DOCUMENT RESUME

ED 162 465 EC 112 508

AUTHOR	Semmel, Melvyn I.; Olson, Jerry
TITLE	System Documentation Manual.
INSTITUTION	Indiana Univ., Bloomingtcn. Center for Innovation in Teaching the Handicapped.
SPONS AGENCY	Bureau of Education fcr the Handicapped (DHEW/OE), Washington, D.C.
PUB DATE	Jun 77
GRANT	OEG-9-242178-4149-032
NOTE	173p.; A part of the Ccmputer-Assisted Teacher Training System (CATTS) Development and Applications series; For related informaticn, see EC 112 509-510 ; Photcgraphs may be marginally legitle due tc poor print quality
AVAILABLE FROM	Indiana University, Center for Inncvaticr in Teaching the Handicapped, Schccl cf Educaticn, Blocmington, Indiana 47401 ($3.00)
EDRS PRICE	MF-$0.83 HC-$8.69 Plus Pcstage.
DESCRIPTORS	Behavior Change; Classroom Otservaticn Techniques; *Ccmputer Assisted Instructicn; Compcter Criented Programs; *Educational Technolcgy; *Feedtack; *Handicapped Children; Infcrmaticn Prccessing; Inservice Programs; Inservice Teacher Education; Instructional Systems; Program Design; *Program Development; Student Eehavicr; Student Teacher Relationship; Teacher Eehavior; *Teacher Education
IDENTIFIERS	*Computer Assisted Teacher Training System

ABSTRACT

The document is a system documentaticr manual of the Computer-Assisted Teacher Training System (CATTS) developed by the Center for Innovation in Teaching the Handicapped (Indiara University). CATTS is characterized as a system capable of providing continuous, instantaneous, and/or delayed feedtack cf relevant teacher-student interaction data to a trainee in the classrccm in order to modify behavior through regulatory teaching moves. Chapter 1 discusses rationale, the importance of immediate feedback, observation systems in teacher training prcgrams, ccmputer technclcgy in teacher education, preservice research ard development activities with CATTS, and other training applications of CATTS. Described in chapter 2 are the component systems (CATTS stations, data flow, data collection, data analysis, feedback, stcrage and retrieval) and an observation system training subsystem. Chapter 3 prcvides a technical overview and detailed descriptions cf CATTS constituent systems, including design configuration, and hardware and software systems. Chapter 4 deals with the hardware components of the system (central processing component, input hardware, output hardware, and support hardware components). The software operating system compcnents are described in chapter 5, including data collecticn, data storage and transmission, device hardlers, console ccmmand control, program development, and program loaders. The final chapter offers conclusions relating to technical experience with CATTS, reliability of CATTS functions, cost-effectiveness, and CATTS technical future. The appendix provides the command list and data file formats. (ED)

THE COMPUTER-ASSISTED TEACHER TRAINING SYSTEM (CATTS) DEVELOPMENT AND APPLICATIONS

SYSTEM DOCUMENTATION MANUAL

Melvyn I. Semmel, Project Director

Jerry Olson, Systems Coordinator

Center for Innovation in Teaching the Handicapped
Indiana University, Bloomington, Indiana
June, 1977

*The development of this training module was supported by grant # OEG 9-242178-4149-032 from the U.S. Office of Education, Bureau of Education for the Handicapped, to the Center for Innovation in Teaching the Handicapped. Contractors undertaking such projects under government sponsorship are encouraged to express freely their professional judgment in the conduct of the project. Points of view or opinions stated do not, therefore, necessarily represent official Office of Education position or policy.

TABLE OF CONTENTS

PART I. CATTS FUNCTIONAL DESCRIPTION

PART II CATTS TECHNICAL MANUAL

LIST OF FIGURES

LIST OF FIGURES (Cont'd)

LIST OF TABLES

CHAPTER 1

Introduction

The Computer-Assisted Teacher Training System (CATTS) is conceptualized as a closed-loop cybernetic system capable of providing continuous, instantaneous, and/or delayed feedback of relevant teacher-pupil interaction data to a trainee in the classroom in order to modify behavior through regulatory teaching moves. These moves are designated in accordance with predetermined training objectives. The system is designed to produce a cost-effective means of collecting data from systematic observations and real-time analysis, storage, and feedback of information relevant to pupil-teacher interactions in special education contexts. Feedback can be provided through instantaneous visual display in the classroom or through hardcopy computer printout immediately following an observed teaching performance. The system permits rapid analysis and accumulation of stored data within and across teaching situations. CATTS also provides a computer-managed technique for building and adapting observation systems, and an efficient means for training reliable observer-coders. The system is designed for application in both preservice and inservice teacher-training contexts. It can be used within a teacher education laboratory on a college or university campus, or directly in public school classrooms in the community. Computer access is available through direct on-line interface, TOUCH-TONE telephone interface from remote locations, or through portable data collection devices called DATAMYTES.

Rationale

The application of computer technology offers a promising solution to a number of existing limitations in the use of systematic observation techniques in teacher education programs. Special educators are currently exploiting opportunities afforded through recent advances in technology. Application of computers in special education is broad in scope, ranging from direct computer-assisted instruction (CAI) of handicapped pupils (Stolurow, 1960), to ongoing branched CAI special education courses at the University level (Cartwright, Cartwright, & Robine, 1972), to statewide computer-managed curriculum objectives and materials retrieval systems for practitioners in the field (Noffsinger & Daiker, 1973). Undoubtedly, as such efforts progress and cost factors are controlled, the field will be faced with the reality of a technological revolution in special education within the coming decade.

In their work, Cybernetic Principles of Learning and Educational Design, Karl and Margaret Smith (1966) base their approach to human learning on the findings of early researchers in human engineering. The Smiths argue convincingly for a cybernetic interpretation of behavior--one quite different from conventional theories of learning. The cybernetic approach is a "general theory of behavior organization which . . . views the individual as a feedback system which generates its own activities in order to detect and control specific stimulus characteristics of the environment" (p. vii). CATTS is currently conceptualized as a closed-loop cybernetic system which provides immediate feedback of relevant teacher-pupil interaction variables to the teacher trainee. This allows modification of trainee behavior to be realized through regulatory teaching moves in accordance with a predetermined strategy, thus creating the desired classroom environment (Semmel, 1975). The system enables a trainer to stipulate clearly those elements or patterns of teaching behavior which he wishes to develop as goals for training. Real-time feedback of performance is provided to the trainee so that regulatory behavior may be initiated toward establishing a desired classroom learning environment for the pupils. The trainees' progress toward achieving objectives can be systematically and cumulatively tracked and evaluated by the computer's analytic and memory storage capabilities.

The Importance of Immediate Feedback

Assuming a methodological sophistication which permits the measurement and feedback of results of training concurrent with the performances of trainees in practicum settings, there arises the question of the ability of trainees to utilize knowledge of results while simultaneously being engaged in attempts to practice specific teaching skills. The work reviewed by Broadbent (1958), and Swets and Kristofferson (1970), offers some theoretical and empirical support for the contention that the human adult has the requisite information-processing competence to recode simultaneous multiple messages.

Heinrich and McKeegan (1969) reported that discrepancies between teachers' beliefs about how they were acting and how they were observed to act were less pronounced when subjects received concurrent immediate supervisory feedback as compared to delayed feedback relative to teaching behavior. The concurrent immediate feedback condition was delivered by the supervisor who raised color-coded cards whenever a desirable or undesirable teacher behavior occurred.

Reddy (1968) demonstrated that counseling trainees who received immediate supervisory feedback through a dictaphone earplug device improved significantly more in empathic skills than those who received either delayed or no feedback. Spaulding (1971) similarly reported that inservice teachers who received a variety of feedback experiences showed greatest improvements under a condition employing immediate feedback during classroom instruction periods. The immediate feedback from the observer was transmitted by means of a wireless audio receiver and ear speaker. It should be emphasized that in these studies the feedback provided to trainees was instantaneous and occurred during rather than after the training sessions. The work cited, therefore, offers empirical evidence for the contention that teachers in training can process and utilize feedback information while attempting to acquire specific teaching skills.

The importance of immediate knowledge of results or feedback in the learning process has been well documented. Greenspoon and Foreman (1956) have reported that delayed feedback, compared to immediate feedback, has a negative effect on human learning of a simple motor task. Tasks involving verbal skills appear also to be facilitated through the immediacy with which feedback can be provided (Bourne, 1957). Some workers have gone so far as to contend that feedback might well be the "strongest" and "most important" variable involved in learning and performance (Bilodeau & Bilodeau, 1961).

Hence, the human learner may be viewed as a self-regulating cybernetic system who relies on feedback in his efforts to maintain goal-directed behavior (Smith & Smith, 1966; Semmel, 1968). The faster the learner can receive feedback, the faster he can be expected to modify his behavior in the direction of discriminable objectives--and thus increase his efficiency in the acquisition of teaching skills (Gibbs, 1954).

Observation Systems in Teacher-Training Programs

It is evident that relevant teaching skills must be developed within the context of a comprehensive philosophical or empirical framework which is hypothesized to positively affect pupil learning. The most relevant objectives in training will probably be those which go beyond the simplistic notions of trainee discrimination and generation of a specified frequency of X or Y behaviors. It is more likely that a training program will need to be concerned with complex interactive patterns of classroom behaviors and the concatenation of these patterns into operational definitions of desired pedagogical environments.

A number of observation-coding systems have been developed by educators (Simon & Boyer, 1970; Medley & Mitzel, 1963). The categories used in these systems constitute operational definitions of what the designers deem to be important classroom processes. When teacher-trainees are encouraged to favor one subset of behaviors or patterns from the total set of categories defining the system, it may be said that a program has established specific behavioral objectives for the trainee. When trainee performance is observed systematically and the codified behaviors are fed back to trainees, the system may be thought of as being a functional teacher-training tool (Amidon, 1970; Bondi, 1970; Flanders, 1970).

Observation-coding systems have an intrinsic appeal to teacher educators. They (a) establish a set of operationally defined behavioral objectives for the trainee; (b) generally suggest an implicit set of training procedures leading to direct practicum experiences for trainees; and (c) generally provide a set of ground rules which permit reliable measurement of trainee progress. Existing systems vary greatly in their specificity of teaching behaviors. Some focus on the affective climate of the classroom (Flanders, 1970), while others focus on the cognitive demands made by the teachers (Lynch & Ames, 1971), teacher control behaviors (Fink & Semmel, 1971), teaching strategies (Bellack, Kliebard, Hyman, & Smith, 1966), nonverbal behaviors (Galloway, 1968), and a host of other interactive skills.

While ideally suited to the requirements of a skill-oriented training program, observation systems are subject to limitations as operational tools for teacher-training programs. They require extensive time commitments on the part of trainers, who, after assisting trainees in discriminating operational objectives, must observe, code, summarize, analyze, and subsequently feed back the results of performance to trainees. Hence, the total training process becomes tedious, and the excessive time commitments seriously limit the feasibility of such an approach.

Secondly, analytic methods available to t[illegible]ner generally prohibit feedback of relevant patterns of interaction [illegible] ne frequencies of simple two-stage transitions. Method of data reduct[illegible]uently lead to distortions of the frequencies of behaviors for [illegible] periods (Collet & Semmel, 1970). Of greatest importance, howeve[illegible] the fact that current methods necessitate relatively long delays of feedback to trainees. Hence, as implied by the literature previously reviewed, it is questionable that the information provided to trainees could have maximum effects on the modification of subsequent teaching performance.

Exploiting Computer Technology in Teacher Education

Earlier sections of this report emphasize the importance of immediate feedback to the acquisition of relevant teaching skills. Observation systems were discussed as potential operational tools for the specifications of training objectives, as feedback instruments in training, and as tools for the measurement of trainee performance. The utility of such observation-coding feedback systems is severely limited by the tedium imposed by data reduction procedures and the resulting delay of feedback to trainees. It would appear that there is a need to explore a skill-oriented teacher-training system which meets the following criteria:

(a [illegible] rmits the adoption and [illegible] neration of a broad spectrum of [illegible] servable teacher and/or pu[illegible] haviors--to be definable within the context of any system of N mutually exclusive cate-tories of behaviors.

(b) It permits the continuous and instantaneous observation, coding, analysis, and feedback of relevant training information to the trainee while he is teaching--with feedback delivered through some me[illegible] ful auditory or visual sourc[illegible] the teaching enviro[illegible]

(c) It permits the utilization of automatic analytic techniques for the continuous, rapid synthesis and description of relevant behaviors, patterns, and environments--[illegible] le maintaining both the frequency and duration of behavior [illegible] well as their sequential relationships.

(d) It permits the rapid cumulative storage and retrieval of all training sessions for any one trainee or group of trainees who uses the system.

One of the most promising means for meeting t[illegible] above criteria is through the exploitation of "real-time" computer technolo[illegible].

Preservice Research and Development Act[illegible] ies with CATTS

Initial work on CATTS was reported by Semmel (1968) and his students; Kreider (1969), Weaver (1969), Schmitt (1969), and VanEvery (1971). In general, these studies support the efficacy of immediate concurrent CATTS feedback when specific behavioral goals are central to training.

Schmitt (1969), for example, used CATTS and a modified version of the FIA system to train preservice teachers to increase their use of broad questions and to reduce the frequency of binary questions in teaching the educable mentally retarded (EMR). The results indicated that CATTS trainees spent significantly more time asking broad questions than did control group trainees. The study also indicated a positive relationship between teachers' use of broad questions and the production of multiple-word responses by EMR pupils.

VanEvery (1971) used CATTS technology to study the training of speech therapists in a clinical setting. A remote telephone line was used to communicate between the speech clinic and the CATTS computer facility. Observations of therapists in training were coded in the clinic and transmitted by telephone line to the computer, which fed back information in real-time. The feedback was presented on an event recorder which traced a pattern representing training objectives on a moving belt of paper. Trainees who received the immediate CATTS feedback showed a significant increase in the use of social reinforcement patterns when compared to a control group. VanEvery's work demonstrated the feasibility of eventually moving CATTS into remote public school classrooms for inservice training opportunities.

Other Training Applications of CATTS

CATTS is not limited to the delivery of instantaneous or delayed feedback to trainees, nor to rapid data collection, retrieval, and analysis. It can also be used to develop discrimination skills of trainers through the training of reliable observers. A newly developed, computer-aided training device called DITRMA (DIscriminate, TRain, and MAintain) aids in the development of these observation skills. DITRMA is based upon a simple consensus-coding principle whereby individual trainees' responses from two or more coding termi[illegible] are [illegible]multaneously compared by the comp[illegible] the resul[illegible]son is instantaneously fed ba[illegible] the [illegible]ees. Through exp[illegible] on of this simple configurati[illegible], the DIT[illegible] system can be used [illegible] mination of relevant teacher-pupil [illegible], and to maintai[illegible] lev[illegible] reliability. DITRMA is a second-gen[illegible] consensus-coding syst[illegible] gin[illegible] developed by Semmel, Guess and Fla[illegible] University o[illegible] chig[illegible] Ann Arbor.

Duri[illegible]g coder-tr[illegible] sessions, observers code videota[illegible] f the obse[illegible] on sy[illegible] egories on data collection boxes [illegible] gur-ation ident[illegible]cal t[illegible] ONE telephones. These button bo[illegible] to the PDP-1[illegible] computer [illegible] Center for Innovation in Teachin[illegible] pped (CITH). The compute[illegible]ts as an impartial judge, as traine[illegible] a small group of up to [illegible] peers. If all coders agree with eac[illegible] the coding of an event, the group receives an auditory reinforcer [illegible]ud speaker, the videotap[illegible]ontinues, and the computer [illegible] in its memo[illegible]. However, should one or more code[illegible] agree [illegible]e others, the computer automati[illegible]ly stops the videotape, and display of [illegible] identified codes appears on the second video monitor for all trainees to study. Trainees subsequently discuss their differences, and the computer "refuses" to continue the first videotape monitor until the group reaches a consensus of agreement on what is the correct discrimination and code.

Evaluation of the system has shown that coder-training time can be reduced by approximately 50% when compared to previous procedures using paper and pencil techniques. Of perhaps greater importance is the serendipitous realization that DITRMA is a potentially powerful device for training many teaching skills through what, in effect, is an automatic, self-instructional group format. The system acts as an impartial and automated discussion leader for small groups of trainees who are viewing videotaped protocols.

The new DITRMA system is also capable of assisting in the development of category observation systems. DITRMA is an invaluable tool for rapid identifying categories which are subject to relatively high rates of observer disagreement, an indication that the categories are imprecisely de

[illegible]TER 2

Description [illegible] Compon[illegible]

CATTS stations

The CATTS configuration pres[illegible] [illegible] sta[illegible]ons: Teaching Station, Obs[illegible] [illegible]ing Station. Figure 1 ill[illegible] [illegible]ma[illegible] [illegible]ram of the present CATTS in[illegible] [illegible] va[illegible] in Teaching the Hand[illegible]apped [illegible]

Teaching Station. The Teaching Static[illegible] [illegible] f a [illegible] [illegible]srooms which can accommodate a feedback devi[illegible] [illegible] feedback source [illegible] [illegible]ted so that the teacher can use th[illegible] inform[illegible] [illegible]smitted [illegible] [illegible]e [illegible]erver with no interference with on[illegible]oing class[illegible] [illegible] feed-back display may be either visual or audi[illegible] [illegible] [illegible]ed either directly by the computer or indir[illegible] [illegible]roug[illegible] [illegible] display hardware.

Current applications permit the de[illegible]ery [illegible] variety of visual displays to the Teaching Station. Visual feedback can be provided through closed-circuit, televised images of a C[illegible]hode-ray Tube (CRT) display under direct computer control, or an external device which displays feedback information by changing light patterns, or x-y chart recordings (Semmel, 1968; Semmel, et al., 1971; VanEvery, 1970).

The system can be adapted to provide instantaneous auditory feedback. In this application, the PDP-12 computer is linked to an audio tape recorder which is capable of moving a tape of prerecorded messages containing verbal feedback relevant to specific teaching behaviors and patterns rapidly across a computer-monitored tape head. For examp[illegible], if teacher talk (as computed from a number of behavioral categories) exceeds a predetermined criterion (e.g., 80% within a particular time period), the comp[illegible] automatically

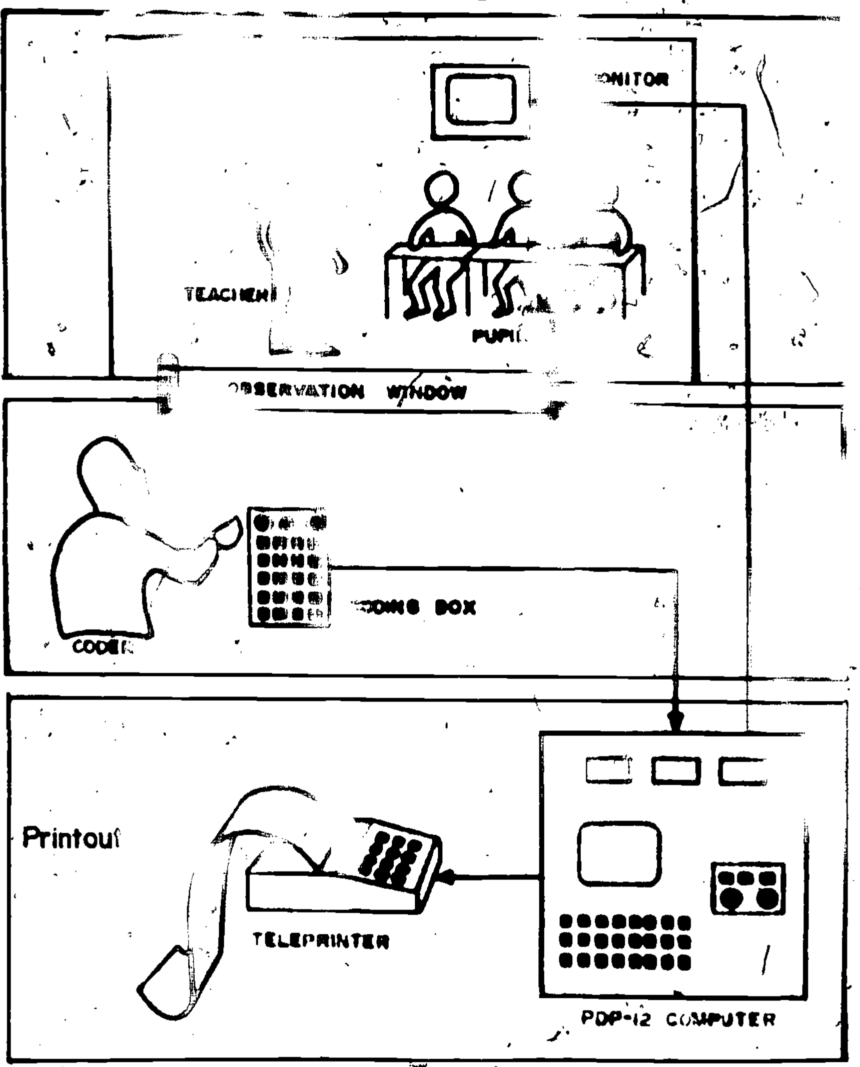


Figure 1. CATTS configuration.

"directs" the tape transport to locate the tape message which says, "Try cutting down teacher talk, and get more pupil participation." While teaching, the trainee receives the message via a wireless, transistorized, audio receiver with earplug speaker. The specification of the parameters for performance criteria through computer programming assures error-free monitoring of trainee behaviors. The resulting CATTS feedback should, therefore, be more reliable than supervisor feedback methods.

Observation-Coding Station. The Observation-Coding Station provides the link between the events occurring in the classroom and the computer analysis of these events. Here, a trained observer codes classroom events consisting of N categories of teacher and/or pupil behavior. Observation may take place within the classroom itself, within an observation booth adjoining the classroom, or by a closed-circuit television connection. Figure 2 illustrates observation of classroom behavior through a one-way glass window. At present, the coding terminal used to input observation data consists of 10 mechanical pushbuttons mounted on a metal box--a configuration similar to a TOUCH-TONE (TT) telephone. Figure 3 shows an observer with a button box. These buttons, in turn, interface directly with the computer.

It should be noted that the coding terminal permits an observer-coder to input data which is transmitted over telephone lines into a TT Data Set, and from there directly into the computer. The TT telephone interface allows direct real-time observation for remote observation coding in community classrooms.

Probably the greatest asset of the coding terminal is that it can accommodate any observation system whose classification scheme can be defined by a series of numbers. The number of categories that can be accommodated is virtually unlimited.

Figure 2. Observation-Coding Station at CITH.

Figure 3. Observer with button box.

The present computer program is capable of accepting a three-stage classification scheme with a maximum of 16 different codes per stage. Hence, the system currently can record, in addition to categories and subcategories, the interactions between teachers and specific children in a classroom.

Analysis-Encoding Station. The Analysis-Encoding Station consists of a small computer (PDP-12) and the associated computing hardware required for the on-line processing of coded input data which is gathered and transmitted from the Observation-Coding Station. Figure 4 shows the PDP-12 computer at CITH. Presently, the PDP-12 computer can simultaneously process input data from as many as 12 classrooms, and up to 12 types of observation systems.

In addition to processing input, the computer system also controls the display devices used in the teaching stations and provides for hardcopy printout, storage, and transfer of the analyzed data. Figure 5 shows the video camera method of visual feedback transmission back to the classroom. It is at this station that the trainer initiates the program options available in CATTS. The teleprinter console, through software program control, allows the operator to select any specific CATTS program or option that will satisfy the objectives of the trainer and/or trainee.

The selection of the mode and content of feedback to the teacher-trainee in the teaching station is also initiated from the console. If a CRT display is chosen as the method for feedback, the operator determines the content of the display by assigning the input data to different computational functions for the computer to calculate and display as feedback. The nature of the display is also selected from the console which allows

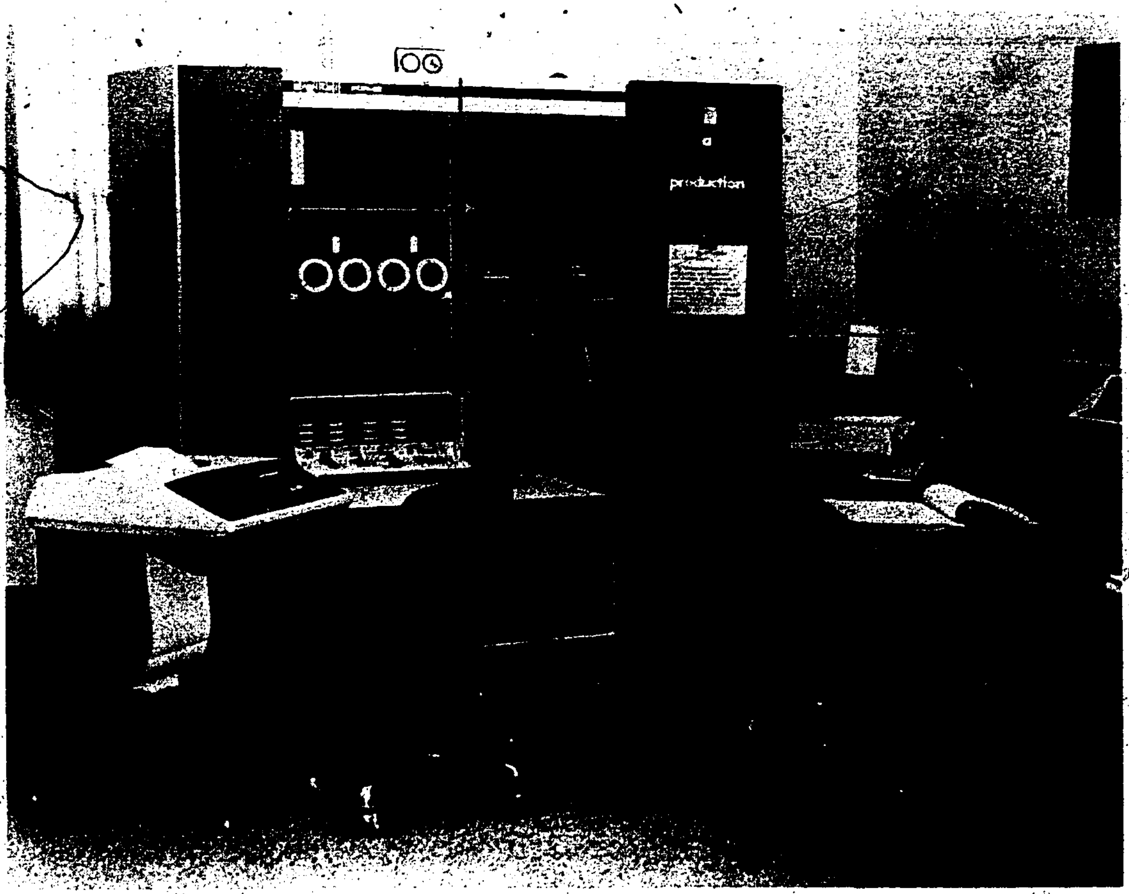


Figure 4. POP-12 computer at CITH.

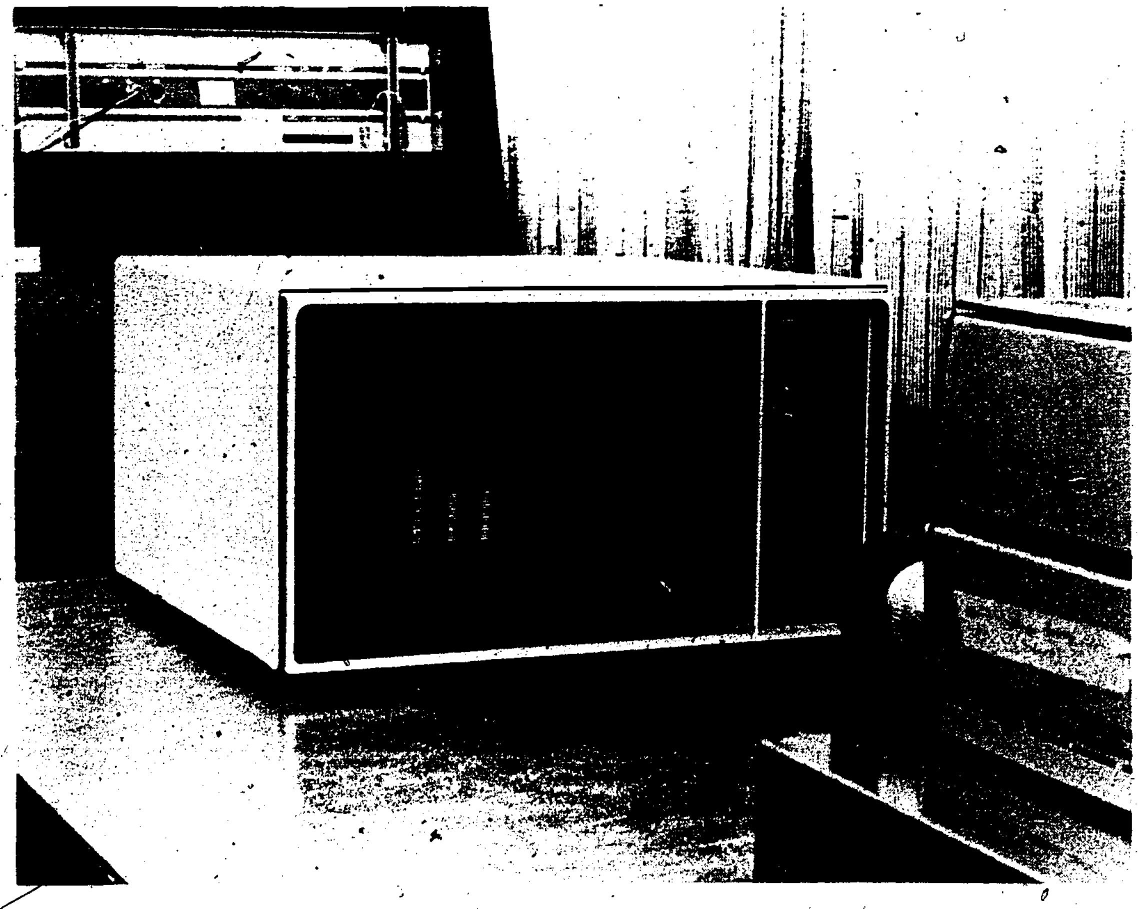


Figure 5. CRT and video camera visual feedback transmission configuration.

continuous feedback information to be presented either in alphanumeric or graphic form. Figure 6 is an example of a graphic feedback display. The completion of the input-feedback cycle is realized in a matter of a few microseconds. Feedback displays can vary from simple frequency counts of desired behaviors, to abstract representations of sequential interaction patterns.

The integrated teleprinter, which also serves as the communications link to the CATTS program, provides hardcopy printouts for inspection immediately following the training session. Figure 7 illustrates an example of a printed feedback summary. These printouts can provide such information as the event times of the coded tallies, percentage of time spent in each behavior category, and frequency of occurrence of each category; or they can provide more complete descriptions and statistical treatments of the data. This information can be used as "delayed feedback" to a trainee, and can be stored for later analysis. It should be noted that the use of the term delayed feedback in the present context is generally defined as immediate feedback by most teacher-trainers (Baker, 1970), since computer printouts are available to trainees immediately following their CATTS training session.

By using the computer in this manner, bits of observation system data are not limited only to storage by frequency and time of occurrence. The duration of behaviors and their order of occurrence are also recorded in the computer's memory, and subsequently on magnetic tape for analysis. CATTS can collect a continuous data record of classroom events, allowing analysis of duration of behaviors, as well as analysis of patterns or chains consisting of up to eight behavior categories. Hence, rapid description and analysis of a continuous sequential vector of observational data can be analyzed and fed back to trainees (Collet & Semmel, 1971).

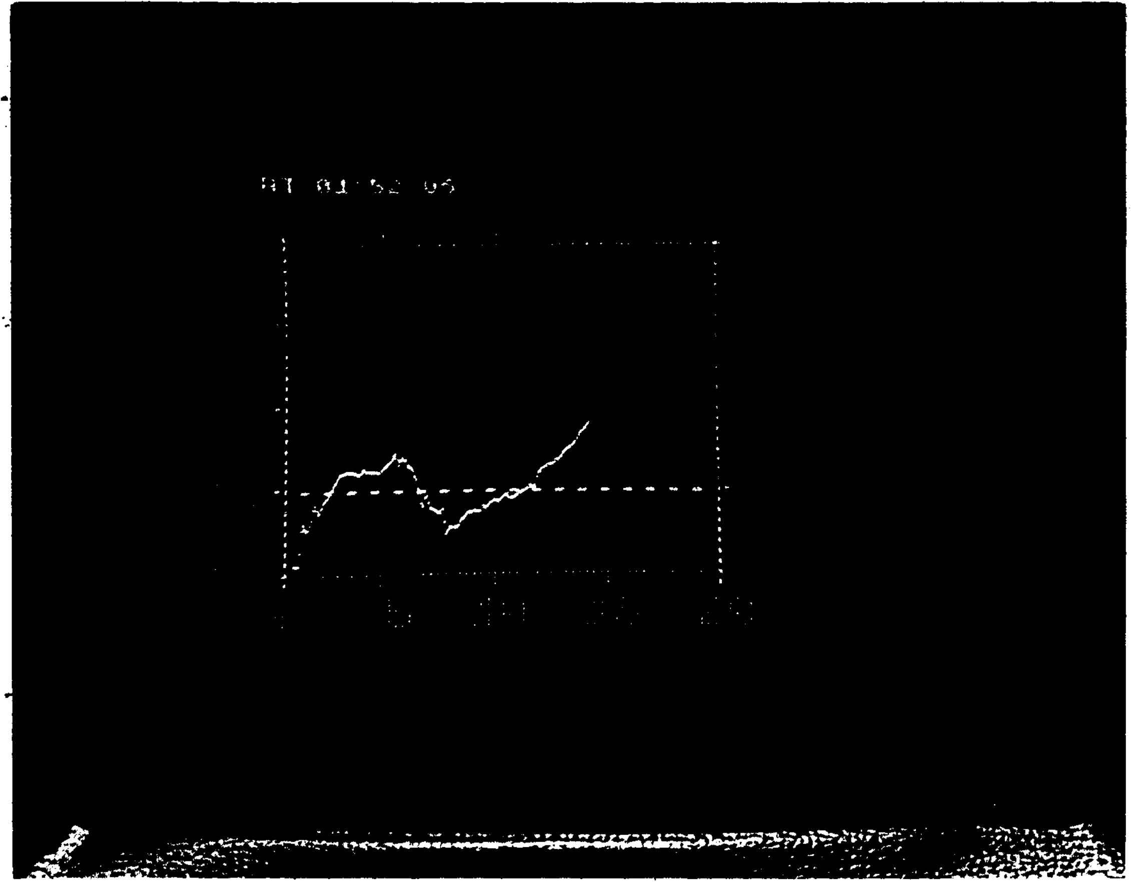


Figure 6. Graphic feedback display.

Figure 7. OROS DATA SUMMARY

```
DATE: 2/16/76

NAME: THERESA REIDY

BOX NO. : 5

LESSON NO. : 8

1) TOTAL OBSERVATION TIME: 14.7 MIN.

2) TOTAL WORDS READ: 250

                                           MISCUES      TEACHER RESPONSES
                                          NO.   PC.           NO.   PC.

3) PUPIL MISCUES TOTAL (21 + 22)          34    13        4)  12    35

   5) MEANING CHANGE (21)                 23    67        6)  12    52
   7) NO MEANING CHANGE (22)               2     5        8)   0     0
   9) SELF-CORRECTED (63)                  9    26

                                        PROMPTS GIVEN   PUPIL SUC. (62+64)
                                          NO.   PC.           NO.   PC.

10) TEACHER PROMPTS TOTAL (31 TO 8)       36   100       11)  22    61

  12) CONTEXT (52)                        11    30       13)   7    63
  14) STRUCTURAL (33)                      0     0       15)   0     0
  16) PATTERN (44)                         3     8       17)   2    66
  18) PHONIC (45)                          2     5       19)   1    50
  20) ATTENTION (34)                       0     0       21)   0     0

  22) OTHER 3, 4, 5 PROMPTS               12    33
  23) POSITIVE FEEDBACK (71)               1     2
  24) NEGATIVE FEEDBACK (72)               0     0
  25) MANAGEMENT (73)                      2     5
  26) TELLING (8)                          5    13
  27) OTHER (9)                            3     8


        SEQUENCE LENGTHS

FREQUENCY DISTRIBUTION (MINUS 15 AND 95)

  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 2
---------------------------------------------------------------------------
 14  5  6  1  3  1  2  1  0  1  0  0  0  2  0  0  0  0  0  0  0  0  0  0
---------------------------------------------------------------------------

          MEAN=   3.53
          MEDIAN=  2.30
          SD=   3.47

          TOTAL DATA STRINGS=  36
 900 730
 213
 213 630 100
 213 310 640 100
 212 100
 212 630 100 520 640 730 100
 213 630 100
 213 100
 211 730 640 510 620 310 640 100
 222 630
 212 510 610 510 610 510 640 710 100
 213 520 100 520 610 510 610 510 610 450 620 610 800 640 710 100
 212 800 640 100
 212 520 620 100
 212 310 620 730 610 440 610 440 640 610 450 610 310 640 100
 212 100
 222
 212 800 640 520 640
 215 100
 213 100
 213 630 100
 212 630 100 520 640 100
 211 620
 210 100 633 520 610 640 710
 212 630 100 630 520 640 710 900 730 100
 213 800 640
 210 520 620
 212 100
 213 630 520 640 710
 213 520 610 310 610 510 610 440 640 710 100
 224 100
 213 100
 213
 213 100
 210 100
10 800 640 900
  E
  DATA FILEEND OF DATA FILE
 PROCESSED=SIONS PROCESSED=   1
```

Figure 7. (cont.)

In summary, it can be seen that the translation of the closed-loop cybernetic principle is achieved through the prototype CATTS by using a human observer-coder as the interface between the events in a practicum setting and a computer. Behavior in the Teaching Station is observed in the Observation-Coding Station and transmitted to the Analysis-Encoding Station. In a matter of microseconds, the computer summarizes, analyzes, stores, and continuously feeds back relevant information directly to the trainee in the classroom. The system also provides a comprehensive printout of the analysis of all variables coded in the observation of the classroom transactions. The printout may be used as summative feedback following a training session or stored for subsequent recall or analysis.

System data flow description

Prior to any specific discussion of the various functional components which comprise CATTS, a discussion of the overall system data flow will assist in placing the individual system components in proper perspective relative to the total system configuration.

Figure 8 contains a block diagram of the initial data collection and feedback-processing data flow as processed on the PDP-12 mini-computer located in the Teacher Education Laboratory at CITH. Beginning at the top of the diagram, data is initially collected from three sources; (1) direct button box entry, (2) TOUCH-TONE telephone entry, and (3) off-line portable data recorder entry. The portable data recorder information must go through a decoding process before being accepted into the first-stage input processing. The first process is to check the data for syntax errors which relate specifically to the observation-coding system being used. This stage cleans the data and eliminates gross observer entry errors. From the syntax process the data flows in two directions. If instantaneous feedback is to be employed, the data is presented to various visual feedback display routines for transmission into the teaching environment. The second path, which is not optional, passes the data through a data index and storage process which places an identification of the collection session on the data file and stores the file on the PDP-12 primary mastery file system for subsequent retrieval.

From the primary master file the data is extracted by a transmission process and sent to the main computing center at Indiana University for final indexing and storage. Optional at this point is the availability of various data report routines for immediate inspection of data stored locally at the PDP-12.

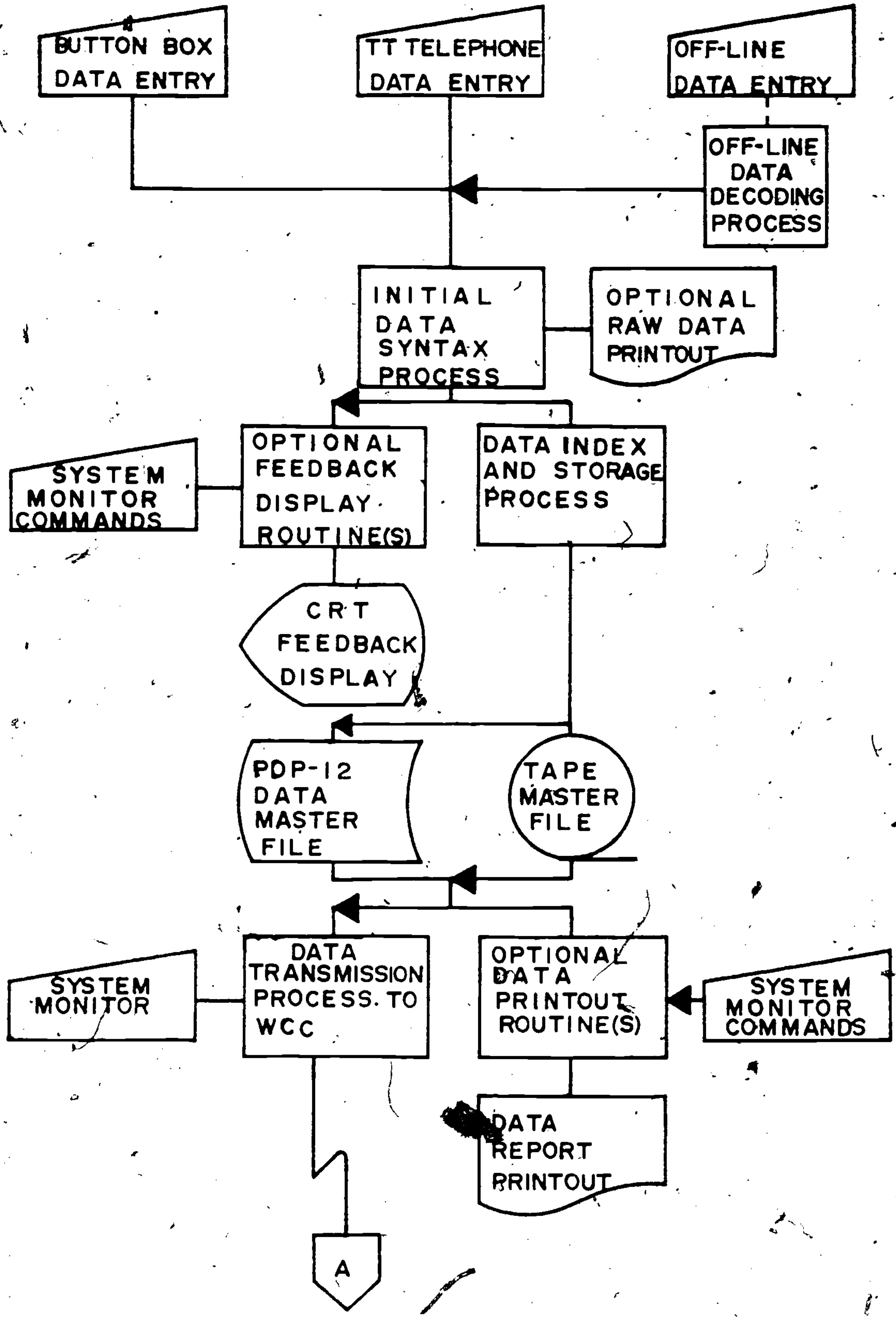

BUTTON BOX DATA ENTRY
TT TELEPHONE DATA ENTRY
OFF-LINE DATA ENTRY
OFF-LINE DATA DECODING PROCESS
INITIAL DATA SYNTAX PROCESS
OPTIONAL RAW DATA PRINTOUT
SYSTEM MONITOR COMMANDS
OPTIONAL FEEDBACK DISPLAY ROUTINE(S)
DATA INDEX AND STORAGE PROCESS
CRT FEEDBACK DISPLAY
PDP-12 DATA MASTER FILE
TAPE MASTER FILE
SYSTEM MONITOR
DATA TRANSMISSION PROCESS TO WCC
OPTIONAL DATA PRINTOUT ROUTINE(S)
SYSTEM MONITOR COMMANDS
DATA REPORT PRINTOUT
A

Figure 8. CATTS data flow: Real-time PDP-12 process.

Figure 9 illustrates the data flow process at the main time-sharing computing center (CDC-6600). When the data arrives at the CDC-6600, it proceeds through the final index and storage process which provides each file with a permanent identification and location for the next process of data retrieval. At this stage a user initiates the retrieval process by requesting specific data files to be removed from the master file system and placed on a temporary scratch file for either examination or further processing.

From here the CATTS system is conceptualized as taking two different directions; individual session feedback summary reports, and multiple session analysis procedures. Individual summary reports operate directly off of the raw data, and the various summary feedback routines are selected by the user for generation of printed feedback reports on time-sharing terminals. Multiple data sessions are passed through various data reduction routines which prepare ordered data file structures acceptable to data analysis packages such as the SPSS and BMD series.

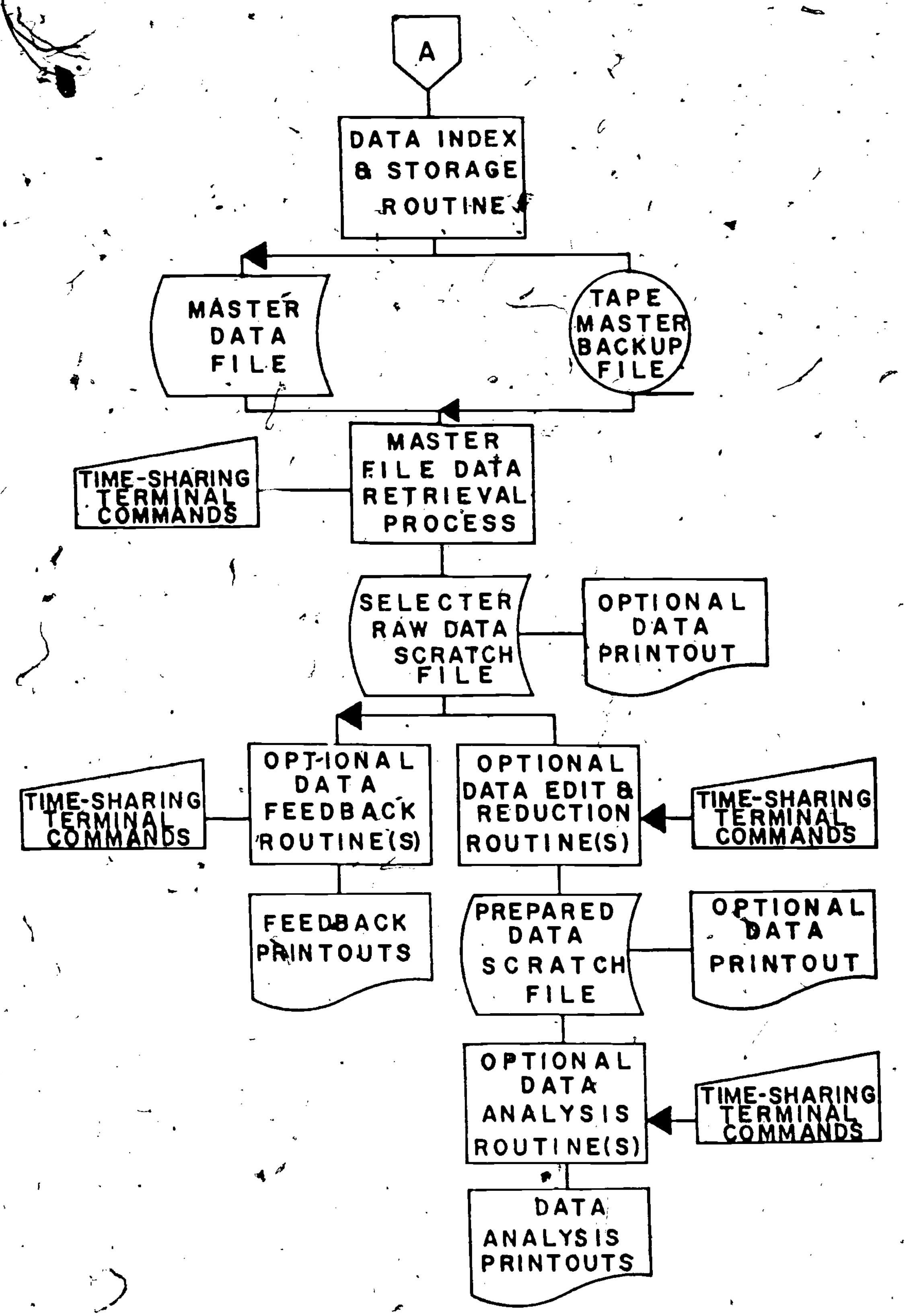

A
DATA INDEX & STORAGE ROUTINE
MASTER DATA FILE
TAPE MASTER BACKUP FILE
TIME-SHARING TERMINAL COMMANDS
MASTER FILE DATA RETRIEVAL PROCESS
SELECTER RAW DATA SCRATCH FILE
OPTIONAL DATA PRINTOUT
TIME-SHARING TERMINAL COMMANDS
OPTIONAL DATA FEEDBACK ROUTINE(S)
OPTIONAL DATA EDIT & REDUCTION ROUTINE(S)
TIME-SHARING TERMINAL COMMANDS
FEEDBACK PRINTOUTS
PREPARED DATA SCRATCH FILE
OPTIONAL DATA PRINTOUT
OPTIONAL DATA ANALYSIS ROUTINE(S)
TIME-SHARING TERMINAL COMMANDS
DATA ANALYSIS PRINTOUTS

Figure 9. CATTS data flow: Remote time-sharing process.

Data collection

Data from observers is input into CATTS from one or more of three alternatives; (1) directly connected button boxes, (2) directly connected TOUCH-TONE telephones and associated tone decoders, and (3) portable data recorder units (DATAMYTES).

Local access to the PDP-12 computer is provided by specifically constructed hand-held button boxes as shown in Figure 10. These units, 12 in all, connect directly through parallel circuitry into the parallel input interface of the computer. The button configuration is similar to that found on a 12-digit TOUCH-TONE telephone. The boxes also contain three indicator lights which correspond to the three levels of nested coding capabilities provided by the system to accommodate most observation system requirements. A toggle switch is also available for the purpose of turning the box off and on.

The data entry procedure is accomplished by first having the observer activate the box through manipulation of the toggle switch. The computer, sensing the box as being "on," turns on the three indicator lights on top of the box. The observer then enters a series of identification codes which become part of the data base. When the observation session is to begin, the observer enters a control code as data. Upon receipt of the control code, the computer begins the timer for that particular box. At the end of the observation session, the observer enters another control code to terminate the data collection sessions, at which time the computer closes the data file and records the data on the PDP-12 mass storage index.

Figure 10. CATTS hand-held button box.

During data collection the indicator lights provide feedback to the observer by indicating acceptance of each data entry and its current position in the coding structure. For example, if the coding system used by the observer has a two-level structure whereby each first-level code has an associated second level of subcodes, then the first light will go off when the first-level code has been entered. This informs the observer that the system is expecting the subcode entry.

The button boxes also contain a "SKIP" key which allows the observer the option of clearing a mistake and re-entering the correct code before striking the "ENTER" key. Once the ENTER key has been pushed, the data associated with that ENTER key cannot be changed.

The toggle switch also allows an observer to pause in the middle of an observation session without ending the session. By turning the switch to "off," the computer time associated with that box is held in a pause state until the switch is turned back on. When this occurs, the middle light stays on to remind the observer that his box is in a pause state. This state is only available during the data collection session before the control code is entered to terminate the session.

Besides the control codes to begin and end a data collection session, other control codes are available to indicate changes in the nature of the data being entered in any one observation session. This allows an observer to use a number of different observation systems in the course of one observation. These control codes become part of the data and are used in the data summary process to indicate that the data following the control codes must be interpreted differently.

The TOUCH-TONE telephone input procedure is very similar to that of the directly connected button boxes, in that data is entered exactly the same way by the observer. TOUCH-TONE telephone access allows data to be collected from remote locations away from the PDP-12 location. The procedure in projects using CATTS has been to install a number of telephone jacks in classrooms connected to a single telephone line from a school. Connected to the PDP-12 is a Data Set that decodes the tone signals and converts them back into parallel numerical information. The observer reports to the school office, picks up the TOUCH-TONE telephone extension and selects the classroom to be observed. The telephone is then plugged into the classroom jack and the telephone number of the Data Set is called. The computer senses the incoming call, makes the connection, and signals this connection to the observer by sending a short confirmation tone back to the hand set. The observer then enters the identification code and the control code to start the session. Whenever a control code is entered, an answer-back tone is sent to the observer in order to confirm the receipt of that code. The box on and off status is keyed to the answering and termination of the telephone call. There is no pause function available to the telephone procedure other than the different method of collecting the data; the data to the computer looks exactly like the data collected on the directly connected button boxes and is stored in the same format.

The portable data recorder collection procedure is also very similar to the other on-line methods. The difference is that the data is stored on cassette tapes for entry into the system at a later time. The recorders have the same button configuration and observers enter data the same way during the observation process. Because of its portability there is no feedback available to the observers with regard to control code entry. To insure against the recorder missing a control code, the procedure followed by the observers is to enter each control code twice. Upon receipt of the cassette tapes from field-based observers, the tapes are replayed through a decoder and the numerical data is input into the system. Again, the data is stored in the same format as with the other input methods.

The portable data recorder contains an interval timer and an incremental cassette tape recorder transport. These features allow an associated interval time value to be appended to each individual data entry, which duplicates the function of the computer clock in regular on-line data collection. The incremental recorder transport only moves to record data when it is entered. For example, where only 15 minutes of tape may be used for every hour of observation, this feature saves the amount of tape used in any observation period. The advantage of this is in the reduction of the amount of time taken in the decoding process at the PDP-12 computer. Newer models of the portable data recorder (DATAMYTE) have eliminated the cassette tape unit by storing the data in a solid-state internal memory chip. Transmission of data into a CATTS data structure is accelerated with this improved data collection device.

Real-Time Data Manipulation and Analysis

One of the important general functions of the CATTS system is to provide real-time data manipulation and analysis to projects using the system. This real-time processing application is centered around the instantaneous feedback capability of the system. Due to the time-sharing nature of the CATTS operating system on the PDP-12, incoming data can be routed to instantaneous feedback routines which generate real-time Cathode-Ray Tube (CRT) displays for visual feedback, and tape recorder transport control for auditory feedback. The present CATTS system has the capabilities of generating two independent CRT displays for transmission via closed-circuit television back into teaching environments. Depending upon the observation and feedback schedule, the PDP-12 operator can assign any incoming-data box or port to any available display routine. The display routines are written to specification, depending upon the observation system used in any project. The data is usually reduced to specific variables required for feedback in the project and converted into time-lines, bar graphs, or into numerical formats which are interpreted by the teacher in the teaching environment.

A similar procedure is employed for auditory feedback, in that the audio tape recorder functions, as in the CRT display routines, are under control of the computer during the data collection process. Instead of reducing the data to specific variables to be continuously displayed, the auditory feedback process continually monitors the incoming data and checks it against predetermined parameters. When any parameter is exceeded, the computer controls the tape recorder to search the feedback audio tape for the appropriate feedback message for playback to the teaching situation.

Although end-of-session feedback summary printouts are produced by the external time-sharing computer, CATTS is also capable of producing limited summary printouts on the PDP-12 for quick data verification. This function is also done simultaneously with the real-time data collection process. The reason for the limited printout capability is that the system's available memory space is preoccupied with the real-time operating system and CATTS data manipulation functions.

Feedback

The CATTS system analyzes three general types of feedback methods. These methods are the Cathode-Ray Tube (CRT) instantaneous visual feedback, an auditory feedback system, and a summary printout feedback system. These systems are discussed in detail in the software chapter.

Instantaneous visual feedback. The current CATTS system employs two real-time CRT units for real-time transmission of feedback information into two classroom situations simultaneously. The transmission into classrooms is accomplished by transmitting the video feedback image to a monitor placed in the teaching environment. This method permits the use of existing video switching networks that are available in many school systems.

Auditory feedback. This system employs FM transmission to a small pocket receiver placed on the teacher in the training situation. The auditory feedback messages are prerecorded by the individual and accessed through a remote audio tape control system in CATTS program control. CATTS software then determines whether selected behavior parameters have been exceeded, and the appropriate feedback message is computer-selected from the recorded tape to be played back to the teacher in the classroom.

Printed summary feedback. After each data collection session is completed, the raw data is transmitted to a time-sharing computer center where it becomes available to summary printout software programs. Users or supervisors who require summaries of these sessions need only locate a time-sharing computer terminal and request printouts to be printed on their terminal. Different summary programs are available for specific feedback applications and are oriented toward the observation systems in use. These summaries contain information from simple category frequency and rate calculations to extensive category sequence analyses.

Storage and retrieval

The data storage and retrieval procedure for CATTS is a two-stage process; the first takes place in real-time on the PDP-12, and the second on the remote time-sharing computer storage system.

As the raw data is collected from the various input devices, it is initially sorted on the PDP-12 disk unit, together with an identification assignment of a data Starting Block Number (SBN). When the disk unit storage area fills up, the data is transferred to LINCtape for permanent storage at Teacher Education Laboratory (TEL). The data stored at TEL, both on disk and LINCtape, is in raw form and it can only be retrieved by a SBN, even though additional identification information is present within the recorded data session itself.

A backup data storage process is also available in case the disk should fail during data collection. If failure occurs, the system senses the inability to access the disk and automatically calls the LINCtape storage process which insures that no data is lost.

The second stage in the storage process is more complete for retrieval of individual and summary session data. After the data is stored on the PDP-12 disk, it is given a new format and transmitted to the large time-sharing computer (CDC-6600) at Wrubel Computing Center, Indiana University, for storage and analysis. At this stage the data carries a more complete identification structure than the PDP-12's SBN procedure. The data is stored on permanent file disks and magnetic tape with project and project design identifiers. With these identifiers, data may be searched, merged and stored for use in various feedback report programs and project summary analyses. Access to the data is through the CDC-6600 time-sharing system (TELEX) which provides interactive data manipulative programs and functions needed to summarize and analyze the data.

DITRMA: Observation System Training Subsystem

System description. Figure 11 illustrates the configuration for the DITRMA training subsystem. It is similar to the CATTS configuration, in that it collects observation data from button boxes and displays real-time feedback information back to the training situation. In addition, DITRMA ties together up to six boxes for observer category entry comparisons and provides remote control of a 1-inch video tape recorder for prerecorded training materials.

The basic function of DITRMA is to present video-taped training material to observer-coder trainees who enter their observations into the button boxes, where they are compared for agreement between coder entries, and provide feedback pertaining to the result of the comparison. If all coder entries are identical, the system continues to present the video-taped material and provides feedback to the trainees in the form of an audio ("beep") tone. If a disagreement is detected, the system stops the video tape recorder and displays, on an adjoining TV monitor, coder feedback which identifies every trainee's button or category entry for visual comparison. The trainees are then encouraged to discuss their disagreements and to arrive at a consensus as to what the "correct" category should have been. Entering the agreed-upon category restarts the video tape recorder and permits continuation of the training session.

Early training procedures require that a trainer controls with a master button box and assists the trainees in system operation and category definitions. Later in the DITRMA training sequence, the trainees may use the system under their own control in order to finish the training material or to refine category discrimination skills.

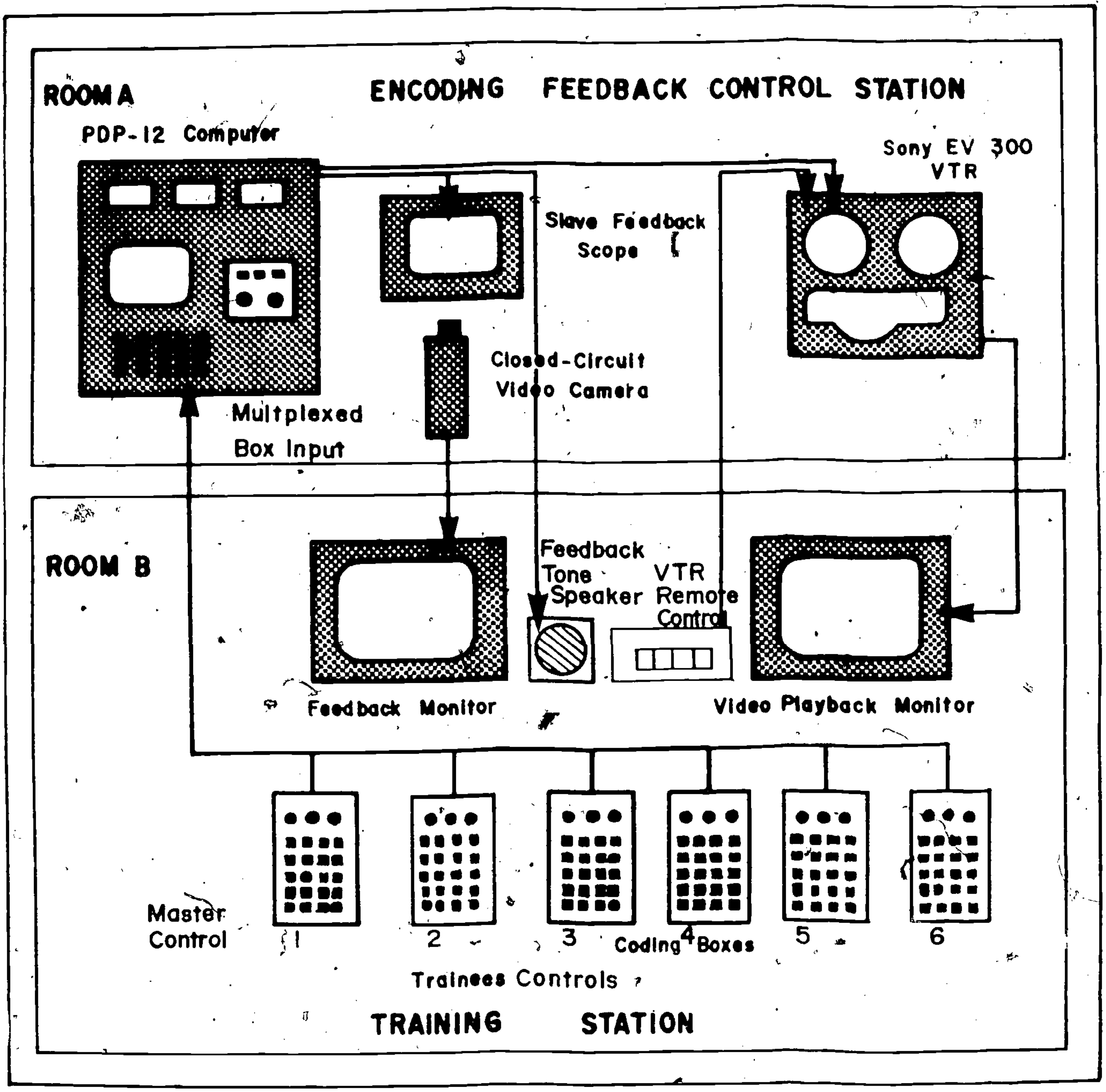


Figure 11. DITRMA configuration.

<u>Hardware configuration</u>. DITRMA adds to the CATTS hardware configuration by connecting to and controlling a remote-control 1-inch video tape recorder. Since a maximum of 12 button boxes already are part of CATTS, the button box connection scheme for DITRMA is already available with the comparison for consensus of codes residing in the DITRMA software. Visual instantaneous feedback is provided back to the training station through the same video camera/monitor vehicle as employed in the CATTS visual feedback system.

The DITRMA training station, as illustrated in Figure 12, consists of six button boxes for trainees, two TV monitors, and a remote-control unit for the video tape recorder (VTR). One of the six button boxes can be assigned for the control box, which is the method for entering the agreed-upon code to restart the system. One of the two TV monitors is used for presentation of training material from the VTR, while the other monitor is used to present trainee feedback when a disagreement is detected. The VTR remote-control unit is used by the trainer, who is controlling the master box, to provide the opportunity to rewind and replay for the trainees any video segment that requires reviewing for discussion. As with CATTS, all processing and control remains at the PDP-12 computer location which is remote from the training site itself.

An additional use of DITRMA, other than the real-time training of observers, is the collection of all data entered by the trainees for summary analysis at a later time. This data includes both the agreement and disagreement entires, with associated interval times. The information that is derived from a postsession summary analysis can lead to analysis of problems in training, such as confusions in the discriminations of certain

Figure 12. The DITRMA station at CITH.

behavior categories. This analysis may lead to trainer classification of certain category operational definitions or the discovery of weak observers who need additional attention.

A more detailed discussion of DITRMA, with examples of feedback available to observers, is found in the software section.

PART II CATTS TECHNICAL MANUAL

CHAPTER 3

Introduction

Part I of this report was written to provide an overview and functional description of CATTS component systems. This section provides the reader with a technical overview and detailed description of CATTS constituent systems, including design configuration, hardware and software systems. The design configuration section discusses the system as a whole, centering on the PDP-12 computer and the various hardware systems that interface with it. A rationale for selection of the various hardware and software components is provided from the point of view of flexibility and effectiveness in teacher training and data collection applications.

The hardware section describes ten specific input and output hardware components used in the system, including the PDP-12 computer and its support components. This section is organized according to the functions each component performs.

The software section is the most extensive in the manual, and includes software design descriptions of CATTS data collection and feedback functions. Due to the unique requirements that the various CATTS functions place upon system resources, all of the real-time software routines were written by TEL staff members. The software description is divided into two general sections; the first is an explanation of the interrupt-driven operating system itself, and the second an explanation of CATTS-related processes such as data collection, feedback and data transmission routines, and the DITRMA system routines. CATTS command lists referred to in the software discussion are found in the appendix.

System Configuration Overview

The general functional applications of CATTS were described in the previous section, and Figure 1 provided an illustration of the system's application in training and data collection situations. The DITRMA consensus coding system was also described and the application of DITRMA to training was illustrated in Figure 11. The central unit which interfaces and regulates the real-time and data collection function of CATTS is a PDP-12 computer. Figure 13 shows the PDP-12 computer configuration with identification of the various I/O interface components necessary to carry out the real-time functions of CATTS. The PDP-12 CATTS console is located in the Teaching Education Laboratory of CITH and interconnects to training classrooms at Indiana University and sites throughout the state of Indiana by directly connected observer button boxes and TOUCH-TONE telephone networks. Descriptions of these various I/O interfaces, which are necessary for data collection and feedback, are described in greater detail in the following sections of this manual.

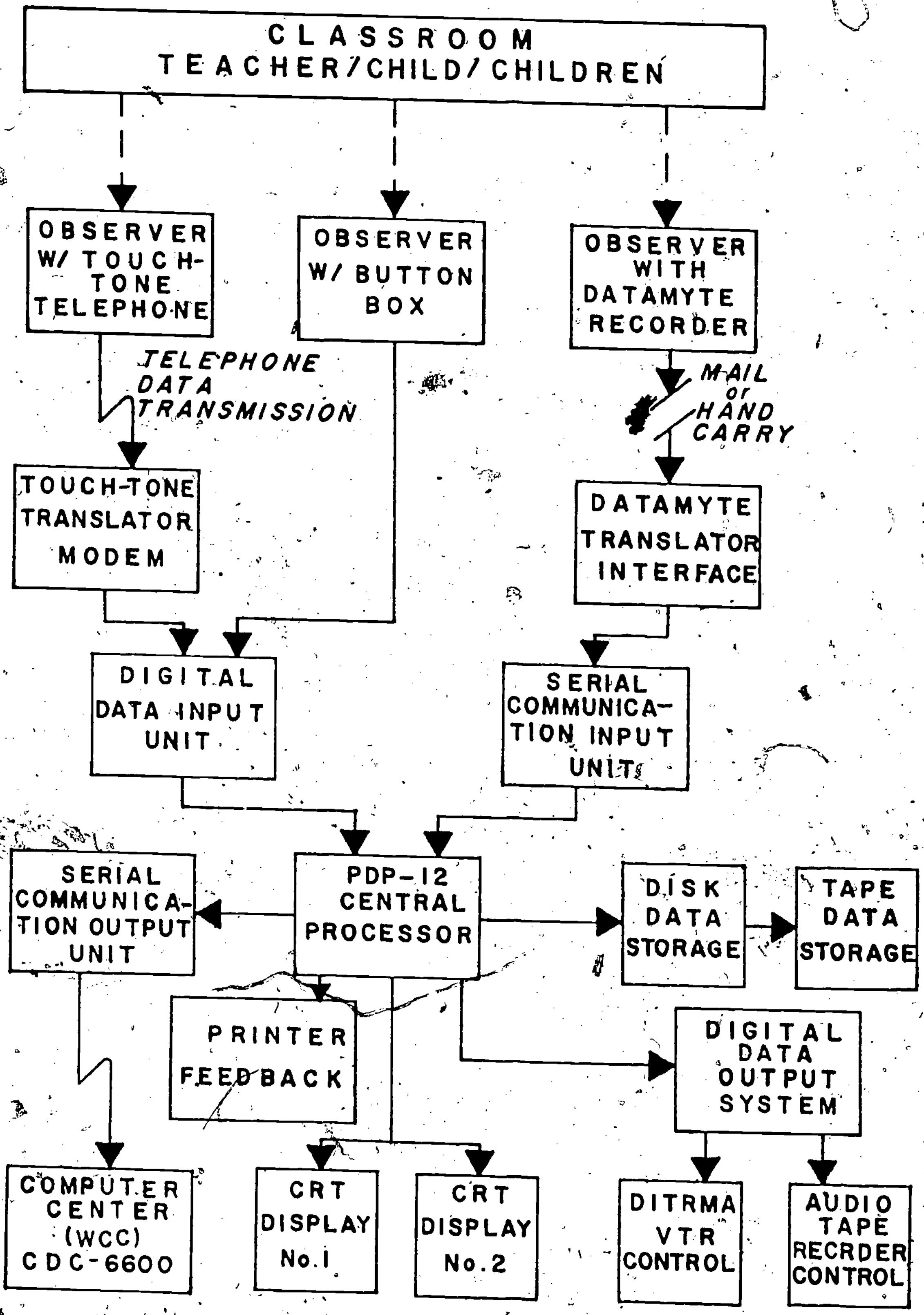

CLASSROOM
TEACHER/CHILD/CHILDREN
OBSERVER W/ TOUCH-TONE TELEPHONE
OBSERVER W/ BUTTON BOX
OBSERVER WITH DATAMYTE RECORDER
TELEPHONE DATA TRANSMISSION
MAIL or HAND CARRY
TOUCH-TONE TRANSLATOR MODEM
DATAMYTE TRANSLATOR INTERFACE
DIGITAL DATA INPUT UNIT
SERIAL COMMUNICATION INPUT UNITS
SERIAL COMMUNICATION OUTPUT UNIT
PDP-12 CENTRAL PROCESSOR
DISK DATA STORAGE
TAPE DATA STORAGE
PRINTER FEEDBACK
DIGITAL DATA OUTPUT SYSTEM
COMPUTER CENTER (WCC) CDC-6600
CRT DISPLAY No.1
CRT DISPLAY No.2
DITRMA VTR CONTROL
AUDIO TAPE RECRDER CONTROL

Figure 13. CATTS functional data collection and feedback process.

System Design Considerations

Hardware. The prime consideration for hardware selection is the implementation of a responsive interrupt-driven central processor. The nature of the CATTS data collection functions places the requirements of many simultaneous input devices on the system, and, due to the requirement of assigning an accurate time interval to every data entry, the system must respond as quickly as possible to every external input event. Therefore, to employ real-time data collection software, an interrupt-driven processor such as the PDP-12 was selected for CATTS.

To further increase the responsiveness of the software, an Automatic Priority Interrupt (API) hardware system was included within the processor. The API allowed the physical assignment of the various input and output devices to be tied directly to different levels of the processor's interrupt structure. This means that interrupt requests occurring at a higher assigned level of priority can interrupt processes taking place at a lower level of priority, and, upon conclusion of the high priority process, pass the processor back to the low priority interrupted process.

Another advantage gained by the API is that, when an external interrupt occurs, processing time need not be taken up with identifying the device source of the interrupt. Instead, processor control can be passed directly to the specific requesting hardware device.

The primary purpose of an API installation is to reduce the time that external I/O requests require for processing, thus maintaining the accuracy of the assignment of event time to incoming data. An API also assists in a more efficient assignment of tasks that use the slower input and output devices. For example, printing, which is slow in comparison to internal processing, can be simultaneously time-shared along with other processing functions. This device and processing time-sharing capability increase the flexibility and responsiveness of the CATTS system.

Other hardware considerations employed in CATTS are centered around installing I/O systems which include their own controllers as part of their function. This is common in hardware devices, but due to the speed required to respond to external interrupts, the processor is dependent upon as much outside I/O control as possible. The PDP-12 configuration employs many built-in devices, such as a point and character generator for the two CRT channels, a multifunction real-time clock, and an interrupt-scanner controller for the digital I/O system. The most important prerequisite in designing the hardware-observer interface is to provide the simplest method of data entry for an observer, so that an observer is able to focus his/her attention primarily on the interactive behavior being coded. This design objective is met by adapting a 12-key pushbutton keyboard similar to a TOUCH-TONE telephone keyboard. The observer enters combinations of pushbutton numbers to represent assigned categories, then pushes the lower left-hand button TOUCH-TONE telephone " * ") key to enter the observed data into the system. The previous data is not recorded until this "ENTER" (" * ") key is pushed. If the observer makes a mistake in the category numbers and recognizes the mistake before pushing the "ENTER" key, the observer may push the lower right hand "SKIP" key (TOUCH-TONE-Telephone " # " key). The "SKIP" key clears all data entered since the last "ENTER" key and resets itself for a re-entry of data.

Since event time is automatically assigned by the computer upon the receipt of each "ENTER" key, the observer is left free to concentrate on the behaviors being observed. When data is entered by the observer into the keyboard, the compact configuration of the pushbutton keyboard permits uncomplicated entry.

Another consideration for employing a TOUCH-TONE keyboard is to permit a positive transfer of data entry skills from directly connected boxes to TOUCH-TONE telephones for data entry. Observers are then able to transfer between box and telephone input units without retraining. The DATAMYTE portable data recorders are also specially ordered with the pushbutton pad altered into a TOUCH-TONE format. In order to design the CATTS system for ease of application in classrooms, easy connection to classrooms is of prime importance. To facilitate this connection, existing closed-circuit television lines are used to collect and feedback data into classrooms. This adaptation of an existing (CCTV) switching network allows plug-in access throughout the public school complex that is adjacent to the CATTS hardware location. The care in selecting hardware that facilitates time-sharing applications, data entry, feedback applications and classroom accesibility, ties into the increased cost-effectiveness of implementing CATTS into developmental and training situations. Increasing the number of classrooms or observation rooms that can be serviced by one hardware facility decreases the cost involved to serve those classrooms. Accessing classrooms quickly without major equipment installations also permits time-sharing of existing CATTS facilities across a wide range of applications. By following the hardware design considerations discussed above, many classrooms within a school complex or, through TOUCH-TONE telephone lines, within a school district, can be served by one responsive facility.

Software. As with the hardware considerations, the development of a responsive software time-sharing program that takes full advantage of the hardware is the most important task. The ability of the software (a) to respond quickly to external input events, (b) to process incoming observer data from many simultaneous sources, and, (c) to simultaneously provide output feedback to as many locations in as many forms as are required, constitutes the flexibility and applicability of CATTS.

The design of the software to reduce the complexity of observer data input and to increase the readability of feedback output is also an important consideration. Great care is taken to reduce the potential impact of complex computer technology on the people who use and interface with CATTS.

CHAPTER 4

Hardware Components

Figure 14 illustrates the current PDP-12 system configuration currently in use to conduct CATTS research and development activities. The system consists of two basic configurations; (1) the central processor unit and associated memory devices, and (2) input/output peripheral devices. Figure 15 illustrates the same configuration, but reorganizes the configuration into an input and output hardware function diagram. Table 1 lists the component parts and model numbers of the Digital Equipment Corporation (DEC) supplied system. This list does not include the support hardware components which will be described below.

Central Processing Complement

Central processor. The PDP-12 central processor is a dual processing system with two instruction sets for assembly language programming applications. One processor is a self-contained PDP-8I which is programmed with DEC PAL assembly language syntax — the primary language employed for external I/O processing. The PDP-8I PAL language organizes memory in 128-word pages and is fully compatable with other programs written for PDP-8I processors. The PAL instructions are used for all communication to externally connected I/O hardware, including the RK8-EA disk unit, the UDC8 digital I/O subsystem, the KWI2 real-time clock, the KF12 multilevel Automatic Priority Interrupt, and the communications and paper tape I/O units accessed through the BA12 peripheral logic expander.

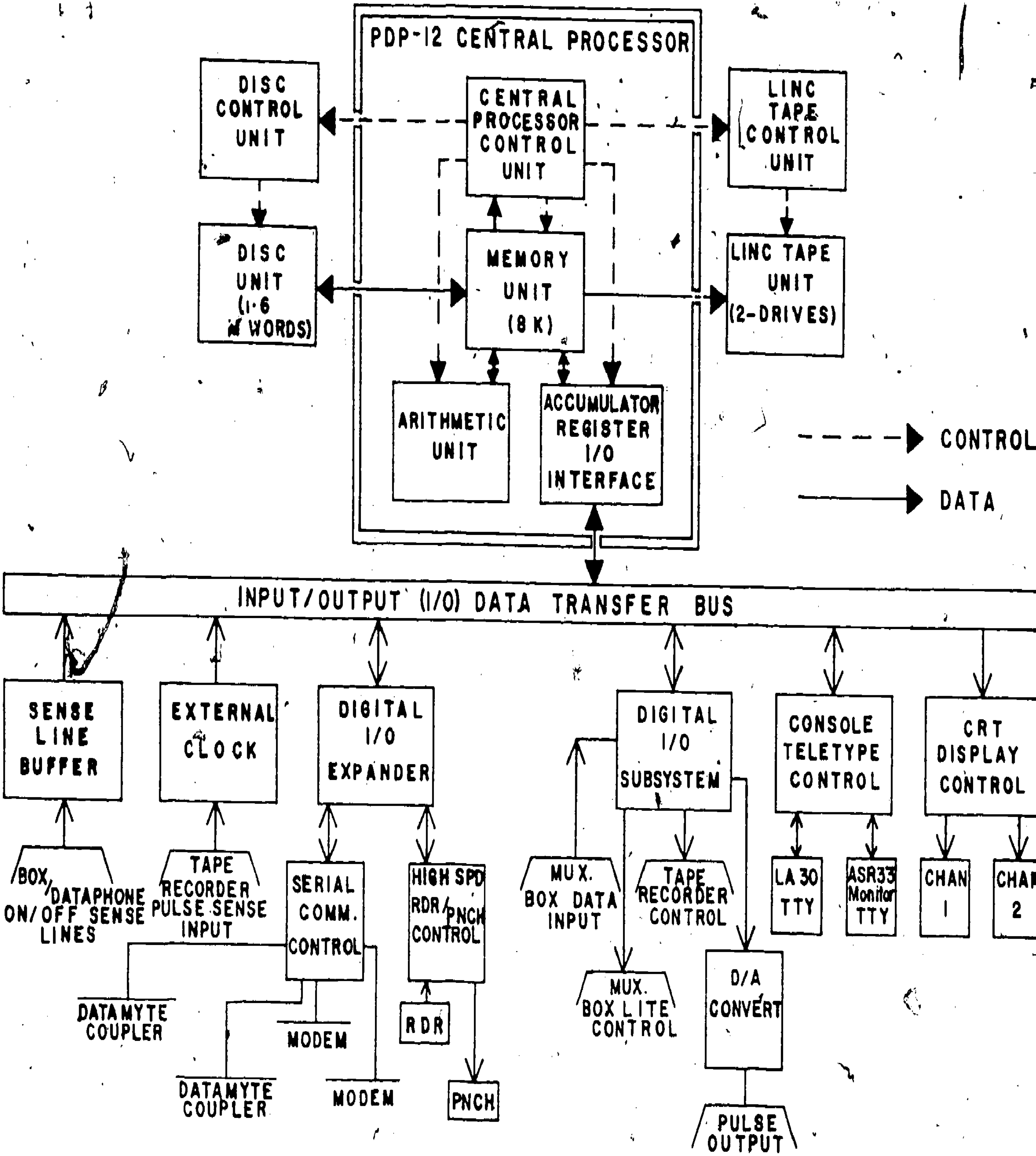


Figure 14. CATTS PDP-12 hardware configuration.

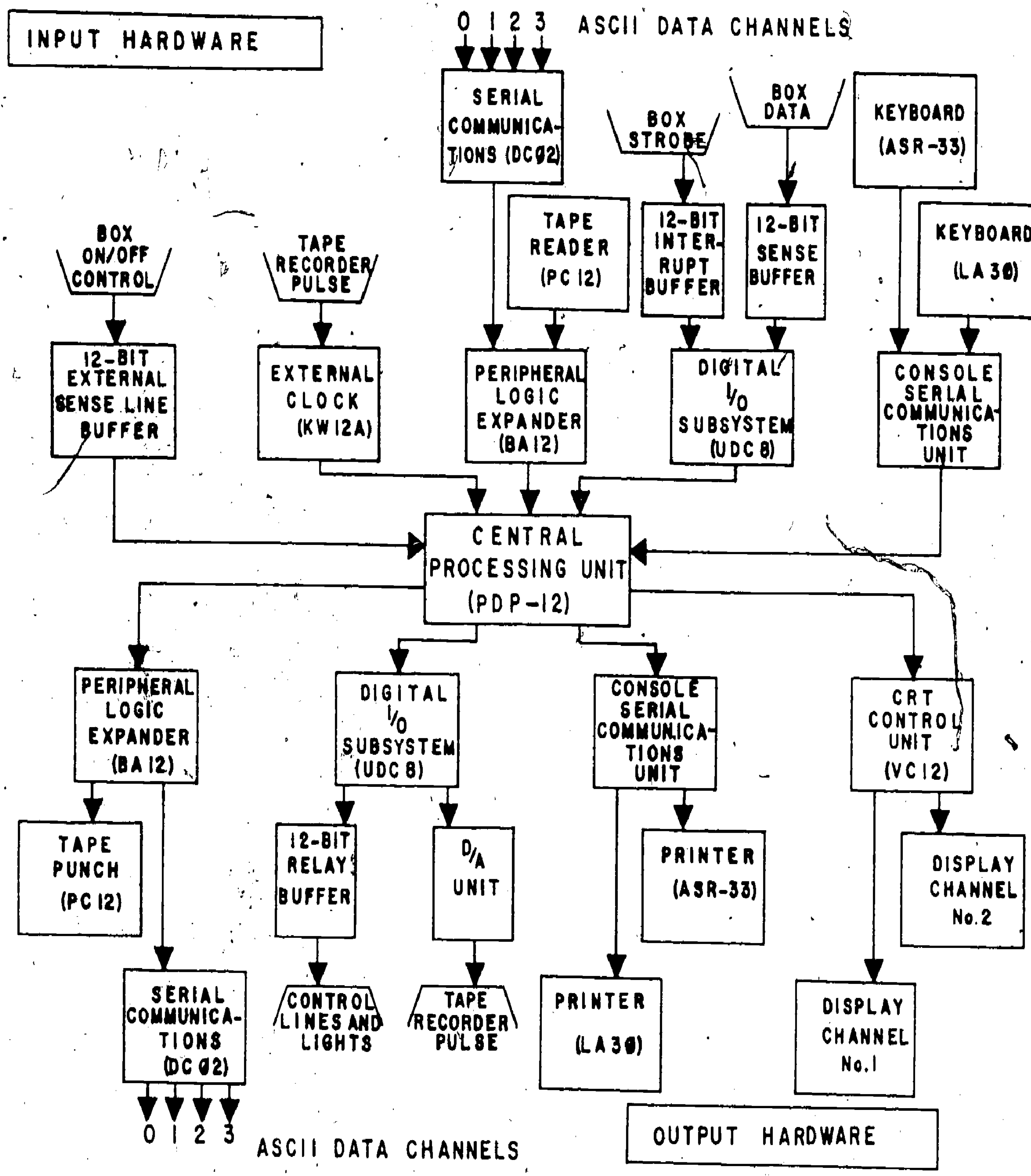


Figure 15. Input/Output hardware functional configuration.

Table 1

CITH/TEL PDP-12 COMPUTER SYSTEM

PDP-12/30 Advanced LINC Computer System including:
-PDP-12 Dual Instruction set Central Processor
-4096 12-bit, 1.6 us Core Memory
-Direct Memory Access Channel
-KF12 Multilevel Automatic Priority Interrupt
-MC12 4K memory extension
-KW12-A Real-time Clock
-TC12 LINCtape Automatic Control, fully buffered, DMA Transfer
-TU56 Dual Tape Transport - LINCtape
-RK8-EA 1.6M word Disk Cartridge system (including controller)
-DW8E-PA Positive I/O Bus to OMNIBUS converter
-BA12 Peripheral logic expander
-VC12 LINCscope Control and Character Buffer, 2 Intensity Channels, 2 sized Charcters
-VR14 7" x 9" CRT Display
-AD12 Analog-to-Digital Converter and Multiplexer, 16 Channels (8 knobs), 10-bit accuracy, Sample and Hold, Differential Preamplifiers, 69 kHz conversion rate, expandable to 32 Channels
-DR12 6 Programmable SPDT Relays
-Data Terminal Panel
-LA30 DECwriter, 30 Char/Sec.
-ASR-33 Teletypewriter, 10 Char/Sec Paper Tape Reader and Punch
-6 Sense Switches
-Hardware Signed Multiply Instruction (9 us)
-15 Auto-Index Registers
-12 Sense Line Inputs
-KE12 EAE Unit (Extended Arithmetic Element)
-KP12 Power Fail/Restart
-VR14 Slave CRT
-PC12 High-speed Reader/Punch
-DC02 Multiplexed 4-Channel communications unit
-UDC8 Digital I/O Subsystem w/Input and Output Buffering and Interrupt Logic
-30" Freestanding Cabinet
-Console Table

The LINC instruction set is used to program and control the LINC processor, which is a special processor used to control devices such as the TC12 LINCtape units, the VR14 display scopes, the DR12 relay buffer, and the AD12 16 channel analog-to-digital converter unit. These devices are not accessible to the PAL language and, in order to make full use of the PDP-12 system, control is passed back and forth between the LINC and PDP-8I processors for maximum use of devices and code efficiency. The LINC instruction set organizes memory into 1024-word segments and contains its own set of auto-indexing registers similar to the PDP-8I instruction set. The power of the LINC instruction set is its access to the LINCtape controller, the display controller (together with both the point and character display buffers), and the A-D converter controller for fast external signal-sampling applications.

Both instruction sets have full capability to process numerical and control types of code, but the instructions differ in control of external I/O hardware processing. The CATTS software system uses both instruction sets interchangeably towards the most efficient use of the total hardware complement.

Multilevel Automatic Priority Interrupt (API). Table 2 lists the current device assignments for the API hardware device unit connected to the central processor. The function of the API is to pass control to designated I/O routine if an interrupt occurs on any specific priority level. The API saves the current machine level and arithmetic registers, then passes control to the interrupting level, and restores the current machine level and arithmetic registers when the interrupting process is completed. The advantage of the API is that interrupts of a higher

Table 2

Automatic Priority Interrupt (API) Level Assignments

API Priority Level		Device
(Decimal)		
0	0	-KP12 Power Fail/Restart Unit
1	1	-TC12 LINCtape Unit
2	2	-KW12 Real-time Clock
3	3	(Unused)
4	4	-LA30 Console Teleprinter Unit
5	5	-LA30 Console Keyboard Unit
6	6	-ASR-33 Auxillary Keyboard Unit
7	7	-ASR-33 Auxillary Teleprinter Unit
8	10	(Unused)
9	11	-RK8-EA Disk Unit
10	12	-BA12 Peripheral Logic Expander with DCD2 Communication Unit and PC12 RDR/PNCH Unit
11	13	-UDC8 Digital I/O Subsystem
12	14	(Unused)
13	15	(Unused)
14	16	(Unused)
15	17	-Background/Machine Processing Level

priority than those currently being executed can take control of the processor, and lower priority interrupts that occur during the current processing of a higher priority interrupt can be saved and processed in order of priority after current higher priority processing is complete.

Real-time programmable clock. The clock is used for the main timing functions of the operating system by supplying interval time buffers to routines that require timing information. Time is used in the CATTS operating system for visual CRT displays, data interval timings, and time-out routines employed in data collection and DITRMA training. The clock is also used to decode the prerecorded audio location signals present on the control track of the tape recorder used in the auditory feedback system.

Extended Arithmetic Element (EAE). The PDP-12 CPU (Central Processing Unit) has an extended arithmetic processing unit to assist in mathematical processing by using the MQ (Multiplier Quotient) register as an additional 12-bit extention. The EAE permits the use of an additional set of PDP-8I instructions for 24-bit arithmetic processing.

Auto-index registers. Both PDP-8I PAL and LINC instruction sets make use of hardware auto-index registers for programming applications. These registers have different functions depending upon the instruction set being used and the application required.

Power fail/restart. The power fail/restart system provides hardware that detects a potential drop or approaching drop in the AC power level to the processor. This causes an interrupt to level 0 of the API, which saves the current machine level and register status before the actual power fail occurs. When power is restored, the hardware automatically restores the original machine status and continues processing without loss of data or important information processed prior to power failure.

Hardware signed multiply instruction. The LINC instruction set provides a 9μs signed multiply instruction for increased arithmetic processing.

Memory and storage

Core memory. The PDP-12 core memory is an 8K 12-bit word configuration which is organized in two 4K stacks for the PDP-8I assembly language requirements, and into eight 1^K stacks for LINC programming. Any portion of memory is available to both processors.

LINCtape units. The PDP-12 system also supports two LINCtape units which are switch-selectable for logical unit assignments from unit 0 through 7. Each unit's internal data organization is identical to that of the disk unit internal organization in that each unit contains 896 blocks of data with 400 octal 12-bit words each. Program and/or data information is read or written in these 400 octal word blocks minimum for the tape units and half block (200 octal words) minimum for the disk logical units. The LINCtape units are programmed through the LINC assembly language instruction set.

Disk unit. A 1.6 M word (12-bit) moving head disk cartridge device is available to the central processor for data and program transfer. The disk is organized logically into seven contiguous LINCtape units that are accessed through the disk instructions found in the PDP-8I assembly language instructions. The logical unit assignments are 10 through 16.

Direct Memory Access channels (DMA). Communication to and from the central processor is through the DMA on a data break cycle scheme. This permits data and program transfer units without loss of program control. A tape and disk unit interrupt structure allows program response to be keyed to the completion of tape and disk transfers.

Input Hardware

UDC8 I/O input subsystem. The Universal Digital I/O subsystem contains incoming interrupt identification and scanning controller to isolate external events, such as button presses and a scanning controller to identify the specific source of the interrupt. When the source is identified, the UDC8 interrupt is presented to the central processor for processing.

The 12-bit interrupt module of the UDC8 corresponds to any of the 12 possible parallel data input ports originating from the directly connected observer button boxes or TOUCH-TONE telephone Data Sets (Module 403-E Data Set). The correspondence from any coder-initiated external button press interrupt is to 12 sets of 4-bit voltage sense points that are sampled from the central processor and processed as BCD coded data. These 4-bit codes allow entry of up to 16 separate categories for numerical and control character processing. Currently, the numbers 0 through 9, and the control characters "ENTER" (send) and "SKIP" (clear) are used. The four remaining unused codes are available to CATTS for additional control or data characters for future applications which may require them.

Console keyboards. The central processor has available two console input keyboards for external operator control. The CATTS monitor receives its directives from a 10 CPS ASR-33 keyboard. All communication with the CATTS monitor originates from this keyboard. An auxilliary

30 cps keyboard is also available to CATTS and is used for independent communication to external time-sharing computing systems for data transfer. This 30 cps LA30 operates independently from the CATTS monitor. In case of a ASR-33 malfunction, CATTS monitor control can be transferred to the LA30.

Input communication ports. There are four multiplexed asynchronous ASCII input ports available to the central processor for external information input requirements. These ports accept standard 8-bit character transmission and present an assembled character buffer to the central processor for processing. Two 300-baud ports are currently assigned to Data Sets for external communication to computer time-sharing systems. The remaining two ports are connected to the Electro-General model 804 DATAMYTE decoding units which decode observation information collected on cassette tapes with the DATAMYTE model DAC-8 portable collection units. The cassette data tapes are decoded and translated into 8-bit ASCII characters, and transmitted to the DC02 communication ports at 110 baud.

High-speed paper tape reader. A 300 cps paper tape reader unit is also available for data and program input. Its use is restricted only to those times when both the disk and tape units are nonfunctional.

Eight-channel Analog to Digital (A-D) converter. The LINC processor and its instruction set can access a multiplexed eight-channel A-D converter device. Currently CATTS does not require the use of this device.

Twelve external sense lines. The central processor also has a 12-bit external voltage sense buffer that can be sampled directly through the LINC instruction set. This 12-bit buffer is currently used to detect coding box "on-off" status for the CATTS operating system.

Output Hardware

UDC8 I/O Output subsystem. The three output components available to CATTS from the UDC8 are the solid-state flip/flop output lines, the relay control lines, and the four-channel multiplexed Digital to Analog (D-A) converter. Through program control, any of these three components can be selected to perform an output function. CATTS uses the solid-state flip/flop output lines to control small LED (Light-Emitting Diode) feedback signal lights located on each of the directly connected button boxes. The relay control lines are used to operate, under computer control, video and audio remote-control tape recorders, and the necessary connections needed to control and answer-back signals required by the various data phones. The D-A converter is used to generate frequency-coded signals on the alternate or control track of an audio tape recorder which is used to locate the appropriate message for playback during the audio feedback application. This is accomplished by using the D-A to generate a tone of a specific frequency.

Console Printers. The two printers associated with the two console keyboards are used in the CATTS operating system. The CATTS monitor printer is the ASR-33 and it types out prompts, status messages, and error messages when requested or generated from the operating system. Because of the reduced and periodic print generated from the monitor, the 10 cps print speed of the ASR-33 is more than adequate for the function served.

For other printing needs, such as memory or data tape dumps and print generated from a remote time-sharing system, the faster LA30 30 cps matrix printer is used. This printer, like the LA30 keyboard, is completely independent of the CATTS monitor.

Output communication Ports. The multiplexed four-channel asynchronous output ASCII character ports represent the companion unit to the input communication scheme described above. Since the DATAMYTE 608 decoders do not require output characters from the PDP-12 system, only two of the four output ports are used. These two, 300-baud ports correspond with the two Data Phones that communicate with the remote time-sharing systems. These two parts, with ASCII input and output capability, allow the CATTS operating system, through a time-sharing scheme, to become a remote terminal for data transfer to and from the CATTS PDP-12 system and the time-sharing system.

High-speed paper tape punch. The 50 cps paper tape punch is used by CATTS as a back-up data dump in case of a data storage malfunction in both the tape and disk storage units.

CRT Displays. Two independent CRT displays with ~~point~~ and character generation subprocessors are available to the PDP-12 system through LINC mode instructions. CATTS uses the displays for instantaneous visual feedback applications. Data is displayed on the CRT screens located in the laboratory and transmitted to the remote classrooms via closed-circuit television.

Support Hardware Components

Solid-state data entry button boxes. Figure 16 is a schematic representation of the directly connected parallel data button boxes used by the observers to enter data into CATTS. The encoded keyboard circuit and the bus driver circuit combine to provide the UDC8 input sense lines and interrupt lines with a 4-bit data code and a single line interrupt pulse. The remaining lines into the button box provide power to the solid-state modules, coder feedback light controls, box on/off sense circuit, and common signal power grounds.

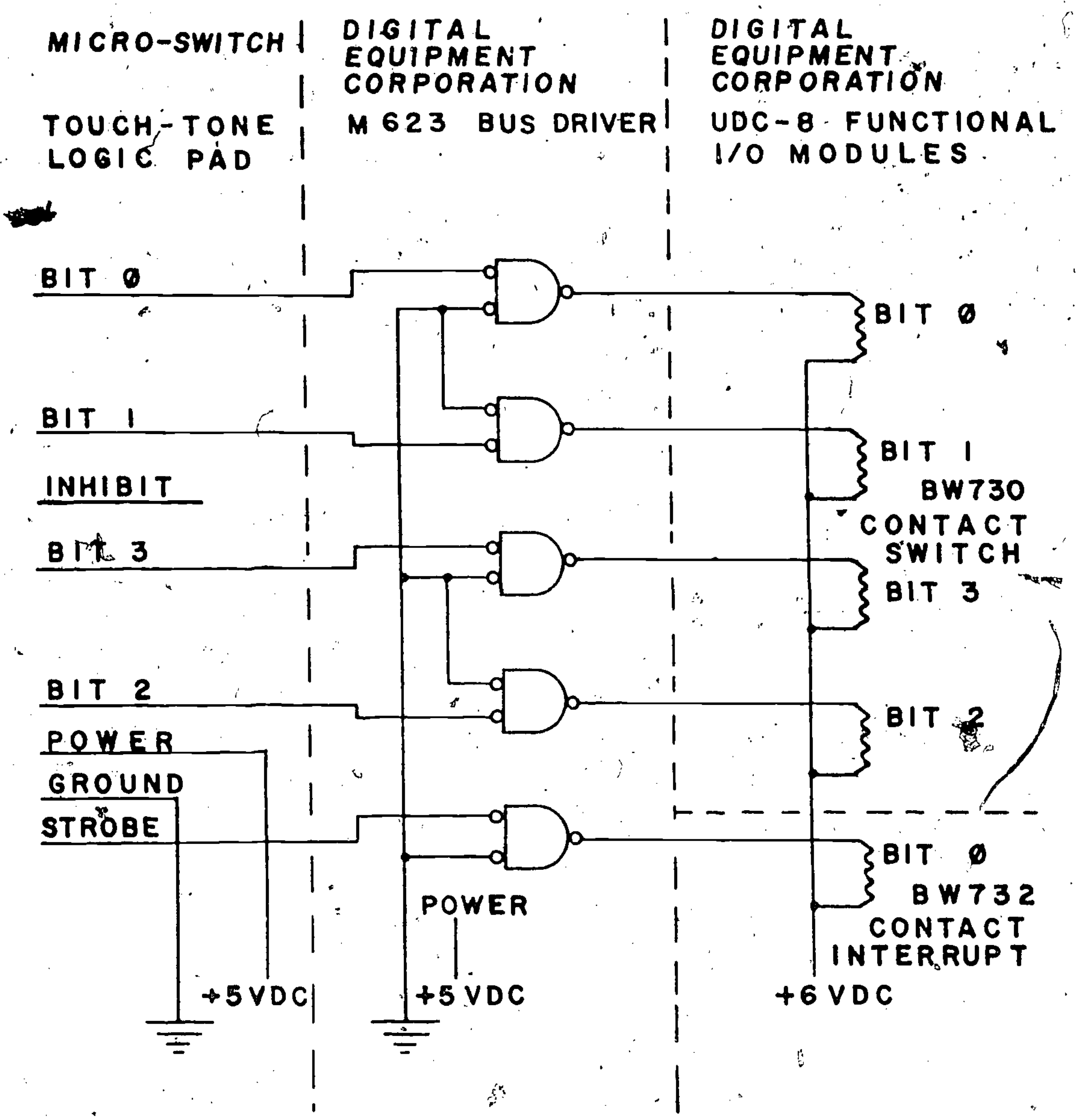


Figure 16. Solid-state button box schematic.

DATAMYTE model 608 coupler/decoder. Two couplers and two cassette tape recorders are connected to the PDP-12 through the DC02 communication ports at 110 baud. CATTS collects the data as ASCII characters and converts the characters into an internal code identical to the code collected by the directly connected button boxes. Thus, at any later time, the couplers can simultaneously process DATAMYTE data with the collection of real-time data.

TOUCH-TONE Data Set. The Bell telephone model 403-E TOUCH-TONE Data Set converts the TOUCH-TONE telephone signals into a 4-bit parallel signal together with a strobe or interrupt signal for real-time data collection from remote classroom locations. Each remotely located TOUCH-TONE telephone requires a corresponding 403-E Data Set for decoding data signals. The visual application of a real-time TOUCH-TONE data collection system is to outfit a school with telephone jacks in selected classrooms and provide observers with a plug-in TOUCH-TONE telephone to carry from classroom to classroom.

ASCII code Data Set. A Bell telephone 103-B Data Set is used for two-way communications between a remote computing center time-sharing system and CATTS for raw data transfer to mass storage files, and for remote transmission of CATTS summary feedback generated by CATTS on the PDP-12 to telephones located in schools where TOUCH-TONE data collection is taking place.

Automatic Calling Unit (ACU). A Bell telephone 801-A Automatic Calling Unit is used by CATTS to automatically place a call to the remote time-sharing service for data transmission to user storage files. The ACU removes the necessary PDP-12 operator intervention after each data collection session is completed and data transmission is required.

CATTS is programmed to automatically call the computer center after each session, establish contact with the time-sharing system, transmit the data, and verify success of transmission.

Remote-control Video Tape Recorders. Two Sony AV300/AV320F 1-inch remote-controllable tape recorders are connected to CATTS through the UDC8 relay buffer. The tape recorders are used in the DITRMA system where the tape stopping and starting is under program control.

Remote-control Audio Tape Recorders. Two TELEX model 1022 audio tape recorders are connected through the UDC8 relay buffer for remote-control operations tied to the auditory feedback system. The recorders are stopped, started, rewound, fast forwarded, and placed into record mode by CATTS in order to fulfill the functions required.

TOUCH-TONE telephone to UDC8 interface. A solid-state interface was designed to translate the signals generated by the 403-E Data Set, making them compatible with the sense-line inputs located in the UDC8 interface. The translation of these signals then corresponds to the same voltage range of the directly connected button boxes. This translation permits CATTS to process the incoming data signals the same for all real-time data collection devices.

Miscellaneous Audio-visual equipment. In order to operate DITRMA and display CATTS immediate visual feedback, various pieces of audio-visual equipment are used. This equipment is not directly connected to the PDP-12.

CHAPTER 5

Software Operating System Components

Operating System: An Overview

The CATTS real-time foreground/background operating system was developed at CITH using the DEC DIAL-MS operating system. The 8K core limitation prompted system development in the early stages of CATTS research and has since evolved into a flexible and responsive system that meets the specific requirements of CATTS applications.

The basic design of the real-time operating system is comprised of foreground I/O handlers and a background processing loop. The foreground handlers are assigned to different levels of the API, and the background loop contains the CRT refresh and task schedular routines. The system is controlled through a console teleprinter monitor routine which permits operator control of and assignment of all CATTS functions. The resident operating system resides in field 0, and various CATTS application routines reside in field 1. When called, some application routines remain core-resident, while other quick-run utility routines (print, status, etc.) are scheduled into a shared 2K utility area of core. All application routines may be loaded or unloaded through monitor control. A memory map is maintained, and current core status reports are made available through the monitor.

The system is generalized to accept data from any type of observation system, and, in turn, to reformat the data into standardized data file structpres. Backup routines for system storage malfunctions are also part of the core-resident system and they are able to automatically exchange disk, tape, and paper tape handlers in order to insure against data loss.

The primary requirements that the operating system was designed to process are: 1) simultaneous data collection from three different sources, 2) real-time CRT feedback on two independent channels, and 3) an intermediate data storage and transmission facility for data transfer to a remote computer time-sharing system.

Data Collection. The three data collection sources (parallel input button boxes, TOUCH-TONE telephone input, and DATAMYTE cassette tape input) are processed simultaneously through a foreground/background interrupt design. The API hardware system provides the facility for responding to external input events in the foreground mode by placing the data onto a background stacking routine for delayed processing. Time intervals between data entries are appended to each individual incoming data point by the foreground input and clock handlers. Incoming DATAMYTE data carries its own corresponding digit time intervals and skips the clock routine.

Real-Time CRT Feedback. The two CRT feedback displays are refreshed from the background display loop. Display buffers are filled with information extracted from assigned incoming data streams. Depending upon the observation system being used to collect the data and the type of CRT feedback to be displayed, different routines are loaded and scheduled during data collection. These individualized display processing routines are the only components within the real-time generalized system that require continuing program development for feedback applications.

Intermediate Data Storage and Transmission. Due to anitcipated time lags and down-time periods in the remote computer time-sharing service, an intermediate data storage disk unit contains completed observation session data files. Each completed data collection session schedules a data sort routine to extract data points corresponding to that input source from an interlaced data scratch file, and appends the sorted data file onto the intermediate file unit. When the remote time-sharing system is available, each file is automatically transmitted to mass storage through a scheduled background loop transmission routine.

Printed data summary processing and analysis is performed on the remote computer time-sharing system. Upon completion of a coded observation session and automatic transmission of the data onto mass storage files, the operator nay log onto the remote computer service from any time-sharing terminal and may request various summaries of selected data files. The decision to transfer all user summary processing to a larger system is prompted by: 1) more efficient and larger storage facilities, 2) multiple user access to data files, and 3) various summary and analysis programs available in higher level languages.

Operating System (O.S.): Functional Description

This section covers the software implementation of the CATTS data collection and feedback functions. CATTS requirements are 1) to collect data from three sources to provide various real-time, instantaneous audio and video forms of feedback during the data collection, and 2) to generate a printed form of feedback at the end of the lesson.

The operating system developed is required to support these various real-time tasks and to fit the PDP-12's memory and interrupt-handling hardware. Specifically, the operating system is a set of procedural rules that manage these computer resources, handle program flow, increase the adaptability of application programs by handling and standardizing input/output functions, and provide program organization guidelines. The CATTS application programs also make use of the hardware I/O features and conform to the procedural rules adopted for the operating system.

O.S. Resources: Features and Use

The teletype hardware interfaces, although programmed somewhat differently, have a standard software interface. The standard console teletype and the additional teletype interface, option DPI2-A, are interchangeable, as is the command console. The command console is a shared device for interactive routines, programmed control functions, and error message printing. Any of the four DC02-D stations are selectable at program run time, and interchangeable from the command console. The high-speed paper tape unit is interfaced in the same manner as a teletype unit, even though the tape reader requires a "fetch character" initiating instruction.

The LINCtape and disk have a similar block data transfer structure. These units are shared and interchangeable for data and program storage functions.

The UDC8 unit is hardware dedicated to particular CATTS I/O applications. The button box function can have each box individually assigned to a different use, such as data collection, DITRMA, or interactive operator control.

The computer lab data panel, which includes relays, A-D channels, and programmable Schmidtt-triggered clock inputs and controls is also hardware dedicated to individual functions.

The programmable clock is able to have tasks added to its timing functions. Initially, it provides a real-time interval interrupt that is shared by Operating System routines and application programs.

Individual data bits of the hardware Special Function register for program control are read and manipulated at will for hardware/software programming code efficiency.

The software interrupt structure is augmented by an Automatic Priority Interrupt (API) hardware option so that a single-level interrupt feature is not used. The LINC mode instruction interrupt trap feature has also not been implemented.

The CRT display scopes are program refreshed and refresh routines are added by the O.S. as needed for TTY "printer" type displays and data collection feedback displays.

The left and right console switches and the program control sense switches on the programmer's console are used by only one routine at a time under operator selection. The external level sense lines are hardwired with some of the UDC8 module I/O boards. There are 12 external sense lines.

Memory is shared according to two methods. One method shares a partition of memory as though it were a device of its own, and the other method accepts absolute address program loads.

Multi-tasking

A task is one of the cooperating routines that collectively constitute a program. Some tasks that handle a particular device take their computer time through the jumper wire assigned to a priority interrupt level of the API. There is no software priority assignment or arbitration for non-interrupt tasks because the PDP-12 does not have the necessary stack instructions for code to be shared across priority levels. Initiating requests for these non-interrupt tasks are lined up in an Attention Stack. The code that continues to loop at the background level of the API deletes these requests to be handled in a sequential manner.

Therefore, through the Attention Stack, external interrupting events initiate background tasks that can use shared and re-entrant code.

A background task can then, in turn, initiate an external event which may be a task that uses another interrupt device to initiate output to a printer, to initiate a disk read, or disk write instruction.

If there are no other tasks which use a specific device, a task can use the dedicated device at will. But, if the device is shared, requests for it must be stacked for sequential handling. The Schedule routine performs this function and can be called only from background tasks. An Eject function handles the transitions from one request to the next for all device handlers.

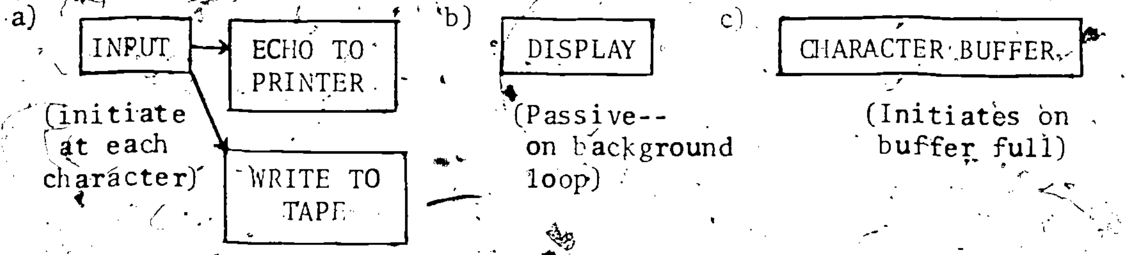
Generally, there are two methods of inter-task communication. The first, from an external interrupting event through the Attention Stack to a background task, corresponds to the input step. The second, from a background task to an external interrupting event, possibly using the Schedular, corresponds to the output step. The buffering and compute step generally occurs at the background level where math packages and buffering routines can be shared.

Some device resources are used only by a particular program to execute a specific hardware function. Others, like keyboard input, may have several uses, such as being dedicated to a program only while that program is core resident with that selected device. Other device resources, like the disk and printers, are shared by several programs that are loaded simultaneously.

The general approach to implementing the CATTS O.S. is to conceptualize system requirements as three separate steps of 1) input, 2) compute, and 3) output. Since input and output run at different speeds, intermediate buffering is required and grouped with the compute step.

Also, since most CATTS programs use more than one device, it is convenient to separate the use of each device into a separate routine. Therefore each application program takes the form of a set of co-routines, with some common data areas shared by two or more of the co-routines.

Examples:


a)
INPUT
ECHO TO PRINTER
WRITE TO TAPE
(initiate at each character)
b)
DISPLAY
(Passive-- on background loop)
c)
CHARACTER BUFFER
(Initiates on buffer full)

Attention Stack

As the PDP-12 does not have the necessary stack instructions for code to be shared across API levels, a combined interrupt and polling method allows characters from different devices with different API priorities to be handled by the same program.

A small routine that uses only the AC (Accumulator) and MQ (Multiplier Quotient) central processor registers can be shared because the AC and MQ are automatically pushed onto a register "Save, stack by the API when an interrupt is serviced. The API then vectors machine control to discrete handler code, which clears and responds to the interrupt. The handler may want to continue processing at the background level, or it may cause a character to be passed to the background level. To do either, an item is queued into the Attention Stack.

Stack Description. Each item on the Stack consists of two 12-bit words. One of the words is the 12-bit address of the waiting background routine. The three higher ordered bits of the second word indicate the field of the waiting background routine. The other nine bits are passed to the waiting background routine and can be the data character read. If required, the indicator of a device from which a character originated can also be encoded into the 9-bit character field.

Input Enable. Each handler has an input enable field register and address register. When input is enabled, the field and address registers contain the 15-bit address of the waiting background routine. When input is disabled, the enable address contains a zero.

Each handler can read the character, set the field bits and load the field into the MQ. The handler then transfers the enable address into the AC.

The routine that places the field and address words onto the Stack will test the AC. If that value is zero, these words will not be placed onto the Stack before the machine is restoreu to the "before-interrupt" status. If nonzero, these words are inserted onto the Stack using an auto-index register. Interrupts are temporarily suspended so that the two words are not separated by another word pair, since this insert routine is shared across all API levels.

The polling is done in a background loop (API level 17_8) which monitors the Attention Stack insert pointer. When the Stack Delete routine detects a stack entry, it removes that stack item and passes the character field to the waiting background task. When the task is completed, the polling monitor returns to the Delete routine to look for additional entries on the Stack. The insert pointer is reset when the stack deleting process is complete.

Control is passed to the waiting background routines from the Attention Stack delete code with a PUSHJ instruction. The last command of all waiting background routines is a RESTORE instruction which returns control to the Delete routine. Ending routines with a RESTORE allows multiple entry points to that routine. Entry at those points can be made as an unconditional transfer using a JMP instruction, or as a subroutine call using the PUSHJ instruction.

The delete code is a task in the background loop. The other tasks in the loop are normal displays. Waiting background tasks acquire machine time only when requested through the Attention Stack.

Schedular

Several devices are shared by the application programs; disk, tape, UDC8 console printer, etc. Requests for these devices must be stacked if a requested device is busy with an earlier request

Requests. Each request for a shared device is made to a routine called the Schedular, and each request has space reserved for the tag to another request, should it be needed. The specific format is:

```
TAD DEVICEN
JMS SCHEDS
 0        }  Tag area and
 0        }  Request busy flag
(Request)
```

This tag area is also the individual request "busy" flag. A zero value indicates the "not busy" state.

Device requests are stacked by putting the address of the requesting routine in the busy flag area of the previously scheduled request. Thus, the request list has the format of a forward-linked list.

For example, if three requests are pending on a particular device, they would look like this:

```
      JMS SCHEDS
      R2
      10
  R1, (Request)
       .
       .
      Field 1
      CIF 00
      JMS SCHEDS
      R3
      1 0
 R2, (Request)
       .
       .
      CIF 00
      JMS SCHEDS
      2
      00
 R3, (Request)
```

Device Control Blocks. Each shared handler has a Device Control Block (DCB), which is a set of parameters that are kept for the scheduling function. Each DCB table is linked to the other DCB with the same format employed in device requests. Position in this list is optimized with respect to the number of times requests will be scheduled. The trunk list for scheduling is this DCB list. Because the GETNXT subroutine is used for traversing this list, it is delimited by the device--not by value. Besides its own linking elements, each DCB has a list header for that device's request list.

Besides the linking element, each DCB is identified by a device number which is item 1 on a DCB table. Item 2 is the list header for that device's request list. It is also used for the device busy value. Items 2, 3, and 4 are for handler initiation. Item 4 is initially a copy of the address of item 2 and can be manipulated by the handler or the request. An example of a DCB list format is given below.

DEVICE CONTROL BLOCK

(item)	(code)	(comments)
0	(Next DCB address)	
	(Next DCB field)	
1	(N)	/ DCB number
2	2	/ Current request address
	CIF CDF 00 or CDF 00	/ Current request field
3	JMP I .+1 or SKP	
4	2 2	/ Handler address
	(Handler)	

Linked-List Format. The particular linked-list format used in the request lists for each device and in the device list itself, uses two locations immediately preceding the item in order to point to the next item. The first word of these two next pointers is the 12-bit address of the next item on the list. Bits six, seven, and eight of the second location indicate the field of the next item. The last item on a list has a special end-of-list value in its next pointers.

There are several lists which the Schedular does not use, such as the background clock job expeditor list or the background display loop list.

Traversing a stream is done with a subroutine called GETNXT which moves its pointer registers from one mode on the stream to the next. It also gives a differential exit when the mode is the end of the list. That end of list value is the device-not-busy value.

Schedular Operation. The Schedule subroutine is called with the device number for each request being scheduled. If the request busy flag is set, scheduling ends. If not busy, the flag is set to the device-not-busy value, thus simplifying the Eject function. The selected DCB is located and its busy flag is tested. If busy, this location and the next will point to the first and current item in this device's request list. To add this request to the list, the end of the list is located and its next request pointers are altered from the device-not-busy to the 15-bit address of the request to be scheduled.

If the device is not currently busy, this request can be immediately initiated. The DCB's current request pointers are set to this request, and the handler is initiated by passing control to the field word of item 2.

The return to the Attention Stack Delete routine is accomplished by a RESTORE instruction. The Schedular is called using the JMS instruction with the selected device number in the AC. The routine's entry point is in field 0 at location 177_8, so indirect addressing is not necessary for the JMS call. The Schedule subroutine ends with an eventual RESTORE, and the JMS gives the Schedular the 12-bit address of the request. The data field register indicates the three field bits of the request.

Handler and Eject. A handler can treat a request as parameters to be used for transfer of information or the handler will pass control directly to the request for execution. Item 3 of a DCB list indicates this.

When the request has been completed, the device handler must be passed to the next request or be set to the not-busy value. The Eject function accomplishes this. This function is called either from the handler or the request at any API level, and it cues its own routine onto the Attention Stack. The Eject function is done at the background to synchronize with requests and to share routines and code with the Schedule function.

The Eject function is called with the number of the device to be ejected. The DCB of that device is located and the address of the next request in line is obtained from the "next request" pointers of the current request. The "next request" pointers of the current request (which functions as the busy flag) are cleared upon completion. Then, the code of the Schedule function which initiates the handler is used to reset the DCB's current request pointers and to reinitiate the handler. If the current request is now the end-of-list marker—i.e., the handler-not-busy value—the handler is not initiated before the ending RESTORE instruction.

The values for the device handler-not-busy indicator are: address = 2, field =0 . If an interrupt sends a handler to that location, the RESTORE instruction that is permanently stored there will essentially ignore this interrupt.

DCB List Devices. The devices on the DCB list are grouped into three areas; the first group consists of the console and auxillary (DPI2-A) teletypes, the peripheral I/O expander (BA12) with four ASCII ports (DC02-E), and the high-speed paper tape reader and punch. These have enable addresses and keyboard handlers for input, and DCBs and printer handlers for output. The enable values are associated with the DCBs, and they are the two locations immediately preceding the linking locations. The enable address is the DCB number location minus three and the field bits minus four.

The second group consists of the mass storage devices, disk and LINCtape units.

The third group is the Utility Area which functions as a memory partition device.

Device Handlers

Mass Storage Handlers. Since the LINCtape and disk functions have a similar blocked transfer format, request formats are the same for these two devices. The LINCtape unit has a device assignment number 20_8 and the disk, number 30_8. The general request format is listed below.

MASS STORAGE HANDLER

```
       (ITEM)                      (CODE)

                                TAD DEVICEN
                                JMS SCHEDS
         0                      0
                                0
         1                      0 or 4000
         2                      PARMP
         3                      (Notify here)


         4   PARMP,             (Unit)
         5                      (Device block)
         6                      (Memory Page)
         7                      (Number of Pages)
```

Item 0 is the tag area. Item 1 is the read or write operation indicator. Bit zero of this word is cleared to read or set to write. The remaining bits of this word must be cleared. Item 2 is a pointer to a table that indicates how much information is to be transfered to/from where in memory from/to where on the device. This table must be in the same field as the request.

The disk is divided by DIAL-MS (DEC program development software) into seven logical units, and that same convention is employed by the O.S. disk handler. So, item 4 for the disk can range from zero to six, for the LINC tape, from zero to seven. Item 5 indicates which program or data block of the selected unit is to be used in the transfer, and item 6 indicates the memory locations to be used. Transfers must occur on page boundaries, with the page number indicating the first page to be used. For example, page 41_8 is the second page of the second field. Item 7 indicates the quantity of pages to be transfered, and this must be greater than zero. The LINCtape is capable of transfering only two-page blocks, so item 7 must always be an even number. But the disk option of transfering half blocks (i.e., single pages) is implemented by the disk handler, so item 7, number of pages, can be any integer number for disk transfers.

The handler first takes the read or write bit from the contents of the request address. Next, the transfer parameter pointer is used to obtain the necessary information, and, after the blocks have been read or written and checked, the normal exit is taken.

Control is returned to item 3 through the Attention Stack from the handler when the transfer is complete or the handler has aborted in an attempt to transfer. The AC is cleared if successful or, if not, a 200_8 character is returned from the handler. This request will have already been ejected, causing the busy flag to be cleared by the time the notify routine, item 3, takes control.

Each handler does error checking and reattempts the transfer if possible. The LINCtape handler uses two tests included in the PIP (Peripheral Interchange Program). Each read or write command issued is accompanied by a maintenance register test to check if the tape is in motion. This test is positive when the unit is not thumbwheel (operator) selected, when the remote select switch is not set to "on," or when the write lock is on during an attempted write. A background clock job allows the transfer forty seconds for completion. If it does not complete in that time, the tape unit error exit is called.

The disk controller gives extensive error information plus an interrupt on transfer problems. Some types of errors may be recovered by reinitiating the call, while others are unrecoverable. A recalibration procedure is used after some errors to realign the disk, by moving the head back to the zero position on the cylinder. Retrys are done twice before the attempt status flag is cleared and the disk error indicator is returned to the O.S.

Printer Handlers. Device independence is provided by a standard interface between the various TTY output handlers and the application routines that request a printer. Device sharing is accomplished by queuing up those application routines. The standard interface also allows the device to be selected at run time, with run-time linkage between the handler and the request.

If the application program is using a shared printer, it must schedule its printing co-routine on that device. These printing request lists are the "execution" type, where each request is a self-contained executable code. This eliminates the need for software to provide for a Wait-For-Buffer Empty function. Each co-routine does a subroutine call to the device handler, the address of which is available when the co-routine is initiated.

The handler subroutine will type the character in the AC and return on the character complete interrupt. As these routines execute at the API level of the selected printer, each application program provides its own text and number-printing routines. When the co-routine is finished typing, it calls the Eject function and notifies other waiting co-routines. If the device is dedicated, that is, not to be shared by cooperating routines, then the handler subroutine is similarly called but without the preceding Schedule and final Eject functions.

A console device character-printing buffer and buffer delete code are provided. These allow short messages to be printed without waiting for completion before continuing with other input or processing which might also generate another short message.

Application programs must buffer characters, using the insert pointer and character counter provided, and call the Delete routine, which does its own scheduling, printing and ejecting. The Delete routine is called for each character that is buffered because the Schedule function tests the Delete routine busy flag.

The ring buffer is 32 words long, and the characters to be printed are stored in stripped 6-bit ASCII, one per half word. The Carriage Return and Line Feed functions are combined as the 36_8 character, which is the "up arrow" key. Since these characters are buffered at the background API level, the buffering code can be shared.

Clock Handler. Since the clock has so many programmable options, it has two lists or executable streams associated with it; one at the API clock interrupt level, and the other at the background level.

The clock interrupt level may have several tasks associated with it that can be added and deleted as required.

The clock task that is initially loaded notifies a background routine every 1/10 of a second. This background routine is called the Clock Job Expediter (CJE). Other CATTS application programs may request different timing requirements. In order to share the real-time clock, a software incremental timer list is implemented. The items on the list have a counter as the first word, with executable code following.

The Clock Job Expediter tests the first word. If it is zero, no action is taken and the next request on the list is tested. If the counter is nonzero, no further action is taken. If it is zero, control is passed to the code after the counter in a manner which allows control to return to the CJE at the end of that request. Each counter is tested every 1/10 of a second. With this, pauses of varying lengths, or routines which run at a constant frequency, or time-out timers are implemented.

Power Fail/Restart Handler. The power fail/auto restart option allows the saving of "volatile registers" when the AC power for the PDP-12 system fails. The API automatically saves the main processor register on the stack so that the only other register saved is the stack pointer. On restart, the machine is reinitialized, and processing is resumed. Some locations in the Restart routine are reserved for re-enabling the interrupt of other specific devices.

Utility Area and Console Command Module.

A memory Utility Area provides console keyboard interactive capabilities for program loading, setup, and control.

Since this memory capability is not required all the time, this utility section of memory is shared similar to a device; for scheduling of requests, execution, and final eject of various utility routines.

The request is a one-word parameter, indicating which program storage block from the system unit is required and how many blocks are needed. The Utility Area device handler reads the blocks from the disk and passes control for execution. If the read is not successful, this request is rejected.

The LINCtape on unit zero is a copy of the first disk logical unit with identical programs located in the same storage blocks. Some parameters of the programs are slightly different in order to distinguish between the two different storage units. For example, the program command that bootstraps the DIAL-MS system on tape uses LINCtape commands to load it from tape. The DIAL-MS system on the tape is an earlier version that can run without the disk device.

There are currently two users of this Utility Area device. One is the Command Module and the other is a utility program, described in the CATTS application program section.

Programs that use this Utility Area usually do not occupy it for more than a minute. These programs usually run to completion or, as in the case of interactive routines, are ended by the operator when the requested keyboard information has been entered.

A Utility Area job may have a display or a clock job of its own to be connected. Rather than appending and deleting in the usual manner, the task's address is substituted for the first address in the appropriate device list after setting the task's next address pointer to the address of the task that was previously first. The task is disconnected from the beginning of the list by restoring the stream's first address pointer to its previous value, which is stored in the disconnected task's next address locations.

The Command Module is initiated by a special test of console input characters. If a line feed character is detected, the routine that schedules the Command Module on the Utility Area device is notified through the Attention Stack.

When the Command Module is loaded, some page 0variables are moved into place, and then an interactive prompt of ">" is printed at the left of the page. Next, a line of characters is entered, with the first seven characters being used to construct the command name.

Programs are added to the O.S. program library by saving the binary version on disk unit ten and backup LINCtape unit zero. The program name assigned for storage in the DIAL-MS index on these units has an asterisk character in the eighth position, with as many blanks as may be required after the name. That name, without the asterisk, then becomes a legal command.

When a command is selected, the second block of the associated binary file (the block immediately following the header block) is read into locations 0 2400 -0 2777 for execution.

Console Control Command Module-Interactive

Command Blocks. The Console Interactive routines, along with the command input line, are left intact in locations 02000 -0 2377 when the command block is read. Two routines-CDBA and CDBI-handle the console output by using the previously mentioned resident console printer buffer and delete code. CDBA buffers a text list of packed 6-bit ASCII characters delimited by a character, all six bits of which are zero. CDB1 buffers just one character. One routine, CDRD, reads the line of characters from the console keyboard, using the enable address of the console Device Control Block. This routine echoes each character with CDB1 and allows rubout key editing. The input line can contain both alphabetic and numeric data. Two routines, GETD and GETO, convert the numeric digits to either decimal or octal numbers of up to 24 bits in length. The input line which contains the command name may have additional parameters following it. Each parameter may also have its own interactive prompt, input, and conversion sequence. Besides these routines, an octal printer for diagnostic error messages is available. The page zero variables contain indirect addresses for these routines; selection of device (disk or tape), block number of the selected command and conversion variables.

Since commands occupy the Utility Area and should not remain resident too long, their functions are limited. These functions include loading and unloading programs from permanent memory space, doing keyboard interactive setups, or running short control or list functions that run to completion and do not require permanent memory space.

Memory. The resident portion of the operating system, which consists of device handlers, device-linked lists and code, and Attention Stack and code, occupies one-eighth of available memory space. The Utility Area occupies another eighth. The balance of memory space is used for programs that need to be core resident for relatively long periods of time, such as display-refreshing, data collection, or long interactive routines. Table 3 outlines the core memory allocation schemes for the [redacted] system. Routines loaded into

CATTS MEMORY ALLOCATION SCHEME

(FIELD)	LOCATION (OCTAR)	OCCUPANT
(0)	0200 - 1777	Resident O.S.
(0)	2000 - 3777	Utility Area
(0)	4020 - 4377	(Available)
(0)	4400 - 5577	CRT Display Routine
(0)	5600 - 5777	(Available)
(0)	6000 - 6777	(Available)
(0)	7000 - 7777	Data Transmission Routine (TMITAUX)
(1)	0200 - 2777	CATTS Application Routines
(1)	3000 - 3377	CATTS Application Routines w/Control Codes
(1)	3400 - 3577	Telephone Data Input Routine
(1)	3600 - 3777	(Available)
(1)	4000 - 4777	DITRMA Routine
(1)	5000 - 5777	CRT Display Routine
(1)	6020 - 6377	Utility Location Display Routine
(1)	6400 - 7377	(Available)
(1)	7400 - 7777	DATAMYTE Data Input Routine

Table 3. CATTS Memory Allocation Scheme

this permanent memory could be used by two programs simultaneously, but these shared routines must never be required at the same time. An example of this incompatability would be the use of video display routines, because both requesting programs would use the same CRT device and thus could not be scheduled to be used at the same time.

A bit map in the resident section of the O.S. specifies whether a section of memory is occupied or not. This bit map uses one word per 1K LINC segment, three bits per 256-word block, and two bits per page. Table 4 illustrates the bit map scheme for core memory allocation by the O.S. Subpage fragmentation is not used. A check of unique locations in the bit map indicates which program currently occupies a specific section of memory.

Page zero is directly addressable from all other pages. To make page zero locations available to more than one program, locations are allocated 32 per LINC segment, four per page, in direct correlation to the page number. LINC mode index registers are shared equally by all occupants of the memory segment. Most program uses of these registers, like displays or output buffering, are at the background level.

Loaders. Each program that occupies a permanent memory area has a uniquely coded loader as its command block. This style of program loading allows sections of the total CATTS program mix to be developed independently of other sections by prescribing independent interfaces to the other sections and the O.S. Some of these connections are made at assembly time by adding symbols that link to other routines or to the O.S. as direct assignments. Some of the routines are linked to the O.S. by the loader. Some shared device handlers like the disk and printer are linked to user routines at run time. The individual functions of the loader are written in macroform so that the loader is

MEMORY BIT MAP

CODE	INDICATED LOCATIONS
MEMMAP, 3333	0-0000 to 0-1777
3333	0-2000 to 0-3777
0000	0-4000 to 0-5777
0000	0-6000 to 0-7777
0000	1-0000 to 1-1777
0000	1-2000 to 1-3777
0000	1-4000 to 1-5777
0000	1-6000 to 1-7777

12-Bit Word

Pages	0	1	2	3	4	5	6	7
LINC Segment	1		2		3		4	

Table 4. Memory Bit Map Scheme

coded as a function of the resources used by each program.

The area in memory must be available, so the appropriate bits of the memory map table are checked before loading. The code to add a display or clock job task is the same as the append code of the Schedule subroutine. Since the stream-linking format is different for displays, the code to append a display is slightly different, but the steps are the same. If a new interrupt device handler is added, the interrupt vector address for this device is altered to pass control to this new routine when the new device causes an interrupt. The device may have to be enabled and re enabled on power fail/ restart, with some locations in the Restart routine reserved for that purpose. If this new device is sharable, its Device Control Block is appended to the DCB list. When a task is added to the programmable clock API level, special list rules are kept to keep the number of instructions to a minimum. Tasks can also be added to the Power Fail and Restart routines, but the standard stream format is not observed. Inputs to waiting programs are enabled by storing the waiting routine's field and address in the input enable locations of the selected input device. Some special functions, such as testing the availability of a device that will be used by the program, may be required. Also, before the last step, program parameters are set which must eject the Utility Area. Generally, any program function which is not required to be with the permanently resident code takes its space in the shared loader area.

To unload a program means simply to reverse the effects of loading. The proper program must be already loaded. This can be checked by testing its unique values in the Memory Bit Map. To make that area of memory available, appropriate bits of the Memory Map must be clear To disconnect a clock job or display task, that position is deleted e stream. This is accomplished by finding the task that points to it, and then changing that task pointers to the values in the next tasks pointers.

Device handlers are disconnected by restoring the interrupt vector values; DCB's can be deleted similarly to clock jobs and displays. The device interrupts must be subsequently ignored, either by disabling, or clearing and restoring machine status. Any power fail/restart steps must be deleted also. Inputs are disabled as well, and, since there is a slight chance that a character could already be on the Attention Stack, putting a RESTORE instruction at the beginning of the waiting routine is sometimes done. Also, shared devices that were set busy to dedicate them to the program to be unloaded, must be set to the not-busy value. Particularly, printer subroutines must have their return field and address set to the not-busy values.

Displays. The displays are program refreshed. Computer time for this function is allocated as a part of the bottom background level of the API. The other part of this loop is the Attention Stack delete task. Tasks can be added and deleted from this loop, but the format for linking these tasks is slightly different. The field of the next task is the starting address minus three, and the address of the next task is the starting address minus one. The instruction "JMP I .+1" is assembled in the starting address minus two so that each task can pass control to the next by jumping through the next task pointer which looks like this:

```
                CIF CDF (next field)
                JMP I .+1
                (next address)
(this address), (display something)
```

There are several types of displays. The CRT hardware will display a dot or a two-by-six half-character pattern. These are used to display a character buffer similar to a terminal CRT "printer," a real-time monitoring display like the location displayer, or various CATTS feedback graphs.

There can be only one or two displays used simultaneously since the scope has two channels and refresh time is limited. Each display uses special character display pattern tables or a subset of the full 64-character set display pattern found in the PDP-12 system reference manual.

Program Development

This data collection O.S. and the CATTS application programs were developed using the DIAL-MS operating system supplied with the PDP-12 computer hardware. For command block loading, the first assigned location in a program or loader block located in FIELD 0 is $*2400_8$. Locations below are used by the CATTS resident O.S. Field zero, page zero locations are available but must be assigned symbolic names using the direct assignment statement (TAG=100) rather than the first symbol of a statement followed by a comma at the page zero location (*100; TAG, 0). Any page zero constants or indirect addresses must be deliberately moved to their assigned locations.

Rather than use the SAVSYM, LODSYM pseudo-operations of the DIAL-MS assembler, symbols from the operating system and other programs must be obtained from their assembly listings and added to a program as direct assignments.

For overlays, the assembler has a FIELD pseudo-operator which allows the programmer to select memory stacks. This can be used to assemble two routines in the same absolute 12-bit address. Routines can be assembled on page boundaries in other areas of memory to be overlayed into the same area, but special attention must be given to cross-page references with indirect addresses. These require the appropriate arithmetic adjustment.

The PDP-12 has a complete programmer console with extensive debugging features. To augment these features, a location displayer is added with the resident O.S. This routine displays a group of location addresses and contents from either memory field and can be used to alter the contents of a selected memory location.

The location displayer uses the programmer console left switches to select an address in one of the 4K stacks of memory. The right switches are used to specify a value that examines or fills a location. Sense switch three enables the display. Sense switch four adds the location selected in the left switches to a table of up to sixteen locations, if that location is not already present in the table. Sense switch five deletes that location specified by the left switches from the table. Sense switch two selects field zero or one for the location table. To fill the location selected with the left switches and sense switch two with the value of the right switches, both the fourth and fifth sense switches are set.

The display is located in the upper right-hand corner of channel one. First displayed is the location selected in the left switches, and then the contents are displayed as four octal digits each. Any locations inserted into the location table are displayed in columns underneath.

To save all of memory, a tape dump procedure uses the programmer console "DO" switch memory control LIF, JMP, and LDF instructions; and the LINCtape group write (WCG) instruction. The O.S. utility "TDUMP" command then dump the contents of the selected blocks of tape into the LA 30 printer. In debugging, several internal locations assist in indicating what code is being processed [illegible]

CATTS Application Programs

CATTS application programs are grouped according to function. The data collection group accepts data and constructs a standard data file. The feedback group uses a standard interface to the data collection file. The DITRMA group combines some data collection and feedback functions. The data-link group uses the standard data file to help produce hardcopy feedback. The operator control group selects programs and parameters.

Each program is further subdivided into routines. These routines and the programs themselves communicate through common data areas, such as the data collection intermediate file buffer block, display buffers, the Box Table Array, and other special-purpose mailbox locations.

CATTS Data Collection

Data is collected from a number of different sources each with its own special features. The final product of a data collection session from one of these sources is a standard data file on mass [illegible] containing the data coded during that observation session, together with associated interval data. To simplify the daily scheduling of the data collection sessions, [illegible] box is independent of the others with respect to observation coding systems being used and the time that data collection is initiated. Furthermore, these different sources, henceforth referred to as boxes, are able to use different coding systems simultaneously.

Data Structures. Each box has its own set of variables. These sets of variables form a two-dimensional array; box number by variable number. The code is therefore re-entrant on each character input and shared by all the boxes.

The box variable set follows:

a) STATUS

b) INTERVAL TIMER

c) END-OF-SESSION TIMER

d) END-OF-SESSION RESET VALUE

e) DATA DIGIT BUFFER (LOW)

f) DATA DIGIT BUFFER (HIGH)

g) MAXIMA

h) SORTP

i) SORTB

Each observation data code [illegible] consists of up [illegible] decimal digits. The series is stored in the [illegible] DIGIT BUFFER with the count of the number of digits as the least significant [illegible] of the STATUS word. [illegible] editing ("SKIP or CLEAR") key on the coding box allows the [illegible] to clear and then reenter the entire series. The "ENTER" [illegible] delimits [illegible] series.

The digit series is then "packed" according to the MAXIMA (maximum digit entry for each observation system) into one 12-bit word. The 12-bit data word has three 4-bit fields. Observation systems can have up to three levels each level being a specific or subdivided area of observation interest. Each level can have up to 16 categories describing each area. Each level can, therefore, use one or two digits to fill its corresponding field in the 12-bit word. The MAXIMA category for each level specifies whether one or two digits will be used. If the MAXIMA category is 10 or greater, a leading zero must be used in all but the final level. There are three tests which verify each code. If any level exceeds its MAXIMA, or there were no digits or too many digits entered, this code is ignored and a message indicating this condition is printed on the operator's console.

The interval time since the last ENTER key is recorded to the data and reset to zero. The data set, consisting of box number, time, and packed data code, is stored in a memory buffer. When this buffer is full, it is written to an intermediate rolling mass storage scratch-pad file, blocks 1-100_8, on storage unit [illegible]. While the data buffer is being written, any new data sets are stored in an overflow buffer. When the buffer write is complete, this overflow buffer is moved to the beginning of the memory data buffer.

The data from each box is mixed with the data sets from other boxes that are simultaneously collecting at the same time. The beginning of each box's session is marked in this intermediate buffer by a special reserved data code.

If the computer is present to notify the coder of the end of the session and to automatically end the session, the END-OF-SESSION TIMER is set to the END-OF-SESSION RESET VALUE before the observation session begins. While the box is collecting data, this END-OF-SESSION TIMER is incremented once each second. When it reaches zero, the box's session is automatically ended. The coder also has the option of ending the session earlier with the button box.

The same reserved code used to mark the beginning of the session is used again to mark the end of the session within the intermediate data storage buffer. The location in the intermediate file buffer of this end mark is stored in SORTP and the intermediate file block number in SORTB. A Sort routine is then scheduled on the Utility Area device. This routine scans the Box Table, or box variable list, looking for nonzero SORTP's. When one is found, the intermediate file is scanned from that point backwards for that box's corresponding data sets. Total time and number of data points are accumulated on this pass. When the beginning of session mark code is encountered, the direction is changed. On the forward pass, the data sets are used to construct the final individual data file which is stored in the next available block(s) on file storage unit 6. The block number that an individual box's data file starts on is used to identify that file and is called the Starting Block Number (SBN). When a data storage unit is full, it is copied to LINCtape and reinitialized to the beginning of the file. Thus, the tape number and the SBN will uniquely identify a file and point to its location.

There is an elaborate backup system in case the disk should fail during data collection. The intermediate and final file storage units are automatically switched to LINCtape. Any boxes that were collecting at the time of the failure are sorted up to the failure block only.

Then, when the disk is repaired, the rest of the sort can be completed and the two partially sorted files can be combined to make one complete file.

Routines. The computer interface for the button boxes and the TOUCH-TONE telephone is the UDC8 I/O subsystem. The input functions for this unit reside in two UDC8 modules. The contact interrupt module accepts one bit per box to indicate when and where the data is present and the contact sense module senses four bits per box to read the data. A device handler responds to the interrupt caused by any change of state on the contact interrupt module. This change-of-state bit indicates which contact sense card is to be read and which 4-bit area of the word contains the incoming data. This data is shifted to the lower four bits of an 8-bit character. Each box has an enable address similar to TTY input, and the box number is combined with the field bits of the waiting routine to form one word. The second word is the address of the waiting routine. These enable addresses form the UDC8 Enable Table. The box number and field of the waiting background input routine are added to the 4-bit character.

This word and the address word of the waiting routine, if any, are queued to the Attention Stack, and then another bit test is made to detect any additional incoming characters. If none are detected, the machine is restored to its interrupted state and processing is resumed.

The functional unit of the UDC8 for output control to the button boxes and telephones is an output driver card, primarily an on/off series of switches, three bits per box. These output bits drive three lights on each box or answer back tones in the 403 Data Sets for the TOUCH-TONE telephone input. Each card also has its current status stored in memory. When a light is to change values on a box, the appropriate bits are changed in the appropriate STATUS word, and that word is reloaded into the driver card buffer. Routines are provided to turn all the lights on or off, or the middle light on for each individual box. One other routine, which works in conjunction with the Digit-Buffering routine, turns a light out when a coding level, one or two digits, is entered.

The background clock job runs once a second (1 H_z) and has three functions. The first clock function is the INTERVAL and END-OF-SESSION TIMER incrementing for each box. One bit of the STATUS word for each box is an up/down indicator. If the box is up, the INTERVAL value and the END-OF-SESSION value are incremented. If down, neither of these values are changed.

The second function is to sample the box switches. Each box has a switch that is connected to an external sense line of the LINC portion of the processor. When a switch position is different from the value found the previous time, the 4-bit value indicating its present position is combined with the values in the UDC8 Enable Table for that box, and they are queued to the Attention Stack as an input character. The previous value of that switch is updated and the rest of the switches are tested.

The third function keeps the current time in minutes plus a text character string of the current hour and minute, updated every minute.

When the input characters come to the waiting CATTS input routine, the box and data are separated by a subroutine. This routine also translates the data into a 7-bit ASCII character. The digits are translated to the ASCII digits (060- 071), the CLEAR key to an asterisk character (052), and the ENTER key to the carriage return character (015). In addition, the characters from the Clock Job routine which monitors the switches on the boxes are translated to a comma (054) for an "on" character or a space (052) for the "off" function. There is also a subroutine which gives differential exits if the character is a CLEAR key, an ENTER key, or a digit.

Printer output uses the console printer buffer and delete code provided in the resident system area. A single character print routine buffers the character in the AC, using the input pointer and character count in the resident system area. A text buffering routine, which is called with the address of the text string in the routine call plus one, uses the single character print routine. A zero character delimits the text string. All messages printed are abbreviated to lessen the possibility of print buffer overlap. A routine that prints the octal value in the AC and one that prints the box number are also available.

The Box Table Array has two subroutines to GET and PUT values. One subscript, the box number, is used to index to that box's set of variables. The other is located in the next location of the call to the GET or PUT function subroutines. The values are passed to or from the Array Table through the AC.

<u>Input Routines</u>. Each of the three possible sources of data button box, TOUCH-TONE Telephone, and DATAMYTE has its own input routine. All button

boxes input their characters to the same routine, and all TOUCH-TONE Telephone input is handled by one input routine. These input routines implement the various special requirements of the input device. The Button Box Input routine has the common functions for each character organized at the end of processing of that character. Thus, this code can be shared by other input routines by using multiple entry points into the Button Box Input routine.

A different coding procedure, implementing control codes, is also implemented as a separate input routine for button box data.

Some functions that are common to more than one input routine can not be placed after the unique functions of the Button Box Input routine and, thus, become subroutines.

The three sources of data employ the same button pad configuration for data entry, and they use 10 numerical digits, together with a CLEAR and ENTER key. The beginning of a session is marked for each input source by turning on the switch of the button box, placing the call with the TOUCH-TONE Telephone, or turning on the power of the DATAMYTE clipboard button pad. Switching off or hanging up the power or telephone is used to end the collection session.

Button Box Input. To begin a session, the coder switches the box "on." This is not a power switch, but a status line monitored by the computer. This switch, plus a special code, control two bits of each box's STATUS word in the Box Table Array. One bit is the ON/OFF status of the box and the other is the UP/DOWN value. The ON/OFF bit is set at the beginning of the session and cleared at the end. The UP/DOWN bit implements a "Pause" function when the automatic END-OF-SESSION TIMER and the INTERVAL since the previous code do not increment as they do in real-time. Both bits are set when the switch is initially turned on. Thereafter, the UP/DOWN bit follows the status of the switch. Data is collected when the switch is on. With the switch off and the UP bit cleared, data is saved only to be tested for the special end-of-session code of three consecutive nines; 9-9-9-ENTER. If the "on" bit has been set earlier, this session's data is then saved as a file on permanent storage.

All three status box lights come on when the switch is turned on. As the digits are entered, each light will be turned off in sequence from left to right, in order to indicate when the digits for any one coding level have been received. All lights are turned off with the ENTER key. If a pack test on the data is positive, all lights are turned on. Otherwise, they are turned back on when the ENTER key is released. These light signals indicate to the coder incorrect codes or computer malfunctions. The middle light only is turned on when the switch is off for the Pause function, and all lights are turned off at the end-of-session code. Figure 17 diagrams the Button Box Input routine.

Button Box Input: Control Code Version. Various button box features have been added for more flexibility and different kinds of data collection information.

A special set of codes was reserved for control functions. This set of codes consists of the codes, 0000 - 0099. The first two digits are zeros and the last two are the same digit.

Also, the ability to enter identification data is provided. Instead of packing the digits into three 4-bit fields, a simple decimal-to-binary conversion is done. This feature is tied to the UP/DOWN bit of the STATUS word. It can also be used during the session to code information not included in the category system being used. When the button box is first turned on, the ON/OFF bit is set, but the UP/DOWN bit remains clear, and only the middle light

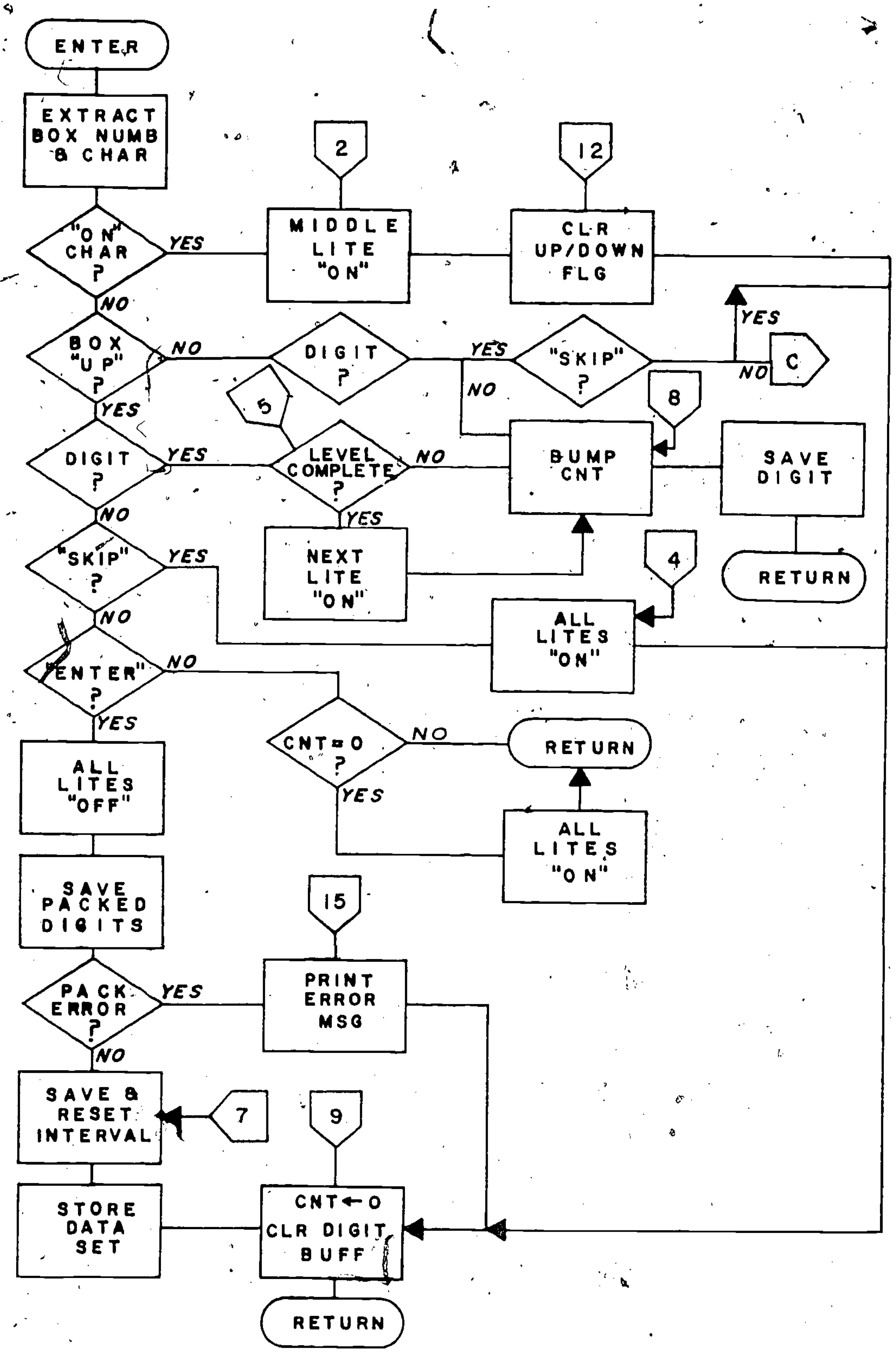


Figure 17a. Button Box Input routine-Part 1.

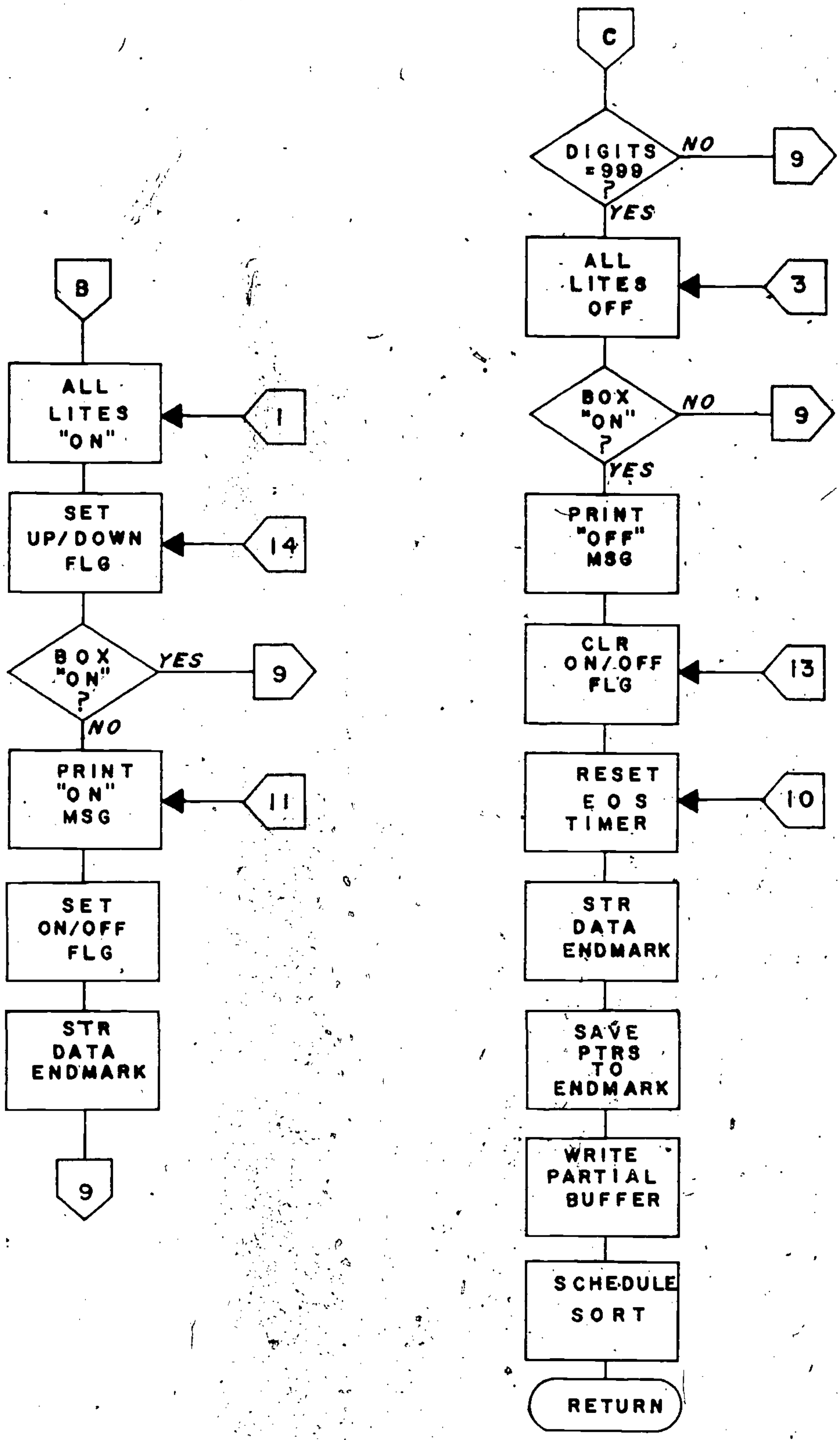


Figure 17b. Button Box Input routine-Part 2.

is turned on. Data is collected [illegible] this time, but the O.S. uses the Decimal Conversion routine rather than the normal pack routine. The coder uses the special control code zero-zero-one-one-ENTER (0011) to indicate the beginning of category data. When all identification information has been entered, all the lights are turned on, the UP bit is set and the control code mark is recorded in the data. The data collection [illegible] ion is identical to the Button Box Input routine previously descri

The control code zero-zero-eight-eig [illegible] (0088) switches the UP/DOWN bit back off (only the middle light r [illegible] on) and records its control code mark. Again, data will not be packed [illegible] ual but will be converted to a 12-bit binary number. These control codes can be used alternately throughout a data collection session.

The End-Of-Data mark is coded as zero-zero-nine-nine-ENTER (0099), which performs the same operation as the zero-zero-eight-eight-ENTER (0088) control code, but records its own control code mark.

Switching the box off is the final operation a coder performs to end a session. The ON/OFF bit is cleared, all lights are turned off, and the data is stored as a permanent file. Figure 18 diagrams the CATTS button box input with control codes added.

One subroutine implements both the control codes and the decimal conversion. It provides three different exits on certain conditions after performing status tests and constructing a data word. The first exit is selected if the buffered characters for the selected box fit the control code criterion of four digits; the first two zeros, and the last two identifical numerical digits. Also, the storage control code mark for each control code is returned as data. The second exit is selected if the UP/DOWN bit set indicates a regular pack to be

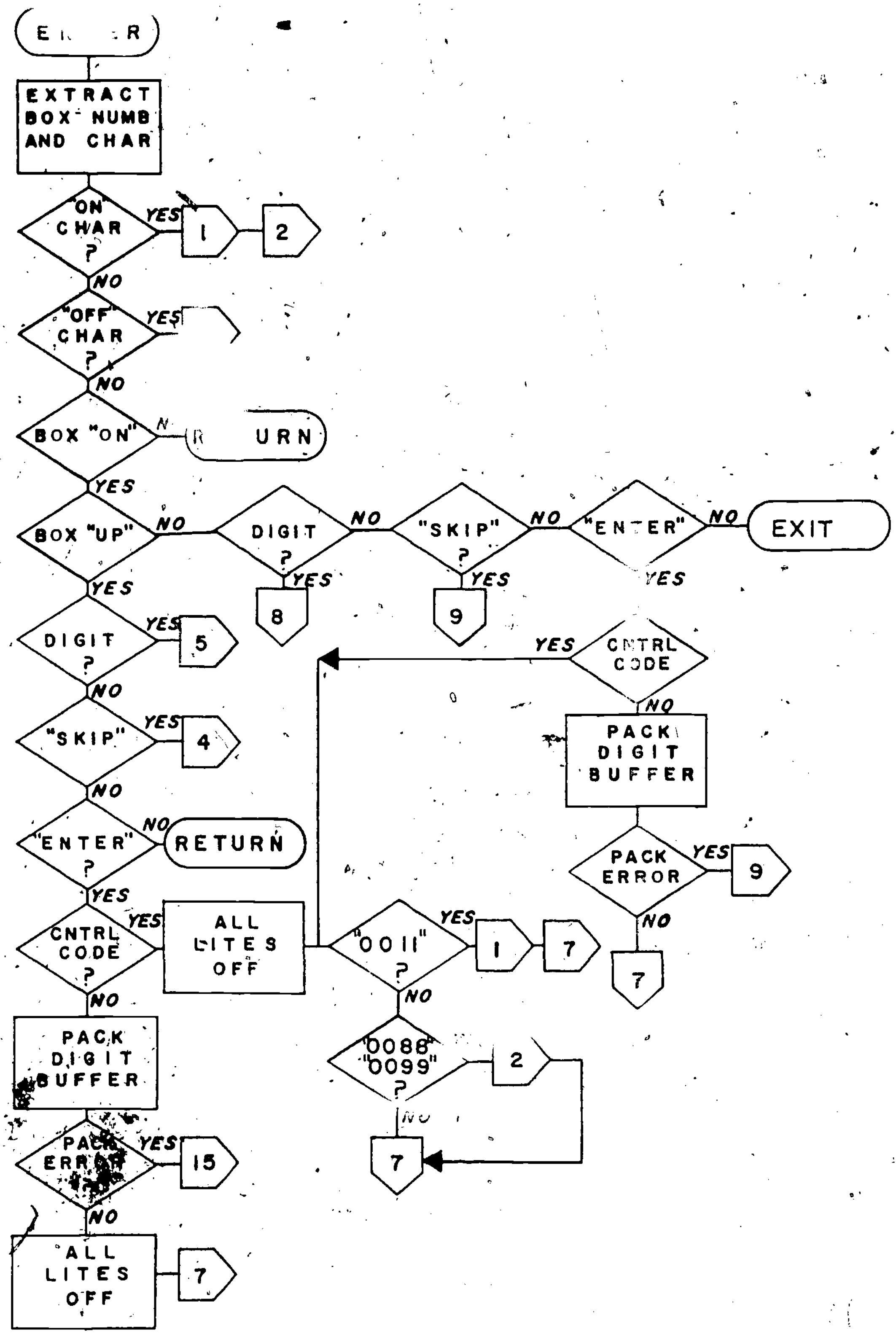


Figure 18. Button Box Input routine with control codes.

used, and a pack test is positive. The last exit is selected with the buffered digits already converted, either by the Pack subroutine if the UP/DOWN bit is set with no tests positive, or by a Decimal-To-Binary Conversion routine if the bit is clear. The End-Of-Data mark is not legal data rd, and is changed to zero.

The control codes are stored in the data file as regular data sets of box, time interval, and data code. The data code has levels one and two set to 15_8; bits zero through seven are all on. Level three contains the digit used in the control code. The reserved data code to mark the beginning and end of data has all three levels set to 15_8; bits zero through eleven are all ones.

Telephone Input. The Telephone Input routine has its own character conversion because the data set uses unique data bit patterns. The order of the input lines are changed to avoid duplication of bit patterns with the call and hang-up characters. The call and hang-up characters are provided from the external level monitor of the CATTS clock job. Character input is very similar to button box input with control codes. Lights are not used but instead, the light driver bits control the te phone answer-back tones. tones can be heard by the observer over the telephone handset whenever a control code is entered. A clock job will turn the tone off after one second. The End-Of-Session Reset Value (REEOS) of the Box Table Array is the tone "on" indicator for the telephone input ports. Figure 19 diagrams the TOUCH-TONE Telephone Input routine.

DATAMYTE

Input. The DATAMYTE is an off-line data collection device. Real-time

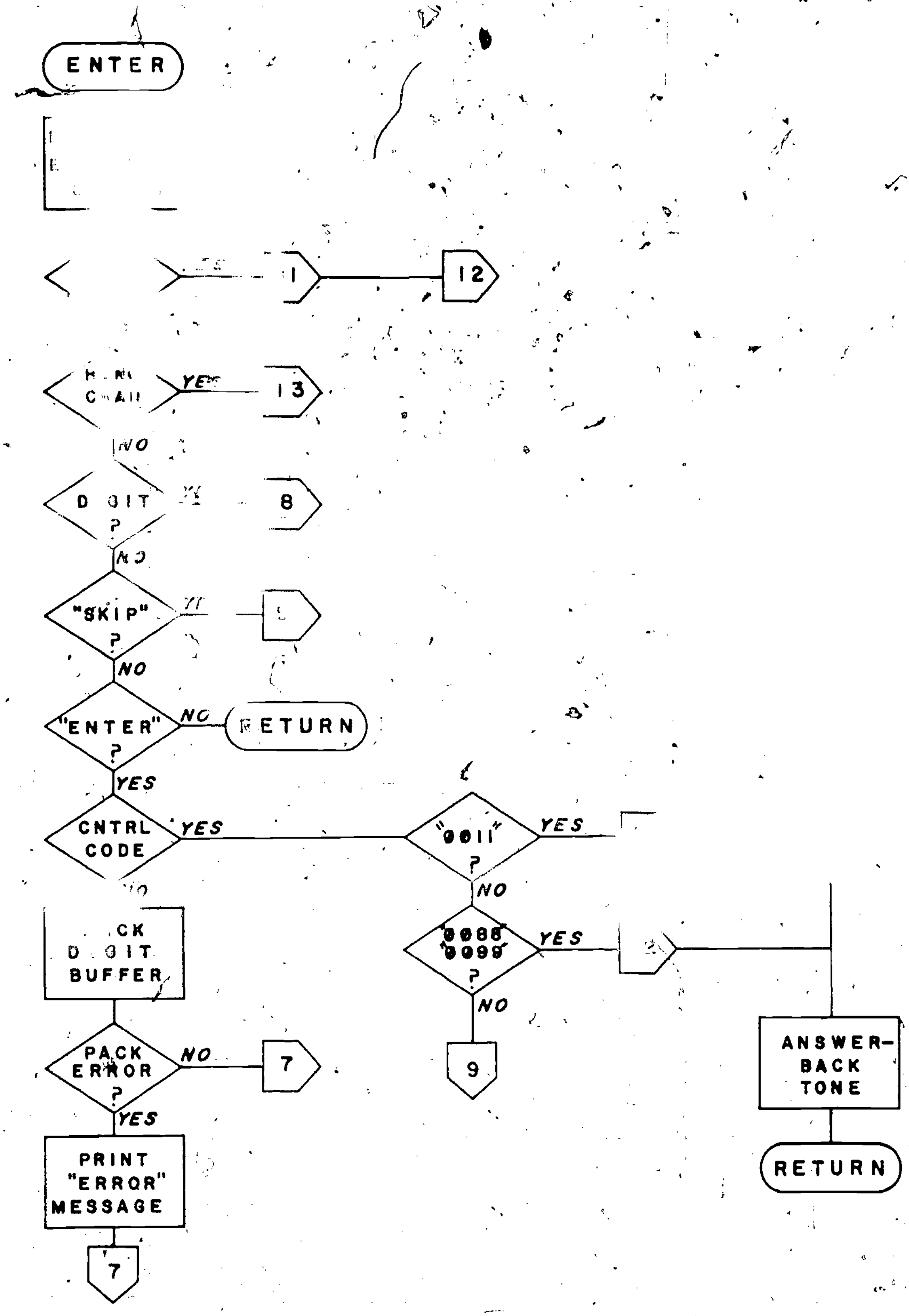


Figure 19. TOUCH-TONE Telephone Input routine.

[illegible] collection functions are provided by four digits recorded when the ENTER [illegible] is depressed. When the tape is played back, these digits are preceded by a comma and a space; thus, a flag switch is kept to indicate if a digit is a data code or a timer digit. The data codes are buffered in the Box Table Array, and the time digits are converted with the Decimal-To-Binary Conversion routine. This 12-bit timer value minus the last 12-bit timer value calculates the time interval since the entry of the last data code.

The On function of storing the reserved data code to [illegible] beginning [illegible] is performed when the program is initially loaded [illegible] stores a permanent data file, is performed when [illegible] control code is detected. The End-Of-Data res[illegible] end of a session serves as the beginning of data for the next [illegible] data.

DATAMYTE data is listed on a teletype as it is played back through the coupler. The Print routine buffers one character at a time into [illegible] four-character ring buffer and schedules the Delete routine. Any coding errors are printed next to the offending data code on the selected printer.

Two couplers are used to play back the data tapes. One is assigned box number 13, and the other, 14. The character received from the coupler is seven bits with odd parity and does not have the box number associated with it. Because the local variables and switches are not available in the Box Table Array, and because of the printer requirements, each coupler has its own input routine and printer handler. Figure 20 diagrams the DATAMYTE Input routine.

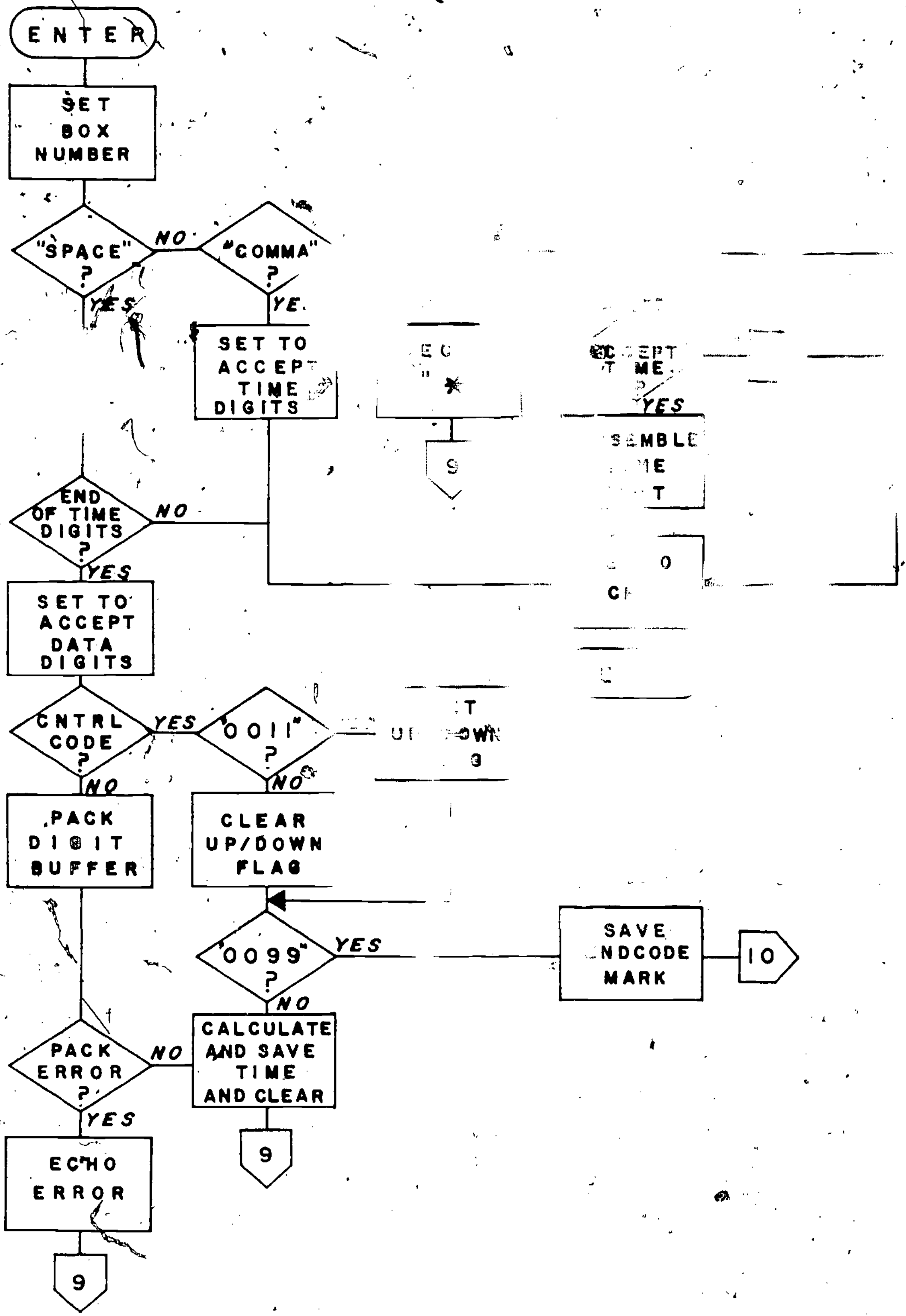


Figure 20. DATAMYTE Input routinε

Program Loader

Button [illegible] The "CATTS" command loads the [illegible]ton box data collection program [illegible] [illegible]r block for [illegible] program was [illegible]ten as a separate block so th[illegible] read and use [illegible] any data collection program, such as the loader for the [illegible]on box input [illegible] control code test. The commands to load these programs [illegible] the Memory [illegible]its, and if that area of memory is empty, the loader blo[illegible] is read and [illegible]oader routine is called.

To load the butt[illegible] box program [illegible]t the intermediate file storage unit and the fina[illegible] [illegible]le unit are [illegible] "read" and "write" ability. Next, the progr[illegible] If any [illegible] data tran[illegible] are unsuccessful, a message is printed [illegible] Utility A[illegible] device is e[illegible]ed The clock job is added to the CJE lis[illegible] [illegible]e device i[illegible] to indic[illegible] whether the current file storage units are dis[illegible] LINCtape. [illegible]-page se[illegible] of code and page zero variables are moved in[illegible] position. A device handl[illegible] [illegible]r UDC8 interrupts is connected to the interr[illegible] vector, enabled, and then the Power Fail/Restart routine is enabled. The Memory Map bits are set to busy when the loader has executed successfully.

Next, an interactive routine is called to set the current date and time. The lower five bits of the current date word are the day, the next four are the month, and the highest three are the current year minus seventy-five. The time word is set from a 24-hour clock and contains the number of minutes since midnight.

Once the button box data collection program is loaded, the Setup routine is initiated.

The Interactive Setup routine queries the operator for the parameters and box numbers to be used. The first parameters requested are the maximum category numbers for each level of the coding system to be used. This maximum packing parameter tests incoming data so that the reserved end data code is not mistaken for a legal code. The other requested parameter is the End-Of-Session (EOS) value. It is entered in minutes and converted to seconds. If the automatically timed session option is not selected by the operator, a zero value is entered.

When the numbers of the boxes that are to use these parameters are entered, they must be within the range of one through 12 inclusive. If the number is within the range, its Box Table Array values of MAXIMA and REEOS are set to previously entered values. The UDC8 input enable table addresses for each selected box are set to the address and field of the waiting input routine, and the box number is encoded with the field bits. Since other boxes may have different parameters, the request for MAXIMA is reiterated. If there is no additional input before the end-of-input character is entered, the setup is complete and the Utility Area is ejected.

Button Box Input with Control Code Loader. "CATTS2" is the command to load and setup the control code version of the button box input program. Like the earlier version, this loader block is separately read and executed if the Memory Map indicates its area is clear. The TOUCH-TONE telephone, DATAMYTE and DITRMA programs use this version.

The loader block first tests if the earlier CATTS button box version is loaded. If not, the CATTS2 loader block is read and executed. In addition, the new input routine and the Control Code Test subroutine are read and the Memory Map is updated to show that these routines are core resident.

The Interactive Setup routines for the two versions of button box input are precisely the same except for a different input routine enable address, and a prompt which identifies this as the control code version. Also, the MAXIMA Prompt routine is not reiterated after the box numbers are entered.

TOUCH-TONE Telephone Loader. The "PHONE" command loads the Telephone Input routine. The loader tests if the control code version is core resident. If it is not, the PHONE loader block is read and executed. If the PHONE Input routine is not resident, that program is read and moved into place, the Clock Job routine is connected, and the Memory Map bit for the area of memory is set.

Since the telephone clock job assigns a special use of the REEOS variables of boxes 10, 11, and 12, these boxes are not used for button box data collection when the phone inputs are being used. So, the first function the setup performs is to clear these boxes enable addresses in the UDC8 enable table, and then the REEOS and EOS variables in the Box Table Array. The maximums are asked for and the box number entered must be 10, 11, or 12. If no digits are entered for the maximum variables, the setup is aborted and the Utility Area is ejected.

DATAMYTE Loader. Since there are two input programs for both DATAMYTE couplers, there are two commands, "DM1" and "DM2". The differences between them are assumed input devices, memory resident areas, and Memory Map bits, page zero variables, and box numbers. The loader first tests if the DATAMYTE routine is already loaded. If it is, the Setup routine is immediately initiated. If not, the code is read and moved into position, input is enabled, and the corresponding Memory Map bit is set.

The Interactive Setup asks for the maximums. A constant box number of 13 or 14 is assigned. The initial box on code, which marks the beginning of the data session, As stored.

Other functions are provided for the operator by entering certain letters in response to the maximum's prompt. The operator can insert a control code into the data file while the tape is transmitting data by entering a "C" followed by one digit of the required control code. Previous data can be ignored so that a tape can be restarted by entering an "S." The output device is selected by entering an "O" followed by a one or a two.

Instantaneous Feedback

One of the functions of the computer system is to feed the coded information back to the classroom for the teacher's use. These instantaneous visual feedback programs extract the data from the button boxes or TOUCH-TONE telephone inputs. Only selected parts of the coded data are returned to the classroom and used by the teacher. Therefore, some interpretation and data reduction is performed by display routines. This process, unique to each type of feedback, describes which coding events will be displayed back to the classroom, ignoring some categories, and keeping tallies of other categories or series of sequential categories called chains, and perhaps some ratios or other descriptive statics. These figures can be relayed to the teacher through a closed-circuit television picture of a CRT display, or through an audio response unit playing prerecorded messages. The display may be of simple frequencies of certain categories, a graph of a selected category moving across an alloted time-line of a session, or histograms of selected categories or category chains.

The category processor interfaces itself to the data collection portion by monitoring the intermediate file buffer block of the data collection routines. Figure 21 diagrams this monitoring process.

At fixed intervals, the insert pointer to the memory buffer of the intermediate scratch-pad file is compared to a local pointer. If the insert pointer

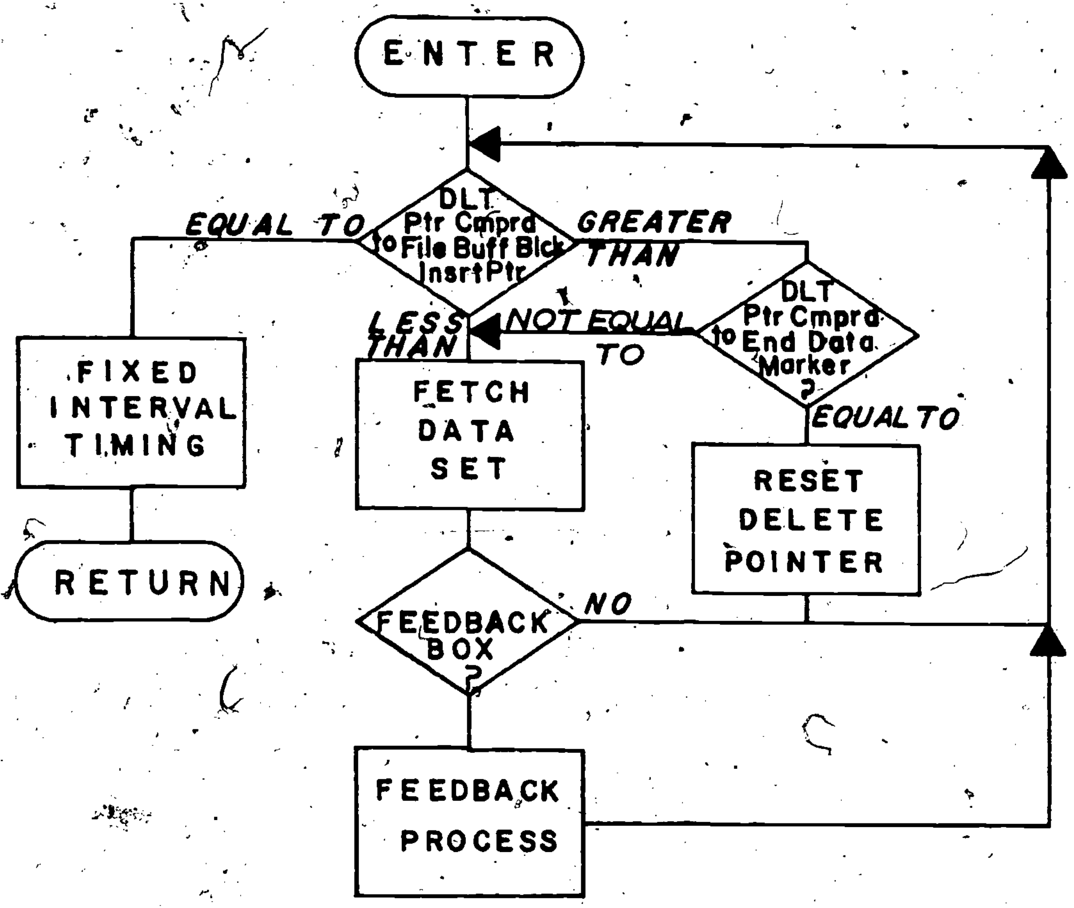


Figure 21. Instantaneous feedback interface process.

has moved, the stored data points are extracted from the data through the local file pointer and checked. If new data is from the target input box, that data code is used by the code processor. If the insert pointer has reset to the beginning of the block, the data sets are tested up to a special end-of-data pointer. Then, the local data pointer is reset appropriately, and other data sets are tested until the local pointer and the insert pointer are equal again.

Computer time for this monitor, located at the beginning of the code processor, is obtained either by a self-resetting clock job, or if the feedback medium is the CRT, a counter in the background display loop that checks every 10 times the scope is refreshed. The sampling rate depends on the response time desired for the feedback.

The interface between the code processor and the feedback medium depends on what requirements are selected. Generally, for a display the values are stored in mailbox arrays, and, for audio, the message numbers are stored in a ring buffer.

OROS Project Feedback. Figure 22 shows a picture of the CRT display used for the Oral Reading Observation System employed in a demonstration project at CITH. The bottom row indicates the category being displayed, and the height of the column above it indicates the number of times it was coded. The number "10" in the upper left indicates the count that signifies the top of the bar graph display. If a column frequency count exceeds the top, all column frequencies are cut in half, the unit increment is halved, and a new count of "20" would be displayed on top. The number "20" would replace the displayed "10." The bar graphs can display a maximum height of 80 entries.

During the OROS demonstration study, the bar increments were filled when a particular "correct" response code was detected following one of the column codes, as in the "44" column.

The code processor, as diagrammed in Figure 23, monitors the memory buffer block of the intermediate data collection scratch file. A start- or end-of-data code reinitializes the mailbox display array, setting the displayed frequencies at zero and the scale indicator at 10. If the code belongs in any of the columns, that column's cell in the common array is incremented by the appropriate increment value.

The display of the columns is done with small-size characters. A complete column is 20 character heights high and four half-characters wide. The top of any odd columns when the scale is 40 is done with half characters.

In the second display version, utilizing filled and empty columns, the process to display these half characters become more complex. Besides employing the usual whole or half characters to fill top or partially empty columns, there are two other special cases handled with special display codes. In one case, the transition from filled to empty space comes in the middle of a character height, and in the other,the same mid-character transition comes when the empty space is only that half-character tall.

The common Array Table is a series of locations, one per column, with a possible range of zero through 40. If the value is zero, no column is displayed. Otherwise, the value minus one is divided by two, and the resulting number of column side characters are displayed. If there is a remainder bit left over from the division by two, a full-character top is displayed. If not, the half-character top is displayed.

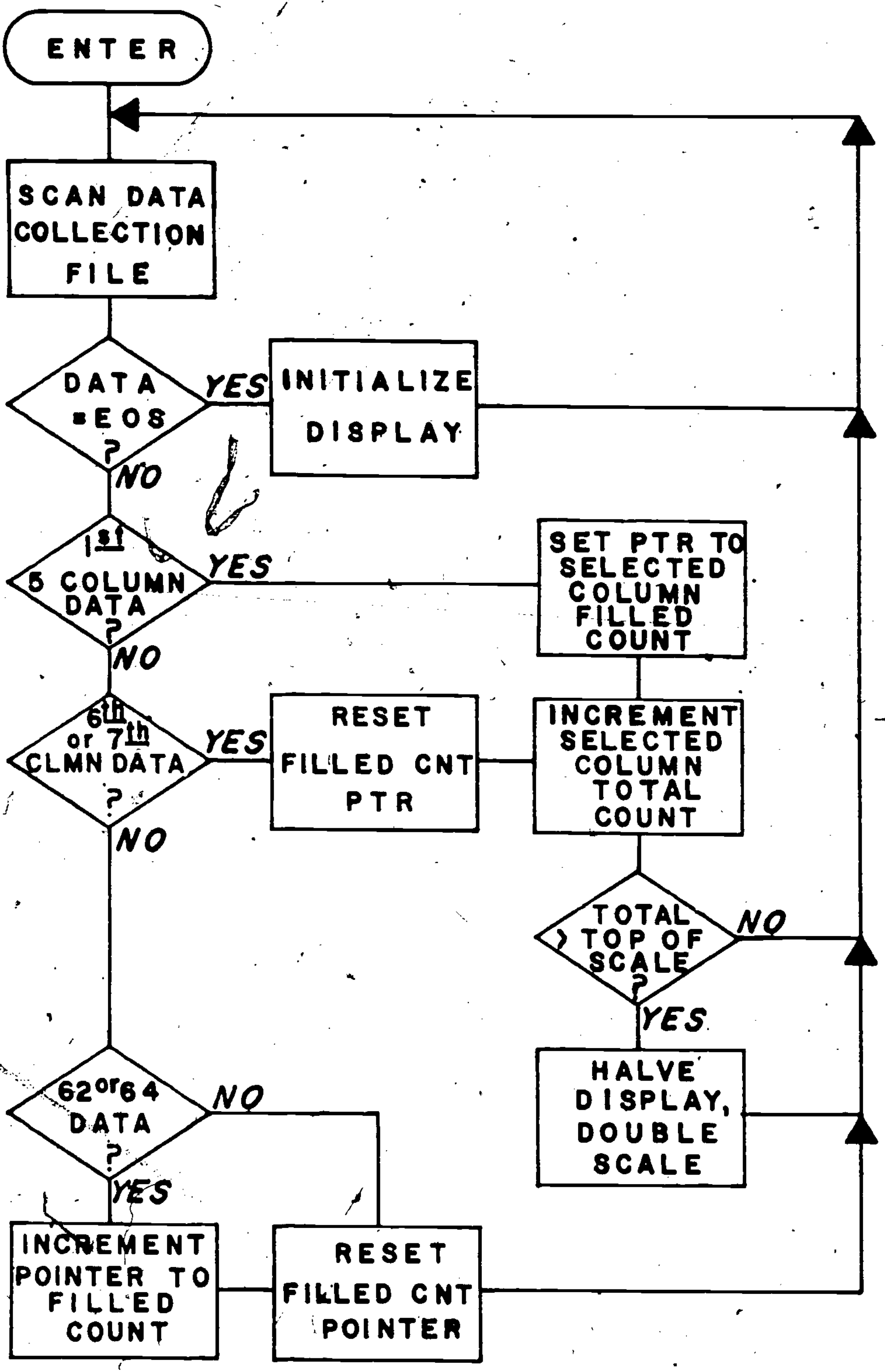


Figure 23. OROS visual feedback display decision process.

The Array Table is expanded to display two values per column in the later version. The first value is the sum total frequency of the target category times the current increment. A pointer remembers the most recently incremented column. If the second code of the selected two-code chain is entered, that column's second value is incremented by the current increment value. The second value is displayed as a column of characters filled solid with dots. The first total frequency value minus the second is then used to finish the column with the column side characters.

There are two independent displays, each with its own scope channel and input box. For each display, the columns are first refreshed. Then, the scale indicator is displayed. Finally, the bottom line of codes that are counted for each column are displayed under that column.

The code processor gets control every tenth time the display is refreshed.

<u>The TPQR Project Feedback</u>. The TPQR demonstration project at CITH used histograms that were part filled and part empty. A code processor, common mailbox array and column display similar to that described above for OROS were employed. Figure 24 diagrams the display process for TPQR feedback. In addition, a pre-programmed ratio was displayed as a percent indicator. Also, under the identifiers of the columns, a blinking arrow is moved left or right according to values interpreted by the code processor. The processor checks the intermediate file buffer block every 20 timesthe display is refreshed. Only one display and input box number were used for this project. Figure 25 shows a picture of the display used in the TPQR project.

<u>Loader and Setup</u>. First, the Memory Map is checked for residency. If resident, the setup is initiated. Otherwise, the load is done first, the program is read, the display is connected, and the residency bits are set in

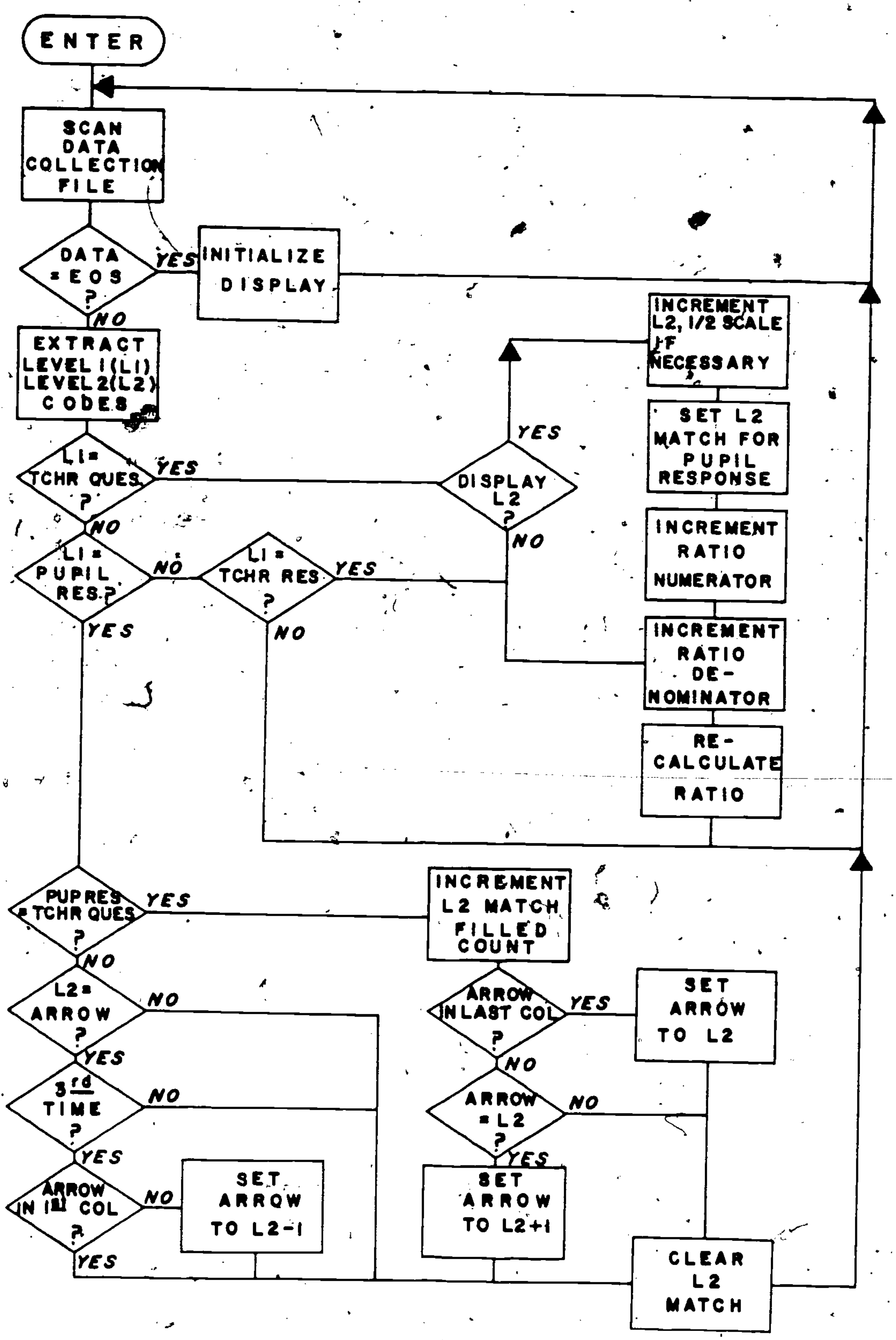


Figure 24. TPQR visual feedback display decision process.

tion

the Memory Map.

The setup asks the box number to be used as input for the display processor. If none is entered, that display is disabled.

Auditory Feedback

The Audio Feedback routine utilizes a remote-controllable stereo tape recorder for the feedback medium. Feedback messages are prerecorded on a control channel under computer control. At the end of each message, a mark pulse is recorded on the second channel. These mark pulses are counted by the processor at fast forward and rewind speed in order to selectively position any message for transmission into the classroom. The messages on the tape are identified by a sequence number.

The computer has five remote-control lines from a driver card in the UDC8 I/O unit to the tape recorder: stop, play, record, fast forward (plus cue), and fast rewind (plus cue). One control line controls the output from channel one so that the sound of the high-speed searches can be turned off. Input to the first channel is the microphone. Input to the second channel is the output signal from a D-A module also located in the UDC8 unit. Output from the second channel inputs into one of the Schmidt-trigger inputs of the programmable clock which senses the mark pulses. An API clock level task counts this input and keeps the very short duration timing required. Interactive operator control is supplied either through a CRT display for instructions, and a button box for the operator's responses, or a teletypewriter input and output handler. The interface between the clock level task and the interactive operator routine is a message counter and a function switch in the clock handler which can be set to; 1) record a mark pulse, 2) do a high-speed search, or 3) play. The interactive routine is notified when the function is done.

The operator can record, position to, and/or play a message. The message number at which the recorder is currently positioned is kept by this interactive routine. The Record function asks the operator to signal the start of a message. The recorder automatically goes into the record mode, and a speaker transmits the message into the microphone. The speaker is then asked to indicate the end of the message for termination. A background clock job keeps the recorder on for at least five seconds, and, when that indicator is entered, a 10-cycle, one-tenth-second pulse is recorded on the second channel by alternating the output of the D-A converter every 5 milliseconds between zero and one volt. This pulse is optimized for the response time in order to detect a pulse-driving forward/rewind speed up to 64 times the normal playing speed. A 2-second pause then allows the recorded pulse to move past the tape recorder read head before the recorder is stopped,and the operator is again asked to enter a system control option.

These message markers are sensed by the clock level tasks when a message is searched for or played back. This sensing is done at normal play speed or fast forward/rewind. Each positive-going edge of the 10 cycles fires the Schmidtt trigger. To help eliminate false pulses caused by line noise, each edge is counted and a 15-millisecond timer is reset. When that counter times out, the count must be greater than six or those pulses will be ignored. This sensing feature subroutine is called during a tape recorder function of play, fast forward, or rewind and with a count of the number of marks to be tabulated before return. The Tape Recorder Control routine loads the control driver buffer/latch to select a Tape Recorder Control function, and sets a background clock task to clear the control driver in 12 seconds.

When a search for a certain message number is to be done, that number must be a legitimate message number for the selected tape. The Recorder Control functions of rewind and fast forward are handled by a search section of the clock level task. The required direction to search is calculated by comparing the current position with the requested position. When the proper number of marks are counted, the direction is reversed, and reversed again, and perhaps again, each time that mark is passed, so as to put the tape directly in front of the specified message-marker. Then the recorder is stopped and the operator is asked to enter a control option. The five-second minimum message length gives time for the recorder to change directions between message marks. The play function places the recorder in play and stops after the first mark pulse.

The loader makes the memory resident test before reading the program. Because of the special clock level task, it must be specially connected to the API Vector. Then the clock must be reinitialized to enable the special real-time rate and input port with interrupt capability. The power fail/restart values for restarting the programmable clock are changed appropriately. The background clock job is connected, and, if the button box-CRT display operator interface is used, the display is connected. The interactive loop is initiated before the loader ejects the Utility Area.

Inter-Computer Data-Link

After a coded teaching session, a printed description of each session is generated from the data file. In earlier CATTS versions, these programs are written in assembly language, and later, an interpretive higher level language (FOCAL) is run locally on the PDP-12. Since these methods require that data collection be suspended during printing, a distributed processing approach is used. With this, the data-line link to the CDC-6600 transfers the data file to the TELEX time-sharing system, which then generates a summary report from a program written in FORTRAN. The report is then available to the user through the time-sharing service from any computer terminal.

Because of hardware, experience turnaround time, and limitations, a dial-up telephone line to the KRONOS/TELEX time-sharing system was selected over a high-speed synchronous data-link designed specifically for intermachine communication.

One particular feature of the interactive terminal facility suited for the time-sharing data transfer function is the TELEX "BINARY" input mode, which allows 8-bit characters to be collected in a file without time-sharing system character translations. The end-of-input data signal to TELEX is a pause with no characters transmitted for two seconds. Each data set consists of two 12-bit words; the first is the time interval, and the second is the packed data word. Thus, each data set can be sent as three 8-bit characters. In addition, the first three characters sent are two 12-bit word sets of date and time of session, the second three characters are box number and number of data points, and the third three character sets are made up of the Starting Block Number (SBN) of the file and zeros.

A longitudinal check sum of the 12-bit values is substituted for the end-of-data code mark.

The following are the functional steps for this data-link.

1. The first step consists of establishing the call. Regular dialing lines are used. One method is to hand dial the TELEX time-sharing system and cradle the handset in a modem. The alternative method uses

an Automatic Calling Unit (ACU) to call the same number with an associated Data Set as the modem.

2. Logging on to the time-sharing system requires the entering of an "H"-"CR" to identify transmission speed and a "W"-"CR" to select the CDC-6600 computer. The operator is then asked to enter user initials, account number, and password.

3. "NEW,A" specifies "A" as the name of the primary file to receive the data. When the "BINARY" time-sharing command is entered, a prompt "ENTER BINARY DATA" is printed on the terminal.

4. The SBN and logical unit number of the file to be sent from the PDP-12 are specified. The file, in the form of characters, is the next string of characters accepted by TELEX after the "BINARY" command. The PDP-12 prints a prompt to the operator when the last character has been transmitted. A two-second interval, during which no characters are received by TELEX, ends the binary input mode.

5. The "PACK" command must be entered before any other time-sharing operations can be done. This command clears a sort file flag and makes file A one logical record. A procedure file is then called, "-Z", which executes a FORTRAN program. This program calculates its own checksum of the file and compares it with the file checksum sent by the PDP-12. If they agree, this file is stored in a permanent file with an indexed entry. If not, an error message is printed, and the file is retransmitted, restarting at step 3.

6. At this time additional data may be appended to the file, the file may be translated, or a summary report may be generated.

7. When interaction with TELEX is complete, the command "BYE" logs off the time-sharing system.

8. Finally, the telephone hand set is returned to its cradle on the telephone. In the case of the ACU, the call request bit is cleared with the "HANGWCC" command.

The method used to establish a call with the Automatic Calling Unit to TELEX employs six lines from a UDC8 driver card which provide four bits of a telephone number digit, one bit of a telephone number digit present signal, and one bit of a call request signal. One line into a UDC8 contact sense card indicates when to present the next telephone digit (PND). One bit from the accompanying Data Set to the contact sense card is the carrier detector. This Data Set is connected to the PDP-12 through a three hundred baud serial ASCII teletype multiplexed port located in the DC02 communications unit.

The other manual method uses a modem connected to a second three hundred baud DC02 port. There are two versions of the transmit program which send data files.

A command "CALLWCC" uses the Automatic Calling Unit to connect the two computers. In order to time the expected interactive responses, a clock job is "quick connected" to the background stream. The call request line is set high and must stay there for the duration of the call. The telephone line is taken from the Data Set to place the call.

Each time the calling unit is ready for another digit of the phone number, it sets the present next digit (PND) on. Thus, the program alternately (1) sets the call request bit only and waits for the present next digit line to go high, and (2) loads the driver buffer with the next digit, the digit present bit, plus the call request bit, and then waits for the PND line to go low.

This load-and-wait function is accomplished by a subroutine called by the main program. It loads the driver value, resets an alternate entry point counter, and sets the clock job to return in .1 seconds. When control is returned, the status of the PND line is read and returned to the main program. This value is tested and, if it's not set, another entry point to this subroutine resets the clock job to return another reading of the PND line in .1 of a second. A count is kept of the number of times this alternate entry point is used. If this count expires, the expected response is not received by the PDP-12 in the expected time interval, and an error exit is made from the program which prints an error message and ejects the Utility Area.

To return the telephone line to the Data Set, an end-of-digit code is sent over the telephone number digit lines. After a pause of eight seconds following the last telephone number digit, it is assumed that the other end has had time to respond by answering and is sending the handshake signals necessary. If the call was successful, the carrier line should be high after about two seconds. If the carrier line does go high, the program prints the success message and ends by "quick disconnecting" the clock job and ejecting the Utility Area. If the line doesn't respond in four seconds, the program prints the failed message before ending.

Once the call is established, either automatically or manually, a program to enable the operator to interact with the CDC-6600 time-sharing system is loaded. The commands "TMITWCC" and "TMIT" are identical except for the data-link port used, which is determined by the method with which the call was made; select and load "TMITWCC" if the call was placed automatically, and select "TMIT" if the modem is being used. Both of these commands stay in the Utility Area until cleared, which causes data collection sort routines to be delayed or ignored. "TMITAUX" is very similar but it occupies its own area of memory and can select either port. All subsequent references to "TMIT" hold true also for "TMITWCC." These Transmit programs are a collection of three co-routines. One routine waits for input from the data-link port and outputs those characters to the operator's console using the buffering routines provided by the Command Module. Another routine waits for input from the operator's console and outputs these characters both to the console printer and the data-link port. No buffering is required for the data-link output, so each character is sent out when it is received. The third routine reformats the selected data file and sends it out the data-link port as sequential characters.

The steps of logging on, preparing TELEX to receive the file, saving the file, selecting and generating the feedback, and logging off are all done by communicating through the PDP-12 and the data-line link to the TELEX time-sharing system just as though the operator were at a terminal. Every character typed by the operator on the teletype is output on the data-link port with no intermediate buffering. Each is also echoed back to the operator's teletype through a ring buffer. Since "TMIT" uses the Command Module's output buffering, carriage returns (215_8) are converted to 36's, and line feeds (212_8) are ignored. Every character to come from TELEX is typed on the operator's teletype through that output buffer with the same special character translations.

The input from the operator's teletype is also monitored for two control characters. One of these is a control/T. This is the feature that allows the operator to select a file from the permanent file storage space on the PDP-12 and to transmit it through the data-line link to TELEX.

The control/T initiates an interactive routine which prompts "Transmit Which?" on the PDP-12. The characters then entered on the operator's teletype are not sent through the data-link, but are buffered by the systems CDRD Command Line Input routine. When a line of input has been accepted, GETO is called to get the SBN of the file to be sent, and transmission is initiated. If an "A" is encountered before the SBN, the auxillary disk storage unit number 15 is selected as the logical unit containing that SBN. When transmission is complete, the prompt 'SENT" is printed and the input from the operator's teletype is reconnected to the data-line link. The operator must wait at least two seconds after the "SENT" prompt to end the TELEX BINARY input mode before entering the next command to TELEX.

The other control character from the operator's console that is monitored by the PDP-12 is a control/E. This function prints "THE END," clears the two inputs enabled from the command console and the DC02 data-link port, ejects the output of the DC02 data-link port and ejects the Utility Area, allowing other routines and commands to ~~use~~ it. Figure 26 diagrams the inter-computer data-link process.

The operation of "TMITAUX" is very much the same as "TMIT." But, since this program is not a Utility Area command and uses the auxillary teletype, short substitutes for teletype input and output are used. "TMITAUX" has its own ring buffer and delete code. To eliminate the need for a text-buffering routine, the "TRANSMIT WHICH?"prompt and the "SENT" indicator are reduced to the single characters of "S" and "Carriage Return," respectively. A special input routine does not buffer a line of input but accumulates an octal SBN number from the console input. Editing is done not with the rubout key but by reinitializing the value with another CTRL/T. Entering a "T" before the SBN directs the program to look for the file on LINCtape rather than the disk.

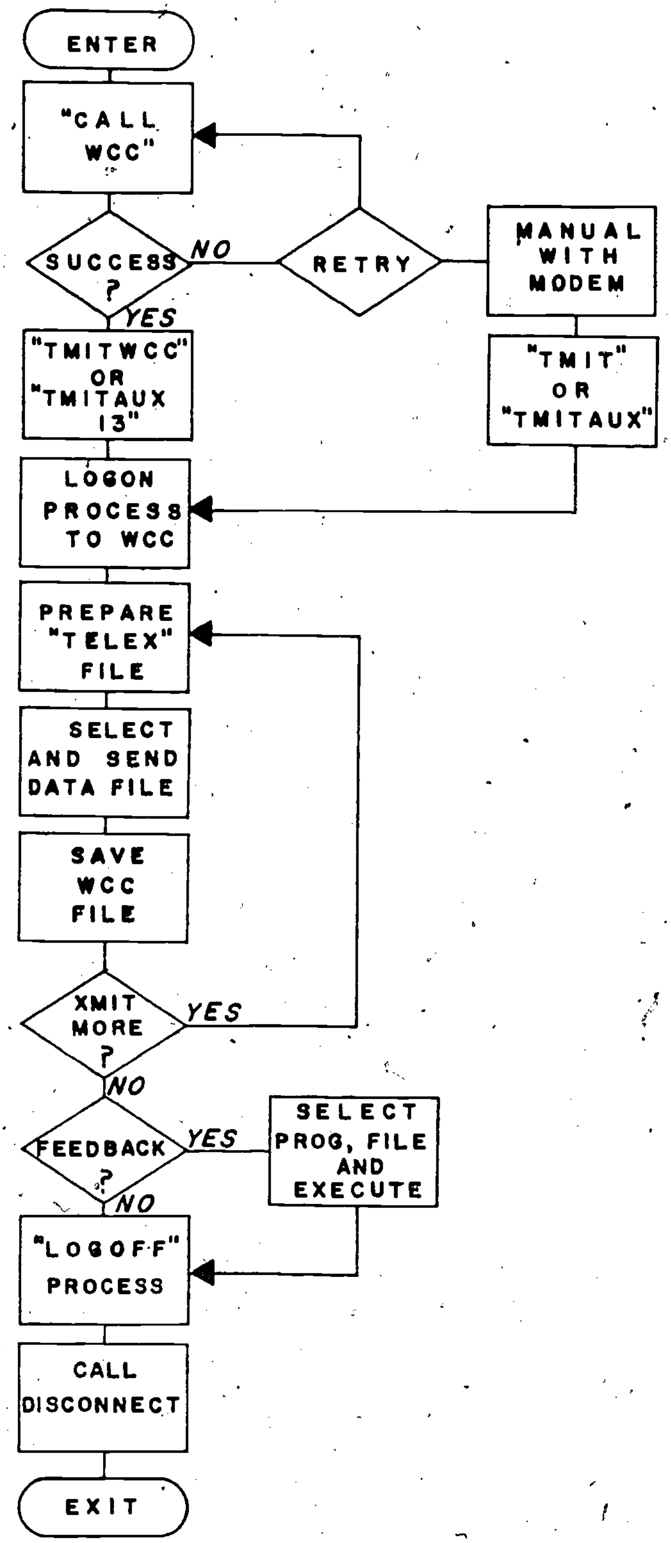


Figure 26. Inter-computer data-link transmission process.

The loader checks to see if the required area in memory is occupied. If not, the program block is read and the Memory Map bits are set. Input from the auxillary teletype is enabled. The data-link port is optionally selected according to the command input buffer. If a "13" is appended to the "TMITAUX" command, the DCO2 port connected to the ACU is used. Otherwise, the manual modem port is assumed. Since there are no other users of these ports, nothing is scheduled on the output device handlers, and input is enabled to the waiting routine.

An unloading command "TMITAUU" clears the program and allows it to be reloaded with a different data-link port. First, the Memory Map bits are tested to check whether a program occupies that area in memory. Then a test is made on the first instruction of that area. If it's the correct program, the memory bits are cleared and the inputs are disabled. The data-link output handler is set to "not busy."

Automated Data-Link. Initially, any program that automatically sends the files to TELEX must be simple and straightforward. An assumption is made that all interactive processes will operate smoothly. If problems occur, the elementary error recovery procedures must also provide documentation on the problem areas. Since this function occupies the Utility Area, the operation must be fail-safe, with timers clocking expected responses and counters limiting retry loops without retrying any step too often.

The operator control initially required is to add or delete the automatic sending option. The Automatic Transmit program is added to the Sort function which uses the Utility Area. If transmission operates smoothly, this function is not required to stay resident in memory. The automatic version of "TMIT" does all the steps of the manual version, except steps six, seven, and eight as outlined earlier.

Each box's data from one session must be sorted before the next session on that box ends. So, if each file is sent after it has been sorted, the log-off step is eliminated by not requiring a re-dialing procedure for each file, thus increasing the response time when many files are being collected. TELEX automatically clocks inactive periods on terminals and automatically logs a user off after 10 minutes of inactivity.

The automatic version mimicks the operator's entries in the manual operation. The function of sending a command and waiting for the response is done by a "Send" subroutine. Each character of the message is transmitted by the DC02 port and optionally buffered for printing on the console device. A "CR" is transmitted at the end of each message. An "LF" character is expected to be transmitted from the CDC-6000 immediately after transmitting the "CR." Another prompt is expected to follow when the "Send" command is completed by TELEX. A 1-second clock job will time out if there is no response to the "CR." This error is counted and the re-dialing code is called by the PDP-12. If there is a returned response of a "LF" character, the clock job is reset to 17 seconds in order to detect the end of the next time-sharing prompt. Each character of that prompt resets the end-of-prompt timer to three seconds. Three seconds after the last character of the time-sharing prompt is received, the Send subroutine is exited, usually to transmit another message.

Each character from TELEX is saved and optionally echoed on the console device. If the Send subroutine is waiting for a response, that character is passed for further interrogation. Any messages received at an unexpected time are optionally echoed but otherwise ignored.

Each character is optionally echoed on the console device. If the PDP-12 sense switch zero is set, the button box printer buffering code is called to type that character. Before each character is transmitted out the DC02 port to TELEX, the carrier detect line is sampled. If it is not on, the carrier must have dropped. This error is counted and the re-dialing code is recalled. Communication checks are reserved to three go/no-go checks in the transmission sequence.

The call is placed with the code from the "CALLWCC" command. An added feature at the start of this code clears the call request bit for five seconds to be sure the phone line is ready for dialing. Carrier detect must be established after placing the call through the ACU. If not, this error is counted and the re-dialing code is recalled. This error exit is also used if the ACU does not respond as expected.

The logging in is performed by transmitting an "H" followed shortly by the "CR." The brief pause allows TELEX to discriminate the two characters.

A brief end-of-prompt pause is needed after transmitting the "W"-"CR" in order to wait for the time-sharing header and user identification prompts. A valid log-in test checks if the last character received after transmission of the initials, account number, and password is a "colon" from the "RECOVER/SYSTEM:" prompt returned by TELEX. If it is not, the user "id" line is retransmitted, up to three times. If the log-in check still isn't positive, the carrier is monitored for up to six minutes. If all of the TELEX ports are busy, any message from TELEX will reinitiate the program in order to transmit the user "id" line. If the "colon" transmission is garbled, the timer elapsing will count that as an error and recall the re-dialing code. If TELEX isn't up or the carrier drops for any other reason, the carrier test before transmitting a character will count that error and re-dial.

After logging in, the "NEW A" and "BINARY" commands are transmitted. The file just sorted is then read and transmitted to TELEX with the same routine that "TMIT" uses. The "PACK" and "-Z" commands are transmitted.

The wait for the "acknowledge" message from TELEX is somewhat longer. Manual operation shows it to be usually less than 30 seconds with no other intervening messages. The next time-sharing prompt, 'READY" appears within five seconds of the acknowledge message. Each file sent must receive the acknowledge signal of "AAAAAA." If necessary, the file is re-transmitted up to three times, restarting with "NEW,A." If still unsuccessful, this error will be counted and the re-dialing code will be recalled.

The re-dialing code will be recalled only twice. When the error count exceeds two, the attempt to transmit this file is aborted, and control is returned to the Sort program to look for more sessions to be sorted. Other errors, such as program or data read errors, will also cause this Automatic Transmission program to abort.

Currently, a record of these untransmitted files is kept by the operator, who can transmit them over with the manual methods.

DITRMA

The DITRMA program assists in training an observation system by comparing the coding entries from several observers and reporting on their agreement or disagreement through a visual feedback method. It is also used throughout an observation study in order to maintain the observer's catego[illegible]ation abilities. The training site has up to six button boxes, [illegible] video [illegible]tors, a VTR remote-control console, and a tone generator. The PDP-12 has three modes of feedback to observers. One of the TV monitors is connected to a VTR at the computer site which plays a video training tape. A remote-control console provides VTR controls for reviewing previous segments of tape. The other TV monitor is a closed-circuit connection with a TV camera which is focused on the CRT display that transmits various status messages and code lists back to the training site. The computer also employs relay connections to the VTR's remote-control start and stop lines. Another relay turns the tone generator on and off for an auditory "agreement" feedback signal.

Operation. One of the button boxes used is designated the "Master" box. It is used to begin and end DITRMA sessions, and to enter the "consensus" code when a disagreement automatically stops the tape. To begin, the Master box is switched on, identification data is entered, and entering an "0011" starts the VTR and begins the DITRMA session. Any of the other boxes can begin at any time in the session by switching the box on and entering the "0011" code. DITRMA does not compare the control codes "0011" and "0099," or the identification data. Any box can drop out of a session by entering the "0099" code and turning the switch off.

When the first code is entered, the PDP-12 CRT displays:

```
      CODE

    INITIATED
```

When all first codes are entered, the program compares the data. A short feedback tone to the observers indicates agreement. Otherwise, the training tape VTR stops and the PDP-12 displays the disagreements. The code list display format is described below. All the box lights are turned off and no further codes are accepted from the boxes. The trainees at this time discuss their disagreement and reach a mutually agreed-upon consensus. When the Master box is turned on, the consensus code is entered to restart the VTR, re-enable input on all the boxes, and return the display to:

Coding resumes from that point or the tape may be rewound to review the taped material.

Observers are not restricted to entering the same code simultaneously. An observer may be ahead of the other observers if the observer has already coded the event that the others are currently coding and then codes a further event. The observers may vary in event synchronization by as many as four codes. All extra codes that are displayed in the code list are ignored after the VTR restarts and must be re-entered.

To end a session, code "0099" must be entered on the Master box to stop the VTR, and the box must be switched off. The other boxes may do likewise, as in an actual data collection session.

Routines and Data Structures. The DITRMA program is similar to the video feedback programs in that it has a code-processing section and a display refresh section. The code-processing section, however, requires response to some of the individual characters from the boxes. Therefore, the code-processing section is implemented similar to a Data Collection Input routine. The first section provides box controls and individual data code construction. From these, the codes are collectively buffered in a Step Table Array prior to the comparison and indicative response.

There are several different items kept in the Step Table Array. A step is a row in the array which contains codes from the boxes for the same coded event. Any code of an event ahead of the event being coded by the slowest coder is stored in the next step row for that box's column position. The column position for any box number is indicated by which position that box number holds in a separate array called the Box Number Table. Any box can be as many as three codes or steps ahead of the slowest. The fourth row is used to indicate which step that column or box is currently on. One column (seven) is used to keep a count of the number of boxes that have made entries in that step. Another column (zero) is the all-codes-entered timer for that step.

Data is stored in the intermediate data collection buffer under the Master box's number, and is stored as a data file with a special format. When all the coders agree on an event or step, that consensus is stored. When there is disagreement and a code list display is generated, a special data flag is stored, followed by the error type and the data codes in step zero only. The consensus code entered on the Master box that restarts the session is also stored.

The beginning box control section allows the Master box to begin and end sessions and performs the box ON/OFF and UP/DOWN functions, plus individual digit buffering [illegible] ting [illegible] put routine. The number of active boxes is kept by this section. The data code is packed on an ENTER key. If any pack tests are positive, a type-two code list will be generated, describing a specific coder error.

Now the Step Table index for this box number is checked in the Box Number Table. If the current step number for this box is already at three, the last step, the code list that says "TOO FAR AHEAD" is generated. If not, the data is saved at the current step number, and that step number for this box is incremented.

If this is the first entry to this step number, the "CODE INITIATED" text is moved into the display buffer area, and the all-codes-entered timer for this step number is set to the wait time. This wait time is selectable under program control (DTIMER command). A clock job increments each step's timer toward zero. If all boxes dial to enter a code, step zero's timer will reach zero, which generates the "TIME" code display.

If all boxes enter a code, the number of entries for step zero will equal the number of active boxes, and the timer will be reset. If each active box's data codes stored in step zero are the same, the steps will shift, moving the timer, data codes, and entries count of step one to step zero, two to one, and three to two. Step three is cleared. The current step indicators must be decremented by one. The tone is fed back and, if there are no entries in the Step Table, the display is changed to "DITRMA." If the comparison shows different codes, the "DISAGREEMENT" display is generated.

The following code lists describe what caused the error display and also list the codes buffered in the Step Table. The display has the general format of:

"ERROR TYPE"

	Entry			
Box	1	2	3	4
1 (Master)	Data	Data	Data	Data
2	Data			
3	Data$_1$			
4	Data	Data	Data	Data$_4$
5	Data	Data$_2$		

Data from the Master box is in row 1. The other boxes are the following rows in descending order. If any observers have entered more codes than others, these extra codes ahead are displayed in that box's row in the second, third, and, possibly, fourth data columns. Numbers subscribed next to various entries indicated in the format above refer to the type of error made by the coder. A description of error types follows:

1. ERROR TYPE: <u>DISAGREEMENT</u>

   Probably the most frequent, the conflicting codes can be seen in the first data column.

2. ERROR TYPE: ERROR

The CATTS Data Collection program makes some tests on the digits entered as a data code. If one of these tests is positive, the violating digits will be followed by an "*".

Those tests are:

a) Code exceeds a maximum for that level.

b) Too many digits have been entered. May be caused by not ending the previous code with an ENTER key.

c) No digits before the ENTER key. A "*" is in that column.

3. ERROR TYPE: TIME

When an observer codes something and the CODE INITIATED message is displayed, all the others must follow with something in a preset amount of time or a timeout will occur. A timeout timer is running anytime CODE INITIATED is displayed.

4. ERROR TYPE: TOO FAR AHEAD

Caused by one fast coder being five codes ahead.

After the data from the Step Table is displayed, the first step (zero) is stored in the data collection buffer, and the Step Table is cleared. All the box lights are turned off and codes are not accepted. The VTR stops and the lockout switch is set which causes all characters from the boxes to be ignored. In five seconds, the lockout switch is changed to look for characters from the Master box only, and the Master box lights are turned back on. When the consensus code is entered, it is stored in the data collection buffer. All box lights are turned back on and the lockout switch is cleared to accept data from all the boxes. The VTR starts again and "DITRMA" is displayed.

The display refresh routine simply displays the packed 6-bit characters currently in the display buffer. A routine moves a requested text list to the beginning of this display buffer. The code list generator then "prints" one or two digits per number and asterisk, space, and CRLF characters where appropriate into the display buffer.

The VTR and tone controls are manipulated by setting a bit in the relay buffer. Then a clock job is set to clear the buffer in two seconds, producing a short pulse rather than a continued ON/OFF, or tone signal. The VTR can be set so that the picture will remain recognizable when stopped and restarted.

A "nonstop" mode of program operation alters the Code List function to eliminate the VTR stopping and waiting for the consensus code. Instead, the code list is generated, the data stored, and the Step Table shift is called to prepare the Step Table for continued data entry. If data codes for the same event become stored in different step numbers, the all-boxes-entered timer will resynch the input codes after a pause of that length with no codes being input.

Loader. Since the DITRMA program uses the Button Box Input routines, the loader makes certain that they are loaded before checking and reading its own routines. If successfully read, the display and clock jobs are added to the appropriate lists. The page variables are moved into place and the Memory Map bits are set to "busy."

The DITRMA interactive setup is similar to the button box setup and requires additional variables and tables to be determined. First, the "MAXIMA" query is prompted which sets the pack parameter for the Box Table. This data is also used to set a number-of-levels indicator and a number-of-digits-per-level table in the code list generator. The responses to the BOX NUMBERS query specify which boxes will have their inputs enabled to the DITRMA input.

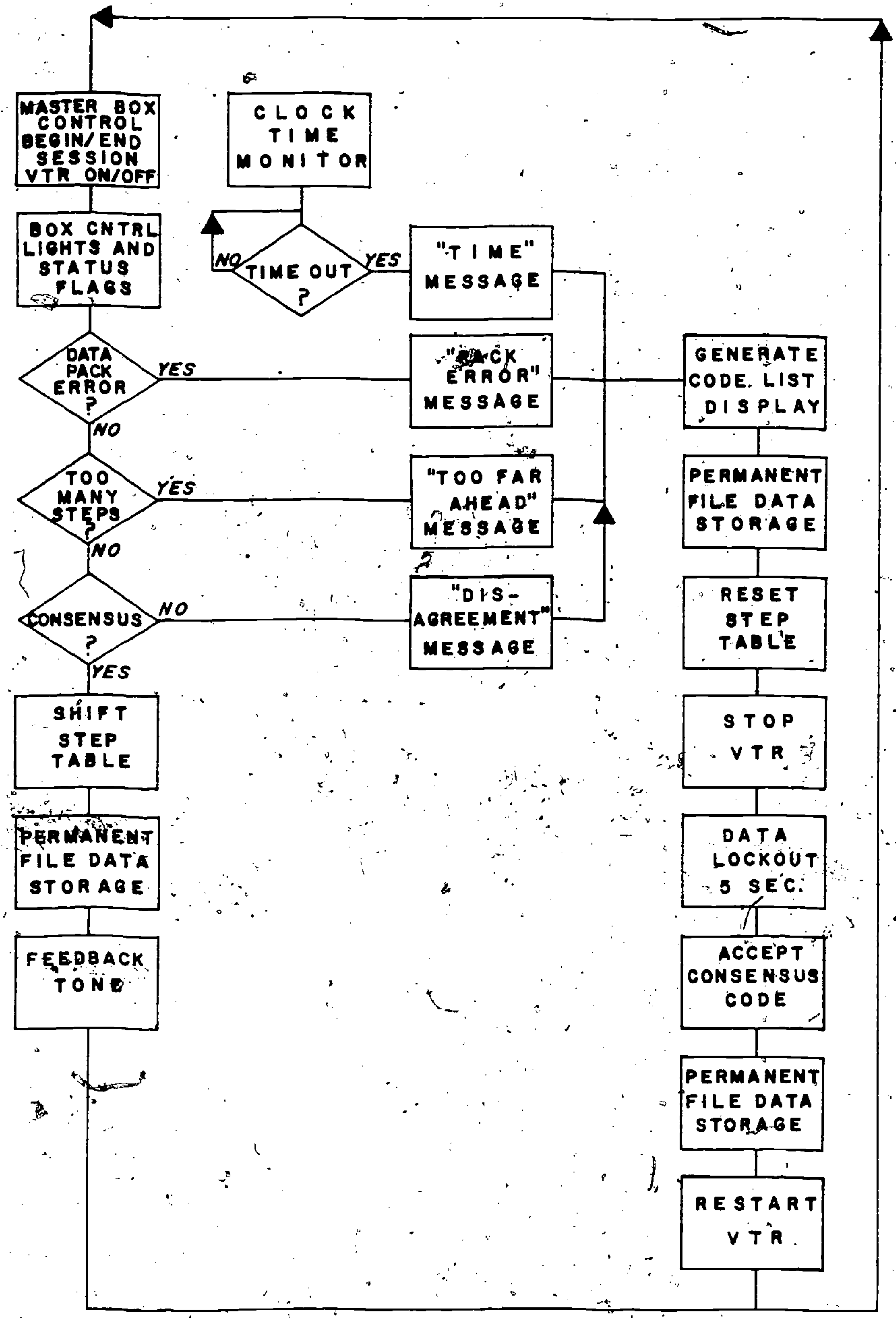


Figure 27. DITRMA program process.

CHAPTER 6

Conclusions

Technical Experience with CATTS

Preservice/Inservice demonstration projects: Data collection. CATTS programming was implemented to relieve the observer from spending too much time with the tedium of mechanically entering observations into the button box, which might take away the observer's attention from the teaching situation under observation. The observer need only discriminate the behavior to be coded and press the appropriate category number into the button pad matrix, then press the ENTER button to complete the category observation. If the observer is aware of an entry error before the ENTER key is pressed, the SKIP button is pressed and the correct category is re-entered before pressing the ENTER key. CATTS automatically senses the SKIP key and clears the initial data buffer to allow for re-entry of the corrected code. CATTS also recognizes an ENTER key that is pressed without category data preceding it and ignores the mistaken entry. An error message is then printed on the monitor teleprinter which identifies the "no data entry" error that was detected.

A data syntax check is also performed by CATTS, which checks the incoming data for the expected number of button entries for each observation and whether or not the data entered exceeds the maximum value that the category system will allow. Both of these errors are also ignored by CATTS, and appropriate messages are printed.

The messages that are printed on the console identify either poor or incorrect coding procedures by the observer, or a faculty button box configuration. The data system self-checking ability provides the observer with the simplest procedure for data entry and releases the observer's attention to the data collection task at hand.

Most of the technical problems encountered with the development of the CATTS data collection configuration occurred with the early DATAMYTE portable collection systems. These portable data collection systems recorded the data onto magnetic tape cassettes for decoding back through the computer at a later time. The high degree of error introduced by this audio tape data transfer system caused missing data problems and data dropouts. The battery power supply of the early DATAMYTES also led to missing sessions when the batteries failed during data collection. Since then, the Electro-General company has marketed a new model DATAMYTE which eliminates the audio tape system and stores the data internally into a solid-state memory chip. Recent experience has shown that the new DATAMYTE data collection system has been reliable in the field.

TOUCH-TONE telephone data collection has also been shown to be a reliable method of data collection, but is dependent upon the integrity of the telephone line supplied by the telephone company. A demonstration project in Hammond, Indiana, 220 miles from Bloomington, was completely successful, while a project in Indianapolis, 60 miles away, experienced telephone line trouble between the two cities. Locally, there has been no data collection trouble with the telephone system which proved to be as reliable as directly connected button boxes.

Preservice/Inservice demonstration projects: Feedback. Data summary feedback and instantaneous feedback to trainees at the data collection sites has proved to be reliable. The instantaneous video feedback applications have been technically successful from the beginning, limiting the initial problems to the development of what to display to the trainees at the teaching site. These problems were instructional in nature and dealt decisions regarding complexity of feedback available in re uch information can be decoded by the trainee in a real-ti The summary printout feedback has progressed through a ser opmental steps all focused upon the ability to get the pri ee in the shortest possible time. Initially, CATTS was no ce summary printouts simultaneously during real-time data intouts had to be generated after data collection sessions d. As CATTS evolved, the printout responsibility was turn ampus computer center so that a printout could be generate sharing terminal soon after the data was collected, transm d at the computer center from the PDP-12, either automatic tor initiated. The placement of the data summary responsi ampus time-sharing system also permitted the use of higher (i.e., FORTRAN) for programming more flexible and compreh ries. This decision released the PDP-12 CATTS operating s nsible only for the data acquisition control processing, a storage and summary processing to a larger time-sharing co ore efficient processing and simplified programming applic

DITRMA Training. Because DITRMA training is limited at th tly connected button boxes, video feedback displays from t emote-control of the 1-inch VTR for training materials, no ems have been experienced. After initial familiarization n and its control, all training sessions have been success ed out. As would be expected, the success of the training ds upon the observation system being trained and the visua tional materials that accompany the training package.

-time and
situation.
s of
out to the
able to
llection,
ere com-
over to
from any
ted, and
ly or
lity onto
evel lan-
sive
em to be
left the
ter system
ions.
time to
PDP-12,
echnical
h the
lly
self
and in-

Reliability of CATTS functions

As outlined in the software section of this manual, the CATTS operating and data collection system is composed of many routines that all must function reliably in order to respond to the real-time collection, storage, and feedback of the data. Other than developmental problems, CATTS has been reliable and accurate from the beginning, with a minimum loss of data. The PDP-12 hardware system has performed very well, with a low record of machine failure and down time. Both software and external hardware backup systems have been installed from the beginning to insure a minimum loss of raw observation data due to any hardware malfunction, either in or external to the PDP-12 system.

Cost-effectiveness of CATTS

Experience gained in the development of both the hardware and software components of CATTS has led to the ability of the system to collect and feed back data from many sites simultaneously. The current input capabilities are 12 real-time data ports channeled through the UDC8 subsystem and two off-line ASCII ports for DATAMYTE data in the DC02 communications units. The logical expansion of both input systems is a simple matter of adding only the input expansion hardware and making minor changes in the software routines. The software has been written to accept additional input sources.

The summary feedback system based at the campus computer center has provided an unlimited resource for access into and summary of a very large data base from any time-sharing terminal on campus or throughout the state of Indiana.

The two CATTS functions of data input and output can provide service to many classroom applications, with a minimum of additional hardware costs. The hardware additions for expansion only exist at the data collection

PDP-12 site and require very little software updates. The DATAMYTE data collection process has also been transferred to the campus computer center through any time-sharing terminal, leaving PDP-12 CATTS responsible only for real-time data collection display and control functions.

CATTS Technical Future

CATTS is not just a specific hardware configuration but rather a methodology which can take many variations in its configuration to fulfill the functions for the analysis of process-type observation data.

CITH/I.U. Central Network. Through the application of telephone communications, the current real-time hardware/software configuration located at CITH at Indiana University would serve the needs of data collection on feedback requests for Bloomington and Indiana cities reachable by I.U. network telephone lines. Real-time data collection would take place with TOUCH-TONE telephones for locations remote from the CITH location. Summary printouts would be available on the Indiana University Computing Network (IUCN) which has special time-sharing local telephone lines to all State-supported campuses throughout Indiana. DATAMYTE data would also be processed through the IUCN.

National Time-Sharing Network. The extension of CATTS functions into a national time-sharing network would extend the data collection and summary requirements of DATAMYTE data applications to users anywhere in the U.S. through local telephone calls from locally accessible computer terminals. A demonstration of this application is currently in operation at the BOCES Pinesbridge School located in Yorktown Heights, New York. Data is collected on DATAMYTES and then transmitted to a national time-sharing network for summary processing and storage. The same software programs now are also accessible to anyone at any location who has permission to use the system.

Portable DITRMA units. Development is currently underway at CITH to provide portable microprocessor-controlled units that will provide restricted DITRMA functions to remote locations that are not accessible or economically feasible for providing real-time or time-sharing services. These will be self-contained units that will allow DITRMA training of observation system anywhere, requiring only a VTR and monitor for showing the training material.

Mobile CATTS unit. The complete real-time CATTS configuration can be reduced in size to fit in a small mobile van for transportation to classroom sites throughout the country. Application of CATTS services from a mobile van would require the van to be placed adjacent to a school or central facility that has telephone line access to a number of schools. The van could then collect, display, and feedback observation data to all locations. This application could be used in conjunction with an intensive inservice instructional training program.

REFERENCES

Amidon, E. J. & Hough, J. B. (Eds.) *Interaction analysis: Theory, research and application*. Reading, Mass.: Addison-Wesley Publishing Co., 1970.

Baker, H. P. *Film and video tape feedback: A review of the literature*. Austin, University of Texas, Research and Development Center for Teacher Education. Report Series No. 53, 1970.

Bellack, A. A., Kliebard, H. N., Hyman, R. T., & Smith, R. L. *The language of the classroom*. New York, Columbia University, Teachers College, 1966.

Bilodeau, E. A., & Bilodeau, I. M. Motor skills learning, *Annual Review of Psychology*, 1961, *12*, 243-280.

Bondi, J. C. Feedback from interaction analysis: Some implications for the improvement of teaching. *Journal of Teacher Education*, 1970, *21*, 189-196.

Bourne, L. E. The effects of delay of information feedback and task complexity on the identification of concepts. *Journal of Experimental Psychology*, 1957, *54*, 201-207.

Broadbent, D. E. *Perception and communication*. London: Pergamon, 1958.

Cartwright, C. A., Cartwright, G. P., & Robine, G. G. CAI course in the early identification of handicapped children. *Exceptional Children*, 1972, *38*, 453-459.

Collet, L., & Semmel, M. I. *The analysis of sequential classroom behavior*. Unpublished paper, University of Michigan, Office of Research Services, School of Education, 1970.

Fink, A. H. & Semmel, M. I. *Indiana behavior management system--II, observers' training manual*. Bloomington, Indiana University, Center for Innovation in Teaching the Handicapped, 1971.

Flanders, N. A. *Analyzing teaching behavior*. Reading, Mass.: Addison-Wesley Publishing Co., 1970.

Galloway, C. M. Nonverbal communication in teaching. In R. T. Hyman (Ed.), *Teaching: vantage points for study*. New York, Lippincott, 1968.

Gibbs, C. B. The continuous regulation of skilled responses by kinaesthetic feedback. *British Journal of Psychology*, 1954, *45*, 24-39.

Greenspoon, J., & Foreman, S. Effect of delay of knowledge of results on learning a motor task. *Journal of Experimental Psychology*, 1956, *51*, 226-228.

Heinrich, D., & McKeegan, H. F. Immediate and delayed feedback procedures for modifying student teaching behavior according to a model of instruction. Paper presented to the American Educational Research Association, Los Angeles, February, 1969.

Kreider, J. M. The effect of computer-assisted teacher training system feedback on increasing teacher use of pupil ideas with EMR children. Unpublished doctoral dissertation, University of Michigan, Ann Arbor, 1969.

Lynch, W. W., & Ames, C. .Individual cognitive demand schedule, observer's training manual. Bloomington, Indiana University, Center for Innovation in Teaching the Handicapped, 1971.

Medley, D. M., & Mitzel, H. E. Measuring classroom behavior by systematic observation. In N. L. Gage (Ed.), Handbook of Research on Teaching, Chicago: Rand McNally, 1963.

Noffsinger, T. & Daiker, J. F. EMR program development: Ohio ESEA Title III. Mentor, Ohio: Mentor Exempted Village School District, 1973.

Reddy, W. B. Effects of immediate and delayed feedback on the learning of empathy. Journal of Counseling Psychology, 1968, 16, 59-62.

Schmitt, J. S. Modifying questioning behavior of prospective teachers of mentally retarded children through a computer-assisted teacher training system (CATTS). Unpublished doctoral dissertation, University of Michigan, Ann Arbor, 1969.

Semmel, M. I. Project CATTS I. A computer-assisted teacher training system. In A. P. VanTeslaar (Ed.), Studies in language and language behavior, Progress Report No. VII, Ann Arbor, University of Michigan, Center for Research on Language and Language Behavior, USOE, Contract OEC-3-6-061784-0508, 1968.

Semmel, M. I., Olson, J. L., & Weiske, W. M. An information and technical manual on the computer-assisted teacher training system (CATTS). Bloomington, Indiana University, Center for Innovation in Teaching the Handicapped (mimeo), 1971.

Semmel, M. I. Application of systematic classroom observation to the study and modification of pupil-teacher interaction in special education. In R. Weinberg and F. H. Wood (Eds.), Observation of pupils and teachers in mainstream and special education settings: Alternative strategies. Minn.: Leadership Training Institute in Special Education, University of Minneapolis, 1975.

Simon, A., & Boyer, E. G. (Eds.), Mirrors for behavior: An anthology of classroom observation instruments. Philadelphia: Research for Better Schools, 1970.

Smith, K. U., & Smith, M. F. Cybernetic principles of learning and educational design. New York: Holt, Rinehart & Winston, 1966.

Spaulding, R. L. Educational intervention in early childhood, Vols. I, II, and III. Durham, North Carolina, Duke University, 1971.

Stolurow, L. M. Automation in special education. Exceptional Children, 1960, 27, 78-83.

Swets, J. A. & Kristofferson, A. B. Attention, Annual Review of Psychology, 1970, 21, 339-366.

VanEvery, H. J. The application of a computer-assisted teacher training system to speech therapist training. Unpublished doctoral dissertation, University of Michigan, Ann Arbor, 1971.

Weaver, P. Effects of a computer-assisted teacher training system and teacher expectancies on teacher-pupil verbal interactions with EMR children. Unpublished doctoral thesis, University of Michigan, 1969.

Appendix

Command List

Console Monitor Command Communication

Typing the "line feed" key on the command teletype starts the command handler. ">" is printed. The name is the command itself. Some commands have parameters with the command name. Others have an interactive setup with prompts.

In the following examples of parameter lists, letters are used to denote variables, and letters enclosed in quotes, such as "0", are the actual characters to be used for that parameter. Underlined characters, words, and phrases are printed by the computer. M&M is a list of all possible messages that could print and their meaning, plus any action that may need to be taken.

M&M	System response:	Possible solution:
	NOT IN INDEX	Check spelling
	CAN'T READ INDEX	Check disk or tape-drive control switches
	NO BINARY FILE	Notify programmer
	URE BBBB	Read error this block

CATTS Data Collection Commands

CATTS

CATTS2

These are the Button Box Input routines with the basic data storage configuration. CATTS2 boxes use control codes. The interactive setup enables input on the box numbers selected and defines coding parameters for those boxes.

MAXIMA I J K These are the maximum categories for levels 1, 2, and/or 3.

SESSION LENGTH L If the machine is to automatically end a session, enter the time in minutes, otherwise enter 0.

BOX #'s M N O . . . Up to 12 box numbers in the range 1 through 12. This setup is ended by pressing CR or RETURN after MAXIMA.

M&M: Real-time Data Collection

ON N HH:MM Box N switched on at HH:MM.

OFF N HH:MM Box N switched off at HH:MM.

A N HH:MM NO DIGITS CODED BETWEEN ENTERS on box N at this hour HH and minute MM.

B N HH:MM CODE EXCEEDED THE MAXIMA on box N at this hour HH and minute MM.

C N HH:MM TOO MANY DIGITS BETWEEN ENTERS on box N at this hour HH and minute MM.

S PPPPBBBB Sort not done on earlier session.

W BBBB Backing-Store-Write error on block BBBB, notify programmer.

M&M: Loader

Console requests:

DATE (MM, D, YY) Jan. 1, 1975 would be 1 1 75.

TIME (H,M) 3:45 in the afternoon would be 15 47.

BAD READ Could not read the program, or unit 0 or 6 not selectable.

-C Write locked.

MEM BUSY Another program occupies needed memory.

M&M: Data Sort Process

SB N HH:MM BBBB	Normal SBN number of box N.
SN N HH:MM EMT	No data codes or SBN.
SB N HH:MM BBBB P	Partial, notify programmer.
SE UBBBB	Read or write error, notify programmer.

DATAMYTE Input Commands

DM1

DM2

MAX I J K "O" L "S" "C" M

I J K MAXIMA, as in CATTS

DM1 is Box 13, DM2; 14.

"O" L L = 1 or 2, sets data listing on 1, the command console on 2, the auxillary printer.

"S" Restart a tape, ignore input from last tape.

"C" M M = 1 through 9 entered where appropriate control code should have been. C9 should give an SBN.

Coding errors A, B, or C are printed after the timer digits.

M&M: DATAMYTE Loader

-R Bad read when loading.

M&M: CATTS Data Collection

TOUCH-TONE Telephone Input Commands

PHONE

MAXIMAS Same as in CATTS2.

BOX #'s 10, 11, or 12. These boxes cannot be used for anything else when PHONE is loaded. Any previous setup of them is cleared. Must enter CR after last MAXIMAS in order to exit setup.

Placing the call is the same as switching the box on; ON N HH:MM.

Hanging up is equivalent to off; OFF N HH:MM.

To hang up a data set that did not do so automatically, depress the talk button on the data set, lift the receiver, and hang up.

M&M: Loader

-P   Bad read when loading.

M&M: CATTS Data Collection

Auxiliary Commands

OFF N   This is the equivalent of switching a CATTS2 box off. Notify the programmer whenever this is used.

$0 \leq N \leq 15$ Box Number

M&M: Loader

WRONG   CATTS not loaded or N out of range.

DATE   Interactively requests a new day and time setting for the basic storage configuration.

Prompts:

(Current date and time)

TIME (HH,MM)

DATE (M,D,PP)

No response will leave values unchanged. Must do CR to exit.

BOXES   This command will generate a list of which boxes are currently being used.

M&M: Routine Report

N (ON) (UP) MM:SS (U)   in any combination. Box N is ON and/or UP with M minutes and S seconds left in session. U means unsorted.

None active at this time.

00 HH:MM

CLIST

CLISTA These commands generate a printout of all data codes entered in a session, 5 points per line. Each types on a different printer.

Prompt:

ENTER SBN B

B Starting Block Number.

M&M: Error Report

BAD READ of data

ELIST "A" "U" I NN . . .

This is another data list command modified to print 1 data point per line, and respond to control codes entered.

"A" Data list is to be generated on auxiliary TTY.

"U" I If data file is not on default unit 16, type "U" and logical unit number, I, 0-7 for LINCtapes, and 10-16 for disk units.

NN . . . Type one or a list of SBN's.

Typing a CR or RETURN will skip to the next list or end the listing.

M&M: Loader

-R Bad read of data

INDEX "A" "U" I "B" J K

This command generates a list of SBN's, Box #'s,

	time down, and data.
"A"	Auxiliary TTY.
"U" I	Unit I; 0-7 = LINCtapes, 10-16 = disk.
"B" J	Starting at block J.
K	Skip K files.

Must use CR or RETURN key to end the list or listing routine will go to the end of the unit and print BAD READ.

M&M: Error Report

BAD READ

CK

This is an early data list program which will also tabulate the number of data points and total time, and print any discrepancy with those figures in the header area.

Prompt:

ENTER SBN

CR or RETURN will stop the list. Another CR at the prompt will exit.

Data Transmission Commands

Automatic Calling Unit (ACU) is port 13.

Anderson-Jacobson Model (AJM) modem is port 10.

CALLWCC

This uses the ACU and calls the WCC TELEX, 79941.

M&M: Transmission Reports

SUCCESS	Established carrier.
FAILED	Did not establish carrier.
EXPIRED	ACU not responding.

HANGWCC

This puts the ACU "on hook."

TMITWCC

This uses ACU and port 13.

TMIT

This uses AJM and port 10.

Otherwise, these are exactly the same, providing keyboard connection through the PDP-12 to TELEX, and a routine to send a data file to the CDC-6000.

CTRL/T Prompts:

TRANSMIT WHICH? "A" N

"A" If data is on backup unit.

N SBN.

SENT Prints when transmission of that file is complete. Typing any keys while the file is being sent will stop the sending of that file.

CTRL/E Exits and prints.

THE END

M&M: Transmission Process

BAD READ Could not read data.

PORT BUSY Output port is being used.

TMITAUX "13"

Similar to preceding TMIT's but does not tie up the Utility Area. "13" is used with CALLWCC. Default is port 10.

CTRL/T Prompts: "T" "A" N

CTRL/T to re-enter.

"T"	Use tape.
"A"	Backup unit, 5 or 15.
N	SBN.
CR	Start sending, echoes as Line Feed.
Carriage Return	Signals Sent.
B	Signals bad data read.

Again, typing suspends the transmission.

M&M: Loader

WRONG	Already loaded or Bad Read

TMITAUU

Unloads TMITAUX

Video Feedback Commands

SDED

OROS

OROS 2 These are the display scope routines which are used to relay selected information back to a teacher in a classroom.

Prompts:

No answer to BOX 1 turns the display off.

M&M: Loader

-R	Bad Read.

SPEDU Unloads SPED.

M&M: Loader

NOT LOADED

Utility Commands

TDUMP "A" U B

TDUMPN "A" U B B . . .

These list a block of 256 12-bit words in octal.

"A"	Auxiliary TTY.
U	Unit 0-1 = LINCtapes, 10-16 = disk.
B B . . .	List of block numbers.

Type CR or RETURN to stop the listing.

M&M: TDUMP Routine

<u>BAD READ</u> of data blocks.

CONSOLE X

This command will determine which teleprinter will be used to enter commands and print error messages.

X - "L" A 30 or "A" SR 33, tests first character only.

DIAL

Bootstrap to the DIAL-MS operating system.

<u>DITRMA Commands</u>

DITRMA

This coder training program uses an interactive setup similar to CATTS.

M&M: Loader

<u>-R</u>	Bad read of program.

Plus CATTS messages.

DITRMAU

To change DITRMA parameters, first unload it with this command.

DTIMER N

This command accepts the number of seconds the "CODE INITIATED" message is displayed while waiting for all the coders to enter before the "TIME" code list display will be generated and the VTR stops.

N Number of seconds to wait for all coders.

DSTOP Usual mode of stopping at each disagreement.

DNSTOP Special mode which does not stop the tape.

DLIST N

This is a special data list program with a format of printing any code lists on the same line.

N DITRMA data file SBN to be listed.

Data File Formats: PDP 12

DITRMA Data File Format.

Word Number	Item	
0	Data (Year-1975/Month/Day)	
1	Time at end of session (Hours * 60 + Minutes)	
2	Master box number	
3	Number of data points	
4	Total time in seconds	
5-37	(Empty)	
40	0	
	N	Number of boxes in a code list
	time	Identification codes till 0011
	data	Control code
	.	
	.	
	.	
	time	
	7761	Master box 0011 control code
	time	
	data	Agreed-upon data
	.	
	.	
	.	
	time	
	7776	Code list flag
	E	Error type[1]
	Data	Code list N boxes long
	0	
	Data	Data = 7776 means nothing was entered.
	.	
	.	
	.	
	time	
	data	Consensus code entered to start tape.
	.	
	.	
	.	
	time	
	7771	Master box 0099 control code
	time	
	7777	End-of-data mark.

[1] All numbers are in octal notation.
0 - Disagreement
1-6 - Coder Error
10 - Time
11-16 - Too far ahead

CATTS Data File Format.

Word Number	Item	
0	Date (Year - 1975/Month/Day)	
1	Time of end of session (Hour * 60 + Minutes)	
2	Box number	
3	Number of data points	
4	Total time in seconds	
5 - 37	(Empty)	
40	time	
	data	
	time	
	data	
	.	
	.	
	.	
	time	
	7761	0011 control code
	time	Data that follows is packed hexadecimally
	data	
	.	
	.	
	.	
	time	
	7770	0088 control code
	time	Data that follows is packed decimally
	data	
	.	
	.	
	.	
	time	
	7771	0099 control code
	time	
	7777	end-of-data mark

All numbers are in octal notation.