*Behavior Research Methods, Instruments, & Computers*
*1990, 22 (3), 344-345*

## — BRIEF NOTES —

# A warning about millisecond timing in Turbo Pascal

MARC BRYSBAERT
*University of Leuven, Leuven, Belgium*

*It is shown that the "readkey" function in Turbo Pascal 4.0 runs in cycles of about 40 msec, and thus must not be used to wait for a keyboard response in reaction time studies.*

As we have argued elsewhere (Brysbaert, Bovens, d'Ydewalle, & Van Calster, 1989), millisecond timing involves more than a resolution of 1000 Hz. First, it is necessary to synchronize stimulus and timer onset. Second, the subject's response must be detected as accurately as possible.

The synchronization of stimulus and timer onset is well documented and is accepted in psychological research. All recent timing routines in this journal deal with it either by temporally disactivating the screen or by screen swapping.

Accurate measurement of the subject's response, however, has received attention only recently. Graves and Bradley (1987) compared timing performance on an IBM PC with reaction times recorded independently with a Gerbrands digital millisecond timer. They found that the standard IBM PC keyboard showed times that were slower by 18.4 msec on the average (*SD* = 4.3 msec). A keyboard from a clone with the same IBM PC showed times that were slower on the average by 36.7 msec (*SD* = 2.9 msec). Since greatly improved accuracy was obtained through the use of buttons or a joystick as the response devices, Brysbaert et al. (1989) recommended that the keyboard not be used as a response device, but that, instead, good external response buttons be connected to the game or the printer ports.

If the keyboard is used as a response device nevertheless, a second danger appears in the properties of the software. Turbo Pascal 4.0 has two wait functions for a keyboard response: readkey and keypressed. The first waits for a keypress and then returns the character of the key that has been pressed. The second merely indicates whether a key has been pressed or not. A readkey must always follow a keypressed to neutralize the keypress in Turbo Pascal 4.0.

At first, it is tempting to use the readkey function, because this shortens the Turbo Pascal program with one line. However, experience with millisecond timing reveals that the function runs in cycles of about 40 msec and thus severely reduces the resolution of the timer. As an illustration, consider an experiment in which the subject has to discriminate between the letters *a* and *b* by pressing two keys with the index (for *a*) or the middle finger (for *b*). Stimulus and timer onset are synchronized with the video_on procedure of Brysbaert et al. (1989, Appendix B). The stimulus remains visible until the subject has pressed a key. Mistakes are dropped out. The essential part of the Turbo Pascal program is the following:

```
video_on;  {wait for vertical retrace and display the stimulus}
gtel := 0;                                {start timer at zero}
ch := readkey;                              {wait for keypress}
saved[i] := gtel;                  {save time interval - msec}
```

Figure 1 displays the results of 1,000 trials by one subject. All 1,000 trials cluster around 17 values: 125, 206, 288, 328, 369, 410, 451, 491, 532, 573, 614, 654, 695, 736, 777, 818, and 981 msec (±1), which happen to be multiples of 41 msec. Because reaction times are a continuous variable, chances are very low that a subject could produce such a discrete pattern of results. Moreover, if we repeat the experiment with the same subject on an IBM XT, and modify the Turbo Pascal program as follows:

```
video_on;  {wait for vertical retrace and display the stimulus}
gtel := 0;                                {start timer at zero}
repeat until keypressed;                    {wait for keypress}
saved[i] := gtel;                  {save time interval - msec}
ch := readkey;                          {neutralize keypress}
```

we obtain the data on Figure 2, which are no longer grouped around a limited number of values but distributed over the entire range (271 different values). Thus, the peculiar pattern in Figure 1 is most likely due to the low resolution of the readkey function.

As we have shown, the readkey function of Turbo Pascal 4.0 reduces the resolution of a millisecond timer from 1000 to 25 Hz (1-40 msec). Because this is usually not what psychologists are interested in when they measure time intervals, and because the mistake can easily be corrected by using the keypressed function, we strongly advise against the use of the readkey function. The experiment used to test the resolution can easily be extended to test the wait statements in other software packages. This is also applicable to Turbo Pascal 5.0.

## REFERENCES

Brysbaert, M., Bovens, N., d'Ydewalle, G., & Van Calster, J. (1989). Turbo Pascal timing routines for the IBM microcomputer family. *Behavior Research Methods, Instruments, & Computers*, **21**, 73-83.

Graves, R., & Bradley, R. (1987). Millisecond interval timer and auditory reaction time programs for the IBM PC. *Behavior Research Methods, Instruments, & Computers*, **19**, 30-35.

Correspondence may be addressed to Marc Brysbaert, Department of Psychology, University of Leuven, B-3000 Leuven, Belgium.

*—Accepted by previous editor, Joseph B. Sidowski*

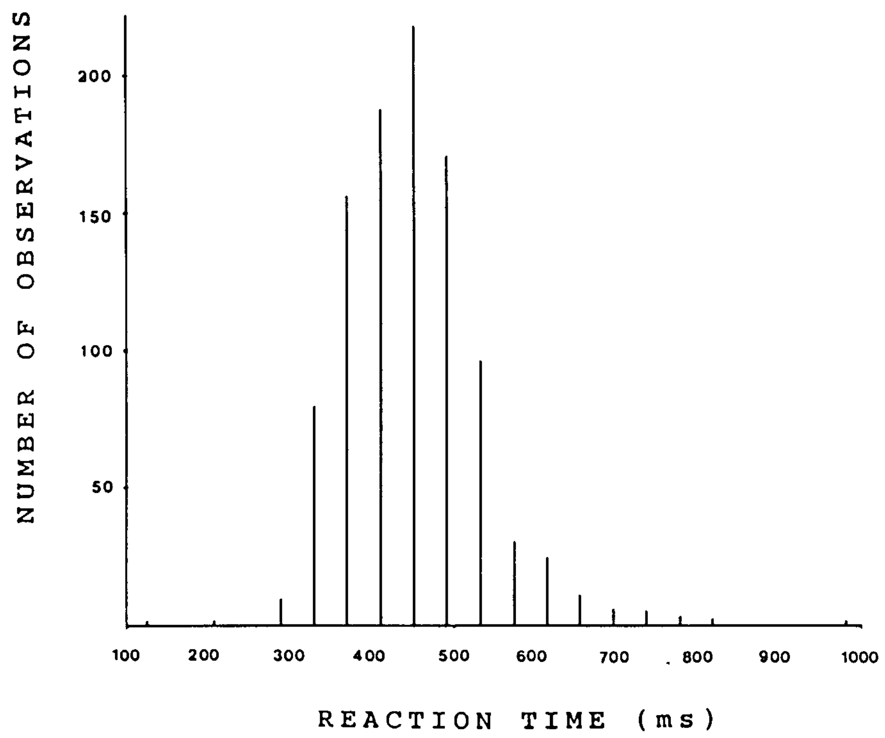

Figure 1. Distribution of reaction times during use of the readkey function. Data grouped in clusters of 4 msec.

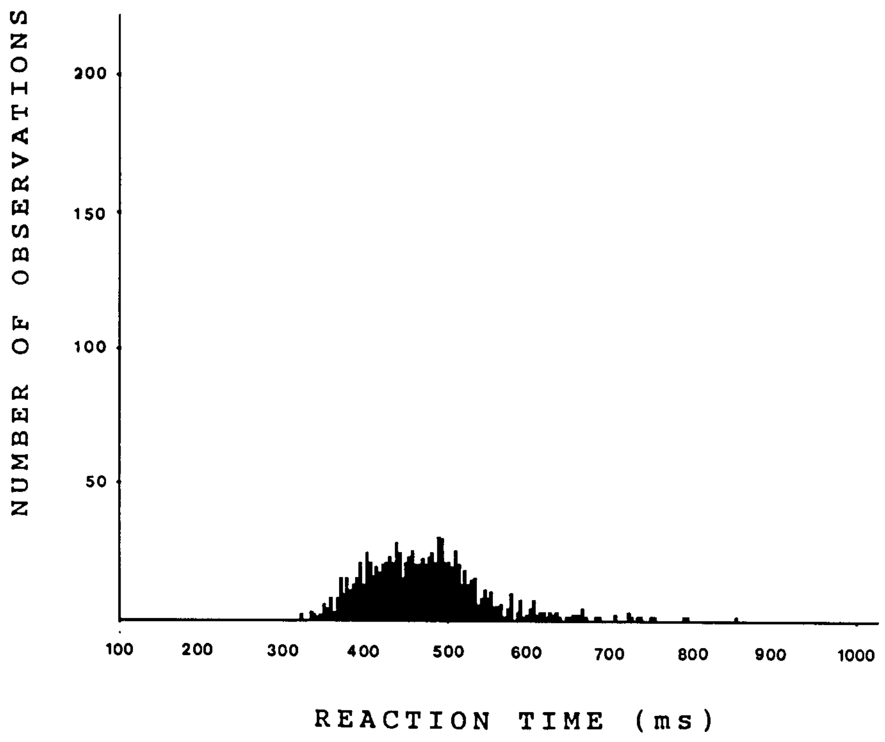

Figure 2. Distribution of reaction times during use of the keypressed function. Data grouped in clusters of 4 msec.

(Manuscript received December 9, 1988;
accepted for publication February 17, 1989.)