Section 6

Scanning low-level DCV

In this section:

Introduction

This application example demonstrates how to use the DAQ6510 to accurately measure DC voltage in a variety of ranges. To ensure accurate data, the NPLC (Number of Power Line Cycles) and autozero options are used for this test.

The NPLC setting can be used to help reduce the induced noise originating from nearby AC power-conditioning circuits. A desktop power supply or power-transmission lines would generate this type of noise. Increasing NPLC cancels out this noise by integrating all sampled data collected in multiples of AC signal periods (n * 1/(transmission line frequency) seconds). The more AC line cycles used in the measurement, the more accurate the reading. The time required to conduct the scan also increases.

The autozero function removes offset voltages that result from thermal EMFs. Thermal EMFs occur when there is a temperature difference at junctions consisting of different materials. For example, leads, instrument inputs, or card terminals. These EMFs adversely affect DCV measurement accuracy by offsetting the measured voltage.

This example shows how to measure voltage in different ranges. To optimize scanning speed, you should set a fixed range. If speed is not an issue, the measurement range can be set to auto.

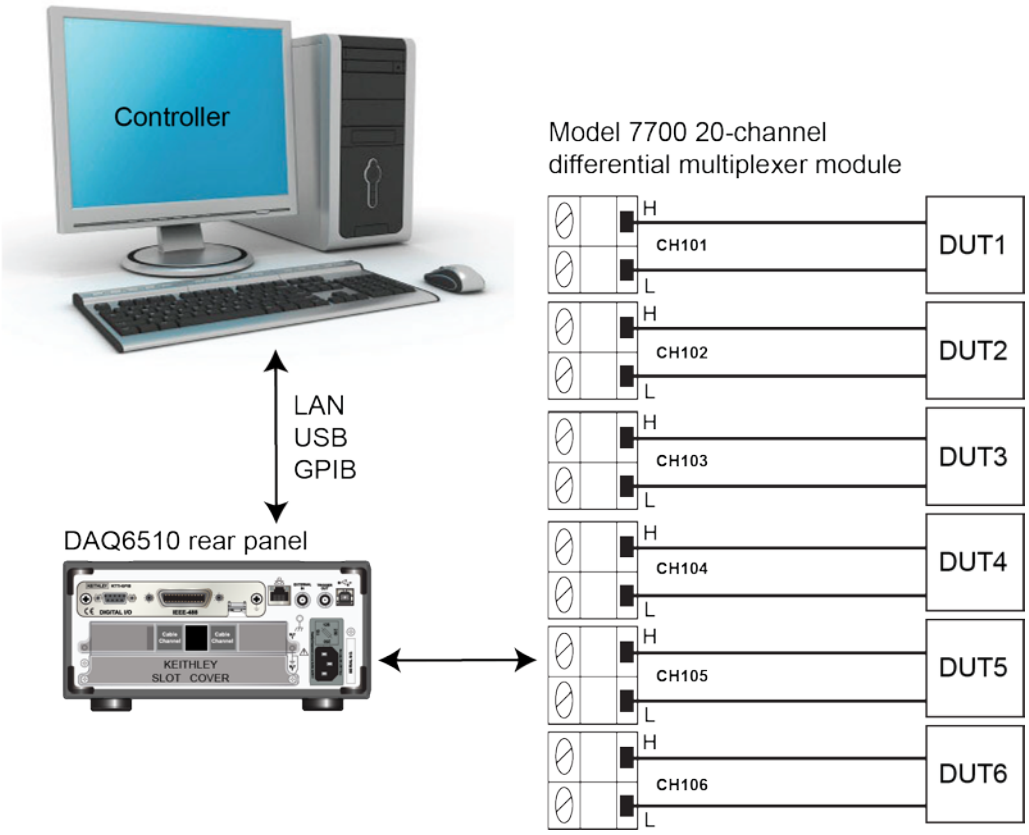
Equipment required

- One DAQ6510
- One Model 7700 20-channel differential multiplexer module
- One computer setup for communication with the instrument
- One device or component to be tested

Device connections

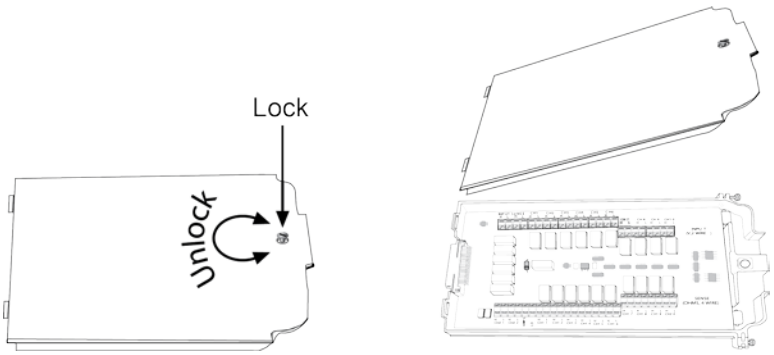
This example uses a DAQ6510 with a 7700 multiplexer module. In this example, channels 101 to 106 are connected to six DUTs (device under test) where DC voltage is measured.

Figure 36: DAQ6510 instrument and devices connections for low-level DCV



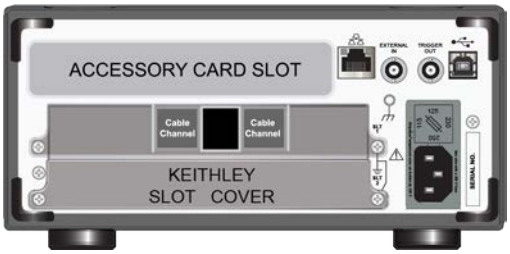
Connecting devices to the instrument

1. Power off the DAQ6510.
2. Remove the 7700 from the DAQ6510.
3. Remove the top cover from the 7700.



4. Connect six devices.
5. Route the cables out through the cable channels and secure the top cover.
6. Ensure that the DAQ6510 power is turned off.
7. Insert the 7700 into a slot on the rear of the DAQ6510.

Figure 37: DAQ6510 with 7700 multiplexer module



8. Press the **POWER** switch on the front panel to turn on the instrument.
9. Set the **TERMINALS** switch to rear.

The remaining connections are to the DUT and are at your discretion.

WARNING

To prevent electric shock, test connections must be configured such that the user cannot come in contact with test leads or any device under test (DUT) that is in contact with the conductors. It is good practice to disconnect DUTs from the instrument before powering the instrument. Safe installation requires proper shields, barriers, and grounding to prevent contact with test leads.

There is no internal connection between protective earth (safety ground) and the LO terminals of the DAQ6510. Therefore, hazardous voltages (more than 30 V_{rms}) can appear on LO terminals. This can occur when the instrument is operating in any mode. To prevent hazardous voltage from appearing on the LO terminals, connect the LO terminal to protective earth (safety ground) if your application allows it. You can connect the LO terminal to the chassis ground terminal on the front panel or the chassis ground screw terminal on the rear panel. Note that the front-panel terminals are isolated from the rear-panel terminals. Therefore, if you are using the front-panel terminals, ground to the front-panel LO terminal. If using the rear-panel terminals, ground to the rear panel LO terminal. Failure to follow these guidelines can result in injury, death, or instrument damage.

Scanning low-level DCV

This application demonstrates how to use the DAQ6510 to perform DC voltage measurement using multiple channels on the 7700 20-channel differential multiplexer module.

For this application, you will:

- Configure channels 101 to 106 for DC voltage measurement.
- Enable autoranging and autozero on each channel.
- Set NPLC to five on each channel.
- Execute 10 scans on all selected channels.

You can program the unit through the front panel or use the remote communications interface of your choosing (LAN, USB, GPIB, RS-232, or TSP-Link) via the SCPI and TSP code commands.

Using the front panel

To set up the application from the front panel:

1. Press the **POWER** switch on the front panel to turn on the instrument.
2. Select the **REAR** terminals**.**
3. Press the **MENU** key.
4. Under Channel, select **Scan.**
5. Select the **+** button to add a group of channels, select channels 101 to 106, and select **OK**.
6. Select **DC Voltage**.
7. On the Settings tab, set NPLC to **5**.
8. Set Auto Zero to **On**.
9. On the Scan tab, set Scan Count to **10**.
10. Select **Start** at the bottom of the left pane.

To watch active readings during scan:

1. Press the **HOME** key.
2. Select the arrow to the right of **Watch Channel**.
3. Select the channels of interest and select **OK** to accept.

To save the scanned measurements to a USB flash drive:

1. Select the **MENU** key.
2. Under the **Measure** column, choose **Reading Buffers**.
3. Insert a USB flash drive into the DAQ6510. Select **Save to USB** at the bottom of the pane.
4. Select **OK**.

Using SCPI commands

This sequence of SCPI commands executes a DCV scan on channels 101 to 106.

You may need to make changes so that this code will run in your programming environment. In the table, the SCPI commands have a light gray background. The light green shaded code represents pseudocode that will vary depending on the programming environments you use.

Send the following commands for this example application:

	Commands	Descriptions
Pseudocode	`int scanCount = 10`	• Create a variable to hold the scan count
	`int channelCount = 6`	• Create a variable to hold channel count
	`int bufferSize = scanCount * channelCount`	• The number of readings
	`int lastIndex`	• The last scan index in the buffer after every two seconds
	`string tmpBuff`	• A temporary buffer that store scanned data every two seconds
DAQ6510	`*RST`	• Put the instrument in a known state
	`:TRAC:POIN bufferSize, "defbuffer1"`	• Set the buffer size (not necessary when using the default buffer, but added to show command use case)
	`:ROUT:SCAN:BUFF "defbuffer1"`	• Assign all scanned data to "defbuffer1"
	`FUNC 'VOLT:DC', (@101:106)`	• Set channels functions to DCV
	`VOLT:DC:RANG:AUTO ON, (@101:106)`	• Set the channels ranges to auto
	`VOLT:DC:NPLC 5, (@101:106)`	• Set NPLC to 5
	`VOLT:DC:AZER ON, (@101:106)`	• Set autozero function to ON
	`ROUT:SCAN (@101:106)`	• Setup the scan list
	`ROUT:SCAN:COUN:SCAN 10`	• Setup the number of repeated scans
	`INIT`	• Initiate the scan
Pseudocode	`for i = 1; i <= bufferSize;`	
	`  delay 2000`	• Delay two seconds to allow readings to accumulate
DAQ6510	`  lastIndex = TRACe:ACTual?`	• Query the current buffer index
	`  tmpBuff = "TRACe:DATA? i, lastIndex, "defbuffer1", READ`	• Query the readings available from index i to index lastIndex
Pseudocode	`  printBuffer (tmpBuffer)`	• Print readings in temporary buffer to console
	`  i = lastIndex + 1`	
	`end for`	
DAQ6510	`:TRAC:SAVE "/usb1/MyData.csv", "defbuffer1"`	• Save buffer data from defbuffer1 to USB

Using TSP commands

NOTE

The following TSP code is designed to be run from Keithley Instruments Test Script Builder (TSB). TSB is a software tool that is available from the Keithley webpage on the Tektronix website (tek.com/keithley). You can install and use TSB to write code and develop scripts for TSP-enabled instruments. Information about how to use TSB is in the online help for TSB and in the "Introduction to TSP operation" section of the *DAQ6510 Reference Manual*.

To use other programming environments, you may need to make changes to the example TSP code.

By default, the DAQ6510 uses the SCPI command set. You must select the TSP command set before sending TSP commands to the instrument.

To enable TSP commands:

1. Press the **MENU** key.
2. Under System, select **Settings**.
3. For Command Set, select **TSP**.
4. At the prompt to reboot, select **Yes**.

This sequence of TSP commands initiates a series of temperature measurements. After the code executes, the data is displayed in the Instrument Console of Test Script Builder.

Send the following commands for this example application:

```
-- Reset the instrument to the default settings
reset()
channelCount = 6
scanCount = 10
bufferSize = channelCount * scanCount
defbuffer1.capacity = bufferSize
scan.buffer = defbuffer1 -- not necessary when using the default buffer, but added
   to show command use case.
-- set up channels
channel.setdmm("101:106", dmm.ATTR_MEAS_FUNCTION, dmm.FUNC_DC_VOLTAGE,
   dmm.ATTR_MEAS_RANGE_AUTO, dmm.ON, dmm.ATTR_MEAS_AUTO_ZERO, dmm.ON,
   dmm.ATTR_MEAS_NPLC, 5)
-- set up Scan
scan.add("101:106")
scan.scancount = scanCount
-- set up trigger model
trigger.model.initiate()
-- print measurement data to console
i = 1
while i <= bufferSize do
   delay(2)
   lastIndex = defbuffer1.n
   printbuffer(i, lastIndex, defbuffer1.readings)
   i = lastIndex + 1
end
-- save buffer data from defbuffer1 to USB
buffer.save(defbuffer1, "/usb1/MyData.csv")
```