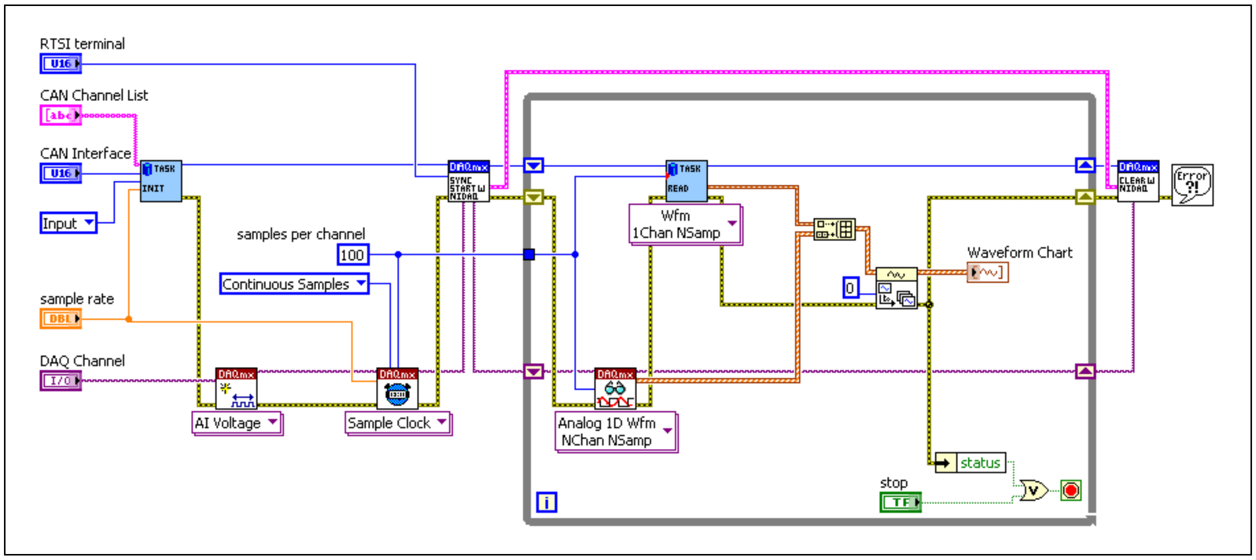


Figure 1-21. Sync CAN & DAQ VI Block Diagram

4. Switch to the front panel.
5. Set the control values:
   - ❑ Interface: **CAN0**
   - ❑ Channel List: **AnalogInToCANCh0**

- ❑ Physical channels: `Dev1/ai0`
- ❑ RTSI Terminal: **RTSI0**
- ❑ Sample Rate: `1000.00`

6. On the CAN Demo Box, attach two wires to jumper between the output of the Functions Generator, the input Analog In to CAN Ch0, and the input Analog In to DAQ Ch0.
7. Run the VI. Notice that the signals read from the CAN interface and the DAQ device are perfectly synchronized. To terminate the VI, click the **Stop** button.
8. Close the VI when you are finished.

End of Exercise 1-4

Notes

Notes

Notes