MICROCOMPUTERS IN PSYCHOLOGICAL EXPERIMENTATION, HEADTURN: A CASE STUDY

by

Brian Pidcock

B.A.Sc., The University of British Columbia, 1971

B.Sc., The University of British Columbia, 1980

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

(Department of Computer Science)

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

September 1993

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

(Signature)

Department of Computer Science

The University of British Columbia
Vancouver, Canada

Date October 10, 1993

DE-6 (2/88)

Abstract

Mass market microcomputers are often used in conducting psychology experiments. Despite steadily falling prices and widespread use of these microcomputers in many areas, there are significant obstacles to their universal use in psychological experimentation. The specialized needs of experimentation are often poorly served by mass market systems, yet the small size of the psychology market precludes the development of low cost, specialized hardware and software.

The "Headturn" technique, an experimental protocol for studying sound and speech recognition in infants, was taken as an example of an experimental procedure that has evolved as the computer tools have evolved. This evolution is traced and the current state of the "Headturn" technique is documented and analyzed. The next step in the evolution of the "Headturn" technique has been designed and implemented to take advantage of current mass market components, capitalizing on the lessons learned using earlier versions of the system.

Timing is one example where the needs of the experimenter are not readily met by mass market components. Different options for improved timing in the "Headturn" procedure were explored and implemented. Timing limits relative to experimental needs are discussed.

Table of Contents

List of Tables

List of Figures

Acknowledgements

Dr. Kellogg S. Booth (Department of Computer Science) and Dr. Janet F. Werker (Department of Psychology) were co-supervisors of the research reported in this thesis. Dr. Mark R. Greenstreet (Department of Computer Science) served as a faculty reader for the thesis and Mr. Gary MacIsaac was the student reader. Ms. Elicia Wolowidnyk provided a helpful introduction to the Infant Research Laboratory in the Department of Psychology at the University of British Columbia. Mr. Brian Moorhead gave valuable advice and assistance concerning a number of hardware issues discussed in the thesis. Dr. Donald G. Jamieson of the Department of Communicative Disorders at the University of Western Ontario, Advanced Gravis Computer Technology Ltd. (Burnaby, British Columbia) and Ms. Sai Choo loaned equipment that was instrumental in the research. Financial support for the thesis research was provided by Ms. Sai Choo and for the Infant Research Laboratory and the Imager Computer Graphics Laboratory by the Natural Sciences and Engineering Research Council of Canada.

Chapter 1

Introduction

Experimentation in the field of psychology is based on an understanding of the current theoretical body of knowledge, good experimental design appropriate to the area under exploration and various supporting tools to conduct the experiment. Some of the computer tools used in psychological experimentation are also used in other fields. Many are extensively used only within the field of psychology. The general purpose computer is an example of a widely used tool. Special packages of experimental tools, which usually include both hardware and software, are examples of specific tools particular to a narrow area of research. This thesis examines such a package of hardware and software that has been developed to provide support for experiments and studies employing the Headturn procedure.

The Headturn procedure was developed to study sound discrimination in infants. Because infants do not understand speech, an indirect method of measurement is required. Previous research had shown that infants would turn their heads in the direction of various stimuli under certain specific conditions. The Headturn procedure was designed to make this phenomenon as controllable and repeatable as possible. A hardware and software system that controls the experimental Headturn procedure has been in use for several years. During this time, significant shortcomings in both the hardware and the software have become apparent. These are due both to failure of the product to meet the original needs of the users and to advances in the field of infant speech research, which have led to additional needs not foreseen in the initial system. These shortcomings were significant enough to justify an assessment of the situation and a reimplementation of the software.

As general background, Chapter 2 briefly surveys the use of computers within the field of psychological experimentation. Chapter 3 then describes the evolution of the Headturn procedure from its inception to the current state of the art. Computer control of experiments allows both the precise timing of the initiation of stimuli and the precise recording of response times; Chapter 4 examines timing issues that are critical to many experimental techniques in psychology, including the Headturn procedure. Chapter 5 analyzes the existing Headturn software and discusses shortcomings that were identified. Chapter 6 discusses the design and implementation of a new Headturn system, with the focus on generalizing the software, utilizing reliable hardware components, improving the user interface and extending previous capabilities to meet specific user requests.

Chapter 2

Experimental Techniques and the Use of Computers

The Concise Oxford Dictionary defines psychology as the "Science of nature, functions & phenomena, of human soul or mind." By this definition, psychological experimentation is any experimentation to discover the nature, functions and phenomena of the mind. Because of the great complexity of human behavior, it is only possible to study a small part of it at a time. Because of practical, ethical and legal considerations and because of the limited amount known about the human mind, measurements of the nature, functions and phenomena are generally done indirectly. The ease with which studies can be conducted depends significantly on the use of techniques that make these indirect measurements easy. Such techniques are often specific to each area of study within psychology because of the intricacy of the relationship between what is being investigated and what is being measured as an indirect indication of the phenomenon under investigation.

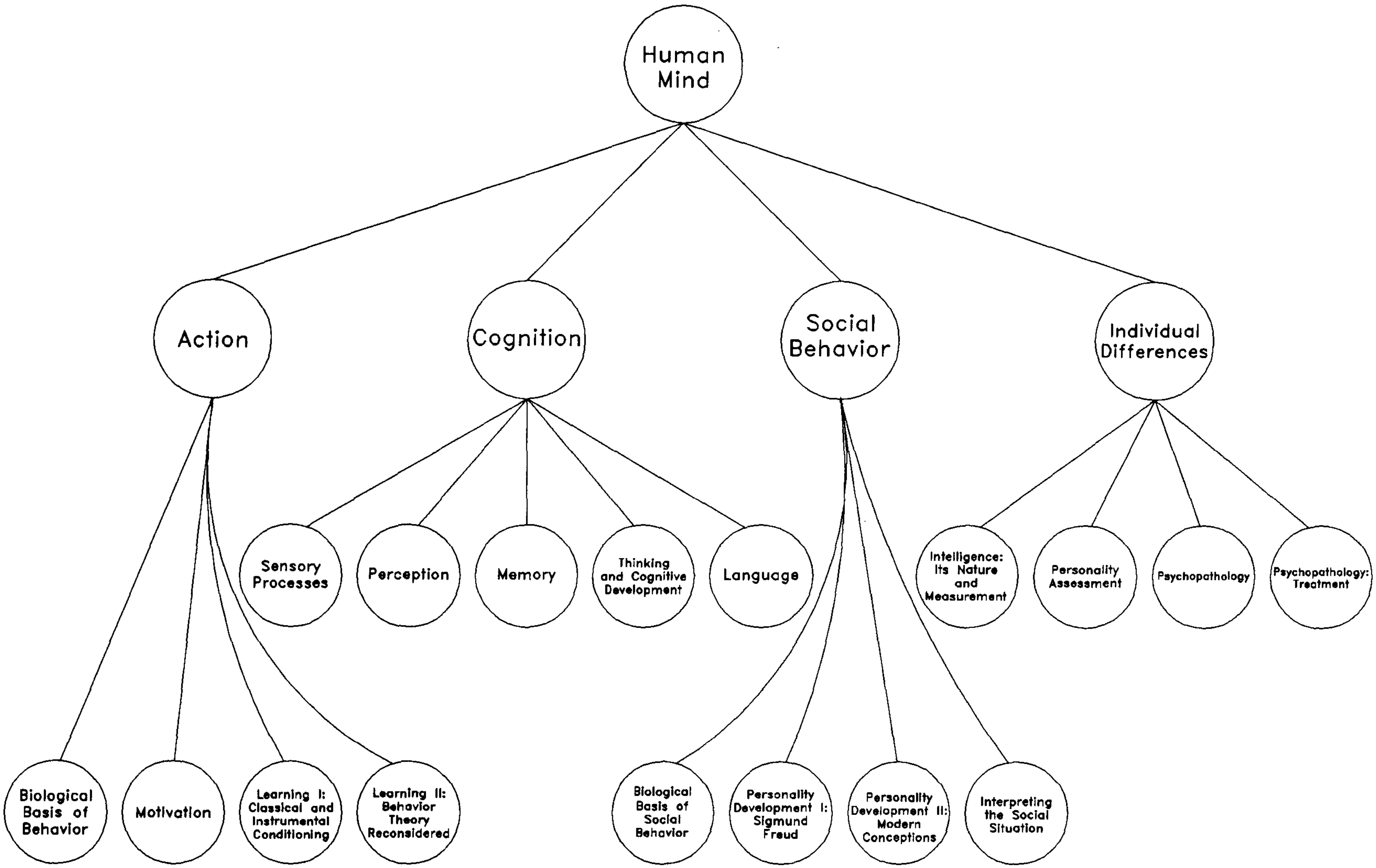


Figure 2.1: Taxonomy of Field of Psychology

2.1 A Taxonomy of Psychology

There are many ways to describe psychology, perhaps as many ways as there are psychologists. The taxonomy of the field shown in Figure 2.1 is derived from Gleitman's general text on psychology [Gleit81]. He divides psychology into four major areas each of which he divides into four or five sub-areas. The two major areas on the left side of the diagram, action and cognition, are currently more amenable to computerized experimentation than the two areas on the right, social behavior and individual differences. This is largely because current computer practices tend to be focused on measuring quantities over relatively short time frames using individual subjects. Most of the sub-areas in the action and cognition areas lend themselves to this type of experimentation. With the evolution of truly portable computer hardware, the evolution of better networking systems and the passage of time, there will likely be more use of computers in the study of both the social behavior and individual differences areas.

2.2 Measurements in Psychological Experimentation

The most commonly measured quantities in psychology are the time and the content of subject responses. Time can be measured directly as a reaction or response time. Duration of stimulus display, inter-stimulus intervals or allowable user response times can be quite varied in duration. Actual subject responses are most commonly motor or verbal responses. These are indirect measures of the more fundamental responses that are being studied. The most common motor response is a button or key press. Another common response technique is the measurement of physical changes in the subjects during the experiment. Scalp potentials, pupil size, eye and head movements can all be measured. In some animal studies, the brain itself can be analyzed during an experiment, although ethical considerations and cost preclude this as a common procedure. With humans, modern technology such as PET or MRI now make it possible to do some direct brain measurement with a minimum of invasiveness. The cost of this technology currently restricts this type of research to laboratories with a high level of funding.

Computers are well suited to performing the timing functions employed in typical psychology experiments. They can keep an accurate record of subjects' responses such as button presses or key strokes for an indefinite period if the presses or key strokes are captured by the computer. Equipped with an analog to digital converter, computers can also record physical quantities measured during an experiment. Where computers fall short is in measuring responses that are difficult to quantify. Complex subject responses involving things like speech or body movement are not easily handled by computers except in rather stylized ways. This problem is not unique to computers or psychology. Research in any area that deals with phenomena that are difficult to quantify causes problems in the comparison and reproduction of results. In psychology, areas of research that are dependent on human judgement in their measurements are poor candidates for computerization. At the other end of the spectrum, areas of research that are dependent on non-computer instrumentation are ideal candidates for computerization. In the former, a hybrid approach can be used in which a human makes the qualitative judgements and a computer records them and makes quantitative measurements.

2.3 The Limits to Computer Use

There are limits to the use of computers in psychological experimentation. The first limit, discussed above, applies to experiments that are dependent on human judgement. Computer use in this type of experiment is generally restricted to administration. Experiments that involve quantitative measurement are much more amenable to computerization.

The second limit is software. Given the specialized nature of research, mass market software generally is not available. Many researchers have neither the funding nor the will to get involved in major software development projects. The end result is that computers are not used in all of the experiments where they could be used.

The third limit is the perceptual problem. In most cases, a computer presents information as two dimensional digitized images displayed by using a video board or as digitized sounds played by using an audio board. An image on a screen is not exactly the same as an image on a slide or as the real object. Reading text from a screen is not exactly the same as reading text from a sheet of paper. Digitized speech is not the same as speech that is directly spoken. As a result, there is no guarantee that a subject's perception of a computer generated event will be identical to the perception of a real event in the modality being studied. This limit is not unique to computers. For example, in the study of response to traumatic events, a subject's response to viewing a picture of an event is very likely to differ from a subject's response to viewing an actual event.

The fourth limit is physical. Experiments performed in the field or on subjects that are in motion may require a combination of power and portability that either isn't available or isn't affordable.

The fifth limit is the computer skills of the available personnel. This limit is compounded by the lack of good software. Again the result is that many experiments that could feasibly be computerized are not. As computers become a more regular part of life, this limitation should recede.

2.4 A Literature Sample

In order to gain more insight into the use of computers in psychology experiments, a sample of the literature was initially reviewed. Fifty papers, describing a variety of psychological experiments, were examined and the important procedural features of each experiment and the computer support that each experiment employed were tabulated. The papers were selected to represent the mainstream of the psychology literature, although no claim is made that the review is complete or definitive.

Papers were selected from the following journals: *Journal of Experimental Psychology: General*; *Journal of Experimental Psychology: Learning Memory and Cognition*; *Journal of Experimental Psychology: Human Perception and Performance*; and *Journal of the Acoustical Society of America*. Table 2.1 gives a summary of the papers surveyed and the computer support each used. Some experiments used more than one type of computer which is why the total for a column will not necessarily be equal to the sum of the numbers in the column. The tabulated results of the survey are displayed in Appendix G.

Table 2.1: Summary of Papers Sampled

Journal	*JEP:GEN*	*JEP:LMC*	*JEP:HPP*	*JASA*	*JASA*	*JASA*
Vol-#	120-all	17-4	17-3	91-1	91-2	91-5
PC	5	1	4	2	1	1
Apple	1	2				
Commodore 64	2		2			
Workstation	1					
Digital Equipment	1		1		1	1
Terak	1					
Amiga		1		1		
None	5	5	1	1		
Unknown	4	2	1	2	2	5
Total	17	11	8	5	3	6

The *Journal of Experimental Psychology* was assumed to represent the mainstream in psychology. It is one of the most desirable journals in which to publish. Three of the four journals that make up the *Journal of Experimental Psychology* were surveyed. The *Journal of the Acoustical Society of America* was assumed to represent the mainstream in the study of hearing and audiology. In preparation for the current research, a considerable number of papers were studied from the *Behavior Research Methods, Instruments, & Computers Journal*, but these were not included in the review because the emphasis in that journal is on the computer techniques themselves rather than on psychological research.

2.4.1 Acoustics

The first observation that emerges from the survey is that the acoustic area of psychology, psychoacoustics, stands apart from the rest of psychology. This is not surprising. In acoustics, the experimenter is dealing with time varying air pressures which can be measured and analyzed. As a result, experiments in acoustics are very likely to use computers and other electronic equipment. This situation carries over into psychoacoustics. Only one of the fourteen experiments surveyed from the *Journal of the Acoustical Society of America* did not use a computer. These were experiments in psychoacoustics. Ten of the thirty-six experiments surveyed from the *Journal of Experimental Psychology* used no computer.

2.4.2 Types of Computers

The second observation is about the type of computer being used in psychological experimentation. The IBM compatible personal computer (PC) was used in fourteen of the thirty-nine experiments that reported using a computer. It is likely, however, that the majority of the sixteen experiments that used a computer of unknown type, in fact, used a PC. This is probably a case where the obvious is not documented. Psychoacoustics appears also to be dominated by PCs. Nine of the fourteen psychoacoustic experiments surveyed used a computer but did not report the type. Commodore 64s and DEC minicomputers were each used in four experiments, Apple in three and Amiga in two. Only one experiment reported the use of a workstation computer. The low hardware prices and the high availability of personnel with PC experience are likely the two major reasons for the dominance of PCs. Minicomputers tend to be used for historical reasons. The use of the Apple, Commodore 64 and Amiga computers is probably the result of history, loyalty and perhaps some superior capability in a specialized area. There are several reasons for a lack of experiments using workstations.

1. Most of the experiments don't require the additional capabilities that a workstation provides.
2. Workstations are generally more expensive than microcomputers.

3. The workstation environment requires a significantly higher level of computer expertise to master.
4. Personnel with workstation experience are harder to find than personnel with PC or Apple experience.

There are areas, such as the study of human computer interfaces, where the superior capabilities of a workstation are desirable but many areas of psychology do not need this level of support.

The use of PCs in psychological experimentation is not without problems. The problems stem from the fact that the uses that psychologists have for PCs are outside of the PC mainstream. One common problem is the need for millisecond timing in both displays and input. Normally, the PC timer interrupt occurs once every 55 milliseconds which is well below the precision required for many experiments. Various different schemes have been implemented to address this situation. The more important of these schemes are described in Chapter 4.

As mentioned earlier, software is a major problem. There is, however, no evidence that the software situation is improved by using any other type of computer. Schneider discusses this issue at length [Schnd91]. On the positive side of the ledger for the PCs, the open architecture and the simple operating system make it possible for a skilled programmer to produce the millisecond timing needed for input and display. The open architecture has also encouraged the development of a large number of third party expansion boards for the PC bus. Some of these boards are useful to the psychologist. As PC hardware becomes cheaper and more powerful, it could become more useful to psychologists. The evolution of multimedia significantly enhances the range of stimuli that can be presented. Unfortunately, the new operating systems like Windows and OS/2 do not directly address the most common problems psychologists currently have with PCs. If anything, they make matters worse. For example, programming under Windows or OS/2 to get millisecond timing control is significantly more difficult than it is under DOS. Direct screen control is also more difficult under Windows or OS/2 than it is under DOS.

2.4.3 Computers for Administration

The third observation concerns the widespread use of computers for administration. Twenty-five of the thirty-six regular psychology experiments used a computer for at least one of ordering of the presentation, recording of results or analysis of results. In the psychoacoustic area, it was only five of fourteen, but it may be that these functions are taken for granted in psychoacoustics and weren't reported. The papers in the *Journal of the Acoustical Society of America* were not as rigorous as the ones in the *Journal of Experimental Psychology* in documenting the equipment set up and experimental procedures. Another administrative task that wasn't reported in any of the papers surveyed was the automatic recording of the configuration and results for future reference. The configuration includes all of the hardware and software components as well as the settings for the hardware and the parameters for the software. Properly organized, a computer can produce and maintain much of this record more reliably and more accurately than a manual system, at least for those components of the experiment under computer control.

2.4.4 Computers for Timing

The fourth observation is the importance of timing. This can be seen from the fact that fourteen of the experiments measured reaction or response time. In addition to the need for millisecond accuracy, these experiments require a reliable method for measuring the delay between the subject response and associated reading of the main PC clock. The alternative would be to use a clock in the peripheral response device. This would simplify the measurement of the time of the user response but could require synchronization of one or more peripheral clocks with the main PC clock.

2.4.5 Computer Screen Use

The fifth observation is that in the use of the screen, the majority of the experiments did not go beyond text, simple line drawings or dot patterns. Most experiments that involve a timed response to the display of a visual stimulus are amenable to computerization. A CRT display is not exactly the same as a paper drawing, a projected slide, a photograph or the real thing, but in many cases it's close enough. The 16 msec delay in filling a screen and the sequential display of the stimulus has not prevented extensive use of CRT screens for tachistoscopic applications. There were two experiments that did simple animation. None of the experiments seemed to be pushing the limits of the available video systems. Given the sophistication of some of the work with screen systems that has been reported in the computer science literature [Ather85] [Beat91], there is the possibility for major advances in screen handling within the psychology mainstream.

2.4.6 Computer Sound

The sixth observation concerns the use of sound. With one exception, none of the experiments outside the psychoacoustic area used anything other than beeps. In the psychoacoustic area, the experiments used speech, analog signals, digitized signals that had been stored, and noise masking.

Virtually all experiments that involve a timed response to an audible stimulus are amenable to computerization. The ease with which digitized stimuli can be manipulated has freed the experimenter from the need for laboriously constructing tapes and highly complex electromechanical control systems. Digitized sound is used in experiments. But none of the experiments in the psychoacoustic area used stored synthetic sound which would seem to suggest that it is not yet in common use. Synthetic stimuli have the advantage that they are composed of known components and can be recreated exactly. The disadvantage of synthetic stimuli is that they are not identical to natural stimuli. The difference may be significant in the studies being done. Synthesis of sound on the fly for experimental purposes did not seem to be in evidence at all.

2.5 Obstacles and Potential Solutions

As the power of readily available computers and software tools grows, it likely will be possible to study a larger part of the human mind at one time. In the extreme, subjects could be immersed in an artificial reality in which a vast array of responses is recorded. These responses would be correlated and analyzed by the computer to give greater insight to the experimenter. But this is still in the future.

Within the current limits of computerization, there are several obstacles that a researcher encounters when attempting to use computers as tools in experimentation. The first obstacle is cost. Despite steadily declining prices, hardware and software still cost money. A potentially greater barrier is the cost of the expertise needed to keep the computer system functioning correctly. Retaining a knowledgeable person committed over a period of time often requires significant funding. The second obstacle is software. We will focus on this.

Because the psychology market is small, there is little incentive to develop commercial software which could be installed and used by non-technical personnel. Because researchers are researchers and not software developers, it is generally difficult to develop good software locally. The high cost of software development is also a strong deterrent to local development. Poorly developed software may contain bugs that can invalidate months or years of research. As described in Edgell [Edge89], a researcher has four choices for software.

1. Assembler programming
2. High level language programming
3. Programming in an experiment control language
4. Using a configurable experiment control system

The assembler option is the only one that provides all of the flexibility and control that a researcher is likely to need. There are two reasons for this. The first is that direct control of specialized hardware may only be possible using assembler. The second is that control of program execution time is much better using assembler. Unfortunately most professional programmers are not fully fluent in assembler. Development using assembler is slow and as a result very expensive. Assembler programs are prone to bugs especially if they must be modified after the original development. A mix of assembler and high level language will usually do the job but the programming task is still beyond the capabilities of most non-programmers, development is still expensive and program bugs are still a serious problem.

Experiment control languages tailored to the syntax and semantics of experiment control would seem to be a good solution. Unfortunately, the market isn't large enough to support the independent development of really good experiment control languages. Configurable experiment control systems are generally specific to a particular area of research. They also tend to lack flexibility. Despite these limitations, the MEL system is installed at over 1000 sites [Schnd88]. The CSRE experiment control system is also widely used [Jam92]. Edgell's solution is a library of subroutines callable from a high level language. This is probably feasible if there exists a large enough community in a particular research area to develop and support these subroutines and if the community is sophisticated enough to make use of the subroutines. Despite the problems with software, the current situation is an improvement over the situation that existed in the past when specialized pieces of hardware had to be connected in a unique manner for each experiment. Yaphe, Raftery and Jamieson describe a system for solving that type of problem [Yaphe89]. Now the problem is largely one of configuring just the software.

The overall situation can be summarized as one in which the absence of specific computer solutions to support specific experimental technique has been a major restriction on the use of computers in experimental psychology. The next chapter traces the development of the Headturn procedure, which is one of these specialized techniques.

Chapter 3

History and Current Headturn Practice

The study of cognitive development in early childhood is an important field of psychology. The study of the development of speech recognition ability is one important part of this field. It is best explored by studying infants directly. In the conventional model of psychological experimentation, subjects are instructed as to the expected responses. Their responses, which provide the experimental data, are normally some sort of verbal or motor response. However, infants are not able to understand instructions, they are not able to verbalize and they have limited motor development. As a result, the conventional model often cannot be applied to experimental research with infants. Other techniques need to be developed for measuring an infant's response to stimuli. An example of this is the Headturn technique which has been developed because infants cannot press buttons or speak in response to stimuli the way that older children and adults can.

Most infants are capable of turning their heads by the end of their fifth month and will do so spontaneously under a number of circumstances. To use the head turn as a research technique, the infant must be conditioned to turn its head under one set of conditions and remain stationary under another. In this way, the infant's ability to differentiate between the conditions can be studied.

In speech research Headturn is used to test an infant's ability to differentiate between various speech sounds. The infant is conditioned to look straight ahead when one sound is presented and to turn its head when the sound is changed. To enhance the conditioning process, the infant receives reinforcement for turning at the correct time. This reinforcement is in the form of a visual display that the infant finds interesting. The reinforcement is required in order to obtain the infant's cooperation in the experiment. Without reinforcement, the infant's responses are likely to be quite variable. The conditioning can be done using two sounds with a very definite difference. The infant can subsequently be tested on its ability to discriminate between sounds which differ in more subtle ways. An important part of the Headturn procedure is the separation of the stimulus, which is audible sounds, from the reinforcement, which is visual. Previous techniques, such as High Amplitude Sucking, use the stimulus itself as reinforcement. The infant is presented with an audible stimulus and the infant's sucking rate is measured. When the audible stimulus is changed, a significant change in sucking rate is taken as evidence that the infant can discriminate between the two stimuli. There is nothing other than the change in sound to reinforce the increase in sucking rate.

The idea of reinforcing an infant's head-turning response to sound and other stimuli has its roots in early studies conducted in the 1930's in which experimenters played a form of "peek-a-boo" to attract the infant's gaze [Dix47]. The major milestones in the development of the current Headturn procedure are summarized in the next sections. Werker's Ph.D. thesis has a more complete history of Headturn research [Werk82].

3.1 Suzuki and Ogiba

As reported by Moore [Moore75], the idea that a visual stimulus can reinforce response to an audible stimulus was developed by Suzuki and Ogiba and first reported in 1961. Children were conditioned to turn toward a sound source to see an illuminated doll. This technique was called conditioned orientation reflex audiometry. Suzuki and Ogiba did not report the use of a control group in their study, so the reinforcement effect of the visual stimulus was not measured.

3.2 Moore I

In a 1974 study, Moore examined the reinforcement effects of different visual stimuli in auditory localization for forty-eight infants between the ages of twelve and eighteen months [Moore75]. Auditory localization means finding the direction from which a sound originates. When an infant turned its head to face in the direction of a sound source in Moore's experiment, it could be considered to have localized the sound source. The infant was seated on its mother's lap. An assistant sat in front of the infant to focus the infant's attention straight ahead. The experimenter played the stimuli from an audiometer in the control room. A loud speaker, which produced the stimuli, was located five feet from the infant and at a 45° angle to straight ahead. A turn toward the loud speaker by the infant was counted as a head turn. Both experimenter and assistant had to be in agreement before a head turn was counted as a localization response. In addition to the intervals with a stimulus, there were intervals with no stimulus. Head turns were recorded during these intervals to measure the infant's tendency to turn toward the loud speaker in the absence of any stimulus.

The infants were divided into four groups. The first group received no reinforcement. The second group received social reinforcement in the form of smiles, pats on the back and encouraging words. The third group received simple visual reinforcement in the form of a flashing red light. The fourth group received complex visual reinforcement in the form of an animated toy animal that moved in place. In all but the first group, reinforcement was given after an appropriate localization response. For the third and fourth groups, the visual reinforcers were activated by the experimenter in the control room. The results of the study suggested that the complex visual stimulus resulted in more localization responses than no reinforcement, social reinforcement or simple visual reinforcement.

Moore's 1974 study was the first published account of the use of the Headturn procedure. It confirmed and quantified the key element of the procedure, which is that a visual stimulus can reinforce an infant's response to an audible stimulus. The procedure has been refined a certain amount over the years and the equipment has become progressively more automated, but a similar spatial arrangement of the personnel and equipment is still in use today. The stimuli in Moore's study were played by the experimenter using an Allison audiometer in the control room according to a preset schedule. There is no description of the procedure for activating the reinforcers or tallying the votes in the reports on the study; it was likely done manually.

3.3 Wilson

In his study reported in 1976, Wilson tested ninety infants between the ages of five and eighteen months for auditory thresholds using animated toys as visual reinforcers

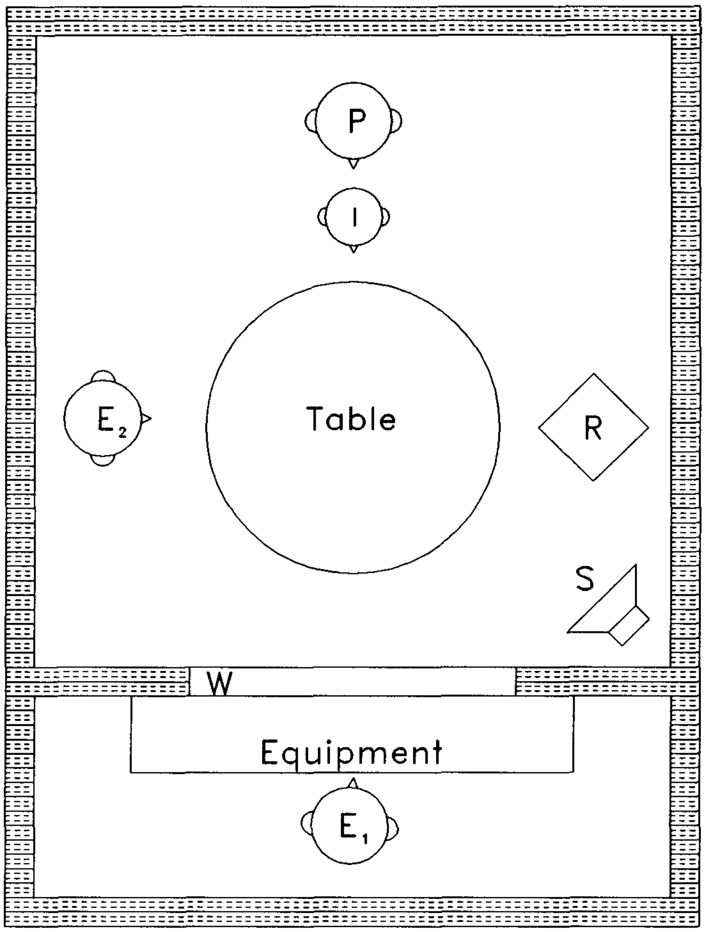
[Wils76]. An auditory threshold is a measure of the minimum sound that can be heard at a given frequency. Only sixty percent of the five month old infants were able to complete the study while all but one of the six month old infants were able to complete, suggesting that five months is about the minimum age for participants in Headturn studies. This study extended the Headturn procedure in two respects. Younger infants were tested and auditory thresholds were measured rather than localization responses.

3.4 Moore II

Moore's 1976 study extended the 1974 study by testing sixty infants between the ages of four and eleven months [Moore77]. The effect of complex visual reinforcement on the auditory localization response was studied. This study established that five months is the minimum age at which the visual reinforcement is effective in reinforcing an auditory localization response. The set up of the study was essentially the same as Moore's 1974 study, however, the description of the 1976 study is more extensive [Moore75]; it describes an intercom for communication between the experimenter and assistant and a manual timer used to measure the interval during which a head turn could be counted [Moore77]. The reinforcers were activated manually by the experimenter in the control room and votes were recorded and tallied manually.

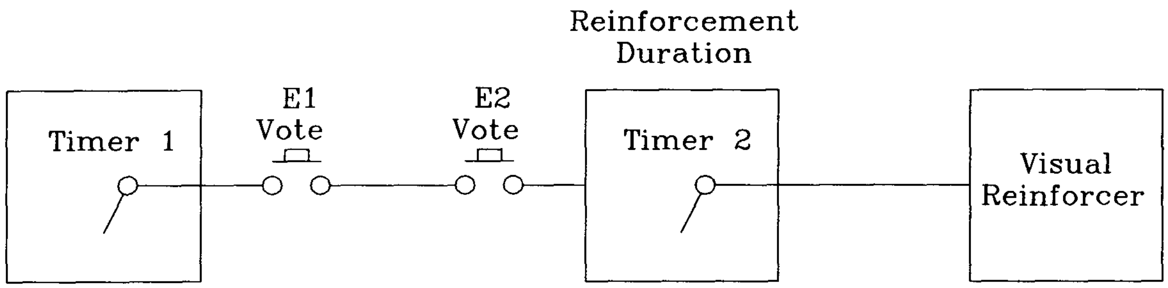
3.5 Eilers

In 1976, Eilers studied seventeen infants [Eiler77]. Nine of them were six months old and the other eight were twelve months old. They were tested for their ability to distinguish between speech sounds. The set up of the experiment, which is similar to the



E_1 – Experimenter
E_2 – Assistant
P – Parent
I – Infant
R – Visual Reinforcer
S – Loud Speaker
W – One Way Glass Window

Figure 3.1 Headturn Implementation (Eilers)



Timer 1 defines signal–change interval, Timer 2 initiates and controls duration of visual reinforcer

Figure 3.2: Visual Reinforcement System FLow Chart (Eilers)

one described in Moore's studies, is shown in Figure 3.1. In Moore's studies, the assistant sat directly in front of the infant. In Eilers' study, the assistant sat 45° to the infant's right. Taped stimuli were played through the audiometer by the experimenter in the control room. The reinforcers were controlled by the system shown in Figure 3.2.

In Eilers' procedure, both the experimenter and the assistant had to vote for a head turn before the reinforcers would be activated. The first timer controls the length of time during which a response may be reinforced. The second timer controls the duration of the reinforcement. The methods for starting the first timer and recording the votes are not described so it is assumed that these were done manually. This was the first study to use the Headturn procedure for speech discrimination.

3.6 Kuhl

Kuhl's 1978 study used synthetic stimuli that were actually stored digitally before being transferred to a two channel analog tape recorder [Kuhl79]. A more sophisticated special purpose logic device was used to assist in the control of the experiment. The stimuli were still played from analog tape, but the logic device controlled all other aspects of the experiment. The logic box was an important enough advance in experiment control to be described in detail. It was a collection of six subsystems: the stimulus control system, the timing control system, the reinforcer control system, the display system, the recording system and the communication system.

The stimulus control system had two input channels. There was one set of stimuli on each channel. One output channel was connected to an amplifier. A probability generator chose input channel one or two with a probability of 0.5, deciding which of the two input channels was to be played on the output channel when the start button was pressed to initiate a trial. Between trials the stimulus from the channel that had been designated as the background channel was played. If the stimulus from the background channel was chosen for a trial, it was a control trial. If the other channel was chosen, it was an experimental or "change" trial. A manual override allowed the experimenter to choose the input channel manually. There was a volume control for the output channel.

The timing control system had a timer that controlled the duration of a trial and a timer that controlled the duration of reinforcement. There was a start button to initiate a trial. There was also an abort button whose purpose wasn't documented.

The reinforcer control system automatically activated the reinforcer whenever both vote buttons were pressed during a change trial. There was also a button to activate the reinforcer manually and a switch to select the left, the right or both reinforcers.

The display system contained five LEDs which indicated the following conditions when they were lit.

1. The trial was a change trial.
2. The experimenter had voted.
3. The assistant had voted.
4. The reinforcer had been activated.
5. The trial was correct. A correct trial is one in which a head turn happens if and only if the stimulus changes.

There were also two digital displays. One of them displayed the trial number and the other displayed the latency time between the onset of the audio in a trial and either the end of the trial or the press of the second vote button. An analog decibel meter could be switched to display the volume of either input channel or the output channel.

The recording system had a vote button for the experimenter and a remote vote button for the assistant. The vote button was pressed to indicate a head turn by the infant. The status of the five LEDs and the value of the two digital displays were printed after every trial. There was also a reset button to return the system to the background channel and reset the trial number, the latency timer and the state of the LEDs

The communication system had a vibrating pin in the assistant's vote button. The pin was activated during a trial to inform the assistant that a trial was under way. There was a "volume" control for this pin. A cassette tape player was used to supply music to the headphones that were worn by the mother and by the assistant to reduce the possibility that one of them might inadvertently influence the infant. The communication system also included a microphone to allow the experimenter to speak to the assistant. There was a button which interrupted the music going to the assistant's headphones and connected the microphone. There was also a volume control for the speech going to the assistant.

The logic box represented a significant step forward in implementing the Headturn technique. It removed an area of potential bias by automatically selecting the type of trial that would be done. It also measured the latency time and recorded the results for each trial. It activated the reinforcers automatically for a predetermined length of time. The major drawback to the logic box is the expense and time involved in designing, building, maintaining and modifying such a specialized piece of hardware.

3.7 Werker

Werker reported two studies, one in 1981 and one in 1983, that used an implementation of the Kuhl logic box [Werk81] [Werk83]. Both of these studies used stimuli on tape. The stimuli were chosen manually by the experimenter using a random number table and the results were recorded automatically using a multi-channel pen-based event recorder. A later study, reported in 1988, played the stimuli from tape but used an Apple II+ computer in place of the logic box [Werk88]. The experimenter controlled the computer with a two-pedal foot press device. One pedal initiated a trial. The computer

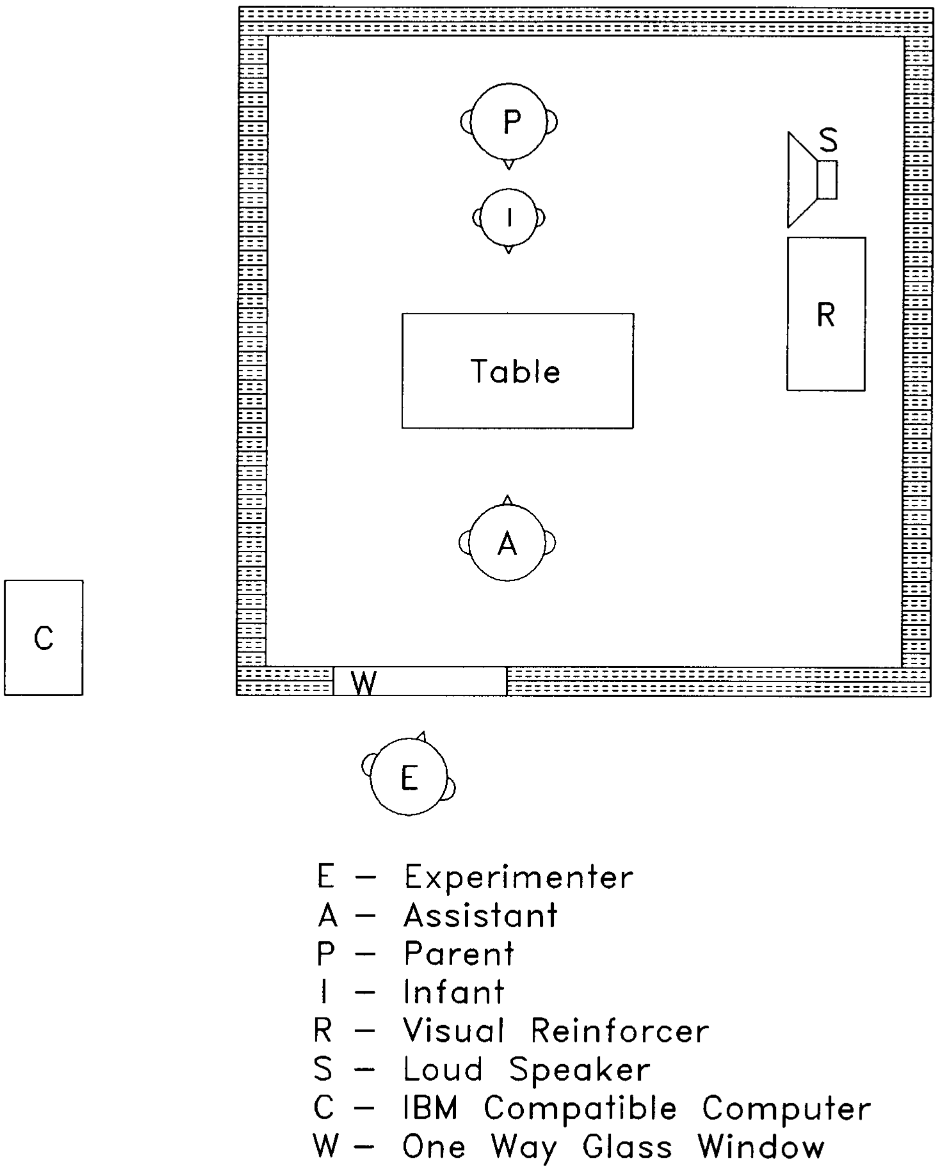


Figure 3.3 UBC Headturn Implementation

chose the type of trial and would switch tracks on the tape when a change trial was required. The other pedal recorded a head turn. The computer recorded the results. The replacement of the special purpose logic box with a general purpose computer was a significant step forward because specialized software is easier to design, implement and maintain than specialized hardware. A study reported in 1991 used stimuli that had been digitized [Werk91]. The use of digitized stimuli allows a great deal more flexibility in stimulus presentation and saves considerable time in stimulus preparation. A Compaq 286 running the Headturn software was used to control the experiments.

The current practice of the Headturn technique used at UBC has the infant seated in an acoustic enclosure on its mother's knee. To the left of the infant is a loud speaker, which produces the stimuli, and an enclosure containing four mechanical toys. The toys are normally hidden by a sheet of dark Plexiglass, which covers the front of the enclosure. An assistant sits directly in front of the infant and provides encouragement and feedback to the infant. Both the assistant and the mother wear headphones which play music so that they cannot hear the stimuli. The experimenter stands outside the acoustic enclosure, viewing the experiment through a one way window and controlling the experiment using the Headturn software. The experimenter controls the Headturn software using a custom-built, special purpose button box which is connected to the serial port of a microcomputer. The set up for a Headturn experiment is shown in Figure 3.3.

This physical arrangement is desired so that the infant turns its head to the left for a correct response in a change trial. This is an important detail because even at birth infants are able to turn to the right more easily than to the left. Adopting this physical arrangement reduces the chances of a false alarm response.

The Headturn software, under the control of the experimenter, is responsible for playing the background and target stimuli. It is also responsible for activating the toys at the appropriate time. When a toy is activated, a light is switched on in the enclosure so that the toy is visible and electric power is supplied to the toy so that it moves and makes some noise. The trial-by-trial results are recorded by the Headturn software and a separate module is used to tabulate and print the results. There is an intercom system that allows the experimenter and assistant to communicate. The intercom is switched off during the testing stages of an experiment. In this implementation of Headturn, the assistant in the booth doesn't vote. In the past, the agreement rates between the votes of the experimenter and the assistant have been very high so it was not considered necessary to require the assistant to vote.

During the training stages of the experiment session, all of the trials are change trials. The reinforcers are activated unconditionally. During the conditioning stage, there are both change and control trials. The reinforcers are activated only after a head turn during a change trial. A more complete technical description of the Headturn computer system in use at UBC is presented in Appendix A and state diagrams are in Appendix B.

Chapter 4

Timing Issues in Computerized Experimentation

The majority of psychology experiments consist of a series of timed events. The maximum precision required is in the 1 msec range. In using computers to control psychology experiments the I/O delays are often important. There is a delay between the initiation of an output command by the CPU and the production of the output by the output device. There is also a delay between the activation of an input device by a subject and the receipt of the input by the CPU. The larger the number of I/O devices used in an experiment, the more complex the I/O delay situation becomes. The software used in experiments must be capable of real-time measurement. In psychology, the real-time restrictions are most often set by human limitations. These limitations are generally linked to the speed with which the human nervous system can process information and to the speed with which the neuromuscular system can move parts of the body. Some typical times for these operations are summarized here.

1. Reflex time: 20-30 msec. A reflex occurs when a muscle flexes in response to an extension of the muscle. The involuntary extension of the knee joint that

occurs when the leg is tapped just below the kneecap is the best known example of a reflex.

2. "Quick loop" time: 60-90 msec. A quick loop is a correction to a perturbation or a rapid adjustment in the grip and load forces in the hand.
3. "Fast correction" time: 100-120 msec. A fast correction is a correction to a visual perturbation or rapid response in the arm trajectory for an ongoing movement.
4. Reaction Time: 120 msec to several seconds. Reaction time is the elapsed time between the presentation of a stimulus and the initiation of a response. If a button is to be pressed when a certain stimulus appears on the screen, the reaction time is the elapsed time between the appearance of the stimulus and the start of movement of the finger.
5. Movement times: 100 msec to several seconds. The movement time is the elapsed time from the initiation to the completion of the response. If a button is to be pressed, the movement time is the elapsed time from the initial movement of the finger until the completion of the button press.

This chapter discusses a number of timing issues that occur in computer support for experiments such as that used for the Headturn procedure. The discussion focuses on the IBM PC compatible platform (the one used for Headturn), but most of the issues remain the same in other hardware and software environments.

4.1 Overview of the PC Architecture

To understand the timing capabilities of IBM PC compatible computers, it is necessary to understand a little about the machine architecture. IBM PC compatibles are the most common general purpose computers in the world. They are based on the Intel series of microprocessors. The original IBM PC used an 8088 processor. Subsequent machines have used the 80286, 80386, 80486 and Pentium processors. Each succeeding processor has featured:

1. Higher clock speeds than the predecessor.
2. Many instructions that execute in fewer clock cycles than they did on the predecessor.
3. A moderate increase in the size of the instruction set.
4. More sophisticated support for such things as virtual memory and virtual machine emulation.
5. Upward compatibility with the predecessor.

The last point is particularly significant. A program written using the earliest instruction set, the 8088, will still execute on the latest processor, the Pentium.

The device level interface on the PC is provided by the Basic Input Output System, commonly known as the BIOS. The BIOS insulates the system and application software from the hardware by providing primitive I/O services. Because the BIOS interface to the system and application software that interfaces to it has remained stable throughout major changes in the hardware, a high degree of compatibility has been maintained. The BIOS reserves the area in the PC main memory from 0400H to 04ffH as a data area. The BIOS is normally resident on Read Only Memory (ROM) chips which are part of the PC hardware. The original IBM PC BIOS was produced and copyrighted by IBM. Since that time, the production of BIOSs has come to be dominated by a small number of companies who are not dependent on any particular manufacturer of PCs. This situation has been a major factor in the high degree of compatibility in the PC world.

The Intel processors used in PC compatibles support vectored interrupts. The interrupt vectors are stored in the PC memory in the range from 0H to 03ffH. Each interrupt vector is four bytes long. The first two bytes of the vector are an offset address. The second two bytes are a segment address. When an interrupt is invoked by hardware or by an INT instruction, the machine state is saved and program execution jumps to the address specified by the segment:offset combination specified in the interrupt vector corresponding to the interrupt number that was invoked. After the interrupt has been serviced by executing the interrupt service routine pointed to by the interrupt vector, an IRET instruction is used to restore the machine state and return execution to the pre-interrupt address. Interrupts are commonly referred to by their ordinal position in the interrupt vector table. For example, int 09H, the keyboard interrupt, is the tenth interrupt in the vector table. Int 00H, the divide by zero interrupt, is the first one.

Int 08H is the timer interrupt. All PCs contain an Intel 8253 or compatible timer chip which interrupts the CPU at periodic intervals. The interrupt service routine for int 08H updates the time of day clock and executes an int 1cH. Normally, int 1cH consists only of an IRET instruction, but the interrupt is provided to allow application software to perform functions at intervals specified by the timer interrupt.

Display systems on the PC can be divided into monochrome and colour. Virtually all monochrome video systems support the Hercules specification. There is one 80 X 25 character mode and one 720 X 348 pixel graphic mode supported. A Hercules compatible card contains 64 kilobytes of memory. Most colour video systems support the Video Graphics Array (VGA) specification. VGA cards support a wide variety of video modes. The 80 X 25 character text mode and the sixteen colour 640 X 480 pixel graphics mode are the ones most commonly used. A VGA card normally contains between 256 kilobytes and 1 megabyte of memory. Higher resolution graphic modes are available on most VGA cards but software drivers for these modes are often unavailable or difficult to configure. This situation is due largely to the size of the market and the lack of standardization in the higher resolution video modes. Operating environments like OS/2 and Windows that insulate the application from the video hardware are improving this situation.

4.2 Timing Precision

When the limits of human performance are being investigated, as in psychology experiments, mass market tools often do not have the necessary timing precision. Normally the timer interrupt on a PC happens 18.2 times per second. The 16-bit counter on the Intel 8253 timer chip is set to 0, it counts down to 0, giving a period of 65536 counts, then interrupts the CPU. The cycle is repeated as long as the PC is running. The number of interrupts since midnight is stored in the BIOS data area. The clock speed of the timer chip is normally 1.1931817 megahertz, resulting in a precision of only 55 msec (the interval between successive blocks of 65536 counts). This is not enough for many experiments. The PC time of day clock is based on this same 55 msec interval. Unfortunately, external clocks are not able to easily interrupt the PC CPU [Buhr87]. Segalowitz talks about a hardware clock on an add-in board that will provide millisecond timing [Segal87].

The Intel 8253/8254 timer chip can be programmed to provide higher precision. The problem is that this reprogramming requires significant understanding of the low level workings of the PC. The method described by Graves and Bradley changes channel zero of the Intel 8253 timer chip into mode two [Grav87]. In mode two, the residual count in the timer chip can be read. The total number of timer counts since midnight is calculated by reading the total number of timer interrupts since midnight from the BIOS data area then adding the number of counts that have occurred since the last interrupt. The number of counts since the last interrupt is calculated by reading the residual count from the timer chip then subtracting it from the initial value, which is normally 65536. The number of counts is converted to milliseconds by multiplying by the number of milliseconds per count.

Graves reported one problem with this method [Grav88]. The conversion of the three integer values into one value representing the current time required about 10 msec if a numeric coprocessor is not available. Graves describes a method for using long integers and integer rather than real arithmetic to do the conversion [Grav88]. The conversion time can be reduced to about 2 msec in this way. There is a further problem with this method which is reported by Beringer, Graves and Bovens [Ber92a] [Grav91] [Bovn90]. The problem is that the timer chip may finish the count and start again while interrupts are disabled. The interrupt count read from the BIOS data area will be in error by one which means a timing error of 55 msec. The solution proposed in all three papers is to check to see if the timer count is in the upper part of the range. If so, the BIOS data area is checked again after interrupts have been enabled then disabled again. If the interrupt count has been incremented, the higher value is used.

Crosbie also puts the timer chip into mode two in order to read the count between timer interrupts [Cros89]. Rather than reading the timer interrupt count from the BIOS data area, Crosbie modifies the 1cH (timer tick) interrupt handler to keep a count of the timer interrupts during the timing interval. This scheme has the same problem that Graves' does in that a timer interrupt may be delayed while interrupts are disabled and the program is reading the timer count from the timer chip [Grav87]. As a result, the timer interrupt count could be one short.

Emerson describes a slightly different approach to measuring the time between timer interrupts [Emer88b]. His scheme leaves the timer chip in mode three. In mode three the timer counts down from 65536 by twos. This is done twice per interrupt. A timer status bit available on the 8254 timer chip indicates whether or not the timer is in the first or second countdown. The residual count is converted to an elapsed count, divided by two, then adjusted by a full half count if the status bit indicates that the second countdown was underway when the timer chip was read. This method does not work on machines with the 8253 timer chip, which includes the original PC and the XT. AT class and later machines used the 8254 or compatible chip, which does have the status bit.

Buhrer, Sparrer and Weitkunat describe a different method for getting millisecond timing on the PC [Buhr87]. The counter on the timer chip is set to 1193. This results in a timer interrupt every .9998 msec. The interrupt service routine for the timer interrupt is modified to count the timer interrupts. Elapsed time from initialization can be found by checking a variable. The major problem with this system is that a timer interrupt must be serviced every millisecond. This could impact on the performance of the PC. This scheme also will invalidate the normal PC time of day clock. Dlhopolsky keeps track of the elapsed time during the period when the timer interrupt frequency is altered, then restores the correct time of day when his timing routines are finished [Dlhop88]. Creeger uses the same fundamental method but modifies the timer interrupt service routine so that the system time of day handling is not disturbed [Creeg90]. Emerson uses a scheme where the CPU is interrupted by a serial port at intervals that can be set by the user [Emer88a]. Both Emerson and Creeger have the same CPU overhead problem but don't disrupt the normal PC time keeping. With faster systems, handling a timer interrupt every millisecond is less of a problem than it was with the original PC.

Dorfman outlines a third method that does not require a timer chip [Dorf87]. He calls it a software clock. The idea is that time intervals can be measured by executing a known sequence of instructions. He also documents the use of the WAIT subroutine on the Apple II. The problem with this method is that the processor cannot be performing another task while the clock is running. Another problem is that the time taken to execute a known sequence of instructions will change if either the clock speed or the instruction set of the CPU changes. This is an old idea that has never worked very well.

All of these schemes depend on a well behaved environment. Normally, under DOS the timer interrupt has the highest priority. Therefore, the timer interrupt timing will not be affected by keyboard, disk, serial port, parallel port, network card or other peripheral interrupt activity. The serial port interrupt usually comes after the timer, keyboard and two other interrupts, one of which is often used for a network card. It is important that the programming of the timer chip itself not be corrupted. Because DOS is a single task operating system which generally doesn't alter the programming of the timer chip, this is not usually a problem. There are, however, Terminate and Stay Resident programs that affect the timer. Use of one of these TSRs in conjunction with one of the timing schemes would likely lead to incorrect times. All of the schemes that involve the 8253/8254 or compatible timer chip won't work if an incompatible timer chip is used. Compatibility of this type tends to be less of an issue with the passage of time because the PC industry demands a very high degree of hardware interface standardization.

Most of the papers in *Behavior Research Methods, Instruments, & Computers* that relate to PC timing precision are specific to one particular language environment. Each of the papers on timing from *Behavior Research Methods, Instruments, & Computers* that was studied used one of Basic, C or Pascal as the main programming environment. From a technical standpoint, assembler is probably best for applications requiring millisecond timing. This is because assembler can be written to run more quickly than code created by compilers or interpreters. The programmer is also in a better position to control the timing of an assembler program because the actual machine cycles can be counted. (This is more difficult if there is a cache.) Unfortunately assembler is not an appropriate development tool for most psychology labs. Where assembler is used it is usually linked to one of the other programming environments. Graves and Bradley, as well as Buhrer, Sparrer and Weitkunat, use assembler linked to Basic [Grav87] [Buhr87]. More commonly, a high level language is used for all of the programming. Emerson uses Turbo C [Emer88b]. Granaas provides a version of Emerson's program converted to Microsoft C [Gran89]. Crosbie as well as Bovens and Brysbaert use Turbo Pascal [Cros89] [Bovn90]. Dlhopolsky uses C [Dlhop88]. Creeger, Miller and Parades are exceptions to the rule that only a single language is supported. Their timing routines are written in assembler but they provide interfaces to C, Fortran, Basic and Pascal [Creeg90].

4.3 Calibration

The overall accuracy of the timer chip is dependent on the accuracy of the oscillator or crystal that is providing the clock frequency. Five PC compatible machines were tested over an interval of between twelve and fourteen hours to see how accurately they measured time. The results are summarized in Table 4.1. Timing was done using a Seiko quartz wristwatch that had an accuracy within one second per week. The time indicated in the Set column of Table 4.1 is the time when each machine was set. The time in the Check column is the time that was shown by the machine when it was checked at the end of the test. The datum time was the time shown on the wrist watch at the end of the test. The Elapsed column is the total duration of the test. The Err column is the error which is the difference between the duration measured by the machine and the duration measured by the wrist watch. The Relative column is the relative error (the Err column divided by the Elapsed column). In the first test, the machines were left powered up. This procedure measures the accuracy of the system timer chip. The five machines were set on the evening of May 1, 1993 and checked on the afternoon of May 2, 1993.

Table 4.1 System Timer Calibration

Machine	Set	Check	Datum	Elapsed	Err	Relative
IBM-XT	22:14:30	12:24:44	12:24:45	14:10:14	1	.0000196
PCD-286	22:18:50	12:28:47	12:28:15	14:09:57	32	.0006275
BIOS-286	22:21:18	12:30:10	12:30:20	14:08:52	10	.0003534
386-33	22:15:50	12:25:34	12:25:35	14:09:44	1	.0000196
486-33	22:18:50	12:25:52	12:26:45	14:07:02	53	.0010429

In the second test, the machines were left powered down. This procedure measures the accuracy of the real time battery powered clock. All the machines were set on the evening of May 2, 1993 and checked on the morning of May 3, 1993. The IBM-XT used an Everex EV-170 Magic I/O Input/Output Adapter which incorporates a real time clock. The other machines used the industry standard real time clock which is based on the Motorola MC146818A.

Table 4.2 Real Time Clock Calibration

Machine	Set	Check	Datum	Elapsed	Err	Relative
IBM-XT	22:13:30	10:46:45	10:46:45	12:33:15	0	.0000000
PCD-286	22:07:10	10:44:39	10:44:45	12:37:29	6	.0001320
BIOS-286	22:09:30	10:42:51	10:43:00	12:33:21	9	.0001991
386-33	22:12:00	10:45:52	10:46:00	12:33:52	8	.0001769
486-33	22:11:30	10:42:26	10:41:30	12:30:56	56	.0012429

The magnitude of the relative errors found in the testing shows that the normal timing apparatus on the PC is accurate enough for most experiments which use timing intervals of short duration.

4.4 Timing Output Delays

When presenting a stimulus to a subject, there will be a delay between the decision to present the stimulus and the actual occurrence of the stimulus. For visual stimuli, the first limiting factor is the scan rate of the monitor. The most common scan rate is 60 hz. This means that a stimulus must appear for at least 16.7 msec. It also means that there will be a delay in the display of the stimulus. For stimuli that occupy a small part of the screen, the delay will be randomly distributed between 0 and 16.7 msec. For a stimulus occupying a major part of the screen, the stimulus will appear over a period of time. The first part of the stimulus would appear at some point on the screen. This would be followed by the gradual appearance of the part of the stimulus below the initial part on the screen. The part of the stimulus above the initial point would appear last. This entire process would take 16.7 msec. As described by Dlhopolsky, the transfer of the stimulus to video memory can usually be done during the vertical retrace [Dlhop89]. For a small stimulus, the time at which it appears on the screen can be established quite precisely using the vertical retrace bit and the known scan frequency. For a large stimulus, it will at least appear continuously from the top down using this method.

As long as the stimulus is text based or is reasonably simple, there is no problem in moving it to the video memory during the vertical retrace, which takes approximately 1 msec. The data bus between the host and the graphics adapter is eight bits wide. Each MOVSB instruction on a 386-33 moves 1 byte and takes 210 nanoseconds. Approximately 4750 bytes can be moved during the vertical retrace. An alternative that is reported by Segalowitz is to use the memory on the display adapter [Segal87]. Many adapters can store more than one screen of data. These screens are called display pages. The active display page, the one that will be displayed, is chosen using a BIOS call. A Hercules card supports only one page in text mode but supports two in graphics mode. A VGA adapter can support eight pages in text mode. In the sixteen colour 640 X 350 mode, the VGA adapter supports two display pages. The sixteen colour 640 X 480 mode supports only one display page [Ferr88]. A sixteen colour 640 X 480 image requires 153,600 bytes of memory and will require about 32 msec to transfer from host to video memory. This cannot be done during the vertical retrace. In fact, it cannot even be done during a single screen refresh cycle.

In manipulating a graphic image on the screen it may not be necessary to modify every pixel each time the image is changed. If a small enough subset of pixels is being changed each time, it is possible that the changes can be made during the vertical retrace. Another possibility is the use of a colour lookup table. Under suitable conditions, all of the colours on the screen can be changed during the vertical retrace by modifying the values in the table [Shoup79].

With audible stimuli, the timing is not as critical. Digital to Analog (D/A) boards are capable of supporting sampling frequencies high enough for the human ear. 10,000 samples per second is commonly used although some applications require higher sampling rates. The timing problem comes down to synchronizing the start of the timer to the start of the audio. This can easily be done if assembly language is being used. With a higher level language, access to special hardware features is usually through function calls and these may involve extra overhead that makes it more difficult to guarantee that timing constraints are satisfied. In particular, with assembly language it is possible to issue the I/O instructions very close together if this helps to synchronize, whereas library routines may make it hard to determine the relative times that the I/O instructions are actually executed. Many D/A boards allow the audio to be initiated by an external clock or an external trigger.

4.6 Timing Input Delays

Once a subject has sensed a stimulus, some sort of response is generally required. From the time that the subject senses the stimulus until the CPU records the response, there will be a series of delays.

According to Posner, "[t]he delay between the occurrence of a stimulus event and the initiation of a response to it is called the reaction time." [Posn67]. As reported by Klemmer in Posner's book, the simple reaction time for humans varies between 150 and 250 msec depending on the degree of temporal uncertainty in the timing of the stimulus [Posn67]. In the extreme case, where the period between the warning signal and the stimulus was both very long and highly variable, the reaction times were in the 700 to 800 msec range. Other studies reported by Posner show variations in reaction time due to the amount of information that the subject must process [Posn67]. The reaction times were always between 150 and 800 msec.

In the Headturn situation, reaction times are likely to be near the upper limit because the experimenter has no warning, the timing of the infant head turns is unpredictable and the infant response is often ambiguous. To this must be added the reaction and movement time of the infant. As a result, reaction time information gathered during Headturn testing of infants is not very precise.

Once an appropriate response has been selected, there is a delay while the subject executes the response. This is called the movement time. Work done by Fitts and Peterson as reported by Posner showed that movement time for simple movements depends on the index of difficulty, which is a function of both the amplitude and precision required in the movement [Posn67]. Movement times vary from 100 to 700 msec. The movement required by the experimenter operating the Headturn system is very small and requires almost no precision. As a result, movement time for the experiments would likely be near the bottom of the range in Headturn. The timing requirements in Headturn are primarily related to the presentation of stimuli (sound tokens) and the automatic activation of the reinforcers.

Response times for the infant are recorded to facilitate the culling of trials that are suspect because of unusually long response times.

Once the input device has been activated by the subject, there will be a delay before the CPU in the PC is aware of the input. The timer cannot be read until the CPU has been alerted. Most input devices interrupt the CPU. The interrupt handler for the particular input device can be programmed to record the time when this happens. If an input device does not generate an interrupt, it must be polled at periodic intervals. This can be done by modifying the interrupt handler for the timer interrupt to check the state of the particular input device. When a state change is detected, the time is recorded. The precision of the timing can be controlled by varying the frequency of the timer interrupts and thus the rate at which polling takes place.

4.6.1 The Keyboard

The keyboard is the most common input device. It has the advantage of near universal availability and a high degree of familiarity for both lab personnel and experiment subjects. The major disadvantage of the keyboard is the significant delay between a key press and the corresponding CPU interrupt. The size and weight of the keyboard are also a disadvantage in some situations and the keyboard is not suitable as a graphics pointing device. The key press delay starts with the key switch scan. PC keyboards contain a microprocessor, generally from the Intel 8048 family. The processor scans the key switches. When a key is pressed, the processor sends the appropriate scan code, called the make code, to the PC. When a key is released, a scan code, called the break code (equal to 080H plus the make code), is sent to the PC.

The scanning time will depend on the processor clock speed and the number of instructions that must be executed for each iteration of the scan. Processing time is required to check the interface with the PC and to send the scan code. The interface with the PC is a serial interface using TTL signals. According to Shearer, it takes 1 msec to transmit each byte [Shear88]. As shown in Appendix E, the BIOS code will take between ten and fifty microseconds to process the key stroke and store the code in the buffer. Graves reported an average delay for an IBM PC keyboard of 18.4 msec with a standard deviation of 4.3 msec [Grav87]. A third party keyboard showed an average delay of 36.7 msec with a standard deviation of 2.9 msec. Segalowitz reported a systematic delay of 10 msec and a variable delay which varies between 0 and 15 msec [Segal90]. Keytronics Corporation, a major producer of PC compatible keyboards, verbally claimed a delay of between 1 and 5 msec.

4.6.2 The Mouse

The mouse is the second most common input device. The majority of mice are attached to a serial port. The major delay in registering serial mouse button presses is the length of time necessary to transfer the information over the serial link. The Mouse Systems mouse sends five bytes of information [Segal90]. The Microsoft mouse sends three bytes of information [Ber92b]. At a transfer rate of 1200 baud, the delay for the Mouse Systems mouse would be 41.7 msec assuming one start, one stop, one parity and seven data bits. For the Microsoft mouse, the delay would be 25.0 msec. If the transfer rate is increased to 9600 baud the delay times will be reduced to 5.2 and 3.1 msec respectively. The mouse delays do not have the variability that the keyboard delays do. Beringer reported the range of the delays to be 1 msec or less [Ber92b].

With a bus mouse, the limiting factor in the delays is the interrupt rate. A bus mouse connects to its own special card, which fits into the PC expansion bus. The rate at which the card interrupts the CPU is variable. If a key press is of shorter duration than the inter-interrupt time, it may not be registered at all. For key presses longer than the inter-interrupt time, the range of delays before registration will be randomly distributed between 0 and the inter-interrupt time [Ber92b]. The default interrupt rate is 30 hz. This results in a mean delay of 16.5 msec. Crosbie recommends setting the interrupt rate to 200 hz which results in a mean delay of 2.5 msec [Cros90]. The resulting overhead to process the additional interrupts will have a negative effect on the CPU's processing capability. This is less of a problem with faster CPUs.

In addition to registering button presses, a mouse detects movement. This movement information is transmitted to the PC where it can be used to control a visual cursor. The detection of movement and the transmission of the movement information by the mouse affects the delay times in the detection of the mouse button presses. Delays will be longer and more variable. If a mouse is to be used strictly to transmit button presses, it must be modified so that no movement information is detected or transmitted. This can be done by removing the mouse ball or fixing the mouse so that it cannot be moved.

4.6.3 Other Serial Devices

Other input devices can be connected to the PC serial port. The button box that is used with the current Headturn system is an example. A change in signal level of one of the modem status lines can be detected by the serial port hardware and an interrupt request generated within microseconds. The major problem is that a special purpose hardware device, like the button box used in Headturn, is required for connection to the serial port. Special purpose hardware is expensive and very often not too reliable.

4.6.4 The Parallel Port

The PC parallel port can also be used for input. Creeger describes such a scheme [Creeg90]. It involves the four control lines in the parallel interface that are controlled by the PC. They are one (Strobe), fourteen (Auto Line Feed), sixteen (Initialize Printer) and seventeen (Select Input). Switches are wired to allow each of these pins to be shorted to ground. The status of these pins can be polled by checking the control port at the I/O address corresponding to the parallel port. The polling rate can be adjusted to allow acceptable input detection delays. Creeger published a circuit diagram that included debouncing. He reported a constant delay of 7 msec in recording switch presses [Creeg90]. Debouncing helps to ensure that one switch press will not be detected as multiple switch presses. Dalrymple-Alford describes a similar scheme [Dalry92]. He doesn't include switch debouncing in his circuit but demonstrates how more than four switches can be used with a single parallel port. This is done by connecting some of the switches to more than one of the pins. Using the parallel port in this way results in the same problems as the connection of special input devices to the serial port does. Special purpose hardware is generally more expensive, less reliable and less available than mass market hardware.

4.6.5 Special Purpose I/O Boards

Input can also be done using special purpose boards that install in the PC expansion bus. For example, the Data Translation 2801A board allows the attachment of an external clock or an external trigger which can time events very precisely. The digital I/O feature of this board can also be used in a manner similar to the parallel port. A bit in the digital I/O space of the board can be set. A switch is connected to the corresponding physical I/O line. When the switch is open the bit will remain high. When the switch is closed, the line will be grounded and the bit will be pulled low. The status of each bit in the digital I/O space can be polled so that the state of the corresponding switch can be determined. The DT2801A has sixteen digital I/O lines which would make this a reasonable solution if large numbers of inputs were required. The problem of constructing a custom-built, special purpose switch box remains, however.

4.6.6 The Game Port and Joystick

A third mass market input device is the joystick, which is normally attached to a game port. Joysticks are most commonly used as input devices for video games. As a result they are designed to support an asynchronous real time interface. Both Graves and Segalowitz reported input timing delays, for the joystick, on the order of 1 msec [Grav87] [Segal90]. Segalowitz reported that a second response could be detected within 2 msec of the first [Segal90]. Joysticks are probably not more widely used in experimental psychology because they tend to be associated with video games. The programming interface to the game port is not widely known among programmers and some PCs do not include a game port. However, the low price, ergonomic design and excellent timing characteristics of the joystick/game port combination rate serious consideration in new systems for conducting experiments. A joystick has most of the advantages of a custom-built button box but a joystick is mass produced and more reliable. This makes it easier for others to duplicate an experimental set up and confirm results.

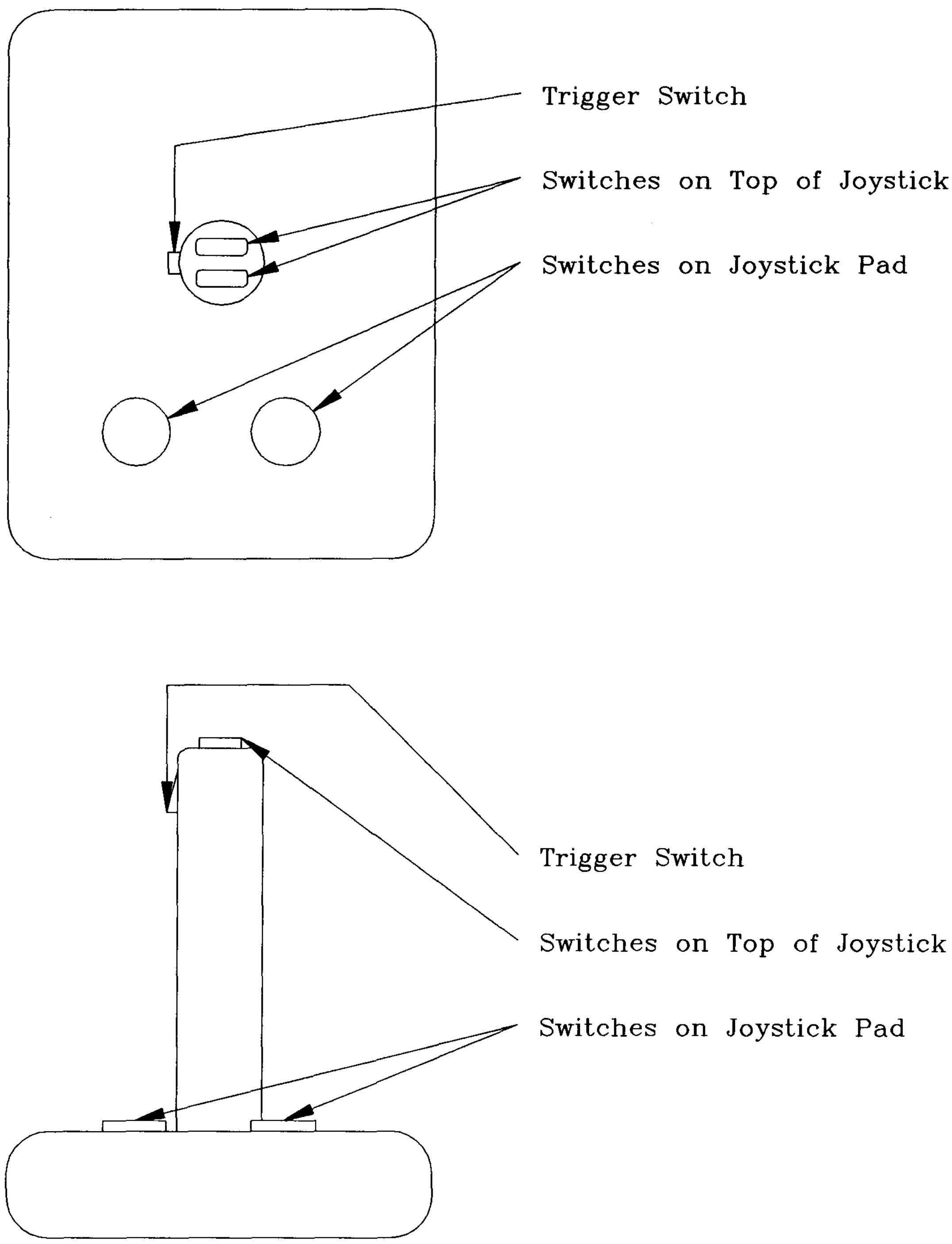


Figure 4.1: Diagram of Joystick

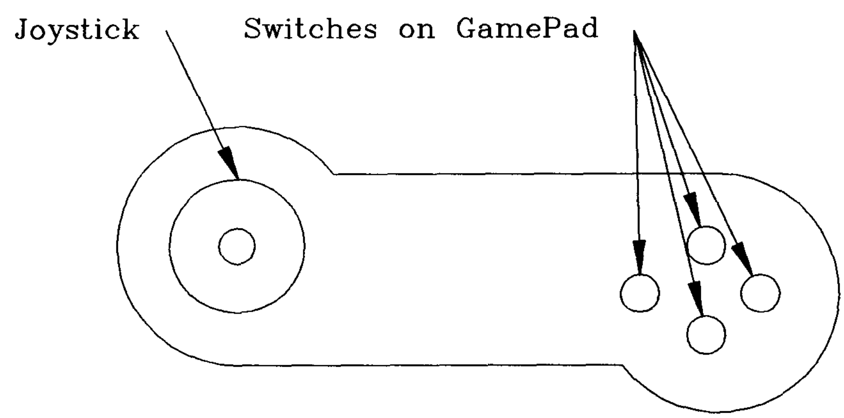


Figure 4.2: Diagram of GamePad

Because the joystick/game port combination looked so promising and because there didn't seem to be too much in the literature about this interface, some testing was done. A Gravis PC Joystick and a Gravis PC GamePad were obtained. Both were tested.

The joystick incorporated a total of five push button switches. The switches were of three types. There were two switches of the first type on the top of the stick itself. There was one switch of the second type on the side of the stick, positioned like a trigger. There were two switches of the third type on the pad which was attached to the stick.

The GamePad had four switches all of the same type. These switches differed from any on the Gravis PC Joystick. There was also provision for a small joystick on the GamePad, but this was not installed.

Two machines were used to do the tests, an 80286 12 Mhz machine and an 80486 33 Mhz machine. The game port data normally occupies I/O address 0201H on a PC machine. Bits four to seven of the I/O port are toggled by the switches on the joystick.

The trigger switch on the joystick toggles the same bit as the first switch on the top of the joystick. When a switch is open (unpressed), its bit is set. When a switch is pressed, its bit is cleared. The game port does not generate an interrupt so an application must poll in order to read the port status. Each of the switches was tested on each of the two machines.

The testing was done using the fragment of 8086 assembler code that is included in Appendix E. The main part of the loop, which is executed continuously until a change in the button state is detected, takes 36 clocks on the 286 and 41 clocks on the 486. The extended part of the loop, which is executed once when a state change is detected, takes 44 clocks for a button down on the 286 and 38 clocks on a 486; it takes 42 clocks for a button up on the 286 and 35 clocks on the 486. At 12 Mhz, the main part of the loop will take 3.00 μsec on the 286 while the 486, executing at 33 Mhz, will take 1.24 μsec; the extended part of the loop for a button down will take 3.67 μsec on the 286 and 1.15 μsec on the 486; the extended part of the loop for a button up will take 3.50 μsec on the 286 and 1.06 μsec on the 486.

The goal of the testing was to discover how quickly state changes in the buttons could be initiated and detected. Two parameters were considered important. One was the minimum length of time necessary to record an actual button press. The second was the maximum length of time necessary for the transition between steady states. The test procedure for each switch on each machine was as follows. First, ten "rapid fire" presses were done. The button was pressed ten times in as short a time interval as possible. Second, ten "quick presses" were done. The button was pressed and released as quickly as possible during each press; between presses, time was allowed for the switch to settle.

Third, ten "slow presses" were done. Time was allowed in both the up and the down state for the switch to settle.

The results of the testing are recorded in Appendix D. The first observation is that every button press was detected. The manual count of the button presses shows the same number as the records made by the software. The second observation is that the difference in timing granularity on the two different machines resulted in minimal differences in the results. The only area where there was an obvious difference was in the number of cycles recorded. The number of cycles is a measure of either bouncing or arcing in the switches. The difference is most apparent in the results for the switches on the joystick pad. The mean number of cycles measured using the 80286-12 was fourteen while the mean number of cycles measured using the 80486-33 was twenty-one. It's likely that this difference is due to the higher clock speed of the 80486-33. The switch press profiles displayed in Appendix D were plotted from the means measured using the 80486-33 and represent typical switch press profiles for the different types of switches.

When measuring the minimum time intervals in which a button press could occur, the means were mostly in the 55 to 65 msec range. Minimums ranged from 14 msec for the joystick switches to 70 msec for the trigger switch on the joystick. In selecting a value to use as the interval between polls of the game port, half the value of the smallest minimum seemed to be a good starting point. This would result in an interrupt every 7 msec to minimize the chance that a button press would be missed.

It is possible that the these times were a reflection of the motor performance of the person pressing the switches rather than the ultimate performance capabilities of the switches themselves. To eliminate the variability of the human motor performance in testing the switches, a variable speed drill driving a cam could be used to press the switches. In many cases during transition times, the game port recorded two state changes within 6 μsec. This is as fast as possible given the CPU clock speeds. This suggests that the game port is very sensitive to the switch state. Changes in the state of the switch will be reflected in the state of the game port within microseconds. This confirms previous claims that the game port could respond in less than a millisecond. The major timing limitation on the game port would appear to be the physical switch itself.

A transition was measured as the time between two steady states. A steady state was defined as a period of at least 10 msec in which no state changes were recorded. A transition could be from an up to a down state or from a down to an up state. The maximum time necessary to make a transition from one state to the other varied from 6.90 msec for the trigger switch on the joystick to 0.19 msec for the switches on the top of the joystick. The trigger switch and the switches on the joystick pad were obviously inferior performers compared to the switches on top of the joystick or on the GamePad. The maximum state transition time for the trigger switch was between 6 and 7 msec. For the switches on the joystick pad, it was between 1 and 2 msec. The corresponding figures for the switches on top of the joystick and on the GamePad were 0.19 and 0.31 msec respectively. As a result, the trigger switch and the switches on the joystick pad were eliminated from further consideration. For the switches on the GamePad, the standard deviation in the state change time was 0.027 msec. Adding two standard deviations to the maximum and doubling yields 0.73 msec. For the switches on the top of the joystick, the standard deviation in the state change time was 0.017 msec. Adding two standard deviations to the maximum and doubling yields 0.45 msec. If the timer interrupts happen at 1 msec intervals, both of these types of switches could be used with little chance that one button press would be detected as two.

A third limitation on the frequency of the timer interrupts is the precision required by the experiment. Based on the times given at the start of this chapter, interrupts every 1 msec would give sufficient precision. Interrupts at 7 msec seconds would not be frequent enough for the more demanding measurements.

To see if it was feasible to have a timer interrupt every 1 to 7 msec, an investigation was done of the time necessary to service a timer interrupt on various machines. The fragment of assembler code shown in Appendix E represents what would typically be executed by the timer interrupt when there was no change since the previous timer interrupt. This will be the situation for the vast majority of timer interrupts. It is clear from the results presented in Appendix E that timer interrupts every 1 msec are feasible on all but the 8088 processor machines.

4.7 Timing Coordination

The preceding discussion has investigated the timing of a relatively simple model of experimentation. One subject responds to one stimulus using one input device. Cheaper, more sophisticated hardware and software as well as advances in the field of psychology are going to make more complex experiments possible. Timing coordination is likely to become a more critical issue. For example, with both visible and audible output, output synchronization is important. On the input side, more than one input could be collected from a single subject or inputs from multiple subjects could be collected. A single subject could be making voluntary responses while one or more channels of physiological data are collected. If a single central clock is used to time all of the inputs and outputs, then the delays along each chain between the central CPU and each I/O device must be assessed and if possible controlled. If the timing is done at the point of I/O then there is the problem of synchronizing multiple clocks. The synchronization of multiple independent clocks is still a significant research issue. It may well be that timing problems in experiment systems are going to become more rather than less critical with advances in personal computing and psychology.

Chapter 5

Analysis of Current Headturn Package

The Headturn package was written by Mr. Kurt Goldhardt of Nirvonics Inc. under contract. The project was commissioned by Dr. Janet Werker of UBC and Dr. Anne Fernald of Stanford University. The first version was installed in the summer of 1987. There are currently installations of this Headturn software at UBC, Stanford, McGill and Kiel University in Germany. The software is used primarily to study language capabilities in infants. Werker and Fernald put together a specification for the system which Goldhardt used for the implementation. The model that was used to engineer the Headturn software was one in which the user specified the functionality that was required and the developer created a program that met this specification. There was minimal feedback from the users during the development of the package. Since the initial release there have been three or four upgrades. The last one was not installed due to operational difficulties.

Unfortunately, communication between the developer and the users has diminished to the point that they are no longer in contact. The end result has been a package that meets the users' fundamental requirements fairly well but which is sometimes difficult to use and sometimes has limited flexibility. The package does provide useable results. It supports a large number of experimental parameters which does result in some flexibility. There are, however, serious shortcomings in the package in part because of the narrow focus of the engineering model that was used.

5.1 The User Interface

The first of these deficiencies is in the user interface. The system is very difficult to install. Even though it is a package intended for personal computers, it has a very strong UNIX orientation. Most personal computer experts know little about the UNIX system and lack the mind set that facilitates working under UNIX. Although Headturn includes an automatic installation procedure, an understanding of the software is required to do a successful installation. This is because the system is installed as a series of semi-independent modules. The installation procedure requires the user to answer a number of questions. The answers to some of the questions are dependent on the answers that were given when other modules were installed. A terminal configuration is also required. This is an alien concept in the personal computer world. Headturn is also difficult to operate. The experimenter is required to use a text editor to edit two different text files. One contains the list of names of the files that contain the digitized sounds, which are called tokens. The other contains the parameters needed to operate the Headturn system. There is minimal feedback to the experimenter while the system is running.

5.2 Functional Flexibility

The second serious deficiency in the system is flexibility. As a result of accumulated user experience and advances in the field, the users now have a list of operations that they would like to perform, but which aren't possible with the current Headturn software. These include the following.

1. More flexibility in conditioning and training. The training and conditioning stages are the ones that teach the infant to turn its head when it detects a change in the stimulus. Currently each of the training and conditioning stages allows only one particular type of trial and allows only one background and one target token. More flexibility would allow more training and/or conditioning stages and allow the use of more than one background or target stimulus in these stages.
2. An option to do automatic retraining after some number of consecutive misses. Currently, the experimenter must initiate each retraining trial manually. Retraining generally involves using background and target stimuli with a large difference between them. The purpose of a retraining trial is to re-emphasize, to the infant, the connection between the sound difference and the visual reinforcement.
3. Correction of a problem in counting misses. The current program counts false alarms wrongly as misses and displays a message after three consecutive misses. A false alarm happens when the infant turns its head during a control trial, which is a trial with no change in stimulus. A miss happens when the infant fails to turn its head during a change trial, which is a trial with a change in stimulus.
4. An option to perform an experiment session according to a preset schedule of specified stages and stimuli.

This list seems minor from a programming point of view, but is very important to the users.

5.3 The Engineering Model

An engineering model for the development that involved user testing of the package before it was completed and user feedback to the developer during the development process likely would have resulted in a better package because of better communication between the developer and the users. The specification provided by the users was essentially a functional specification. A user review of the developer's design for the package before implementation might have headed off some of the current problems. User testing of the partially completed package at specified milestones during the implementation almost certainly would have uncovered many of the current problems at a point where they could have been fixed more easily.

5.4 File Formats

Another area where Headturn seriously lacks flexibility is in the digitized sound files that it accepts as input. Only one file format is supported. This format appears to be unique to Headturn. Files in this format must be produced using the digitizing tools supplied with Headturn. No conversion program is provided to allow the use of any of the more common sound file formats with Headturn or to allow the use of the Headturn files with any other hearing research tools. The format of the files required by Headturn is not well documented. It has been possible to put together a conversion program that allows files in another format to be converted to the Headturn format, but this was tedious.

Awareness by the users of the importance of standards and insistence on a well known file format could have avoided this particular problem. Formats usually differ only in their headers (the information that precedes the digital samples) and in a few parameters such as sampling rate and the number of bits per sample.

5.5 The Development Process

There are several reasons why the package was developed in the way that it was. It is very difficult for most people to specify in advance exactly what they want in a software package. But from the point of view of the developer, it is much easier to write the system to meet a specification and to not worry about changes along the way. Unfortunately, the cost of any change tends to be higher if the change is done later in the process. In a project of this type, which was financed by only two users, the developer cannot afford to spend a great deal of time engineering the user interface. The developer was known to have worked extensively in UNIX before, so apparently the easiest thing to do was to put a simple UNIX-style interface on the Headturn package. Without user exposure to pre-release versions of the package and without feedback to the developer, the users were unaware of what they were getting by way of an interface and the developer was probably unaware that this particular interface was not very suitable.

There are also shortcomings in the Headturn package that are apparently due to poor hardware engineering.

5.6 The Button Box

The button box control system is not reliable. The system was examined in detail and an attempt was made to modify the button box system to produce an acceptable level of reliability.

5.6.1 The Original Design

It is important to understand the reasons for using this particular interface rather than the mouse or the keyboard, which are the two most popular devices for PC input. Both the mouse and the keyboard exhibit significant delays between the time that the button or key is pressed and the recognition of the press by the CPU. These delays are discussed in Chapter 4. Direct access to the serial port hardware provided a way of circumventing these delays. As described in the MS-DOS Encyclopedia, the INS8250 UART chip that provides the serial port interface on the PC has an 8-bit modem status register [Dunc88]. The upper 4 bits indicate the voltage levels of the associated RS232C lines. The lower 4 bits indicate whether the voltage levels have changed since the register was last read.

Table 5.1

8250 Modem Status Register Bit Values		
Bit	Binary	Meaning
7	1xxxxxxx	Data Carrier Detect (DCD) level
6	x1xxxxxx	Ring Indicator (RI) level
5	xx1xxxxx	Data Set Ready (DSR) level
4	xxx1xxxx	Clear To Send (CTS) level
3	xxxx1xxx	DCD change
2	xxxxx1xx	RI change
1	xxxxxx1x	DSR change
0	xxxxxxx1	CTS change

The RS232C interface standard specifies that when a voltage of between −3 and −15 volts exists on a line, a logic 1 will be detected and when a voltage of between +3 and +15 volts exists on a line, a logic 0 will be detected. The RS232C interface standard also specifies that to assert a logic 1 a voltage of between −5 and −15 volts is needed and to assert a logic 0 a voltage of between +5 and +15 is needed. The 2 volt difference between the send and receive values is a noise margin to allow for signal degradation over the maximum distance which is 50 feet. The response time of the 8250 chip is much less than 1 msec.

The 8250 chip also provides a modem control register.

Table 5.2

8250 Modem Control Register Bit Values		
Bit	Binary	Meaning
4	xxx1xxxx	Turns on UART self-test configuration
3	xxxx1xxx	Controls INS8250 interrupt signals
2	xxxxx1xx	Resets Hayes 1200b internal modem
1	xxxxxx1x	Sets RTS output
0	xxxxxxx1	Sets DTR output

It is possible to construct an interface that controls a program by changing the status of the four modem status lines. Emerson outlines a serial port interface which does exactly this [Emer88a]. Emerson's interface also does millisecond timing. Emerson chose to use battery power to supply the voltages for the modem status lines. With Headturn, the voltage for the modem status lines was to be provided by the two modem control lines (RTS and DTR). The RTS line was also going to be used by Headturn to power the ready light, which indicated a certain Headturn program state.

Unfortunately, the button box as originally specified was not functional. The schematic for this circuit is included in Appendix A as part of the system specification supplied by the developer. The obvious problem is that the RTS line cannot be used to provide the positive voltage for the three modem status lines and also be toggled to turn the ready light on and off. The 10K resistors are a further problem. They are not the correct size to get the appropriate voltages on the modem status lines. These problems were discovered by the users when the system was first installed.

5.6.2 The Improved Button Box

An improved button box design was done by Brian Moorhead, an electronics technician in the Department of Psychology at UBC, when Headturn was first installed. A schematic diagram of his improved button box circuit is shown in Figure 5.1. The problem of using the RTS line for two purposes was avoided by connecting the three modem status lines to the signal ground. This new configuration still has a potential problem. The RS232C interface defines voltages between −3 and +3 to represent an undefined state, which may result when a line is grounded. Another potential problem with this button box is caused by the connection of the RTS line to the DTR line through the ready light. When the ready light is on, the voltage on the DTR line is raised far enough to put it into the undefined range. This can be seen in the voltage readings that were done when the ready light was on while testing this button box, as explained later in this section. It might have been better to connect the ready light between the RTS line and the signal ground. It is worth noting, however, that this button box has worked pretty well most of the time.

A test button box was constructed using the original drawing. As predicted, that circuit was unworkable so the test button box was reconstructed using the improved design. The button box has three buttons, named "Begin", "Vote" and "Abort". There is also an LED which is called "Ready". It is likely that the Headturn software monitors the state of the three button switches by monitoring the modem status register bit values for the Data Carrier Detect, Data Set Ready and Clear to Send lines. Without the source code, it is difficult to be certain. During the operation of the button box, the Data Terminal Ready (DTR) is kept high (logic 1). When a switch is closed, the associated modem status line is connected directly to the signal ground. This would appear to be producing a logic 0 voltage on the line. The corresponding modem status register level bit would then be cleared. When a switch is opened, the connection to the Data Terminal Ready line will lower the voltage level on the associated modem control line to a logic 1 state. The corresponding modem status register level bit would be set. Whether the Headturn software reads only the level bits for the DCD, DSR and CTS lines or whether the software also monitors the change bits for these three lines is not clear. To illuminate the ready LED, the Request to Send line is set to logic 0.

A button box, built to the improved design, was tested using the button box test program that is supplied with Headturn. Very often, the test program indicated that one of the switches was being opened and closed very rapidly when it was not. The problem was more severe on the Compaq machine in the Infant Research Laboratory than it was on any of the four test machines. It was hoped that perhaps a change of serial ports would alleviate the problems that were being encountered in the lab. To this end, the box was tested with the four test machines to see if any correlations could be found between button box performance and the serial port voltages. The voltages were measured across the button switches when they were pressed (open). Table 5.3 summarizes the voltages that were measured. Where continuous state changes were noted, they continued throughout the time the switch was pressed. This seems to suggest that the state changes were not caused by either switch bounce or switch arcing. The following was observed.

1. PC Designs 286

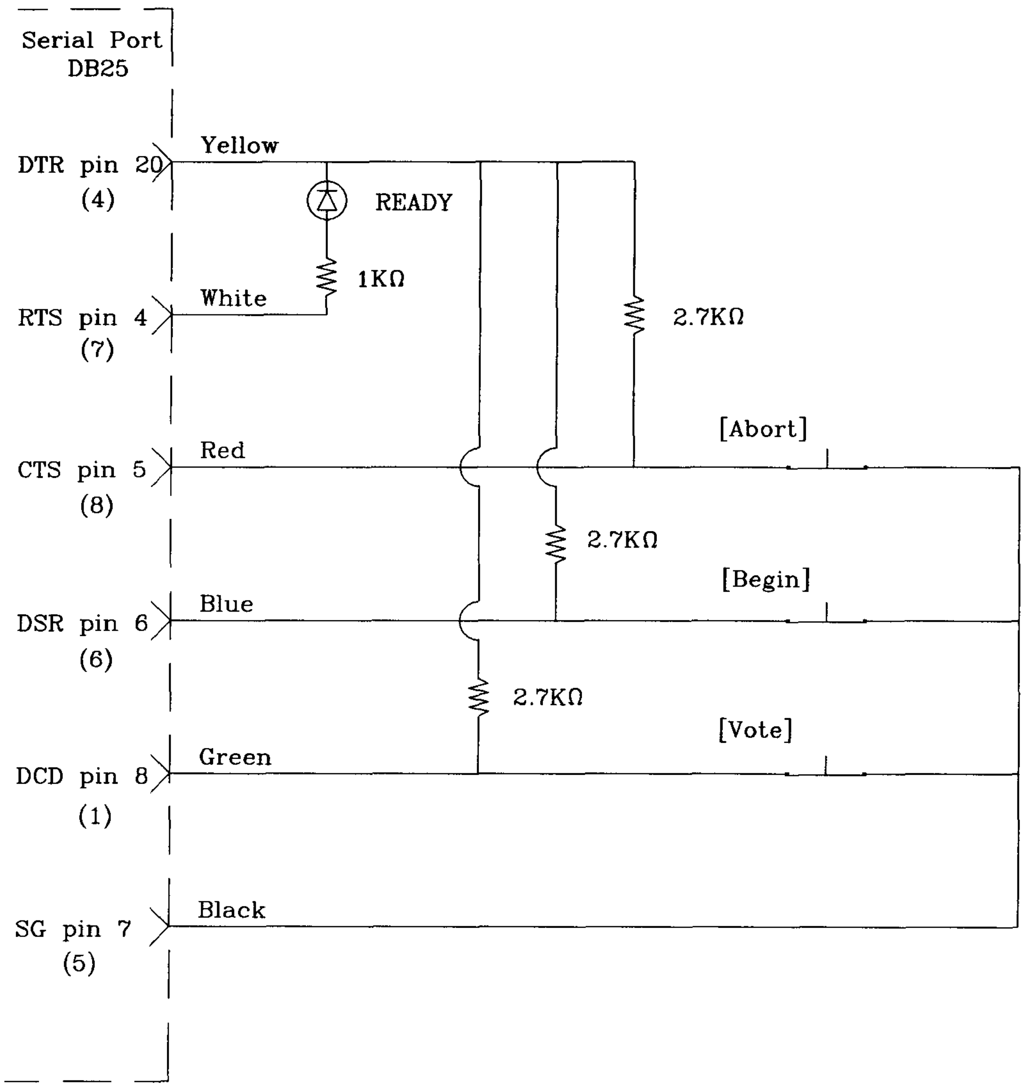
   The Abort button registered multiple state changes when pressed. The rate of multiple state changes was faster when the ready light was off. The Begin and Vote buttons worked correctly.

2. Generic 486 Clone

   The Abort button occasionally registered multiple state changes when pressed. The Begin and Vote buttons worked correctly.

3. BIOS 286

   The Vote button indicated continuous state changes as long as the ready light was off. When the ready light was on, the Vote button behaved correctly. The Begin and Abort buttons worked correctly.



The numbers in parentheses are the DB9 pin numbers.
The colours are the wire colours for a standard telephone jack box.

Figure 5.1: Improved Button Box Circuit

Table 5.3 Voltages Measured Using Improved Button Box

PC Designs 286 – Voltages		
Button	Ready Light Off	Ready Light On
Begin Vote Abort	–4.08 –4.12 –4.11	–.230 –.227 –.231
Generic 486 – Voltages		
Button	Ready Light Off	Ready Light On
Begin Vote Abort	–4.75 –4.79 –5.46	–0.446 –0.447 –0.538
BIOS 286 – Voltages		
Button	Ready Light Off	Ready Light On
Begin Vote Abort	–5.04 –5.09 –5.08	–1.02 –1.03 –1.02
IBM XT – Voltages		
Button	Ready Light Off	Ready Light On
Begin Vote Abort	–3.78 –3.80 –3.81	–0.036 –0.038 –0.034
Compaq 286 – Voltages		
Button	Ready Light Off	Ready Light On
Begin Vote Abort	–3.68 –3.59 –3.36	–0.730 –0.720 –0.744

4. IBM XT

   The Vote button indicated some multiple state changes when the program was started. The Begin and Abort buttons worked correctly.

5. Compaq 286 in Infant Research Laboratory at UBC

   There have been ongoing problems with the production button box in use in the lab. Contact cleaner spray is used extensively to facilitate the operation. The test button box appeared to produce identical results to the production model.

It is apparent from this investigation that the button box scheme currently in use for controlling the PC is very unreliable. The problems do not seem to stem from weaknesses in any particular button box or serial port. The experience of the lab personnel seems to indicate that there is a correlation between the condition of the button switches and their performance. If the switches are in new condition, the Headturn software can be used without problem. As the switches deteriorate, problems mount. A further problem with the button box is the fact that it has to be custom built for this program. This adds to the expense of getting a Headturn installation up and running and is a significant issue for users.

5.6.3 The Experimental Button Box

An experimental version of the button box was constructed using the design shown in Figure 5.2. The purpose of this experiment was to see if a reliable system could be constructed if the "Ready" light was not required. The button box test program included with Headturn was used to do the testing. The RTS line was kept in a logic 0 state and the

DTR line was kept in a logic 1 state throughout. All voltages were measured relative to signal ground. Table 5.4 summarizes the voltages that were measured. The following was observed.

1. PC Designs 286

   The Begin button registered multiple state changes when the switch was closed. The rapid state changes stopped when the switch was opened or if the voltmeter was attached to the DSR line. The Abort and Vote buttons worked correctly.

2. Generic 486 Clone

   All three buttons worked correctly.

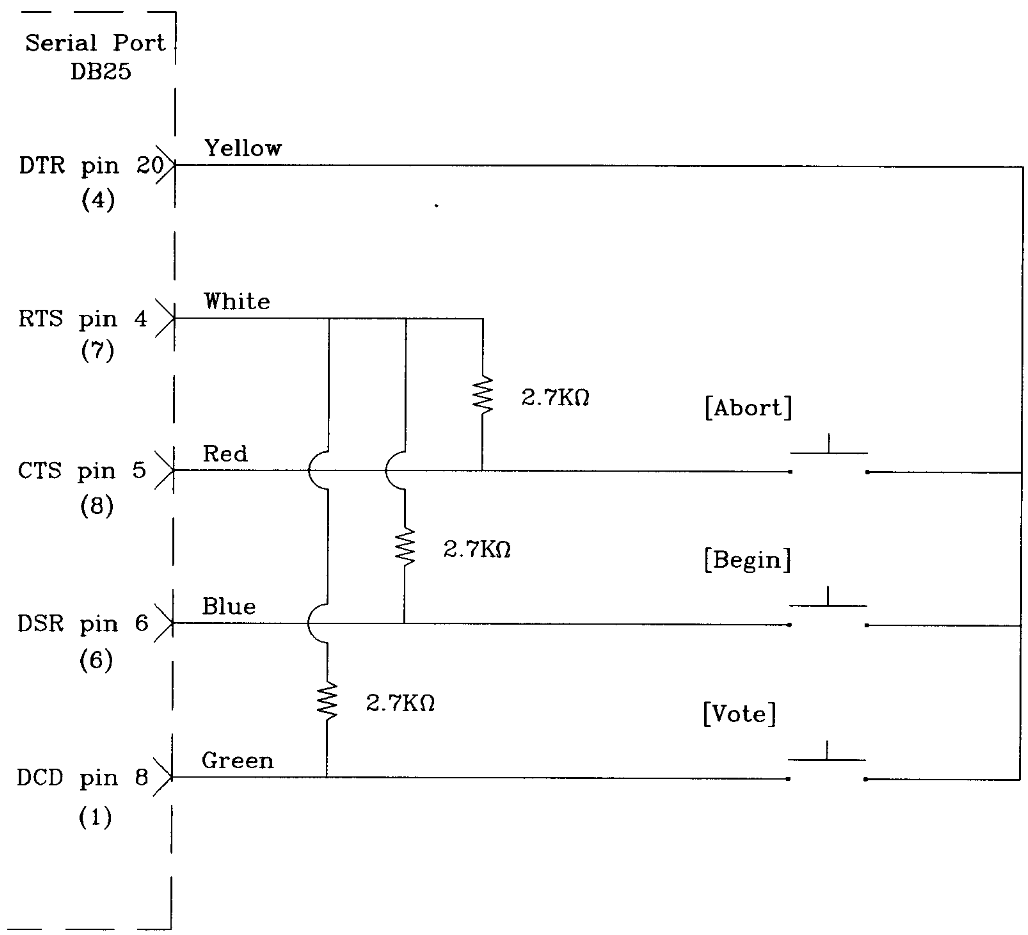
3. BIOS 286

   Both the Begin and Vote buttons indicated continuous state changes under all circumstances. The Abort button worked correctly.

4. IBM XT

   All three buttons worked correctly.

All of the voltages recorded in this experiment fell within the range specified for the RS232C interface. Unfortunately, the button box did not perform correctly with two of the serial ports. The voltages were measured with a voltmeter so perhaps some of the computers were not sensing the RS232C signals correctly due to noisy voltage references or timing problems. When the reliability problems of the button box interface are added to the trouble and expense of constructing special purpose hardware, it is difficult to justify the use of this particular interface in a new system.



The numbers in parentheses are the DB9 pin numbers.
The colours are the wire colours for a standard telephone jack box.

Figure 5.2: Experimental Button Box Circuit

Table 5.4 Voltages Measured Using Experimental Button Box

PC Designs 286 - Voltages		
Button	Switch Closed	Switch Open
Begin(DSR) Vote(DCD) Abort(CTS)	-5.98 -6.04 -5.93	6.07 6.12 6.04
Line	One Closed	Switches Open
DTR RTS	-6.07 6.60	-11.42 9.67
Generic 486 - Voltages		
Button	Switch Closed	Switch Open
Begin(DSR) Vote(DCD) Abort(CTS)	-6.42 -6.82 -6.75	5.91 5.94 5.91
Line	One Closed	Switches Open
DTR RTS	-6.70 7.10	-11.28 9.48
BIOS 286 - Voltages		
Button	Switch Closed	Switch Open
Begin(DSR) Vote(DCD) Abort(CTS)	-8.33 -8.35 -8.35	5.65 5.69 5.69
Line	One Closed	Switches Open
DTR RTS	-8.35 6.58	-11.35 9.35

Table 5.4 is continued on the next page.

Table 5.4 Voltages Measured Using Experimental Button Box (continued)

IBM XT - Voltages		
Button	Switch Closed	Switch Open
Begin(DSR) Vote(DCD) Abort(CTS)	-5.07 -5.05 -5.10	6.03 6.00 6.05
Line	One Closed	Switches Open
DTR RTS	-5.11 7.10	-11.49 9.72

5.7 Processor Speed

The Headturn software seems sensitive to processor speed. On an Intel 286-12 processor, an unwanted trial generally occurs at the start of each stage. This does not happen on an Intel 486-33 processor. However, on the Intel 486-33 processor, the program sometimes goes into an infinite loop playing one of the stimuli over and over again at high speed. This problem was never noticed on the slower processors. These problems could possibly be due to a lack of personal computer development experience on the part of the developer and the specific difficulties of writing time critical programs for the MSDOS operating system.

5.8 Contention in the Data Translation Board

Because the Data Translation board has only one processor, the board can execute only one command at a time. While a digital sound file token is being converted to an analog signal, the board cannot execute another command. The current Headturn package controls the reinforcers using the digital I/O feature of the Data Translation board. As a result, the reinforcers cannot be turned on or off while a stimulus is being presented. For short tokens, this probably does not make a big difference, but it is wrong and should be fixed. A better way to control the reinforcers would be to use a parallel port. The parallel port interface specifies eight lines of TTL signal levels. Reinforcer control requires only four lines of TTL signal levels.

5.9 Conclusions

This Chapter highlights the difficulties for the user in trying to specify exactly what is required without seeing any feedback from the developer while the system is being developed. Providing more flexibility would have resulted in a more useful system.

Because the users do not have the source code to the system and because they are no longer in contact with the developer, most of the problems with the system are not easily addressed. A set of supplementary installation instructions was prepared to assist users in installing the existing system. These supplementary instructions are contained in Appendix C. They are intended to guide the user through the regular instructions. For the remainder of the problems, the only solution appeared to be a rewrite of the software. After consultation with the users, we decided to re-engineer and rewrite the package. The new package was to be developed with feedback from the users based on pre-release versions of the package. It was to be engineered specifically for the personal computer environment. There was to be sufficient flexibility to meet the users' current needs and perhaps some of their future needs.

Chapter 6

Design of the New Software Package

In designing a new software package to implement the Headturn procedure, the existing package was used as the starting point. This was done for two reasons. The existing package has produced useable results and it is familiar to the personnel in the lab. Some degree of compatibility was thus desirable. The new package was intended to generalize the implementation of the Headturn technique. This was done by making all stages and trial protocols both equal and general. An experiment session consists of an arbitrary collection of stages. Stages are configured with arbitrarily chosen trial protocols. Trial protocols are configured with arbitrarily chosen sound files (commonly called tokens). The initial defaults are configured to allow the new package to resemble the old one. In this way, the transition between the two packages is facilitated.

The list of user requests that was presented in Chapter 5 was also addressed in the new package. The generalizing of the program provides the flexibility in conditioning and training that was requested. An option has been added to the generalized notion of a stage to allow automatic retraining after a specified number of misses. A scheduling option has also been added to allow the sequence of change and control trials in a stage to be chosen according to a preset schedule.

Some thought was given to using a general experiment control system to implement Headturn. The Canadian Speech Research Environment (CSRE) experiment control system was evaluated. Unfortunately it did not provide a simple way of controlling the reinforcers. The CSRE system also could not be easily programmed to play sound files continuously.

6.1 The User Interface

In planning a new user interface, the primary principle was that it must be possible for a computer literate research assistant to install and operate the program after no more than one day of training. Installation consists of the creation of an appropriate directory on the hard disk and of the copying of the files from the floppy disk to that directory. A reasonably chosen set of initial defaults allows the program to run usefully "right out of the box". The choice of sound files and the program parameters is done using a configuration program within the Headturn environment.

Generally, the screen interface for the package is text based and keyboard driven. Enough information is presented on the screen to allow a user to deal with most situations without reference to any written documentation. The screens for the package are shown in Appendix F.

For experimenter control of the program, the Gravis four button PC GamePad was chosen. The GamePad was selected for several reasons.

1. It provides four identical switches with characteristics suitable for the Headturn technique.
2. It is small and easily held in one hand or two hands.
3. It is available in retail outlets for less than $100.
4. It is constructed well enough to be used in the video game market and warranted against defects.
5. The serial/parallel communications adapter in a PC often incorporates a game port.
6. Game ports can be used to measure timing intervals with one millisecond precision.

The buttons are named similarly to those on the earlier button box, with one new button introduced. The red button is [Begin]. The blue button is [Vote]. The green button is [Abort]. The yellow button is the new [Pause] button that has been added. The [Abort] button can be used as a shift key in combination with either [Begin] or [Vote]. A [Begin] press shifted by an [Abort] press is indicated as [Abort]-[Begin]. A possible problem with the GamePad is that it requires a game port that supports all four buttons. The very cheapest game ports support only two buttons. This feature can be easily checked using the test program that is supplied by the manufacturer of the GamePad. It was assumed that the price of a four-button game pad was not excessive, so we require it.

When the Headturn program is run the Main Screen appears. There are seven choices on the screen. The program reads its configuration from a file called `headcnfg.dat` from the current directory. If this file cannot be found, a warning message is displayed and the initial defaults are used. The choices displayed on the Main Screen, which is diagrammed on page 160 in Appendix F, are the following.

1. Run Experiment
2. Edit Configuration File
3. Edit General Parameters
4. Edit Session
5. Edit Stage
6. Edit Schedule
7. Edit Protocol & Stimuli

6.2 The Configuration Program

The different parts of the configuration program are accessible from the main screen of the Headturn program.

6.2.1 Edit Configuration File

The complete configuration for an experiment session is stored in a configuration file. The Edit Configuration File option reads, updates, creates and writes these files. Multiple files can be created and saved in any directory of any available disk. This screen is diagrammed on page 163 of Appendix F.

6.2.2 Edit General Parameters

The Edit General Parameters option edits the parameters that apply to an entire session. This screen is diagrammed on page 164 of Appendix F. There are the following parameters.

1. **ID#, Subject Name, Subject Birthday, Experimenter** and **Assistant** document and identify the experiment session.
2. **Response File** is the name of the file where the experiment session, stage and trial protocol parameters as well as the results of each trial are written as a record of the experiment session.
3. **Sequenced Mode** specifies whether the stages are to be run automatically in sequence or if the experimenter must progress manually through the stages.
4. **Trial Screen Colour** and **Between Trial Colour** select screen background colours that will visually identify to the experimenter whether or not a trial is in progress.
5. **Sampling Rate** is the frequency in Hertz at which the original analog signals stored in the sound files were digitized.
6. **Sampling Precision** is the number of bits used to store each sample in the sound files.
7. **File Offset** is the byte position relative to the start of the sound file (location zero) at which the digitized information begins. This permits various header formats to be accommodated.
8. **DMA Channel** is the direct memory access channel that is to be used to communicate with the Data Translation board. The jumpers on the board

must be set to match this value. Normally DMA channel 1 is used unless it is being used by another device in the PC.

9. **I/O Address** is the port address, in the PC I/O space, of the Data Translation board. Normally 2ecH is used unless it is being used by another device in the PC.

6.2.3 Edit Session

The Edit Session option creates the list of stages that comprise an experiment session. Up to fifteen entries may be made in the list. A particular stage can appear more than once in the list. An experiment session can be created and run without saving the configuration in a file. This screen is diagrammed on page 165 of Appendix F.

6.2.4 Edit Stage

The Edit Stage option edits the parameters that apply to an individual stage. A stage is composed of a series of trials. The timing and presentation of the trials is controlled by the user with the GamePad and by the stage parameters. This screen is diagrammed on page 166 of Appendix F. There are the following parameters.

1. **Stage Name** uniquely identifies the stage.
2. **Main Protocol** is the type of trials that are used by the stage.
3. **Retraining Protocol** is the type of trial that is used when the [Abort]-[Vote] combination is pressed.
4. **Schedule Name** specifies the type and ordering of change and control trials. If this field is set to "(random)", the schedule is computed automatically.

The probability of a change trial, the maximum number of consecutive change trials, the maximum number of consecutive control trials and the block size are used to construct the schedule. Schedules are 200 trials long which is the maximum length for a stage.

5. **Probability of Change Trial** is the probability with which the target stimuli are played during a trial. This probability is based on sampling without replacement to decrease the chances of a long series of change or control trials.
6. **Maximum # Successive Change Trials** is the maximum number of consecutive change trials that are allowed in a sequence of trials in a session.
7. **Maximum # Successive Control Trials** is the maximum number of consecutive control trials that are allowed in a sequence of trials in a session.
8. **Block Size** is the number of trials over which the statistical parameters are balanced. Each block will contain the number of change trials specified by the change trial probability.
9. **Minimum # of Trials** specifies the minimum length of the stage in trials.
10. **Criterion** is the number of correct answers that the subject must produce to successfully complete the stage. If 0 is specified, the stage will continue until the [Abort]-[Begin] combination is pressed.
11. **Criterion Window** specifies the number of trials over which the subject must produce the correct answers.
12. **Required # of Criteria** specifies how many times the subject must achieve the criterion to successfully complete the stage.

The last two parameters on the screen handle situations in which the subject is having difficulty.

1. **# of Consecutive Misses Required** is the threshold of difficulty. Only failures to turn during change trials will be counted. False alarms, which are head turns during control trials, will not.
2. **Automatic Retraining?** specifies whether or not an automatic retraining trial is done when the subject reaches the threshold. The computer beeps whenever the subject reaches the threshold whether or not automatic retraining has been specified.

6.2.5 Edit Schedule

The Edit Schedule option creates, updates and deletes schedules of trials. A schedule is a specification for the ordering of change and control trials during a stage. This screen is diagrammed on page 167 of Appendix F.

1. **Schedule Name** uniquely identifies the schedule.
2. **Schedule Body** specifies the ordering of the trials during a stage. An asterisk is used to specify a change trial and a dash is used to specify a control trial. A blank is used to specify that the trial type is to be chosen using the change trial probability.

6.2.5 Edit Protocol

The Edit Protocol screen edits the parameters that apply to an individual trial. A protocol consists of tokens and specifications for playing them as well as specifications for controlling the reinforcers. This screen is diagrammed on page 168 of Appendix F. There are the following parameters.

1. **Protocol Name** uniquely identifies the protocol.
2. **Reinforcer Initiation** specifies whether reinforcers are to be activated unconditionally or when the [Vote] button is pressed. Both may be specified, in which case the reinforcers are activated at the end of the specified delay or when the [Vote] button is pressed, whichever comes first. (See section 5.8.)
3. **Initiation Delay** is the delay in activating the reinforcers when the activation is unconditional. The delay is from the start of the trial. If activation is triggered by the [Vote] button, the reinforcers are activated as soon as the button is pressed.
4. **Reinforcer Duration** is the length of time for which the reinforcers are activated. This parameter controls the number of target stimuli that are played.
5. **Target Stimulus Reps** is the number of times target stimuli are played during a trial. This parameter is used only if there is no reinforcement during the trial.
6. **Vote Button Extension** specifies whether or not the reinforcers are kept turned on as long as the [Vote] button is depressed. The target stimulus is also played as long as the [Vote] button is depressed in this case.
7. **Observation Interval** is the length of time during which a head turn can be recorded with the [Vote] button. The observation interval starts when the trial starts.
8. **Inter Stimulus Interval** is the length of time between the end of one stimulus and the start of the next one.

6.2.7 Edit Token

The tokens (sound files) for a trial protocol are handled from the Edit Token option which is reached from the Edit Protocol option by pressing the "F5" key. Each trial protocol can have up to ten target and ten background tokens. This screen is diagrammed on page 169 of Appendix F.

1. **Target Token** is the file name of a token to be played as part of a change trial.
2. **Probability** specifies the probability that this token will be played during a trial. Tokens without an explicit probability will be assigned a probability automatically. This will be done by assigning each token without an explicit probability an equal part of the total unassigned probability. As an example, if no probabilities are assigned to any of five target tokens for a trial, all five tokens will have an equal probability of 0.20. If all tokens have an explicit probability and the sum is less than one, the last token in the list will be assigned the additional probability.
3. **Background Token** is the file name of a token to be played between trials and as part of a control trial.
4. **Probability** specifies the probability that this token will be played. Tokens without probability will be assigned a probability automatically in the same way as the target tokens.

6.2.8 Edit Reinforcers

The reinforcer configuration for a trial protocol is handled from the Edit Reinforcers option which is reached from the Edit Protocol option by pressing the "F6". Each trial protocol can have up to four different reinforcer configurations. This screen is diagrammed on page 170 of Appendix F.

1. The **First Configuration, Second Configuration, Third Configuration** and **Fourth Configuration** each designate four fields. Each field contains a "Y" or an "N" which specifies whether or not a particular reinforcer is to be activated in a particular configuration.
2. The **Probability** fields specify the probability that the associated configuration will be chosen for reinforcement during a trial. By default, all configurations receive equal probability. If some configurations are assigned an explicit probability and others are not then probabilities are assigned using the same method that is used to assign probabilities to background and target tokens.

6.3 Designing an Experiment Session

A session for an experiment is specified from the bottom up. The trial protocols and schedules that are needed are specified first. Next the stages are specified using existing schedules and trial protocols. Finally, the session itself is specified using existing stages.

6.4 Running an Experiment Session

When the Run Experiment option is selected from the Main Screen, the next step will depend on whether or not sequenced mode has been specified for the experiment session.

If sequenced mode has been specified, the first stage in the sequence is started. The background stimulus is played at the specified intervals and the Running Stage screen is displayed. The program is controlled primarily by the buttons on the GamePad. [Begin] initiates a trial. [Abort]-[Begin] terminates a stage and starts the next stage automatically. After the last stage, the experiment session terminates and control returns to the main screen. During the session, [Abort]-[Vote] selects a retraining trial as the next trial. [Pause] suspends execution of the program. A second press on the [Pause] button restarts the program. Pressing the escape key on the keyboard while the program is paused terminates the experiment session and returns control to the Main Screen. The escape key has no effect if the program is not paused.

If sequenced mode has not been specified, the Select Stage screen is displayed with the list of stages that have been specified for this experiment session. This screen is diagrammed on page 162 of Appendix F. When a stage is selected, the stage is run as described above. If the stage completes by the criterion or by [Abort]-[Begin], control is returned to the select stage screen. If the stage is terminated with the [Pause] button and escape key, control is returned to the Main Screen.

While a stage is running the Running Stage screen is displayed. This screen is diagrammed on page 161 of Appendix F. The name of the stage that is running is displayed at the top of the display. The critical measurement in a stage is the criterion. To meet the criterion, a subject must have a certain number of correct trials within a specified total number of trials. A subject may be required to meet the criterion more than once to successfully complete a stage. The following parameters, reflecting the state of the program, are displayed to help the experimenter in the conduct of the experiment session:

1. **Completed Trials** is the number of trials, for this stage, that have been completed.
2. **Correct So Far** is the number of correct responses so far in the current stage.
3. **Criterion Window** is the number of trials during which the subject must achieve the criterion before the stage is complete.
4. **Required Correct** is the number of correct trials that the subject must achieve within the window.
5. **# of Criteria** is the number of times that the subject must achieve the criterion to successfully complete the stage.
6. **Re-training Trials** is the number of retraining trials that have been done during this stage.
7. **Consecutive Misses** is the current number of consecutive times the subject has failed to respond during a change trial.
8. **Correct in Window** is the number of correct responses so far in the criterion window.
9. **Still to Go** is the number of additional correct responses required to achieve the current criterion.

10. **Times Met** is the number of times the subject has achieved the criterion.
11. **Remaining Criteria** is the number of times remaining that the subject must achieve the criterion to successfully complete the stage.
12. The trial state of the program is shown by the screen colour, which is user selectable.

At the start of each stage, the stage will be checked to see if it includes a schedule. If it does, no further action is required. If it does not, a schedule is automatically computed using the change trial probability, the maxima for consecutive change and control trials and the block size.

6.5 Timing

The timing part of the program can be divided into two parts. The first part, the main loop, cycles continuously checking for button presses and the need to process a timed event. The other part is driven by the timer interrupt.

The count in the timer chip is changed from 65536 to 1193. With a clock frequency of 1.19318 Mhz, this will cause an interrupt every .9998 milliseconds. The regular BIOS interrupt service routine for interrupt 08H is replaced with one that correctly maintains the time of day under these circumstances. The interrupt 1cH service routine, which is normally just an IRET instruction, is replaced with a routine which checks the game port. Button state changes from up to down are recorded by this routine.

An event table is maintained by the program to determine exactly when certain events must happen. Each event table entry contains the time at which the event is to occur and the type of the event. The events in the table are ordered chronologically and they are removed from the table when they have been processed. The main loop of the program checks the event table during every iteration.

The main loop of the system performs the following tasks.

1. Check for a button press.
   a. [Begin]
      If the abort flag is set, the stage is terminated and the abort flag is cleared. If the abort flag is cleared and if the program is between trials, the trial-in-progress flag is set and a trial is initiated. The type of trial is selected based on the schedule for the stage and the retraining flag. If a trial is in progress, the press is ignored.
   b. [Vote]
      If the abort flag is set, the retraining trial flag is set and the abort flag is cleared. If the abort flag is cleared and if a trial is in progress, a head turn is recorded. The length of time between the start of the trial and the button press is recorded. If a trial is not in progress, the press is ignored.
   c. [Abort]
      The abort flag is set.
   d. [Pause]
      Execution of the program is suspended until the [Pause] button is pressed again or until the escape key is pressed. The keyboard is checked only while execution is paused. If an escape key is detected while the program is

paused, the program returns to the main screen. All other key presses are cleared and ignored.

Button presses are cleared when they are detected. With the exception of the [Abort] button, the appropriate action can be initiated as soon as the press is detected. When an [Abort] press is detected, the abort flag is set and an Abort event is placed in the event table. If a [Begin] or a [Vote] press is detected before the event is processed, the abort flag is cleared and the appropriate action for the multiple button press is taken. If the abort flag is still set when the event is processed then the trial is aborted. A long press on a button is recorded as only one press because the routine that records the button presses, int 1cH, only responds when there is a state change from up to down.

2. Check the event table to see if an event must be processed:
   a. Token play event. A stimulus is selected and played. Selection is based on the trial protocol and the list of tokens specified for the current protocol. The probabilities are used in making a random selection from the list. The time at which the next token is due to be played is calculated and entered in the event table based on the length of the token and the inter-stimulus interval.
   b. Reinforcers on event. The reinforcers are activated according to the configurations and probabilities specified for this trial protocol. This happens if the reinforcers are to be activated unconditionally and there is a delay specified. When the trial is initiated, the delay is added to the current time and an entry is made in the event table.

c. Reinforcers off event. The reinforcers will be turned off. This will happen unless the vote button extension has been specified. When the reinforcers are turned on the reinforcement time will be added to the current time and an entry made in the event table.

d. Trial completion event. The trial results are written to the response file and the trial-in-progress flag is cleared. When a trial is initiated with the [Begin] button, the completion time of the trial is calculated by adding the observation interval to the current time. This time will be entered in the event table. On trial completion, if the criterion has been met and the minimum number of trials have been completed, the stage is terminated. If sequenced mode is enabled, the next stage is initiated. If not, the system returns to the stage selection screen.

e. Abort event. If a trial is in progress, it is terminated and the results are ignored. If not, the abort event is ignored.

Because the main loop involves only two tests, unless there is a button press or a timed event to process, the loop time will be significantly less than one msec. Thus the program should be capable, under most circumstances, of responding to a button press in less than two msec.

6.6 Token Processing

The digitized sound files (tokens), comprise the largest volume of data to be handled. At the start of an experiment session, tokens are read into memory until all of the tokens for the experiment session have been read or the available memory has been filled.

256 kilobytes of memory is reserved for storage of tokens. This is enough space for all of the tokens used in most current experiment sessions. If all of the tokens will not fit into memory, the tokens for each stage are read at the start of the stage. The maximum size for an individual token that can be handled is 64 kilobytes. The last four memory pages in the conventional memory area are used to store the tokens. This is the memory space from 060000H to 09ffffH. The PC must have 640 kilobytes of conventional memory installed. There must be a minimum of 384 kilobytes of free memory to run the Headturn system.

A token descriptor table is constructed at the start of the experiment session and updated if necessary at the start of each stage. The table contains:

1. The file name of the token
2. The segment address of the memory page in which the token is stored.
3. The offset within the memory page where the token starts.
4. The length of the token in bytes
5. The elapsed time to play the token. This is based on the length in bytes and the sampling frequency.

For speed of handling during an experiment session, each token is referenced internally by the software by its index in the table.

When a token is played, the memory page and offset of the token in memory and the length of the token are sent to the DMA controller. The length of the token and the sampling rate are sent to the Data Translation board. The DT board's continuous write with DMA command is used to play the token. This frees the CPU from the task of moving the large volumes of digitized sound data from memory to the Data Translation board. If necessary, the DT board can be stopped, while executing a continuous write with DMA command, by using the "STOP" command. Aborting a trial is one example of when this must be done.

6.7 The Implementation

A prototype system, which includes all of the functionality specified in the design, has been implemented. Most of the prototype was written in C. Assembler was used for the timer interrupt service routine. As discussed in Chapter 2, the combination of a high level language with assembler for the timing critical functions is reasonable. This prototype is being tested in the Infant Research Laboratory at UBC. To date, the design seems robust enough so that minimal effort should be required to turn the prototype into a system suitable for production use in ongoing experiments.

Bibliography

[Ather85] P. R. Atherton and L. R. Caporael. A Subjective Judgement Study of Polygon Based Curved Surface Imagery. *CHI '85 Proceedings,* pages 27-34, April 1985.

[Beat91] J. C. Beatty, W. B. Cowan and M. W. Schwarz. An Experimental Comparison of RGB, YIQ, LAB, HSV and Opponent Color Models. *ACM Transactions on Graphics,* Volume 6, Number 2, pages 123-158, 1991.

[Ber92a] Beringer, J. Notes on Graves and Bradley's Timing Algorithm. *Behavior Research Methods, Instruments, & Computers, Vol. 24, No. 3*, 453-455, 1992.

[Ber92b] Beringer, J. Timing Accuracy of Mouse Response Registration on the IBM Microcomputer Family. *Behavior Research Methods, Instruments, & Computers, Vol. 24, No. 3*, 486-490, 1992.

[Bovn90] Bovens, N. and Brysbaert, M. IBM PC/XT/AT and PS/2 Turbo Pascal Timing with Extended Resolution. *Behavior Research Methods, Instruments, & Computers, Vol. 22, No. 3*, 332-334, 1990.

[Buhr87] Buhrer, M, Sparrer, B. and Weitkunat, R. Interval Timing Routines for the IBM PC/XT/AT Microcomputer Family. *Behavior Research Methods, Instruments, & Computers, Vol. 19, No. 3*, 327-334, 1987.

[Creeg90] Creeger, C. P., Miller, K. F. and Paredes, D. R. Micromanaging Time: Measuring and Controlling Timing Errors in Computer-Controlled Experiments. *Behavior Research Methods, Instruments, & Computers, Vol. 22, No. 1*, 34-79, 1990.

[Cros89] Crosbie, J. A Simple Turbo Pascal 4.0 Program for Millisecond Timing on the IBM PC/XT/AT. *Behavior Research Methods, Instruments, & Computers, Vol. 21, No. 3*, 408-413, 1989.

[Cros90] Crosbie, J. The Microsoft Mouse as a Multipurpose Response Device for the IBM PC/XT/AT. *Behavior Research Methods, Instruments, & Computers, Vol. 22, No. 3*, 305-316, 1990.

[Dalry92] Dalrymple-Alford, E. C. Response Key Input Via the IBM PC/XT/AT's Parallel Printer Ports. *Behavior Research Methods, Instruments, & Computers, Vol. 24, No. 1*, 78-79, 1992.

[Dix47] Dix, M. R. and Hallpike, C. S. The Peepshow: New Technique for Pure Tone Audiometry in Young Children. *British Medical Journal*, 719, 1947.

[Dlhop88] Dlhopolsky, J. G. C Language Functions for Millisecond Timing on the IBM PC. *Behavior Research Methods, Instruments, & Computers, Vol. 20, No. 6*, 560-565, 1988.

[Dlhop89] Dlhopolsky, J. G. Synchronizing Stimulus Displays with Millisecond Timer Software for the IBM PC. *Behavior Research Methods, Instruments, & Computers, Vol. 21, No. 4*, 441-446, 1989.

[Dorf87] Dorfman, D. Software Timing of Events in Cognitive Psychology Experiments. *Behavior Research Methods, Instruments, & Computers, Vol. 19, No. 2*, 185-190, 1987.

[Dunc88] Article 6: Interrupt Driven Communications. In R. Duncan, editor *The MS-DOS Encyclopedia*, pages 167-246, 1988.

[Edge89] Edgell, S. E. and Hertel, S. A. Running Laboratory Experiments Using the RSX Operating System and FORTRAN on the PDP-11. *Behavior Research Methods, Instruments, & Computers, Vol. 21, No. 2*, 303-306, 1989.

[Eiler77] Eilers, R. E., Wilson, W. R. and Moore, J. M. Developmental Changes in Speech Discrimination in Infants. *Journal of Speech and Hearing Research.*, Vol. 20, pp 766-780, 1977.

[Emer88a] Emerson, P. L. Using Serial Interfaces and the C Language for Real-Time Experiments. *Behavior Research Methods, Instruments, & Computers, Vol. 20, No. 3*, 330-336, 1988.

[Emer88b] Emerson, P. L. TIMEX: A Simple IBM AT C Language Timer with Extended Resolution *Behavior Research Methods, Instruments, & Computers, Vol. 20, No. 6*, 566-572, 1988.

[Ferr88] Ferraro, R. F. *Programmer's Guide to the EGA and VGA Cards*, Addison-Wesley Publishing Co. 1988

[Gleit81] Gleitman, H. *Psychology,* W. W. Norton & Company Inc., New York 1981.

[Grav87] Graves, R and Bradley, R. Millisecond Interval Timer and Auditory Reaction Time Programs for the IBM PC. *Behavior Research Methods, Instruments, & Computers, Vol. 19, No. 1*, 30-35, 1987.

[Gran89] Granaas, M. M. TIMEX2: A Modified C-Language Timer for PC AT Class Machines. *Behavior Research Methods, Instruments, & Computers, Vol. 21, No. 6*, 619-622, 1989.

[Grav88] Graves, R. and Bradley, R. More on Millisecond Timing and Tachistoscope Applications for the IBM PC. *Behavior Research Methods, Instruments, & Computers, Vol. 20, No. 4*, 408-412, 1988.

[Grav91] Graves, R. E. and Bradley, R. Millisecond Timing on the IBM PC/XT/AT and PS/2: A Review of the Options and Corrections for the Graves and Bradley Algorithm. *Behavior Research Methods, Instruments, & Computers, Vol. 23, No. 3*, 377-379, 1991.

[Jam92] Jamieson, D. G., Ramji, K. V., Kheirallah, I. and Nearey, T. M. *Canadian Speech Research Environment* 1992.

[Kuhl79] Kuhl, P. K. Speech Perception in Early Infancy: Perceptual Constancy for Spectrally Dissimilar Vowel Categories. *Journal of the Acoustical Society of America.* Vol. 66, pp 1668-1679, 1979.

[Moore75] Moore, J. M., Thompson, G. and Thompson, M. Auditory Localization of Infants as a Function of Reinforcment Conditions. *Journal of Speech and Hearing Disorders.* 40, pages 29-34, 1975.

[Moore77] Moore, J. M., Wilson, W. R. and Thompson, G. Visual Reinforcement of Head-turn Responses in Infants Under 12 Months of Age. *Journal of Speech and Hearing Disorders.* 42, pages 328-334.

[Posn67] Posner, M. I. and Fitts, P. M. *Human Performance* Brooks Cole Publishing Co., 1967.

[Schnd88] Schneider, W. Micro Experimental Laboratory: An Integrated System for IBM PC Compatibles. (1988). *Behavior Research Methods, Instruments, & Computers, Vol. 20, No. 2*, 206-217, 1988.

[Schnd91] Schneider, W. Equipment is Cheap, But the Field Must Develop and Support Common Software for Psychological Research. *Behavior Research Methods, Instruments, & Computers, Vol. 23, No. 2*, 114-116, 1991.

[Segal87] Segalowitz, S. J. IBM PC Tachistoscope: Text Stimuli. *Behavior Research Methods, Instruments, & Computers, Vol. 19, No. 4*, 383-388, 1987.

[Segal90] Segalowitz, S. J. and Graves, R. E. 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, Vol. 22, No. 3*, 283-289, 1990.

[Shear88] The Shearer Group. *Operator's Manual*, Dallas Texas, 1988.

[Shoup79] Shoup, R. G. Colour Table Animation. *Computer Graphics Vol. 13, No. 2*, (Siggraph '79 Conference) pp. 8-13, August 1979.

[Werk81] Werker, J. F., Gilbert, J. H. V. Humphrey, K. and Tees, R. C. Developmental Aspects of Cross-Language Speech Perception. *Child Development.* 52. pp. 349-353, 1981

[Werk82] Werker, J. F. Development of Cross-Language Speech Perception: the Influence of Age, Experience, and Context on Perceptual Organization. *Unpublished Ph.D. Thesis.* 1982

[Werk83] Werker, J. F. and Tees, R. C. Developmental Changes Across Childhood in the Perception of Non-native Speech Sounds. *Canadian Journal of Psychology.* 37(2). pp. 278-286, 1983

[Werk88] Werker, J. F. and Lalonde, C. E. Cross-Language Speech Perception: Initial Capabilities and Developmental Change. *Developmental Psychology.* Vol 24 No 5. pp. 672-683, 1988

[Werk91] Polka, L. and Werker, J. F. Developmental Changes in Perception of Non-native Vowel Contrasts. *Journal of Experimental Psychology: Human Perception and Performance.* (in press).

[Wils76] Wilson, W. R., Moore, J. M. and Thompson, G. Sound-Field Auditory Thresholds of Infants Utilizing Visual Reinforcement Audiometry (VRA). *Paper presented at the Annual Convention of the American Speech and Hearing Association.* Houston, 1976.

[Yaphe89] Yaphe, M., Raftery, E. and Jamieson, D. G. A General-Purpose Facility for Adaptive Testing in Psychoacoustics. *Behavior Research Methods, Instruments, & Computers, Vol. 21, No. 2*, 275-289, 1989.

Appendix A

Headturn Specification

This specification was supplied by Mr. Goldhardt as part of the Headturn package. A request for permission to reprint this specification was sent to the address that is included in the specification. The request was returned as undeliverable by the post office. What follows is a re-formatting of the original documentation. The figures have been re-drawn.

Software Specifications for "HEADTURN"

(A Program for Visually-Reinforced Head-Turn Experiments)

Revision 2

Nirvonics, Inc.
P.O. Box 5062
Plainfield, NJ 07061

Introduction

HEADTURN is a program for the IBM-PC family of computers which is designed to assist an experimenter by controlling a **visually-reinforced head-turn procedure.**

The setup consists of a **subject infant** (S) in a room in view of the **experimenter** (E) who is in a darkened booth; thus, **E** can see **S**, but **S** cannot see **E**. To the right of **S** is a **speaker** and an array of four **reinforcers** (mechanical toys which can be activated individually or in combination). In the booth with **E** is a computer **console** (terminal) and a **button box** (see Figure 1).

Each experiment consists of a number of **stages** (2 training stages, a conditioning stage, and 1-4 testing stages). The basic procedure for each stage is as follows. **E** uses the console to establish a set of operational parameters, some of which are implicit in the selection of the stage, and others which are specified explicitly by **E** (at this time or previously, by creating a parameter file). When **E** presses the [**Begin**] button, the experiment (stage) begins.

During the entire duration of a stage of the experiment, some **stimulus** (e.g. speech sound /a/) is repeatedly played over the speaker at a constant **inter-stimulus interval.** (The length of the interval is one of the parameters.) While the interval remains constant, the actual stimulus may change. The initial stimulus is called the **background** stimulus.

When **E** determines that **S** is in an attentive state, s/he presses the [**Begin**] button to begin a **trial**; the trial actually begins at the next stimulus onset time. A trial is a specified period of time (the **observation interval**) during which **E** looks for **S** to make a head-turn (towards the speaker). In most trials, the background stimulus is replaced by a **target** stimulus for a specified number of repetitions (usually 3). Also, one or more of the reinforcers may be activated, either unconditionally or contingent upon a head-turn. The 4 types of trials are listed below.

Trial Type	**Stimulus**	**Reinforcer Activation**
control trial	background	none
training trial	target	immediate (unconditional)
re-training trial	target	at head-turn, or after onset delay
change trial	target	at head-turn

Note that in a re-training trial, the reinforcers are activated automatically after a specified delay; but if a head-turn occurs first, the reinforcers are activated at the head-turn.

After each trial, the reinforcers are de-activated and the background stimulus resumes. Trials continue until specified criteria are met (e.g. minimum # of trials, hit ratio), or until **E** manually terminates the process by holding down the [**Abort**] button and pressing the [**Begin**] button.

The Button Box

Once the parameters have been set, all control of the program is through the Button Box. **E** starts the experiment running and begins each. trial by pressing the [**Begin**] button. **E** uses the [**Vote**] button to indicate a head-turn.

The READY light is on whenever the program is ready to begin a trial (i.e. it's not in the middle of a previous trial). As soon as [**Begin**] is used to begin a trial, the READY light goes off. If the [**Abort**] button is pressed any time while the READY light is off, the trial is discontinued (at the first inter-stimulus interval) and discarded.

The [**Abort**] button can also be used while the READY light is on, as a sort of special-function shift key. If [**Abort**] is held down while [**Vote**] is pressed, the next trial will be a re-training trial. If [**Abort**] is held down while [**Begin**] is pressed, the experiment (stage) is terminated (this is the only way to terminate Training Stage 1).

In training trials (Training Stage 1), the [**Vote**] button has a special use. (Its normal use is not applicable to training trials, since reinforcement does not depend on head-turn.) When the reinforcer is activated at the onset of the first target stimulus, it normally remains active for a previously-specified length of time. However, if the [**Vote**] button is pressed during this time, the reinforcer will be de-activated when the [**Vote**] button is released, whether this is before or after the normally scheduled time; if this extends the time beyond what would have been the beginning of the next background stimulus, additional target stimuli are inserted, as necessary.

The Stages

The initial stage of an experiment is Training Stage 1. This stage consists only of training trials (which use one background stimulus and one target stimulus). This stage is terminated by **E** pressing [**Abort**]-[**Begin**]. This is the only stage which does not allow re-training trials with [**Abort**]-[**Vote**].

The next stage is Training Stage 2. This stage consists entirely of re-training trials. As mentioned above, the only difference between training trials and re-training trials is that in re-training trials, the reinforcer is not activated until a specified onset delay, unless **E** indicates (with the [**Vote**] button) that a head-turn was made. Typically, the background and target stimuli will be the same as in Training Stage 1.

Note that in Training Stage 2, both the normal trials and [**Abort**]-[**Vote**] trials are re-training trials. This allows **E** to specify two different onset delays as parameters. Typically, the onset delay for [**Abort**]-[**Vote**] trials in this stage will be zero.

The third stage is the Conditioning Stage. In this stage, change and control trials are randomly interspersed. The single background and target stimuli used will typically be the same as in Training Stage 1. The probability of each type of trial and the maximum number of consecutive trials of each type are specified as parameters.

The remaining stages are Testing Stages 1-4. These are identical to the Conditioning Stage, except that instead of single background and target stimuli, **E** specifies sets of stimuli to be chosen from random. **E** must also specify the (single) background and target stimuli to use for re-training trials; these are typically the same as the stimuli for Training Stage 1.

The procedure the program uses for selecting from the stimulus sets is as follows. As each background or target stimulus is ready to begin, the program randomly chooses a stimulus from the corresponding (background or target) stimulus set. The probability that each stimulus will be chosen at any given time may be specified in the operational parameters given by **E**.

E has total control over the order of stages. Normally, **E** will use the order described above, but may skip ahead or go back to a previous stage at any time.

The order of stages can also be programmed in advance. This is described in the Automatic Sequencing section.

Operational Parameters

The operational parameters which **E** specifies at the beginning of each session are divided into several Groups: general parameters, which apply to the entire experiment, and a group for each stage of the experiment.

The general parameters are listed below:

Parameter file to use for defaults:
Parameter file to save new values in:

ID #:
Subject's name:
Subject's birthdate:

Response file:
Stimulus set file:

Inter-stimulus interval (msec):
Number of stimulus repetitions per trial:

Number of Testing Stages in this experiment:
Run stages in sequence from parameter file instead of menu:

The <Parameter file to use for defaults> expects the name of an existing file which will supply default values for the remaining parameters; if this is left blank, no defaults will be provided.

For the rest of the parameters, including stage-specific

parameters not yet mentioned, if a blank response is given, the default values (if any) will be used. If any response (including the first one) ends with a dollar-sign ("$"), the remaining parameters will automatically take their default values.
The <Parameter file to save new values in>, if non-blank, is the name of a file which will receive all the values you specify for subsequent parameters, plus unused defaults from the <Parameter file to use for defaults>, if any. This file can be used later as a <Parameter file to use for defaults>. The default for this parameter is always blank.

<ID #>, <Subject's name>, and <Subject's birthdate> are used for identification only, and are simply copied to the <Response file>.

The <Response file> receives a log of the experimental results and parameters for the session. Its format is described in another section. The <Stimulus set file> provides the mapping between stimulus numbers (used to select stimuli in subsequent responses) and actual stimulus files. Each line contains a speech file name. The first line is stimulus number 1, the second is stimulus number 2, and so on.

The <Inter-stimulus interval> is the duration, in milliseconds, of the silence between the end of one stimulus and the beginning of the next.

The <Run stages in sequence from parameter file instead of menu> parameter should normally be left as No, unless you want automatic sequencing (see Automatic Sequencing, below).

The remaining general parameter prompts are self-explanatory.

After the general parameters (and subsequently, after each stage has been terminated), **E** uses a menu to select the stage to run. **E** is then prompted for parameters appropriate to that stage (which may have defaults in a parameter file). Again, **E** can use a "$" to select defaults for all the parameters.

The following parameters are common to all stages:

Experimenter:
Assistant:

Reinforcer configuration
Duration of reinforcement (msec):
Observation interval (msec):

<Experimenter> and <Assistant> are used for identification only, and are simply copied to the <Response file>. Note that these parameters have no defaults, and must always be entered.

The <Reinforcer configuration> indicates which of the 4 reinforcers will be activated (when appropriate). A reinforcer configuration is a list of one or more reinforcer numbers (1-4), separated by commas (","); for example, "2,3" means reinforcers 2 and 3 will be activated simultaneously. The response to this prompt may actually be several configurations (up to 4) separated by semi-colons (";"); this means that the various configurations are to be chosen from randomly (with equal probability).

The <Duration of reinforcement> is the number of milliseconds that reinforcers remain activated once triggered (unless overridden in Training Stage 1). Note that the <Duration of reinforcement> takes precedence over the <Number of stimulus repetitions per trial>. If reinforcement ends before the beginning of one or more (target) stimuli which would normally be part of the trial, the trial is terminated prematurely and the next stimulus will be a back-ground stimulus. If reinforcement extends past the beginning of what would normally be the first background stimulus after the trial, the trial is extended and additional target stimuli are inserted as necessary.

The <Observation interval> is the length of time, in milliseconds, from the onset of the first stimulus in a trial, during which the **[Vote]** button is recognized as an indication of a headturn. Outside of this interval, **[Vote]**s are ignored.

The following parameter is only used in Training Stage 2:

Normal trials: Delay in onset of reinforcement (msec):

This parameter gives the onset delay (in milliseconds) for normal trials, which in this stage are always re-training trials.

The following parameter is used in all stages except Training Stage 1:

Re-training trials: Delay in onset of reinforcement (msec):

This parameter gives the onset delay (in milliseconds) for re-training trials invoked with **[Abort]-[Vote]**.

The following parameters are used in all stages except Testing Stages:

Target stimulus #:
Background stimulus #:

<Target stimulus #> and <Background stimulus #> identify the stimuli to be used in all trials in the stage. Typically, these will be the same for all stages.

Instead of the above parameters, Testing Stages have the following

Target stimulus/stimuli #s:
Background stimulus/stimuli #s:
Re-training target stimulus #:
Re-training background stimulus #:

<Target stimulus/stimuli #s> and <Background stimulus/stimuli #s> identify the sets of stimuli to choose from for trials in the stage. Each of these may be one or more stimulus numbers, separated by commas (","). Also, each stimulus number may be followed by a colon (":") and a decimal fraction (e.g. 0.23) indicating the probability of occurrence of that stimulus. Any stimuli within each set which don't have probabilities specified get the remaining probability (out of 1.0) split equally between them.

<Re training target stimulus #> and <Re-training background stimulus #> identify the stimuli to be used when **E** invokes re-training trials. Typically, these will be the same as the <Target stimulus #> and <Background stimulus #> for other stages.

The following parameters are used in all Stages except Training Stage 1:

> Minimum number of trials:
> Criterion: Number of (almost) consecutive correct trials:
> Number of trials to check for criterion:
> Number of times criterion must be met:
> Number of consecutive misses to cause a warning beep:

The first four parameters provide the termination criteria for a stage. When these criteria are met, the program will automatically terminate the stage.

Specifically, after each correct trial (hit or correct rejection), the last <Number of trials to check for criterion> trials are checked to see if at least <Criterion. Number of (almost) consecutive correct trials> of them are correct. If they are, the criterion-met count is incremented. Once the criterion-met count is at least <Number of times criterion must be met> and there have been at least <Minimum number of trials> trials, the stage is terminated.

The final parameter of this set, <Number of consecutive misses to cause a warning beep>, provides the number of consecutive misses which will generate a warning beep to let **E** know it may be appropriate to select a re-training trial. If this is left blank or zero, no warnings will be given.

The last parameters are used in the Conditioning Stage and in Testing Stages:

Probability of change (vs. control) trials:
Maximum number of successive change or control trials:

The <Probability of change (vs. control) trials> is a decimal fraction between 0.0 and 1.0 giving the probability of selecting a change trial instead of a control trial. The <Maximum number of successive change or control trials> imposes a constraint on the random process.

Summary of Operational Parameters

General Parameters

> Parameter file to use for defaults:
> Parameter file to save new values in:
>
> ID #:
> Subject's name:
> Subject's birthdate:
>
> Response file:

Stimulus set file:

Inter-stimulus interval (msec):
Number of stimulus repetitions per trial:

Number of Testing Stages in this experiment:
Run stages in sequence from parameter file instead of menu:

Training Stage 1

Experimenter:
Assistant:

Reinforcer configuration:
Duration of reinforcement (msec):
Observation interval (msec):

Target stimulus #:
Background stimulus #:

Training Stage 2

Experimenter:
Assistant:

Reinforcer configuration:
Duration of reinforcement (msec):
Observation interval (msec):
Normal trials: Delay in onset of reinforcement (msec):
Re-training trials: Delay in onset of reinforcement (msec):

Target stimulus #:
Background stimulus #:

Minimum number of trials:
Criterion: Number of (almost) consecutive correct trials:
Number of trials to check for criterion:
Number of times criterion must be met:
Number of consecutive misses to cause a warning beep:

Conditioning Stage

Experimenter:
Assistant:

Reinforcer configuration:
Duration of reinforcement (msec):
Observation interval (msec):
Re-training trials: Delay in onset of reinforcement (msec):

Target stimulus #:

Background stimulus #:

Minimum number of trials:
Criterion: Number of (almost) consecutive correct trials:
Number of trials to check for criterion:
Number of times criterion must be met:
Number of consecutive misses to cause a warning beep:

Probability of change (vs. control) trials:
Maximum number of successive change or control trials:

Testing Stages 1-4

Experimenter:
Assistant:

Reinforcer configuration:
Duration of reinforcement (msec):
Observation interval (msec):
Re-training trials: delay in onset of reinforcement (msec):

Target stimulus/stimuli #s:
Background stimulus/stimuli #s:
Re-training target stimulus #:
Re-training background stimulus #:

Minimum number of trials:
Criterion: Number of (almost) consecutive correct trials:
Number of trials to check for criterion:
Number of times criterion must be met:
Number of consecutive misses to cause a warning beep:

Probability of change (vs. control) trials:
Maximum number of successive change or control trials:

Response File Format

HEADTURN keeps a log of operational parameters used, and trial results, in a response file. HEADTURN always appends to the end of response files; it never writes over existing information.

Information in the file is divided into sections corresponding to each stage of the experiment. The stages are identified by the following numbers:

1 - Training Stage 1
2 - Training Stage 2
3 - Conditioning Stage
4 - Testing Stage 1
5 - Testing Stage 2
6 - Testing Stage 3
7 - Testing Stage 4

Each of these sections has two parts, one for operational parameters and the other for trial results. The format is shown below:

> *** STAGE <stage_number>
> {<parameter>}
> *** TRIALS
> {<trial result>}
> *** TERMINATED <reason>

Curly braces ("{" and "}") indicate zero or more occurrences, and are not part of the file. <stage_number> is from the list above. A <parameter> is:

> <param_name> = <param_value>

<param_name> is a parameter name (see the next section), and <param_value> is the value for that parameter as it would be typed in response to a prompt.

A <trial_result> is:

> <start_time> <type> <background_list>;<target_list>;<vote_time>

<start_time> is the number of milliseconds from the beginning of the stage to the trial onset. <type> (the type of stimuli in <trial_list>) is B for background (control trial) or T for target (change trial); for a retraining trial, <type> will be R (and <vote_time> will be undefined).

<background_list> and <target_list> are of the form:

> <stimulus_number>{,<stimulus_number>}

The <background_list> will never contain more than 3 stimuli (the last 3). <vote_time> is the number of milliseconds from the trial onset to the onset of a **[Vote]**; if no **[Vote]** was made (in the observation window), <vote_time> will be -1.

The <reason> for termination will be MANUALLY if **[Abort]**-**[Begin]** was used, or BY CRITERIA if the termination criteria were met.

General parameters are also put into the response file, using the special stage number 0. Note that this section omits the TRIALS and TERMINATION parts:

> *** STAGE 0
> {<parameter>}

Actually, any stage may have the TRIALS and TERMINATION parts omitted. This will happen if **E** specifies the parameters for the stage but cancels the stage before pressing **[Begin]**.

Parameters and Parameter Files

These are the parameter names used for operational parameters in response files and parameter files:

Parameter Name	Prompt
ID	ID #:
SNAME	Subject's name:
SBDAY	Subject's birthdate:
RFILE	Response file:
SFILE	Stimulus set file:
INTER	Inter-stimulus interval (msec):
SREP	Number of stimulus repetitions per trial:
STAGES	Number of Testing Stages in this experiment:
SEQ	Run stages in sequence from parameter file instead of menu:
ENAME	Experimenter:
ANAME	Assistant:
RCONF	Reinforcer configuration:
RTIME	Duration of reinforcement (msec):
OTIME	Observation interval (msec):
ODELAY	Normal trials: Delay in onset of reinforcement (msec):
RDELAY	Retraining trials: Delay in onset of reinforcement (msec):
TSTIM	Target stimulus #
	Target stimulus/stimuli #s:
BSTIM	Background stimulus #
	Background stimulus/stimuli #s:
RTSTIM	Re-training target stimulus #:
RBSTIM	Re-training background stimulus #:
MINT	Minimum number of trials:
CRIT	Criterion: Number of (almost) consecutive correct trials:
CRITWN	Number of trials to check for criterion
NCRIT	Number of times criterion must be met:
NMISS	Number of consecutive misses to cause a warning beep:
CHANGE	Probability of change (vs. control) trials:
MXSUCC	Maximum number of successive change or control trials:

The syntax of a parameter file is like that of a response file, except that all stages, not just stage 0, omit the TRIALS and TERMINATION parts.

Each group of parameters specific to a particular stage is preceded by a line of the form:

*** STAGE <stage_number>

The general parameters are preceded by:

*** STAGE 0

See the Response File Format section for a list of stage numbers.

Automatic Sequencing

Normally, the defaults for all the stages are loaded in at once, and **E** chooses each stage to run interactively, one at a time, from a stage selection menu. However, if the SEQ parameter (<Run stages in sequence from parameter file instead of menu>) is set to Yes in a parameter file, the program will automatically sequence stages in the order they are listed in the parameter file, bypassing the stage selection menu and the parameter screens.

All of the parameters for each stage must be specified in the parameter file, including ENAME and ANAME, which are ignored in parameter files for non-sequenced mode. If the program encounters a missing parameter at any time, it will revert to non-sequenced mode and present **E** with a parameter screen so s/he can fill in the missing parameter(s).

Particular stages may be repeated in the sequence simply listing them multiple times. Parameter values for a repeated stage may be omitted from the parameter file; if they are, they will take on the values from the previous instance of the same stage.

In both sequenced and non-sequenced modes, **E** may abort a stage before it begins, by pressing any key on the keyboard. This will also cause the program to revert to non-sequenced mode if it was in sequenced mode and bring up the stage selection menu.

Hardware Specifications for

"HEADTURN"

(A Program for Visually-Reinforced Head-Turn Experiments)

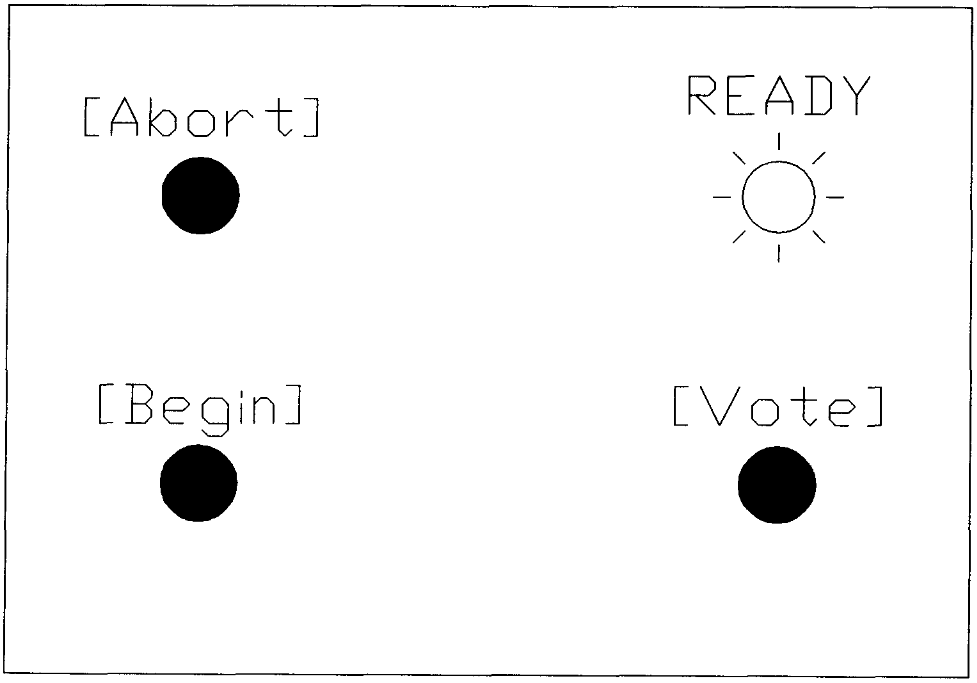
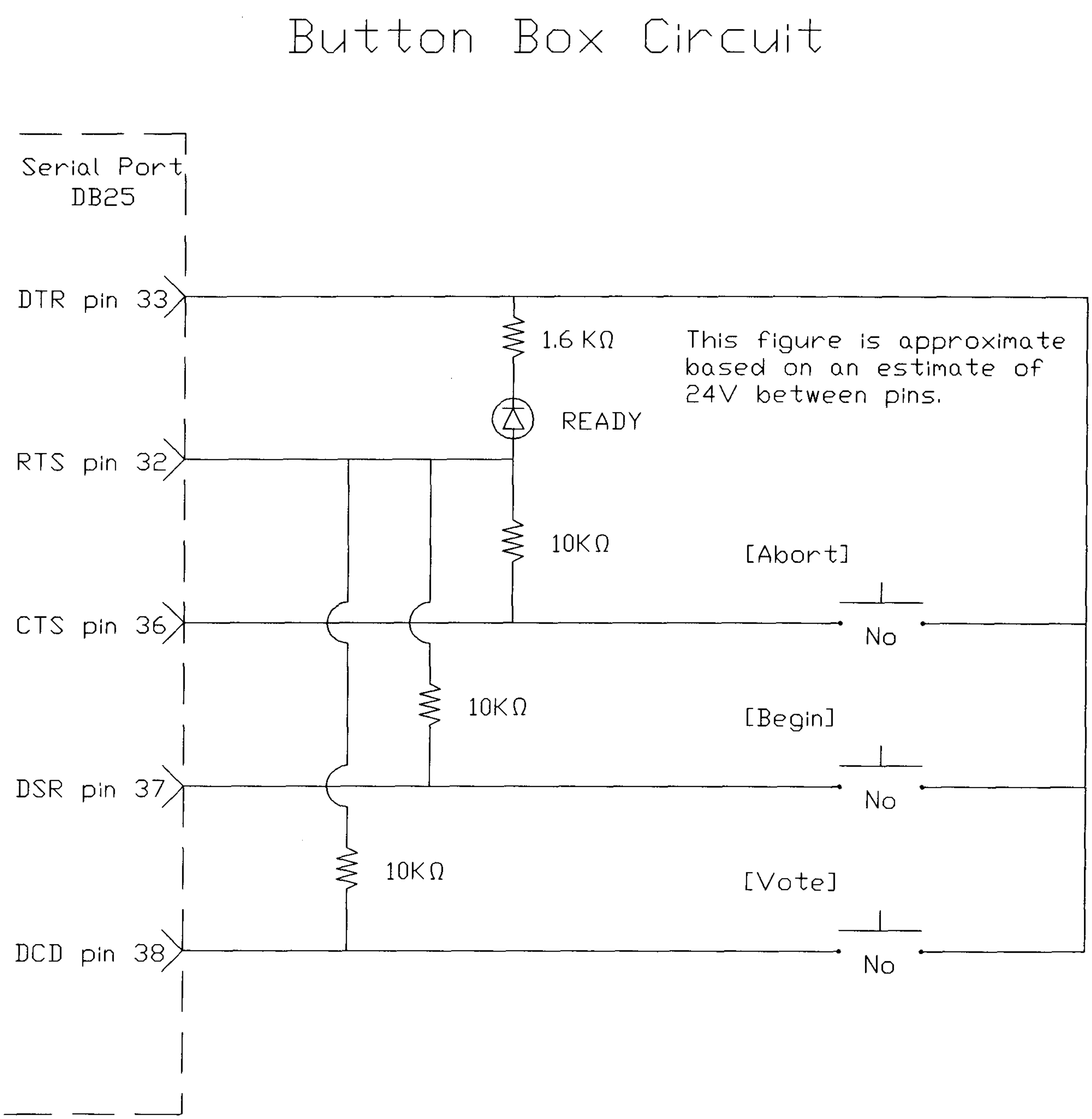


Figure 1. The Button Box


Button Box Circuit
Serial Port
DB25
DTR pin 33
RTS pin 32
CTS pin 36
DSR pin 37
DCD pin 38
1.6 KΩ
READY
10KΩ
10KΩ
10KΩ
This figure is approximate
based on an estimate of
24V between pins.
[Abort]
No
[Begin]
No
[Vote]
No

A.6 DIGITAL I/O SUBSYSTEM

DIGITAL INPUTS	
Input Type	Level sensitive
Logic Family	LSTTL
Logic Load	Presents 1 LSTTL load
Logic High Input Voltage	2.0V minimum
Logic Low Input Voltage	0.8V maximum
Logic High Input Current	1.0mA @ 5.5v maximum 0.07mA @ 2.7V (current at threshold voltage)
Logic Low Input Current	0.250mA maximum

DIGITAL OUTPUTS	
Fanout	Drives 30 LSTTL loads
Logic Family	LSTTL
Logic High Output Voltage	2.4V minimum
Logic Low Output Voltage	O.5V maximum
Logic High Output Current	24mA maximum 6.5mA @ 2.4V (current at threshold voltage)
Logic Low Output Current	24mA maximum

Compatible with OPTO-22 solid state relays

Connect reinforcers to DT2801 digital outputs

Reinforcer 1	DIO 0 BIT 0 (pin 28)
Reinforcer 2	DIO 0 BIT 1 (pin 29)
Reinforcer 3	DIO 0 BIT 2 (pin 30)
Reinforcer 4	DIO 0 BIT 3 (pin 40)

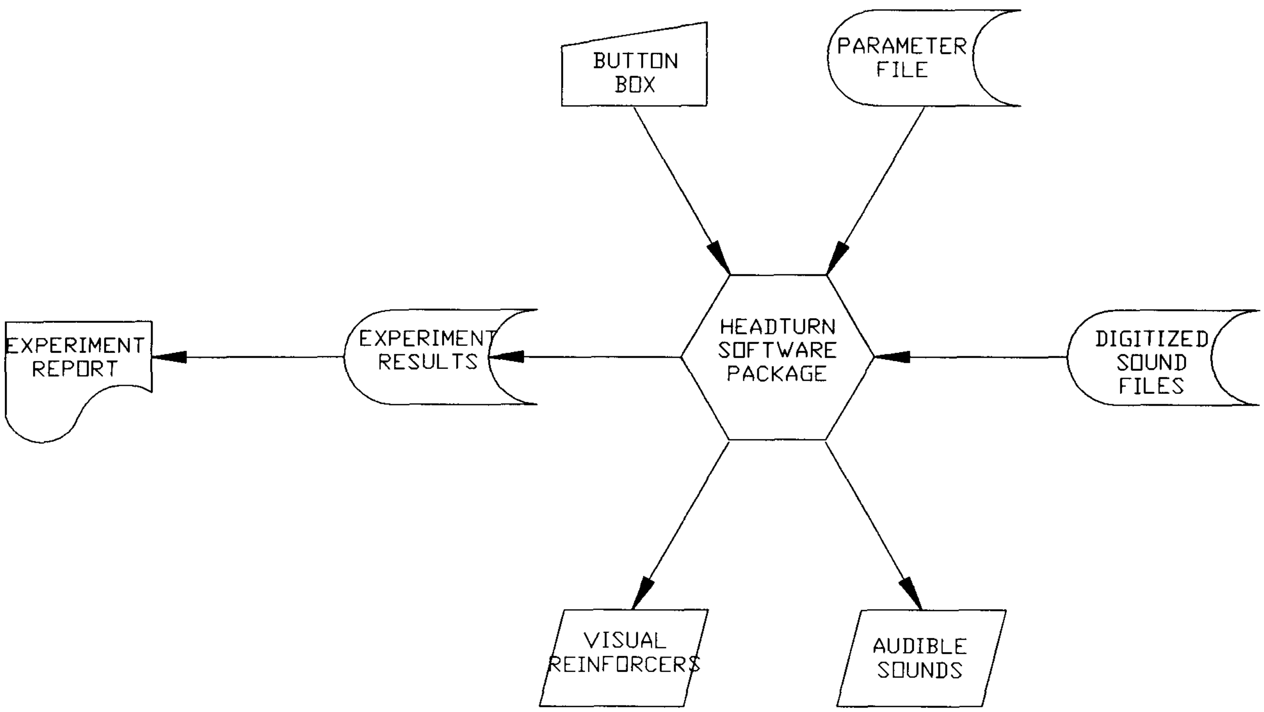
Note: Do not connect any signal sources (inputs) to any other bits of DIO 0.

Appendix B

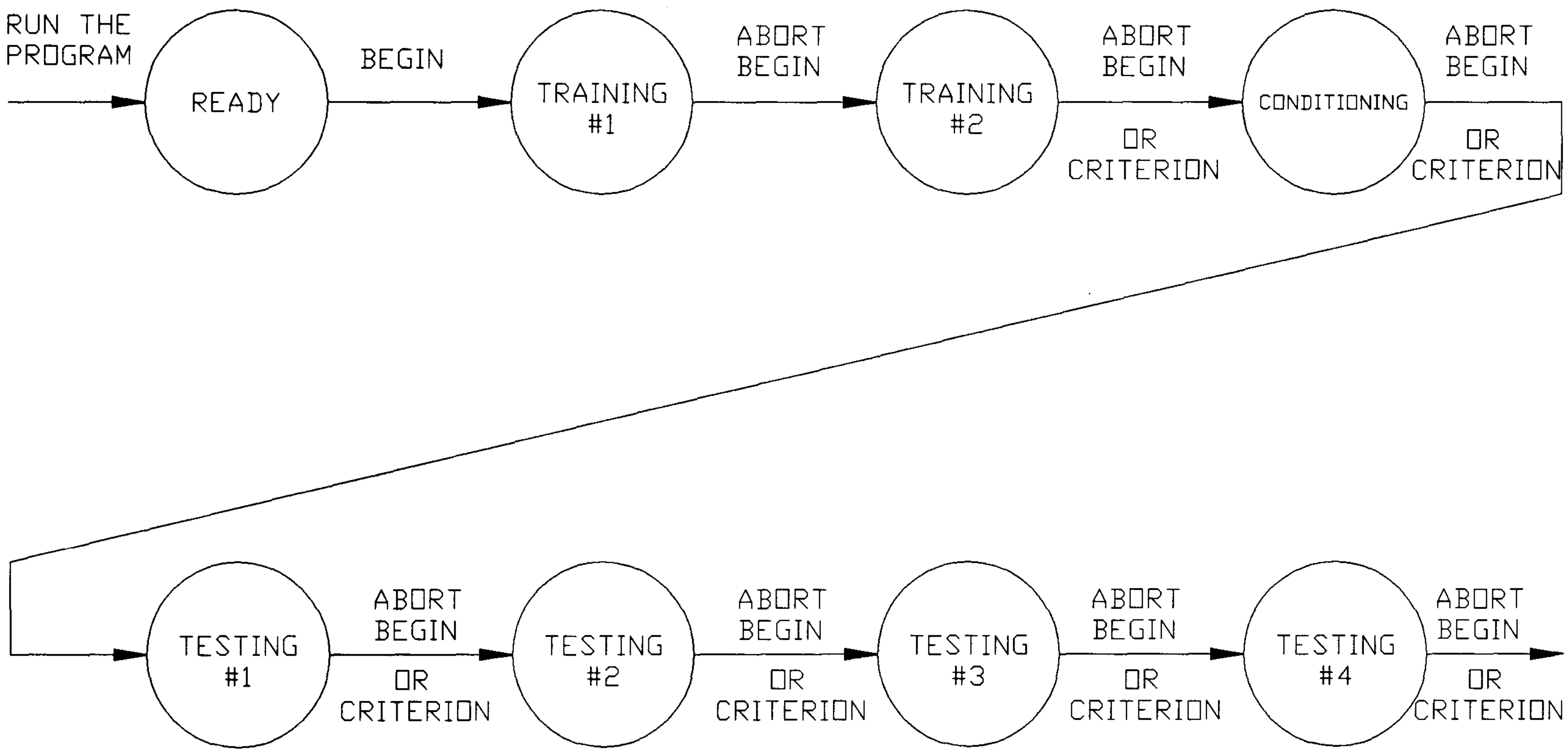
Headturn Diagrams

The diagrams in this appendix provide a graphical representation of the operations of the current Headturn software.

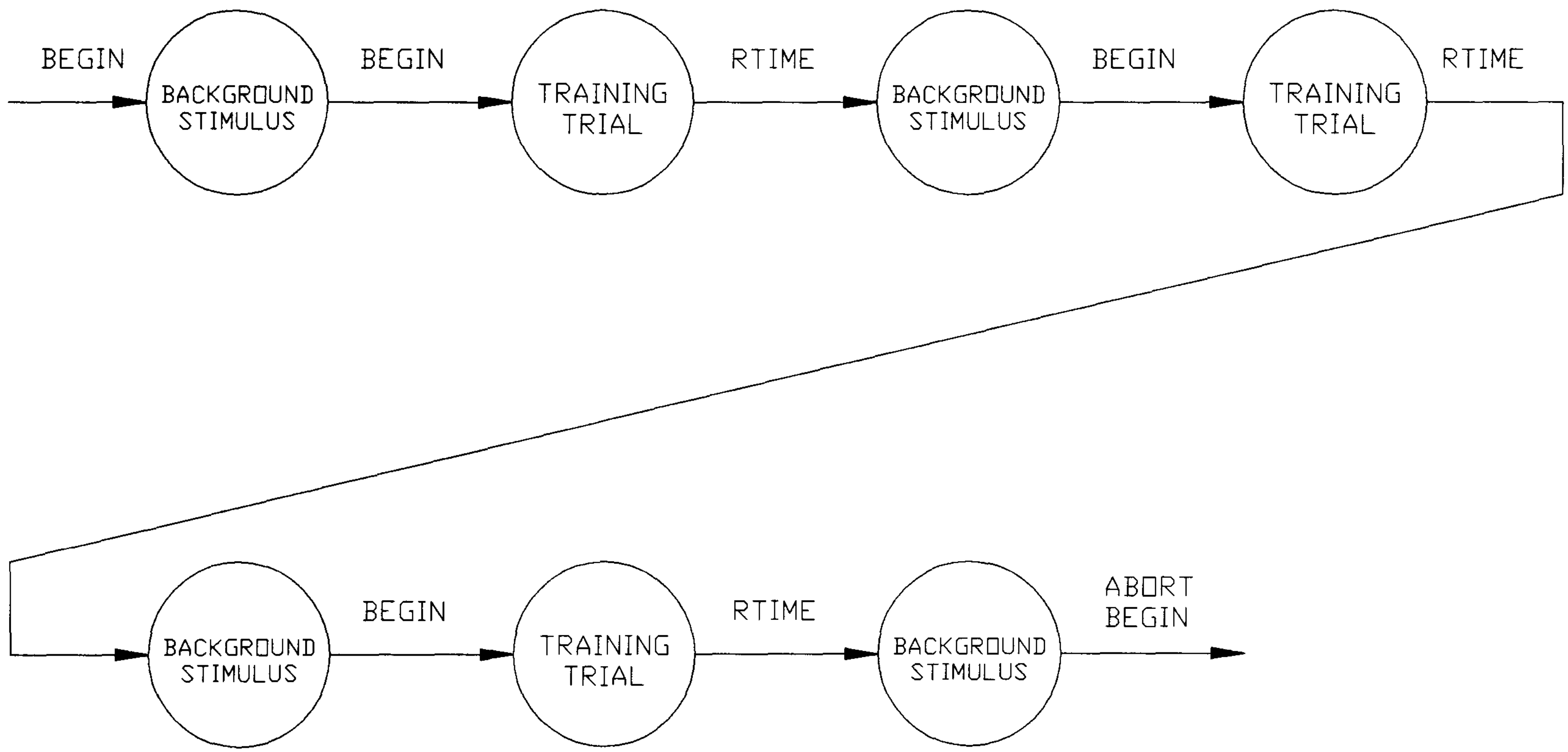
HEADTURN SOFTWARE SCHEMATIC



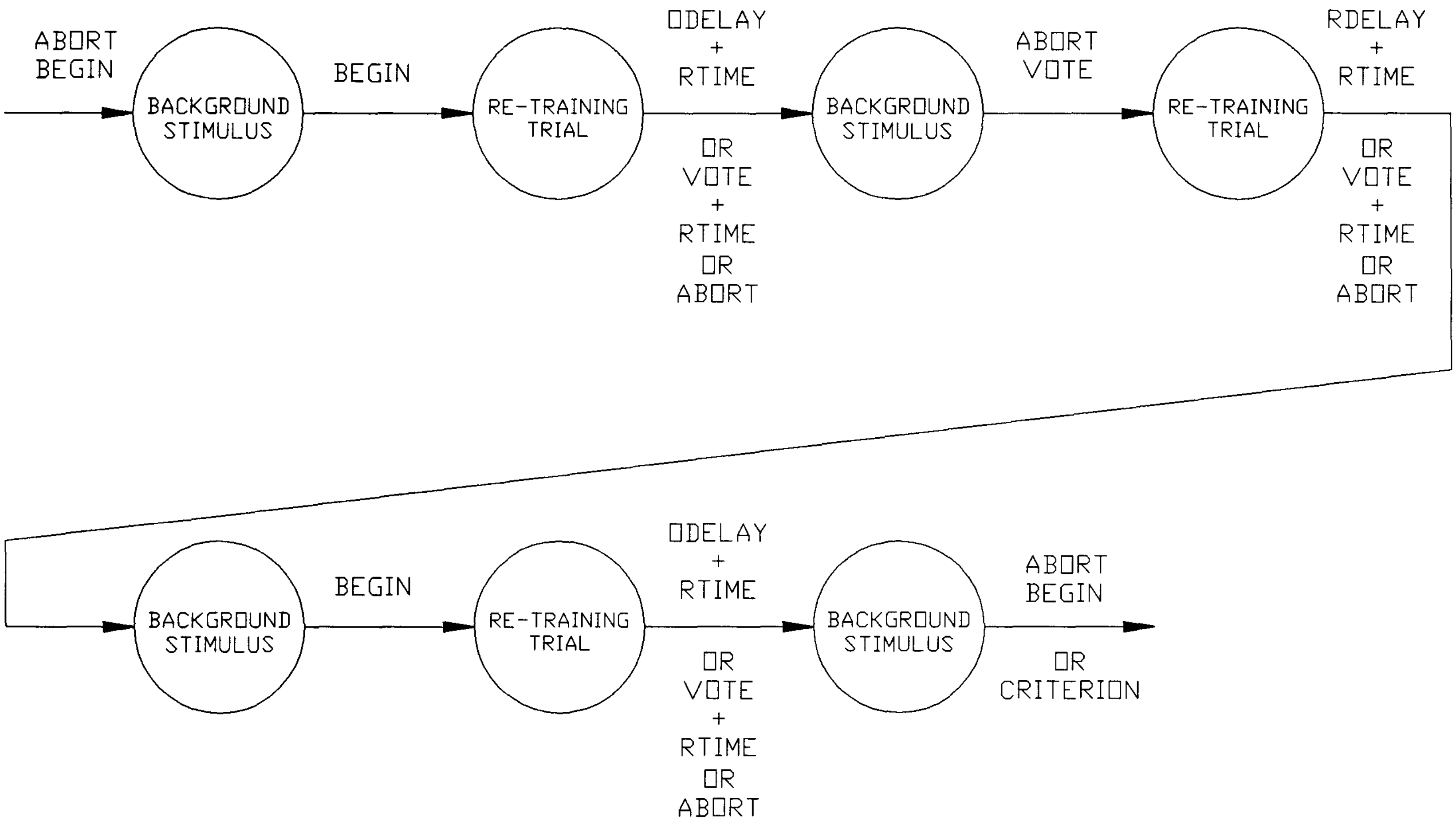
HEADTURN STATE DIAGRAM
TOP LEVEL
SEQUENCED MODE



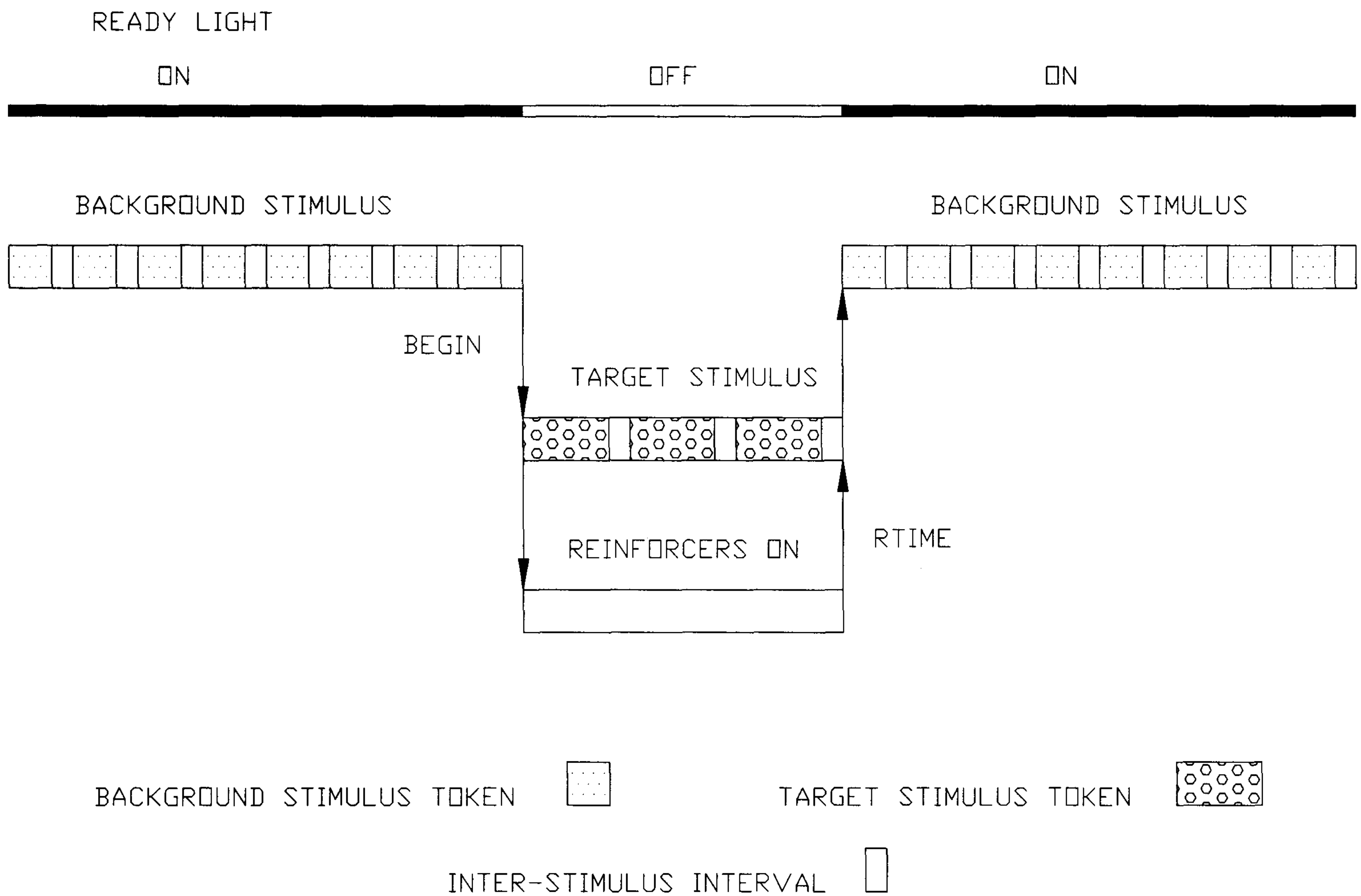
HEADTURN STATE DIAGRAM
TRAINING STAGE #1



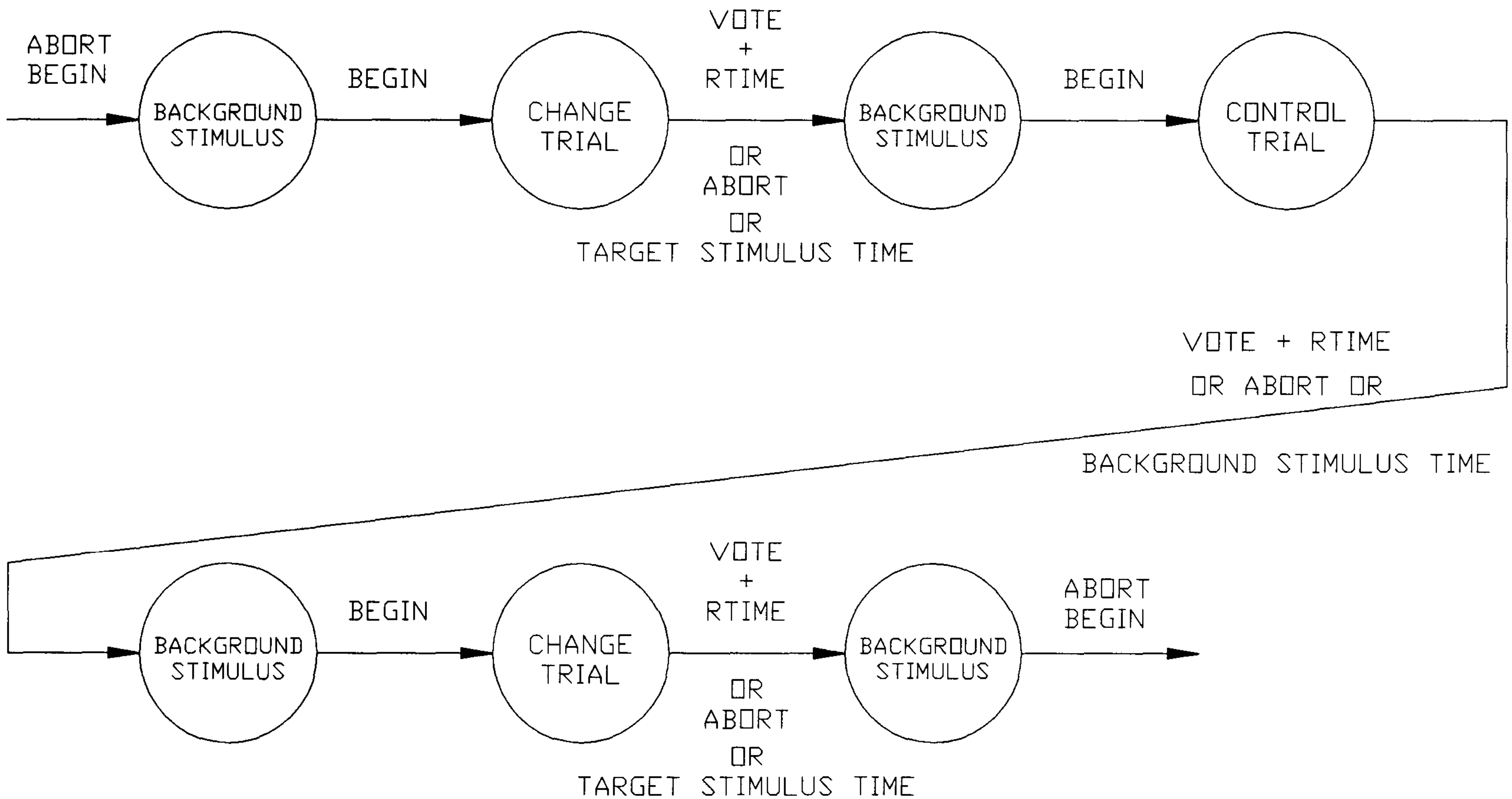
HEADTURN STATE DIAGRAM
TRAINING STAGE #2



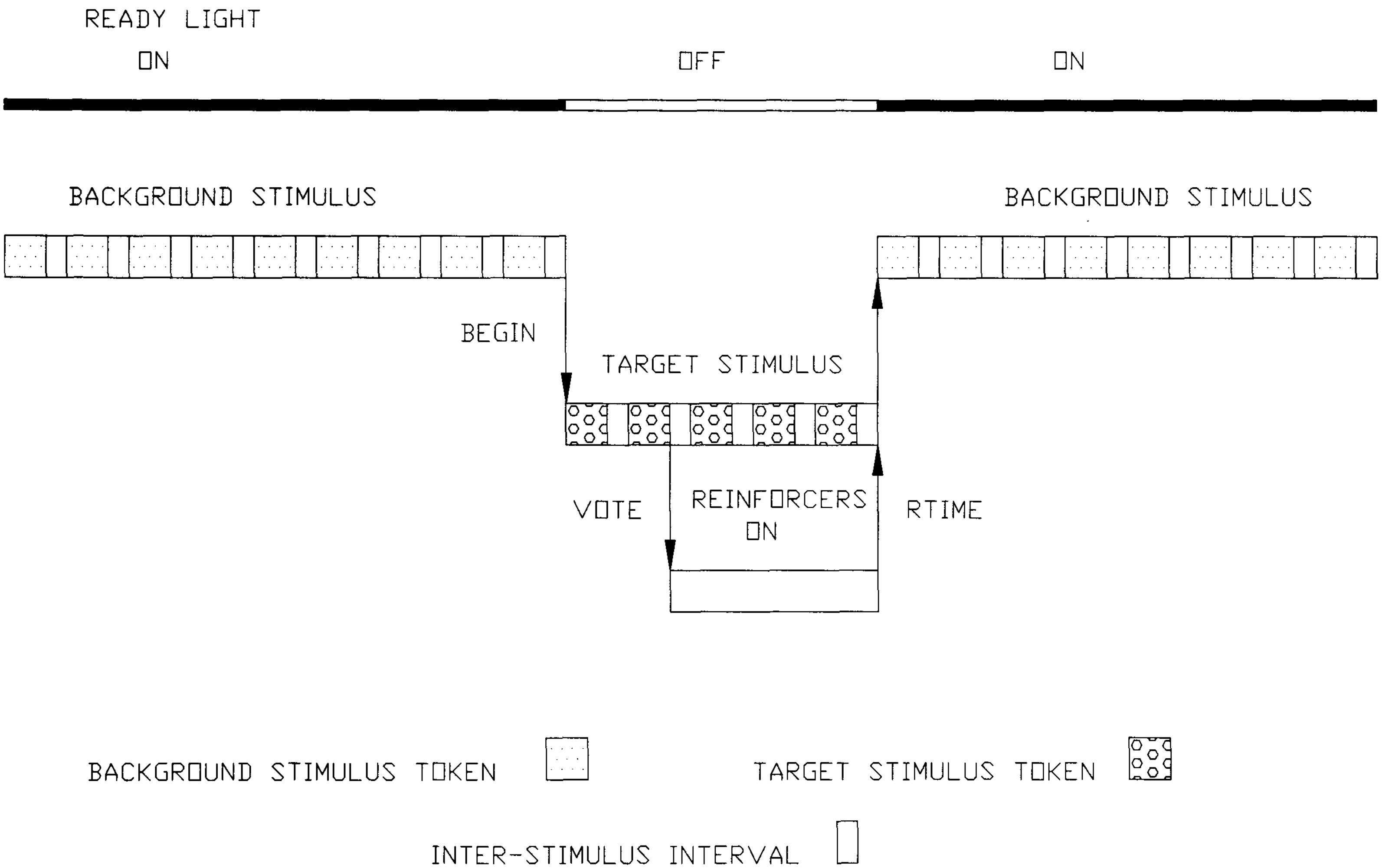
HEADTURN STATE DIAGRAM
TRAINING TRIAL



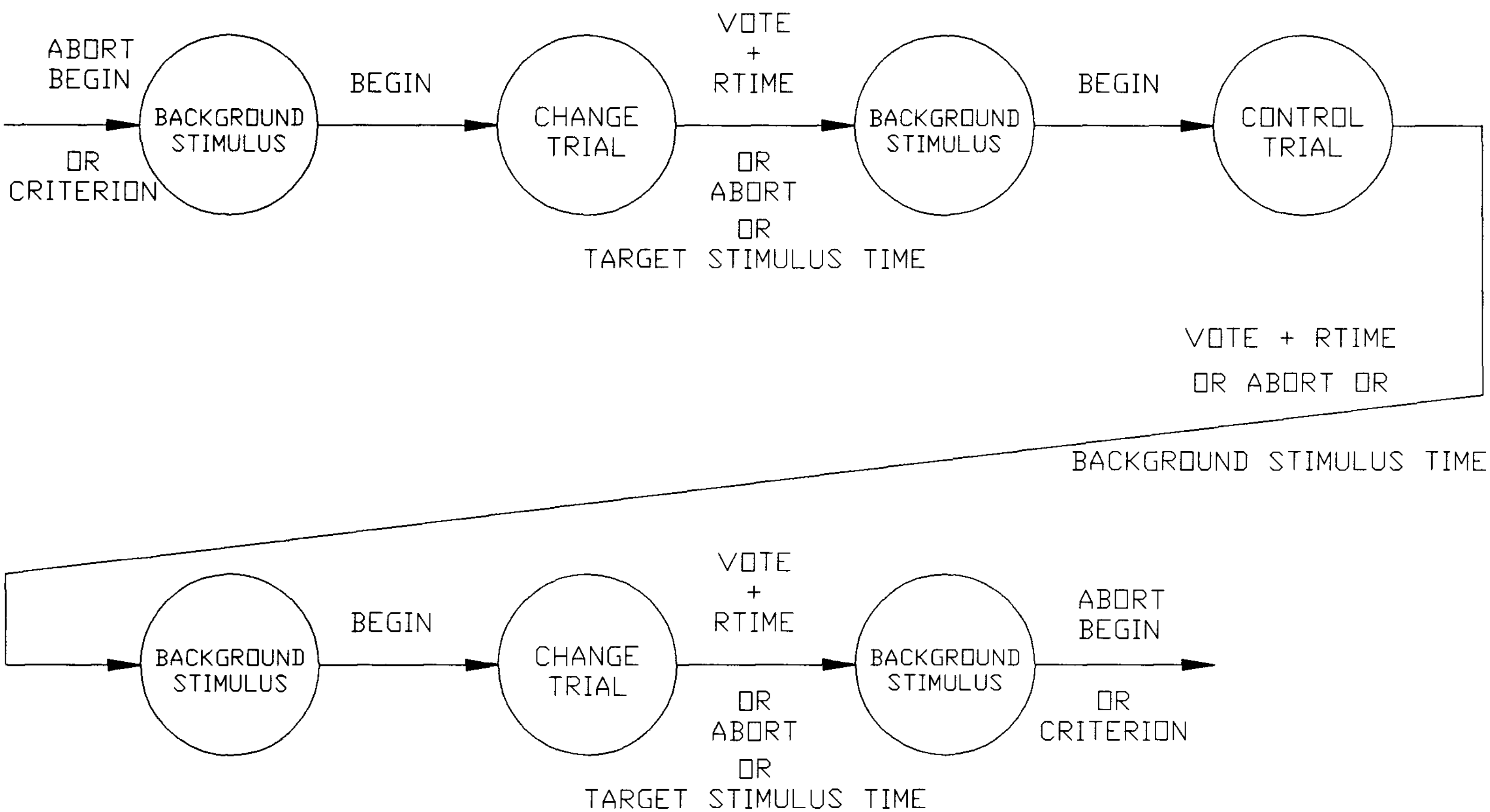
HEADTURN STATE DIAGRAM
CONDITIONING STAGE



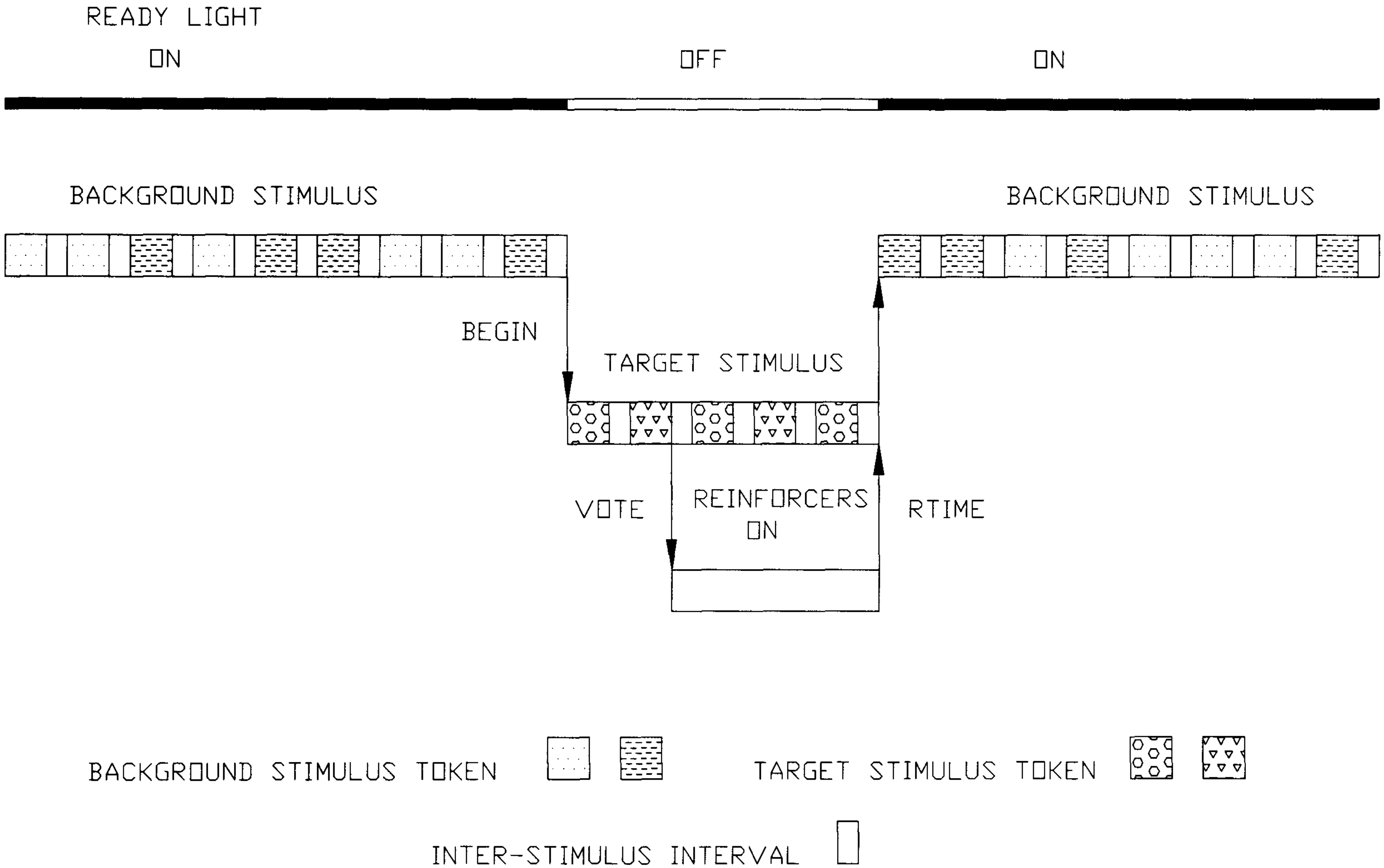
HEADTURN STATE DIAGRAM
CONDITIONING CHANGE TRIAL



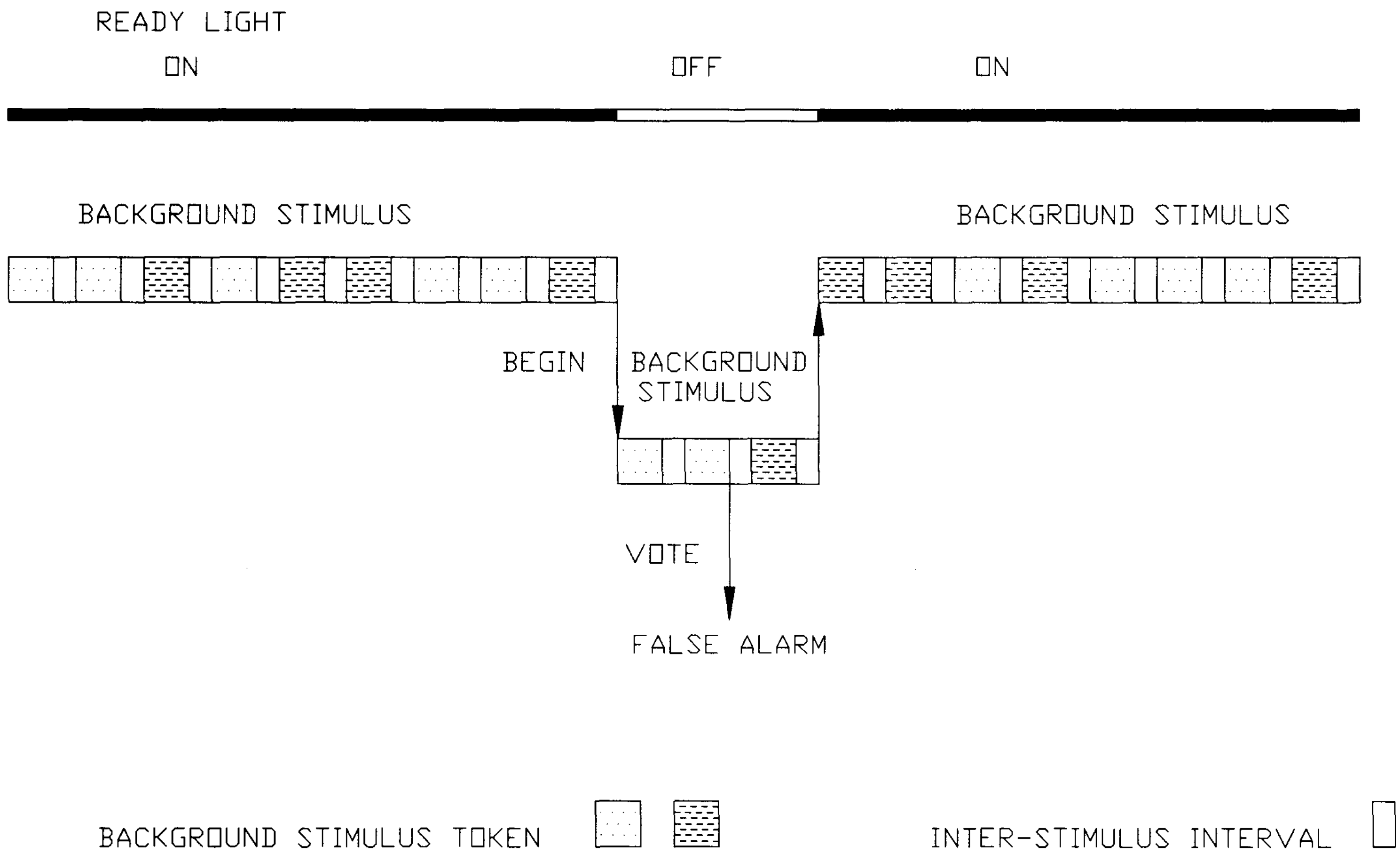
HEADTURN STATE DIAGRAM
ALL TESTING STAGES



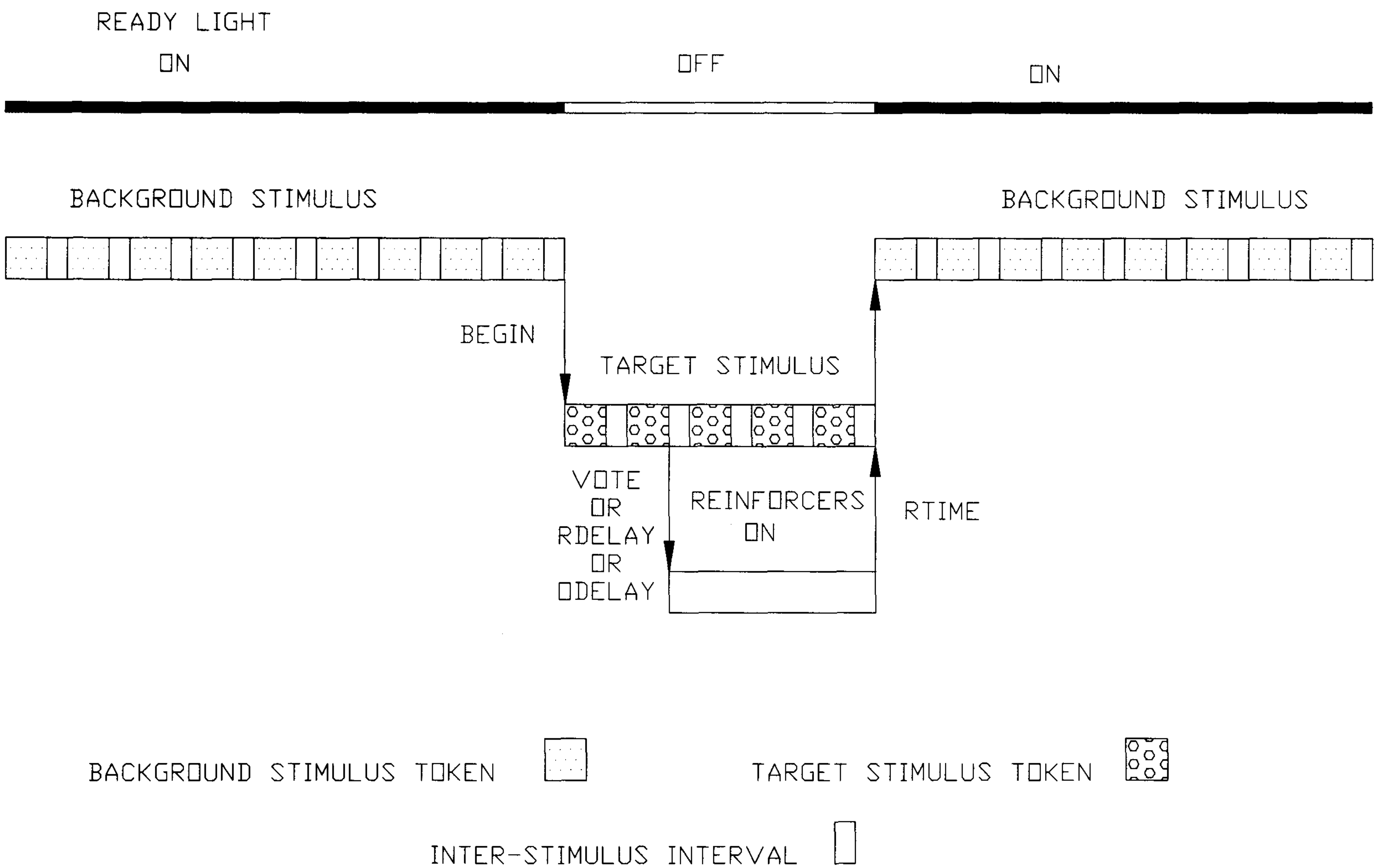
HEADTURN STATE DIAGRAM
TESTING CHANGE TRIAL



HEADTURN STATE DIAGRAM
CONTROL TRIAL



HEADTURN STATE DIAGRAM
RE-TRAINING TRIAL



Appendix C

Supplementary Installation Instructions

The purpose of these supplementary instructions is not to replace the installation instructions that come with the original Headturn package but to supplement them. There are two ways in which this is done. First, a simplified overall picture of the system is presented. Second, specific answers are provided to most of the questions asked by the Headturn installation instructions. The goal is to allow a user, new to Headturn, to get the system installed and working easily in a standard mode. Later, changes can be made if the system is not exactly the way the user wants it to be.

The following typographic conventions are used in this appendix

1. Courier bold typeface indicates items that must be typed by the user.

   for example

   `copy AUTOEXEC.BAT AUTOEXEC.SAV`

2. All capitals indicates a file or directory name.

   for example

   C:\ETC\INFO.

   The MSDOS operating system is not case sensitive. File and directory names can be entered in either upper or lower case.
   All capitals may also be used for abbreviations.

   for example

   MSDOS

3. Italic script indicates an item that will be displayed on the screen by one of the installation programs.

   for example

   Is this installation for a hard drive?

The Headturn package requires the following hardware and software environment.

1. An IBM PC compatible computer. The package will run on a PC or XT (8088) class machine but there will be significant waits between stages for disk access. An AT (80286) class or better machine is recommended.

2. MSDOS or PCDOS 3.0 or later. Check for this by typing **`ver`** at the DOS prompt.

3. A hard disk with a minimum of 2.5 megabytes of free space. The complete headturn package requires 1.8 megabytes. The remaining disk space is required for sound files and for the output files from the package. Check this by typing **`chkdsk`** at the DOS prompt. If you get a *Bad command or file name* message, try moving to the directory in which the CHKDSK.COM file is stored or inserting the diskette with the CHKDSK.COM file.

4. At least 256K of main processor memory. This can be checked by entering **`chkdsk`**.

5. A Data Translation DT2801A analog/digital I/O board. The board can be installed in the expansion bus of the PC by anyone familiar with PC hardware.

6. A serial port. This can be checked by looking at the back of the computer. A serial port will have either nine or twenty-five pins sticking out. All other nine or twenty-five pin connectors on the back of the computer will have holes rather than pins. The button box is connected to the serial port.

There are five software modules to be installed:

1. Headturn - main headturn program
2. SWFT - digitizing program
3. SSP - Standard Support Programs
4. DT_2801A- Interface to the A/D board
5. PC Video - Graphics interface files

The installation of each of these modules is done separately from the installation of the other four.

Initial Preparation

1. Enter the command

   **cd **

   to move to the root directory.

2. Enter the command

   copy AUTOEXEC.BAT AUTOEXEC.SAV

   to save your AUTOEXEC.BAT file. The installation program automatically modifies the file AUTOEXEC.BAT which is not always desirable.

3. Enter the command

   md \HEADTURN

   to create a directory.

4. Enter the command

   cd \HEADTURN

   to move to the directory.

5. Enter the commands

   md HEADERS
   md SSP
   md VIDEO
   md PROG
   md DATA

   to create a group of directories for the headturn programs and data. These particular names aren't mandatory, but they do provide a convenient convention. Other names can be used but they must be consistently referenced.

Headturn Installation

1. Put the headturn disk in drive A and enter

   `A:`

2. Enter the command

   `install`.

3. If you follow these supplementary instructions carefully, you can disregard all of the instructions on the screen. Simply keep pressing the Enter key until the first question appears.

4. *Is this installation for a hard drive?* **`Y`**

5. *Where do you want the utility programs?* **`C:\HEADTURN\PROG`**

6. *Would you like the termset data files in the same directory?* **`Y`**

7. *Would you like to run termset to set up a terminal configuration?* **`N`**

   This question will not be asked if the installation procedure finds a NIRVOSYS.PSF file on your hard disk. For a new installation this question will always be asked. The reason for saying "no" at this time is that unless you are actually in the directory that contains the termset command, the installation procedure will be unable to find it and so the terminal set up will fail.

8. *Where would you like to put the headturn programs?* **`C:\HEADTURN\PROG`**

9. *Do you want the files for HTANAL1 in the same place?* **`Y`**

10. *Where do you want the info file?* **`C:\ETC`**

    It appears that the info file must be kept in this directory.

SWFT Installation

1. Put the SWFT disk in drive A and enter

   `A:`

2. Enter the command

   `install`.

3. If you follow these supplementary instructions carefully, you can disregard all of the instructions on the screen. Simply keep pressing the Enter key until the first question appears.

4. *Is this installation for a hard drive?* **`Y`**

5. *Where do you want the swft programs?* **`C:\HEADTURN\PROG`**

6. *Where are the character font data files located?* **`C:\HEADTURN\PROG`**

7. *Where do you want the info file?* **`C:\ETC`**

DT_2801 Installation

1. Put the PC DT 2801A disk in drive A and enter

   A:

2. Enter the command

   install.

3. If you follow these supplementary instructions carefully, you can disregard all of the instructions on the screen. Simply keep pressing the Enter key until the first question appears.

4. *Is this installation for a hard drive?* **Y**

5. *Where would you like to put the DT2801 speech interface?* **C:\HEADTURN\PROG**

6. *Does your board have the standard jumper settings?* **Y**

   The DT2801A board comes from the factory jumpered to use I/O address 02ECH and DMA channel 1. Headturn uses these settings by default. If some other device in your computer conflicts with these settings, you must select different settings using the Data Translation documentation and the Headturn installation instructions.

7. *Where do you want the info file?* **C:\ETC**

PC Video Installation

1. Put the PC Video disk in drive A and enter

   `A:`

2. Enter the command

   `install`.

3. If you follow these supplementary instructions carefully, you can disregard all of the instructions on the screen. Simply keep pressing the Enter key until the first question appears.

4. *Is this installation for a hard drive?* **`Y`**

5. *Where do you want the PC VIDEO graphics interface?* **`C:\HEADTURN\VIDEO`**

6. Select the video device that matches your system.

7. *Where do you want the info file?* **`C:\ETC`**

SSP Installation

1. Put the SSP disk #1 in drive A and enter

   `A:`

2. Enter the command

   `install`.

3. If you follow these supplementary instructions carefully, you can disregard all of the instructions on the screen. Simply keep pressing the Enter key until the first question appears. Put disks #2 and #3 in drive A when they are requested.

4. *Is this installation for a hard drive?* **`Y`**

5. *Where do you want to put the SSP programs?* **`C:\HEADTURN\SSP`**

6. *Where do you want to put include files?* **`C:\HEADTURN\HEADERS`**

7. *Where do you want to put character font data files?* **`C:\HEADTURN\PROG`**

8. *Do you want to install sample graphics programs?* **`Y`**

9. *Do you want them put in the same place as the SSP programs?* **`Y`**

10. *Where do you want the info file?* **`C:\ETC`**

Terminal Setup

This procedure is done after the rest of the installation is done.

1. Enter the command

   `cd \HEADTURN\PROG`

   to go to the directory

2. Enter the command

   `termset`.

3. *Which (disk) directory should nirvosys.psf be in?* **`C:\HEADTURN\PROG`**

4. *Do you want to:* **`1`** (1 is the menu choice to select a terminal configuration.)

5. Enter
   `IBM-PC`

6. *Do you want to make any changes to this configuration?* **`N`**

7. *Do you wish to test this configuration?* **`N`**

8. *What name do you want to give to this new terminal configuration?* **`IBM-PC`**

9. *Do you want to:* **`4`** (4 is the menu choice to exit.)

Data Setup and Program Start

1. Enter the command

   `cd \`

   to move to the root directory.

2. Enter the command

   `copy AUTOEXEC.SAV AUTOEXEC.BAT`

   to restore your AUTOEXEC.BAT file.

3. Enter the command

   `cd \HEADTURN\DATA`

   to move to the directory.

4. Create a batch file called HEADSTRT.BAT with the following contents:

```
set savepath=%path%
set info_file = C:\ETC\INFO
path C:\HEADTURN\GRAPHICS;C:\HEADTURN\SSP;C:\HEADTURN\PROG;C
:\HEADTURN
headturn
set info_file =
path %savepath%
```

5. Copy the stimulus files to this directory.

6. Create the stimulus text file in this directory.

7. Create the parameter file in this directory.

8. Enter the command

   `headstrt`

   from the C:\HEADTURN\DATA directory to start the Headturn system.

Appendix D

Switch Test Results

This appendix contains the results of the switch testing. The following are included:

1. Profiles of typical switch presses.
2. Statistics for all of the switch types.
3. A single example of the trial-by-trial results.

The following terms are used in the tables of statistics:

1. **Number of Cycles** is the number, less one, of state changes measured for the switch when pressed or released.
2. **Elapsed Time** is the time measured for the switch to change from one steady state to the other when pressed or released.
3. **Press Time** is the length of time the switch remained in a steady state. This measure was taken in situations where the switch was being pressed or released as quickly as possible. and was intended to explore the shortest possible time the switch would remain in a steady state during a switch press.
4. **Down – Up** refers to the transition period between a steady down and a steady up state.

5. **Up - Down** refers to the transition period between a steady up and a steady down state.
6. **Rapid** refers to a switch press in which the switch is presses and released as rapidly as possible with the minimum delay between presses.
7. **Quick** refers to a switch press in which the switch is presses and released as rapidly as possible with time allowed between presses for the switch to settle.
8. **Slow** refers to a switch press in which both the press and release are done slowly to allow the switch to settle in both the up and the down states.
9. **Rapid Down** is a measurement of the time the switch was in the down state during a rapid switch press.
10. **Rapid Up** is a measurement of the time the switch was in the up state during a rapid switch press.
11. **Quick Down** is a measurement of the time the switch was in the down state during a quick switch press.
12. **Mean−12** is the arithmetic mean for the measurements done using the 286−12 machine.
13. **Mean−33** is the arithmetic mean for the measurements done using the 486−33 machine.
14. **Mean−all** is the arithmetic mean for the measurements done using both machines.
15. **sd−12** is the standard deviation for the measurements done using the 286−12 machine.
16. **sd−33** is the standard deviation for the measurements done using the 486−33 machine.

17. **sd–all** is the standard deviation for the measurements done using both machines.

15. **max–12** is the maximum value for the measurements done using the 286–12 machine.

16. **max–33** is the maximum value for the measurements done using the 486–33 machine.

17. **max–all** is the maximum value for the measurements done using both machines.

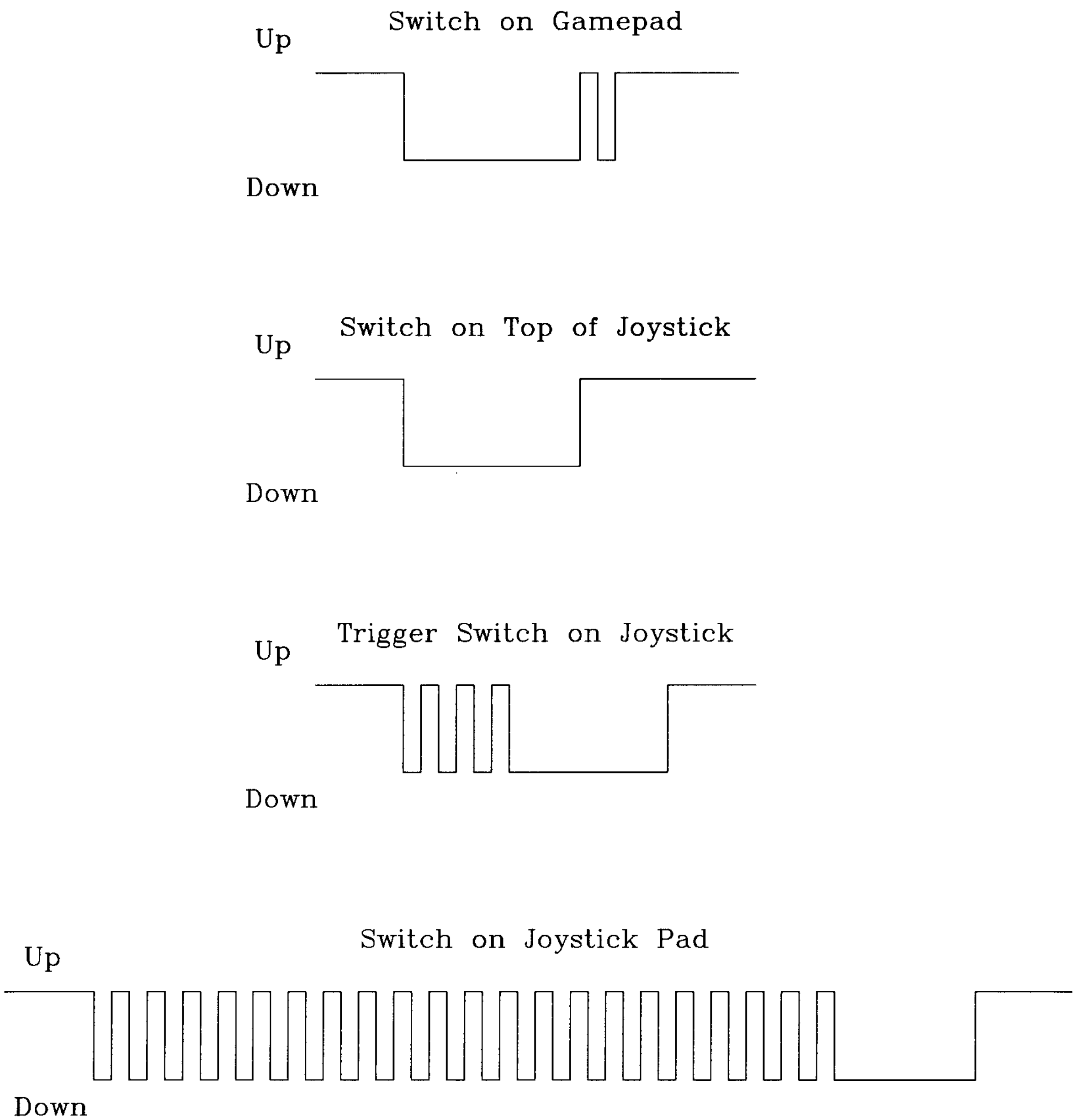
18. **min–12** is the minimum value for the measurements done using the 286–12 machine.

19. **min–33** is the minimum value for the measurements done using the 486–33 machine.

20. **min–all** is the minimum value for the measurements done using both machines.

For the table of trial-by-trial results, each individual switch press occupies a row in the table. The following terms are used in this table:

1. **Up** is the time the switch was measured in the up state.
2. **Down** is the time the switch was measured in the down state.



Profile of Typical Switch Presses

Switches on Game Pad - Number of Cycles

Down - Up Statistics

	Rapid	Quick	Slow	All
mean-12	0	0	1	1
mean-33	2	1	2	1
mean-all	1	1	1	1
sd-12	1	1	2	1
sd-33	1	1	2	2
sd-all	1	1	2	2
max-12	2	2	8	8
max-33	7	4	9	9
max-all	7	4	9	9

Switches on Game Pad - Elapsed Time

Down - Up Statistics

	Rapid	Quick	Slow	All
mean-12	6	3	27	12
mean-33	12	5	17	11
mean-all	9	4	22	12
sd-12	10	8	55	34
sd-33	13	8	24	17
sd-all	12	8	43	27
max-12	32	35	309	309
max-33	69	36	91	91
max-all	69	36	309	309

Press Time Statistics

	Rapid Down	Rapid Up	Quick Down
mean-12	64739	63355	65080
mean-33	53013	57337	54666
mean-all	58876	60346	59873
sd-12	29699	13334	13488
sd-33	17726	7431	12375
sd-all	25221	11205	13951
min-12	30058	42262	40735
min-33	35251	43985	32695
min-all	30058	42262	32695

Switches on Top of Joystick - Number of Cycles

	Down - Up Statistics				Up - Down Statistics			
	Rapid	Quick	Slow	All	Rapid	Quick	Slow	All
mean-12	0	0	0	0	0	0	0	0
mean-33	0	0	0	0	0	0	0	0
mean-all	0	0	0	0	0	0	0	0
sd-12	0	0	0	0	0	0	0	0
sd-33	0	0	0	0	0	0	0	0
sd-all	0	0	0	0	0	0	0	0
max-12	0	0	0	0	0	0	0	0
max-33	0	1	0	1	0	1	0	1
max-all	0	1	0	1	0	1	0	1

Switches on Top of Joystick - Elapsed Time

	Down - Up Statistics				Up - Down Statistics			
	Rapid	Quick	Slow	All	Rapid	Quick	Slow	All
mean-12	0	0	0	0	0	0	0	0
mean-33	0	9	0	3	0	0	0	0
mean-all	0	5	0	2	0	0	0	0
sd-12	0	0	0	0	0	0	0	0
sd-33	0	41	0	24	0	2	0	1
sd-all	0	30	0	17	0	2	0	1
max-12	0	0	0	0	0	0	0	0
max-33	0	190	0	190	0	10	0	10
max-all	0	190	0	190	0	10	0	10

Press Time Statistics

	Rapid Down	Rapid Up	Quick Down
mean-12	78232	60289	67953
mean-33	41668	67524	42411
mean-all	59950	63907	55182
sd-12	13834	8609	14476
sd-33	16349	5465	12740
sd-all	23740	8067	18682
min-12	58903	40990	50872
min-33	17158	55897	14070
min-all	17158	40990	14070

Trigger Switch on Joystick - Number of Cycles

	Down - Up Statistics				Up - Down Statistics			
	Rapid	Quick	Slow	All	Rapid	Quick	Slow	All
mean-12	0	0	0	0	2	3	3	3
mean-33	0	0	1	0	2	3	3	3
mean-all	0	0	0	0	2	3	3	3
sd-12	1	0	0	1	1	1	1	1
sd-33	0	1	2	1	1	1	0	1
sd-all	1	1	2	1	1	1	1	1
max-12	3	1	0	3	2	4	5	5
max-33	0	2	7	7	3	4	3	4
max-all	3	2	7	7	3	4	5	5

Trigger Switch on Joystick - Elapsed Time

	Down - Up Statistics				Up - Down Statistics			
	Rapid	Quick	Slow	All	Rapid	Quick	Slow	All
mean-12	364	512	0	292	184	311	281	259
mean-33	0	373	1017	463	165	259	232	219
mean-all	182	443	509	378	174	285	257	239
sd-12	776	1248	0	876	71	27	64	79
sd-33	0	1118	2182	1477	39	44	31	55
sd-all	578	1187	1625	1217	58	45	56	71
max-12	2425	4156	0	4156	221	332	406	406
max-33	0	3727	6852	6852	189	344	255	344
max-all	2425	4156	6852	6852	221	344	406	406

Press Time Statistics

	Rapid Down	Rapid Up	Quick Down
mean-12	93608	76162	113480
mean-33	76786	45170	94971
mean-all	85197	60666	104225
sd-12	9407	33558	7999
sd-33	4227	5790	21183
sd-all	11132	28635	18494
min-12	80371	45145	98524
min-33	67561	36244	70288
min-all	67561	36244	70288

Switches on Joystick Pad - Number of Cycles

	Down - Up Statistics				Up - Down Statistics			
	Rapid	Quick	Slow	All	Rapid	Quick	Slow	All
mean-12	0	0	0	0	13	13	14	14
mean-33	0	0	0	0	20	22	21	21
mean-all	0	0	0	0	17	18	17	17
sd-12	0	0	0	0	2	2	3	2
sd-33	0	0	0	0	2	4	3	3
sd-all	0	0	0	0	4	5	4	5
max-12	1	1	1	1	17	18	18	18
max-33	1	0	1	1	25	32	27	32
max-all	1	1	1	1	25	32	27	32

Switches on Joystick Pad - Elapsed Time

	Down - Up Statistics				Up - Down Statistics			
	Rapid	Quick	Slow	All	Rapid	Quick	Slow	All
mean-12	2	14	13	10	947	1007	995	983
mean-33	5	0	15	7	934	998	998	977
mean-all	4	7	14	8	941	1003	997	980
sd-12	8	62	43	44	88	101	86	95
sd-33	12	0	50	30	136	220	146	174
sd-all	10	44	47	38	115	171	120	141
max-12	37	283	190	283	1118	1152	1156	1156
max-33	47	0	230	230	1193	1504	1435	1504
max-all	47	283	230	283	1193	1504	1435	1504

Press Time Statistics

	Rapid Down	Rapid Up	Quick Down
mean-12	62848	55811	61575
mean-33	52595	59438	71444
mean-all	57722	57624	66509
sd-12	12966	11674	20476
sd-33	9726	10213	12042
sd-all	12555	11117	17507
min-12	40373	42088	33898
min-33	30837	35112	48873
min-all	30837	35112	33898

80486-33 Machine Used
First Button on Joystick Pad

10 Rapid Fire Presses - Microseconds

Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down
4	2	7	7	4	15	14	4	6	4	52	5	11
4	4	7	6	4	15	15	2	6	4	67	5	14
4	2	9	6	4	13	15	2	6	4	81	2	5
4	2	9	6	5	13	15	2	6	4	84	5	10
4	4	7	6	4	13	15	4	5	5	67	2	10
4	2	9	6	4	13	15	4	5	5	67	2	15
5	2	7	7	4	13	15	4	5	5	84	4	10
4	2	9	7	4	13	15	2	6	4	67	2	10
4	2	7	7	4	13	15	2	6	4	84	4	10
4	2	7	7	4	13	15	2	6	4	85	2	11

10 Quick Presses - Microseconds

Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down
4	4	7	6	4	15	14	4	6	4	53	4	22
4	4	7	6	5	15	14	4	6	4	53	4	22
4	4	7	5	5	15	14	4	6	4	53	2	28
4	4	7	6	4	15	15	4	5	4	53	4	22
4	4	7	6	4	15	14	4	6	4	84	5	9
4	4	7	6	4	15	15	2	6	4	53	4	28
5	4	6	7	4	15	15	2	6	4	53	4	22
4	4	6	7	4	15	15	4	5	4	53	4	23
4	4	7	6	4	15	14	4	6	4	67	2	15
4	4	7	6	4	15	15	4	5	4	53	4	23

10 Slow Presses - Microseconds

Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down
4	2	7	7	4	15	14	4	6	4	67	4	15
4	4	7	6	4	15	15	2	6	4	53	2	12
4	2	9	6	4	15	15	4	5	4	68	2	10
4	4	6	7	4	13	15	2	6	4	53	2	14
4	4	7	6	5	13	15	4	5	4	67	4	14
4	2	9	6	4	15	14	4	6	4	53	2	12
4	4	6	7	4	13	15	2	6	4	53	5	16
4	4	7	6	4	15	15	4	5	4	53	4	23
5	2	7	6	5	12	16	2	6	4	68	2	10
5	2	7	7	4	13	15	2	6	4	67	5	14

80486-33 Machine Used
First Button on Joystick Pad - Second Page

10 Rapid Fire Presses - Microseconds

Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	Up
2	10	2	5	10	5	2	5	9	4	233
11	9	263	5	5	5	9	4	7	6	2
4	10	11	6	223	4	5	6	7	4	10
4	2	6	6	229	4	5	5	23	14	7
2	5	5	9	13	5	249	6	4	5	27
9	6	4	5	7	4	248	12	22	15	4
2	5	5	6	260	4	5	5	12	162	198
4	4	5	10	13	4	239	5	4	6	20
12	6	237	4	2	7	25	169	162	4	5
10	7	226	4	6	5	9	2	11	16	5

10 Quick Presses - Microseconds

Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	Up
4	4	4	5	2	5	7	10	208	5	5
2	5	5	9	5	14	197	5	4	6	7
6	7	15	5	207	5	2	9	7	2	16
4	5	4	9	9	10	211	5	5	5	7
4	14	226	5	5	5	20	19	4	152	152
5	7	10	9	203	4	6	5	7	4	16
4	5	4	9	10	9	214	5	5	5	9
2	4	6	7	10	9	206	5	5	5	6
7	7	5	2	228	5	5	6	6	4	17
2	5	5	9	5	2	2	9	221	4	6

10 Slow Presses - Microseconds

Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	Up
10	5	2	4	243	4	4	6	26	12	9
2	5	2	10	5	9	4	4	234	5	6
2	5	6	9	2	5	230	5	4	6	7
4	14	11	10	229	5	5	6	5	5	18
11	10	233	9	2	5	6	4	17	15	5
4	15	10	10	226	5	4	6	9	2	9
2	17	4	10	237	5	4	6	22	15	11
2	5	10	10	229	5	5	6	7	4	13
2	6	5	9	4	4	229	5	4	6	7
11	9	235	4	5	6	7	2	15	14	5

80486-33 Machine Used
First Button on Joystick Pad - Third Page

10 Rapid Fire Presses - Microseconds

Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down
4	5	6	26	5	4	5	9	5	2	139
6	2	150	2	2	165	4	4	7	6	4
17	2	165	140	4	4	7	5	5	15	14
150	4	4	150	5	4	6	7	4	15	150
22	2	128	180	12	7	4	11	252	4	50351
141	2	4	190	5	4	6	7	4	12	23
4	2	7	7	4	13	15	4	5	5	38
12	2	159	166	5	2	7	7	7	5	108
6	7	4	11	79	4	23	4	48095	6	20
161	147	4	4	7	6	4	16	14	2	6

10 Quick Presses - Microseconds

Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down
5	9	4	11	17	4	40	4	19	4	130
4	15	6	4	5	6	5	6	14	4	6
5	5	5	7	5	4	15	5	5	4	129
4	14	15	5	6	2	19	2	14	2	20
4	5	5	7	4	10	81	2	25	2	6
5	5	5	2	9	6	15	5	139	2	14
4	4	17	2	5	10	147	4	9	138	4
4	20	5	2	7	5	5	6	15	5	5
5	2	6	6	144	152	4	6	5	7	2
5	6	4	20	15	5	5	4	125	2	6

10 Slow Presses - Microseconds

Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down
141	147	4	5	5	7	4	13	22	5	84
5	5	4	21	14	7	131	2	11	149	4
4	17	5	4	6	4	6	6	15	4	128
5	2	9	5	5	5	15	4	115	2	17
5	5	150	124	4	6	5	6	4	15	22
6	2	5	6	15	4	15	5	147	102	4
146	10	4	130	4	4	7	6	4	13	22
15	6	150	4	10	125	4	4	7	6	4
4	15	6	4	5	7	6	2	23	2	124
6	2	151	130	4	5	6	6	4	15	15

80486-33 Machine Used
First Button on Joystick Pad - Fourth Page

10 Rapid Fire Presses - Microseconds

Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	Up
171	4	4	7	6	5	10	81	2	125	4
15	15	2	7	2	38	2	14	4	22	4
4	6	5	83	2	41	2	16	4	40373	53930
5	15	2	33	4	47068	5	42	44765		
43529										
2	22	2	31	4	149	47	50503	43297		
2	14	4	121	89	6	2	15	7	58119	2
4	53908	51583								
45190										
5	38	2	38	2	5	2	30	4	10	2

10 Quick Presses - Microseconds

Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	Up
85	14	6	5	6	22	2	47383	1327541		
2	5	4	17	4	125	6	16	20	5	9
2	12	105	4	2	7	5	5	12	45465	1045574
2	109	12	12	90	5	5	6	5	5	12
2	40808	1005074								
97	4	5	6	2	7	10	46388	997987		
6	5	6	4	14	15	2	6	5	53	4
4	6	2	113	2	15	9	9	83	5	4
14	22	5	52	2	23	2	64	5	50633	959398
2	11	4	4	126	5	4	6	6	4	15

10 Slow Presses - Microseconds

Up	Down	Up	Down	Up	Down	Up	Down	Up	Down	Up
4	15	2	42	2	905364	2	10	1208122		
5	5	9	2	11	72	2	819774	1023756		
130	4	5	6	6	4	16	15	2	6	4
120	5	4	6	6	4	15	21	2	630182	1348118
2	115	2	35	4	733963	1206517				
5	6	5	5	12	802861	1354694				
2	85	5	57	2	766015	1098727				
15	14	2	7	2	68	2	16	4	874991	78
4	2	121	4	5	5	6	4	13	22	4
2	6	4	67	4	16	2	40	2	6	2

80486-33 Machine Used
First Button on Joystick Pad - Fifth Page

10 Rapid Fire Presses - Microseconds

Down	Up	Down	Up	Down
67901	44763			
48237	50746			
14	45534			
16	4	33	2	63009

10 Quick Presses - Microseconds

Down	Up	Down	Up	Down	Up	Down	Up	Down	Up
40873	1035570								
33898	979085								
23	2	33	5	10	7	11	4	39300	918351
6	4	6	7	46238	1068941				
14	4	6	4	84	2	59	4	52138	

10 Slow Presses - Microseconds

Down	Up	Down	Up	Down	Up	Down	Up	Down	Up	Down
113	5	15	5	15	5	26	2	750221	2	14
152	1080781									
744073	1032464									
932005										

80486-33 Machine Used
First Button on Joystick Pad - Sixth Page

10 Slow Presses - Microseconds

Up

1205963

Appendix E

Assembler Code Used in Timing

The following three segments of assembler code were used to assess execution times.

1. The length of time to process a keyboard interrupt. This was code extracted from the system BIOS.
2. The timing profile of switch presses for switches attached to the game port. This was code written specifically for this purpose.
3. The length of time to process a timer interrupt. This was a mixture of BIOS and user supplied code.

Timing Analysis of BIOS Processing of Keyboard Interrupt

The IBM BIOS listing was used. A 386-33 with memory caching and the ROM BIOS shadowed to RAM was assumed for the best case timing. 300 nanosecond ROM was assumed for the worst case timing. The time calculated was to process the interrupt for a single lowercase character. The best case and worst case times are in nanoseconds.

Line #	Instruction	Clock Cycles	Best Case	Worst Case
1850	sti	3	90	360
	push	2	60	330
	push	2	60	330
	push	2	60	330
	push	2	60	330
	push	2	60	330
	push	2	60	330
	push	2	60	330
	push	2	60	330
	cld	2	60	330
1860	call	8	240	510
5489	push	2	60	330
	mov	2	60	330
	mov	2	60	330
	pop	4	120	390
5493	ret	11	330	600
1861	in	12	360	630
	push	2	60	330
	in	12	360	630
	mov	2	60	330
	or	2	60	330
	out	10	300	570
	xchg	3	90	360
	out	10	300	570
	pop	4	120	390
	mov	2	60	330
	cmp	2	60	330
	jnz	3	90	360
	jmp	8	240	510
	and	2	60	330
	push	2	60	330
	pop	4	120	390

Line #	Instruction	Clock Cycles	Best Case	Worst Case
	mov	2	60	330
	mov	4	120	390
	repne	56	1680	3040
	mov	2	60	330
	je	3	90	360
1889	jmp	8	240	510
1965	cmp	2	60	330
	jae	3	90	360
	test	5	150	420
1968	jz	8	240	510
1991	test	5	150	420
	jnz	3	90	360
1993	jmp	8	240	510
2078	test	5	150	420
2079	jz	8	240	510
2134	cmp	2	60	330
	jae	3	90	360
	test	5	150	420
2137	jz	8	240	510
2202	cmp	2	60	330
2203	jb	8	240	510
2207	mov	2	60	330
	dec	2	60	330
	xlat	5	150	420
	cmp	2	60	330
	je	3	90	360
	cmp	2	60	330
	je	3	90	360
	test	5	150	420
2227	jz	8	240	510
2254	mov	4	120	390
	mov	2	60	330
2256	call	8	240	510
1779	inc	2	60	330
	inc	2	60	330
	cmp	6	180	450
	jne	8	240	510
	ret	11	330	600
2257	cmp	6	180	450

Line #	Instruction	Clock Cycles	Best Case	Worst Case
	je	3	90	360
	mov	2	60	330
	mov	2	60	330
2261	jmp	8	240	510
1973	cli	3	90	360
	mov	2	60	330
	out	10	300	570
	pop	7	210	480
	pop	7	210	480
	pop	4	120	390
	pop	4	120	390
	pop	4	120	390
	pop	4	120	390
	pop	4	120	390
	pop	4	120	390
1985	iret	22	660	930
Totals		451	13530	38110

Code Used to Time Switch Presses

```
                                                            Clocks
                                                           286   486

pprl:   in    al,dx        ; Input the game port            5     12
        test  bp,01H       ; See if first transition        3     2
        jz    ppr4         ; It hasn't so do nothing        3     3

        add   si,1         ; Count the button presses       3     2
        jnc   ppr4         ; No need to increment high      11    11
        inc   di           ; Increment the high order

ppr4:   cmp   al,ah        ; See if button has changed      2     2
        jz    pprl         ; It hasn't                      3/9   3/9

        test  al,080H      ; Check for button 4 down        3     2
        jz    ppr3         ; Jump if it's a button down     3/11  3/11
        test  ah,080H      ; Check for button 4 before      3     2
        jnz   ppr2         ; Exit if previously up          3     3

ppr3:   or    bp,01H       ; Set the flag to count          3     2
        mov    ta[bx],di   ; Save the high order part       3     2
        add   bx,2         ; Point to next location         3     2
        mov    ta[bx],si   ; Save the low order part        3     2
        add   bx,2         ; Point to next location         3     2
        xor   si,si        ; Reset the loop counters        2     2
        xor   di,di                                         2     2
        mov   ah,al        ; Save the current state         2     2
        jmp   pprl         ; Do the next loop               9     9
```

Timing Analysis of Timer Interrupt

Instruction / Processor		Clock Cycles 8088	80286	80386	80486
int		71	25	37	28
sti		2	2	3	5
push	ds	14	3	2	3
push	ax	15	3	2	4
push	dx	15	3	2	4
call	DDS	23	8	8	3
push	ax	15	3	2	4
mov	ax,data	14	5	4	3
mov	ds,ax	2	2	2	3
pop	ax	12	5	4	4
ret		20	13	12	5
inc	timer_low	29	7	6	9
jnz	T4	16	9	9	3
cmp	timer_high,018H	20	6	5	4
jnz	T5	16	8	8	3
dec	motor_count	29	7	6	9
jnz	T6	16	9	9	3
int	1CH	71	24	37	32
push	ax	15	3	2	4
push	dx	15	3	2	4
mov	dx,0201H	4	2	2	1
in	al,dx	12	5	13	14
and	al,button_mask	15	7	6	4
cmp	al,prev_state	15	6	6	4
je	gpc3	16	8	8	3
pop	dx	12	5	4	5
pop	ax	12	5	4	5
iret		44	19	22	23
mov	al,EOI	4	2	2	1
out	020H,al	14	3	10	16
pop	dx	12	5	4	5
pop	ax	12	5	4	5
pop	ds	12	5	7	5
iret		44	19	22	23
Total Clock Cycles		658	244	276	251

For the 8088 and 80286 machines, it is necessary to add some time. These machines normally do not have a memory cache or shadow RAM. The sequence shown contains 23 instructions that will be executed from the BIOS. The ROM chips have an access time of 300 nanoseconds. An additional 90 nanoseconds per instruction must be added to the total time for the 8088. An additional 220 nanoseconds per instruction must be added to the total time for the 80286.

Processor	8088	80286	80386	80486
Processor Speed (mhz)	4.77	12	33	50
Microseconds/Interrupt	137.94	20.33	8.36	5.02
BIOS Adjustment	2.07	5.06	0	0
Total Time	138.01	25.39	8.36	5.02

	% of CPU used by Timer Interrupt			
Interrupt Frequency	8088	80286	80386	80486
$500\mu s$	27.60	5.08	1.67	1.00
1 ms	13.80	2.54	0.84	0.50
3 ms	4.60	0.85	0.28	0.17
5 ms	2.76	0.51	0.17	0.10
10 ms	1.38	0.25	0.08	0.05
20 ms	0.69	0.13	0.04	0.03

Appendix F

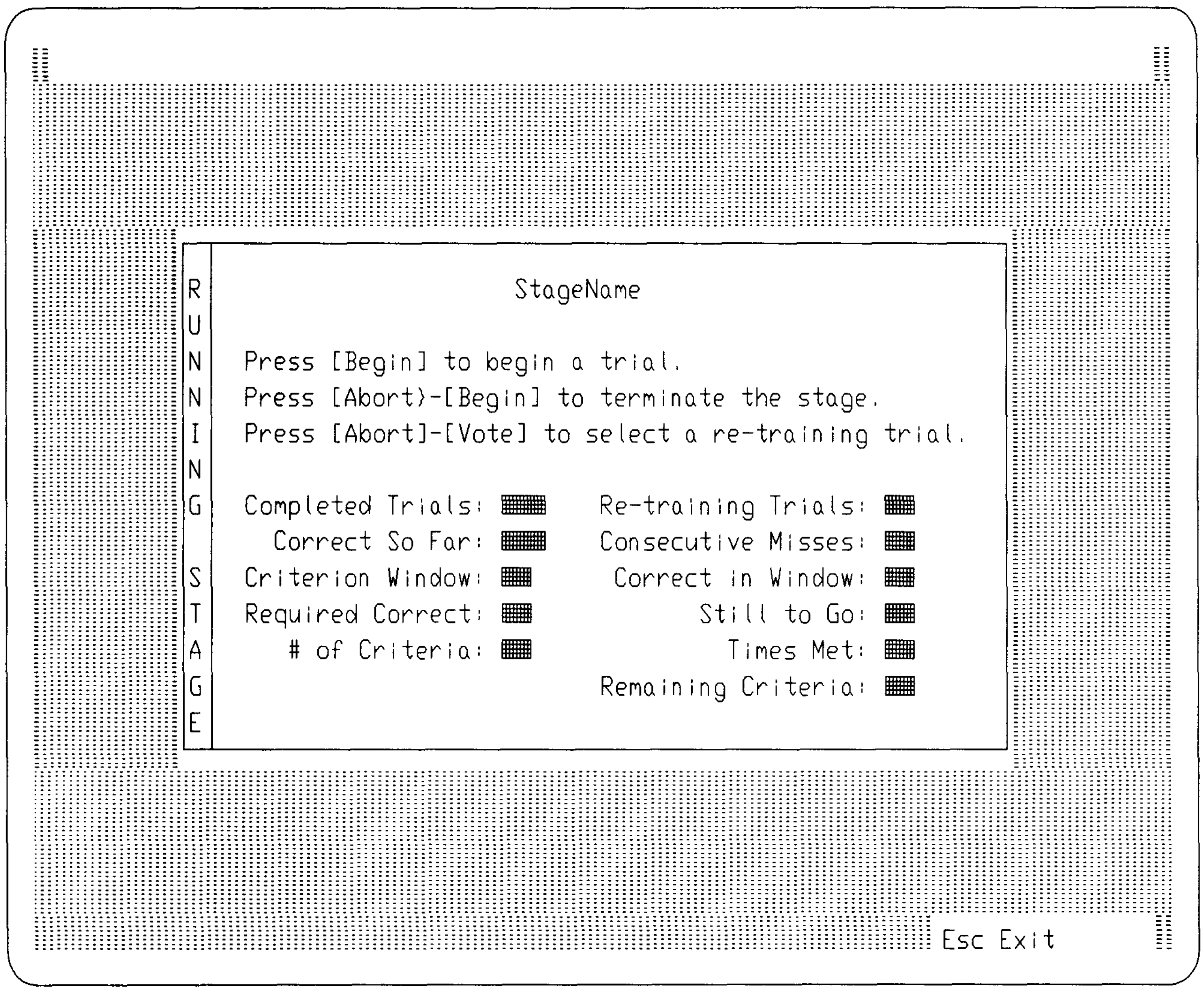
Screen Diagrams for System Implementation

The following diagrams are graphical representations of the screens for the new implementation of the Headturn procedure.

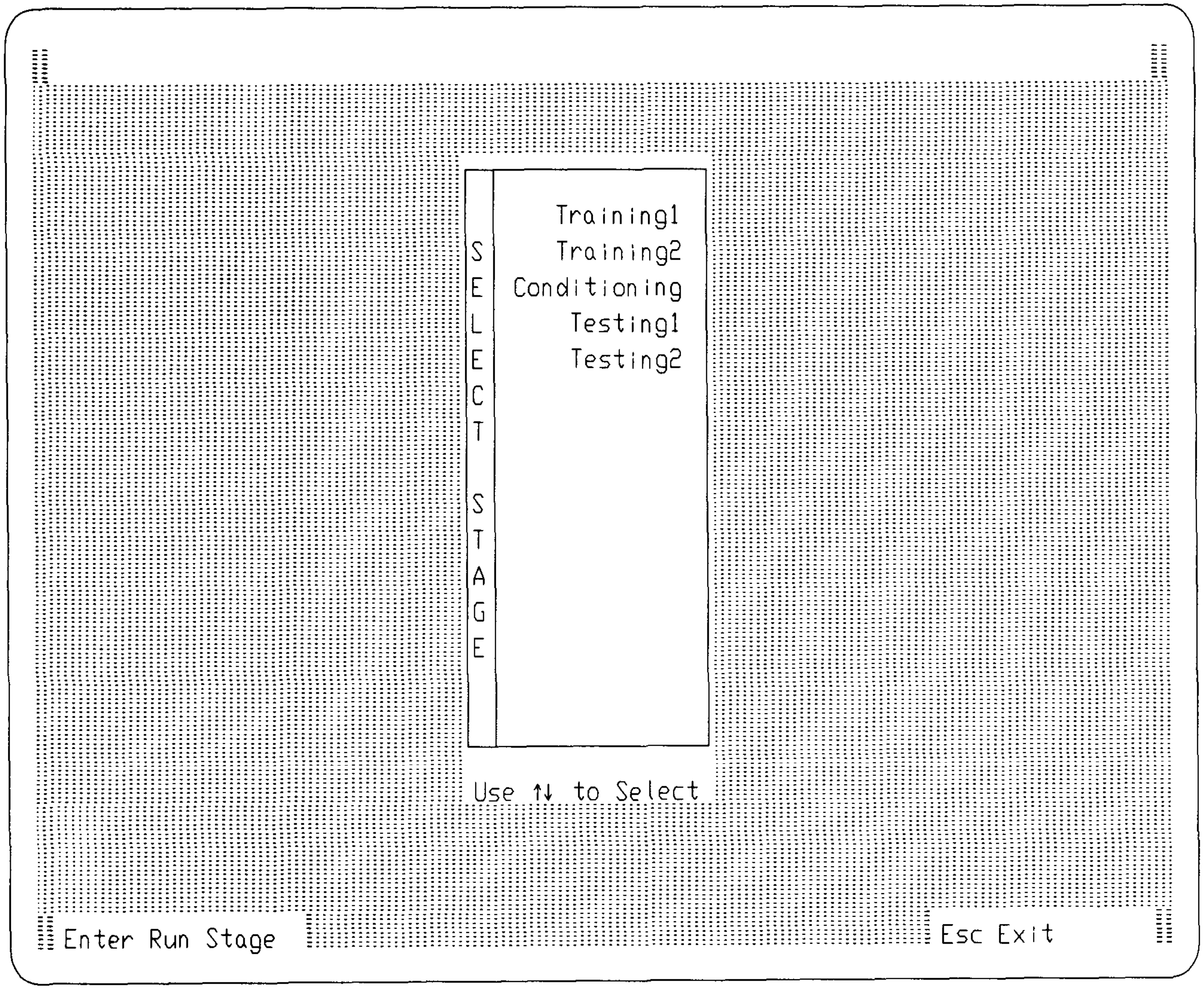
MAINSCRN


MAIN
Run Experiment
Edit Configuration File
Edit General Parameters
Edit Session
Edit Stage
Edit Schedule
Edit Protocol & Stimuli
Use ↑↓ to Select
Version 1.0 Copyright (c) 1993 Brian Pidcock.
Enter Run Program
Esc Exit

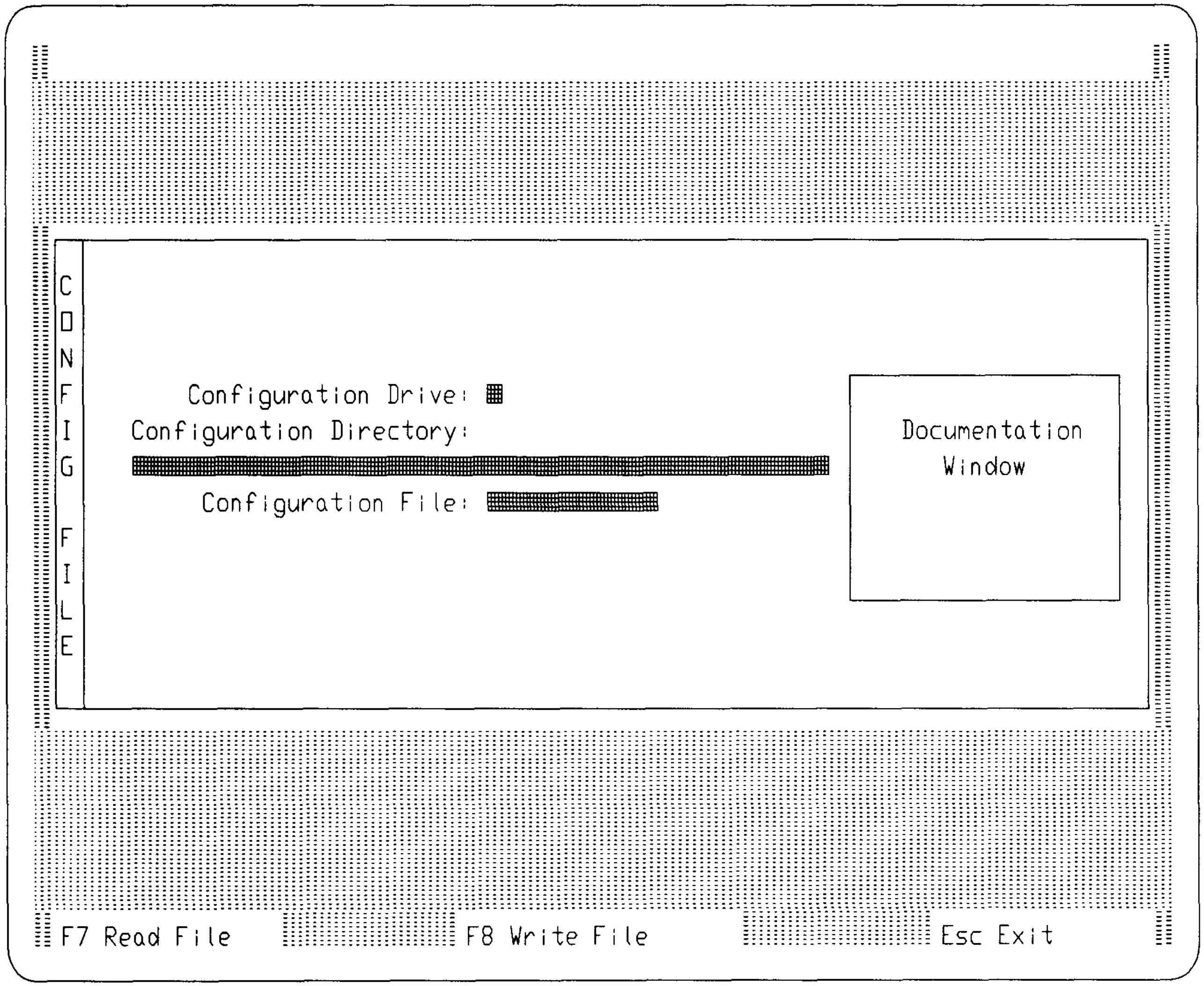
RUNGSCRN


RUNNING STAGE
StageName
Press [Begin] to begin a trial.
Press [Abort}-[Begin] to terminate the stage.
Press [Abort]-[Vote] to select a re-training trial.
Completed Trials:
Correct So Far:
Criterion Window:
Required Correct:
# of Criteria:
Re-training Trials:
Consecutive Misses:
Correct in Window:
Still to Go:
Times Met:
Remaining Criteria:
Esc Exit

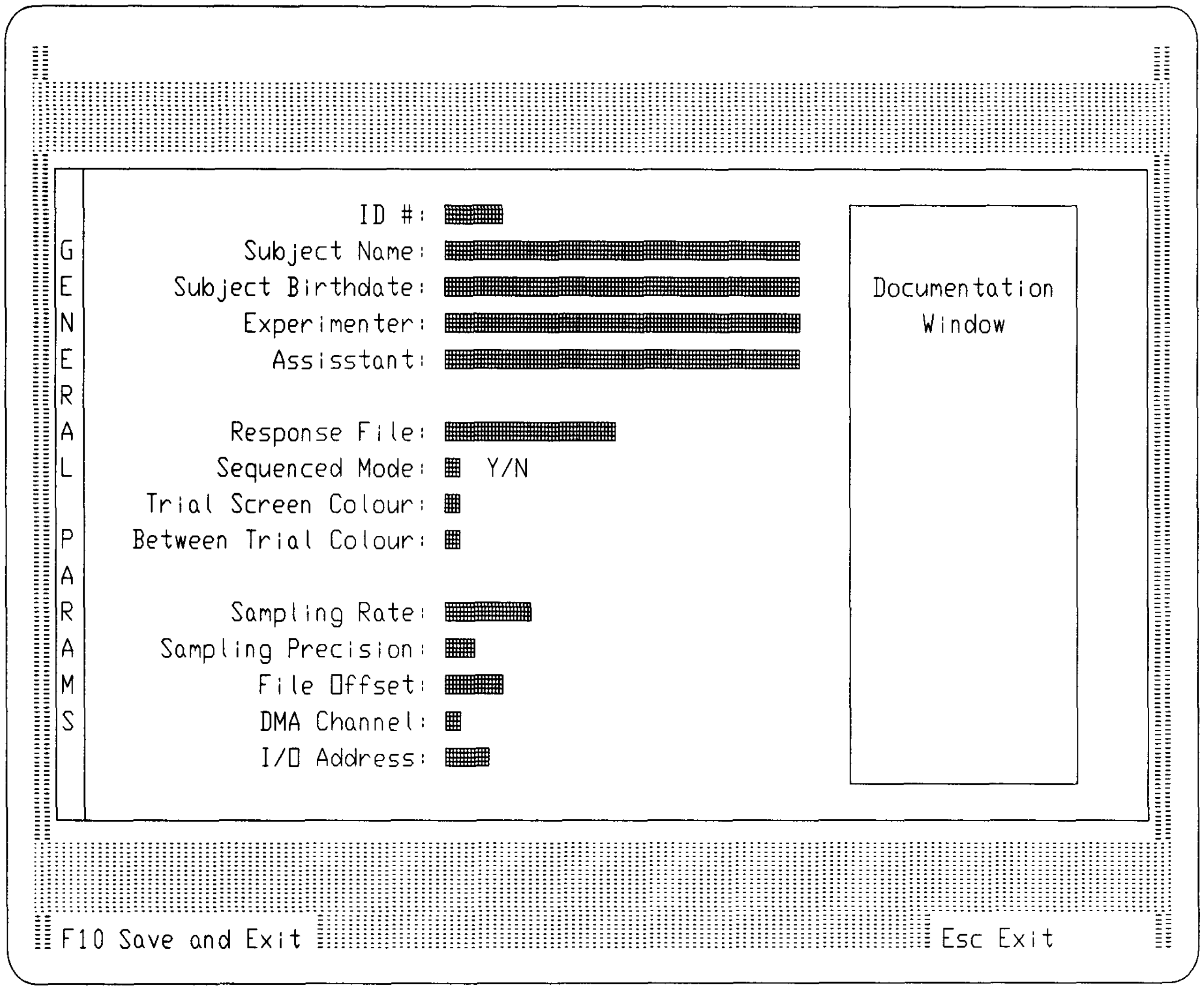
SLCTSCRN


SELECT STAGE
Training1
Training2
Conditioning
Testing1
Testing2
Use ↑↓ to Select
Enter Run Stage
Esc Exit

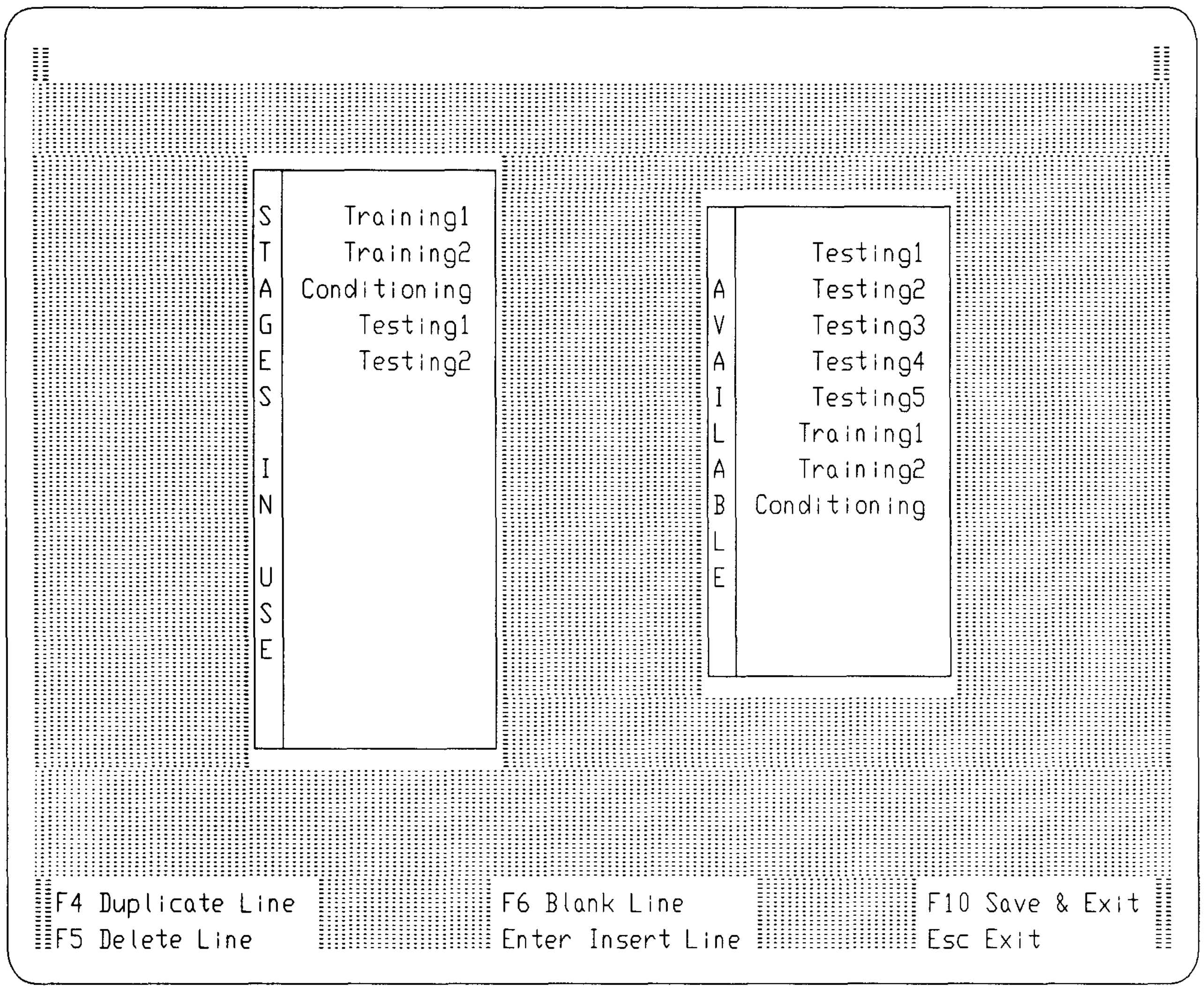
FILESCRN


CONFIG FILE
Configuration Drive:
Configuration Directory:
Configuration File:
Documentation Window
F7 Read File
F8 Write File
Esc Exit

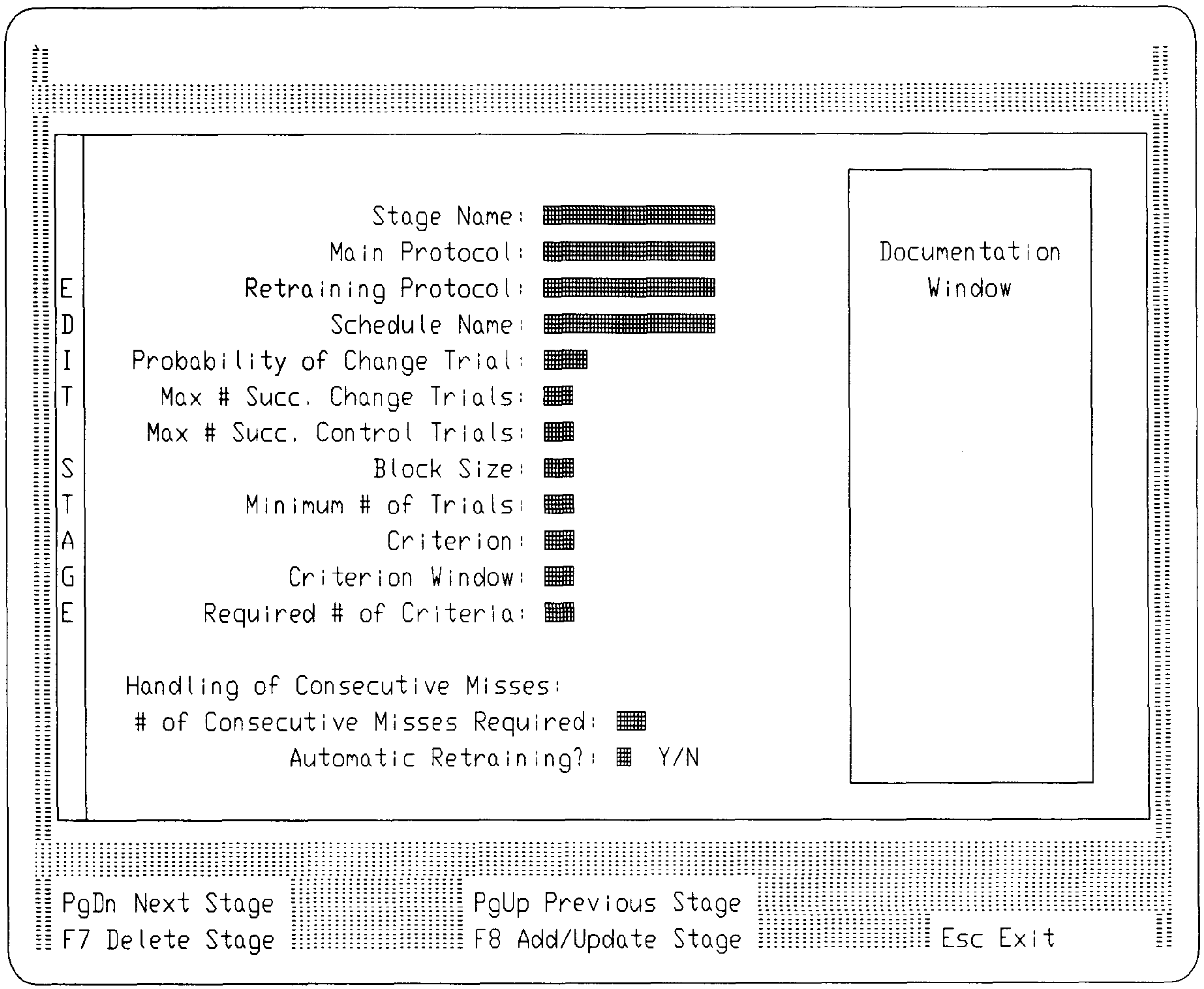
GENRLSCN


GENERAL PARAMS
ID #:
Subject Name:
Subject Birthdate:
Experimenter:
Assisstant:
Response File:
Sequenced Mode: Y/N
Trial Screen Colour:
Between Trial Colour:
Sampling Rate:
Sampling Precision:
File Offset:
DMA Channel:
I/O Address:
Documentation Window
F10 Save and Exit
Esc Exit

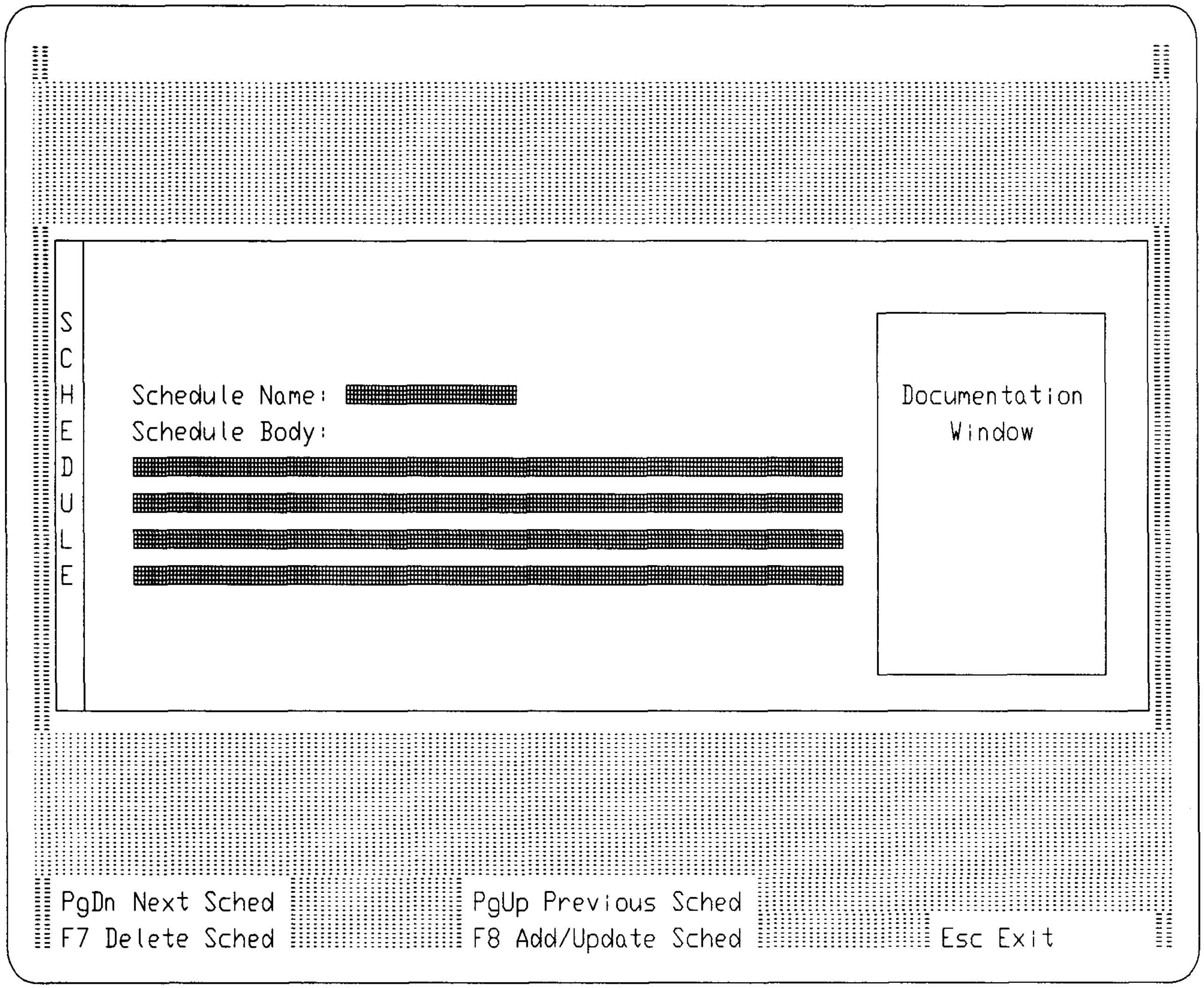
EXPTSCRN


STAGES IN USE
Training1
Training2
Conditioning
Testing1
Testing2
AVAILABLE
Testing1
Testing2
Testing3
Testing4
Testing5
Training1
Training2
Conditioning
F4 Duplicate Line
F5 Delete Line
F6 Blank Line
Enter Insert Line
F10 Save & Exit
Esc Exit

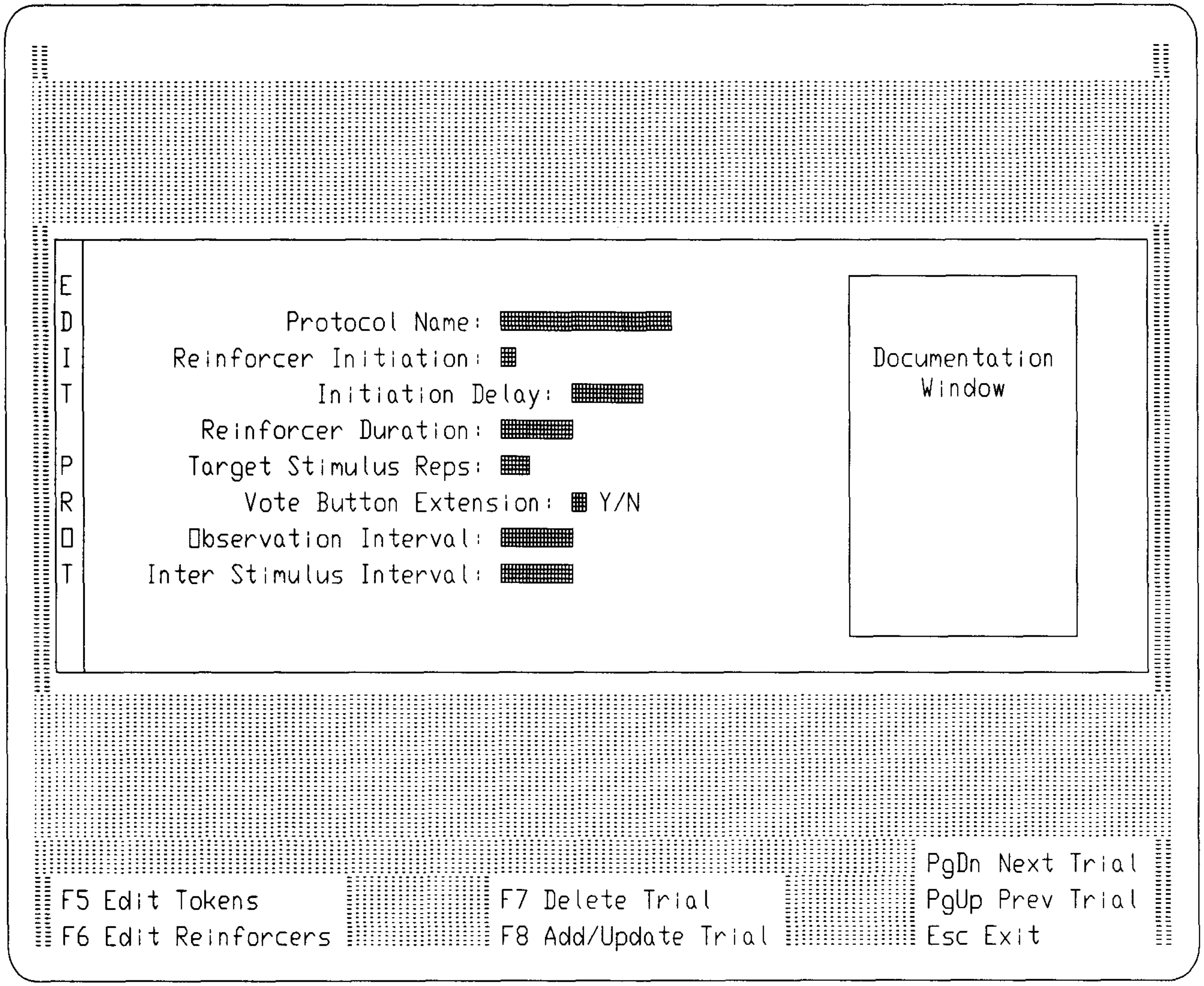
STAGESCN


EDIT STAGE
Stage Name:
Main Protocol:
Retraining Protocol:
Schedule Name:
Probability of Change Trial:
Max # Succ. Change Trials:
Max # Succ. Control Trials:
Block Size:
Minimum # of Trials:
Criterion:
Criterion Window:
Required # of Criteria:
Handling of Consecutive Misses:
# of Consecutive Misses Required:
Automatic Retraining?: Y/N
Documentation Window
PgDn Next Stage
PgUp Previous Stage
F7 Delete Stage
F8 Add/Update Stage
Esc Exit

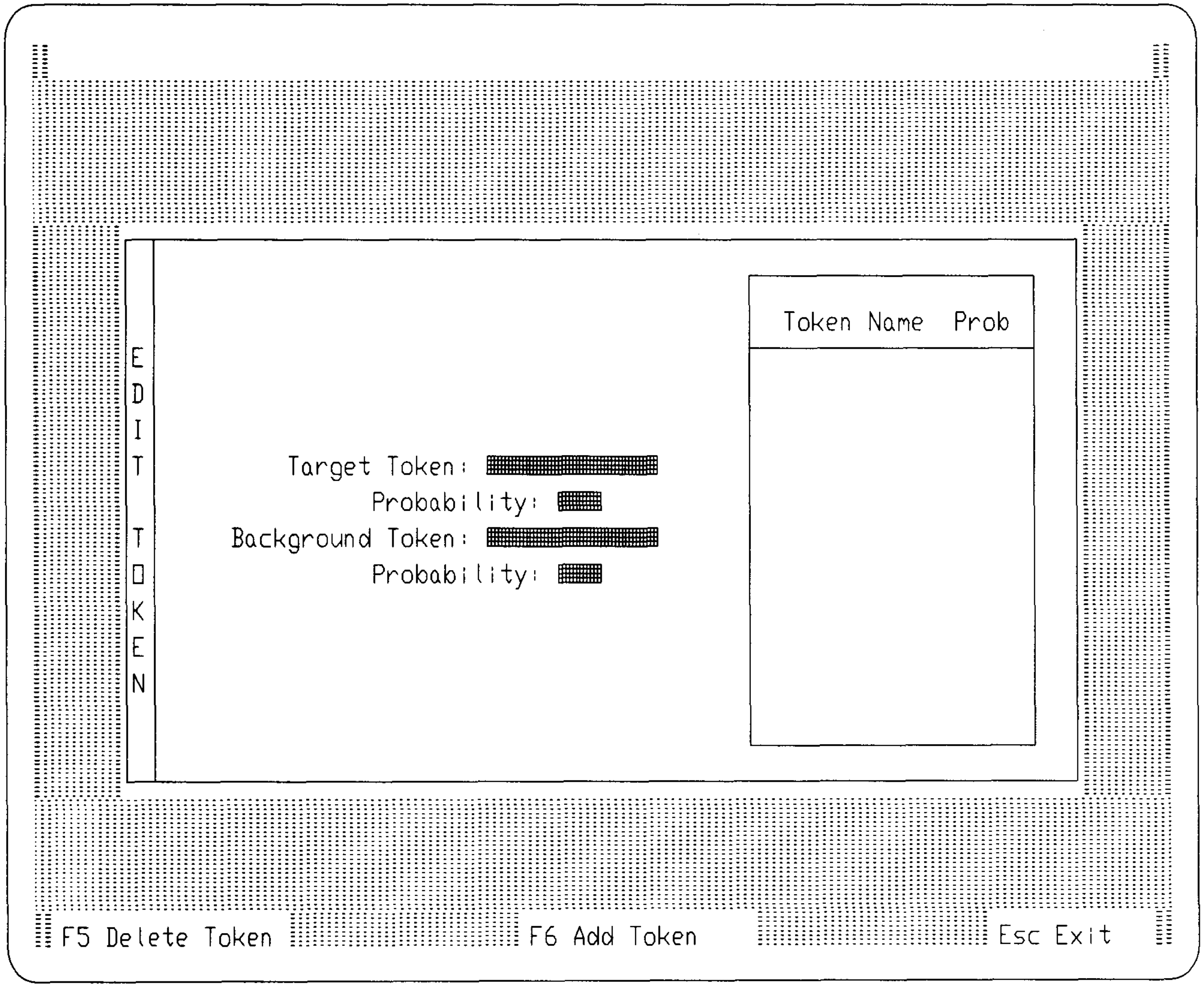
SCHDSCRN


SCHEDULE
Schedule Name:
Schedule Body:
Documentation Window
PgDn Next Sched
F7 Delete Sched
PgUp Previous Sched
F8 Add/Update Sched
Esc Exit

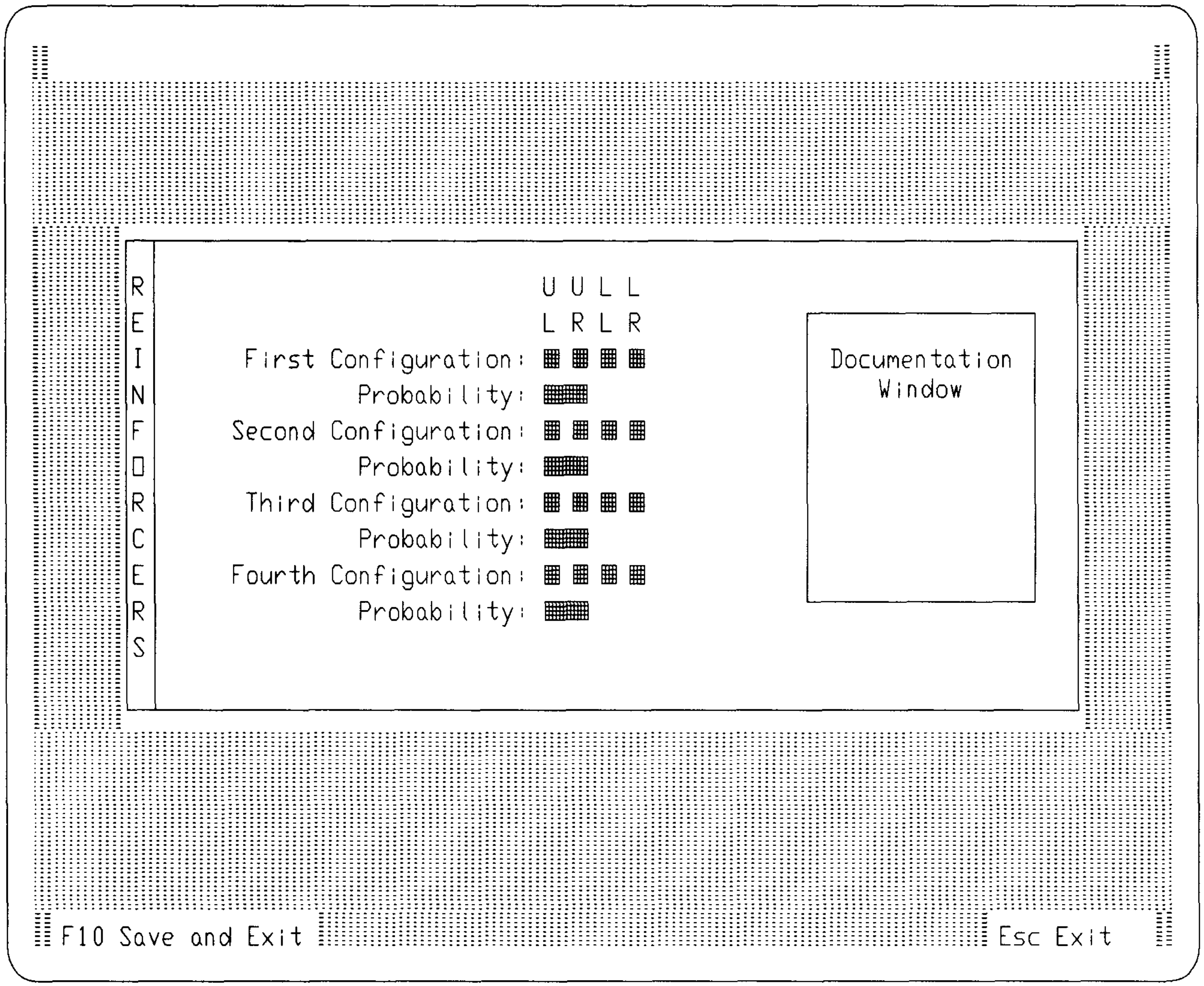
TRIALSCN


EDIT PROT
Protocol Name:
Reinforcer Initiation:
Initiation Delay:
Reinforcer Duration:
Target Stimulus Reps:
Vote Button Extension: Y/N
Observation Interval:
Inter Stimulus Interval:
Documentation Window
F5 Edit Tokens
F6 Edit Reinforcers
F7 Delete Trial
F8 Add/Update Trial
PgDn Next Trial
PgUp Prev Trial
Esc Exit

TOKNSCRN


EDIT TOKEN
Token Name Prob
Target Token:
Probability:
Background Token:
Probability:
F5 Delete Token
F6 Add Token
Esc Exit

REINSCRN


REINFORCERS
U U L L
L R L R
First Configuration:
Probability:
Second Configuration:
Probability:
Third Configuration:
Probability:
Fourth Configuration:
Probability:
Documentation Window
F10 Save and Exit
Esc Exit

Appendix G

Experimentation Literature Sample

The results of the experimentation literature sample are tabulated and presented in this appendix. The paper numbers in the tables refer to the paper numbers in the reference section of this appendix.

EXPERIMENTAL FEATURES

Paper Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
Word Completion																		■								■																								
Sentence Completion														■																																				
Selection of Alternatives					■																																													
Reaction Time				■	■					■													■																											
Response Time	■				■											■					■	■		■							■	■	■	■								■								
Visual Search																																																		
For Feature						■																									■		■																	
Comparison																																																		
Scene Analysis												■	■												■																									
Classification (Search)	■																																																	
Tachistoscope																				■			■	■	■																									
Target Tracking																																																		
Visual Audible Correlation								■																																										
User Manipulation of Screen																																																		
Adaption Based on Response																																														■	■	■	■	■
Feedback Supplied																																																		
Visual																																																	■	
Audible										■														■							■	■																		
Both								■																																										

EXPERIMENTAL FEATURES

Paper Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
Text Recognition and Comprh	■							■	■		■					■				■	■																													
Long Term Memory		■	■														■																																	
Knowledge Application	■		■																																															
Memory Scanning						■																																												
General Recognition							■																																											
Medium/Short Term Memory														■												■	■	■				■										■								
Memory Organization															■																	■																		
Measured Analog Input																																	■																	
Proofreading																			■																															
Motion Control																													■	■																				
Visual Processing																																		■	■	■														
Audio Frequency Discrim																																													■					
Spectrogram Shape Discrim.																																														■		■		
Separating 2 Audio Signals																																															■			■
Pitch Determ-Fund Freq																																					■												■	
Speech Analysis																																						■	■	■										
Threshold Measurement																																									■	■	■							
Suprathreshold																																												■						

COMPUTER FEATURES IN EXPERIMENTS

Paper Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
Visual																																																		
Text					■	■		■			■			■		■		■				■				■						■	■	■								■								
Graphics																																																		
Simple Line Drawings				■			■																	■							■	■		■		■														
Dot Patterns										■																									■															
Non Alpha Script																					■	■																												
Moving Simple Shapes																																																		
Animation																																			■	■														
Analog Pictures																																																		
Timing Critical																																																		
Audible																																																		
Beeps						■				■				■									■	■							■	■																		
Music																																																		
Speech								■																														■										■		
Analog																																														■	■		■	■
Digitized/Stored								■																														■	■						■			■	■	
Synthetic																																																		
Noise Masking																																									■	■	■	■			■	■		

COMPUTER FEATURES IN EXPERIMENTS

Paper Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
Other																																																		
Ordering of Presentation				■	■	■	■	■		■	■			■		■				■		■		■		■					■	■	■	■	■	■									■					
Recording of Results	■			■	■					■		■		■	■	■		■											■			■						■	■							■				
Analysis of Results				■	■					■				■	■	■														■							■		■											
Randomization																																	■	■													■			
Simulation or Model															■																																			
Voice Activated Switching																						■												■																
Type of Computer																																																		
Workstation								■																																										
PC Compatible				■		■						■		■		■										■			■		■	■		■			■		■			■				■				
Apple										■												■		■																										
Commodore				■				■																								■				■														
None		■	■						■				■				■		■				■		■		■	■												■										
Unknown	■						■				■				■			■		■										■								■			■		■	■	■		■	■	■	■
DEC Minicomputer					■																												■						■							■				
Terak					■																																													
Amiga																					■														■															

Fifty Papers Surveyed from the
Journal of Experimental Psychology
and the
Journal of the Acoustical Society of America

1. Allen, W. and Brooks, L. R. (1991). Specializing the Operation of an Explicit Rule. *Journal of Experimental Psychology: General, 120*, 3-19.

2. Bahrick, H. P. and Hall, L. K. (1991). Lifetime Maintenance of High School Mathematics Content. *Journal of Experimental Psychology: General, 120*, 20-33.

3. Fong, G. T. and Nisbett, R. E. (1991). Immediate and Delayed Transfer of Training Effects in Statistical Reasoning. *Journal of Experimental Psychology: General, 120*, 34-45.

4. Hopkins, W. D., Morris, R. D. and Savage-Rumbaugh, E. S. (1991). Evidence for Hemispheric Priming Using Known and Unknown Warning Stimuli in Two Language-Trained Chimpanzees *(Pan Troglodytes)*. *Journal of Experimental Psychology: General, 120*, 46-56.

5. Kyllonen, P. C., Tirre, C. T. and Christal, R. E. (1991). Knowledge and Processing Speed as Determinants of Associative Learning. *Journal of Experimental Psychology: General, 120*, 57-79.

6. Fisk, A. D. and Rogers, W. A. (1991). Toward an Understanding of Age-Related Memory and Visual Search Effects. *Journal of Experimental Psychology: General, 120*, 131-149.

7. Ashby, F. G. and Lee, W. W. (1991). Predicting Similarity and Categorization From Identification. *Journal of Experimental Psychology: General, 120*, 150-172.

8. Navon, D. (1991). Testing a Queue Hypothesis for the Processing of Global and Local Information. *Journal of Experimental Psychology: General, 120*, 173-189.

9. Durso, F. T. and Shore, W. J. (1991). Partial Knowledge of Word Meanings. *Journal of Experimental Psychology: General, 120*, 190-202.

10. Cleeremans, A. and McClelland, J. L. (1991). Learning the Structure of Event Sequences. *Journal of Experimental Psychology: General, 120*, 235-253.

11. Graesser, A. C., Lang, K. L. and Roberts, R. M. (1991). Question Answering in the Context of Stories. *Journal of Experimental Psychology: General, 120*, 254-277.

12. Brooks, L. R., Norman, G. R. and Allen, S. W. (1991). Role of Specific Similarity in a Medical Diagnostic Task. *Journal of Experimental Psychology: General, 120*, 278-287.

13. Merriman, W. E., Schuster, J. M. and Hager, L. (1991). Are Names Ever Mapped Onto Preexisting Categories?. *Journal of Experimental Psychology: General, 120*, 288-300.

14. Hertel, P. T. and Rude, S. S. (1991). Depressive Deficits in Memory: Focussing Attention Improves Subsequent Recall. *Journal of Experimental Psychology: General, 120,* 301-309.

15. Farah, M. J. and McClelland, J. L. (1991). A Computational Model of Semantic Memory Impairment: Modality Specificity and Emergent Category Specificity. *Journal of Experimental Psychology: General, 120,* 339-357.

16. Jared D. and Seidenberg, M. S. (1991). Does Word Identification Proceed From Spelling to Sound to Meaning?. *Journal of Experimental Psychology: General, 120,* 358-394.

17. Conway, M. A., Cohen, G. and Stanhope, N. (1991). On the Very Long-Term Retention of Knowledge Acquired Through Formal Education: Twelve Years of Cognitive Psychology. *Journal of Experimental Psychology: General, 120,* .

18. Tulving, E., Hayman, C. A. G. and Macdonald, C. A. (1991). Long-Lasting Perceptual Priming and Semantic Learning in Amnesia: A Case Experiment. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 595-617.

19. Danemen, M. and Stainton, M. (1991). Phonological Recoding in Silent Reading. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 618-632.

20. Perfetti, C. A. and Zhang, S. (1991). Phonological Processes in Reading Chinese Characters. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 633-642.

21. Baluch, B. and Benser, D. (1991). Visual Word Recognition: Evidence for Strategic Control of Lexical and Nonlexical Routines in Oral Reading. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 644-652.

22. Lukatela, G., Turvey, M. T. and Todorovic, D. (1991). Is Alphabet Biasing in Bialphabetical Word Perception Automatic and Prelexical? *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 653-663.

23. Vitkovich, M. and Humphreys. G. W. (1991). Perseverant Responding in Speeded Naming of Pictures: It's in the Links. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 664-680.

24. Tipper, S. P., Weaver, B., Cameron, S. Brehaut, J. C. and Bastedo, J. (1991). Inhibitory Mechanisms of Attention in Identification and Localization Tasks: Time Course and Disruption. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 681-692.

25. Christianson, S. A., Loftus, E. F., Hoffman, H. and Loftus, G. R. (1991). Eye Fixations and Memory for Emotional Events. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 693-701.

26. Nairne, J. S., Riegler, G. L. and Serra, M. (1991). Dissociative Effects of Generation on Item and Order Retention. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 702-709.

27. Brown, S. C., Connover, J. N., Flores, L. M. and Goodman, K. M. (1991). Clustering and Recall: Do High Clusterers Recall More Than Low Clusterers Because of Clustering?. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 710-721.

28. Fisher, R. P. and Chandler, C. C. (1991). Independence Between Recalling Interevent Relations and Specific Events. *Journal of Experimental Psychology: Learning, Memory and Cognition, Vol 17 No 4,* 722-733.

29. Patla A. E., Prentice, S. D., Robinson, C. and Neufeld, J. (1991). Visual Control of Locomotion: Strategies for Changing Direction and for Going Over Obstacles. *Journal of Experimental Psychology: Human Perception and Performance, Vol 17 No 3,* 603-632.

30. Schmidt, R. C., Beek, P. J., Treffner, P. J. and Turvey, M T. (1991). Dynamical Substructure of Coordinated Rhythmic Movements. *Journal of Experimental Psychology: Human Perception and Performance, Vol 17 No 3,* 635-651.

31. Treisman, A. (1991). Search, Similarity, and Integration of Features Between and Within Dimensions. *Journal of Experimental Psychology: Human Perception and Performance, Vol 17 No 3,* 652-676.

32. Venturino, M. (1991). Automatic Processing, Code Dissimilarity, and the Efficiency of Successive Memory. *Journal of Experimental Psychology: Human Perception and Performance, Vol 17 No 3,* 677-695.

33. Smid, H. G. O. M., Lamain, W., Hogeboom, M. M., Mulder, G. and Mulder, L. J. M. (1991). Psychophysiological Evidence for Continuous Information Transmission Between Visual Search and Response Processes. *Journal of Experimental Psychology: Human Perception and Performance, Vol 17 No 3,* 696-714.

34. Yee, P. L. and Hunt, E. (1991). Individual Differences in Stroop Dilution: Tests of Attention-Capture Hypothesis. *Journal of Experimental Psychology: Human Perception and Performance, Vol 17 No 3,* 715-725.

35. Shulman, G. L. (1991). Attention Modulation of Mechanisms That Analyze Rotation in Depth. *Journal of Experimental Psychology: Human Perception and Performance, Vol 17 No 3,* 726-737.

36. Delucia, P. R. (1991). Pictorial and Motion-Based Information for Depth Perception. *Journal of Experimental Psychology: Human Perception and Performance, Vol 17 No 3,* 738-748.

37. Hauser, M. D. and Fowler, C. A. (1992). Fundamental Frequency Declination Is Not Unique to Human Speech: Evidence from Nonhuman Primates. *Journal of the Acoustical Society of America, Vol 91 No 1,* 363-369.

38. Flege, J. E. and Fletcher, K. L. (1992). Talker and Listener Effects on Degree of Perceived Foreign Accent. *Journal of the Acoustical Society of America, Vol 91 No 1,* 370-389.

39. Andruski, J. E. and Nearey, T. M. (1992). On the Sufficiency if Compound Target Specification of Isolated Vowels in /bVb/ Syllables. *Journal of the Acoustical Society of America, Vol 91 No 1,* 390-411.

40. Eefting, W. (1992). The Effect of Accentuation and Word Duration on the Naturalness of Speech. *Journal of the Acoustical Society of America, Vol 91 No 1,* 411-421.

41. Smoorenburg, G. F. (1992). Speech Reception in Quiet and in Noisy Conditions by Individuals with Noise Induced Hearing Loss in Relation to Their Tone Audiogram. *Journal of the Acoustical Society of America, Vol 91 No 1,* .

42. van Rooij, J. C. G. M, and Plomp, R. (1992). Auditive and Cognitive Factors in Speech Perception by Elderly Listeners. III. Additional Data and Final Discussion. *Journal of the Acoustical Society of America, Vol 91 No 2,* 1028-1033.

43. Carlyon, R. P. and White, L. J. (1992). Effect of Signal Frequency and Masker Level on the Frequency Regions Responsible for the Overshoot Effect. *Journal of the Acoustical Society of America, Vol 91 No 2,* 1034-1041.

44. Grose, J. H. and Hall, J. W. III. (1992). Comodulation Masking Release for Speech Stimuli. *Journal of the Acoustical Society of America, Vol 91 No 2,* 1042-1050.

45. Huanping, D. and Green, D, M. (1992). Auditory Intensity Perception: Successive versus Simultaneous, Across Channel Discrimination. *Journal of the Acoustical Society of America, Vol 91 No 5,* 2845-2854.

46. Kidd, G. Jr. and Mason, C. R. (1992). A New Technique for Measuring Spectral Shape Discrimination. *Journal of the Acoustical Society of America, Vol 91 No 5,* 2855-2864.

47. Bacon, S. P. and Takahashi, G. A. (1992). Overshoot in Normal-Hearing and Hearing-Impaired Subjects. *Journal of the Acoustical Society of America, Vol 91 No 5,* 2865-2871.

48. ter Keurs, M., Festen, J. M. and Plomp, R. (1992). Effects of Spectral Envelope Smearing on Speech Reception.I *Journal of the Acoustical Society of America, Vol 91 No 5,* 2872-2880.

49. Moore, B. C. J. and Peters, R. W. (1992). Pitch Discrimination and Phase Sensitivity in Young and Elderly Subjects and its Relationship to Frequency Selectivity. *Journal of the Acoustical Society of America, Vol 91 No 5,* 2881-2893.

50. Woods, W. S. and Colburn, H. S. (1992). Test of a Model of Auditory Object Formation Using Intensity and Interaural Time Difference Discrimination. *Journal of the Acoustical Society of America, Vol 91 No 5,* 2894-2902.