Behavior Research Methods
2007, 39 (4), 797-801

How to make your own response boxes: A step-by-step guide for the construction of reliable and inexpensive parallel-port response pads from computer mice

Andreas Voss, Rainer Leonhart, and Christoph Stahl
Albert-Ludwigs-Universität Freiburg, Freiburg, Germany

Psychological research is based in large parts on response latencies, which are often registered by keypresses on a standard computer keyboard. Recording response latencies with a standard keyboard is problematic because keypresses are buffered within the keyboard hardware before they are signaled to the computer, adding error variance to the recorded latencies. This can be circumvented by using external response pads connected to the computer's parallel port. In this article, we describe how to build inexpensive, reliable, and easy-to-use response pads with six keys from two standard computer mice that can be connected to the PC's parallel port. We also address the problem of recording data from the parallel port with different software packages under Microsoft's Windows XP.

It is well-known that standard computer keyboards are suboptimal for the recoding of response latencies because keypresses are not immediately forwarded to the computer's central processing unit (CPU); rather, all keyboard events occurring within a certain time frame (usually 10–30 msec) are stored in a buffer within the keyboard before the contents of that buffer are forwarded to the computer's CPU for processing. Due to this buffering, a variable delay is added to each response latency that depends on the occurrence of the response within the buffer window, and as a result, increases the error variance of the response latency measure (see, e.g., Crosbie, 1990; Plant, Hammond, & Turner, 2004; Segalowitz & Graves, 1990). Because external response pads do not contain a buffer, they are the better choice for recording response latencies, especially whenever small effects are expected, or when intervals between two consecutive responses are measured (Segalowitz & Graves, 1990). A number of different response devices are available, and they usually connect to the computer's USB, serial, or parallel port. Most of the commercially available response boxes actively preprocess the incoming signal, which is necessary for devices that connect to the USB or the serial port, and they therefore need an external power supply. In our view, the most straightforward method of recording keypresses is via the parallel port,[1] because no preprocessing of signals in the response device is necessary, the timing is most accurate (Beringer, 1992), and a parallel port device can be built with a minimum of electronics. We feel that the experimental setup in a lab should be as lean and simple as possible, and for this reason, we provide a description of inexpensive, easy-to-build, and nevertheless highly reliable response pads that can be plugged directly into the parallel port and do not need any additional devices for power supply or preprocessing. In the following, we give a step-by-step guide on how to build your own response pads for the parallel port (see Morris, 1992, for a description of a serial port device). We further address the timing accuracy of the device, and describe how the device's signals can be registered as responses by standard experimental software.

Hardware: The Response Pads

For a response device, the central point is the mechanics of the keys or buttons; if buttons are not working properly, this will obviously cancel out all advantages of the direct connection to the parallel port. To solve this problem, we suggest using the buttons of standard computer mice. For experiments that require responding with two hands, two mice can be used simultaneously. With such a solution, up to six different buttons can be used for an experimental setup.

First, we want to give a word of warning: Wiring a parallel port device the wrong way can, theoretically, cause damage to important parts of your computer (e.g., the mainboard). Therefore, some care is recommended in building the devices. In the following, we explain in a step-by-step guide to how to proceed.

Step 1: Choosing the response device. Before constructing the hardware for external response devices, the

A. Voss, voss@psychologie.uni-freiburg.de

first decisions that have to be made are how many buttons will be needed and how the buttons should be assigned to participants' hands. In this article, we will provide an example of a device with three buttons for each hand, built from two three-button computer mice. This six-button solution should be sufficient for a wide range of psychological experiments. However, using the input mode of a bidirectional parallel port, it is possible to attach up to eight buttons to one parallel port. Thus, any response device with up to eight buttons can be attached to the parallel port the way we describe it below. We decided to use simple old-fashioned three-button serial-port mice without a scrolling wheel or laser technology because they are cheap and have a simple electronic architecture.

Step 2: Removing the electronics. From the mice, only the mechanics of the button switches are needed. When you open the mice, you will probably find the switches soldered to the circuit boards. The boards should remain intact to keep the switches where they are. All other electronics have to be removed or disconnected from the switches by cutting off the conducting paths leading to the switches.

Step 3: Connecting the switches. Next, the switches should be connected with a new cable to a male-type D-SUB-25 plug for the parallel port. To connect your PC with the response pads, two cables are required in order to connect the mice to the parallel port plug. Each cable must contain four wires: one wire for the signal from each switch, and an additional wire for the ground. Figure 1 shows how the six switches of both mice are connected to the pins of the plug.

Some important points should be noted: First, each switch connects one pin of the data byte of the parallel port with the ground. This logic works only for ports that set 5 volts to all pins of the data byte (Pins 2–8), once the port is switched to input mode (see below). Unfortunately, parallel ports are not completely standardized, and different protocols are used (e.g., ECP or EPP; Axelson, 1997).[2] Some ports have an inverted logic in input mode: For these ports, by default there is no current at the eight data lines, and they have to be connected to a power source to record a signal. With the program pp_monitor available on our Web page (www.psychologie.uni-freiburg.de/Members/voss/research/response-pads), you can easily check whether this is the case for your parallel port. We will describe a solution for this problem below (Step 4).

Second, we recommend not short-circuiting the pins to the ground directly, because high amperages may occur. Rather, we suggest inserting a small resistor (100 Ω) between the data pin and the ground pin, as illustrated in Figure 1. Third, in our experience it is most convenient to solder the resistors directly to the plug so that they are enclosed within the plug's coverage, rather than connecting them to the switches within the mice.

Step 4: Dealing with inverted parallel ports. For parallel ports with an inverted logic—that is, for ports in which all pins of the data byte are set to zero when switched to input mode—one needs a different wiring of the key pads: In this case, the data pins have to be connected not to the ground but to a 5-volt source to record a signal. Fortunately, the port itself provides such a voltage, so no external power supply is needed. For parallel ports that show this effect, the switches are required to connect the data pins directly to a 5-volt pin. This can be achieved by connecting to Pin 1 instead of Pin 18 that is used in Figure 1. Alternatively, an adapter can be plugged in between the computer and the response pads constructed according to Figure 1. We prefer this procedure, because it provides a flexible solution and allows usage of the response pads on both types of parallel ports. Figure 2 provides a schematic diagram for the adapter. The central feature of the adapter is that the ground pin of the response pads (Pin 18 at the female plug; K1 in Figure 2) is connected to Pin 1 of the computer's parallel port (the male plug; K2 in Fig-

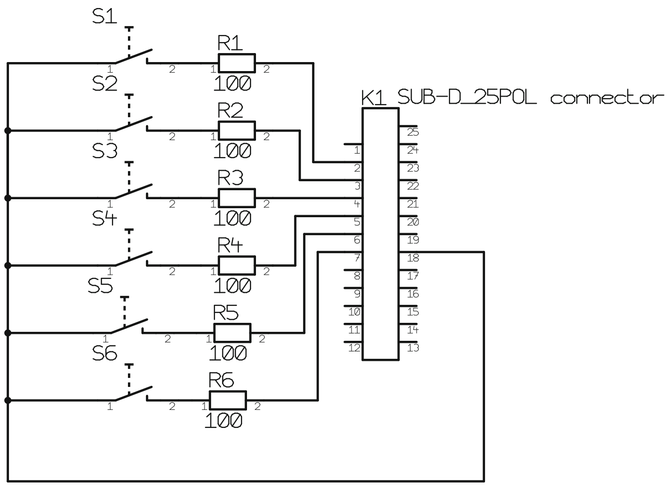

Figure 1. Circuit diagram for the response pads with six buttons. K1 is a D-SUB-25 male type plug for the parallel port. S1 to S6 are the buttons of the response device. R1 to R6 are 100-Ω resistors. If eight buttons are needed, it is possible to use Pins 8 and 9 likewise.

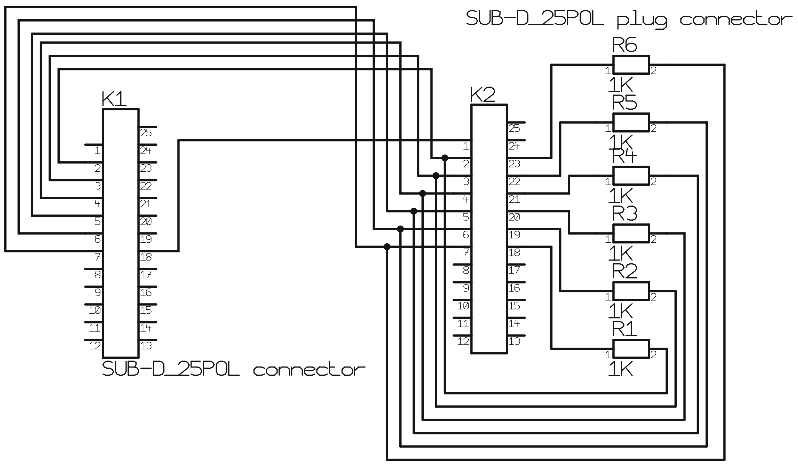

Figure 2. Circuit diagram for an adapter for inverse parallel ports. K1 is a D-SUB-25 female type plug (which can be connected to K1 in Figure 1), and K2 is again a D-SUB-25 male type plug for the parallel port. R1 to R6 are 1-kΩ resistors.

ure 2), which we use as the power supply. In addition, we recommend connecting all used data pins with the ground with a high resistor (1 kΩ). This guarantees that after a switch is opened, the voltage goes down again immediately. Again, the necessary resistor can easily be applied within the cover of one of the plugs, so this adapter looks like a simple cable with two D-SUB-25 plugs.

Reliability of the Response Pads

To test the reliability of our device, two experiments were conducted that address the accuracy with which the latencies of buttonpress responses are registered, as well as the potential problem of rebouncing effects.

Experiment 1: Timing accuracy. To check the timing accuracy of our devices, we replaced the mechanic switch in one of our response pads with a phototransistor (MCT2E), which works like an electronic switch. The phototransistor was then connected to an external high-accuracy clock signal generator. On a Windows XP computer with a dual-core Pentium 4 2.8-GHz processor, which was connected to the response pads, the offsets of 1,000 consecutive signals were recorded using the Windows high-performance timer. The offsets had a range of 0.129 msec and a standard deviation of 0.028 msec. These results indicate that the parallel-port devices work with excellent timing accuracy.

Experiment 2: Rebouncing effects. Rebouncing can occur when a mechanic switch is closed or opened: When both contacts are very close, the circuit might oscillate between a closed and an opened state for a very brief period before the final state is reached. If, during that brief interval, the state of the switch is assessed repeatedly, this rebouncing may be interpreted as multiple signals, when in fact only a single buttonpress was performed. The extent of rebouncing depends on the mechanics of the switch.

In most experimental settings, rebouncing is not problematic, since only the first response in a given trial is usually registered. However, if multiple consecutive responses with the same button are recorded, rebouncing might cause a single response to be registered as several. Because rebouncing occurs at a very short time frame, this potential problem can be avoided if the state of the response pad is queried only once every millisecond—a common sampling rate for experimental software and computer programming libraries. To demonstrate this empirically, one hundred manual responses were recorded with one of our devices, and the state of the buttons was registered using a timer from a standard C library with millisecond accuracy. In none of the 200 button events (100 rising edges and 100 falling edges) did rebouncing occur—that is, there was always an immediate change from 0 to 1 (or vice versa), without an oscillation between both values.

Whereas we believe that the devices we describe here are of high quality, please note that the quality of self-made response pads may not be as good as that of commercially available devices, because it depends on the mechanic quality of the switches, and the results obtained in Experiment 2 might thus not hold for different mice. We therefore recommend carefully evaluating all self-made response boxes before they are used for research. However, the results from Experiment 1 regarding timing accuracy are independent of the type of switch used and can thus be generalized.

Initializing the Parallel Port

Before the response pads can be used on the parallel port, it has to be set to bidirectional mode, and the data byte has to be set to input mode. In addition, we strongly advise removing any other device driver (e.g., a printer driver) that might access the parallel port and thus interfere with data recording.

Changing your port to ECP/EPP. Nowadays, almost all parallel ports are bidirectional—that is, the data byte can be used for input as well as for output. If you have any trouble with the response pads, you should check in your computer's BIOS whether your port is configured as ECP/EPP (see notes 1 and 2).

Switching the data byte to input mode. By default, the port is configured for output on the data byte. Because we want to read data, the port has to be switched to input mode before the response pads can be used. To prepare the port for input, Bit 5 of the status byte of the parallel port has to be set to "high." After this has been done, the parallel port is switched to input mode, and you can check whether your port sets the data bits low or high (see Hardware section). You can do this by measuring whether or not there is voltage between the data pins (Pins 2–9) and the ground (e.g., Pin 18). Alternatively, you can use the program pp_input from our Web page to check whether the bits are set or not.

Software: Addressing the Parallel Port Under Windows XP

To illustrate how responses can be registered from the response pads, we provide examples how this can be done with some of the most commonly used experimental software packages. For researchers with programming experience, we also demonstrate how this can be achieved with C code.

Registering Responses With Experimental Software Packages

After initializing the parallel port as described above, experimental software packages such as Inquisit, E-Prime, or DirectRT can register the press of a button on the response pads as easily as they can register keyboard responses. We provide a brief description how the port inquiry can be accomplished with each of the three packages (for implementations of these examples, visit www.psychologie.uni-freiburg.de/Members/voss/research/response-pads; for a review of the software packages, see Stahl, 2006).

Inquisit. By default, Inquisit (Millisecond Software, 2004) reads input from the status byte of the parallel port. However, with the present response pads, data needs to be read from the data byte. To accomplish this, the memory address of the parallel port has to be adjusted by minus 1. For example, the first parallel port, LPT1, is by default mapped to 0x0378. In this case, the data byte is at 0x0378, and the status byte is at 0x0379. Thus, Inquisit would read from 0x0379. Pointing the address of the LPT1 port to 0x0377 causes Inquisit to read from byte 0x0378, which is the correct address for reading response pad responses. In addition to specifying the memory address, byte codes have to be specified as valid responses that correspond to the valid buttonpresses. Otherwise, Inquisit would not wait for a buttonpress but would immediately take the default byte value for a response. For example, to accept responses from the left button of the right-hand mouse and the right button of the left-hand mouse (i.e., responses from both index fingers), specify 196 and 200 as valid response codes. To avoid Inquisit interpreting a single, longer buttonpress as multiple responses, use a "pretrial signal" command to instruct the software to wait for the default value before initiating a trial.

DirectRT. With DirectRT (Empirisoft, 2004), the address of the data byte can be specified in the response field of a trial description. In a given trial, to switch from keyboard to response pad, replace the keyboard response codes with those for the response pad, and add the decimal data port address in parentheses (e.g., "rt:196,200(888)" for the two index-finger buttons). To initialize the port, a single signal has to be sent through the port before responses can be registered. A sample experiment is provided with the package, illustrating both initialization and registering responses.

E-Prime. In E-Prime 1.1 (Psychology Software Tools, 2002), upon addition of the port device (available from the Edit/Experiment/Devices menu), the address of the data byte can be edited in the Properties menu of the port device. The port device can now be specified as the input device for experimental trials. Responses are coded and logged by the number of the button, counted from right to left, across both mouse (i.e., from "1" for the right button of the right-hand mouse to "6" for the left button of the left-hand mouse). Using these codes, valid and correct responses can optionally be specified.

Accessing the Parallel Port With a C Program

If you are programming your own experiment using Windows XP, you will need a software interface to access the parallel port (i.e., direct access to the parallel port is prohibited by the operating system). We recommend using the inpout32.dll, which is available on the Internet (e.g., at www.logix4u.net/inpout32.htm).[3] With this interface, you can address the port from any contemporary programming environment (e.g., C, C++, or Delphi). Here, we provide examples for the C language. Please also check our homepage (www.psychologie.uni-freiburg.de/Members/voss/research/response-pads) for some examples in C, including the source code.

Switching the data byte to input mode. Instead of using the program we provide, parallel-port initialization can be easily done with a few lines of C code:

```
#define kPP_BASE 0x378
#define kPP_STATUS kPP_BASE + 2
#define ONLY_BIT(b) ((2 < < b)/2)
_int16 x = Inp32(KPP_STATUS);
x = x | ONLY_BIT(5);
Out32(kPP_STATUS, x)
```

where 378 is the hexadecimal port address[4] and Inp32 and Out32 are commands provided by inpout32.dll.[5]

Recording responses. When the response pad is connected to the parallel port, a buttonpress is directly mapped onto one of Bits 1–6 of the data byte. You can check for a buttonpress with the following C code:

```
if  ((Inp32(kPP_BASE)  &  ONLY_
BIT(button)) = = 0) {
    /* Bit is low */
} else {
    /* Bit is high */
}
```

Again, 378 is the port address, and *button* is the number of the button in question, counting from 0 (for Pin 1) to 5 (for Pin 6). On most ports, a low bit will represent a buttonpress. If your data bits are high without a buttonpress, you will need the adapter described above, and buttonpresses are indicated by a low bit.

Conclusions

In this article, we describe external response pads to be connected to the parallel port that are superior in timing accuracy to standard computer keyboards. Accuracy is higher with our response pads because they do not make use of buffering that adds error variance to latencies recorded from a standard computer keyboard. Another advantage is the superior simplicity of a device with only a small number of keys, rendering it less likely that participants slip to a wrong key or are unsure which keys they have to use (Crosbie, 1990).

If one wants to record responses with a computer mouse, it is obviously possible to use a standard mouse connected to the serial port or the USB port. However, in this case similar timing accuracy problems occur as for the standard keyboard (Beringer, 1992).[6] An additional advantage of the response pads we describe is the possibility of combining two mice for two-handed responses.

Creating external response pads from computer mice has several advantages: First, the mechanics of the mouse buttons are usually of high quality and are very reliable. Second, materials for these response pads are very inexpensive (about €15 for two mice, cable, plugs, and several resistors). Third, they do not need an external power supply. Fourth, the construction process is simple: With some practice, one pair of response pads can be constructed in less than 1 h. Fifth, the devices are quite ergonomic and handy to use. Last but not least, they have proven to provide excellent reliability. For these reasons we are convinced that for registering response latencies with high accuracy, these self-made response pads are a good alternative to a standard keyboard as well as to commercially available external response devices.

AUTHOR NOTE

The research reported in this article was supported by Grant Kl 614/31-1 from the Deutsche Forschungsgemeinschaft to Karl Christoph Klauer. Correspondence concerning this article should be addressed to A. Voss, University of Freiburg, Department of Psychology, 79085 Freiburg, Germany (e-mail: voss@psychologie.uni-freiburg.de).

REFERENCES

Axelson, J. (1997). *Parallel port complete: Programming, interfacing, and using the PC's parallel printer port*. Madison, WI: Lakeview Research.

Beringer, J. (1992). Timing accuracy of mouse response registration on the IBM microcomputer family. *Behavior Research Methods, Instruments, & Computers*, **24**, 486-490.

Crosbie, J. (1990). The Microsoft mouse as a multipurpose response device for the IBM PC/XT/AT. *Behavior Research Methods, Instruments, & Computers*, **22**, 305-316.

Empirisoft (2004). DirectRT (Version 2004.1.0.55) [Computer software]. New York: Author.

Millisecond Software (2004). Inquisit (Version 2.0.41230) [Computer software]. Seattle, WA: Author.

Morris, C. C. (1992). Using the IBM-compatible microcomputer's serial port as an input–output interface. *Behavior Research Methods, Instruments, & Computers*, **24**, 456-460.

Plant, R. R., Hammond, M., & Turner, G. (2004). Self-validating presentation and response timing in cognitive paradigms: How and why? *Behavior Research Methods, Instruments, & Computers*, **36**, 291-303.

Plant, R. R., Hammond, N., & Whitehouse, T. (2003). *Behavior Research Methods, Instruments, & Computers*, **35**, 276-284.

Psychology Software Tools (2002). E-Prime (Version 1.1) [Computer software]. Pittsburgh, PA: Author.

Segalowitz, S. J., & Graves, R. E. (1990). Suitability of the IBM XT, AT, and PS/2 keyboard, mouse, and game port as response devices in reaction time paradigms. *Behavior Research Methods, Instruments, & Computers*, **22**, 283-289.

Stahl, C. (2006). Software for generating psychological experiments. *Experimental Psychology*, **53**, 218-232.

NOTES

1. Further information about parallel ports is provided by Axelson (1997). Also, it might be informative to check the Internet.

2. The enhanced parallel port (EPP) is a half-duplex bidirectional interface designed to allow a transmission of large amounts of data to the host. The extended capability port (ECP) is a half-duplex bidirectional interface, too, but it also allows data compression (run-length encoding).

3. Please note that Inpout32 works with a 32-bit architecture, whereas the original inp() routines provided in C work with 16 bits only. For the 64-bit version of Windows, a new port driver will be needed.

4. The default port address for LPT1 is 378, and 278 is the default address for LPT2. In laptop computers, other addresses might be used. You can check the correct address in the BIOS settings.

5. If you are using the adapter described in the Hardware section, you have to be careful not to change the first bit of the status byte (the "strobe") from 0 to 1, because this would turn off the power on Pin 1, which is used as the power supply.

6. Although the timing accuracy of some PS/2 mice seems to be excellent as well, accuracy may vary substantially even between mice from the same series (see Plant, Hammond, & Whitehouse, 2003).

(Manuscript received August 2, 2006;
revision accepted for publication December 8, 2006.)