Agilent Technologies
EducatorsCorner.com Experiments

Reverse-Engineering a PC Keyboard

By: Michael A. Covington
Anil Bahuman
David Barnhard
B. C. Bridger
Fred Maier
Artificial Intelligence Center
The University of Georgia
Athens, GA 30602-7415
mc@ai.uga.edu

Abstract

This is a lab exercise to familiarize students with the analysis of digital signals using an Agilent 54645D, Agilent 54645A, or similar digital oscilloscope. The exercise consists of discovering the serial code used for communication between the keyboard and system unit of a PC.

Equipment

- Agilent 54645D or Agilent 54645A Oscilloscope
- PC-compatible computer
- PC keyboard
- A test jig for accessing the five conductors of the keyboard cable during use. This can consist of the appropriate 5- or 6-pin DIN socket and plug, connected together in such a way that the terminals are accessible (see picture in Section 3).

1. Oscilloscope familiarization procedure

This procedure presumes no prior experience with oscilloscopes. However, prior experience is very helpful. Complete newcomers to the oscilloscope will probably need to consult its instruction manual and/or their instructor.

Background:

An oscilloscope is a device that draws a graph of voltage versus time (a “waveform”). Before the microcomputer era, this was only possible if the waveform was recurrent so that it could be displayed over and over to produce a visible trace on the screen. Constantly changing waveforms would produce a constantly changing trace.

Modern oscilloscopes use digital technology to analyze and display the input signal; they can remember, and continue to display, a signal that existed only for a short time. We will make crucial use of this capability when capturing and analyzing serial signals from the keyboard of a PC.

Procedure:

(1) Turn on the oscilloscope and let it boot up. Adjust the brightness control as necessary so you can see the display.

(2) Reset the oscilloscope to clear any settings made by the previous user. This is done by pressing Setup, then pressing Default Setup just below the screen.

(3) Attach probes to the input BNC connectors A1 and A2. You will need a grounding clip (a black wire with an alligator clip) installed on the probe that goes to input A1, but not on the other probe.

(4) Clip both probe tips to the calibration signal that is provided on the front panel of the oscilloscope, just below the screen. (It’s labeled “0V – 5V, 1.2 kHz.”) Leave the ground clip unconnected.

(5) Confirm that the oscilloscope is running (actively accepting data). The upper right corner of the screen should say RUN. If it says STOP, press

Run/Stop.

(6) Press **Autoscale**. The oscilloscope will automatically adjust itself for the amplitude and frequency of the incoming signal. At least, it will try to; with a simple signal like this one, it will do well. (Fortunately, the signal you are looking at is very similar to the signals used by the PC keyboard you'll be investigating.)

(7) You should see two square waves on the screen.

(8) Unclip the probes one at a time; note which is A1 and which is A2. Reconnect them.

(9) Try the **Volts/Div** and **Position** knobs for both A1 and A2. Note that the vertical scale (volts per division) is displayed at the top of the screen. Choose 2 volts per division for both A1 and A2. Place both waveforms at convenient positions.

(10) Try the **Time/Div** knob; note that the time scale (milliseconds or microseconds per division) is shown at the top of the screen. Adjust it to a convenient setting, one that enables you to see the shape of the waveforms.

(11) Using the grid on the screen, measure the height and width of the pulses you are viewing. Note that you can change the scale to magnify the waveform horizontally or vertically and get a more accurate reading. (Typical values: 5.0 volts high, 400 microseconds wide.)

(12) Try disconnecting the probes one at a time again, and note that the display rolls wildly in a horizontal direction when one of them is disconnected but not the other.

The reason for this is that you are viewing a recurrent waveform, and the display will only hold still if the oscilloscope's sweeps across the screen are synchronized with the recurrence of the same point in the wave. This is called *triggered sweep* and is controlled by the **Trigger** portion of the control panel, which we'll get to.

(13) You can also disconnect a probe from the oscilloscope momentarily by pressing the **Ref** button on its side. This is a quick way to see where the 0-volt level is on the screen. Try this also, with each probe.

(14) On the **Trigger** portion of the control panel, press **Edge**. A menu appears under the screen. Note that you can tell the oscilloscope to trigger on (synchronize with) either A1 or A2, on either the rising or falling edge of the waveform. Try it.

(15) Select trigger source A1, rising edge.

(16) Stop the oscilloscope by pressing **Run/Stop**. The display freezes. Demonstrate that you can disconnect the probes without affecting it.

(17) Press **Single**. This tells the oscilloscope to make a single sweep across the screen and freeze again. Try this out.

(18) Start the oscilloscope running again. You are ready for the next step in the exercise.

You may also want to investigate some other waveforms, such as the output of your function generator if one is available, and even the waveform that is picked up by your body through capacitive coupling with the surrounding electrical wiring. Given that the human body picks up so much electrical noise, can you make an educated guess about how an electrocardiograph works?[1]

Grounding and the ground clip:

If you are new to electronics, you are probably a bit unsure of what to do with the ground clip on the oscilloscope probe. Before you fry something, we should take a moment to explain grounding.

[1] In case you can't, the basic idea is that it measures the difference in voltage between several different points on the body, all of which are picking up approximately the same noise.

The ground clip of the oscilloscope is connected to the ground wire of the power line.

Most of the equipment you work on – computers, for example – has its chassis connected to power line ground. This “circuit ground” is used as a common reference point for all voltages and signals.

When working on line-powered equipment, the only safe place to connect the ground clip is circuit ground. If you connect it anywhere else, you are creating a short circuit.

When working on battery-powered equipment, this does not apply, but you should still connect the ground clip to circuit ground (often the negative terminal of the battery) for best protection against electrical noise.

Bear in mind that a “point” in a circuit can cover quite a bit of space. “Circuit ground,” for instance, usually comprises the chassis, the shields on all the shielded-cable connectors, and several wide traces on every printed circuit board, as well as narrower traces that branch off of them. (Not every wide trace is grounded, of course; some of them carry power or serve other purposes.)

2. Advanced triggering: Capturing a transient signal

In this exercise, you’ll need to make the oscilloscope wait for an incoming signal, display it, and then freeze when the signal disappears, so you can still see it after it’s gone. Here’s how to do this.

(1) Start with both probes connected to the 5-volt test signal, and the oscilloscope running.

(2) On the **Trigger** portion of the control panel, press **Mode/Coupling.** Instead of **Auto Level**, change the selection to **Normal.** That means, “Don’t sweep unless the input signal exceeds the preset trigger level.”

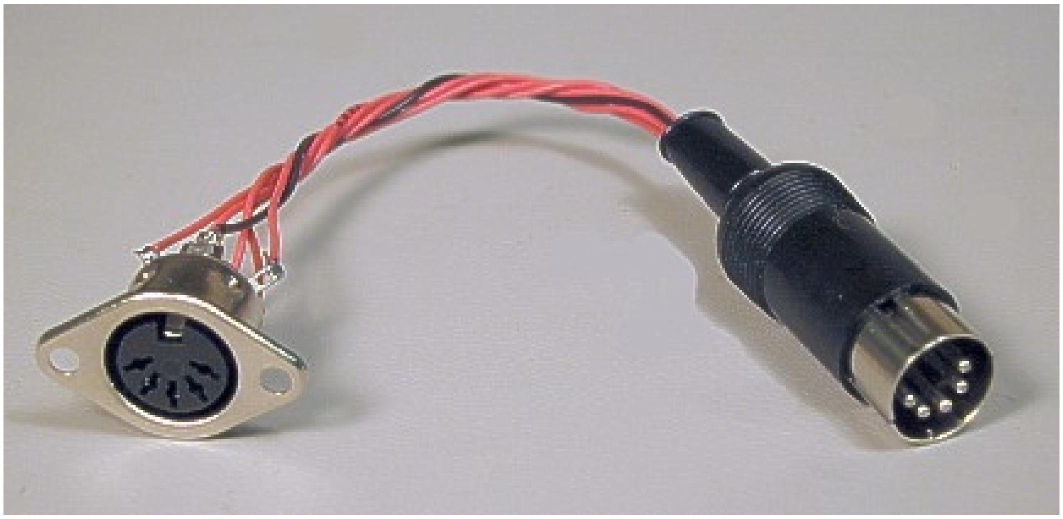
(3) Adjust the **Analog Level** knob to put the trigger level halfway up the height of the pulses. Now the oscilloscope won’t run until the input signal on A1 (the selected trigger input) reaches this level.

(4) Disconnect the A1 probe from the signal source. Observe that the display freezes.

(5) Connect the A1 probe to the test signal again. Notice that the display resumes being updated.

3. Exploring the keyboard signals

Now you’re ready to tap into the communications between a PC and its keyboard. That is most easily done by building a test jig out of the appropriate DIN or mini-DIN plug and socket, joined together with individual wires, like this:



Be sure to connect the outer shells of the two connectors together, as well as the five or six pins. *Check with an ohmmeter to make sure each pin actually goes to the corresponding socket hole;* be sure not to get everything mirror-imaged. When the jig is ready, proceed as follows:

(1) Connect the test jig between PC and keyboard and boot up the PC normally. Run a piece of software that will let you press keys randomly. Under DOS, the command

```
COPY CON: NUL:
```

will make the PC ignore all input until you hit either Ctrl-Break or Ctrl-Z.

(3) Now turn your attention to your test jig. The outer shell of each connector is grounded. Connect the oscilloscope's ground clip to that point in the circuit.

(4) Your task is now to find out what is on each of the connector pins. You will find that:

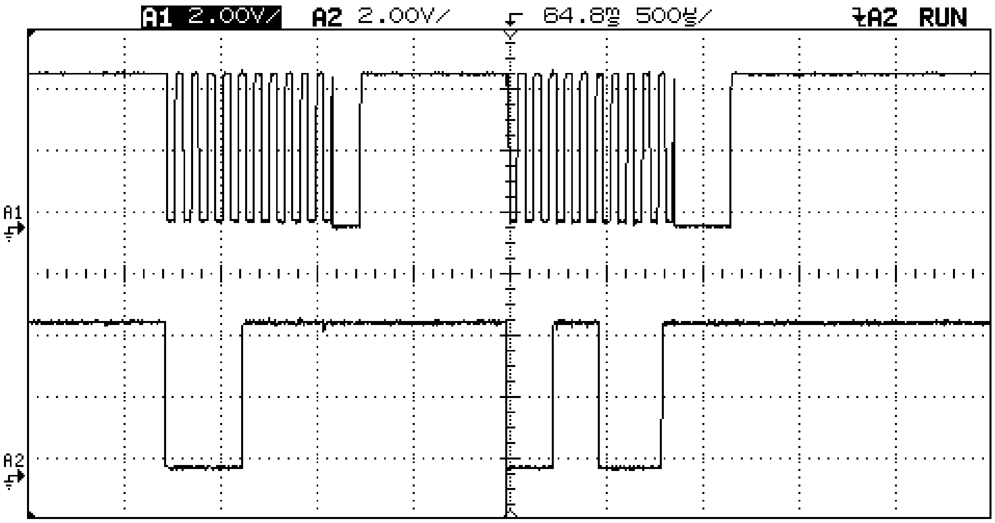
- One pin is grounded;
- One pin carries +5 volts DC;
- Two of them carry signals when you press a key;
- One or two pins are not used (carry no signal), though they may pick up weak noise signals from other wires in the cable.

To find out which is which, connect the oscilloscope probe to each pin in succession while you try pressing keys. Make careful notes of your results.

(5) Now that you know which two pins carry signals, your next task is to capture the signals.

Connect one oscilloscope probe to each of the two signal pins, and use "normal" triggering (as in section 3) to capture exactly what happens when you press the A key. Arrange it so that the display will be updated only when there is a signal.

In order to see what's going on, set **Time/Div** to 20 ms, and trigger on the falling edge, not the rising edge. Use the **Delay** knob to shift the display horizontally until you can see all of it. Then switch to 500 µs to expand the display. If you captured the signal from the A key, you'll see something like this:

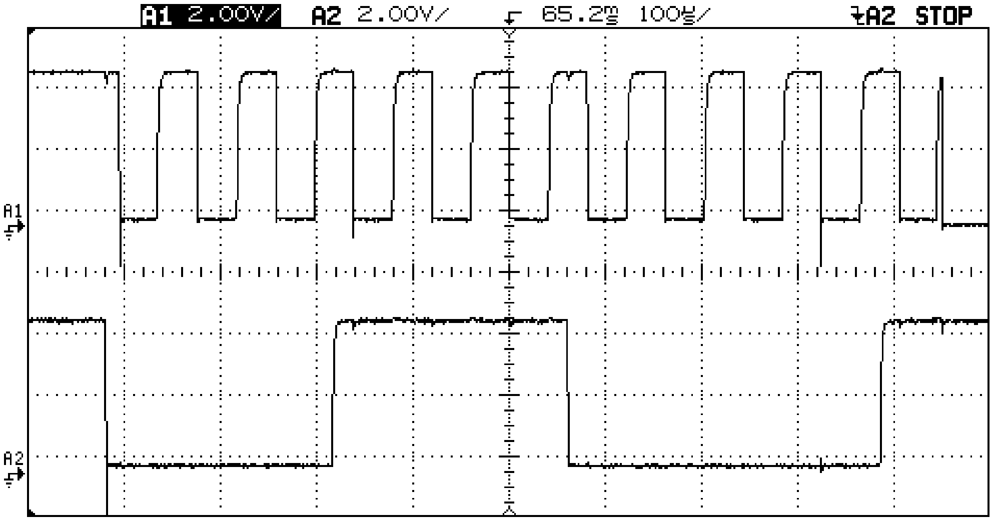


(6) Your next task is to decode these signals. What you are looking at is called *synchronous serial communication.* Here are some basic facts about it:

- One data packet is transmitted when you press a key, and two data packets are transmitted when you release a key.

- One of the signals is the *clock* and always makes 11 downward transitions per data packet.
- The other signal is the *data.*
- Each downward transition of the clock indicates *when* to read the data (as 0 or 1).

So, taking +5V to be 1 and 0V to be 0, this data packet:



represents binary 00011100001.

Or does it? Now the real fun begins. According to IBM's documentation, each sequence of 11 bits actually consists of:

- A start bit, always 0;
- Eight bits of data, *backward* (least significant bit first);
- A parity bit, indicating whether the number of ones in the data was odd or even;
- A stop bit, always 1.

So the signal that we initially read as 00011100001 actually decodes as:

- 0 = start bit;
- 00111000 = binary 00011100 (hex 1C) transmitted LSB first;
- 0 = parity bit (there was an odd number of ones in the data);
- 1 = stop bit.

With generous use of the **Delay** knob and **Time/Div** adjustment (both of which you can twist even when the display is frozen, thank goodness), you can now read keyboard scan codes from the face of the oscilloscope.

5. Your assignment

Experiment and answer the following questions:

(a) What is the relationship between the packet sent when a key is pressed, and the pair of packets sent when it is released?

(b) Does the keyboard use ASCII codes or something else?

(c) What about the shift key? Does it modify other keys or does it just send codes of its own?

(d) Are there any keys that send more than one or two packets?

(e) Notice that after each data packet, the clock line goes low for a long time (the width of three or more ordinary clock pulses). This is an "acknowledge" signal sent back from the PC to the keyboard.

Are there any other signals sent from the PC to the keyboard? How is it possible to communicate in both directions over the same wire?[2]

6. Where to go from here

Because PC keyboards are cheap and abundant, they are a handy source of input for microcontroller circuits. See the bibliography for two articles on how to use them as such.

Synchronous serial communication is very commonly used in embedded systems. It is, for instance, the basis of the I^2C (Inter-Integrated-Circuit) bus and other microcontroller communication protocols. Now you know how to crack the codes whenever you have to investigate such a system.

Bibliography

Peacock, Craig. "Using a PC Keyboard." *Poptronics,* July 2000, pp. 48—53, 69.

Philips Semiconductors. *Connecting a PC Keyboard to the I2C Bus.* Application note AN434 (1992), available online at *http://www.semiconductors.com/products/all_appnotes.html.* (Very detailed; includes 8051 assembly code.)

Sargent, Murray, and Shoemaker, Richard L. 1985. *Personal Computer from the Inside Out: Programmer's Guide to Low-Level PC Hardware and Software.* 3rd edition. Reading, Mass.: Addison-Wesley.

-end-

[2] If, after thinking for a while, you can't figure it out, here's the secret. The wire is connected to to +5V with a resistor. Signals are sent by momentarily connecting the wire to ground (through a switching transistor inside a logic gate). This pulls its voltage down to 0V without causing excessive current to flow. That's why the signal level is always +5V when nothing is being transmitted. The wire can safely be connected to ground at either end, or even both at the same time. It can also be sensed at both ends. This is how all computer buses work. It would not be safe for two logic gates to actually drive signals into the wire from opposite ends at the same time.