EXPERIMENT 2

COMPUTER INTERFACING

1 Equipment list

- A computer with PCI-6025E data acquisition card and LabVIEW software installed.
- A 100 pin Input/Output connector for the 6025E device. This connector is divided into two 50 pin connector blocks. We will only use one of these blocks.
- The LM35DZ temperature sensor and red, green and yellow light emitting diodes.

References: DAQ 6023E/6024E/6025E User Manual, National Instruments.
DAQ NI-DAQ User Manual for PC Compatibles, National Instruments.
National Instruments LabVIEW User Manual, National Instruments.

2 Aim

This practical introduces you to a method that enables you to interface computers to experiments. In particular, you will be able to make your computer behave as a digital voltmeter and a thermometer. This is part of today's technical skills required by the experimental physicist in order to carry out his/her experiments.

3 Introduction

Almost all laboratory experiments use computers to record their results. In the past, the recording could have been done by many methods such as chart records, xy-plotters, oscilloscopes, or simply by eye and using pen and paper. But with technological sophistication, such as medical imaging machines (MRI, X-ray CT scans, etc), or monitoring the readings from instruments in airplanes or spacecraft, there is a need to transfer readings into a computer for ease of data manipulation or communication. The method of transferring data into a computer in real-time is known as **data acquisition (DAQ)**. This experiment is an introduction to this field.

Data acquisition is carried out by installing an electronic interface card in a computer. There are many card designs but the most basic function of any interface card is the conversion of an external voltage into a digital number, known as **analog to digital conversion (ADC)**. This digital output is collected by software that communicates with the card. The card must also be able to output an analog voltage when the software applies a number to the card, which is known as **digital to analog conversion (DAC)**.

This is basically how a computer talks to the outside world. That is, an input voltage applied to the interface card is converted to a number inside the computer, or a number form the computer is converted to an output voltage from the interface card. It is up to the experimental setup to deliver a voltage that is proportional to the parameter under investigation. For example, say you were dealing with an experiment that was measuring light intensity (as in an optical spectrometer). In this case, the optical detector, which can be a photodiode, phototransistor, photomultiplier, or CCD camera must output a voltage proportional to the light intensity. Similarly, if we needed to control an experiment via a computer, then our experiment must be able to accept voltages from an interface card. For example, the computers that control the robots in a car factory will receive their instructions as voltages from an interface card.

To program such interface cards, there is no need to know exactly what the electronics on the card is doing. The manufacturers generally make it easy for you to talk to the card using software. However, it is important to know some basic concepts such as analog to digital and digital to analog conversion, resolution, and digital input/output (IO), which we shall now discuss.

3.1 Analog to Digital and Digital to Analog Conversion

This is one of the most important functions of an interface card. Assume a voltage is applied to the card, the circuitry on the card must be able to convert it to a number that is read by the software. This number is generally an integer. Similarly, a number (also an integer) is output by the software to the card then to the outside world as an analog voltage. The integer that you use is determined by the **resolution** of the card. We will explain resolution with the following example.

Usually interface cards can only accept a limited range of input voltages, and similarly, can output a limited range of voltages. Say that a card can accept a maximum of ± 10 Volts, or output a maximum of ± 10 Volts. Say our card had **8 bit resolution**. The 8-bit means that our voltage range can be divided into $2^8 = 256$ equal parts. In our case, $-10V \rightarrow +10V$ is a range of $20V$. The smallest voltage we can obtain or output is given by $20/256 \simeq 0.078V$ or 78mV. This also means that a continuously varying signal (such as a sine wave) will only be acquired or sent out in steps of 78 mV. This might look light it's pretty good, but an 8 bit card is generally regarded as cheap and not of sufficient resolution for most experiments. Usually a 12 bit card is regarded to be of sufficient quality. This means that the maximum voltage range can only be resolved in steps of $range/2^{12} = range/4096$. The card used in this experiment has 12 bit resolution.

3.2 Digital Input and Output (DIO)

Sometimes we simply need to receive or send a digital signal without the conversion to analog. It may be that the device that we are communicating with needs the digital number directly. For example, say the number 9 had to be sent. So we send this number in binary format. That is, we send 1001. Note that in the binary system there are only two numbers 0 and 1. To know what a binary number is in decimal, you must carry out the following procedure. So for the example given here we write

$$1001 = 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 8 + 0 + 0 + 1 = 9 \tag{1}$$

From this you can see that the power of 2 increases as you move left along the binary number. The 0's and 1's are known as **bits**. The right most bit in the binary number is know as the least significant bit and is given by 2^0. The 0's or 1's in the binary number simply act as the multipliers.

Another example is the number 250 which is given by

$$11111010 = 1 \times 2^7 + 1 \times 2^6 + 1 \times 2^5 + 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 250 \tag{2}$$

Quite often in experimental work we simply want to send a signal to an apparatus telling it either to start or stop. By using the digital lines from our data acquisition card we send a 1 or a 0. In practice the 1 is some fixed voltage, say $5V$, and 0 is simply $0V$. Digital lines are unable to apply any intermediate voltage. Using this we can synchronize all of our experimental apparatus from one central computer. Data acquisition cards usually have more digital lines than analog. The card that we will be using has 32 digital lines.

3.3 Hardware

The data acquisition card has already been installed in the computer. This card has a connector that pokes out of the back of the computer. A ribbon cable is connected to this connector, which terminates in a green connector block. This block has a connector for each pin on the card. Each pin has a specific function. For example, 32 pins are devoted to the 32 digital input/output lines. Another 16 pins are devoted to the analogue input/output. For example, pin number 3 (labelled on the green connector, shown in Fig. 1) is the analog input channel number 0, usually labelled as ACH0. To input a voltage we connect a wire into the number 3 pin and tighten its screw to hold the wire securely in the block. The wire then goes to our experiment. The bench notes for the card indicate the function of each pin. The card that we will be using is the National Instruments card PCI-6025E. The PCI-6025E features 16 channels (12 bits resolution) of analog input, two channels of analog output and 32 lines of digital Input/Output (I/O). More information about the PCI-6025E can be found in 6023E/6024E/6025E User Manual.

3.4 Software

Throughout this experiment you will be acquiring and outputting data using the LabVIEW$^{\text{TM}}$ software, which is a product of the National Instruments company. It is one of the more popular programming environments for computer interfacing because it is an icon based programming language. That is, you program by dragging the appropriate icons onto the screen and connecting lines between them. This experiment is about learning to use LabVIEW$^{\text{TM}}$ and interface cards. We will be using version 6 of LabVIEW$^{\text{TM}}$ to carry out computer interfacing to experiment.

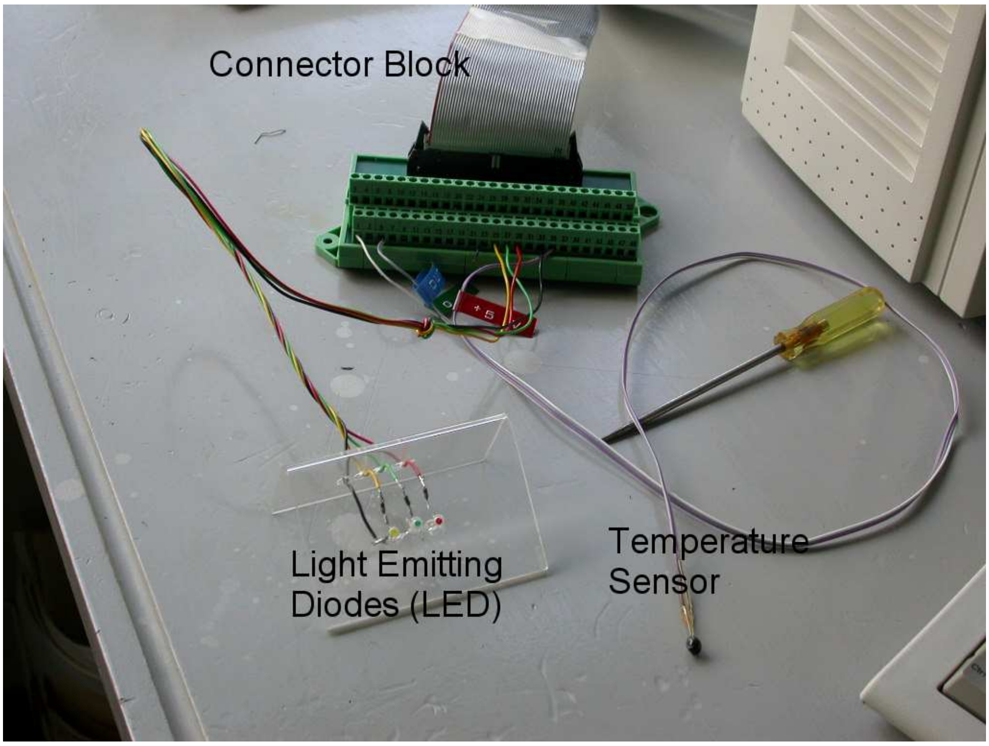


Figure 1: The ribbon cable that connects to the green connector block. Connected to this block are the temperature sensor and light emitting diodes (LED)

3.5 Boolean algebra

In the last part of this experiment you will need to know about Boolean algebra. If you are familiar with such concepts such as AND, OR, NOT, XOR, etc gates, then you do not necessarily have to read this subsection on Boolean algebra.

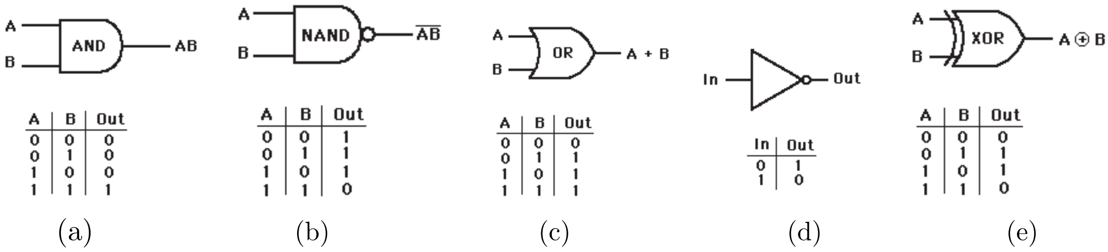
The most basic units for digital logic are the logic gates. These can be both actual electronic circuits or software symbols. The latter is what matters for this experiment. The most common of these are shown in Fig. 2. Their explicit function is explained in the caption to this figure. Generally, they consist of an input on which the gates perform a specific function with a resulting output that is either a 0 or a 1. In real electronic circuits a 1 can be a real voltage such as 5 volts. Using gates in software, also deals with **True**(which means 1) and **False** (which means 0).

4 Procedure

Our goal is to build a simple voltmeter and a simple digital thermometer using the temperature sensor LM35DZ.

Question: The range on our card for an analog input is -5.00V to +5.00V. Remembering that we have a card with 12 bit resolution, what is the smallest voltage step that the card will measure?

C1 ▷ **Tutor checkpoint. Obtain a tutor's signature before proceeding.**



A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

A	B	Out
0	0	1
0	1	1
1	0	1
1	1	0

A	B	Out
0	0	0
0	1	1
1	0	1
1	1	1

In	Out
0	1
1	0

A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0

Figure 2: Some logic gates with their truth tables that show the output states depending on the input (a) A AND B must both be 1 for the output to be 1 (b) NAND is the exact opposite of AND (c) OR means A OR B have to be 1 for the output to be 1 (d) NOT means that the output is the opposite to the input (e) Exclusive OR (XOR) is the same as the OR gate except that if both A and B are 1, then the output is 0 .

At all stages of this practical you should print out key windows from LabVIEW and stick them in your logbook. In addition, you should comment about them.

4.1 A simple voltmeter

We want to be able to apply a voltage to the connector block and have the computer display its value on the screen. The following procedure will show you how to do this.

- The first step in building a digital voltmeter is to configure an analog input channel that will be used for measurement. Let it be analog channel 0, ACH0, connected to pin 3 of the 1-50 pin connector.
- Open the "Measurement & Automation Explorer (MAX)" programme on the desktop and follow instructions to insert a new DAQ virtual channel. The wizard will help you create the Analog Input named ACH0 that will sense voltage in the range -5.00V to +5.00V without scaling.
- Choose PCI-6025E as the DAQ hardware and channel 0 in Referenced Single Ended mode and click the Finish button at the end of the wizard. The new channel will appear in the "Data Neighborhood".
- To test or edit the properties of this channel click the right mouse button over a channel's icon. Now we are ready to build a virtual instrument that makes use of this channel.
- Close the MAX window and open the National Instruments LabVIEW programme.
- Choose the "New VI" button. Two windows named "Untitled 1" and "Untitled 1 Diagram" are created.
- Activate the diagram window and from the "Window" menu choose "Show Functions Palette" if it is not visible yet.
- Choose *Functions > Data Acquisition > Analog Input > AI Sample Channel.vi* (these names become visible when you run the mouse pointer over the different icons) and place the icon in the diagram window.

- Click the right mouse button over this icon and choose "Help" to learn more about this virtual instrument.

This virtual instrument requires the "device number" and the "channel name" to be defined. The following procedure will enable you to do this.

- Activate the panel window (Untitled 1) and from the "Window" menu choose "Show Control Palette" if it's not already visible.

- Choose *Controls > Numeric > Digital Control* and place it on the panel. It appears also in the diagram window.

- Click the right mouse button over this control and change its representation to "Word (I16)" as required by "AI Sample Channel.vi".

- If the "Tools" palette is not visible, open it from the "Window" menu and choose the "Edit Text" icon (symbolized by the "**A**" icon).

- With this tool selected, edit the label of the "Numeric" control to become the "Device Number". In addition, change the value from 0 to 1.

- Click the right mouse button over this control and choose *Data Operations > Make Current Value Default*.

- Change the tool to the "Connect Wire" (symbolized by the reel icon).

- Switch to the diagram panel. Connect the "Device Number" to the terminal of the "AI Sample Channel.vi" named "device" by placing the mouse (now the reel icon) over the **I16** and hold down the left mouse button and drag to the **analog input** icon.

- At this stage it is a good idea to save your virtual instrument under a meaningful name, for example "Tom's Digital Multimeter.vi". The new name should appear in the title of the panel and the diagram window.

- From the "Controls" palette select *I/O > DAQ Channel Name* and place it on the panel. Using the "Edit Text" tool write the name "ACH0" and set it as a default value of the channel name. Connect this control to the "channel" terminal of the "AI Sample Channel.vi".

- From the "Controls" palette choose *Numeric > Digital Indicator* to create a display for the measured voltage. Change its name to "Voltage" and connect it to the "sample" terminal of the "AI Sample Channel.vi".

- Your "Digital Multimeter" is ready to use. To test it connect a 1.5V battery between pin 1 and 3. Click "Run" (right arrow) or "Run Continuously" (two arrows following each other) icon from the panel or diagram window to read voltage of the battery.

- Compare the result with a voltage obtained by using hand-held commercial multimeter.

C2 ▷ **Tutor checkpoint. Obtain a tutor's signature before proceeding.**

4.2 A simple digital thermometer

This project is very similar to the previous one. To measure temperature we will use the temperature sensor LM35DZ. This sensor has 3 terminals: 0V, +5V and output. The output is a voltage that is proportional to the temperature of the sensor. A 0V output corresponds to a temperature of 0°C. A temperature change of 1°C produces a change in output of 10mV. We will use the analog channel "ACH1" (pin5) to read the temperature.

- To create this channel follow the same procedure as you did for channel "ACH0".
- When you are asked about the type of the sensor, during the wizard, choose "Voltage" and check "This will be a temperature measurement" check box.
- Define "Units" to be "DegC" and the "Range" 0-100DegC.
- Next choose "New custom scale", and define the name of the scale for example: "LM35DZ" and the "Scale Type" as "Linear Scale".
- Set $m = 100$ and $b = 0$.
- Choose channel 1 as an analog input and finish the wizard.
- Now we can try this new channel using your previous digital multimeter vi. From the "DAQ Channel Name" list choose "ACH1" and run digital multimeter. In the "Voltage" monitor you should see the value corresponding to the room temperature in degrees Celsius.
- We will now modify your digital multimeter, so it is a good idea to save it as a different name such as "Tom's Digital Thermometer" or similar.
- Make "ACH1" default DAQ channel for this virtual instrument. Change the label of the voltage monitor from "Voltage" to "Temperature degC" and save changes.

Now we will add to our thermometer three Light Emitting Diode (LED) indicators: yellow that will light if temperature is lower than minimum, red if temperature is higher than maximum and green if temperature is between minimum and maximum. We will create three digital channels to supply the LEDs.

- The procedure is almost identical to that used to create an analog channel. This time channel type should be "Digital I/O", name "DIO0" (pin25), digital type "Write To Line", hardware "Dev1: PCI-6025E", port "DIO", line 0 and "No" invert line.
- Repeat this procedure to create channels "DIO1" and "DIO2" just changing the name of the channel and the lines to be 1 (pin27) and 2 (pin29) accordingly.

The next step is to test the newly created channels. Check if the LEDs are correctly connected to the connector block: the Black wire should be connected to digital ground (DGND pin33) and yellow, green, red to DIO0 (pin25), DIO1 (pin27), DIO2 (pin29), respectively. Lets add the digital channels to "Tom's Digital Thermometer.vi" diagram.

- Choose *Functions > Data Acquisition > Digital I/O > Write to Digital Line.vi* and place three channels on the diagram.

Get help on the "Write to Digital Line.vi" to read more about it. In this section we want to create 3 inputs for "digital channel" terminal and 3 inputs to be connected to the "line" terminals of the "Write to Digital Line.vi". The procedure is identical to that used for the analog channel. Here is how this can be carried out.

- Connect the "device" terminal to the existing control "Device Number".
- Go to *Controls > I/O > DAQ Channel Name.* Place three of these on the Front Panel Window.
- Switch to the Diagram Panel. Connect the "I/O" icon to the "digital channel" input on the "Write to Digital Line" icon.
- Switch to the front panel and select the **hand** icon from the tools menu.
- By default the digital channel name class is "Analog". Click the right mouse button over "DAQ Channel Name" control and select its "DAQ Class" as "Digital Output". Set the channel names to "DIO0", "DIO1", "DIO2" as defaults.
- Click on the down arrow of the DIO0, DIO1, and DIO2 boxes and select the DIO0, DIO1, and DIO2 options.
- At this stage save your current virtual instrument. The best thing to do is use your own name as the filename e.g. "Tom's Digital Thermometer and LEDs.vi" or similar.

To supply the "line state" terminal of the "Write to Digital Line.vi" we need to compare the current temperature with the value that can be set by the user. Let's make a connection for the red LED.

- From the "Functions" tool box choose *Comparison > Greater?.* Place it on the diagram panel. On the front panel window create *Control > Numeric > Digital Control* and label it as "Maximum". Set it's default value to 30.
- Connect the "x" terminal of the "Greater?" to the "sample" terminal of the "AI Sample Channel.vi" and the "y" terminal with the "Maximum Digital Control".
- Connect the "x>y?" terminal of the "Greater?" with the "line state" terminal of the "Write to Digital Line.vi" controlling the red LED. If the connecting line now looks broken, then right click the mouse button over the " >", then left click on **"compare aggregate"**.

Run the application and check that after the temperature rises above 30°C, the red diode lights. Now try to make connection for the yellow LED and then the green LED. What "Boolean" function you should use for the green light?

C3 ▷ **Tutor checkpoint. Obtain a tutor's signature before proceeding.**

4.3 Optional

Undertake an innovative project above and beyond what you have carried out in these notes for a bonus mark. Remember it cannot simply be a cosmetic change to the display on the screen.

C4 ▷ **Checkpoint for a bonus mark**