1 Computer Measurement

What does someone do who wants to use computers for data acquisition? No matter which way the goal is pursued, it's necessary to know how computers communicate with the external circuits which make measurements. This chapter summarizes choices and introduces the approach used in this book.

1.1 Ports and Expansion Slots

Measurement Circuits (MC's) can be connected to computers through game ports, serial ports and expansion slots. In each case, software must send values to and get values from the MC's. The following sections review and compare the approaches.

1.1.1 Game Port Communication

On an Apple II, communication between its game port and an MC occurs as follows. Up to three digital signals generated by an MC are connected to the three game port push buttons. The computer ascertains a signal's high or low state by peeking a specified address and then seeing if the value is greater than 127 which means the signal is high. Similarly, one or more of the game port's four enunciator outputs are connected to an MC. The computer makes an output high by peeking a specified address and low by peeking a different address.

Game port data acquisition is practical when the number of input and output bits can be limited which is the case with the few commercially available systems. They are typically self-contained units starting with, for instance, a temperature probe and ending with a data plot. Their low cost, reliability and operational simplicity make them attractive for instructional laboratories.

1.1.2 Serial Communication

The most widely adopted standard in computers is RS-232 serial communication. For virtually any machine, serial ports either are or can be installed (at low cost).

To communicate serially, an MC must have a Universal Asynchronous Receiver/Transmitter (UART). This large scale integrated circuit receives a byte one bit at a time from the computer's serial port and makes the value available as eight separate signals. Similarly, the chip takes eight signals (generated by the MC) and sends them one bit at a time to the computer's serial port which assembles the value. To realistically set up and control the UART, the MC needs its own microprocessor, memory and software. This means serial data acquisition systems are also computers.

Even at maximum bits-per-second, serial communication is relatively slow. Therefore, measuring one voltage thousands of times a second or several voltages hundreds of times a second requires the MC to temporarily store data values in its own memory. Then, after acquisition, the host computer inputs the values. Data acquisition and storage are most efficiently carried out by the same microprocessor which manages the UART.

Serial systems are necessarily complex and require two levels of software: programs for the MC's processor and programs for the host computer (which include calls to the system software which operates the serial port).

A number of excellent serial data acquisition systems are available for popular computers. They are typically self-contained and come with all hardware and software.

1.1.3 Parallel Communication

Apple II, Macintosh II, IBM PC/XT/AT, IBM PS/2 and all compatibles have expansion slots. Communication between an MC occupying a slot and the computer is fast and easy. For systems with Intel microprocessors, an 8- or 16-bit value is sent to an MC by execution of an out to I/O port. Similarly, an 8- or 16-bit value is obtained from an MC by execution of an input from I/O port. With 8 or 16 bits sent or received in parallel, communication is hundreds of times faster than is possible with serial or game ports.

Data acquisition systems with every imaginable capability are available for the expansion slots of popular computers, especially IBM PC's.

Since these systems work only in the expansion slot for which they were designed, they cannot be moved from say an IBM PC to an IBM PS/2.

1.2 GPIB Standard

Because serial MC's are overly complex and expansion slot systems are not portable, a need exists for a fast parallel interface which makes different computers "look" the same to data acquisition circuitry. Of the several standards proposed over the years, by far the most successful is the IEEE-488 General Purpose Interface Bus (GPIB). It presents to an MC eight connections for sending and receiving data and eight management lines. The protocol is sufficiently elaborate that multiple MC's can be simultaneously interfaced. In addition, a variety of printers, plotters and other devices work through GPIB.

GPIB systems may be purchased for any computer with expansion slots. Versions are also available which interface serially (although their operation is slower). GPIB's usually come with software which initializes and sets up the system and then communicates with whatever MC's and other devices are connected.

Because GPIB makes different computers "look" the same, manufacturers can produce a variety of MC's knowing the market is independent of whatever computer is currently popular. And users can purchase MC's knowing they can upgrade computers without replacing their data acquisition systems.

1.3 Parallel I/O Ports

Another communications approach has many of GPIB's advantages and some of its own. Parallel I/O ports can be acquired for any computer with expansion slots. An OUT (or POKE) operation sends an 8- or 16-bit value to a port where it remains until the next OUT (or POKE). An IN (or PEEK) operation gets the 8- or 16-bit value currently at an input port. MC's connected to the ports supply the inputs and use the outputs.

The ports make different computers "look" the same to MC's and have the advantage of fast parallel communication, the same as GPIB. However, the ports require simpler and less expensive hardware. And software is just sequences of OUT's and IN's (or POKE's and PEEK's) as opposed to calls to system routines supplied by the manufacturer.

Parallel ports are a reasonable interface for data acquisition systems. Also, they are especially helpful for the teaching goals of this book. Their design in Chapter 3 illustrates and reinforces the digital electronics in Chapter 2. After visualizing IN's and OUT's, Chapters 4, 5 and 6 explain and demonstrate how software uses the ports to operate an analog-to-digital converter, an 8K static RAM and a programmable interval timer. The later chapters show how to combine these devices into a variety of useful MC's.

2 Introduction to Digital Electronics

Modern Integrated Circuits (IC's) have changed the effort and skills required to design and construct systems which carry out sophisticated digital operations. While it's desirable to understand how IC's work in terms of semiconductor physics and while good engineering practices are necessary in commercial products, it's now possible to successfully put together data acquisition hardware without first mastering these subjects.

Only fifteen different IC's are required for the parallel ports circuits of Chapter 3, for the analog-to-digital converter, memory and programmable interval counter circuits of Chapters 4-6, and for the measurement systems of Chapters 7-10. The purpose of this chapter is to introduce these and a few other IC's as well as associated design principles. Section 2.1 covers basic logic gates. Section 2.2 introduces integrated circuits, breadboarding and troubleshooting. Section 2.3 presents digital clocks. Section 2.4 introduces the 7474 flip-flop and a variety of timing and control operations. Section 2.5 outlines electrical properties of logic gates. Section 2.6 presents binary, hexadecimal and decimal number systems. Section 2.7 covers the 74192 and 74193 counters and several applications. Section 2.8 presents the 74138 decoder/ demultiplexer and additional control applications. Section 2.9 introduces the 74152 multiplexer. And Section 2.10 presents tri-state gates, buses and the 74373 and 74244 buffers. While these topics provide a complete background for this book, readers are referred to Appendix A for more comprehensive introductions to digital electronics.

In general, a logic gate may be thought of as shown in Figure 2.1. It has one or more inputs and one or more outputs, and requires +5 volts and ground. The inputs must be in one of two states (+5 volts or zero volts). The outputs, with an exception discussed later, must be in one of the same two states. Depending on the circumstances, +5 volt inputs and outputs are thought of as **true, on, high** or the digit **1**. Zero volt inputs and outputs are referred to as **false, off, low** or the digit **0**.

> *Actually, low is indicated by a range of voltages around 0 and high by a larger range around +5. Sometimes it's necessary to worry about where low ends and high begins, but not here.*

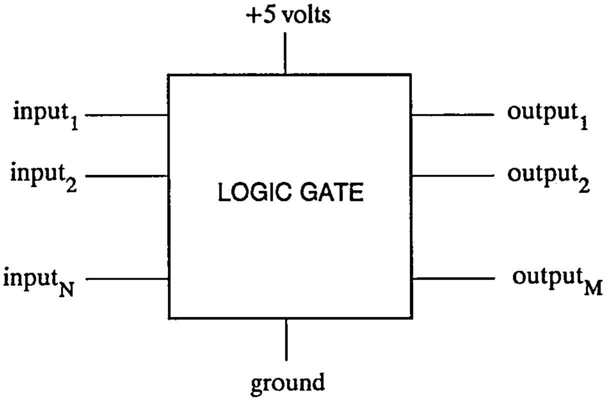


Figure 2.1. Logic gates have one or more inputs and one or more outputs. Inputs and outputs are either high (+5 volts) or low (zero volts).

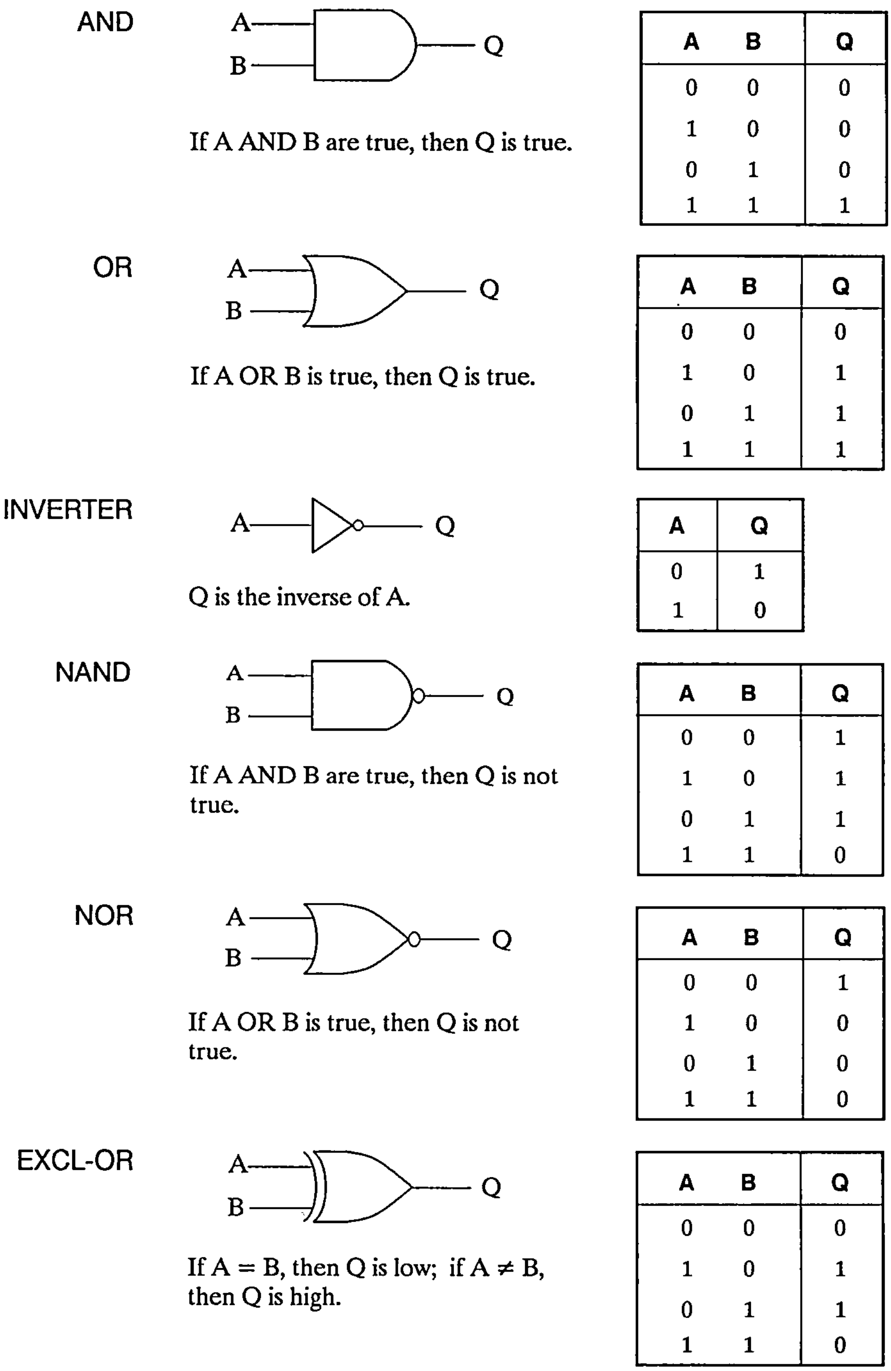
Logic gates are combined to carry out desired operations. This is accomplished on two levels: symbolic and actual. Circuits are thought about in efficient symbolic terms. Afterward, they are implemented by wiring together actual integrated circuits. The approach in this chapter is to first present logic operations and then discuss building, testing and troubleshooting procedures.

2.1 Basic Logic Gates

Basic logic gates are characterized by one or more inputs and a single output. A common way to specify a gate's operation is with a truth table which gives the output for every possible combination of inputs.

Only three basic gates are required for all circuits in this book: AND, OR, and INVERTER. Their circuit symbols, logic operations and truth tables are given in Figure 2.2. For completeness, NAND, NOR and EXCLUSIVE-OR are described. For all gates except the INVERTER, it's straightforward to extend the logic operation, circuit symbol and truth table from two to three or more inputs.

As can be seen from the truth table, the output of an AND gate is high only if all inputs are high. On the other hand, the output of an OR gate is high if only one input is high. An INVERTER gate's output is always opposite the input. NAND and NOR gates are the same as AND and OR, respectively, with the outputs inverted. Finally, the output of an EXCL-OR is high if two inputs are not equal and low only when all inputs are equal. The following sections show how to use the truth tables in Figure 2.2 to think about and apply basic gates.



AND

If A AND B are true, then Q is true.

A	B	Q
0	0	0
1	0	0
0	1	0
1	1	1

OR

If A OR B is true, then Q is true.

A	B	Q
0	0	0
1	0	1
0	1	1
1	1	1

INVERTER

Q is the inverse of A.

A	Q
0	1
1	0

NAND

If A AND B are true, then Q is not true.

A	B	Q
0	0	1
1	0	1
0	1	1
1	1	0

NOR

If A OR B is true, then Q is not true.

A	B	Q
0	0	1
1	0	0
0	1	0
1	1	0

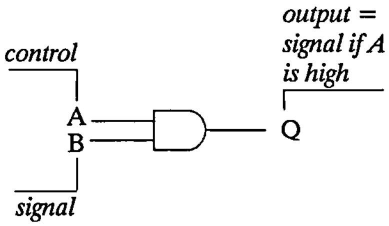
EXCL-OR

If A = B, then Q is low; if A ≠ B, then Q is high.

A	B	Q
0	0	0
1	0	1
0	1	1
1	1	0

Figure 2.2. Circuit symbols and truth tables for six basic logic gates.

Figure 2.3. An AND application where one input is a control which determines whether or not the signal at the other input passes to Q.



2.1.1 AND

An application of AND logic is to think of one of the inputs as a control and the other as a "signal," such as a digital clock. If the control input A is high, the output Q equals the signal input B, be it high or low or alternating between high and low. This can be seen by looking at input B and output Q on the rows of the truth table where A is high. On the other hand, when control is low, the output is low no matter what the signal is. This can be seen by looking at the rows of the truth table where A is low. So the signal passes the AND gate only when control is high. The idea is shown in Figure 2.3.

Consider the four-input AND gate in Figure 2.4. The output is high only if all four inputs are high.

2.1.2 OR

A way to think about OR logic is that the output is low only if all inputs are low. This helps visualize the operation of the circuit in Figure 2.5. With the two inverters, the four inputs to the OR gate are all low only when the 4 bits A, B, C and D are, in order, 0110. If any bit is different, at least one OR input is high and therefore the output is high.

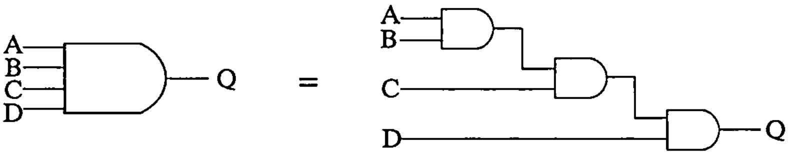


Figure 2.4. A 4-input AND gate implemented with three 2-input gates.

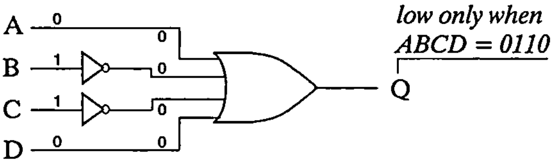


Figure 2.5. An OR based circuit.

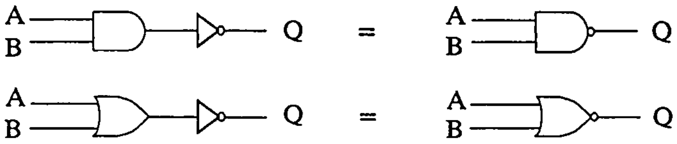


Figure 2.6. NAND and NOR gates are AND and OR gates inverted.

Figure 2.7. An INVERTER made from a NAND.

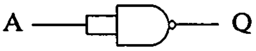
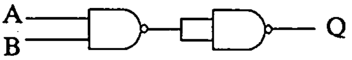


Figure 2.8. An AND made from two NAND's.

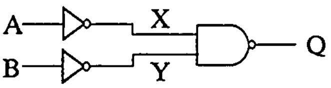


2.1.3 INVERTERS, NAND'S and NOR'S

NAND and NOR gates are identical to AND's and OR's with outputs inverted as shown in Figure 2.6. Similarly, the outputs of NAND's and NOR's sent through inverters produce AND's and OR's.

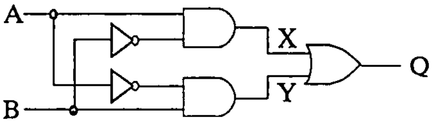
An inverter can be created from a NAND gate by connecting the two inputs together as shown in Figure 2.7. The operation is verified by just looking at the respective rows of the truth table where both A and B are the same. The circuit in Figure 2.8 shows an AND gate made from two NAND's.

Consider the circuit in Figure 2.9. The expanded truth table shows that the circuit carries out OR logic. An INVERTER added to the output would produce NOR logic. So, in addition to INVERTER and AND gates, OR's and NOR's can be made from NAND's. Since the flip-flop, counter and decoder/demultiplexer circuits presented in later sections are all implemented with basic gates, "everything" can be built from NAND's. Hence, it's the most basic gate.



A	B	X	Y	Q
0	0	1	1	0
1	0	0	1	1
0	1	1	0	1
1	1	0	0	1

Figure 2.9. Circuit and truth table for an OR gate made from a NAND and INVERTERS.



A	B	X	Y	Q
0	0	0	0	0
1	0	1	0	1
0	1	0	1	1
1	1	0	0	0

Figure 2.10. An EXCLUSIVE-OR circuit and explanatory truth table.

2.1.4 EXCLUSIVE-OR

EXCLUSIVE-OR logic is implemented by the circuit in Figure 2.10. The truth table shows the intermediate states X and Y for all possible inputs. The final output is deduced by applying X and Y to the OR gate. The output is low when A and B are equal and high when they are not equal.

2.2 Integrated Circuits, Breadboarding and Troubleshooting

Digital circuits are thought about in terms of combinations of the symbols in Figure 2.2, but they are built using integrated circuits. The following five IC's contain the basic gates needed for this book.

7404	contains six independent INVERTER gates
7408	contains four 2-input AND gates
7432	contains four 2-input OR gates
7421	contains two 4-input AND gates
7425	contains two 4-input NOR gates (1G and 2G must be high)

Three additional IC's are introduced.

7400	contains four 2-input NAND gates
7402	contains four 2-input NOR gates
7486	contains four 2-input EXCL-OR gates

Pin assignments for all eight chips are given in Figure 2.11 and Appendix C.

Integrated circuits are implemented in a variety of logic "families" and "subfamilies," each with its own electrical characteristics. In this book, only TTL (transistor transistor logic) chips are used. The differences among TTL subfamilies (e.g., 7404, 74LS04, 74H04), while significant in engineering commercial products, are not important for the circuits in this book. However, popular LS chips are recommended, if available. Appendix A gives references which describe logic families and their characteristics.

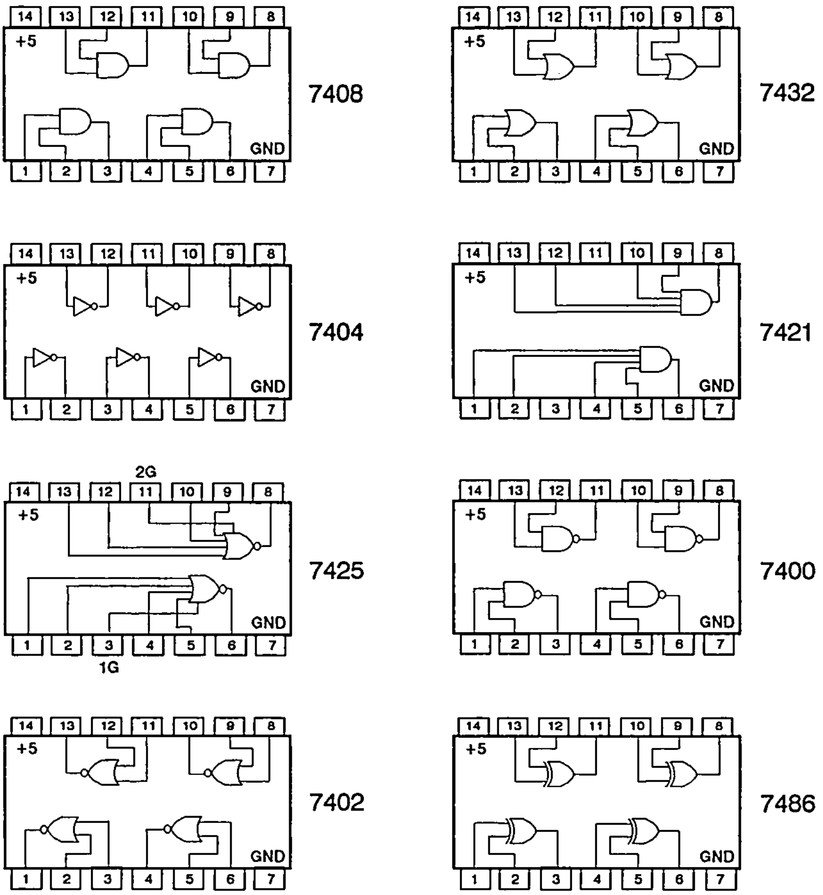


Figure 2.11. Pin assignments for eight basic gate integrated circuits.

2.2.1 Breadboard Systems

In order to build and test circuits in this and later chapters, a breadboard system is required. The E&L Instruments DD-1, shown in Figure 2.12, contains a +5 volt power supply, logic switches, pulsers, digital clock, light emitting diodes (LED's) and a breadboard on which IC's are wired. Chips are traditionally oriented so pin 1 is on the bottom left as shown.

A system LED is "lit" when connected to a high output. It is not lit when the output is low or disconnected. A logic probe distinguishes between disconnected and low and is an essential tool for troubleshooting.

Appendix B lists vendors for breadboard systems and logic probes.