SESSION III PRESIDENTIAL ADDRESS

RICHARD MILLWARD, *Brown University, President*

Microprocessors: Their impact on real-time computing in psychology

PETER G. POLSON

*Computer Laboratory for Instruction in Psychological Research,
University of Colorado, Boulder, Colorado 80309*

This paper attempts to evaluate the impact of microprocessors and other products using large-scale integrated circuit technologies with a set of examples of past, current, and proposed computer applications in psychological research. I first discuss a number of developments in real-time computing that have taken place during the last 10 years. The major part of this paper describes future developments in the Computer Laboratory for Instruction in Psychological Research. We are developing an intelligent high-speed communication network that will enable computers and terminals to communicate with each other. In addition, we are developing a microprocessor-based real-time computer system. The network, the real-time system, and the general assumption underlying the development of the next generation of CLIPR facilities are described.

This paper presents my opinions on the future impact that microprocessors, and other large-scale integration (LSI) technology, will have on automation in psychology laboratories. I begin with a discussion of views on the functional requirements for real-time computer systems for psychological research, and then describe an early PDP-8 based system and the system at Colorado based on the Xerox Sigma 3. These two systems are examples of the end points on a continuum defined by a laboratory computer for a single individual vs. a central, shared facility for a whole department.

The major portions of this paper describe a new multiprocessor system that we are developing in the Computer Laboratory for Instruction in Psychological Research (CLIPR). This new system will integrate a wide variety of microprocessor-based applications ranging from special-purpose controllers and intelligent terminals to general-purpose real-time computing systems. The separate processors will be linked over a microprocessor-controlled communica-

tion network. The communication network has been named CLIPRNET; the microcomputer is called CLIPR-1.

Although the opinions expressed in this paper are mine, the substantive work presented represents the collaborative efforts of everyone associated with CLIPR. Many of the ideas concerning the wide variety of applications of microprocessors were developed by the CLIPR staff and in a grant application by Daniel Bailey, Director of CLIPR. CLIPRNET and the detailed design of CLIPR-1 were developed by the technical staff: Tom Christian, Terry Spear, Joann Waite, and Ron Leatherman.

SYSTEM REQUIREMENTS FOR LABORATORY AUTOMATION

Automation of a psychological research laboratory is a very demanding application area for the current state of the art in real-time computing. Input and output data rates can range from a few bits per second (bps) to over 100,000 bps across classes of experimental paradigms. A constant requirement is that the system must be able to respond very rapidly to real-time events such as the end of a time interval or a keypress in order to control durations of stimuli and record response latencies accurately. In addition, a maximally useful system must have the software tools to enable a researcher to quickly develop new applications programs. Research paradigms and problems change rapidly in many areas of psychology, and thus investigators are unwilling to wait 6 months to 1 year for the development of a single program.

This paper is an expanded version of the Presidential Address that I presented to the Seventh Annual Meeting of the National Conference on the Use of On-Line Computers in Psychology. The work described here represents the joint efforts of all members of CLIPR. I especially wish to thank Terry Spear and Tom Christian for their contributions to my understanding of the work described here and for their critical readings of earlier versions of the manuscript. I am solely responsible for the opinions and other errors that may be found in this paper. CLIPR was originally supported by NSF Grant GJ 453 to D. E. Bailey and is currently supported in part by funds from the University of Colorado. Requests for reprints should be sent to Peter G. Polson, Department of Psychology, University of Colorado, Boulder, Colorado 80309.

A maximally useful, laboratory computer system should have an operating system that supports concurrent execution of real-time tasks in foreground, with program development and data processing in a background partition. Many readers' initial reaction will be that it would be nice to have such a system, but that it is not necessary. I disagree for several reasons. First, many research paradigms require the concurrent execution of several real-time tasks, for instance, input-output to disk storage overlapped with stimulus generation and response recording. Furthermore, some experimental paradigms involve the simultaneous control and monitoring of two or more streams of real-time events; such an experiment is automated using several real-time tasks that are executed concurrently. Second, a computer system should enable one to make the laboratory more productive, and thus it is important to be able to run several subjects at one time. Third, program development and data analyses should be both convenient and rapid, or investigators will prefer to use more traditional types of equipment to carry out experiments.

DEVELOPMENTS IN REAL-TIME COMPUTING SYSTEMS

Comprehensive reviews of past developments and applications of real-time computers have been published by Aaronson, Grupsmith, and Aaronson (1976), Cassady, Patterson, and Roemer (1976), and Sidowsky (1972, 1976). I will confine my presentation to descriptions of the earlier Digital Equipment Corporation PDP-8 systems and the Xerox Sigma 3 based systems. My purpose is to illustrate the dramatic reductions in cost made possible by the rapidly developing microelectronic and systems design technologies.

The DEC PDP-8 was the first powerful laboratory computer that was widely used in psychological research. Early versions of this system were typically equipped with 4K words of memory, a Model 33 ASR TTY for both input and output of programs and data, and some type of primitive display. This very limited system could support an assembler that produced absolute code and an editor. Real-time clocks and other types of specialized peripherals for experimental work had to be designed and built by the user. The total cost of this primitive system was between \$22,000 and \$24,000.

Today, a functionally equivalent system based on an INTEL 8080A microprocessor is available from Heathkit for under \$2,000. In addition, the Heathkit H-8 has manufacturer-supplied interfaces, an inexpensive character-oriented display, and many other amenities. An example of a microprocessor-based system costing as much as the PDP-8 is a DECLAB 11/03 with 28K words of memory, dual floppy disks, very sophisticated interfaces for both digital and analog

signals, a CRT display, and a very competent disk-based real-time operating system with FORTRAN and a complete library of specialized FORTRAN-callable subroutines. Programs that would take several weeks to develop for the PDP-8 can be running in a few hours on the 11/03. The 11/03 is a far more functional system.

The PDP-8 and the DEC 11/03 are exemplars of one possible strategy for laboratory automation: specialized small systems in individual laboratories. This is the dominant strategy today. CLIPR represents a very different strategy: a large computer system shared by all laboratories in a department, as well as supporting instructional and data processing activities.

The CLIPR laboratory has been described in detail elsewhere (Bailey, 1972, Note 1; Bailey & Polson, 1975). The laboratory is based on a large Xerox Sigma 3 CPU with 64K 16-bit words of memory, over 100M bytes of disk storage, a line printer and card reader, a large amount of communications equipment that interfaces various types of terminals to the Sigma 3, and numerous specialized interfaces for laboratory equipment. The CLIPR/RBM operating system supports the concurrent execution of up to eight independent real-time programs in foreground and a background subsystem that provides all of the facilities of a small, but powerful, single-stream batch system. The overall system is organized such that the background subsystem cannot interfere with programs executing in the foreground. Thus, assemblies, FORTRAN compilations, debugging, execution of real-time functions, remote batch communication to the main computer center, and the execution of data processing programs take place while the system is concurrently executing several real-time programs.

The primary objection to the one central facility strategy is its cost in terms of both time and money. Uttal (1972) suggests that the resources invested in a central facility can be better spent on developing separate systems for individual laboratories. The counter arguments are that a central facility enables users to support very high-quality instructional programs without interfering with ongoing research, to provide specialized data processing facilities, to provide special-purpose software development tools for application programs, and to support expensive high-performance peripherals like large disk storage units, high-speed line printers, and tape drives. Recall that the major argument in support of developing time-shared systems is that they give an individual convenient access to a much more powerful system than he can support himself.

FUTURE DEVELOPMENTS IN CLIPR

During this past year, we have started the planning, design, and hardware and software developments

that will evolve into a new CLIPR system to eventually replace the Sigma 3. We are rapidly outgrowing the Sigma 3; the major restriction is the lack of memory. Mapping hardware and the software necessary to support it were never developed for the Sigma 3. In addition, maintenance costs may rise rapidly in the next few years. The Sigma 3 has grown old gracefully, but it will not last forever.

We are not going to replace the Sigma 3 with a single CPU. Even a DEC 11/70 cannot satisfy the response time requirements of an ever growing number of real-time application programs. In fact, the 11/70 comes nowhere near the Sigma 3 in its ability to provide very quick response times (less than 5 msec) for a large number of real-time tasks. However, the 11/70 can support a much larger number of tasks than the Sigma 3 and still provide response times in the tens of milliseconds to 500 msec.

There seems to be a rapidly developing consensus that the central system should not be required to support real-time tasks with very demanding response time requirements. These tasks should be performed by dedicated satellite processors that transmit and receive information from the central system and are programmed using software development tools on the central system. Obviously, computer networks are not a new idea (the ARPANET, DECNET, SUMEX, etc.), and networks have been advocated as solutions to various kinds of computing problems, typically for sharing of specialized resources (Sedelow, 1976).

We plan to develop a hierarchy of microprocessor-based real-time systems that are interconnected to each other and a central site through a very high-performance communication network. The central site will be the Sigma 3 during the first few years of the development of the overall system. The network will enable microprocessor-based systems in individual laboratories to utilize the facilities of the central site, for instance, the tape drive for archival storage of data, the line printer for program listings, specialized software development tools, etc.

All new real-time applications will be developed on microprocessor-based systems that are connected to the network. In addition, we plan to move all current real-time application programs from the Sigma 3 to microprocessor-based systems. All of the new software necessary to program the microprocessors and to permit remote use of the facilities of the Sigma 3 will be written using modern techniques for the generation of system-independent code. Then it will be possible for us to replace the Sigma 3 with minimal interruption of the ongoing activities of the separate laboratories.

CLIPRNET¹

The communication network that will link the

separate laboratories with the Sigma 3 and each other is CLIPRNET. Logically, CLIPRNET will be organized as a star with the net control processor at the central node.² A schematic diagram of the net is shown in Figure 1. Each logical limb of the network consists of a full-duplex path between the control processor and a terminal node. Our current design goal is a data rate of 1,000,000 bps in each direction; one-quarter of this rate would provide us with more than adequate performance from the network. The terminal node on each limb is a network interface module (NIM). There will be one NIM and, therefore, one logical limb for each device connected to the network. A device can be a Teletype terminal or some other type of dumb terminal, an intelligent terminal, a modem for communication with a remote site, or another real-time computer system. Since CLIPRNET is an intelligent network, almost any type of device can communicate over it.

A device connected at a node will request that the control node connect its node to another node of the network. This logical path, once established, will have the following characteristics: *data transparency*, functional equivalence to a hard-wired *point-to-point connection*, and *error-free transmission*. The transmission process will be data transparent. This means that there are no restrictions on the length and content of the message, in other words, no required control characters and no restrictions on bit patterns contained to the message. Thus, two devices communicating over the net do not have to reformat the information they exchange in order to satisfy restrictions imposed by the net.

Restricting most interconnections to be function-

ally equivalent to a hard-wired point-to-point connection simplifies the logic in any device that will use the net.² No provisions have to be made for the possibility of multiple inputs from different sources. This restriction will be relaxed for one terminal node of the network, the node with the Sigma 3 connected to it. Any number of devices will be able to communicate with the Sigma 3 concurrently. Obviously, this is necessary. The primary function of the network is to enable systems at other terminal nodes to use the facilities of the Sigma 3. The software for network communication in the Sigma 3 will be far more comprehensive than that at any other node.

Finally, a device connected to the network can assume that the interconnection is error-free, which dramatically simplifies the use of the network. Anyone who has written a tape or disk driver or any type of external device driver knows that these very complex programs would be trivial if one could assume that the hardware functioned in an error-free manner.

Transmission from one node to another will be controlled by the NIMs. The NIMs incorporate much of the net intelligence, and they give the communication process the characteristics that were outlined earlier. A schematic diagram of the NIM is shown in Figure 2. The NIMs incorporate the first three levels of the network hardware and software. Level 0 is the interface to the physical data path, a two-conductor coaxial cable and the modem. Level 1 is the logic that implements and controls the low-level line protocol. We will use the ANSI Standard Advanced Data Link Communications (ADLC) protocol. This communications protocol gives complete data transparency and a very high degree of error checking with low overhead. Level 2 of the NIM is the message transfer protocol. Transmission and reception of messages, error-handling procedures, the exchange of acknowledgment between sending and receiving terminal nodes are all part of the message transfer protocol.

An error-free message will be in the buffer of the receiving NIM at the completion of the message transfer protocol. This message will then be transmitted by EIA RS-232 serial interfaces to the device connected to the NIM. The interpretation of the message is done by the receiving device. It could be a character for a TTY terminal, a binary data record, or an absolute core image being down-line loaded into a microprocessor.

Our current goal is for a NIM to cost between \$200 and \$500. The logic for the ADLC protocol is on a single LSI chip. The message-level protocol will be implemented in a Motorola 6800 microprocessor. Physically, CLIPRNET will be a single two-conductor coaxial cable with NIMs inductively coupled to the cable. All the NIMs in the network

Figure 2. A schematic diagram of the network interface module (NIM).

will in effect timeshare this single transmission medium. The logic for such networks was developed at Xerox's Palo Alto Research Center in conjunction with the ETHERNET (Metcalfe & Boggs, 1976; Schwartz, 1977).

In summary, CLIPRNET will enable devices connected to its terminal nodes to communicate easily with each other in real time by isolating the device from the complexities of the actual communication process, while at the same time imposing no constraints on the messages being transmitted between nodes.

The low-level line protocol implemented by the ADLC chip and the message-level protocol control program in the Motorola 6800 would be equivalent to a 6,000- to 8,000-word subsystem in a computer that was interfaced directly to the net. The complete communications process is very complex and could absorb up to 15% to 30% of the machine's CPU cycles. Furthermore, until very recently, the hardware necessary to interface a system to a high-speed communication channel has been very expensive. The design of CLIPRNET is a dramatic example of our rapidly increasing understanding of communication networks and the incredible advances in LSI technology. Without both, CLIPRNET would not be financially or technically feasible.

REMOTE FILE SYSTEM AND OTHER FACILITIES OF THE CENTRAL SITE

The next level of the communication process will be subsystems in the various computers that are connected to the network. The most complex versions will be in the Sigma 3 to permit devices at all other terminal nodes in the network to utilize the facilities of the central site. The remote file system will permit a user to transmit or receive information from a peripheral connected to the Sigma 3. The remote file system will implement the most general definition of a file so as to include in a single structure various types of disk files, tape files, input from a card reader, output to a line printer, and all other types of input/output. Operations implemented will include: read/write a record from/to a file, open or close a file, and other necessary file operations where the definitions of all of these operations have been properly generalized. The UNIX operating system developed at Bell Laboratories is based on the generalized concept of a file (Ritchie & Thompson, 1974). Many of the operations to be implemented by the remote file system are available in DECNET.

DEVICES INTERFACE TO THE NETWORK

CLIPRNET is an intelligent network, thus a very wide range of devices can be interfaced to the network. Three classes of devices will be connected to the network: dumb terminals, intelligent terminals, and other computer systems.

"Dumb terminal" refers to various models of Teletypes, DEC writers, Lear Sigler ADM-3s, etc. A terminal node connection to a dumb terminal will have a default connection to a control program, probably resident on the Sigma 3, that will provide for the processing of requests for services entered into the terminal, in other words, edit a file on the Sigma 3, make a connection to another node, etc.

INTELLIGENT TERMINALS

The term "intelligent terminal" refers to a wide range of microprocessor-based devices that perform various types of real-time functions and are capable of communicating with a computer. Dan Bailey has developed a design for a family of such systems in a series of working papers and a proposal to NSF. His objective is to develop terminals that utilize a university's central computer facilities for program development and data analysis. The terminal would actually control an experiment, performing all real-time control and data collection functions. There would be a family of terminals, with each member being specialized for a different class of research paradigms. The overall objective of this development

effort would be to enable an investigator to have high-quality real-time control and data logging facilities without having to develop and support a separate computer system. Potts (1976) described a very simple version of such a terminal that he used with the timesharing system at Dartmouth. Durrett (1974) has described a scheme for programming experiments that can easily be modified to utilize an intelligent terminal.

In addition, these small, microprocessor-based systems can be used with any central computer to perform various real-time functions while dramatically simplifying the software on the central machine. Response recording, latency measurements, control of event durations, and other real-time functions are difficult to perform with millisecond or better accuracy on a multiprogrammed system (Christian & Polson, 1975; Kaplan, 1977). Thus, a general strategy that we are using in the development of our new systems is to move most real-time functions out of the main CPU into microprocessor-based real-time device controllers or intelligent terminals.

An early example of this general strategy was the PDP-8/I based SCAT system developed by Grason-Stadler. Time and response recording were done in hardware connected to the CPU through a multiplexer. Moving real-time functions into special-purpose hardware enabled the developers of SCAT to support multiple experiments on a small system executing interpreted code.

If the general strategy of moving real-time functions out of the CPU has been known for several years, why has this strategy rarely been utilized? The primary reasons have been the difficulty and cost of designing and building the real-time control units and interfacing them to the CPU. The appearance of the inexpensive, \$200 to \$400, single-board microprocessor with erasable, programmable, read-only memory (EPROMs) has dramatically simplified the development of such hardware. One writes a program to perform these functions for the microprocessor; very complex hard-wired logic often is reduced to a 100- to 200-line program that takes 2 or 3 days to write and debug. The debugged program is burned into an EPROM. The experimental apparatus can be interfaced to the microprocessor through the parallel I/O ports. Control information from the main CPU and data to the main CPU are transmitted through an EIA RS-232 serial interface which is part of the single-board system.

CLIPR-1

CLIPRNET has been designed so that it will be easy to connect any type of computer system to it. However, we do not want to support a large number of different computer systems in the Psychology

Department. Attempting to maintain several different systems would seriously dilute our staff resources. Members of the professional staff would have to spend a large amount of time familiarizing themselves with different systems in order to assist in the development of individual laboratories.

The far better alternative is to develop a standard system for all laboratories in the Department. The CLIPR professionals could then write the necessary specialized systems software, software tools for applications program development, and the subsystems to permit the computers in separate laboratories in the use of the facilities in the central site.

CLIPR-1 is a small real-time computer system based on a Texas Instrument's TMS-9900 microprocessor interfaced to the S-100 bus. Why did CLIPR choose the TI microprocessor, and why did we interface it to an 8-bit bus?

The TI 9900 microprocessor is a 16-bit CPU with an extensive instruction set that includes hardware integer multiply-divide. The instruction set is comparable in power and similar in style to the instruction set of the DEC PDP-11 machines. The CPU has memory mapped I/O ala the 11 and, in addition, bit- and byte-oriented I/O through the CPU registers. Most important of all, we feel that the TI is the first 16-bit machine with multiprogramming capabilities that are equal to those of the Sigma 3. The hardware will perform a complete context change in 10 microsec. The single-chip version can address up to 64K bytes of memory, and has a register-to-register add time of 4 microsec. The comparable figure for the DEC PDP-11/03 is 3.5 microsec. TI is in the process of developing a series of upward-compatible systems; the top of the line, single-board CPU is equivalent to the 11/34, in other words, with memory mapping and floating point hardware.

We have interfaced the TI 9900 to the S-100 bus. The S-100 bus was originally developed for the INTEL 8080-based microprocessor-based systems for the hobbyist market. The S-100 is laid out in a manner that is similar to the UNIBUS in the 11 series systems; the major difference is that it is an 8-bit bus. The S-100 bus has become the de facto standard in the hobbyist market. An incredible amount of hardware is made to interface to the S-100 bus, and this market is extremely competitive in terms of both price and performance.

As an illustration, the memory board that has been selected for CLIPR-1 has 8K bytes of 250-nsec static RAM. The board is assembled, tested, guaranteed for 1 year, and satisfies military specifications for ruggedness. Its cost is \$217. Single boards containing 64K bytes of 450-nsec dynamic memory are available for about \$1,500.

Additional hardware, including printers, floppy and hard disk units and controllers, serial and paral-

lel interfaces memory mapping units, displays of various types, and so on, has been developed for S-100 bus systems. Products developed for the hobbyist markets are cheaper and some of the more unusual products may have useful applications in the psychology laboratory. Most home computing systems emphasize various types of interactive real-time programs. Do not knock the technological requirements of complex interactive games like the more elaborate versions of STAR TREK.

There is one fairly serious drawback to the S-100 bus; it is a nonstandard standard. Approximately 50 of the 100 lines on the bus have standard functions. Many of the lines were unused in the original 8080-based systems, and different manufacturers used these lines for special functions for their boards. All I can say is, beware.

SYSTEMS SOFTWARE FOR CLIPR-1

The software developed for CLIPR-1 will include a resident real-time multitasking executive with a generalized file system, network control routines, device drivers, and the software development tools for the TI 9900 microprocessor (i.e., assembler, link editor, and compilers). In general, we plan to have our software development tools run on the Sigma 3. Large disk files, high-speed line printer, on-line editors, etc., make the Sigma 3 a much more congenial environment for program development.

We currently have a cross-assembler written in FORTRAN IV that runs on the University's CDC 6400. The assembler was programmed by Texas Instruments and is designed to run on a large machine with at least 32-bit words. We have developed a linking loader for the TI 9900 that runs on the Sigma 3. Relocatable object modules from the assembler are transmitted to the Sigma 3 from the CDC 6400 and loaded on the Sigma 3. The executable program is then loaded into CLIPR-1. A PASCAL cross-compiler that will run on the Sigma 3 and generate relocatable TI-9900 code is currently under development.

The resident real-time executive for the TI 9900 will provide for task priorities, hardware interrupt handling, measurement and control of time intervals, interrupt-driven input-output, and intertask communication. This executive will enable a single applications program to run several subjects concurrently with asynchronous execution of the experimental procedure. The design of the resident kernel of this real-time system will be taken from the INTEL Corporation's RMX/80 real-time multitasking executive for their SBC 80 and System 80 microcomputers (Burgett & O'Neil, 1977). Specialized system programs and application programs will be developed within the framework defined by the kernel. Real-

time functions, such as recording a latency, presenting a stimulus on a CRT display, output of a data record, etc., will be performed by routines that are part of the kernel or by service routines that are part of a specialized operating system for a given class of experiments. Applications programs will perform input-output and real-time functions by calls to subroutines that are part of the operating system.

USER INTERFACE

The ultimate success of CLIPR-1 and CLIPRNET will depend on the software tools available for the user to develop application programs for the Sigma 3 or CLIPR-1. Many recent discussions of the state of the art of real-time computing in psychology have concluded that a vast majority of users will program such systems in a higher level language (Bailey & Polson, 1975; Castellan, 1975; Wood, Sette, & Weiss, 1975). Very few researchers would or could deal directly with the complexity of a multiprocessor system like the Sigma 3/CLIPR-1 combination at the assembly language level.

We can roughly partition higher level programming languages into programming aids and problem solving aids (Sammet, 1969). The primary function of a programming aid is to assist in the mechanical task of generating machine code; a problem solving aid provides a conceptual structure that facilitates the development of applications programs (Polson, 1973). Most real-time applications programming is done using FORTRAN or BASIC as a programming aid (Castellan, 1975; Donchin & Heffley, 1975; Fitzhugh & Glaser, 1975; Bailey, Note 1). FORTRAN or BASIC is used to generate linkages to library routines that perform real-time functions and input-output and to write code that performs the calculations necessary to control the experiment.

A problem solving aid is a highly specialized programming language that is based on an abstract model of the class of problems to be solved using a computer (Millenson, 1971; Sammet, 1969). Such a language permits the user to remain ignorant of technical details of the system while writing the program in a language that facilitates the statement of the experiment in a form suitable for a computer. Examples of such languages for the control of psychological experiments include SCAT (Polson & Campbell, 1972; Stadler, 1969), ACT (Millenson, 1970, 1971), SKED (Snapper, 1974), and DTES (Spear, Overgard, & Christian, 1975).

The special-purpose languages developed at Colorado have fulfilled their objectives. Users with relatively little technical background can rapidly program experiments that are within the scope of a given special-purpose language. Bailey and Polson (1975) point out that the limited scope of a given language

is a very serious drawback. Such languages are incredibly expensive to develop. Writing a compiler, operating system and subroutine library, and other support subsystems for stimulus preparation and data analysis and then documenting the complete system can involve 2 to 5 man-years or more of effort. Can these costs be justified for the limited clientele served by a given system?

The other approach of extending BASIC or FORTRAN with a large number of assembly language subroutines turns out to be an unsatisfactory solution for other reasons. The correct use of such subroutines often requires detailed knowledge of the operating system, the file structure of the system, and of the hardware in a laboratory controlled by the system. Thus, users often have to master a large amount of technical detail concerning a given system before they can write applications programs. However, there are more serious problems with the tactic of generating a real-time programming language by combining FORTRAN or BASIC with a large number of subroutines.

The resulting combination does not define a coherent programming language. Consider FORTRAN combined with a collection for FORTRAN-callable subroutines that provide relatively direct access to the various components of the operating system, including real-time functions. This "language" is defined by a mixture of the properties of FORTRAN, the operating system, and the various real-time functions. This mixture is obviously very difficult to master and does not provide the user with a coherent structure within which to define a solution to his or her particular application.

Thus, we are faced with a dilemma. Programming aids like FORTRAN augmented with various service routines are very badly human engineered programming languages. Successful problem solving aids are much more adequate software development tools, but such special-purpose languages have unacceptable cost-benefit ratios because of their limited scope.

Future attempts to solve this problem will involve the development of a collection of well-designed service routines. We think that experimental paradigms can be analyzed into the meaningful elements that occur in various paradigms. The service routines will then be designed to implement these atomic elements. In addition, much attention will be given to the various calling sequences so that there are regularities in the structure of calling sequences across different service routines.

CONCLUSIONS

In summary, how will microprocessors and, more generally, LSI technology impact the use of real-time

computers in psychology? The first and most obvious way is cost. The continuing advances in large-scale integration are leading to large cost reductions for all types of computing equipment, not just microprocessors. Newly introduced systems like the DEC PDP-11/60, which makes extensive use of microcode stored in LSI memory, offer dramatic improvements in performance for systems costing in the \$50,000 to \$100,000 range. The hobbyist market is also having a very powerful influence on cost. This is a large and highly competitive market.

A second important impact of this technology is that it makes possible the development of various types of multiprocessor systems like CLIPRNET and special-purpose real-time controllers interfaced to a central system. I feel that earlier proposals for various kinds of networks were premature for both cost and technical reasons. Before 1975 or 1976, the cost of the computers at the individual nodes and of the high-speed communications hardware were too high to make a network a feasible solution for a psychology department. Furthermore, the communication process would have absorbed 15% to 30% of the CPU cycles in each node of the net. The development of single LSI chips that implement the low-level communications protocol solves both cost and technical problems. We are going to see more high-level functions incorporated into single chips, which will further reduce the cost and complexity of very sophisticated systems.

The development of specialized powerful intelligent terminals will be a third benefit of this technology. Many experimental paradigms in psychology are defined by variations in stimulus materials and time parameters within a structure specified by a very limited number of real-time operations. Stimulus materials and other information for a particular experimenter can be prepared on a central computer using standard text editors. A relatively simple program can then be used to transmit this material to the terminal that actually performs the experiment. The terminal transmits data back to the main computer. The two primary advantages of such terminals are low cost and a simplified user interface. Programs can be written that run on the central computer to facilitate the generation of materials for the intelligent terminal. The user is relieved of most of the problems involved in the programming and maintenance of a complete system. The current LSI technology makes it feasible to develop many types of specialized intelligent terminals.

What is the most important pitfall of this new technology? The user interface. At this year's meeting of the National Conference, I heard papers describing systems that had fewer and/or more primitive software development tools than those available on DEC PDP-8s in the late 1960s. It was sort of an unpleasant

déjà vu experience to hear people describe how they struggled to develop a system that was in most respects equivalent to the system that I developed in 1966-1967. The only real difference is that these new systems are cheaper by about a factor of 10, so that more people can return to the dark ages.

Thus, I hope that a major benefit of microprocessor-based systems will be to force us to focus on the real problem, the user interface.

NOTES

1. The following material is a provisional description of the design of CLIPRNET.

2. This aspect of the design is still under consideration.

REFERENCE NOTE

1. Bailey, D. E. *CLIPR user's manual*. (CLIPR Publication Series No. 10). Boulder, Colo: Computer Laboratory for Instruction in Psychological Research, Department of Psychology, University of Colorado, September 1975.

REFERENCES

- AARONSON, D., GRUPSMITH, E., & AARONSON, M. The impact of computers on cognitive psychology. *Behavior Research Methods & Instrumentation*, 1976, 8, 129-138.
- BAILEY, D. E. The computer laboratory for instruction in psychological research: CLIPR. *Behavior Research Methods & Instrumentation*, 1972, 4, 95-96.
- BAILEY, D. E., & POLSON, P. G. Real-time computing at the University of Colorado. *American Psychologist*, 1975, 30, 212-218.
- BURGETT, K., & O'NEIL, E. F. An integral real-time executive for microcomputers. *Computer Design*, 1977, 16, 77-82.
- CASSADY, M., PATTERSON, M. M., & ROEMER, R. The use of on-line computers in psychobiology. *Behavior Research Methods & Instrumentation*, 1976, 8, 153-156.
- CASTELLAN, N. J. The modern minicomputer in laboratory automation. *American Psychologist*, 1975, 30, 205-211.
- CHRISTIAN, T. W., & POLSON, P. G. The precision of latency measures on real-time computer systems. *Behavior Research Methods & Instrumentation*, 1975, 7, 173-178.
- DONCHIN, E., & HEFFLEY, E. Minicomputers in the signal-averaging laboratory. *American Psychologist*, 1975, 30, 299-312.
- DURRETT, H. J., JR. A multiprogram approach to real-time experimentation. *Behavior Research Methods & Instrumentation*, 1974, 6, 259-262.
- FITZHUGH, R. J., & GLASER, R. A general-purpose computer system for a research and development center. *American Psychologist*, 1975, 30, 219-225.
- KAPLAN, H. L. Clock-driven FORTRAN task scheduling in a multiprogramming environment. *Behavior Research Methods & Instrumentation*, 1977, 9, 176-183.
- METCALFE, R., & BOGGS, D. ETHERNET: Distributed packet-switching for local networks. *Communications of the Association for Computing Machinery*, 1976, 19, 395-404.
- MILLENSON, J. R. Language and list structure of a compiler for experimental control. *Computer Journal*, 1970, 13, 340-343.
- MILLENSON, J. R. A programming language for on-line control of psychological experiments. *Behavioral Science*, 1971, 16, 248-256.
- POLSON, P. G. SCAT: Design criteria and software. *Behavior Research Methods & Instrumentation*, 1973, 5, 241-244.
- POLSON, P. G., & CAMPBELL, G. D. EXTENDED SCAT. *Behavioral Science*, 1972, 17, 558-565.

- POTTS, G. R. Use of a campus-wide timesharing computer to run reaction time experiments. *Behavior Research Methods & Instrumentation*, 1976, **8**, 179-181.
- RITCHIE, D., & THOMPSON, K. The UNIX time-sharing system. *Communications of Association for Computing Machinery*, 1974, **17**, 398-414.
- SAMMET, J. E. *Programming languages: History and fundamentals*. Englewood Cliffs, N.J: Prentice-Hall, 1969.
- SCHWARTZ, M. *Computer-communication network design and analysis*. Englewood Cliffs, N.J: Prentice-Hall, 1977.
- SEDELOW, W. A. Some implications of computer networks for psychology. *Behavior Research Methods & Instrumentation*, 1976, **8**, 218-224.
- SIDOWSKI, J. B. Various uses of minicomputers in psychology. *Behavior Research Methods & Instrumentation*, 1972, **4**, 43-50.
- SIDOWSKI, J. B. On-line instrumentation in psychology: Dildo or the real thing? *Behavior Research Methods & Instrumentation*, 1976, **8**, 52-56.
- SNAPPER, A. G. Alternative versions of SKED: Current systems and future plans. *Behavior Research Methods & Instrumentation*, 1974, **6**, 171-173.
- SPEAR, T. L., OVERGARD, D., & CHRISTIAN, T. W. The CLIPR display terminal experiment system. *Behavior Research Methods & Instrumentation*, 1975, **7**, 107-112.
- STADLER, S. J. On the varieties of computer experience. *Behavior Research Methods & Instrumentation*, 1969, **1**, 267-269.
- UTTAL, W. R. Misuse, abuse, overuse, and unuse of on-line computer facilities by psychologists. *Behavior Research Methods & Instrumentation*, 1972, **4**, 55-60.
- WOOD, R. W., SETTE, W. F., & WEISS, B. Interfacing the experimenter to the computer: Languages for psychologists. *American Psychologist*, 1975, **30**, 230-238.