Journal of Neuroscience Methods, 28 (1989) 205–208
Elsevier

NSM 00952

Enhancing high-speed digitization of single-unit neuronal activity on a microcomputer using a hybrid software–hardware technique

J. Wayne Aldridge, Jonathan L. Walden and Sid Gilman

Department of Neurology, University of Michigan, Ann Arbor, MI (U.S.A.)

(Received 13 May 1988)
(Revised 8 December 1988)
(Accepted 10 December 1988)

Key words: Single-unit recording; Data acquisition; Digitization; Software algorithm; High-level language

A new data acquisition technique allows a microcomputer simultaneously to digitize spikes at high rates, analyze spike waveforms for computer-based spike separation and manage other control tasks. The technique has two key features: a software scheduling routine written in a high-level language and a hardware analog delay of neuronal signals using simple hardware external to the computer. The technique provides an alternative for real-time data acquisition and can be used on microcomputers without requiring interrupt processing and assembly language programming.

Introduction

The technique described in this paper enables a microcomputer and external signal processing hardware to digitize spike discharge activity at high rates. This technique frees the computer from continuously waiting for data and provides an additional mechanism for simultaneously controlling other experimental tasks in real-time. An important aspect of the technique is the capability to compute waveform parameters on-line for a computer-based spike separation procedure (Schmidt, 1984, Vibert and Costa, 1979), a more precise and flexible approach than external hardware spike discrimination (Schmidt, 1984).

Interrupt processing is one common method for data acquisition when an experimental control task must be interleaved in real-time. Although well-suited to this assignment, interrupt processing can be complex to prepare and maintain and may not be installed on all microcomputer systems. Interrupt processing requires a detailed consideration of all computing resources such as memory and disk management and, generally, an assembly language interface. Also, interrupt driven software is often not easily portable to different computers.

The new hybrid polling technique described here takes advantage of the normal properties of neuronal discharge activity. Neurons exhibit a wide range of firing rates, including rates as high as 1000 Hz, although sustained high firing rates are rarely observed. Most neurons, including those capable of reaching these high rates, exhibit bursts of spikes interspersed with periods of relative silence. In the globus pallidus, a structure known for high rates of neuronal activity (DeLong, 1971; Aldridge et al., 1980), 16% of units have a median rate of 100 Hz or more, however, less than 1% of the units reach a median rate of 200 Hz (Aldridge, unpublished observations). Even at a sustained rate of 200 Hz, a computer will be engaged by the digitization process only 20% of the time, assuming a spike duration of 1 millisecond. At low rates

Correspondence: J.W. Aldridge, Department of Neurology, University of Michigan, Neuroscience Laboratory Building, 1103 East Huron, Ann Arbor, MI 48104, U.S.A.

of firing the proportion of time needed to digitize spikes is miniscule. The polling method we have devised incorporates a new hybrid design that permits the computer to utilize effectively and easily the potentially large amount of processing time between neuronal spikes.

Methods and Results

Unit activity for this study was recorded from the striatum and pallidum of monkeys and cats surgically prepared for chronic single unit recording. The discharge of single units was recorded with standard electrophysiological techniques. Tungsten microelectrodes were manipulated to recording sites with a hydraulic microdrive attached to a recording chamber. Electrode signals were preamplified and monitored on an oscilloscope. A schematized plan of the analog signal processing steps is shown in Fig. 1.

The microcomputer (Compupro, CP/M 68K) had an analog-to-digital converter and a real-time clock. The external hardware consisted of an analog delay line, a voltage comparator and a retriggerable pulse generator (Fig. 1). The analog delay line (EG&G Reticon SC5106) coupled the neuronal signal to the computer's analog-to-digital converter with a delay of 5 ms. The voltage comparator, connected to the undelayed signal, was set to a level above the background noise and below the level of acceptable spikes. Every spike exceeding the comparator window triggered the pulse generator. The pulse duration was set to a length equivalent to the analog delay. In operation, spikes initiated a pulse at the same instant of time at which they entered the delay line. The computer, by polling the pulse generator, detected the presence of a spike in the delay line. When no spikes were detected, the program could initiate and execute other experimental control tasks.

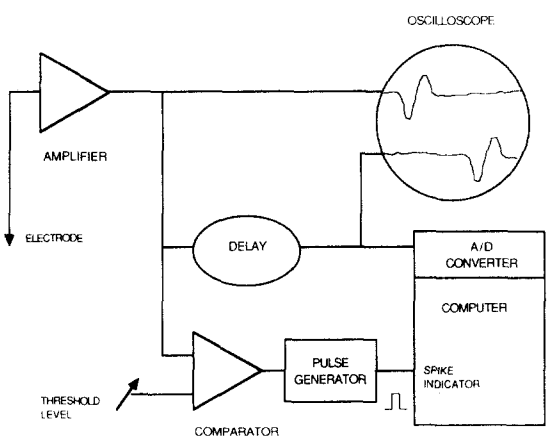

Fig. 1. Analog signal processing. The neuronal signal from the electrode was amplified and connected to the oscilloscope, audio monitor, delay line input and voltage comparator. The delayed analog signal was connected to the computer's analog-to-digital converter and the oscilloscope. The computer had a Motorola 68000 microprocessor running at a 10 MHz clock rate and utilized the CP/M 68K operating system. One megabyte of memory was available for the program and data storage.

The delay line served as a temporary storage location for the spike, allowing the computer time to interleave other experimental tasks. Whenever a spike was detected, the program immediately initiated a routine that digitized the spike from the delayed signal. The digitization routine read the analog data at regular intervals (10 μs) and stored the waveform in a memory buffer to be analyzed by a separate routine. A voltage window set just above the background noise and below the level of acceptable spikes defined the onset of a spike. Data collection continued until the level fell to zero after the second, positive peak in the spike waveform was passed or until a period 1 ms from the onset of the spike elapsed. No spikes were missed in high-frequency bursts as each spike retriggered the pulse generator and the digitization routine remained in effect until at least 1 ms after the pulse generator indicated that no spikes were present. The computer calculated the actual time of spike occurrence by subtracting the length of the analog delay from the time read on the block at the onset of the spike.

The critical software component was a scheduling module that supervised digitization and all control functions. The scheduling module (Fig. 2) consisted of a program loop that performed one of four prioritized operations on each iteration. First, the module polled the spike indicator (pulse generator) and, if a spike was present, executed the digitization routine. The second priority of the scheduling module was to look for a keyboard

```
/* SCHEDULING MODULE PSEUDOCODE */
LOOP:
    read the spike indicator;
    if there is a spike indicated
        execute spike digitization;
        goto LOOP;
    read the keyboard command port;
    if there is a keyboard command
        execute the keyboard command;
        goto LOOP;
    read the time;
    if there is a scheduled task AND it is time to execute the task
        execute the scheduled task;
        remove the task from list;
        goto LOOP;
    if there is a queued task
        execute the next queued task on the list;
        remove executed task from list;
        goto LOOP;
    goto LOOP;
```

Fig. 2. Scheduling module pseudocode. The scheduling module consisted of an endless loop that could be broken only by a "stop" command from the keyboard or a scheduled "stop-task" at a particular time. The loop performed only one action each time it was traversed. Precedence was given to spike digitization if the spike indicator signaled that a spike had entered the delay line. If there was no spike, a keyboard command was executed if one was found. If there was neither a spike nor a keyboard command, the module would execute a scheduled task if the time was appropriate or execute a queued task if one was on the list. Scheduled commands included stopping data collection at prearranged times and regular terminal screen updates to inform the user of the number of spikes collected and the elapsed time. Queued tasks were devoted to tasks such as measuring peaks of digitized spikes and managing buffers.

stop command and, if one was present, to execute the command and leave the loop. The third priority was to control time-scheduled experimental tasks. Scheduled tasks were maintained in a sorted list and executed at specific times read on the real-time clock. The 4th priority was to execute queued tasks, which were processed in the order they were placed on the list. All commands or control tasks were executed in less than 5 ms, the length of the analog delay. Scheduled and queued tasks were written as single functions and none took longer than 3 ms.

Scheduled and queued tasks were used for experimental control and on-line analysis of spike waveform parameters. Scheduled tasks including stopping at a preset time and regularly outputting to the terminal information on elapsed time and the number of spikes collected. Scheduled tasks could also include timed delivery of sensory stimuli and monitoring of behavioral events. The most important queued task was to compute descriptive waveform parameters from spikes buffered in memory. In our study, a point by point analysis was made to compute the amplitudes of the nega-

tive and positive peaks and the peak to peak durations (Vibert and Costa, 1979). Other parameters, such as risetime, could be determined as well.

Using the method described above, we have studied several hundred neurons in the basal ganglia of both monkeys and cats. Firing rates are high in the globus pallidus and in some cases the electrode records activity from two units simultaneously. Even in this situation the computer was capable of collecting and analyzing data. The comparator setting was not critical providing it was below the peak amplitude level of all spikes. If the comparator level was set too low, spurious electrode noise could initiate spike digitization. If this occurred, noise would be collected as spikes, and this could overload the memory buffers and disable data collection. Overloading was easily avoided by correctly setting the comparator levels outside the noise boundaries. We have never encountered units that were discharging too fast to be collected using this technique.

Discussion

Using a new combination of hardware and software, we have developed an alternative method for digitizing spike discharge activity and, simultaneously, managing other experimental control tasks with a microcomputer. The technique relies on the fact that most neuronal activity occurs sporadically in time. The time between spikes is used to execute other experimental tasks by delaying the arrival of the spike to the computer with hardware. The hardware is readily available, simple to use and requires no internal modifications of the computer. The scheduling algorithm is flexible and can be used to provide a general purpose software interface for timing and administering experiments in real-time.

An advantage of our method is the ability to implement it on a microcomputer that does not have interrupt processing. We used only a standard 'C' compiler for development and did not have to contend with the intricacies of the assembly language interface to the operating system. Another important advantage is the liberty to write control programs in a high-level language, and thereby avoid the tedium and complications of assembly language programming. The advantages of using a high level language in well defined, limited applications have been demonstrated in other settings (Simmons, 1985). Assembly language subroutines, although difficult to write, can greatly increase data acquisition speed (Kegel et al., 1985). There are some disadvantages to this technique. It relies on the time between spikes to execute control tasks. When neurons fire in sustained, extremely high rates, this technique is not applicable, however, most neuronal activity does not have this property. Another disadvantage is that control functions must be short enough to be completed within the delay time. A delay of 5 milliseconds is long enough to permit a microcomputer to complete even very complex control tasks, but it may not be sufficient for all applications.

Acknowledgements

This work was supported in part by NIH Grants NS 19613 and NS 07222 and by a grant from the United Cerebral Palsy Research and Education Foundation, Inc.

References

Aldridge, J.W., Anderson, R.J. and Murphy, J.T. (1980) Sensory-motor processing in the caudate nucleus and globus pallidus: A single unit study in behaving primates, Can. J. Physiol. Pharmacol., 58: 1192–1201.

DeLong, M.R. (1971) Activity of pallidal neurons during movement, J. Neurophysiol., 34: 414–427.

Kegel, D.R., Wolf, B.D., Sheridan, R.E. and Lester, H.A. (1985) Software for electrophysiological experiments with a personal computer, J. Neurosci. Methods 12: 317–330.

Simmons, P.J. (1985) Signal averaging by microcomputer using a program written in a high level language, J. Neurosci. Methods, 12: 235–240.

Schmidt, E.M. (1984) Instruments for sorting neuroelectric data: a review, J. Neurosci. Methods, 12: 1–24.

Schmidt, E.M. (1984) Computer separation of multi-unit neuroelectric data: a review, J. Neurosci. Methods, 12: 95–111.