

## Lecture 3 - Digital I/O

**Introduction.** As noted in lecture 1, the computer uses two-state (binary) logic and arithmetic. One of the possible states is known as true, 1 or high, while the other is known as false, 0 or low. These signals are often identified with specific voltages. In the most common scheme, known as TTL positive logic, true is represented by a voltage between 2.2 and 5.5, while false is represented by a voltage between 0 and 0.8. A digital line is a path to the outside world where a single digital signal can be sent or received. Digital lines are usually either input or output lines, but sometimes can be bi-directional. On some boards, digital lines must be configured as input or output: they can't act as both at the same time.

A port is a collection of digital lines that are configured in the same direction and can be used at the same time. The PCI-MIO-16E-4 board has eight digital lines configurable as one eight-line port, two four-line ports, or even eight one-line ports. Port width is the number of lines in a port.

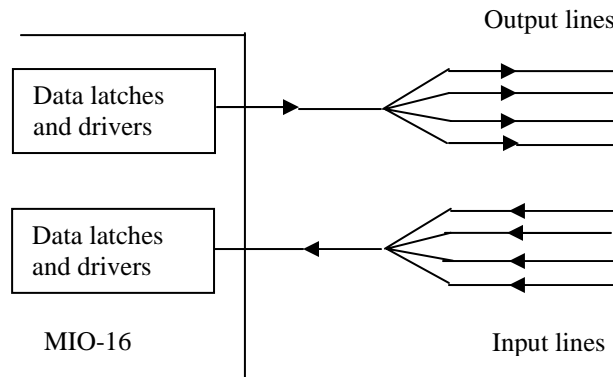


Figure 1. Digital Port

A pattern is a sequence of binary states, often expressed as a binary number, which describes the states of each of the lines on a port. For example, a four-line port might be set with the pattern 1101, meaning the first, third and fourth lines are true and the second line is false. The first bit, or least significant bit (LSB), is the rightmost bit on the pattern. The last (fourth in this example) or most significant bit (MSB) is the leftmost bit in the pattern. This pattern can also be converted from its binary equivalent to the decimal number 13.

There are two types of digital data acquisition: immediate and handshaking. In the former, the data is made available to or from the computer as soon as the appropriate instruction is issued. In a handshaking protocol, the computer does not accept or send data until it receives a ready signal from the outside world. The PCI-MIO-16E-4 8 bit port operates in immediate mode. Moreover, this port is latched. That is the state of the port remains stable until another instruction calls for an update.

**Applications.** As an input device, the digital port can be used to sense the status of a binary (on-off) device, e.g. to determine whether a switch is in the on or off position, or whether the liquid level in a tank is high or low. As an output device, the digital port can

be used to send commands to an actuator, e.g. to open or close a valve or to turn a pump on or off. Often digital input and output lines are used in combination to implement control systems. As an example, consider the use of digital I/O to control the level of liquid in a tank. In the simplest scheme, the status of a digital level sensor is used to turn on or off a pump, which fills the tank from a reservoir. More sophisticated tank control systems would involve four level sensors: the high and low target level sensors and two fault sensors, tank empty and tank overflowing.

Another application of the parallel I/O port is to control the rate and direction of rotation of a stepping motor. In this device the rotor steps (rotates) incrementally to discrete positions that are fixed by the positions of the coils within the motor. Changing the patterns of energization of the coils causes the rotor to step to the new equilibrium position. Typical stepping motors have four coils whose status can be controlled from a parallel digital port.

**Programming the Port.** In this week's lab you will be using the DAQmx Assistant express vi (Functions>>Input>>DAQassist) to write to and read from the binary port on your data acquisition board. After selecting this express vi, you will be prompted to customize its function by the assistant wizard.

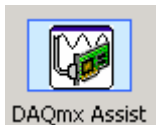


Figure 2. DAQmx Assistant express vi

We will customize the first express vi as a write to digital port vi. The user should select “Digital I/O” and then “Port Output” as the measurement type, and then select “port 0” as the physical device. After selecting “Finish,” a help window describing the customized vi with test function appears. At this juncture, you will want to use the test functions to see whether the hardware setup is correct. Then you can select okay to return to the block diagram. It will resemble Figure 3 once you create a control for the data terminal.

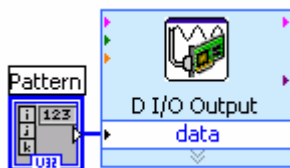


Figure 3. DAQmx Assistant express vi customized as a Digital Output with coded *Pattern* at the first cell of an array of data.

Here, ‘*Pattern*’ is the data the VI writes to the digital channel. For example, a decimal pattern of 15 results in an output pattern of 00001111 (for an 8-bit port). The input then is an array since the data is entered as an array (string of ones and zeros). Because each element of the array controls one separate output of the eight outputs, it is possible to control multiple (up to eight) devices with appropriate input patterns. Since we only have one output device, we put our decimal pattern in the first cell of the *Pattern* array.

We shall not consider the other input and output lines on DAQmx Assistant express vi at the moment.

The second express vi will be a read from digital port vi. For this vi invoke the DAQmx Assistant express vi as before and select “Digital I/O” and then “Port Input” as the measurement type. Perform further customization in analogy with the first express vi.

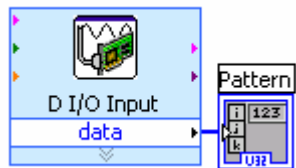


Figure 4. DAQmx Assistant express vi customized as a Digital Input that outputs coded *Pattern* at the first cell of an array of data.

Note that *pattern* is now an output and to see it you will need to wire it to an indicator or display.