AN EVENT-TRACE STUDY OF THE PERFORMANCE OF THE I/O SUBSYSTEM FOR A CDC CYBER 172 COMPUTER

Carlo LoCicero

A Thesis

in

The Department

of

Computer Science

Presented in Partial Fulfillment of the Requirements
for the degree of Master of Computer Science at
Concordia University
Montreal, Quebec, Canada

March, 1980

ABSTRACT

AN EVENT-TRACE STUDY OF THE PERFORMANCE OF THE I/O SUBSYSTEM FOR A CDC CYBER 172 COMPUTER

Carlo LoCicero

The performance of a CDC CYBER 172 computer system has been observed using an event-driven probe embedded in the operating system (NOS 1.1). General measures of the performance of the various components of the system have been obtained. The disk subsystem has been carefully instrumented. It has been demonstrated that missed revolutions occur very infrequently, and that significant opportunity exist for seek/read-write overlap in the operation of the disk subsystem.

To Renee and Paul

for making it all worth-while

"Nothing in the world can take the place of PERSISTENCE ....

Talent will not;
nothing is more common than unsuccessful men with talent...

Genius will not;
unrewarded genius is almost a proverb.

Education will not;
the world is full of educated derelicts.

Persistence and determination alone are omnipotent.
The slogan PRESS ON has solved, and always will solve, the problems of the human race."

Anonymous

ACKNOWLEDGMENT

I would like to express appreciation to my advisor, Dr. J.W. Atwood, who suggested the thesis topic, for his advice, discussion and confidence.

I wish to thank my wife, Renee, for her patience and encouragement during these past years.

I would like to thank the Software Team of the Computer Center, who did the upgrade modifications of the event trace system, from KRONOS 2.0 to NOS 1.1, without which we could not have achieved our study.

This study was supported in part by the National Research Council of Canada, through the Department of Computer Science at Concordia University.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION

In recent years, with the tremendous growth in size and complexity of computer systems and consequently their cost, increasing attention has been paid to their efficiency of operation. There are two aspects of what can be termed computer system performance evaluation. Firstly there is performance prediction and measurement. Secondly there is performance improvement, commonly referred to as system tuning. Performance evaluation may involve either estimating a proposed system capacity to handle a projected workload or the measurement of an existing system to determine its efficiency (PELTV 75).

Trace-driven modeling is a technique which combines measurement and simulation for the purpose of evaluating and predicting the performance of a system. All good analysis consists of both theoretical analysis, which may be done through a mathematical or simulation model, and empirical measurement, which is designed to test the theory (SCHERS 72a). The use of the combination of analysis and measurement for the solution of problems is the classical scientific method (CANTH 68) and the foundation for the technique of trace-driven modeling.

This study of the performance of a large-scale computer system (the CDC CYBER 172 computer under the NOS operating system in use at the Concordia University Computer Center) is

based on empirical data gathered while the system was in operation. The evaluation was done by use of an event-driven probe imbedded in the operating system. The results of the probe were written to disk and disk these files later analyzed.

C H A P T E R I

PERFORMANCE EVALUATION

1.1 Early Measurements and Evaluations

When the comparative efficiencies of computer systems started becoming more important to the users, measurements were made. These measurements usually took the form of "job times". In going over old records, it is interesting to note that these times were expressed in tenths of hours, obtained from the clock on the wall. Stop watches were also used to obtain basic information on job performance. By the mid-fifties, two primary techniques of measurement had evolved: the program-addressable clock and the technique of counting entries into program routines, or counting the number of passes through a program loop. There is another programmed technique which is used in measurement: the instruction trace. The origin of this technique, however, was for debugging and not especially for measurement. Within a few years these techniques evolved into the basis for today's programmed measurement techniques (DRUMM 73).

The program-addressable clock is available today in many forms. The precision and accuracy have gone from fractions of a minute to microseconds or sometimes, nanoseconds and the implementation may be external to the main computing system or

internal to it. It may be called by many different names but the basic facility is still the same (DRUMM 73).

While computers have been generally operational since the late forties, systems evaluation techniques have been lagging by at least a decade. In the early days, systems were designed with a fairly precise objective in mind. Designers and engineers had essentially one designed criterion: make it as fast as possible with acceptable reliability. Sometimes, the reliability criterion was put by the wayside in the quest for faster and faster technology. Early methods of evaluating computers were well oriented to the intended use of the system. Scientific systems were judged by their ability to perform arithmetic. Commercial systems were judged by their ability to process records (DRUMM 73).

1.1.1 Performance Evaluation

Computer and operating systems have grown in complexity since those days when the time that an add instruction took was considered a major factor in the evaluation of a computer (LUCAH 71). With the current complexity of systems, the data needed to measure the system has grown in volume and the measurements themselves have become very sophisticated (HAWTP 74). In recent years the computational speed of computers has increased to the point where the factor which often determines their performance is their I/O capacity.

There are three primary purposes for evaluating a computer system:

- to decide which computer system to select from systems in existence,
- to project the performance of a system that does not yet exist,
- to measure the performance of an existing system and forecast the impact of changes in the system (LUCAH 71).

Our study is concerned with the last: measuring the performance of an existing system.

Why measure an existing system? Well, many different goals can be achieved:

- reduction of the cost of DP center operations,
- increase in throughput,
- reduction of computing job turnaround time,
- rational evaluation of suggested changes in hardware and software (AUERP 74).

Our goal is concerned with the last: suggested changes in hardware and software.

Why it is necessary to change the hardware configuration of a computing installation? As was said in the preceeding section, too often the reliability criterion has been put aside and no care has been taken, in a higher level view of a system, as to how these pieces can be put together in a way that they can work together in a structured hierarchical system instead of working as disconnected sets. Bauer (BAUEF

73) expresses it well when he says: "it comes from the fact that people are forced to live with machines that they do not want. They have not constructed them; they simply receive them and have to make the best out of it. ... I hope one day software engineering considerations will dictate how machines are to be built and then to be used."

1.1.2 Measurement Techniques

There are two classes of techniques for the acquisition of data for evaluation purposes that are finding highly efficient use. One is software measurement and the other is hardware measurement. In the general case almost all data acquisition may be performed with software techniques. The principal reasons for utilizing hardware techniques are: ease of use, ease of installation, removal of overhead, and the ability to obtain data in a way which does not interfere with the workload being processed by the host system. On the other hand, there are many attributes which are much more easily obtained by software techniques than by hardware techniques. These attributes would be such items as job name identification, data set identification, origin of requests for facilities and other data dependent information. In the general case, a hardware technique will more easily provide information as to "what" happened in the system, and a software technique will more easily provide information as to "why" something happened in the system (DRUMM 73).

Hardware techniques generally sense electronic signals in the host system which when decoded by the hardware monitor circuitry provide the instrument with the capability to determine what is going on inside the system (DRUMM 73). Hardware monitors usually gather a prodigious amount of data in a very short time or reduce the data by counting events. Sometimes this information is produced at such a fast rate that the machine on which the system is running must be slowed down or stopped in order for the monitor to be able to record all the information (SHERS 72a).

Software techniques have the attribute of being inserted some way into the normal flow of programmed procedures to obtain the required information. A software monitor is a program that collects data on computer performance. It generally operates as a user program which observes the operation of the operating system and other user programs by interrupting normal processing at periodic or aperiodic intervals, examining control blocks and/or machine registers, and recording appropriate data (AUERP 74).

Most programmed measurement techniques are based on either an "intercept" concept or a "sampling" concept. The intercept basis is one in which additional coding is inserted at key points in the system control program and sometimes the problem program. The sampling basis is one in which at a certain time or when some other criteria for sampling are met, an added program takes control to scan the status of activity within

the system (DRUMM 73).

In the sampling techniques, as said before, at a regular or non-regular interval a sampling routine receives control of the system for scanning activity. This sampling routine then notes the time and scans the system to determine the status of events. Typical events of interest to the sampling routine would be the following information just before a sample is taken (DRUMM 73):

- CPU active or inactive,
- Location of the instruction being executed,
- Location of the data being referenced by the instruction,
- The instruction itself,
- The operating status of channels, control units and devices,
- The locations of the currently operating channel programs,
- The origin or destination of the data concerned with I/O.

All the above information is obtained at the time of the sample. The set of data collected is sometimes known as a system "snapshot". This data will be used in analytic or simulation programs which are based on sampled-data techniques.

1.1.2.1 Trace-Driven Modeling

A way to use monitors which has already proved useful and promises to become even more so is in connection with simulation to produce what has been called a "Trace-Driven

Model" (SHERS 72b).

Trace-driven modeling of computer system could not begin until techniques for inserting event-driven ("intercept") software probes in operating systems were developed (SHERS 73). The principal advantages of trace-driven model simulations are credibility, workload characterization and ease of comprehension. The credibility of the trace-driven model is a result of the accuracy of the validation, the reproducibility of the results, and the homogeneity of the trace data. The main advantage that trace-driven modeling has over other performance evaluation techniques is credibility (SHERS 73). The disadvantages of event-driven probes are: they are hard to maintain, they are not portable, and they generate a great deal of data. Programming an event-driven probe is very difficult and extensive knowledge of the operating system and the machine is required. The probe is frequently not portable from one version of the operating system to the next one, much less from one machine to another. Trace-driven modeling has three distinct facets:

- the trace data must be collected by a probe in an operating system;
- the trace data must be processed to prepare it for the model;
- the processed trace data is used as input to a simulation program constructed from a conceptual model of a system.

The series of operations that together constitute the

technique of trace-driven modeling are illustrated in fig. 1.1.2.1-1 (SHERS 73).

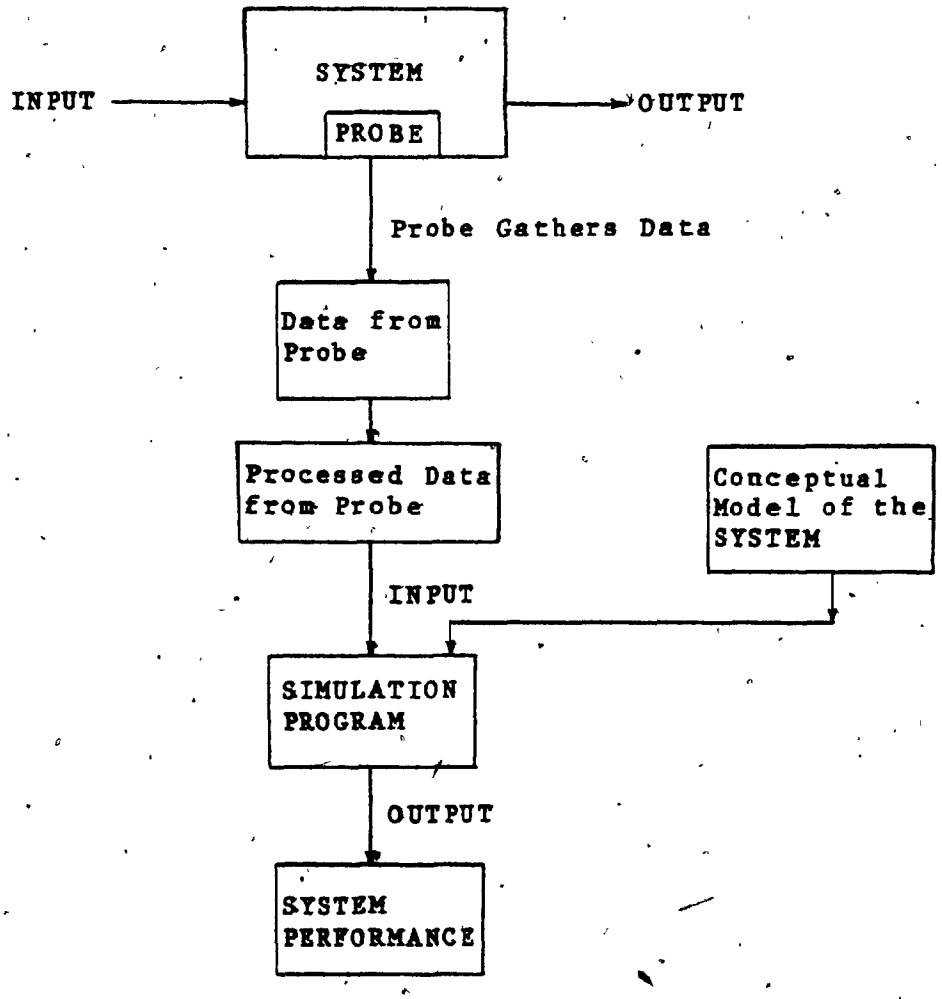


Fig. 1.1.2.1-1 Trace-Driven Modeling

1.1.3 Previous Work on Computer Systems Performance

One of the early papers on performance measures is that of Hutchinson (HUTCG 65) where he describes a macroscopic, job-shop-like, simulation of a computer center, the center operations, and some performance measures. He found that improvements to the system could be achived by scheduling the jobs according to their priority-class. This was one step ahead in performance evaluation.

Two years later, Calingeart (CALIP 67) published an article concerned with system performance evaluation where he makes the point: "Existing tools for evaluating performance of CPU's and of stand-alone, disk-oriented, batch-processing systems are not yet wholly adequate. How can we evaluate multiprogramming and multiprocessing if we cannot evaluate the performance of one task by one processor? ... Little has been published, and even less of value, on the subject of the goals and tools of performance evaluation. The merits of different measures of performance, the significance of various experiments, the validity of assorted evaluation techniques are of concern to the entire industry."

In the same year, Nielsen (NIELN 67) published another paper on system performance where he describes various types of simulation techniques. Cited: "It has already been demonstrated that system performance simulations can be used effectively for this purpose -- if appropriate models can be

developed. Thus, the third generation offers an exiting challenge". We can see that people were worrying about the future of performance evaluation with the impact of the new and complex technology.

One of the earliest reported projects is the study of the Compatible Time Sharing System (CTSS) performed by Scherr (SCHEA 67) at M.I.T. This study used an event-recording addition to the operating system to gather data about user interactions with the system and about system responses to user actions. Scherr also developed a simulation model of the system and used the gathered data as both input to this model and as a means of verifying the results of the simulation. He further used simulation to predict system performance when changes were made in the scheduling algorithms.

The developers of the GECOS II and GECOS III operating systems for the GE 625/35 computers made use of a data-gathering facility designed into these systems to evaluate system performance while they were being developed. These efforts are discussed in papers by Cantrell and Ellison (CANTH 68) and Campbell and Heffner (CAMPD 68).

In Shemer and Heying's (SHEMJ 69) study of a system designed for both batch and terminal users, the system response to an increased number of users was modelled. Their model was then validated by probing the system to determine if the response function was, in fact, as they had modelled it to

be. Response time to on-line demands and percentage of CPU time available for batch jobs were monitored.

At that time we should say as Campbell said: "Today, there is perhaps no single techniques more in vogue than simulation".

Trace-driven system modeling was first defined by Cheng (CHENP 69) in a paper in which it is referred to as a lesser-known modelling technique. Cheng defines a trace-driven model as a data-flow structure, in which data, recorded on an operating system, flows among system components, queues build up, and contention patterns establish themselves. Trace-driven modeling was used to study gross system performance as various system configuration changes were made but Cheng did not report any details of his study.

The two following years were almost tutorial ones. Several articles about simulation performance and modeling were published.

MacDougall (MACDM 70) published an article on computer system simulation. It was a tutorial paper describing the construction of a basic simulation model for a disk-based multiprogrammed computer system. This paper is frequently referenced. It was required reading for anyone involved in computer system simulation and computer performance measurement. In addition, a fully annotated bibliography provides references for more in-depth research.

Hellerman (HELLH 70) published an article assessing the performance of some simple idealized input/output. The objective of the paper was to gain insight into bounds on throughput performance and to give equations useful for checking computer simulation.

Lucas (LUCAH 71) is an often-referenced categorization of efforts in the area of computer performance evaluation and monitoring which performs a much needed function. In an area where a new methodology is advancing, Lucas attempts to classify the goals and techniques applied. For example, he cites three general purposes of system evaluation: selection evaluation, performance projection, performance monitoring. Then he tabulates the various techniques on their suitability in achieving these goals. The techniques are also explained in brief. The approach taken to unify such a broad body of knowledge is appropriate.

The dissertation by Schwetman (SCHWH 70) is a performance evaluation of a CDC 6600. In it he studies disk channel availability, pool peripheral processor availability, disk half-track scattering, control point availability, and central memory availability. Schwetman cites the need for more definitive studies of computer systems. Data is needed about how large-scale systems actually respond to the demands placed on them; data also is needed about the nature of the demand. Schwetman gives three reasons why empirical data about systems in operation is needed:

- to increase the knowledge of system structure and component interactions in order to increase productivity and reliability;
- to verify and calibrate the results of theoretical methods of analysis;
- to evaluate the cost-effectiveness of changes to the system.

He also introduces the concept of "balance of resources" (B) which is a numerical figure corresponding to the ratio of the holding time (H) of a resource to the sum of the waiting time (W) plus holding time (H)

$$B = H / (W + H).$$

This ratio is an expression of the response plus demand. The ideal balance of utilization is 1 (the waiting time is zero in that case). As the waiting time increases, the holding time remaining constant, the balance figure goes to zero. The use of the balance of utilization determines where there are bottlenecks in the system.

In 1971, the TRW Business Data Processing Center was operating two IBM 360/65 large-scale computer systems (AUERP 75). Management became concerned about DP costs and raised the question: Could DP costs be reduced without substantially impairing performance by doing the processing on a single, somewhat larger, computer system, the IBM 370/155? After the performance measurement had been taken with hardware and sotware monitors, the conclusion was reached that with some

additional core and a second card reader, the one-computer configuration was satisfactory.

Noe and Nutt (NOE,J 72) were interested in validating a trace-driven model of a CDC 6400. They did this by analyzing the dayfile provided by the operating system. The dayfile is a recording of such statistics as program size and job turnaround time. The figures Noe and Nutt believed important were: job turnaround time, memory usage, and job dwell time in the input and output queues. The trace-driven model that they constructed was not used to predict the effects of modifications to the system, but studies evaluating the effects of changes to the configuration and the nature of jobs were planned when the study was reported.

About 1972, trace-driven modelling became more popular and more widespread applications were reported.

Trace-driven studies by Sherman, Browne and Baskett (SHERS 72b) were used to study the effect of CPU scheduling algorithms and deadlock prevention algorithms on the performance of the CDC 6600 at the University of Texas. The scheduling methods tested included the best possible and worst possible methods, Round-Robin, FCFS, and dynamic predictors. The relative and absolute performance of these scheduling methods were given. It was concluded that a successful CPU scheduling method must be preemptive, and must prevent a given job from holding the CPU for too long a period. The number of

algorithms examined illustrated the versatility of the trace-driven models used.

A later paper (the Ph.D. dissertation) by Sherman (SHERS 72a), the most referenced in trace-driven modeling, extended some of the results of the first studies and modeled a large number of hardware and software changes of interest to the CDC 6600 installation at the University of Texas. Trace-driven modeling was used to analyze the effect of increasing the speed of the CPU, peripheral processors, and disks.

A year later, Sherman (SHERS 73) published another paper in which he presents a review of the simulations of computer systems which have used trace-driven modeling as the simulation tool. The advantages and disadvantages of trace-driven modeling are examined and a comparison to the conventional distribution-driven simulation technique is made.

Gotlieb and Metzger (GOTLC 73) reported a trace-driven model study of a simple processing system (IBM 370/165 OS) which receives batches of jobs from a number of remote terminals. Cited: "The trace-driven modeling approach avoided much of the statistical analysis needed for traditional simulation models. It is suggested that this technique should be more widely used for system tuning studies. ... The utility of trace-driven modeling as a system tuning tool has been demonstrated on a non-trivial problem." The conclusion obtained was that the use of the "preferential with delay t"

[PREFD(t)] policy with a delay limit of three-quarters of a minute would maintain current system response characteristics while reducing the impact of HSJS (High Speed Job Stream) processing on other system activities, primarily through data channel and auxiliary device loading.

Beretvas (BERET 73) describes the principles of a system-independent tracing facility, the Generalized Trace Facility (GTF). This paper defines the technique of obtaining traces characterizing application programs independent of the system on which the tracing was performed. The paper proposes a scheme for obtaining such a system-independent trace and then proceeds to describe an implementation of tracing technology which approximates the desired goal. Cited: "If modeling of operating systems becomes more widespread -- as it would be desirable -- then in fact the modeling capabilities must be built concurently with the design of the system; and an appropriate tracing mechanism of the type described must become an integral (even though probably optional) part of the operating system."

At Boeing Computer Services, Bedoll (BEDOR 74a) has done some performance measurement on a KRONOS operating system running a CDC 6600 and a CYBER 74 linked together. The measurements taken were divided into two major areas: Low-level measurements, such as channel utilization, PP usage, and high-level measurements which involved system job mix, job swapping and scheduling. After analysis of the results, he

found a disadvantage of the KRONOS priority-driven scheduler which is that it is insensitive to job size. Since large jobs (100-130K) age the same as small jobs but require more machine resources to run, it is possible for a few large interactive jobs to seriously affect users doing editing and other small core tasks. They implemented a scheduler modification which allows the scheduler to pick the smallest job (rather than the largest) in the event of a priority tie, and further allows a priority window to be set in which the smallest job of a group of jobs is chosen.

Schwetman (SCHWH 74) at Purdue University has reported a trace-driven model for the MACE-5 operating system running on a CDC 6500. The initial version of the trace-driven model focused on the following resorces: the PPUs, the CPU and the CM/CP resource. After some initial experimentation with this first version, some modifications were suggested. The most obvious one was the installation of a round-robin service discipline for the CPU resource with a slice-time of twenty msec. Other changes were made which contributed to the realism of the model. One such change was directed at the system tasks being modeled.

The M.Sc. thesis of Hawthorn (HAWTP 74) presents a performance evaluation of a CDC 6600 under the KRONOS 2.0 (level 5) operating system. The evaluation was done by use of an event-driven probe imbedded in the operating system. The study has been conducted over three major classes: channel

statistics, peripheral processor statistics, and job statistics. The main problem with the system was the wait time for the channels. The problem was caused by two factors: missing revolutions on disk accesses (on CDC 808 disks only) and long rollout/rollin times.

Conti (CONTD 74), at the Naval Surface Weapons Center, published a paper which presents a realistic survey of performance evaluation. This paper is restricted to CDC machines, however, it gives an idea of what is going on in performance measurements. Cited: "The performance measurement field, in spite of all its claims of 'improved throughput', 'better CPU utilization', 'more bang-for-the-buck', etc., etc., is not without its share of problems. ... Generally an installation begins a performance measurement effort for one or more of the following reasons: (1) there is a known component which is degrading the performance of the system and the installation wants to know why; (2) there are one or more unknown components degrading the system; (3) it is not known if the system is even degraded but the installation simply wants to see if it can improve the general performance of the system; (4) the installation merely wants to claim 'we have monitored' -- for whatever political or nonpolitical reasons. ... Numerous papers (including this one!) have been written on how to collect performance measurement data but very few discuss how to interpret it. ... There is still very little understanding as to how the various system components (e.g.,

the CPU, the PP's) interact with each other. 'Tuning' one such variable may degrade another. System modeling has provided some answers, however determining component relationships is still a black art."

A year later, at the same Center, Wilson (WILSC 75) describes the use of a software monitor to compare the performance of their early NOLOS (modified SCOPE 3.2) with their new installation of the KRONOS operating system. In their latest measurements, they found that KRONOS was doing a rather poor job of keeping the memory well filled.

The paper written by Boyse and Warn (BOYSJ 75) describes a cyclic queuing model which is both easy to understand and easy to use. This tutorial paper leads the reader through the development and use of an easily understood analytic model. They have introduced a queueing model and shown its application to computer performance prediction by means of a case study.

In 1976, Anderson and Browne (ANDEJ 76) published a paper about a new technique, System Process Graphs (SPG), applicable to system performance evaluation analysis. The paper defines a graph structured modeling system which is designed to support performance analysis studies on complex computer systems. The modeling system is applied to a complex multi-processor multiprogrammed computer system, a CDC 6600 driven by the UT-2 operating system. The authors suggest that this

methodology provides a framework for system models for complex structured processes of all types.

Lindsay (LINDD 76), in the same year, published a paper about measurements done on a KRONOS operating system. The measurements have been taken by hardware and software monitors. CPU utilization, disk and tape I/O, ECS and disk rollout terminal response time, CM lockout, and exchange-jumps, were the points of interest. His conclusions were: disk and tape channels well balanced; inefficient ECS rollout, one hundred times slower than it would be if implemented as a CM/ECS direct transfer; an execessive average of exchange-jumps rate was also found. His figures show that 40 per cent of the real time is spent in rolling jobs in and out of CM; it appears to be a very large amount of wasted time. The solution presented by Lindsay, CM/ECS direct transfer, is now an option in standard NOS systems. The high exchange-jump rate was also surprising, for a machine in which CPU interrupts are supposed to be negligible (as opposed to IBM), they have increased at a tremendous rate. Lindsay proposed that some system tuning effort should be done urgently.

Because workload and job mixes change, and because new insight into how computers operate can occur, performance measurement is a continuing and neverending process (AUERP 74).

1.2 Improving Disk I/O Performance

It is well known that efficient I/O management is crucial to the successful operation of modern multiprogramming systems in which a so-called virtual memory is provided to the users. Particular concern with most large systems of this type is in obtaining maximum performance from magnetic drum or disk systems operating in the role of auxiliary memory systems (COFFE 69). This need is even more crucial in systems (such as the CDC Cyber) that do not provide virtual memory, but must rely on efficient rollin/rollout mechanisms.

In today's computers the central processing speed is so high that even if faster CPU's were available, the throughput would not be improved. There is no purpose in having the fastest processor in the world and in being unable to keep it busy (efficiently, not only doing swapping, as some particular installations have demonstrated) because of the slowness of the I/O peripherals. But it is necessary to be conscious that increasing the speed of a line printer to be as fast as the central processor would be a game lost from the first move. Instead, it is desirable to find a balance among CPU power, memory size, and peripherals, which maximizes effective parallelism, and thereby achieves maximum performance with a given hardware investment. One of the aims of this study is to measure the activity in the system, with specific emphasis on the input/output to disk, in order to assess the performance of the system, and then suggest improvements.

The architecture of the third generation systems differs from that of its predecessors in several aspects. The most significant advances have occurred in the areas of memory protection, paging techniques, terminals, and mass storage devices. The latter perhaps has been the area of most spectacular development because of the capacity of auxiliary storage attainable by these systems and the speed with which this storage can be accessed. Nevertheless, these devices create the greatest problems in large systems because congestion is most often encountered at the I/O channels interfacing the rotating memories. Unfortunately, the congestion at the data channel is not due entirely to the productive transmission of data; the data channel is also seized during the search for a record on a track. This delay is referred to as the rotational latency and it contributes to the utilization of the data channel. However, since this seizure of the channel is nonproductive, it is desirable to design auxiliary storage systems which reduce this effect and thereby improve throughput. (This is the intent of the Block-multiplexer channel introduced with the IBM System/370 series.) Just as important as the proposal of such a design is the need to know quantitatively the performance of the system incorporating a particular design (ABATJ 69).

Much study has been and continues to be done on methods of improving disk I/O performance. The methods chosen usually fall into one of two categories:

- schemes for physically ordering the data on a disk pack. An example of this approach is found in section 2.2.5;
- schemes which attempt to organize the order in which the various I/O requests are serviced. These schemes are commonly referred to as scheduling algorithms.

The most currently used scheduling algorithms are listed in appendix A, with a short description and a reference to where a more detailed description may be found.

1.2.1 Previous Work on Disk Performance

Appendix A gives a summary description of all the disk scheduling algorithms found in this section.

Denning (DENNP 67) studies the significant increases in the performance of both disk and drum systems brought about by scheduling their usage. Scheduling means the sequencing of existing demands on the system according to the state of the system and the characteristics of the I/O devices. Denning evaluates two service strategies for movable-head disks [as well as service policies for drums, Shortest Access Time First (SATF)]. The first procedure is the SSTF policy. He correctly points out that such a policy does not guarantee service to all requests. In fact, the policy will tend to discriminate against requests to inner and outer tracks. The second policy discussed by Denning is called SCAN. No discrimination is made, this time, between a recent request

and one which has waited a long time, so a service request at an inner or outer track does not have to wait an excessive amount of time. Denning comes to the conclusion that SCAN is the best policy for use in disk scheduling.

In 1969, many people were interested in optimizing the throughput of disk and drums. The most popular publications are listed below.

In Coffman's (COFFE 69) study, magnetic drums in the role of auxiliary memories are studied in a context of modern multiprogramming systems featuring a paged environment. The basic aim with regard to the analysis is to extend the methodology so far developed for assessing the performance of I/O devices viewed as servers in a stochastic queueing systems. Cited: "direct approaches to the analysis of input/output systems appear to be a pressing need at the moment."

Frank's (FRANH 69) paper appears to be the only one, at that time, which considered the effects of the probability distribution of information stored on tracks. Cited: "If dynamic records are stored in a central portion of the disk, the average seek time can be slightly reduced. One way of locating a large number of dynamic records is to store all small records in central locations. Thus, if 80 per cent of the records can be stored in 25 per cent of the disk, we can approach a situation where references to records outside the

central group are infrequent." Frank also promotes the advantage of having two independent heads, with each head having access to all tracks. He concludes that these heads may be utilized in such a way that rotational latency is simultaneously reduced by a factor of two.

Abate and Dubner (ABATJ 69) published a paper where they analyze the performance of a head-per-track type auxiliary storage system in a real-time environment. Their study was directed toward a system design for a head-per-track type storage which attempts to always execute first that request for a data transfer which will incur the shortest latency time relative to all the possible waiting transactions. They conclude also, in comparison with Coffman's (COFFE 69) analysis for a paged environment, that a paged file is better by a factor of about two.

Slotnick (SLOTD 70) was interested in modifying the mechanism of the usual single read/write head. He considered his study applicable to problems where there is much data and little calculation or where the disk is to be used as a large associative memory, i.e., where the association quantity is broadcast from a central location and where in one disk revolution, the entire disk contents are scanned for a match. Cited: "I know of no such system being built. If this is really the case, it should be remedied."

1972 was the most prolific year for publications in scheduling disciplines. The following are the most referenced.

Teorey and Pinkerton (TEORT 72) provide a clear and succinct analysis of the FCFS, SSTF, and SCAN algorithms. They developed some variations (LOOK and C-LOOK) of Denning's (DENNP 67) algorithms and presented a small correction to his original work in their paper. They make the same assumptions as Denning, but now explicitly. Based on these assumptions, they derive expressions for the expected service time of an I/O request. They conclude that, under light loading conditions, the maximum performance was achieved by LOOK (a variation of SCAN) and, under heavy loading conditions, by C-LOOK.

Coffman, Klimko and Ryan (COFFE 72) formulate and analyze simple mathematical models of head motion in disk systems in which two different scanning policies are implemented (a modified SSTF named by him SCAN, and FSCAN). The principal conclusions are that his SCAN produces uniformly better response times than FSCAN but that his SCAN discriminates against requests for addresses at the extremities of the address space.

Minsky (MINSN 72) points to several latent potentialities of rotating memories and describes a method for utilizing them for a realization of "partial associativity". Here, Minsky

defines a new approach to disk subsystem architecture where two heads, one adjacent to the other (one reading and one writing head) with a constant distance in between, can do correction of a "data cell" by reading it with the reading head, correcting it by a processor; then, when the writing head gets to the data cell, the corrected information is written on it. Minsky suggests that this approach should be considered and some effort should be made to look at how a disk subsystem could be constructed to improve its performance.

Fuller (FULLS 72) defines a new scheduling algorithm called an "optimal drum scheduling discipline", which will be renamed as MTPT on a later paper. The algorithm considers the schedule for processing the records as a single-cycle permutation over the set of records. It first finds a permutation that minimizes the total latency time, regardless of the number of disjoint cycles in the permutation. The algorithm then transforms the unconstrained minimal latency time permutation into a single-cycle permutation while increasing the total latency time of the schedule by a minimal amount. The algorithm has not been implemented on any machine, however, it has been written in Algol.

Here, we could summarize the scheduling policies such as: FCFS performs no optimization; SSTF, SCAN, and N-SCAN tend to optimize on seek times; and the Eschenbach scheme optimizes on rotation as well as on seek times. All other policies were

derived from these.

Some experiments done at the University of Newcastle-upon-Tyne by Lynch (LYNCW 72) demonstrate that in the case of an IBM 360/67 running MTS (Michigan Terminal System), the arm did not have to move in 63 per cent of the disc accesses (MTS uses an FCFS policy). For the remaining 37 per cent of the disk accesses, the seek lasted an average of 30 msec. Upon inspection of the 2314 seek time characteristic (shown in TEORT 72), it is clear that the disks were not operating in the linear portion of the curve. This violates Teorey and Pinkerton's (TEORT 72) third assumption (the seek time characteristic is extrapolated so as to assume linearity). Indeed, if Lynch's data are to be accepted, the average seek length is of the order of five to ten cylinders. Lynch concludes that his study strongly suggests that locality of reference is a dominant factor in filing systems and that data references cluster together, making a multiplicity of disk arms desirable while disk arm movement becomes relatively less important. Lynch recognized that one set of experiments based on a particular environment should not be regarded as conclusive evidence and suggested that further measurements should be performed to characterize disk arm activity for other real operating system environments. This study is the first to question the utility of scheduling algorithms for improvement of disk I/O performance.

An interesting application-oriented algorithm used for masking rotational latency has been suggested by Gold and Kuck (GOLDD 74). The authors propose to apply this policy to managing I/O for programs with a "predictable behaviour". Typical examples of programs which they consider to fall into this category are those which consist largely of matrix operations or programs involved in solving partial differential equations by iterative methods.

The MTPT scheduling discipline is compared to the SLTF discipline in Fuller's (FULLS 74) paper. This article investigates the application of the MTPT scheduling policy described in an earlier paper by the same author (FULLS 72) which at that time had not been implemented. The MTPT scheduling discipline is shown here to be a promising intra-cylinder disk scheduling policy. For heavy arrival rates, significant improvements over the SLTF discipline are consistently achieved and just as important, this improvement is relatively independent of the MTPT discipline used.

A study done at the University of Toronto (PELTV 75), shows that scheduling algorithms do not improve the disk performance as theoritical formulas have indicated. Cited: "One might expect that if one scheduling technique performs significantly better than others, this performance improvement should be measurable in the form of less arm activity. It will be shown that this is not always true". This study has found that no great differences were evident in applying three

well-known scheduling algorithms. This study used the GTF trace facility (BERET 73) supplied by IBM with some local modifications. Lynch's (LYNCW 72) study and this one found almost the same properties such as: if increases in the performance of disk I/O are desired, scheduling is not the place to look.

Another study, which suggests that file locality and a specific configuration of disk driver and controllers can improve performance, is by Telford (TELFD 75) at the Lockheed Missiles and Space Co. He found that better performance and smaller average queues result when there is at least one scratch device per controller or two scratches on a dual-controller; on the average, the stack request queues for a 7054/844 in his configuration were quite short.

Wilhelm's (WILHN 76) study shows that, under probable conditions, FCFS seek scheduling is superior to SSTF scheduling! This controversy verifies two previous studies, one by Lynch (LYNCW 72) and the second by Peltz (PELTV 75), and leave us with the question: is seek scheduling really necessary? Lynch's data cast serious doubts on the need for any technique other than FCFS since, at least in some systems, there is sufficient locality to disk accesses that the required seek motion is minor.

The choice of a seek scheduling strategy is dependent on the functional characteristics of the computer system in which

it is to be used so that an *a priori* decision could very possibly be incorrect. One should either study data taken from a comparable system already in operation to find estimates of such critical parameters or initially implement FCFS scheduling and change to a more complex discipline only if measurements show that such a change is warranted (WILHN 76).

1.3 Thesis Objective and Organization

This thesis has been inspired by three other studies:

- Peltz's (PELTV 75) thesis: his objectives were to obtain profiles of disk arm movement for three commonly used disk queueing algorithms. His central questions were, "how much do disk arms move?" and "what effect does changing the queueing algorithm have?".
- Lynch's (LYNCW 72) paper: his objectives were to bring attention to the fact that disk accesses are, in general, not randomly distributed on a disk.
- Hawthorn's (HAWTP 74) thesis: her thesis objectives were to present a performance evaluation for a CDC 6600 using an event-driven data gathering technique.

The aims of this study will be to verify Lynch's observation (as he suggested it); to show if seek overlap does exist and if it might be useful; to show if there are missed revolutions (as stated in Hawthorn's thesis), and to give a detailed performance evaluation of the system at Concordia with an accurate evaluation of the I/O subsystem. This study will not be carried to the point of a trace-driven model, but the data gathering method is based on trace-driven modelling as an ultimate goal.

To support these goals, Hawthorn's thesis has been used as a base. Her study reports on the overall performance of the operating system with a breakdown on channels, PPs and job

class types. This study modifies and extends Hawthorn's general system measures; it includes data on memory management and physical record unit distribution. To provide data on disk performance, a completely new section of the analysis program has been built, which gives information for each disk unit: average seek time vs. seek distance, seek time, seek distance, accessed cylinders, device wait time, and data transfer time.

The organization of the ensuing chapters is as follow: chapter II presents the description of the system measured (hardware and software) and the structure of the I/O subsystem; chapter III discusses how the measurements were done and the data reduced; chapter IV presents the results by means of histograms; chapter V presents, in a tabulated form, a summary of the experiments with a discussion and conclusions.

1.4 A Note

As in any technical field, systems programming has a great deal of technical jargon. Much of it is indigenous to the particular machine upon which the work is being carried out, in this case a CDC Cyber 172. This work assumes a general familiarity with CDC Cyber Series computers. To avoid losing the thread of a discussion by delving into what would be lengthly technical explanations of some points, certain details are omitted and explanations presented in only enough detail to substantiate the discussion. Further explanation may be obtained by consulting the appropriate appendices and the references listed in the bibliography.

C H A P T E R II

THE SYSTEM MEASURED

2.1 Hardware Architecture (CDC Cyber Series)

The CDC Cyber Series is architecturally similar to its predecessor, the CDC 6000 Series. Many descriptions of the CDC Cyber and CDC 6000 series exist and there in no point in repeating a lengthy discussion (see THORJ 70, REDDS 76, CDC 73a). Only the salient features of the system will be described here in order to show its general structure.

2.1.1 Central Processor Organization

Twenty-four registers are included in the Central Processor (CPU). Eight of these are assigned as operands or data (Xi), eight are assigned as index registers (Bi), and eight are assigned as address registers (Ai). Except for data registers which are 60-bit long, all registers are 18-bits long. The 24 operating registers, plus certain processor registers such as the program address (P) register, form a 16-word register set called an Exchange Package (EP). The CPU is well equipped for arithmetic calculation but it is not capable of explicitly communicating with its environment [that is, it has no Input/Output (I/O) or similar instruction in its instruction set].

2.1.2 Memory Organization

A Central Memory (CM) of between 32K (100,000 octal) and 131K (400,000 octal) 60-bit words is available to the system and the user programs. An optional Extended Core Storage (ECS) of between 125K and two million 60-bit words is available. Registers A0 and X0 are used to reference the ECS. A program in CM uses a contiguous amount of storage which is a multiple of 100 (octal) words in length. The block of CM assigned is defined by a starting address called the Reference Address (RA) and a word count Field Length (FL). A user program cannot access memory beyond its boundaries of RA and RA+FL. The CPU uses the RA in conjunction with the P register to convert program addresses to absolute addresses. If the program attempts to read or write beyond its boundaries, the CPU detects the error and aborts the job.

2.1.3 Peripheral Processor Organization

As noted before, the CPU has no I/O capability. The Peripheral Processors (PP or PPU), on the other hand, are designed to handle interaction with the environment through a number of I/O channels which may be accessed by any PP. Ten PPs are included in the Cyber Series (options are available which permit system configurations with as few as seven PPs or as many as twenty PPs).

Each PP has 4096 (10,000 octal) 12-bit words of private storage, and four registers (A, P, Q, K). All the PPs are identical. They operate independently and simultaneously as stored-program computers, executing a program located in their private storage unit. There is a separate instruction set for the PPs.

The PP can transfer data between peripheral devices and central memory and can supervise the operation of the devices. The PPs have the capability of establishing paths to I/O devices through peripheral channels. A PP can interrupt the operation of the CPU by means of an "exchange jump". When an exchange jump is issued, the CPU makes an exchange between the contents of its 24 registers and the contents of the EP which starts at a location in the CM specified by the PP. The EP specifies the new contents of the 24 central registers. Once the exchange is made, the CPU starts on the new program specified by the P register. A PP is also capable of monitoring the CPU by transferring the contents of the CPU program address register to one of its registers. The CPU can also initiate the exchange jump.

2.1.4 Input/Output Organization

Twelve channels are included in the Cyber Series, however options are available which permit system configuration with as few as nine channels or as many as twenty-four channels. All channels are 12-bits wide and each may be connected to one

or more external devices. Only one external equipment can utilize a channel at one time, but all channels can be simultaneously active. Each channel has a single register which holds the data word being transferred in or out, and two control flags which allow the PP to monitor the status of the data channels.

2.1.4.1 Mass Storage Devices

Some channels (one or more) are dedicated to the Mass Storage (MS) devices or disks, which are of primary concern in this study. Each CDC 844-21 disk holds more than 11 million 60-bit words of storage. The data rate is 46.4K 60-bit words per second; the access seek time is from 10 to 50 msec, the average seek time is 30 msec, and the average latency time is 8.3 msec. The CDC 844-41 disk, the newest double density disk, holds nearly 24 million 60-bit words of storage. CDC Model 881 or 883-60 Disk Packs are used with the 844-21/41 MS. The 881/883-60 consists of a stack of ten 14-inch diameter magnetic recording disks and two protective disks. Nineteen surfaces are used for data recording (only 18 are used by the operating system) and one is used for permanently-recorded positioning information.

2.1.5 Overall Configuration of the Machine Under Study

- A CDC Cyber 172 (see fig. 2.1.5-1);
- A Central Memory (CM) of 131K (400,000 octal) 60-bit words;
- Ten Peripheral Processors (PPs);
- Twelve channels, connected as follows:

  Channels 0, 1, 5, 6 and 7 are not used;

  Channels 2, 3 and 4 are connected to three subsystem controllers (7054) for the five CDC 844-21 and five CDC 844-41 MS devices.

  Channel 8 (10 octal) is the display channel for the continuous console display;

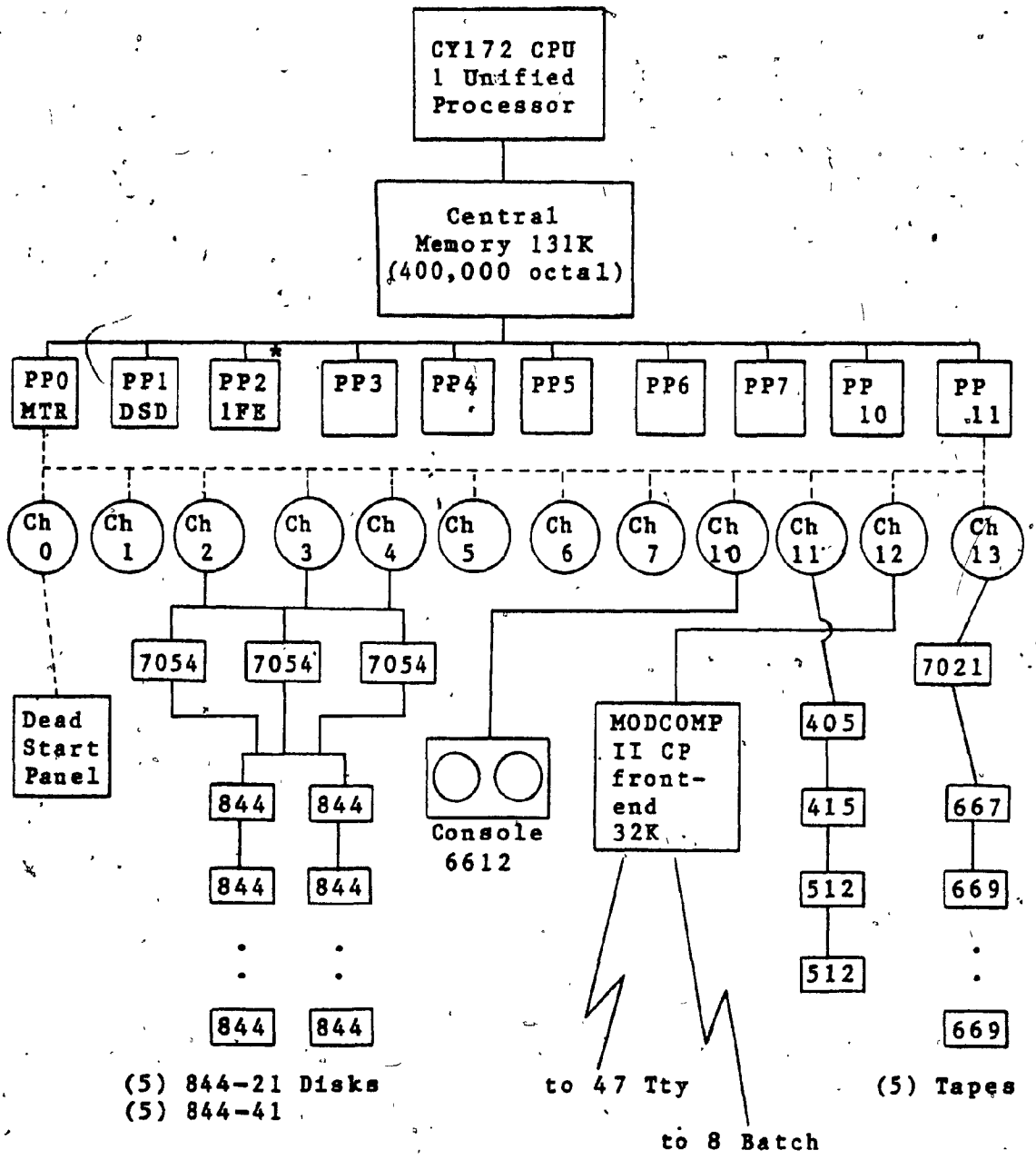
  Channel 9 (11 octal) handles the unit record equipment: two CDC 512 line printers, one CDC 405 card reader, and one CDC 415 card punch;

  Channel 10 (12 octal) is used for the MODCOMP II CP front-end computer. The MODCOMP II CP handles 47 asynchronous ports (teletype terminals), and 8 synchronous ports (batch terminals);

  Channel 11 (13 octal) is connected to the controller for the four CDC 669 nine-track tape drives and one CDC 667 seven-track tape drive.

The above configuration is not unique. It could be modified or changed according to the requirement or needs. The above-mentioned configuration was in use for regular production when our tests were run. The tests were run under controlled conditions, with a substantialy reduced

configuration. A summary description of the configuration used during the tests is shown in appendix D, as output of the "Preprocessor" program.



* This PP is a random choice within PP2-PP11.
This PP is dedicated when Export/Import is assigned to its own CP.

Fig. 2.1.5-1 CDC Cyber 172 Hardware Configuration

2.2 Operating System

In order to function, all the Control Data Cyber Series machines require an operating system. It is not within the scope of this paper to give a complete exposition of the Cyber Series operating systems. It should be noted that the operating system described here is one of many that might be conceived for the Cyber Series. Other systems may attempt different emphasis on the handling of jobs and resources (see CDC 73a, 74c, 74d, 75b, 76b, HARRM 67, KASPH 74a, 74b, ABELV 70, FRANW 74). The one described here is the one on which the study was done (that is, the NOS 1.1 operating system).

2.2.1 CM Organization and the Control Point

In the NOS operating system, the large central storage is divided into a user portion and a Central Memory Resident (CMR) system area. This system area, which occupies just under 21,000 (54,000 octal) words, contains allocation tables, routine and file directories, a small amount of system central processor code (most of the system executes in the PPs), copies of a number of key PP routines, and a set of job control blocks known as control points (CPs) (see fig. 2.2.1-1). Two elements are basic to the NOS operating system: Files and CPs.

A File is an organized collection of data known to the system by a given name. Data is organized in one or more

logical records and terminated by an end-of-information (EOI).

A CP is a pivotal area, occupying 128 words (200 octal) of CM, through which job execution is controlled and to which the resources for job execution are allocated. The CP may be thought of as the control element of an individual computer, and the entire set of CPs as a division of the hardware machine into a number of separate machines, each of which can execute an independent task.

One CP is allocated to various system overhead functions--storage movement, MS space allocation, etc. The remaining CPs can be assigned to active jobs, including the control of I/O devices such as card readers, line printers, remote batch stations, and the keyboard console.

A job is assigned to an active CP after it has been queued to a system MS device (usually a disk storage unit). The resources required for the execution of the job are allocated to the CP. These include CM space, CPU time, PP assistance, mountable equipment (EQ) (tapes, disk packs, etc.), MS space, and file pointers.

The resources are allocated to CPs through a monitor program (MTR) which runs in a dedicated PP (PP0). A second dedicated PP (PP1) runs a display program, DSD, that provides operator-system communication via a twin screen display-keyboard console.

The remaining PPs are pooled for I/O and job sequencing functions. Each contains a small resident executive containing communication, overlay loading, and MS driver subroutines. The pool PPs constitute one of the resources assigned to CPs by the monitor, and execute programs which communicate with the monitor through CM buffer areas called "registers".

The CP area, which occupies a fixed portion of CM, contains pointers relating to job status and the resources assigned to the job. Included in the CP area are a 72 (110 octal) word buffer used to contain the control statements supplied for the processing of the job, and a 16 (20 octal) word area for the EP.

The EP is used by the system monitor to control CPU allocation. When a job is at a CP waiting for the CPU, the EP area contains the register contents that are required to start or resume processing of the job. When the monitor performs an exchange jump for that CP, the registers are loaded from the CP area, and the CP area is loaded with an EP that the monitor uses to return control to the system when the job is interrrupted or terminated.

The rapid CPU switching capability provided by the EP operation works in conjunction with a relocation and limit register (see RA and FL in section 2.1.2) in each CPU to provide an efficient method of memory allocation. The

relocation registers in the CPU permit the assignment of a contiguous region of CM to a program which is totally isolated from any other area, and which can be moved rapidly to/from and within the user portion of CM.

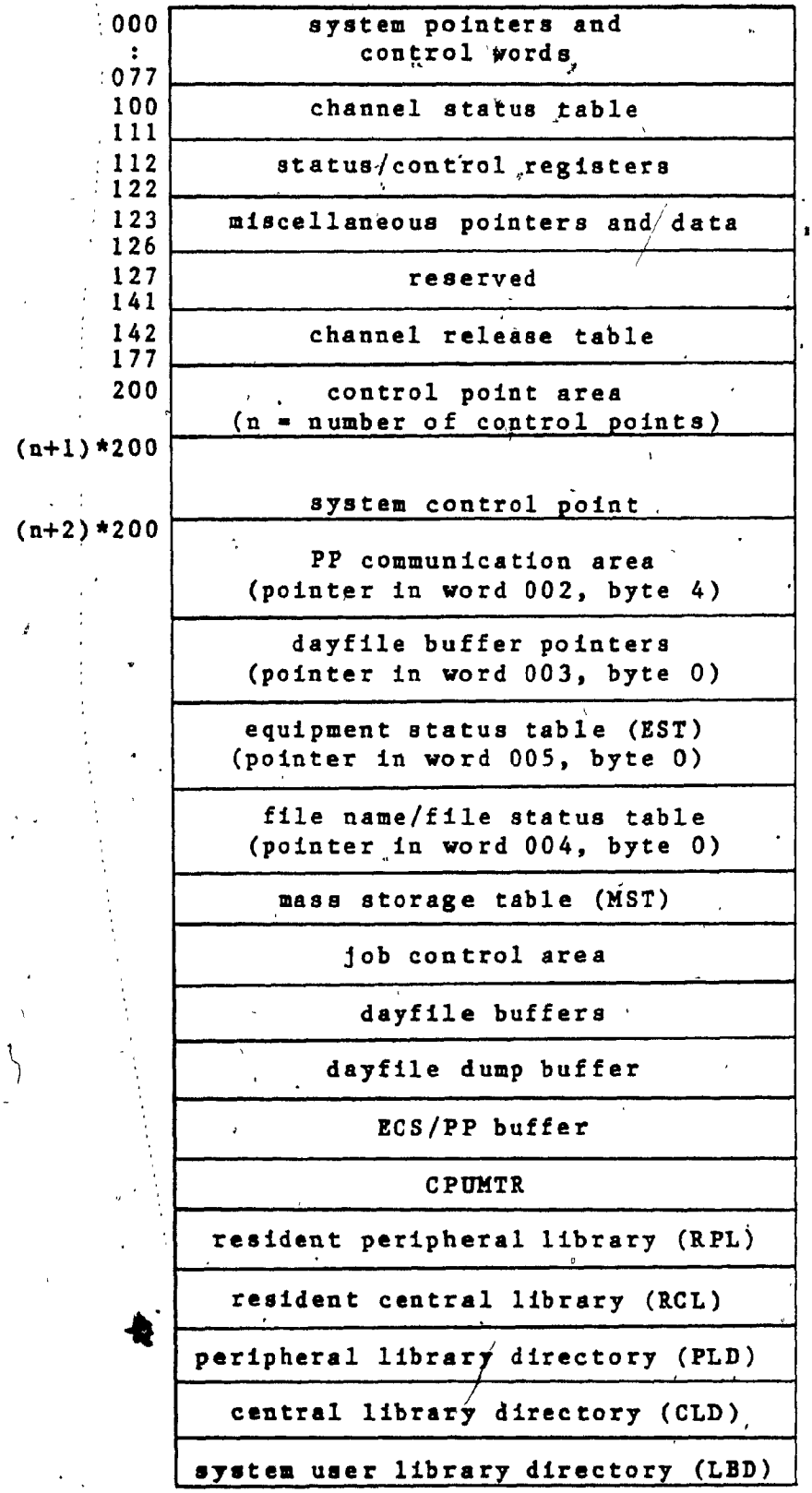


Fig. 2.2.1-1 Central Memory Resident Layout

2.2.2 CP and CPU Utilization

NOS typically runs a 16 (20 octal) CP configuration with one CP allocated to basic system functions as described in section 2.2.1. Three others are reserved for:

- TELEX and EI200 [CP15 (17 octal)], the routines that handle Input/Output from the remote teletype terminals and the remote job entry terminals;
- BATCHIO [CP14 (16 octal)], the routine that handles I/O from the local unit record equipment;
- MAGNET [CP13 (15 octal)], the system process that handles the calling of the PP routines for magnetic tape operation.

The remaining CPs (12), are used for the execution of user programs. This is a minimum number because the control points for TELEX, BATCHIO, and MAGNET are CPs that are not used by the associated process unless there is a need for them.

The CPU is cycled among active jobs on a round-robin basis. Each job at a CP which requires a CPU is allocated one for a 64 (100 octal) msec. time slice. The Exchange Jump operation keeps the switching overhead very low.

A job that issues an I/O request may retain the CPU for the full time slice and attempt to overlap its own computing with its I/O transfers. Alternatively it may give up the CPU for the duration of the I/O transfer, in this case we say that the job is in "Recall". The Recall may be automatic or

periodic. Auto-recall is used when a program requests I/O or other system action and cannot proceed until the request is completed. NOS will not return control until the specific request has been satisfied. Periodic recall can be used when the program is waiting for any one of several requests to be completed. The program will be activated periodically, so that it can determine which request has been satisfied and whether or not it can proceed.

2.2.3 Roll-in/Roll-out System

A feature of NOS is the ability of the system itself to suspend a job and free its CP. This process, called ROLL-OUT, consists of the copying of the complete job status, including CP data, to a MS file (or ECS if available). The CP thus freed can be assigned to another job. In the reverse process, called ROLL-IN, a rolled out job can be assigned to any available CP, the data in the file can be copied back into any available area in main memory, and the job can be resumed.

The scheduling mechanism in the NOS system is referred to as an AUTOROLL system. The basic component of the system is a job scheduler that can interrupt jobs and cause them to be rolled out from main memory to make room for other jobs that, at least temporarily, have higher priority. The queues from which the scheduler selects the jobs that are to be brought into memory, consist of input files and rollout files. The major function of the scheduler is to use the autoroll

mechanism to control the movement of jobs between the job queue and main memory in such a way as to provide for optimum utilization of system resources.

2.2.4 Monitors

In the NOS system, there are two separate monitors: Central Memory Monitor (CPUMTR) which controls CPU monitor mode execution and CPU scheduling, and Peripheral Processor Monitor (PPMTR or MTR) which is in general control of the system and operates in PP0. These two monitors work together, yet independently, to allow the system to run smoothly and effectively.

MTR is loaded into PP0 at dead start time and remains there for the duration of system execution. MTR performs the following functions:

- Processes certain PPU requests.
- Allocates central memory.
- Checks the CPU for arithmetic errors.
- Maintains the real-time clock.
- Checks the contents of (RA+1) of active central programs for system requests that are performed by MTR.
- Checks the status of active control points.

MTR is used in the assignment and release of all data channels, disk storage, and other I/O equipment. Communication between MTR and the PPs is accomplished through ten PP communication areas in CM. Each communication area

contains an input register (IR), an output register (OR) and six words for a Message Buffer (MB) area.

A PP idles in its resident program as long as IR is clear. CPUMTR enters a control word in IR of the selected PP communication area in order to call a transient PP program to that PP. The resident PP program of the selected PP senses the IR entry, locates the called program, and loads it into PP storage. After loading, the PP resident program then jumps to the beginning of the transient program. Following completion of the transient program, IR is cleared.

A PP may communicate a request to MTR by entering a value in its OR. A request too long for a single location is continued in the MB. MTR repeatedly scans the communication area; when a message is found, MTR jumps to a subroutine to process the request, then continues scanning.

CPUMTR is loaded in CMR and is entered at various places depending on what exchanged the CPUMTR. If a CP program wishes to call a PP program, it places the PP program name and up to two arguments in RA+1. [The first 100 (octal) words of a field length are a communications area for a program.] If Auto-recall is desired, bit 40 is set. If CPUMTR determines that RA+1 call should be assigned to a PP, CPUMTR will write the RA+1 word into the PP's IR in CMR. The name and any parameters in bits zero through 35 appear in the IR exactly as they did in RA+1. Parameters are passed from a CP program to

a PP program through this parameter field. The format for the PP communication area is shown in fig. 2.2.4-1.

For example: if the PP program Combined Input/Output (CIO) is called, CIO will find the relative address of the File Environment Table (FET) to be used in the operation by reading its IR. It can find the RA of the CP field length by reading the CP number from its IR, computing the address of the CP area, and reading the value of RA from the CP area. By adding the RA to the relative FET address, CIO obtains the absolute address of the start of the FET. CIO then reads the parameters for the I/O operation from the FET.

Figure 2.2.4-1 is an overview of system interaction showing both monitors as a controlling entity. PPs communicate to CPU and vice versa through monitor by means of the IR, OR and the RA+1 calls. Figure 2.2.4-2 shows the interaction between this monitor concept and PP resident using PP IR and OR.

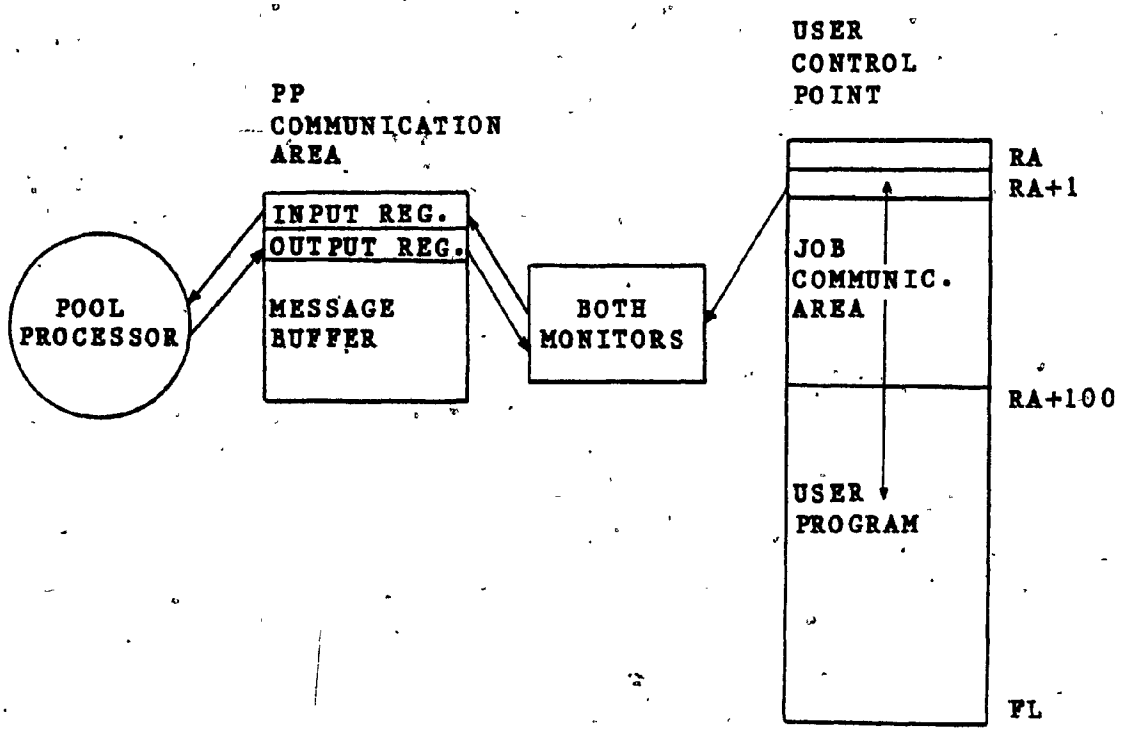


Fig. 2.2.4-1 Overview of System Interaction

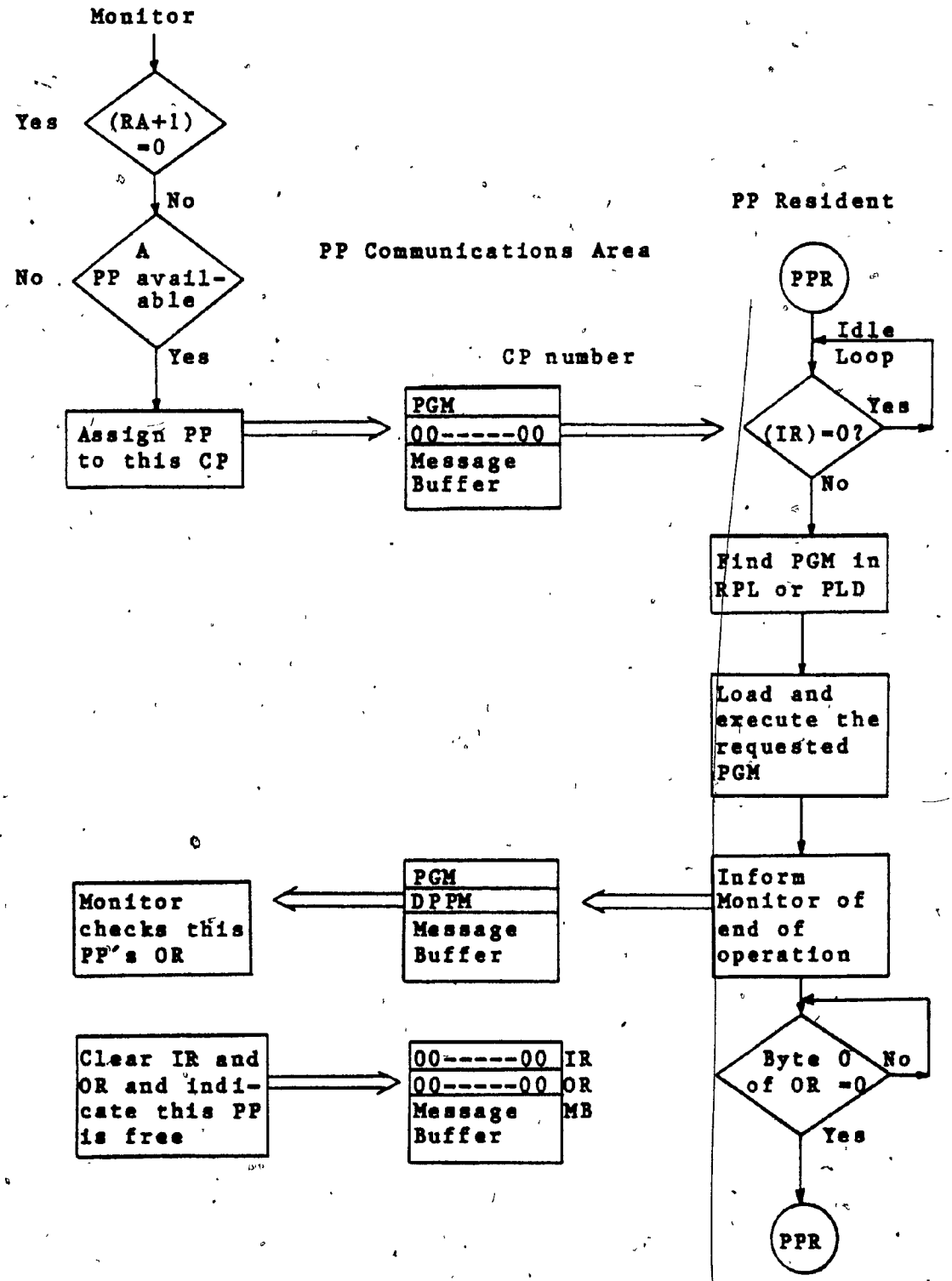


Fig. 2.2.4-2 System Interaction between Monitor and PPR

2.3 The Disk Subsystem (General Information)

As can be seen through the references in chapter one, a disk can have fixed heads (like a drum) or movable heads. The ones used in our study have movable heads, and are typical of disks currently used in most computing installations.

A single disk unit is composed of a series of "platters" of like radii, equally spaced along a common vertical axis. Both surfaces of each platter are coated with a magnetic substance, so that information may be stored on either side. The entire assembly rotates in time T. Each platter has a set of "tracks" (in some models a track is subdivided into several "sectors") situated near the outer edge so that each track is of maximum length. One can imagine that the Kth track of each platter is situated on the surface of a "cylinder"; thus, if the platter is W tracks wide, there are W concentric cylinders on which to store information. There is an assembly of movable "arms", equipped with read/write heads. There is one set of heads for each side of each platter, sufficient to read or write one track, so that all tracks on a specific cylinder may be read without moving the arm mechanism. Each time a track on a different cylinder is requested, it is necessary to position the arms. The operation of positioning the arms is known as a "seek". The mechanism for doing this is hydraulic, and is therefore inherently slow (DENNP 67).

2.3.1 The CDC Disk Storage Subsystem

The information described in this section has been extracted primarily from the two reference manuals CDC 76e and CDC 76f. For complete information about these disk subsystems, see these two references.

Data Organization-- information on a disk pack is divided into cylinders, tracks, and sectors. A cylinder consists of all the information accessible by all heads in one position. It includes one track for each recorded platter surface in the pack. A track consists of all the information accessible by one head in one position. A track is further divided into sectors. A sector is the smallest addressable area on a disk pack. There are twenty-four sectors on a track.

Access Time-- access time is the time a Disk Storage Unit (DSU) requires to locate the addressed sector ('a' time in fig. 2.3.4-1). Before transferring data, a DSU moves the heads to a cylinder and selects one of the data heads ('s' time). The selected head then transfers data as the appropriate sector passes under (or over) it ('r' time). Thus, total access time ('a' time) consists of the time required for the arm movement plus the time spent waiting for the appropriate sector to reach the selected head (rotational latency). Head select time is negligible.

Data Transfer Rate-- the transfer rates are the average rates at which from 2 to 456 sectors (physically there are, on

the same cylinder, 19 available surfaces and 24 sectors per surface) can be transferred. Both rates take into account subsystem overhead time required for addressing, error checking, etc. The 2:1 interlace transfer rate is half the 1:1 interlace rate since the 2:1 mode transfers only half the sectors on a track per disk revolution.

2.3.2 Disk Pack

The disk pack is the recording medium for the drive. The disk pack consists of 12 14-inch platters, center-mounted on a hub. The top and bottom platters are protective only. This leaves only 10 platters, with 20 recording surfaces. However, one surface is used for servo information, so there are 19 recording surfaces for data, see fig. 2.3.2-1. KRONOS/NOS uses only 18 surfaces because it divides the disk pack into two equal parts, 9 upper surfaces and 9 lower surfaces. The servo surface contains pre-recorded information that is used by the servo logic to position the heads to the desired cylinder, and to identify the sectors.

The 411 recording tracks on each platter (KRONOS/NOS uses only 408 tracks, to have some spares in case some get out of order) are grouped in a 2-inch band near the outer edge of the platter. Track 410 has a diameter of approximately 9 inches, while the diameter of track 0 is about 13 inches. The tracks are spaced about 0.005 inches apart.

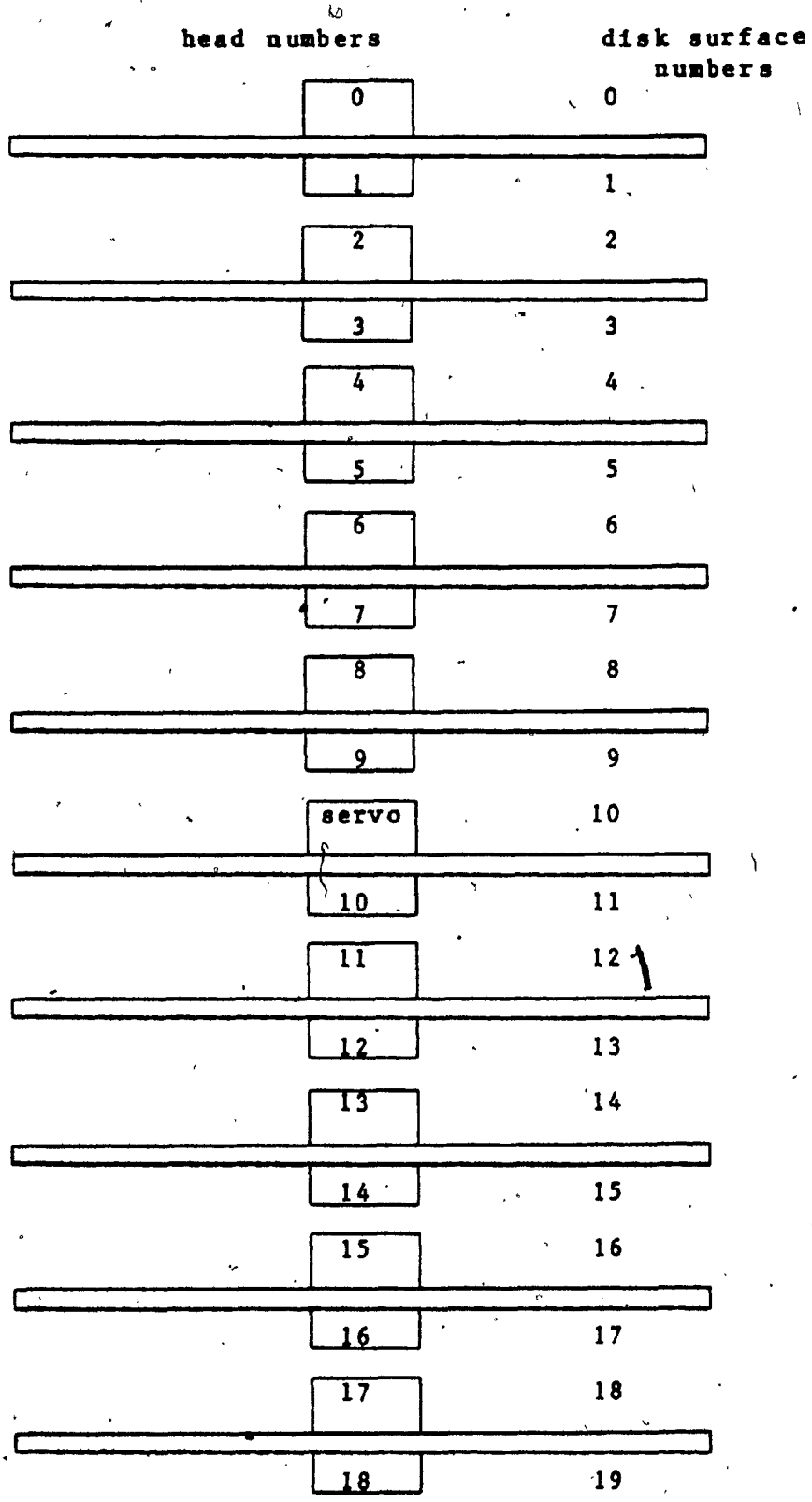


Fig. 2.3.2-1 Track Disk Layout

Track Format-- each track has an Index Mark as its starting point. The track is further subdivided into 24 sectors. Sector 00 is the first sector following the Index Mark. One track is operated upon by one read/write head. The heads are numbered from 0 to 18. These heads are positioned vertically with respect to each other. As a result, all 19 of the heads may be used without moving the actuator. Since any of the heads may be addressed at practically instantaneous rates, the recording medium may be thought of as a cylinder rather than as 19 discrete surfaces. Since the actuator may be positioned horizontally to any one of 411 rings or tracks, there are 411 cylinders. Any track may be addressed by seeking to the desired cylinder and by selecting one head. Only one head may be selected at a time.

Track format, sector control, and data record format are functions of the operating system. These functions are directly controlled by the controller (DSC). See section 2.3.5 for operating system specifications.

2.3.3 Access Times

Seek movement of one cylinder requires a time S(min), while a seek for the full width of the platter, (W-1) cylinders, requires a time S(max). Typically S(min)=10-25 msec and S(max)=50-130 msec; thus there is little difference between a seek of K tracks and a seek of (K+1) tracks. The bottleneck is getting the arms moving in the first place.

Once the arms have been positioned, any platter surface on the cylinder is addressable.

The disk waiting time parameters are shown in fig. 2.3.3-1. After a request is made, it waits a time W(q) until it is chosen for service. Once it has been chosen, a seek time 's' elapses. At the completion of the seek, an additional delay 'r' occurs while the desired data rotates into position. This time is called the rotational latency. Transmission then begins, and we assume it lasts for 't' seconds. The access time is thus

$$a = s + r$$

and the service time is

$$t(s) = a + t = s + r + t$$

The waiting time in the system is

$$W(f) = W(q) + t(s)$$

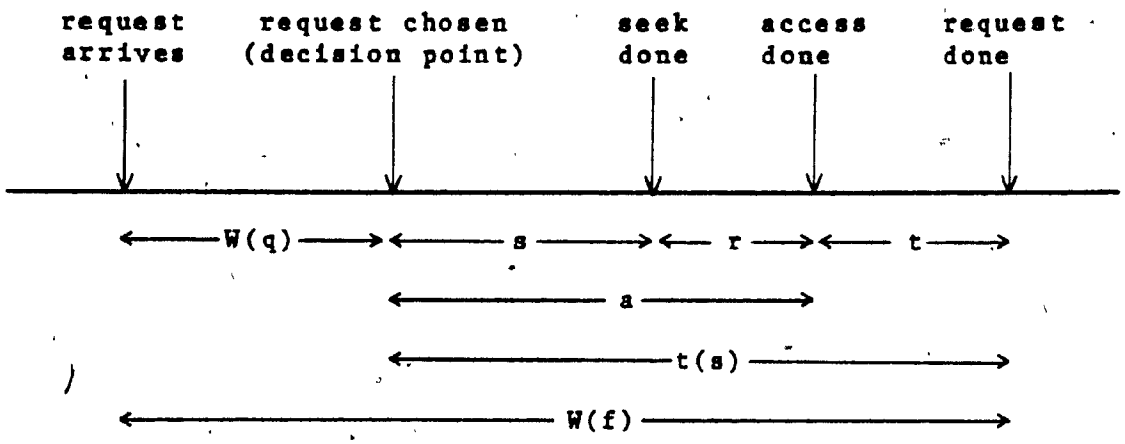


Fig. 2.3.3-1 Waiting Time Parameters for a Disk Access

This formula for waiting time is however theoretical, on a live system additional wait time is expected. For instance in our system W(q) will include:

- the waiting time for the PP,
- the time the PP waits for the channel,
- the time the PP waits for the EQ once it has the channel.

2.3.4 Interlacing

Interlacing-- depending upon PP selection, a subsystem can store data blocks either in consecutive sectors (1:1 interlace) or in alternate sectors (2:1 interlace). The 2:1 interlace allows a PP one sector time of nonsubsystem related program activity for each sector time of data transfer. When the 2:1 interlace is utilized, the cylinder processing order is as follows:

- from sector 0, track 0 (the first even sector on the

cylinder), the Disk Storage Controller (DSC) increments to sector 2, track 0, etc.

- from sector 22, track 18 (the last even sector on the cylinder), the DSC increments to sector 1, track 0 (the first odd sector on the cylinder) and processes odd sectors.
- after sector 23, track 18 (the last odd sector on the cylinder) has been processed, the PP must issue another seek command to move processing to another cylinder.

The 2:1 interlace is used to save one (or more) revolution each time two (or more) consecutive sectors are requested. Because, if they were linked one adjacent to the other, at the time the channel is transferring the first sector to the controller, the controller to the PP, and the PP to the CM, the disk has already passed through the next (adjacent) sector. Thus, if there would be a need for the information on that sector, it has just been missed. However the 2:1 interlace leaves enough time to transfer the first sector; then, if the next one (alternate) is desired, the read/write head will be just in the right position to read/write the next sector.

The KRONOS/NOS operating systems use the 2:1 interlace function in the disk driver routine (6DI). For more information see also sections 2.2 and 2.3.5.

2.3.5 System I/O (Mass Storage)

All active files residing on Rotating Mass Storage (RMS) are described by a File Environment Table (FET) and by a File Name Table (FNT). The FET is supplied by the user and resides within the job field length. The FET is shown in fig. 2.3.6-2. The FNT is supplied by the system and is used by system routines to coordinate user requests for I/O and file positioning. Only two other mass storage tables are involved with controlling I/O. These are the Equipment Status Table (EST) and the Mass Storage Table (MST). The linkage between these tables is simple and reduces system overhead to a minimum:

FNT --> EST --> MST

The FNT entry for a file consists of two CM words (fig. 2.3.5-1). The first word is the FNT word and contains the logical file name for the file. The second word is the File Status Table (FST) word and contains the file status, position, and equipment. The EST entry is one word which describes the device type, the channel(s), and a pointer to the MST entry for this device. The MST contains a complete description of the RMS device showing which tracks are in use and which are available.

The MST entry consists of a Track Reservation Table (TRT) and a 20 (octal) word header. The header words describe the

TRT, as well as providing other pertinent system information describing the device. The TRT provides information about each track available on the RMS device. Since TRT sizes vary depending on the device type, the MST entries vary in size accordingly. However each MST entry begins on a 10 (octal) word boundary so that they can be addressed with the 12-bit field in byte 4 of the EST entry. The MST entries are built at dead-start time by a routine named SET.

The TRT contains single-word entries that define up to four tracks, a link to another track, and control information. Bytes 0,1,2, and 3 represent a particular track, while byte 4 contains three 4-bit control settings as shown in fig. 2.3.5-1. The track link bytes either contain a pointer to the next track in the chain or indicate the EOI sector of the file.

Similar to the first and current track fields in FST, the track link byte contains a number which can be broken down to determine the word within the TRT and the byte within that word which is used to represent the track number, shown in fig. 2.3.5-1.

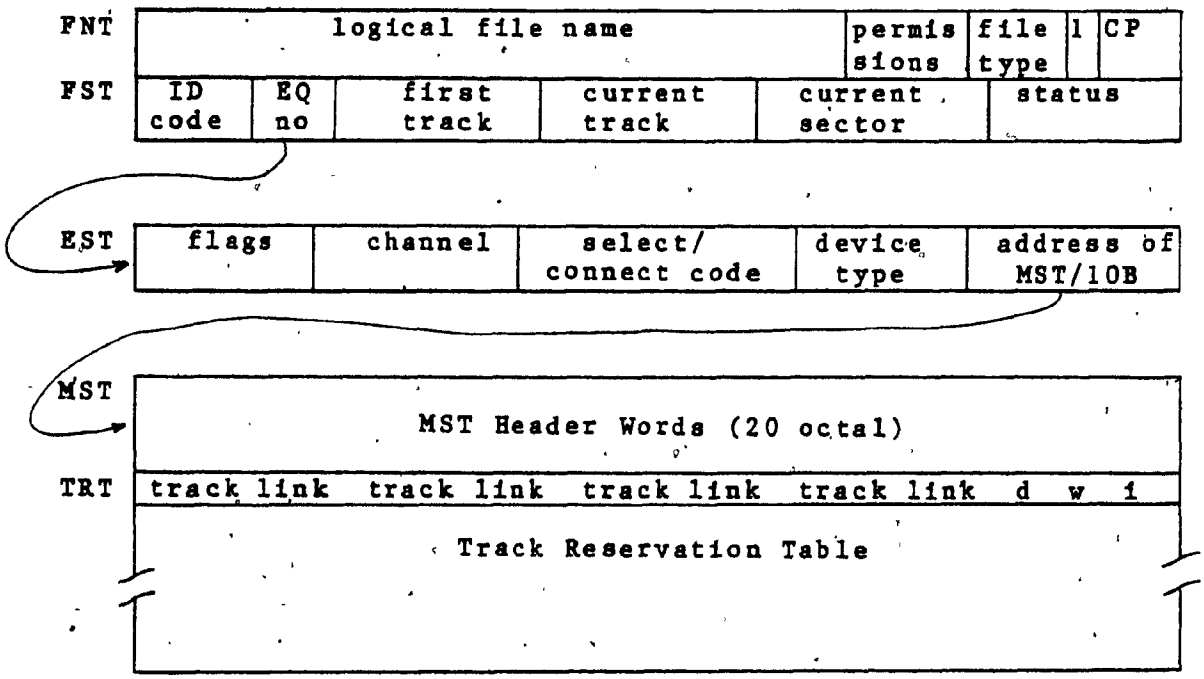
Every sector on a MS device, as seen by the user, contains up to 64 CM words (100 octal); however, the system always prefixes the sector with two header bytes (24 bits). These two header bytes contain file linkage and other information.

Let us now, describe the CDC 881 pack and TRT as used by KRONOS/NOS. The pack consists of 408 cylinders, 18 recording

surfaces or tracks (there are actually 19 recording surfaces, but KRONOS/NOS only uses an even number of them), and 24 sectors per track. KRONOS/NOS divides the pack into logical tracks; each cylinder contains four logical tracks as follows. The even sectors of physical tracks (recording surfaces) 0-8 (lower half) constitute a logical track, the even sectors of physical tracks 9-17 (upper half) constitute the next logical track. This is followed by the odd sectors for both the lower half and upper half of the pack. This means that each logical track consists of nine physical tracks of twelve sectors each (odd or even) for a total of 108 sectors of which only 107 (153 octal) are used by KRONOS/NOS. The last sector of each logical track is not used; it is used for positioning in most cases. The cylinder number 0-407 (0-627 octal) is the relative position of the TRT entry, within the TRT, containing the four logical tracks for the corresponding cylinder.

Examples:

Cylinder	Track	Sector	Byte Position in TRT Entry 4
43	2	5	2
43	2	4	0
43	9	0	1
43	12	3	3



d - each bit set indicates corresponding byte (0-3) is first track of a reserved file
w - track interlock bits
i - track reservation bits

Track Link Byte

z - 1 for next link in chain in bits 0-10
0 EOI sector number in bits 0-10
x - TRT word relative to word 0 of this TRT
y - byte within word x

Fig. 2.3.5-1 RMS Table Linkage

2.3.6 CIO Call with Operations Flow

Before we start talking about the CIO routines, we should introduce the concept of circular buffers which are used to perform all I/O operations by passing data between a user's program circular buffers and a peripheral device (mass storage, magnetic tape, etc.).

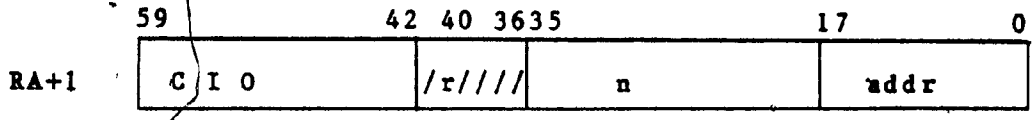
A circular buffer is a temporary CM storage area that contains data during I/O operations. It is called a circular buffer because routines that process I/O treat the first word of the buffer area as contiguous to the last word of the buffer area. The buffer parameters (FIRST, IN, OUT, and LIMIT) in the FET describe the circular buffer (refer to fig. 2.3.6-2).

- FIRST is the "first word address" of the buffer area.
- LIMIT is the "last word address plus one" of the buffer area.
- OUT is the next location to read to remove data from the circular buffer.
- IN is the next location to write data into the circular buffer.

- When IN=OUT, the buffer is empty. When IN=OUT-1, or IN=LIMIT-1 and OUT=FIRST, the buffer is full. That is, one location is left empty in a full buffer to distinguish an empty buffer from a full buffer. A buffer is normaly initialized with IN=OUT=FIRST, and IN and OUT circle the buffer as data is inserted and extracted.

In this section the operations flow is described, for a CIO call resulting from a Read instruction in a user program. An overall flowchart is given in fig. 2.3.6-3.

If an I/O operation is required, the CP program will put the mnemonic "CIO" and the address of the File Environment Table (FET) in RA+1 of the user Field Length, see fig. 2.3.6-1.



r Auto recall (20B), if desired
n Count for skip operations
addr Address of the FET

CIOP If CIO with auto-recall

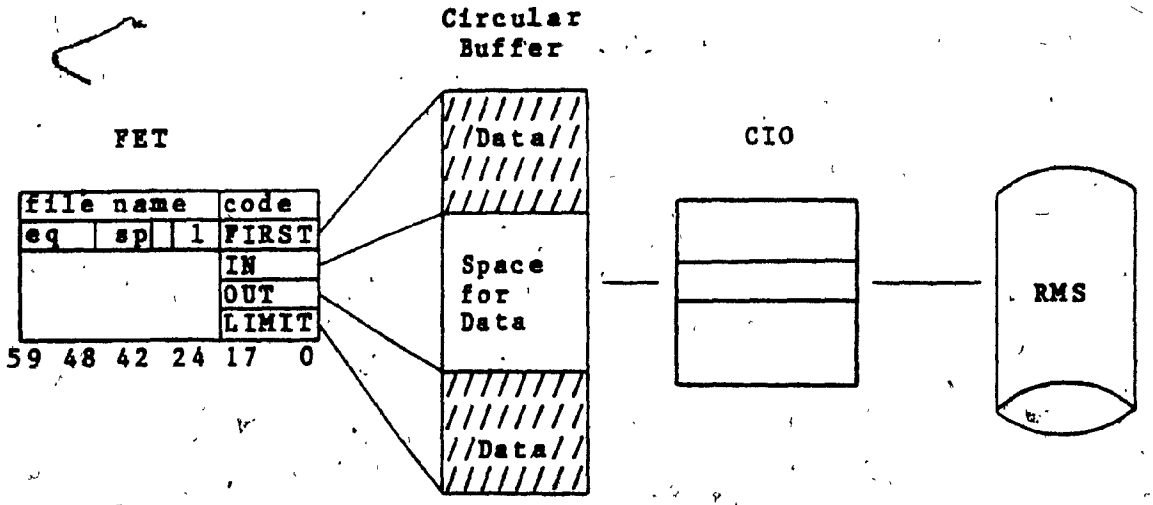
Fig. 2.3.6-1 RA+1 CIO Call

Then the program will execute an exchange jump to CPUMTR, where a routine will assign a PP to it. The request will be sent to the selected PP IR (fig. 2.3.6-1).

PPR will detect that its IR is no longer clear so it will load in the program name and set the CP number. Then the PPR will load the required program, which is CIO in the present case.

CIO clears the File Status Table (FST) address, reads the buffer status, as well as FIRST and LIMIT of the FET (fig.

2.3.6-2). It checks if the FET address is valid. If it is correct, the search in the Table Request code (TREQ) will be initialized. Then if it is a Read/Write operation, the File Name Table (FNT) pointer will be read at this time, and the FNT address will be checked for validity. If the FNT entry (after having read all of them) is not found, the new file entry is made in the FNT. CIO now stores the FNT entry in MB and calls CPUMTR (SBFM function) to set the status of this file to "busy". CIO picks up the EQ type from the Equipment Status Table (EST) entry, checks if it is an MS device, and finally checks if the request is legal on MS. If it is valid, CIO tests if it is a Read/Write function. If it is, CIO then reads IN and OUT of the FET table, and checks the buffer parameters LIMITS, FL, OUT, IN and FIRST (fig. 2.3.6-2). If the results of these tests are valid, CIO then searches in the table for the matching EQ. If the match is found, then it loads and executes the overlay that is required, 2CB (Read Mass Storage) in the present case.



1 FET length-5
sp special processing bit
eq equipment type

Fig. 2.3.6-2 User/CIO Interface

2CB tests if the track is equal to zero because, if it is different from zero, it means that the file is being used. Then, if it is a random file function requested, 2CB will load and execute the Preset routine (PRS) of the 2CF overlay (Position Mass Storage). Furthermore it will test (Buffer Status: BS) if it is a special MS read. If so, 2CB will load SMR (Special Mass storage Reads) from 2CC overlay and will initialize for read. 2CB now will Initialize the Mass Storage (IMS), Reserve a Channel (RCH), Position disk (POS), then do the following:

- adjust pointers,
- advance sector count by 1,
- Read Sector (RDS), this routine is in 6DI (it provides the capability to access the 7054/844-21 disk pack

system), which is included in the 2CF overlay, and it will generate the sequence of operations which follow:

- store the buffer address,
- decrement remaining sector count for the unit,
- advance the sector number,
- check if unit release required,
- check if consecutive physical sector (word LDAB in 6DI); if so, go to transfer the sector, else:
- call CPUMTR with LDAM (Load Address for Mass Storage Drivers) function, which will compute the physical address from the logical sector and track,
- now back to 6DI, read the physical address computed by LDAM, which is in the MB with some other information, like the status of the unit,
- test if the unit is busy. If so, call PPMTR with, this time, the DSWM (Driver Seek Wait) function, which will drop the channel reservation, set a flag and set up re-entry conditions to re-assign the channel on this or a later cycle (cycle of approximately 10 msec.) whenever both the channel is available and the selected unit is no longer reserved. Then it will give back control to 6DI,
- [illegible]d the seek function and the physical address to the channel,

- wait for channel response, if channel is already "on cylinder" wait for sector to be transferred or else call PPMTR, with DSWM, to drop the channel reservation and set up re-entry condition to re-assign the channel on this or a later cycle whenever both the channel is available and the logical unit is no longer reserved; then control is returned to the driver and the seek will be issued,
- back to 2CB, get the next sector from the link byte,
- store IN pointer and read OUT pointer from FET,
- test if it is at the End of File (EOF) or Record (EOR),

until the buffer is full or an EOR is encountered. At this point, it releases the channel (DCH) and updates FST table, as well as the track and sector number.

Then 2CB branches back to CIO in UFS (Update File Status). CIO checks if it is an EOR or EOF. If so, it sets the file status complete in FST entry, then in both cases (buffer full or EOR/F), it sets the status of the file to "not busy" in the FST. CIO clears the upper status bits in the FET, checks if any accounting is to be performed. If yes, it calls CPUMTR with the UADM function to update accounting and, in both cases, drops the PP and returns back to PPR routine where PPR will inform CPUMTR that it has finished. CPUMTR will then clear the IR and OR of that PP. Now the user program is ready to regain control and continue exexcution.

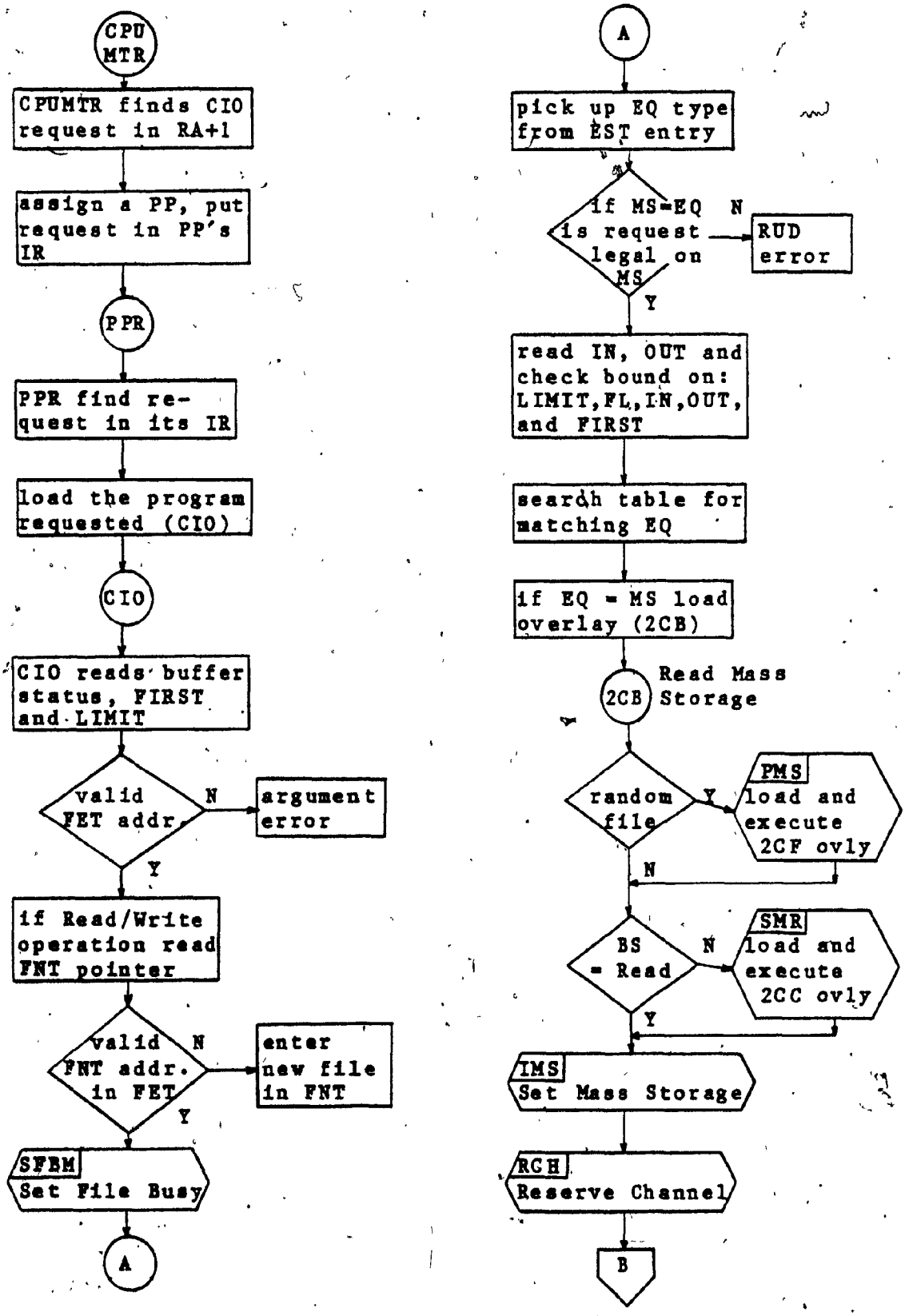


Fig. 2.3.6-3 Simplified CIO Logic for a Read Operation

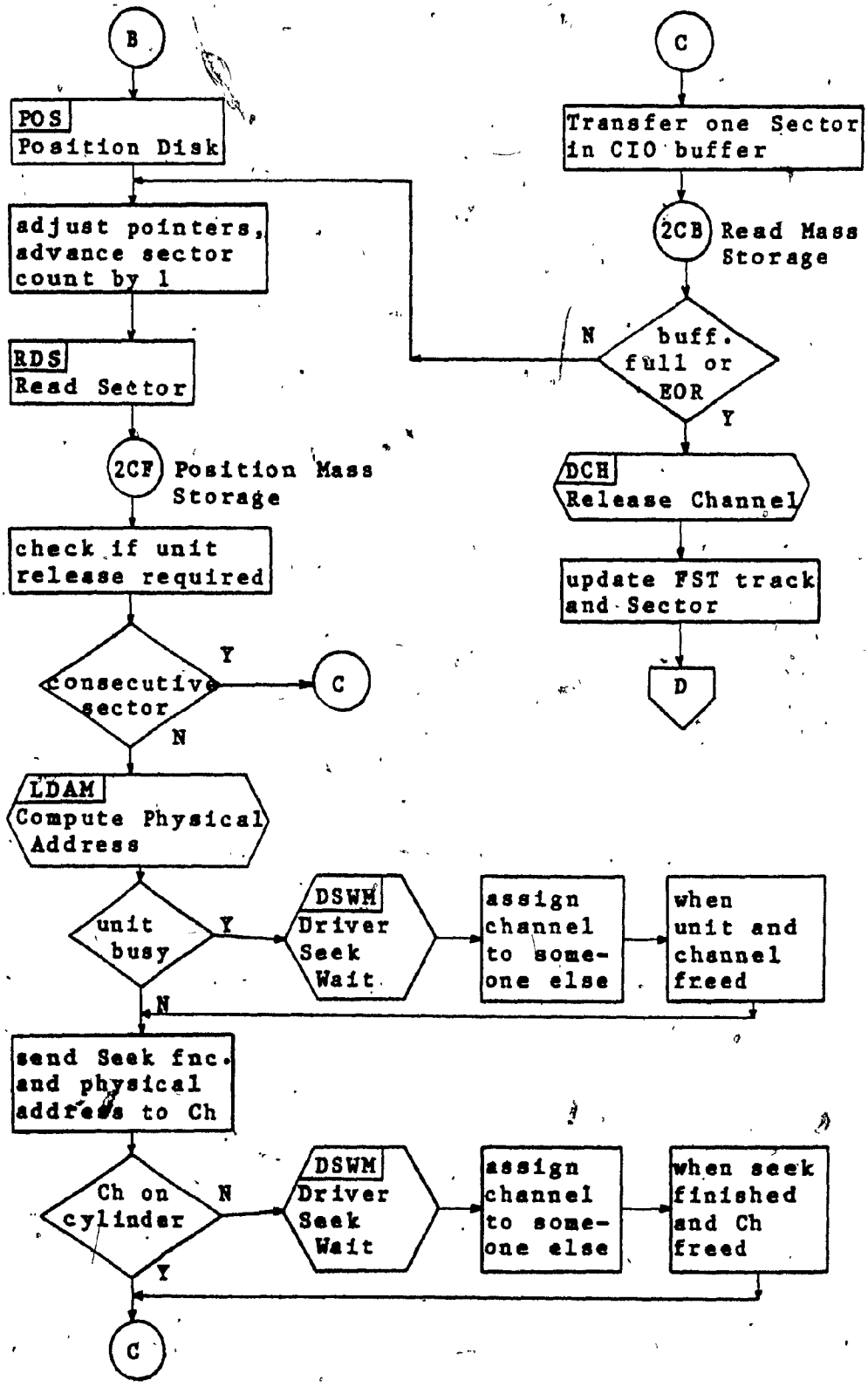
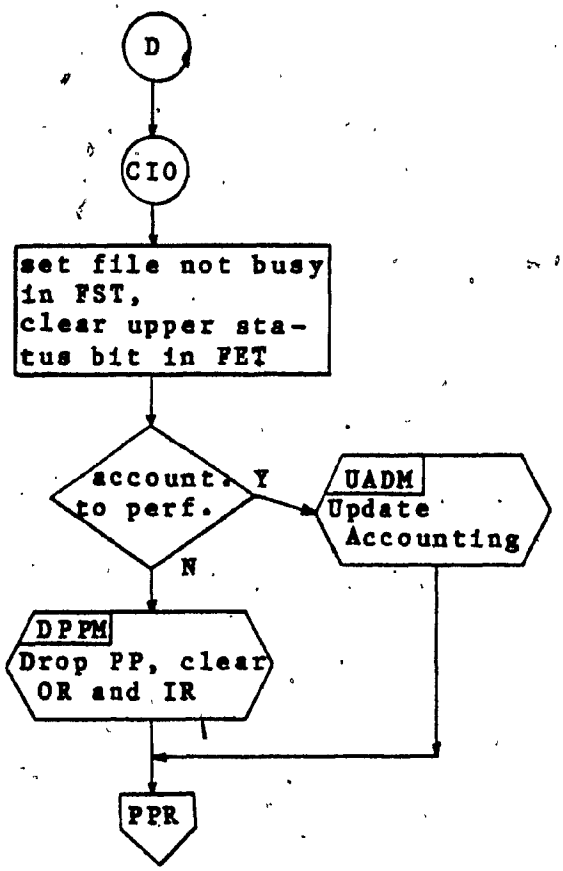
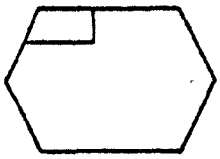


Fig. 2.3.6-3 Simplified CIO Logic (continued)



Monitor Functions



PP Predefined Routines

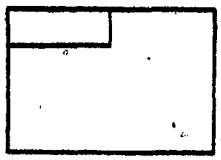


Fig. 2.3.6-3 Simplified CIO Logic (continued)

C H A P T E R III

MEASURING THE I/O SUBSYSTEM AND THE SYSTEM

3.1 Disk Performance

Some interest lies in deciding whether disk performance can be improved by planning the motion of the arm (DENNP 67). There are three factors which influence access time for a disk storage device. These factors are:

1 - Seek time,

2 - Rotational latency,

3 - Transfer rate.

Each of these factors is subject to physical design limitations. However, given an existing disk, it may be possible to control the way in which the disk is utilized in order to "reduce" the total access time for the disk (FRANH 69). Seeks are represented by determining the distance traversed by the heads in moving from the cylinder last accessed on the unit to the cylinder specified in the current request and computing the corresponding move time, data transfer times are computed on the basis of block lengths, and so on. It is important to reduce the average seek time as much as possible. Since the performance of the disk is limited by physical factors, such a reduction must depend on the way in which records are stored on the disk or on the way

in which records are accessed. In other words, we must alter the uniform record distribution and/or the random access properties of the disk (FRANH 69).

Now, consider the study done by Peltz (PELTV 75), over three well known scheduling algorithms, where it is concluded: "Subsequent analysis of the data collected revealed no significant difference in performance of three queueing techniques". Lynch's study (LYNCW 72) comes to the conclusion that disk performance can be improved by a better filing system. The latest work of Wilhelm (WILHN 76) raises the doubt: "Is seek scheduling really necessary?". If these studies [the only ones existing up to now (known to the author) about disk performance] are examined more closely, it may be seen that the problem, disk performance, does not come under scheduling algorithms but rather under operating system tuning, and that any reasonable queueing algorithm will suffice (PELTV 75) on a highly-tuned system, as a large scale system should be.

The aims of this thesis will then concern the general performance of the system and more particularly the disk performance. For the purposes of this study, the following charactestics of the system in general will be analyzed:

- The assignment and release of CPs, as indicated by the job initiation/job termination sequences, and/or by job rollin/rollout.
- Job elapsed time and job CPU time.

- Rollout activity counts and times.
- Central memory utilization, as indicated by changes in the relocation address and/or field length of a task.
- The assignment and release of PPs.
- The assignment and release of channels.
- The number of Physical Record Units (PRU) transferred during indvidual I/O operations.

In addition, the disk subsystem will be analyzed in detail, specifically:

- Disk arm movement (seek) times and distances.
- Data distribution on the disk sectors.
- Disk data transfer times.
- The potential for overlapping of the seek on one or more disk units with the data transfer on another unit.

The intent is to provide computer center management with information about bottlenecks in the system. Lynch's (LYNCW 72) observation, that disk accesses are not, in general, randomly distrubuted but follow a pattern according to the distribution in the system will be verified. [A similar result was observed by Peltz (PELTV 75), who reported that use of special scheduling algorithms for disk head movement made little difference in the performance of IBM's OS/MVT.] Hawthorn's (HAWTP 74) assumptions, about missed revolutions on disk accesses will also be examined, and an attempt will be made to determine whether seek overlap really does exist in the NOS operating system.

As has been discussed in the previous chapter, the scheduling algorithm for PP allocation in the NOS operating system is very trivial. That is, it is not even First Come First Served but somewhat like First-Encountered First Served. MTR services each PP in turn in a round-robin fashion. When a disk unit is requested and found to be busy, a flag is set in the MST. When MTR services that PP on the next cycle, the flag in the MST is checked and the unit is allocated if it is not busy. As an example: assuming that PP3 is presently using disk unit no.2, MTR scans, in a cyclic way, the contents of each PP output register (OR) and now it has reached PP7, which is also asking for the same unit; so, assuming that the unit is still busy, MTR will set the flag and go on to the next PP. While MTR is cycling, PP3 finishes, and releases disk unit 2. Now assume that MTR has reached PP1, which is by chance asking for unit 2. If the unit is free, then PP1 will get this unit, and PP7, which has asked earlier, will not get the unit on this cycle.

It may be seen that a particular PP asking for a particular unit could see his turn pass several times, depending on the frequency of that particular unit being accessed, and the length of time these accesses are taking. At a first glance, this scheduling algorithm seems inappropriate. However, it should not be forgotten that the MTR cycle time is very short (usually less than 10 milliseconds, in our experiments cycles of 1 to 2 milliseconds

were recorded) and the practical probability that all the PPs ask for the same unit simultaneously is very small, but still possible!

3.2 Implementation of the Measurements

Chapter 2 discussed the structure of the NOS operating system, and gave specific details concerning I/O actvity. In this section the modifications which were made to the operating system, the techniques used to obtain these measurements, and the program which analyzes and produces the histograms which were used to analyze the results, will be described. See fig. 3.2-1 for the system flowchart of the event-trace performance evaluation. To perform the desired measurements, it is necessary to have:

- an accurate clock,
- a monitor which records the time and the information pertinent to the measurements wanted,
- programs to analyze the data collected,
- a collection of programs which approximate a normal workload.

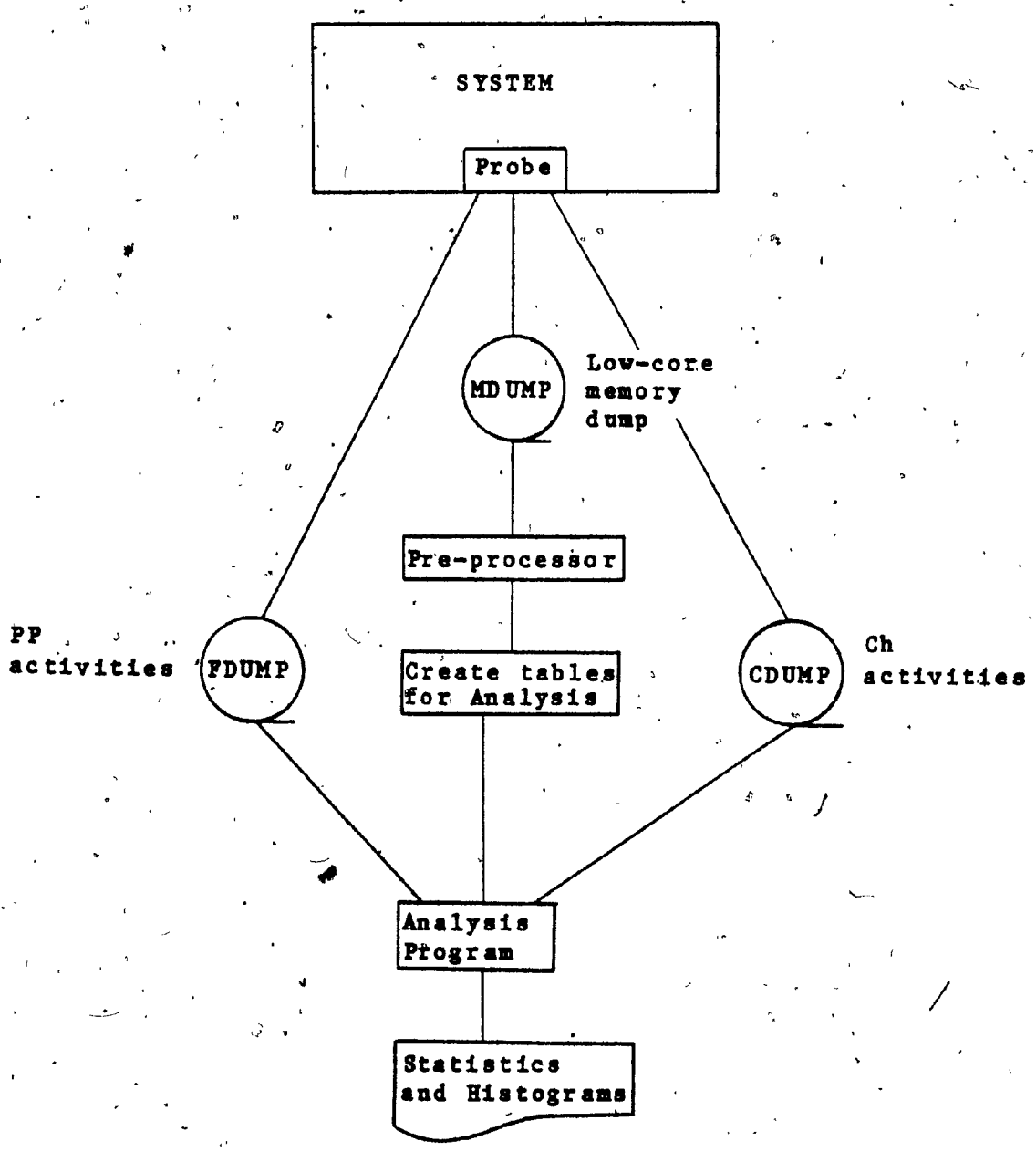


Fig. 3.2-1 System Flowchart of the Performance Evaluation

3.2.1 The Clock

The hardware real-time clock starts with power on and runs continously. It may be read by any peripheral processor with an input instruction from channel 14 (octal) which is not counted as a regular channel. This channel is separate from the data channels.

The clock period is 4096 microseconds. It is a 12-bit register that is advanced once each microsecond from 0000 to 7777 (octal). When it reaches 7777 (octal), it starts over at 0000. It must, therefore, be read at least every 4.096 milliseconds for accurate timing.

The Advance Clock (AVC) subroutine in MTR updates the clock. If the real-time clock has advanced at least 1 millisecond, AVC will update the real-time clock in CMR, RTCL word 106 in the channel status table, fig. 2.3.1-1.

3.2.2 The Software Monitor

The software monitor is composed of two different modules:

- SAMPLE-- the program which dumps the information gathered onto disk or tape;
- TRACE-- the probe which gathers the data, and which consists of modifications to various parts of the operating system.

These two modules, originally written by George Spix, have been received from the University of Houston. However, these two modules were modifications to the KRONOS Level 5 operating system and it was therefore necessary to upgrade the modifications to our actual NOS 1.1 operating system (a non-trivial job). Also, modifications have been added to gather more detailed data on I/O activity, because the original version was only collecting information for the overall operation of the system.

3.2.2.1 SAMPLE

The SAMPLE program is written in assembly language (COMPASS), and runs in about 3500 (7000 octal) 60-bit words. It resides at a particular CP and dumps, on disk or tape, the information gathered from the trace. The program starts when loaded by the operator and runs continously until the operator sets sense switch 1 "on" at the console. This program checks the buffer pointers, residing in the low-core memory, in the "installation area", each time-slice it is given control; it empties the buffers and remembers the new values of the pointers. The buffers used are for function activities (FBUF) and channel activities (CBUF) which are dumped on files FDUMP and CDUMP respectively, the files associated with the two buffers. At the beginning the program dumps, on a file named MDUMP, the low-core image up to and including the EST table. An overall flowchart of SAMPLE is shown in fig. 3.2.2.1-1.

The program and the description of the buffer and pointers used are given in appendix B.

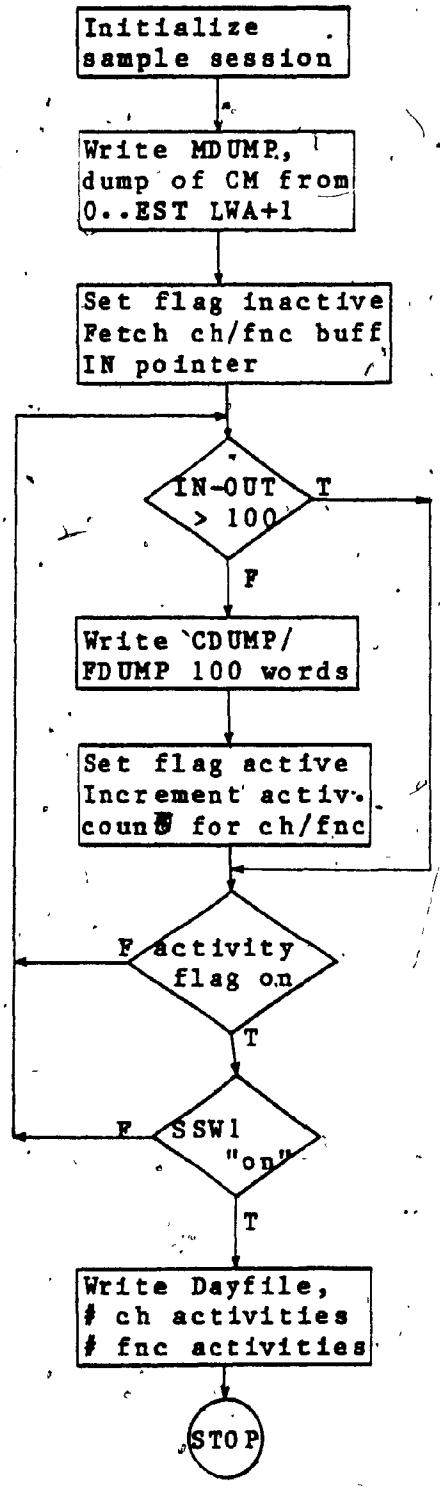


Fig. 3.2.2.1-1 Overall Flowchart of the SAMPLE program

3.2.2.2 Event-Trace Probe

The probe consists of modifications to MTR, CPUMTR and SET in the original version, however modifications have been added to gather I/O subsystem information, in addition to the upgrade modifications. Appendix C shows all these modifications.

A CIO call has been described in section 2.3.6. Here some important steps will be outlined in order to point out where code has been inserted to gather information about the I/O subsystem. When a read (for example) is requested, 6DI is called which in turn will request the CPUMTR function LDAM (Load Address for Mass Storage Drivers) which computes the physical address to be read from.

Thus, in LDAM everything required is available: sector, track, channel, unit number, and even the real-time clock for storing the seek starting time. This real-time is not the true seek starting time because the seek is not yet started. Nevertheless, the time is saved as the last thing done before jumping back to 6DI; and 6DI will, as its next operation, if the unit is not busy, send the function code to the channel to start the seek. This will only take a few microseconds. As the time is recorded in milliseconds, this inaccuracy is acceptable.

On return to 6DI, if the unit is not busy, the seek is started. However, if the unit is busy, 6DI calls MTR with a

Driver Seek Wait (DSWM) function to wait until the unit becomes free. So the seek-start time saved just before is no longer valid; however the other information is still valid because the sector to be read is not going to change its position. Now, when MTR finds that the unit is no longer busy (and the channel too), just before leaving the time of completion of DSWM is stored in CBUF with the function code and the PP output register address (0A).

Thus, in the analysis program, we have to keep a stack that will record the seek-start time anyway and if a DSWM is found, the time is changed. For the seek-end information, code could not be inserted in 6DI because there was no space left for additional code, however there is enough information which the analysis program can manipulate to get the seek-end time and the data transfer time as well. Since the function DSWM is issued every time MTR scans the PPs until the seek is finished, the last DSWM will give the seek-end time with fair accuracy but not the latency time. The subsequent channel drop will give the data transfer time which includes latency, data transfer and a small software overhead.

3.2.2.2.1 CPUMTR

The modifications to the CPUMTR program trap requests for assigns and drops of PPs, changes of CP assignment, field length changes, and seek-start times. All these requests are recorded in the function buffer (FBUF) in a 2-word entry and

then dumped to the FDUMP file. The following CPU monitor functions will cause a trace dump:

- ABTM (Abort CP),
- CCAM (Change CP Assignment),
- DPPM (Drop PP),
- RPPM (Request PP),
- UADM (Update Accounting and Drop),
- JACM (Job Advancement Control),
- TIOM (Tape I/O Processor),
- LDAM (Load Address for Mass Storage Driver).

For a PP request the following data are recorded:

word 1 - PP program name from IR (with the first bit=0 if an assign or 1 if a drop),
- CP number (with an auto-recall bit, if set),
- time that the request waited to be satisfied (PP assign), or
  number of PRUs transferred by the PP I/O activity (PP drop),
- real-time clock in msec,

word 2 - last three characters of the job name (with first bit=0),
- origin type (OT) of the job. There are five system-defined "types":
  0 - system (SYOT),
  1 - local batch (BCOT),
  2 - remote batch (EIOT),

3 - remote time-sharing (TXOT),

4 - multi-terminal (MTOT).

- PP number of requesting PP, or zero if request from CPU,
- PP number assigned to that task,
- CPU time used by the processor for that job.

If it is an assign, the time that the request waited to be satisfied is recorded and in the event that a PP calls another one, the calling PP number is included. If it is a drop, the number of PRUS transferred by the PP I/O activity is recorded. These data are dumped from JACM and DPPM; the PP request from another PP is non-zero only if called from RPPM.

If the request was for a relocation of the relative address of a job or an FL change, the information recorded will be:

word 1 - code, that will be 1 for an FL change and 2 for an RA change,
- CP number,
- previous FL,
- real-time clock,

word 2 - last three characters of the job name (first bit=0),
- origin type,
- updated FL,
- CPU time used by the processor for that job.

A change CP assignment causes two dumps, one immediately following the other, as follows:

word 1 - PP program name (with first bit=1),
- old CP number (with auto-recall bit, if set),
- number of PRUs transferred (if any),
- real-time clock in msec,

word 2 - last three characters of the job name (with first bit=1),
- origin type of old CP,
- PP number assigned,
- CPU time used for that job at the old CP;

word 1 - PP program name (with first bit=0),
- new CP number (with auto-recall bit, if set),
- real-time clock in msec,

word 2 - last three characters of the job name (with first bit=1),
- origin type of new CP,
- PP number assigned,
- CPU time used by the job at the new CP.

This trace data is only dumped from CCAM. If called by TIOM, the number of PRUs is non-zero, else it is zero.

For the I/O subsystem activity, the following is recorded:

word 1 - code, that will be 3 for seek-start;
- CP number,
- real-time clock in msec,

word 2 - PP number assigned,
- Channel number,
- Flag to indicate software or hardware reserved or

error,

- logical track number,
- logical sector number,
- device number,

For the seek-start information, code has been inserted in LDAM (Load Address for Mass storage driver) which is called by the PP from 6DI. For the seek-end information, the analysis program will refer to CDUMP, keeping track of DSWM samples (see the following section).

For the Peripheral Library Search the following is recorded:

word 1 - code, that will be equal to 4;
- CP number,
- PP number,
- real-time clock in msec,

word 2 - residency code:
0 - if found in Resident Peripheral Library (RPL),
1 - if found in Peripheral Library Directory (PLD),
2 - if Special Function Processor (SFP) routine;
- PP program name to be searched.

This sample occurs when the resident loader in a PP requests a search for the location of a routine to be loaded and executed. The routine can be in central memory (in the RPL), on disk (its location was found in the PLD), or it can be a routine which is executed by the SFP. The Periheral

Library Search sample type is necessary because certain PP routines terminate by overlaying themselves with another routine, rather than by informing CPUMTR.

More precise information about the format of the trace words can be found in appendix B.

3.2.2.2.2 MTR

The channel buffer (CBUF) records information coming from MTR about channel requests, assignments and drops of channels and, in particular cases, the new field length, seek-start time or storage move requests. All these requests are recorded in a 1 word entry in the buffer and then dumped onto the CDUMP file. The following MTR functions will cause a trace dump:

- CCHM (Check Channel),
- DCHM (Drop Channel),
- RCHM (Reserve Channel),
- RSTM (Request Storage),
- RSYM (Request System),
- DEPM (Disk Error Processor),
- DSWM (Driver Seek Wait).

The information recorded for channel usage is as follows:

- real-time clock in msec,
- PP function (with first bit "on" if the request was not honoured),

- second channel (on request) or zero,
- channel number,
- PP output register address.

This data is dumped from CCHM (if request is satisfied only), DCHM, RCHM, RSYM, and DEPM.

The second channel is equal to zero if the function is RCHM or RSYM. If we have an RCHM or RSYM, the first attempt is dumped. If not satisfied, then a repeat request must be made to MTR. These repeated requests are not dumped unless the request is satisfied. If we have an RCHM or RSYM function, then the second channel is not equal to zero, only if the request was honoured. In this case the channel number is the primary channel requested and the second channel is the alternate channel requested. The second channel is not equal zero only if the requesting PP specified an alternate channel. (Ch0 is the hardware deadstart channel and will never be requested or used).

For a storage request the sample contains:

- real-time clock in msec,
- PP function = 21(octal),
- FL requested (divided by 100 octal),
- PP output register address.

This data is dumped on an RSTM function only. It is dumped only on the first occurence of the request, that is, a subsequent request for the same FL at the same CP by the same PP is not dumped.

For a storage move the sample contains:

- real-time clock in msec,
- CP number being moved plus 2000 (octal),
- move block length (MBL),
- PP OA.

The data is dumped whenever a job is moved in storage. The MBL is the distance which the job is being moved and is a 12-bit 1's complement number. MBL less than zero indicates a lower move (toward word zero absolute). MBL greater than zero indicate an upper move. Move blocks are 100 (octal) words long, MBL is the distance of the move divided by 100 (octal).

For a Driver Seek Wait request the sample contains:

- real-time in msec,
- PP function = 33(octal),
- flag to determine if waiting for seek or for reserved unit,
- channel number,
- PP OR address.

This data is dumped whenever a DSWM request is finished, i.e., when control is returned to the calling PP.

3.2.2.2.3 SET

The modification to SET (Initialize System) initializes and defines the buffer pointers, and reserves a central memory area. The buffer pointers are maintained in the system installation area, in CMR. Each buffer is defined by one

word, where the first part contains the first word address of the buffer, and the second part contains the current IN pointer relative to the first word address. The SAMPLE program checks these pointers each time-slice it is given control; it empties the buffer and sets its own internal counters to keep track of how much data has been read.

3.2.3 Analysis Program

The analysis program is composed of two different modules:

- PREPROCESSOR-- a pre-processor which reads the low-core memory dump and creates tables in recognizable format for the analysis program.
- ANL-- a program which breaks down the information of the files created by the probe and merges it. The program gathers statistics and prints histograms for a wide range of different system and subsystem activities.

3.2.3.1 PREPROCESSOR

This program, written in PASCAL, runs in about 2500 (5000 octal) 60-bit words and it is about 400 lines long. The pre-processor program reads the file MDUMP created by the probe at the beginning of the run of SAMPLE. This file is a dump of the first 2000 words of memory (approximately). The pre-processor program analyzes this file to set up the three major tables used in the statistical analysis: the job accounting table, the PP accounting table and the channel table, and some

other information. This information is available except past history items (real-time began and real-time PP's were assigned, etc.). These are all assumed to begin at the time the run (SAMPLE) started. These tables are then written to a file PREPR which will be read by the analysis program. Some examples of output of this program and some utility programs are given in Appendix D.

3.2.3.2 ANL

This program, written in FORTRAN (FTN 4.6), runs in about 53.3K (150,000 octal) 60-bit words and it is a program of about 5,000 lines. This program is based on an early archive copy of the program developed by Paula Hawthorn (HAWTP 74) at the University of Houston. It was necessary to revise it extensively in order to duplicate Hawthorn's results. Once the analysis on the overall operating system was working correctly, the program was modified to accept the new data wich looks deeply into the I/O subsystem. The program breaks down the information of the two files (FDUMP and CDUMP) and merges it; the information is treated on a per-job basis by keeping track of the job name/PP assignments and noting when the CP changes (due to roll-out).

The two main features of the analysis program are that:

- everything happens in pairs-- for every assign there is a drop; action (updating of tables) does not take place until the drop since it is only then that all the required

information is available-- the amount of real-time taken, the amount of CPU time used by the job, etc.

- all data is kept in tabular (histogram) form.

The three major tables used in the statistical analysis are:

- the job accounting table, which contains 13 words per job active (either assigned to a CP or rolled out),

  1 - Job Name,

  2 - CP assigned (0 if rolled out),

  3 - real time began,

  4 - PP wait time,

  5 - field length (FL),

  6 - origin type (OT),

  7 - number of FL changes,

  8 - number of PP calls,

  9 - number of times rolled-out,

  10- total amount of time spent in PP calls,

  11- CPU time of last FL change,

  12- CPU time of last PP call,

  13- CPU time of last roll-out.

  This table has a limit of 100 jobs active at once; it is a self cleansing table (jobs that end are removed from the table). Jobs are in order only by the time they are first detected by the analysis program; when a job is removed, all the other jobs are moved up in the table.

- the PP accounting table, which includes 6 words per PP,

  1 - PP program name that holds the processor; if the

processor is not in use, this word is zero,

2 - job name of the job calling for the service provided by the PP,

3 - real-time when the PP was assigned,

4 - job CPU time when the PP was assigned,

5 - real-time the PP first asked for a channel,

6 - last channel dropped, assigned to this PP.

- the channel activity table,

1 - which PP is holding the channel,

2 - real-time when the channel was assigned.

There is another table in connection to the last two, which contains for each PP assigned, the list of channels assigned to that same PP [which could be more than one, a PP could call (theoretically) as many channels as are free!].

The program makes only one pass of the data files since these files can be as large as a million words. The reading of the files only once has given a fairly fast execution time.

A feature has been implemented in the program, which allows the selection of debugging printout (with up to 32 options) and, in addition, a particular class (up to seven) of histograms to be printed can be selected, thus saving a lot of execution time if only a specific category of histograms is needed or if only a specific part of the program needs some debugging. The program has also the capability of choosing four different logarithmic scales for the histograms. Thus,

according to a specific class of histograms more appropriately scaled histogram can be choosen. This has been found useful for printing finer scales for the seek-time and channel histograms.

3.2.3.2.1 Structure of the Program

The analysis program is a tree-structured, modular program that reads the files created by the probe and gathers statistics. When the analysis program begins, it reads the PREPR file to initialize its tables; then it calls all the modules to initialize themselves. It then reads FDUMP (function activity file) and CDUMP (channel activity file), to find the first sample in each file after the start of the experiment. The information in these files is handled on a per-entry basis, in time sequence; if the time is the same, the decision concerning which (FDUMP or CDUMP) entry to handle first is made on whether the FDUMP entry is an assign or a drop; if it is an assign PP, frequently the channel activity at the same time is an assign channel to that PP, so the assign PP is taken first, thus it will be known what PP program is using the channel. If the FDUMP entry is a drop, the channel activity is taken first; frequently the channel activity is a drop for the channel associated with the PP that is dropping and, again, it is necessary to know which PP to post the channel activity to. However some exceptions occur when the time is equal:

- if the sample is a change CP assignment, CTCHPP is called first, because the two consecutive dumps are processed as a drop and as an assign respectively, usually the second is an assign to the scheduler (1SJ), which will enter a new job or a roll-in;
- if the sample is an FL change, the first bit is equal to zero, as for an assign PP, but usually there is an FL request just before from the channel activity, thus it is necessary to read first CDUMP then FDUMP, which requires this trace word to be treated as a drop PP.

If the entry taken is the FDUMP entry, the routine called is CTCHPP. It is determined in that module if the entry is a field change or a seek-time entry; if so, FLCHNG or SEEKTM is called. If the entry taken is a CDUMP entry, RDCHAN is called. RDCHAN screens those entries that are requests for FL changes or seek-times; for these it calls FLCHNG or SEEKTM to handle the sample.

As we have seen, the two main routines of the analyzer are CTCHPP and RDCHAN, which subsequently can call FLCHNG or SEEKTM accordingly. There is another routine called by FLCHNG, JBACCT, the job accounting routine that records data when a job begins or ends; when the job's FL went from a value to nothing, when it was rolled out or from zero to something with no previous record of the job (meaning it was beginning).

There are several other routines used in the program. The following gives an idea of their objectives:

- CTRACE, this module keeps track of the CP usage. It is called whenever a CP is assigned or dropped;
- PTRACE, behaves as the previous one but instead for the PPs;
- HIST and PLOT, two service routines that generate all the histograms;
- CHNWRT, a module that is called only at the end of all jobs and which builds the tables for the histograms for all the channels;
- PRUC, the routine that keeps track of the number of PRUs written per channel; it is also keeping track of the number of transfers and PRUs transferred per CIO call per channel;
- ERROR, an error routine that prints all the different data errors detected.

3.2.3.2.2 Summary Description of the Major Modules

The CTCHPP routine takes the two-word PP entry; if it is a seek-time or a field change, it calls FLCHNG or SEEKTM and returns; otherwise, PTRACE is called. CTCHPP then tests to see if the entry is an assign or a drop PP. If it is an assign, the PP accounting tables are updated and the routine returns; however, a special case should be considered: the case of an entering job (new job or roll-in). The scheduler

(1SJ) does not have the job name and the origin type yet. Thus it is necessary to wait until this information is available which means saving that assign for a later comparison. If a drop, PRUC is called so that the histogram tables for the number of PRUs per channel can be updated. The histogram tables are checked for the name of the PP program. If not there, it is added and the tables updated. If it is a roll-out, FLCHNG is called. Then, the roll-out histogram tables are updated. If a PP call that is the drop of the previous assign with no job name is detected, then the corresponding tables are updated. If a roll-in is detected, FLCHNG is called. If the job does not exist, it could be that it was rolled out before the experiment started. In this case there is no accounting entry for it; the new job is created and FLCHNG is called to handle the FL change.

The RDCHAN module handles the assigns and drops for the channels. It tests first if the sample is a DSWM or a FL change function. Then it calls SEEKTM or FLCHNG and returns. If the upper bit of the function name is "on", it means that the request has been denied. In this case the time is posted in the PP accounting table and the routine returns. If it is an assign, the channel accounting table is updated and the channel posted in the table which keeps track of all channels assigned per PP. Then, the program returns. If it is a drop, RDCHAN finds the name of the PP program that was in the PP held by the channel, and the wait and hold time. It then

writes the information gathered on the file for that particular channel. In this module it was found that a PP was allowed to ask, when it was assigned, for more than one channel; thus it was necessary to build a table to keep track of all the channels assigned to a particular PP. The channel activity is filtered and written to several files, one per every two channels. The end of job program CHNWRT then reads these files and generates the histograms.

The SEEKTM routine handles the disk unit time. It tests if the sample is an LDAM sample, if so, it stores in a table all the information about that seek, such as: unit number, track, sector, time, etc. It also dumps on a file all the pertinent information belonging to the previous seek for the same PP. If a drop Ch is encountered, the information for the previous seek is also dumped. If the sample is a DSWM sample, the SEEKTM routine tests first for the unit reserved flag, if true, it changes the start time and computes the wait time; if the unit was not reserved, it saves the time (used to compute the data transfer time) and accumulates some counters. With the available information from FDUMP and CDUMP the different times are computed as follows:

DSWM(00) = waits for seek, a sequence of these are issued when waiting for the seek to be completed;

DSWM(10) = wait for unit reserved, one occurs at the end of the waiting period;

device wait time (if any) = DSWM(10) - LDAM time;

seek time = last DSWM(00) time - LDAM time
(a seek after an address computation);

seek time = last DSWM(00) time - DSWM(10) time
(a seek after a wait for unit reserved);

data transfer time = Ch drop time - last DSWM(00) time
(only one address computation has been issued);

data transfer time = Ch drop time - DSWM(10) time
(a wait for unit reserved has been issued);

data transfer time = next LDAM time - last DSWM(00) time
(the first of a sequence of address computations);

data transfer time = next LDAM time - LDAM time
(one among several address computations).

The information gathered for each seek is as follows:

1 - always get 1 (one) LDAM,

2 - always get (if seek took place) at least 1 (one) DSWM, may have:

a - J = 0 or 1 (one) software waits DSWM(10) (reserved unit),

b - K >= 1 (one) hardware waits DSWM(00) (seek).

Because channels are freed during a seek or while waiting for the unit, we collect a sequence of DSWMs for the original

PP, which do not belong to seek waits or unit waits (i.e., only the last DSWM of a series is significant, since it marks the end of the waiting period). Thus we had to manipulate the data to discard the unnecessary DSWM samples. Accounting tables are kept to record all these activities. At the end of the analysis run, histograms per unit and total for seek time, cylinder distribution, traveled length and data transfer time are printed; a seek time table is also printed.

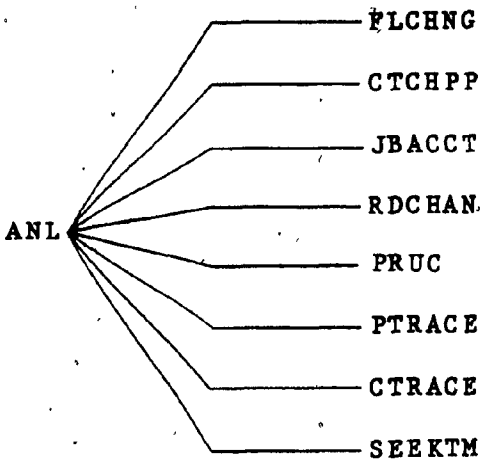
The FLCHNG module handles all the FL changes occurring during the execution of a job, including when it enters and leaves the operating system. First, it tests if the request is coming from CDUMP. If yes, this request is saved in a stack to wait for the corresponding FL request from FDUMP, that is, for each FL request from CDUMP there will be (usually), immediately or later on, an FL request from FDUMP. Then the routine returns. If the request is from FDUMP, the job is removed from the stack. If it was in, then FLCHNG updates all the tables for the histograms and the memory space. If it is a new job or a finishing one, JBACCT is called to update the job accounting table.

3.2.3.2.3 Interconnection between all Program Routines

The analysis program has been written in three distinct parts:

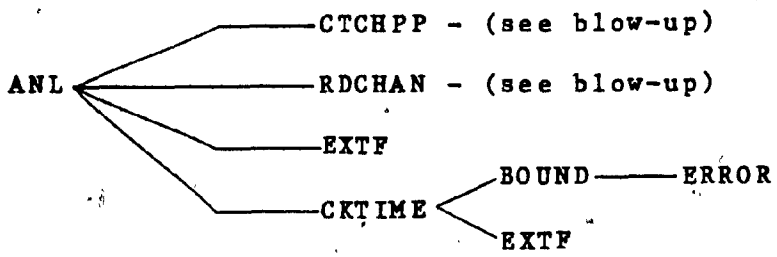
- Initialization; each module that will produce information for the histograms is called to initialize itself: setting up constants, initializing tables, and setting the appropriate information read from the PREPR file.
- Processing; all the information in CDUMP and FDUMP is read, analyzed and processed, to compute statistics and build histogram tables.
- End-of-Job; all the histograms (if selected) and statistics are printed.

The initialization phase is shown in fig. 3.2.3.2.3-1. Fig. 3.2.3.2.3-2 shows the processing part and fig. 3.2.3.2.3-3 the end-of-job section.



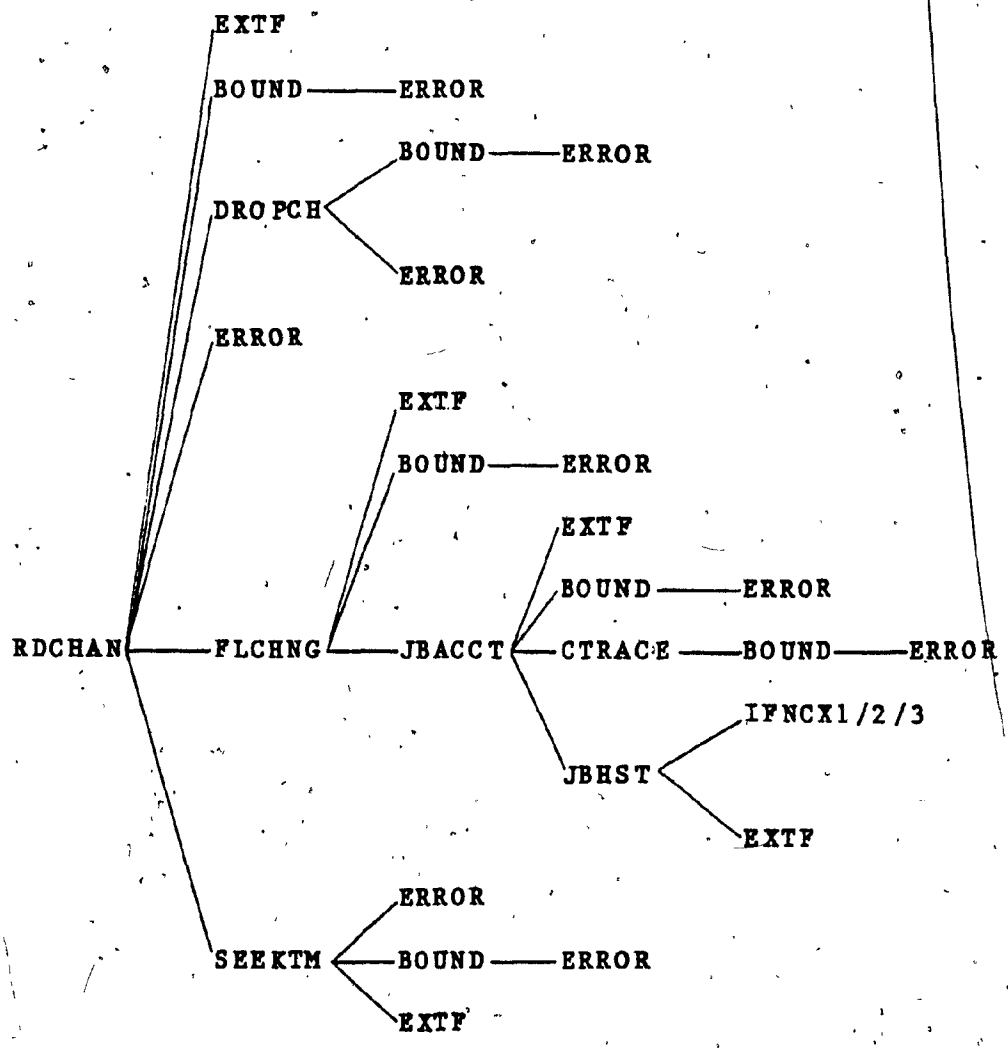
ANL - Analysis program (main program).
FLCHNG - Field Changes, memory management.
CTCHPP - Catch PP, analysis of the PP-word trace pair.
JBACCT - Job Accounting, whenever a job finishes or ends.
RDCHAN - Read Channel, analysis of the Channel-word trace.
PRUC - Pru Accounting.
PTRACE - PP Accounting.
CTRACE - CP Accounting.
SEEKTM - Seek Time, analysis of the seek-word trace.

Fig. 3.2.3.2.3-1 Modules Called to Initialize themselves



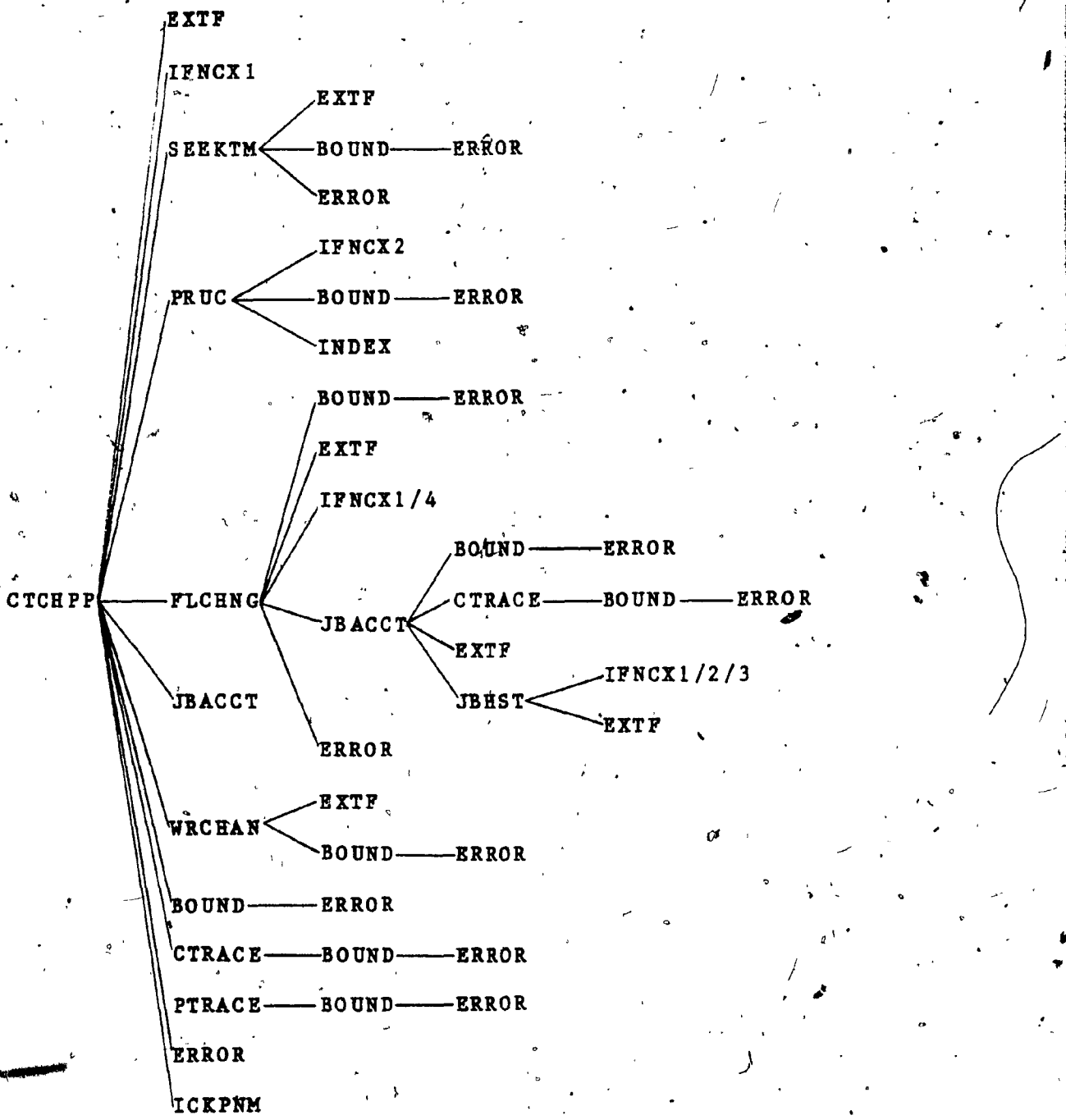
EXTF - Extract a field inside a word (CM word).
BOUND - Check if a value is inside a bound.
ERROR - Prints error messages.
CKTIME - Check if real time is always increasing.

Fig, 3.2.3.2.3-2 Routines Called during the Processing Phase



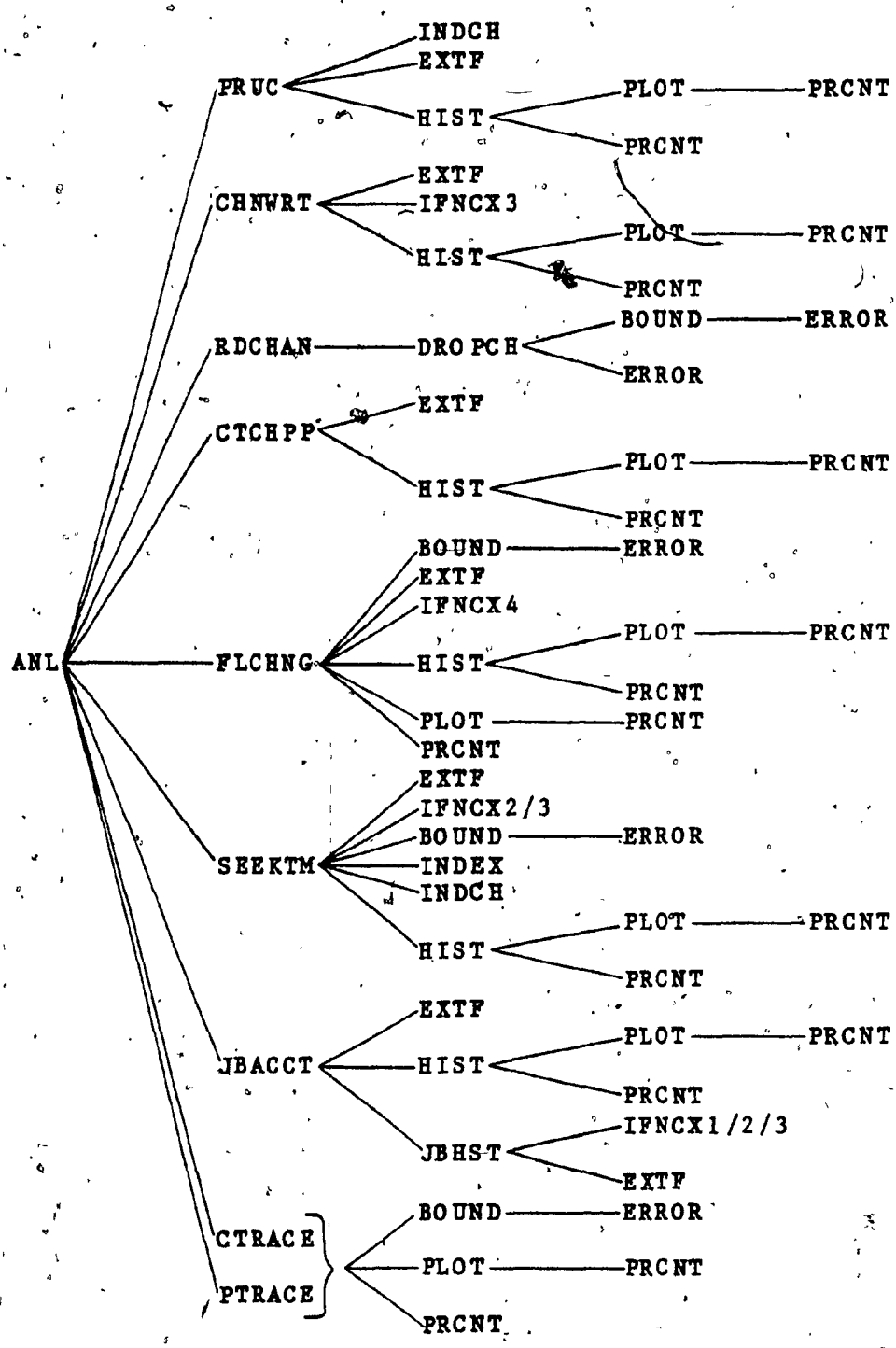
IFNCX1/2/3 - Logarithmic function for the X-axis.
JBHST - Update job accounting tables.
DROPCH - Handle drops of channels.

Fig. 3.2.3.2.3-2 ... Processing Phase (continued)



WRCHAN - Writes the PP name, in the Channel-file, of the corresponding PP dropped.
ICKPNM - Check if PP name is valid.
INDEX - Find index given Ch number.

Fig. 3.2.3.2.3-2 ... Processing Phase (continued)



HIST - Set up title and tables for the histogram.
PLOT - Print the histogram.
PRCNT - Compute grouped percentages for the X-axis.
INDCH - Find channel number giving an index.

Fig. 3.2.3.2.3-3 Routines Called at End-of-Job

3.2.4 Test Data Base

A benchmark tape was available with a collection of what are considered to be typical user jobs. The tape contains 31 batch jobs of different categories, most of them were FORTRAN programs (FTN), some COBOL programs, and one call of the sort-merge package (SORTMRG) which will give to our test a large number of I/O transfers which is our point of interest. Most of the programs were a sequence of "compile" and "execute" steps, while a very few were only a "compile" step. These were not the only programs running during the test; over 40 jobs had been encounterd by the end of the test, because of system jobs like: the System itself, Magnet, Batchio, the Sample program, and the system program CMS (Check Mass Storage) which comes in at inervals of 60 seconds to check the MS equipment.

To this extent they do not represent a typical day's workload. However, they are useful in providing a repeatable benchmark by which system changes may be evaluated.

According to the site's system analyst, during the benchmark run the system was as busy as during the busy time of a normal day's workload. Further information on this point is given in Chapter V.

C H A P T E R IV

HISTOGRAMS

4.1 Introduction

It was impossible to leave all the histograms in the thesis since there would be over 1,000 pages, a non standard and impractical size for a publication. Thus it has been decided to insert only the most important histograms of Run 7, which does have the most realistic configuration.

Following is the list of histograms for Run 7. For all PP and Ch routines, only those with more than 100 occurrences have been included:

hist.# 1-57 PP routines,

- hold time (# 1-26): only CIO, 1MT, 1IO, 1SJ, 1AJ, 1MA, LDR and PRU are included;
- CPU time (# 27-44): only CIO and PRU are shown, the scale is too coarse for the others;
- wait time (# 45-54): only CIO and PRU are shown, for the others there is no real wait time, only the clock ticking is recorded;

hist.# 58-71 Overall information: not shown since chapter V describes it in enough detail;

hist.# 72-78 CM utilization: all shown;

hist.# 79-84 PP and CP utilization: all shown;

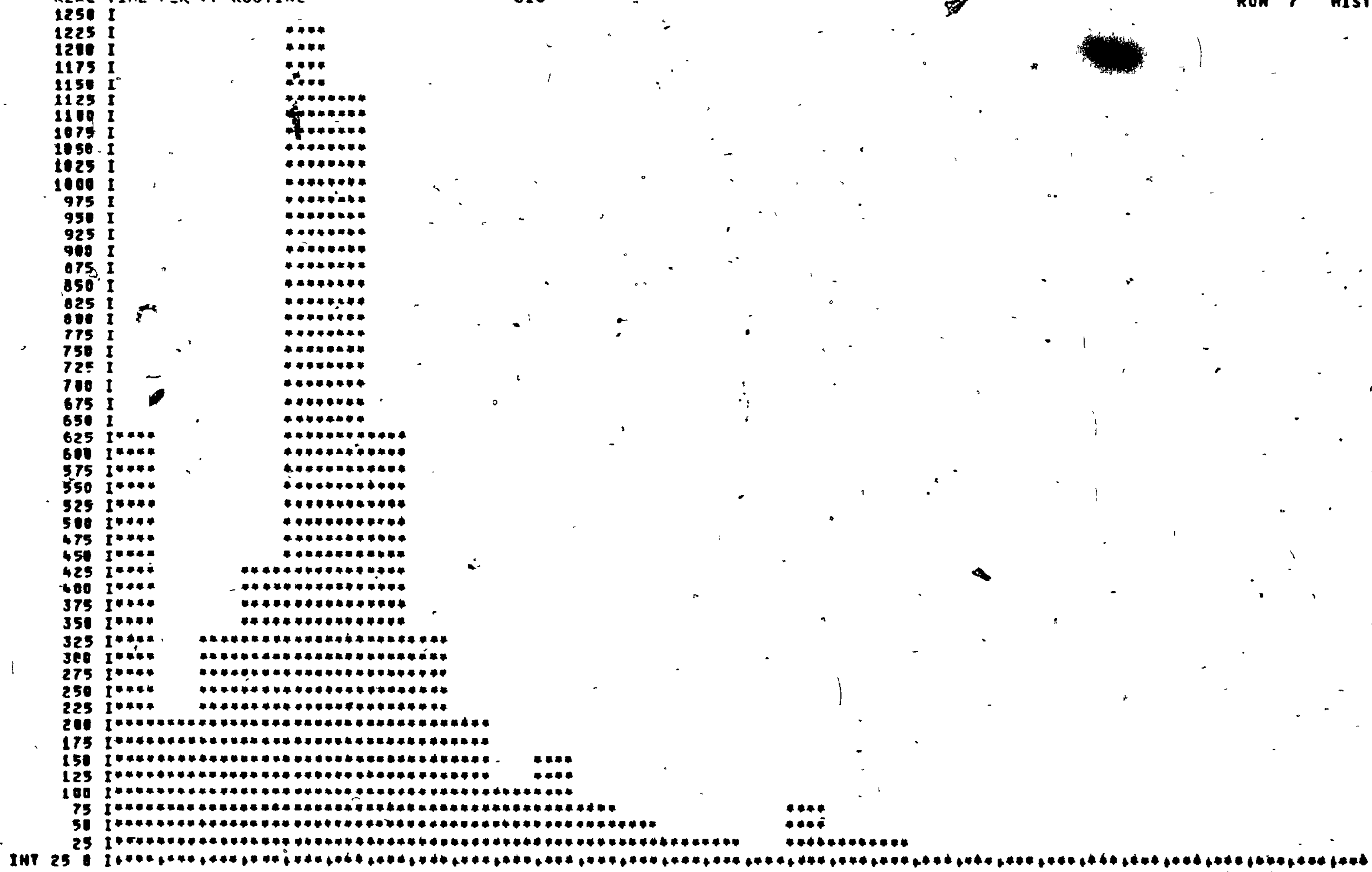
hist.# 85-162 Ch routines,

- Ch2 (# 85-119): only QFM, CIO, IAJ, LDR, PRU and Total are included;
- Ch4 (# 120-138): only CIO, PRU and Total are included;
- Overall PRU is also included;

hist.# 163-173 PRUs statistics: only overall information is included, in chapter V there is enough detail for the subject;

hist.# 174-191 Disk statistics: all included.

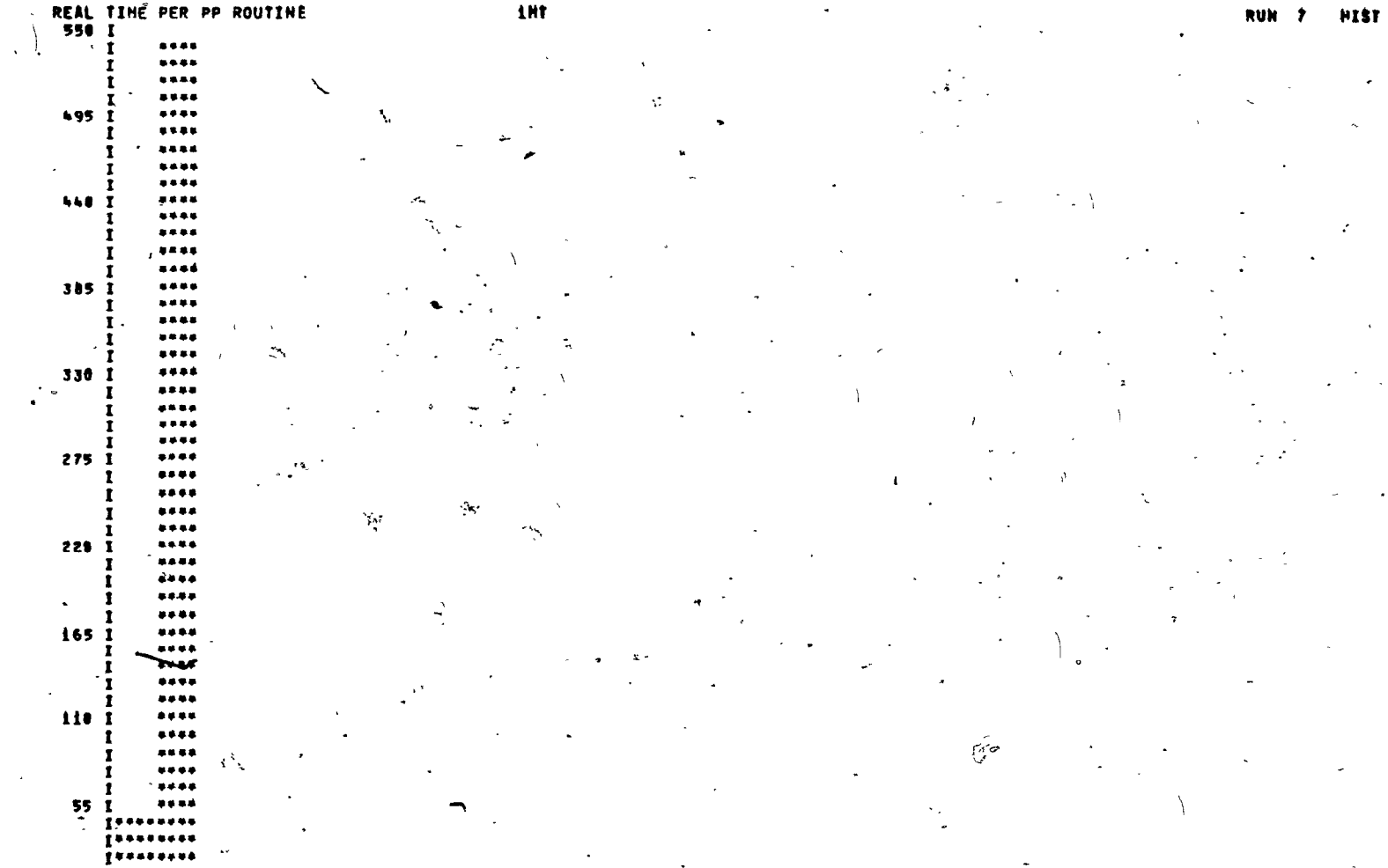
REAL TIME PER PP ROUTINE CIO RUN 7 HIST 1



	0	5	10	15	20	30	40	50	60	70	80	100	120	140	160	180	200	300	400	500	600	700	800	1.0	1.5	2.0	2.5	3.0	3.5	4.0
% Y-AXIS	11	4	6	7	21	19	11	6	3	2	3	2	1	0	1	0	2	1	1	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	1	1	1	3	11	14	10	7	5	3	5	4	2	2	2	1	8	6	5	4	3	2	1	0	0	0	0	0	0	0
GROUPED PERCENTAGES (X)						30.34					33.54							21.12						15.01						0.00

TOTAL Y-AXIS	5905	MAXIMUM Y-AXIS	1239
TOTAL X-AXIS	279965	MAXIMUM X-AXIS	30460
AVERAGE (X-AXIS)	46.73	STANDARD DEVIATION	74.98
MAXIMUM OBSERVED	832	TOTAL NUMBER OF PP DROPS	8365

REAL TIME PER PP ROUTINE INT RUN 7 HIST 2



% Y-AXIS	7	77	3	1	2	3	2	1	2	0	1	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	1	38	2	2	3	6	5	2	7	2	5	0	14	3	2	0	2	6	0	0	0	0	0	0	0	0	0	0	0	0

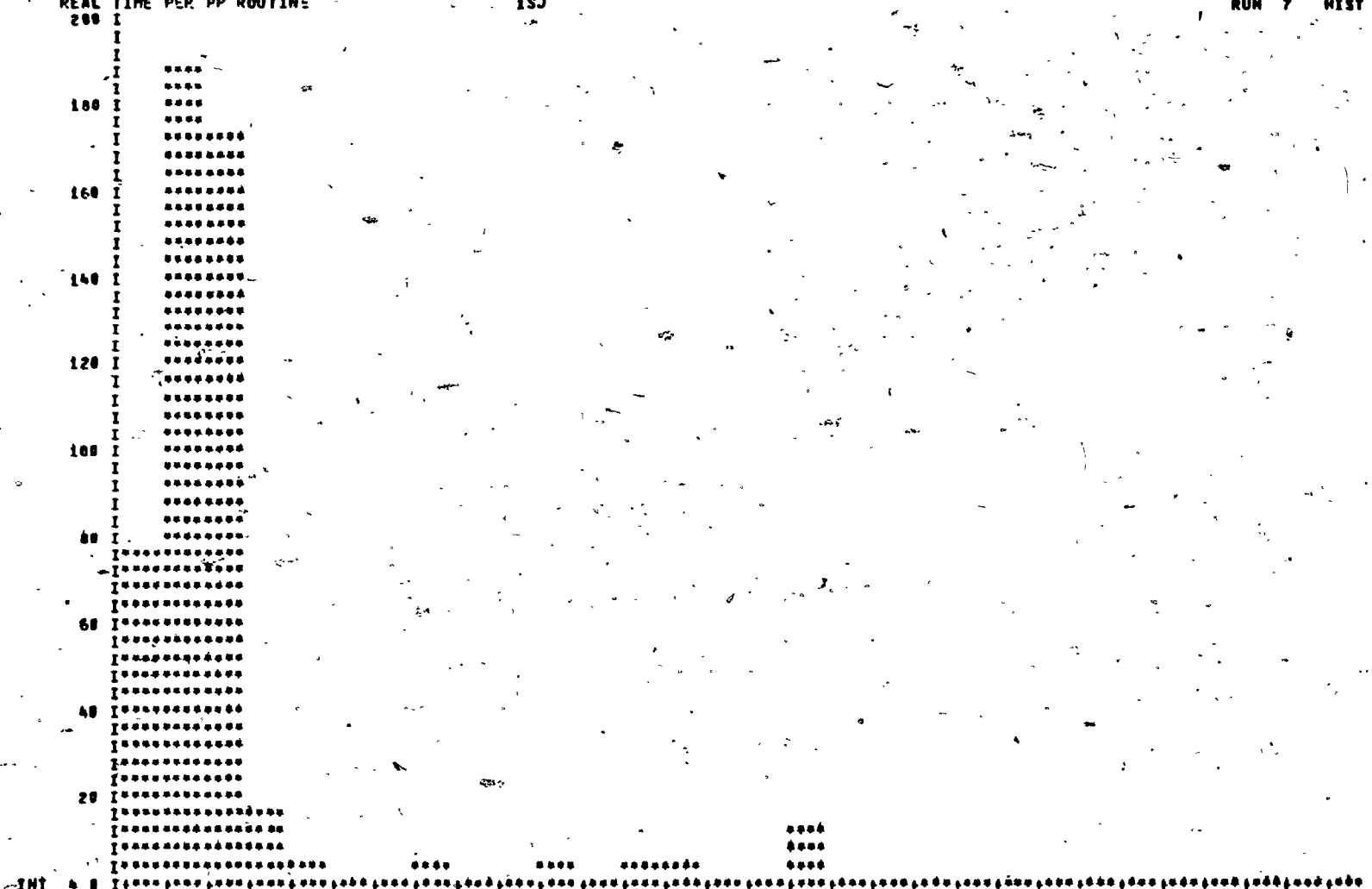
GROUPED. PERCENTAGES (X) 52.00 28.93 27.00 0.00 0.00

TOTAL Y-AXIS	718	MAXIMUM Y-AXIS	549
TOTAL X-AXIS	10031	MAXIMUM X-AXIS	3820
AVERAGE (X-AXIS)	14.13	STANDARD DEVIATION	28.18
MAXIMUM OBSERVED	339	TOTAL NUMBER OF PP DROPS	8365

```
REAL TIME PER PP ROUTINE                    110                                                    RUN 7   HIST   3
 250 I
 245 I
 240 I
 235 I
 230 I
 225 I
 220 I    ****
 215 I    ****
 210 I    ****
 205 I    ****
 200 I    ****
 195 I    ****
 190 I    ****
 185 I    ****
 180 I    ****
 175 I    ****
 170 I    ****
 165 I    ****
 160 I    ****
 155 I    ****
 150 I    ****
 145 I    ****
 140 I    ****
 135 I    ****
 130 I    ****
 125 I    ****
 120 I    ****
 115 I    ****
 110 I    ****
 105 I    ****
 100 I    ****
  95 I    ****
  90 I    ****
  85 I    ****
  80 I    ****
  75 I    ****
  70 I    ****
  65 I    ****
  60 I    ****
  55 I    ****
  50 I    ****
  45 I    ****
  40 I    ****
  35 I    ****
  30 I    ****
  25 I    ****
  20 I    ****
  15 I    ****
  10 I    ****
   5 I    ********
INT 5 0 I+---+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
        0    5   10   15   20   30   40   50   60   70   80  100  120  140  160  180  200  300  400  500  600  700  800  1.0  1.5  2.0  2.5  3.0  3.5  4.0

% Y-AXIS    0   97    3    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
% X-AXIS    0   94    6    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
GROUPED PERCENTAGES (X)   100.00                       0.00                     0.00                     0.00                     0.00
        TOTAL Y-AXIS                 231          MAXIMUM Y-AXIS              223
        TOTAL X-AXIS                1450          MAXIMUM X-AXIS             1367
        AVERAGE (X-AXIS)               6.20       STANDARD DEVIATION             .78
        MAXIMUM OBSERVED              11          TOTAL NUMBER OF PP DROPS   8365
```

REAL TIME PER PP ROUTINE $SJ RUN 7 HIST 4

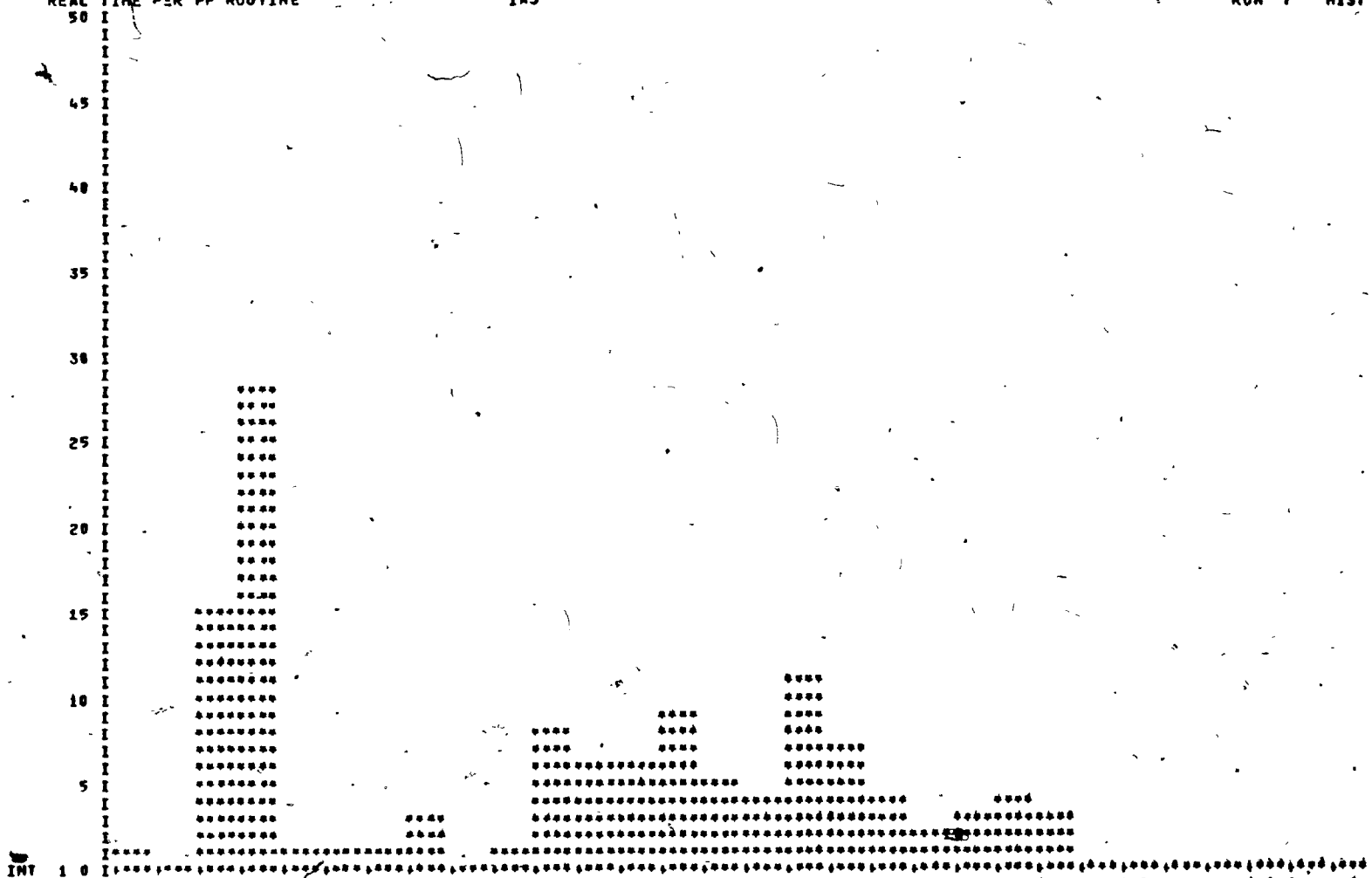


% Y-AXIS	15	38	34	3	1	0	1	1	0	0	1	0	1	1	0	1	3	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	2	10	20	2	1	0	1	2	0	1	3	2	5	6	3	5	29	3	4	0	0	0	0	0	0	0	0	0	0	0

GROUPED PERCENTAGES (X) 35.32 9.97 50.43 4.28 0.00

TOTAL Y-AXIS	505	MAXIMUM Y-AXIS	198
TOTAL X-AXIS	10925	MAXIMUM X-AXIS	3115
AVERAGE (X-AXIS)	21.63	STANDARD DEVIATION	80.98
MAXIMUM OBSERVED	468	TOTAL NUMBER OF PP DROPS	8365

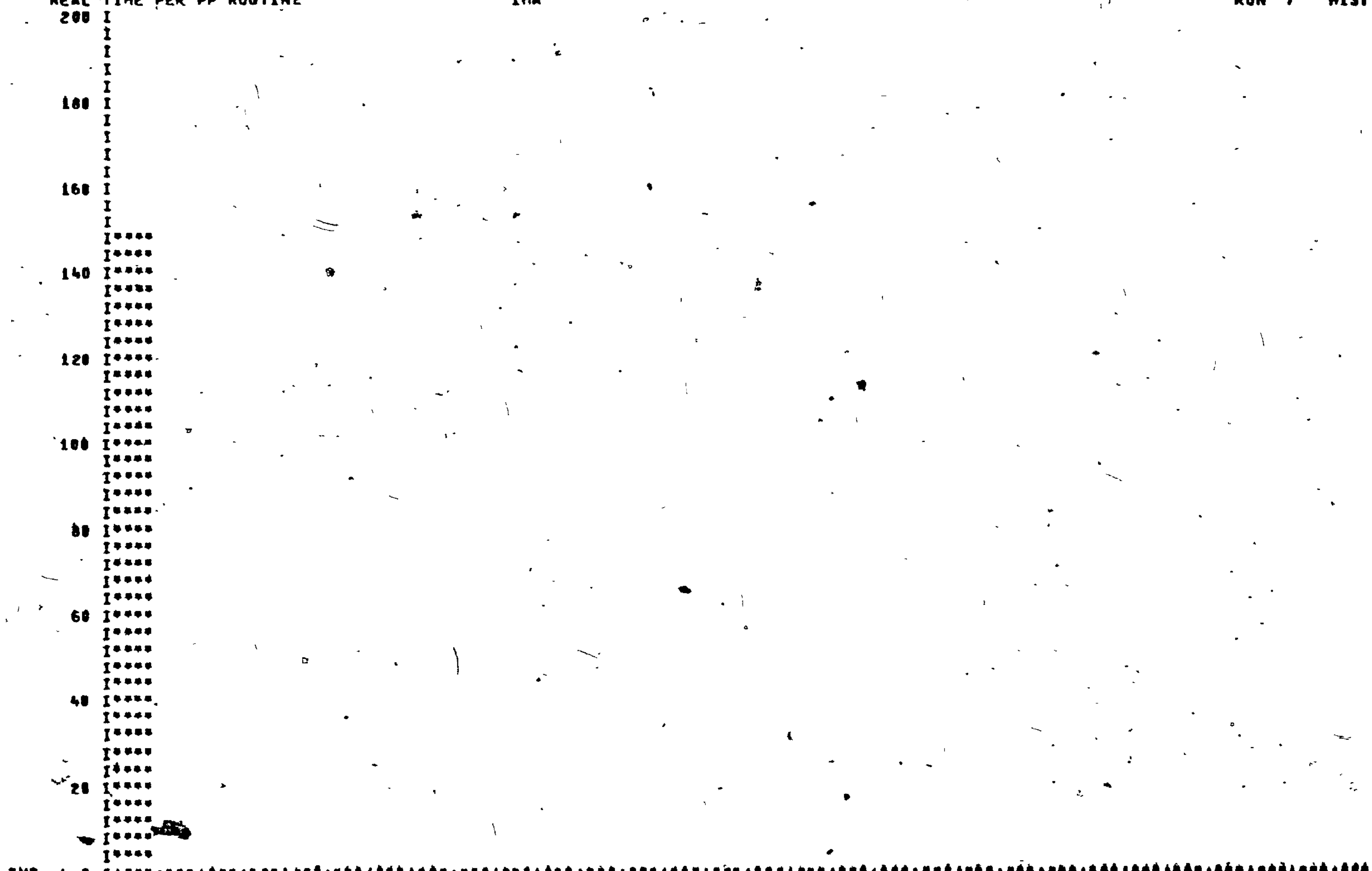
REAL TIME PER PP ROUTINE 1AJ RUN 7 HIST 0



% Y-AXIS	1	0	12	23	1	1	1	2	0	1	7	5	5	7	4	3	9	6	3	2	2	3	2	0	0	0	0	0	0	0	0
% X-AXIS	0	0	1	2	0	0	0	1	0	0	3	3	3	6	4	3	12	12	8	5	9	13	13	0	0	0	0	0	0	0	0
GROUPED PERCENTAGES (X)				3.41							7.59						40.88						48.12								0.00

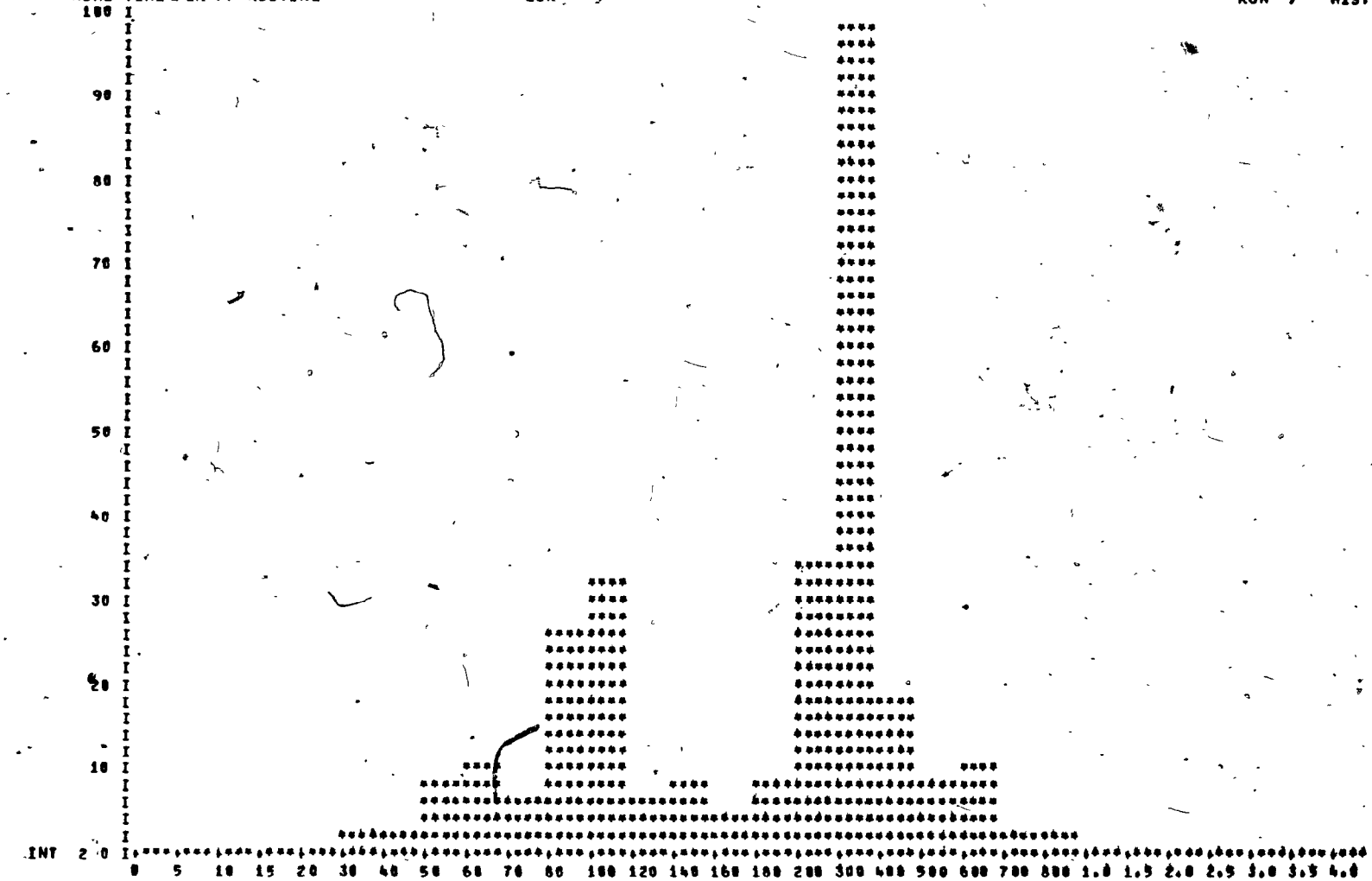
TOTAL Y-AXIS	123	MAXIMUM Y-AXIS	28
TOTAL X-AXIS	21836	MAXIMUM X-AXIS	2848
AVERAGE (X-AXIS)	177.53	STANDARD DEVIATION	216.74
MAXIMUM OBSERVED	993	TOTAL NUMBER OF PP DROPS	8365

REAL TIME PER PP ROUTINE 1MA RUN 7 HIST 13



INT	0	5	10	15	20	30	40	50	60	70	80	100	120	140	160	180	200	300	400	500	600	700	800	1.0	1.5	2.0	2.5	3.0	3.5	4.0
% Y-AXIS	93	2	0	0	1	1	1	1	2	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	21	2	0	0	2	3	4	9	17	6	0	0	0	0	0	0	0	0	40	0	0	0	0	0	0	0	0	0	0	0
GROUPED PERCENTAGES (%)					28.23						31.57							0.00						40.21						0.00

TOTAL Y-AXIS	162	MAXIMUM Y-AXIS	150
TOTAL X-AXIS	1169	MAXIMUM X-AXIS	470
AVERAGE (X-AXIS)	7.22	STANDARD DEVIATION	30.35
MAXIMUM OBSERVED	470	TOTAL NUMBER OF PP DROPS	8365

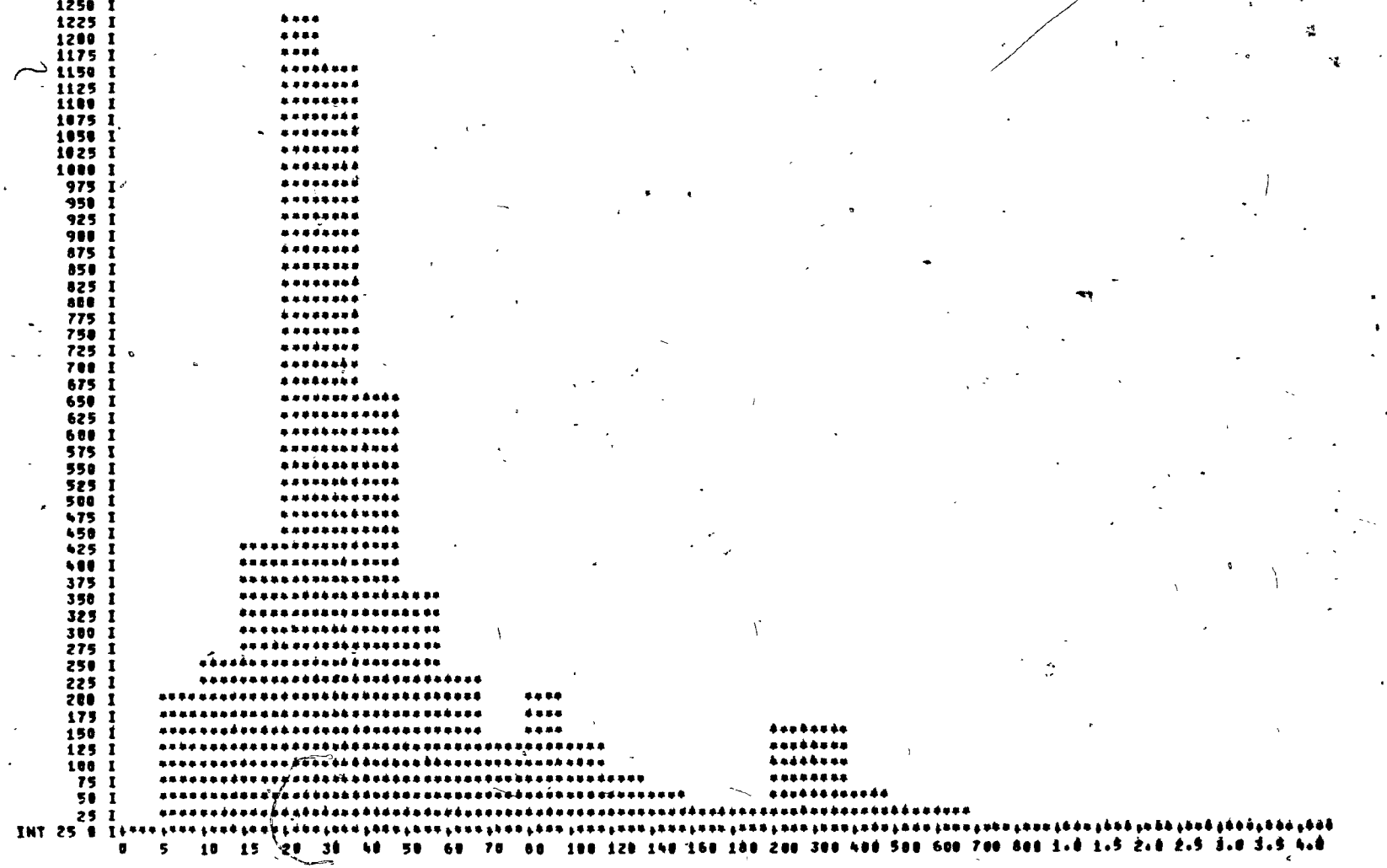


% Y-AXIS	0	0	0	0	0	1	1	3	3	2	9	11	2	3	1	3	12	34	6	3	3	1	1	0	0	0	0	0	0	0
% X-AXIS	0	0	0	0	0	0	0	1	1	1	3	5	1	2	1	2	11	43	11	6	6	3	2	0	0	0	0	0	0	0

GROUPED PERCENTAGES (%) .14 9.81 59.97 30.09 0.00

TOTAL Y-AXIS	294	MAXIMUM Y-AXIS	99
TOTAL X-AXIS	78552	MAXIMUM X-AXIS	33966
AVERAGE (X-AXIS)	267.18	STANDARD DEVIATION	163.78
MAXIMUM OBSERVED	848	TOTAL NUMBER OF PP DROPS	8365

REAL TIME PER PP ROUTINE PRU RUN 7 HIST 26

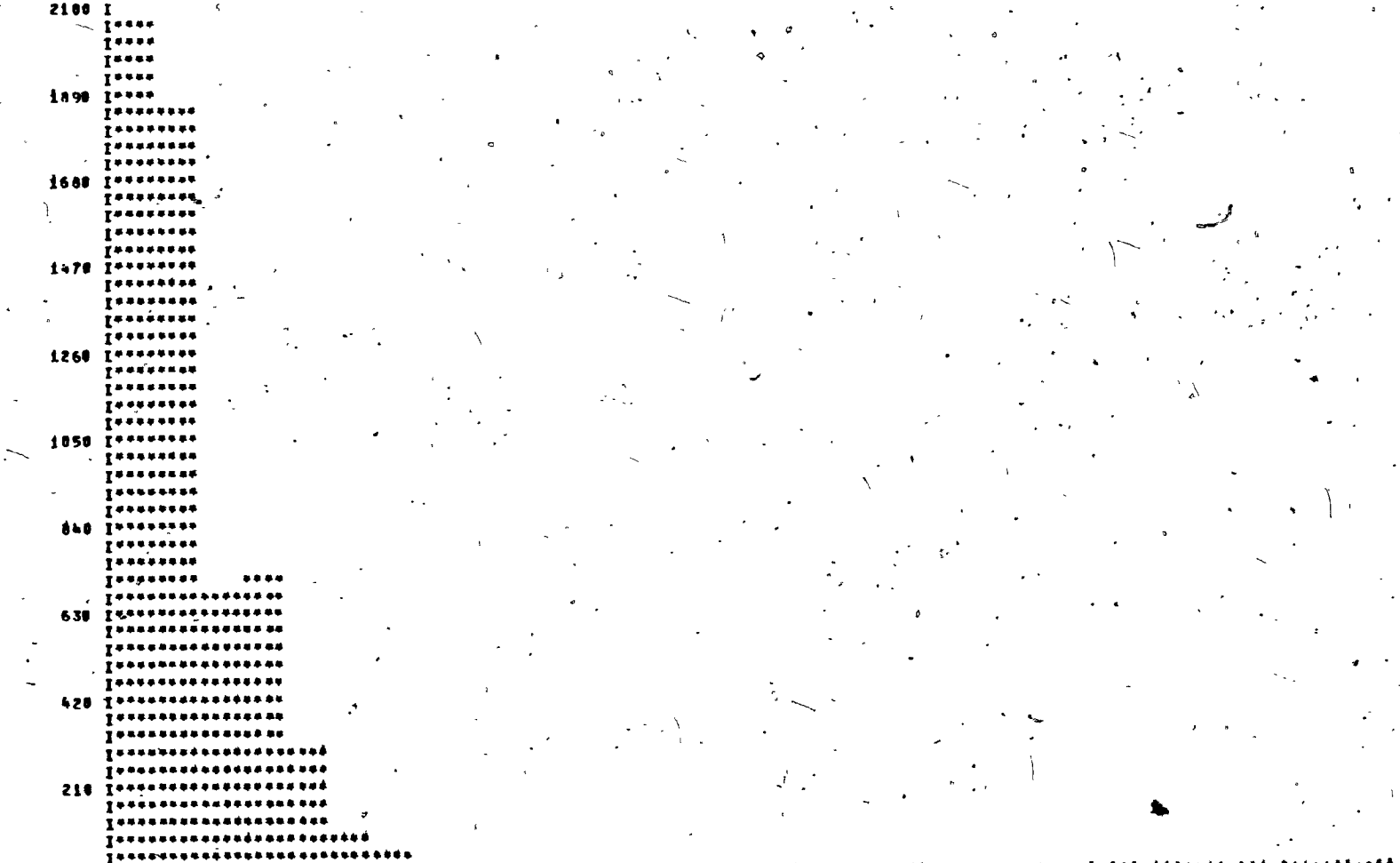


% Y-AXIS	0	4	5	8	21	28	11	6	4	2	4	2	1	1	1	1	3	3	1	1	1	0	0	0	0	0	0	0	0	0
% X-AXIS	0	0	1	2	7	9	6	4	3	2	4	3	2	2	2	2	8	13	7	8	5	3	3	3	1	1	0	0	0	0

GROUPED PERCENTAGES (X) 18.42 24.15 29.32 26.42 1.70

TOTAL Y-AXIS	5819	MAXIMUM Y-AXIS	1243
TOTAL X-AXIS	448540	MAXIMUM X-AXIS	57731
AVERAGE (X-AXIS)	77.08	STANDARD DEVIATION	134.86
MAXIMUM OBSERVED	2346	TOTAL NUMBER OF PP DROPS	8365

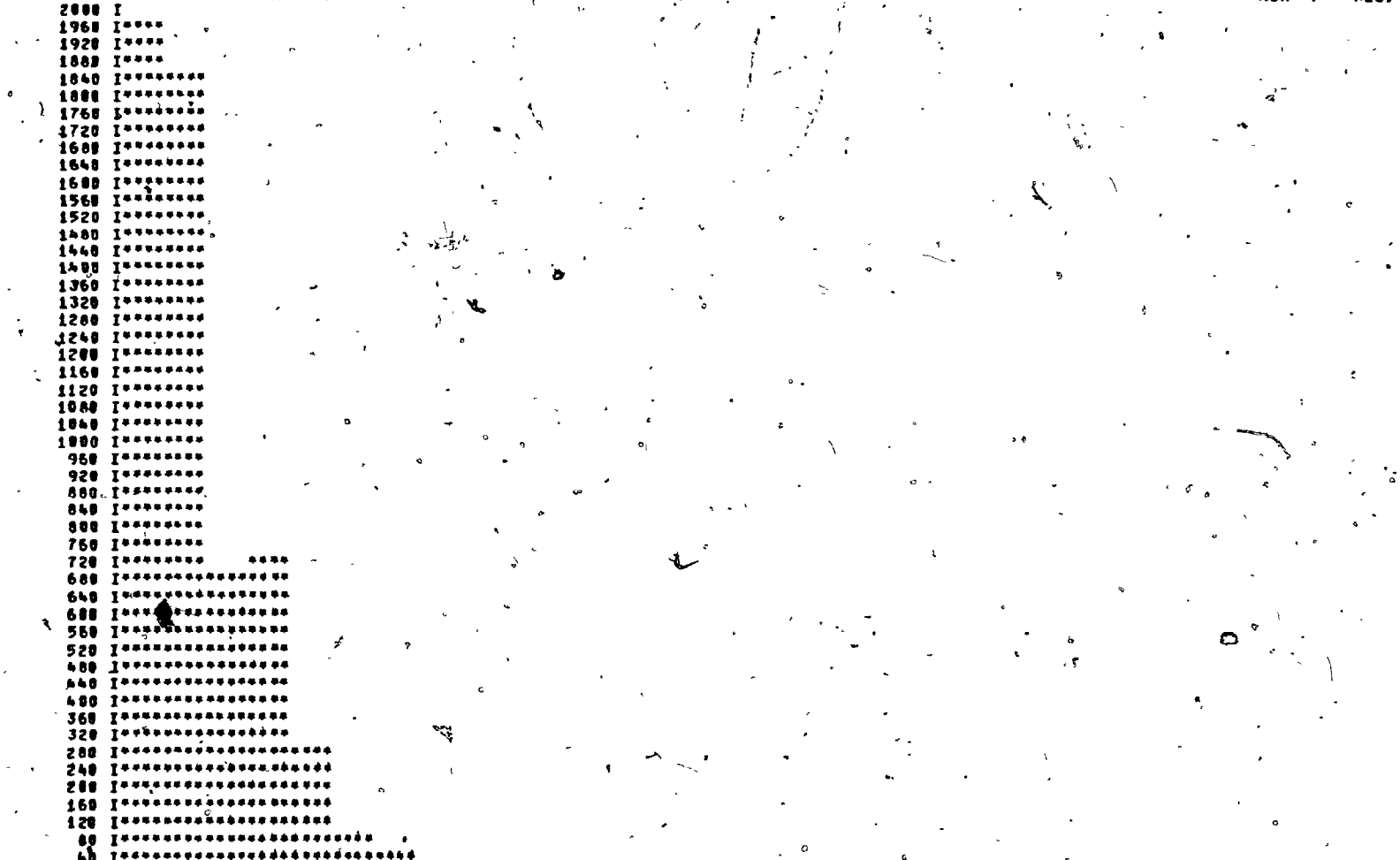
CPU TIME PER PP ROUTINE CIO RUN 7 HIST 27



% Y-AXIS	35	32	12	13	5	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	6	25	15	22	13	6	4	3	1	2	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
GROUPED PERCENTAGES (X)			86.87							11.88							2.84					0.00							0.00	

TOTAL Y-AXIS	5986	MAXIMUM Y-AXIS	2879
TOTAL X-AXIS	54926	MAXIMUM X-AXIS	13668
AVERAGE (X-AXIS)	9.38	STANDARD DEVIATION	11.33
MAXIMUM OBSERVED	293	TOTAL NUMBER OF PP DROPS	8369

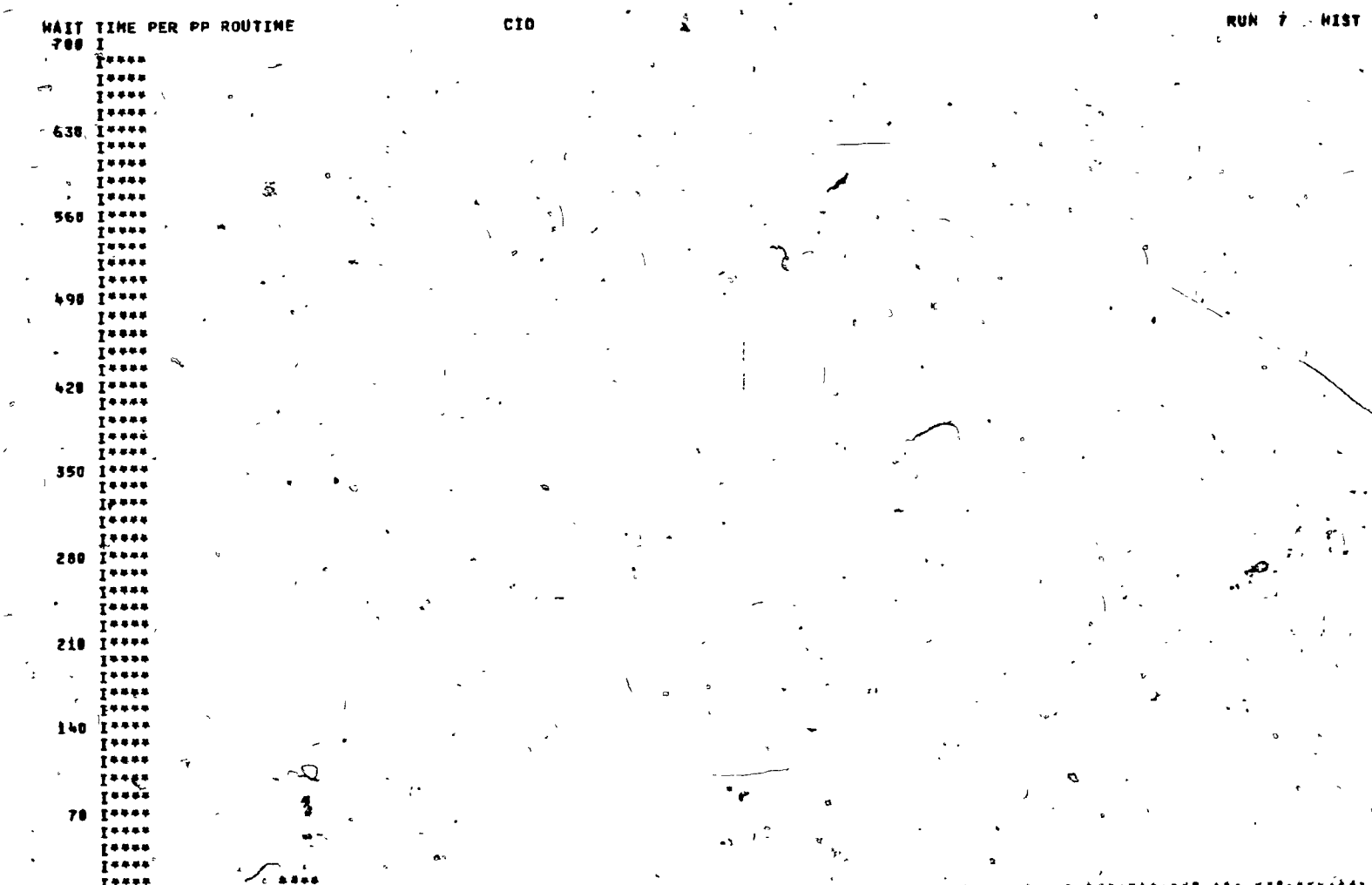
CPU TIME PER PP ROUTINE . RRU RUN 7 HIST 44



% Y-AXIS	34	32	12	13	5	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X X-AXIS	6	25	16	21	13	6	4	3	1	2	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

GROUPED PERCENTAGES (X)	86.63	11.34	2.03	0.00	0.00

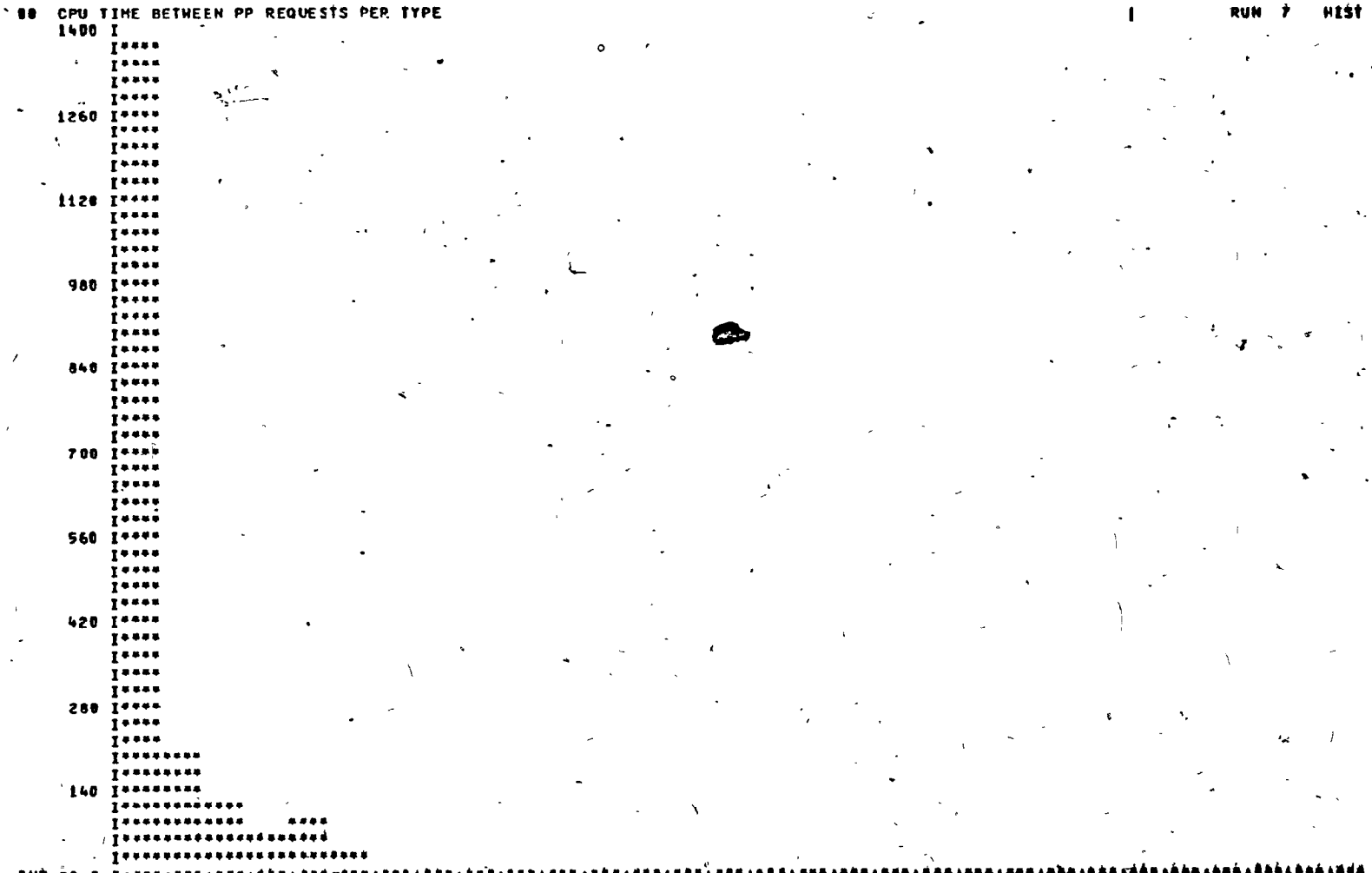
TOTAL Y-AXIS	5819	MAXIMUM Y-AXIS	1965
TOTAL X-AXIS	55327	MAXIMUM X-AXIS	13667
AVERAGE (X-AXIS)	9.51	STANDARD DEVIATION	11.45
MAXIMUM OBSERVED	293	TOTAL NUMBER OF PP DROPS	8365


WAIT TIME PER PP ROUTINE
CIO
RUN 7 HIST 45
700
630
560
490
420
350
280
210
140
70
INT 14.8
0 5 10 15 20 30 40 50 60 70 80 100 120 140 160 180 200 300 400 500 600 700 800 1.0 1.5 2.0 2.5 3.0 3.5 4.0
% Y-AXIS 93 1 0 1 3 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
X X-AXIS 25 2 0 2 10 0 10 2 2 3 19 0 9 0 0 7 0 0 0 0 0 0 0 0 0 0 0 0 0 0
GROUPED PERCENTAGES (X) 47.38 36.58 10.13 0.68 0.00
TOTAL Y-AXIS 747
TOTAL X-AXIS 2789
AVERAGE (X-AXIS) 3.73
MAXIMUM OBSERVED 193
MAXIMUM Y-AXIS 695
MAXIMUM X-AXIS 707
STANDARD DEVIATION 14.11
TOTAL NUMBER OF PP DROPS 8369

```
WAIT TIME PER PP ROUTINE               PRU                                                  RUN 7   HIST 54
 750 I
 735 I****
 720 I****
 705 I****
 690 I****
 675 I****
 660 I****
 645 I****
 630 I****
 615 I****
 600 I****
 585 I****
 570 I****
 555 I****
 540 I****
 525 I****
 510 I****
 495 I****
 480 I****
 465 I****
 450 I****
 435 I****
 420 I****
 405 I****
 390 I****
 375 I****
 360 I****
 345 I****
 330 I****
 315 I****
 300 I****
 285 I****
 270 I****
 255 I****
 240 I****
 225 I****
 210 I****
 195 I****
 180 I****
 165 I****
 150 I****
 135 I****
 120 I****
 105 I****
  90 I****
  75 I****
  60 I****
  45 I****
  30 I****
  15 I****                  ****
INT 15 0 I+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
          0    5   10   15   20   30   40   50   60   70   80  100  120  140  160  180  200  300  400  500  600  700  800  1.0  1.5  2.0  2.5  3.0  3.5  4.0 .

% Y-AXIS   96    0    0    0    2    0    1    0    0    0    1    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0    0
% X-AXIS   33    0    0    1   16    0   11    2    0    0   16    0   12    0    0    9    0    0    0    0    0    0    0    0    0    0    0    0    0    0
GROUPED PERCENTAGES (X)      50.40                     29.45                      20.14                    0.00                          0.00
          TOTAL Y-AXIS                    775            MAXIMUM Y-AXIS              746
          TOTAL X-AXIS                   2234            MAXIMUM X-AXIS              747
          AVERAGE (X-AXIS)                2.88           STANDARD DEVIATION           12.52
          MAXIMUM OBSERVED                193            TOTAL NUMBER OF PP DROPS   8365
```

88 CPU TIME BETWEEN PP REQUESTS PER TYPE | RUN 7 HIST 55

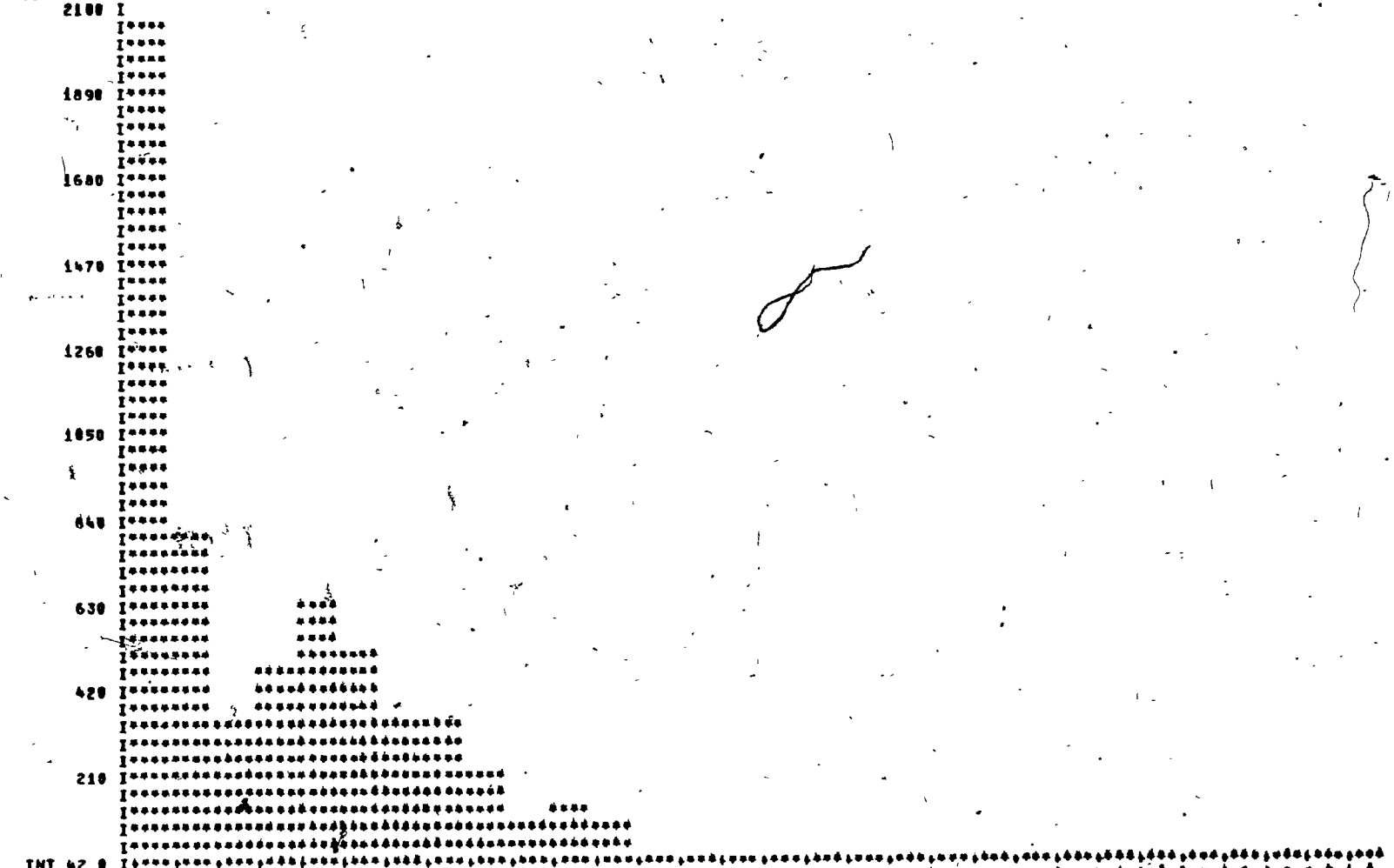


% Y-AXIS	69	11	7	4	5	2	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	7	8	9	7	15	9	3	2	1	1	3	1	2	6	3	6	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GROUPED PERCENTAGES (%)					56.54						11.95							31.51						0.00							0.00

TOTAL Y-AXIS	2031	MAXIMUM Y-AXIS	1384
TOTAL X-AXIS	17428	MAXIMUM X-AXIS	2667
AVERAGE (X-AXIS)	8.58	STANDARD DEVIATION	24.87
MAXIMUM OBSERVED	261	TOTAL NUMBER OF PP DROPS	8365

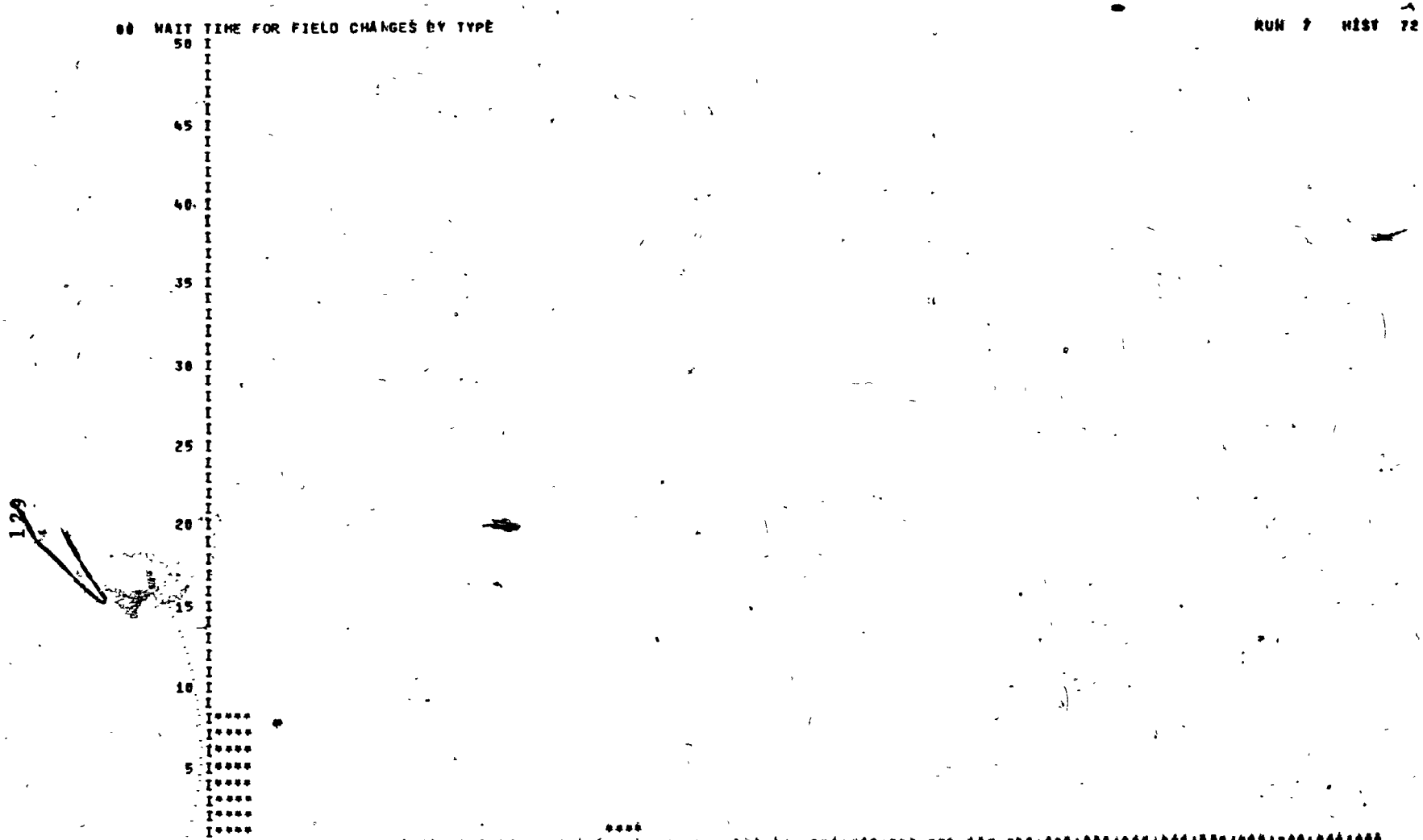
01 CPU TIME BETWEEN PP REQUESTS PER TYPE

RUN 7 HIST 56



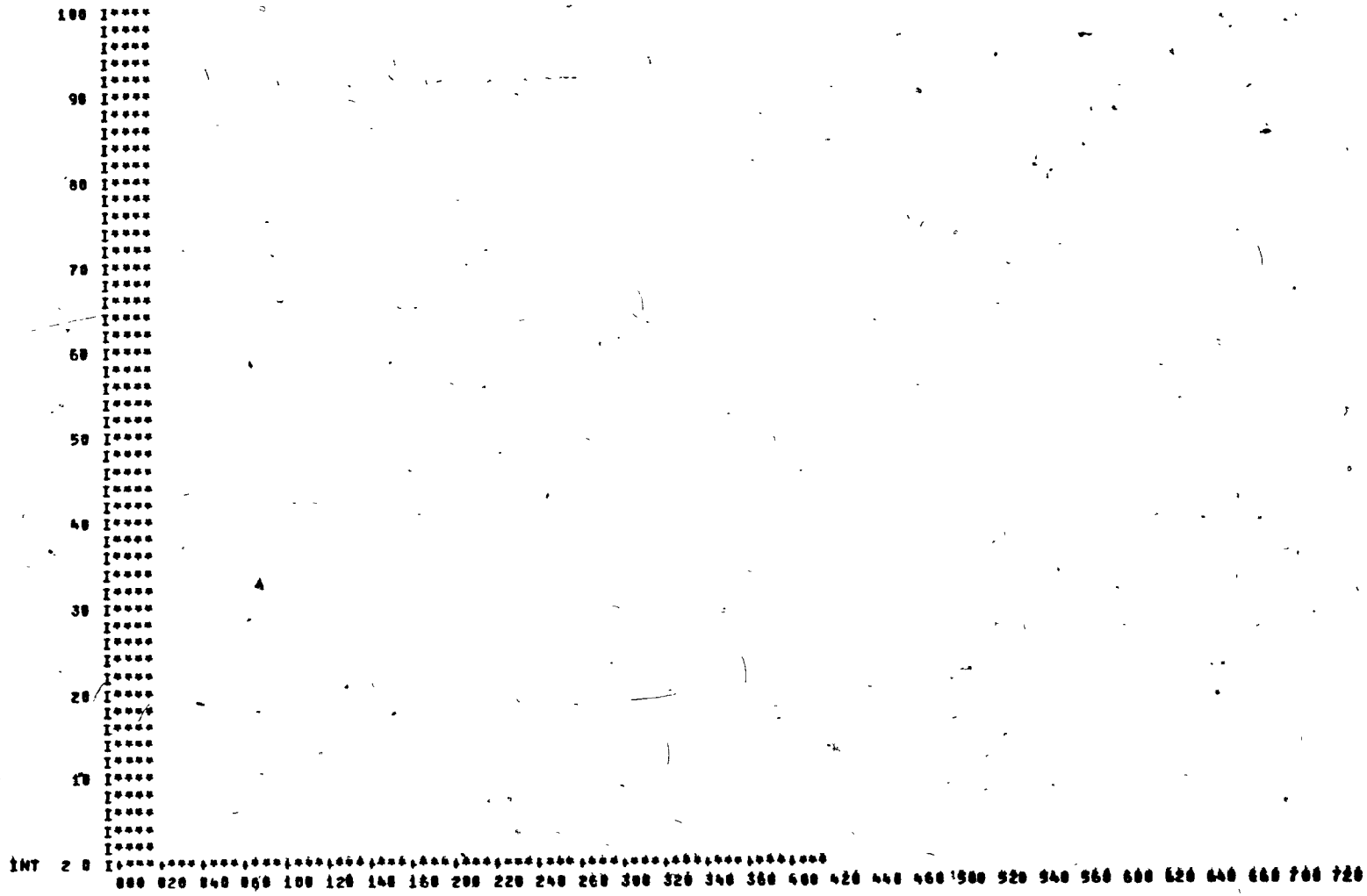
% Y-AXIS	33	13	6	8	10	8	5	5	4	2	2	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	26
% X-AXIS	1	3	2	3	7	8	7	8	7	4	5	5	2	2	1	1	3	2	1	0	0	0	0	0	0	0	0	0	0	0	0
GROUPED PERCENTAGES (X)					23.68						35.94						18.48						1.95								20.03

TOTAL Y-AXIS	6264	MAXIMUM Y-AXIS	2884
TOTAL X-AXIS	223939	MAXIMUM X-AXIS	59886
AVERAGE (X-AXIS)	35.75	STANDARD DEVIATION	377.14
MAXIMUM OBSERVED	20682	TOTAL NUMBER OF PP DROPS	8365


88 WAIT TIME FOR FIELD CHANGES BY TYPE
RUN 7 HIST 72
50
45
40
35
30
25
20
15
10
5
INT 1.0
0 5 10 15 20 30 40 50 60 70 80 100 120 140 160 180 200 300 400 500 600 700 800 1.0 1.5 2.0 2.5 3.0 3.5 4.0
% Y-AXIS 89 0 0 0 0 0 0 0 0 0 11 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
% X-AXIS 9 0 0 0 0 0 0 0 0 0 91 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
GROUPED PERCENTAGES (X) 8.89 91.11 0.00 0.00 0.00
TOTAL Y-AXIS 9
TOTAL X-AXIS 91
AVERAGE (X-AXIS) 10.00
MAXIMUM OBSERVED 82
MAXIMUM Y-AXIS 8
MAXIMUM X-AXIS 82
STANDARD DEVIATION 25.46

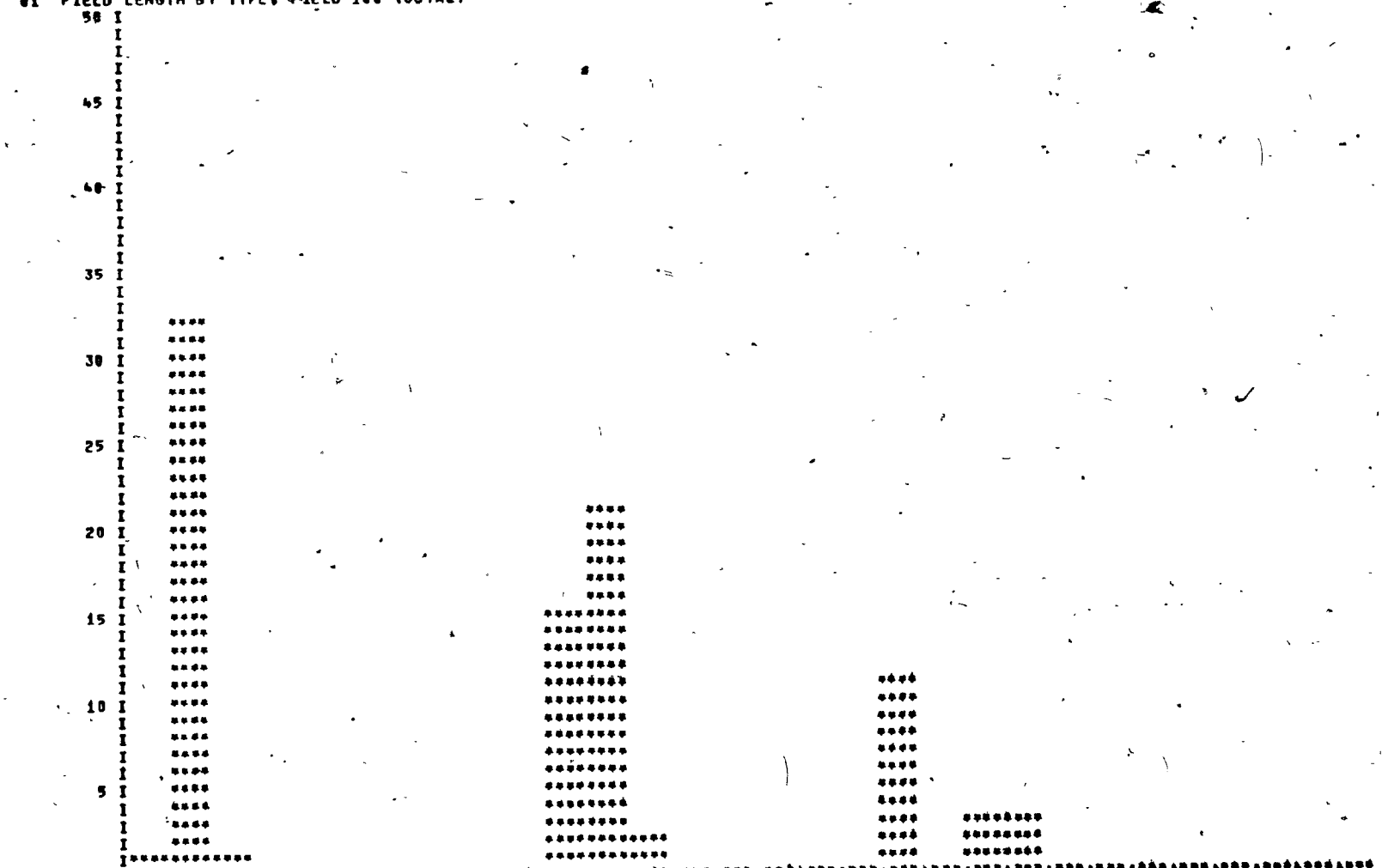
PERCENT OF TOTAL TIME TYPE 0 HELD THIS TOTAL FIELD LENGTH (PARTIONS OF 20,000 (OCTAL))

RUN 7 HIST 73



TOTAL TIME 247981

01 FIELD LENGTH BY TYPE, FIELD*100 (OCTAL)



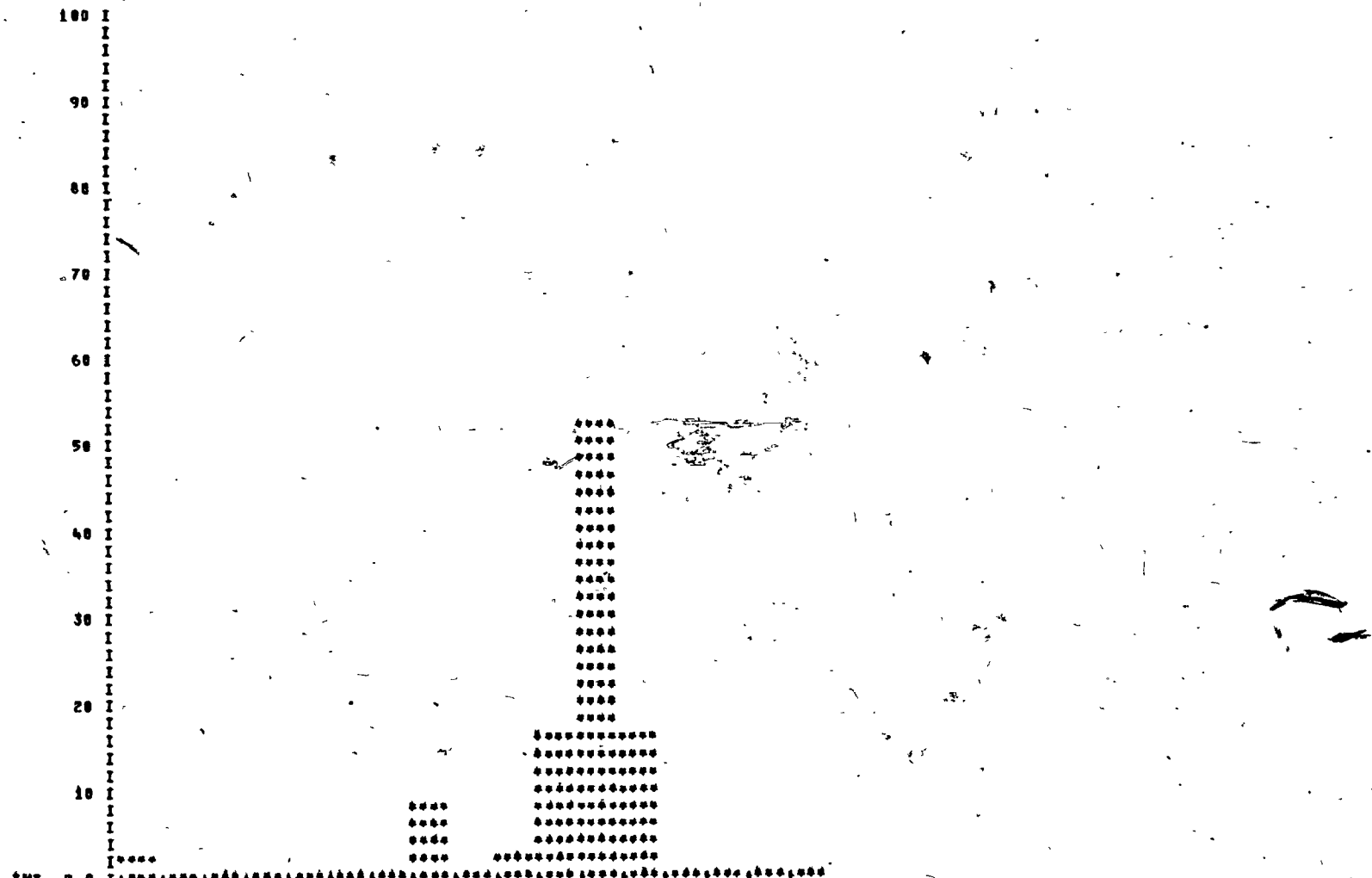
% Y-AXIS	1	36	1	0	0	0	0	0	0	0	17	24	2	0	0	0	0	0	12	0	3	3	0	0	0	0	0	0	0	0	0
% X-AXIS	0	2	0	0	0	0	0	0	0	0	14	23	3	0	0	0	0	0	34	0	11	13	0	0	0	0	0	0	0	0	0

GROUPED PERCENTAGES (X) 2.25 36.38 2.56 58.82 0.00

TOTAL Y-AXIS	89	MAXIMUM Y-AXIS	32
TOTAL X-AXIS	10330	MAXIMUM X-AXIS	3548
AVERAGE (X-AXIS)	116.07	STANDARD DEVIATION	127.58
MAXIMUM OBSERVED	448	07000	

PERCENT OF TOTAL TIME TYPE 1 HELD THIS TOTAL FIELD LENGTH (PARTIONS OF 20,000 (OCTAL))

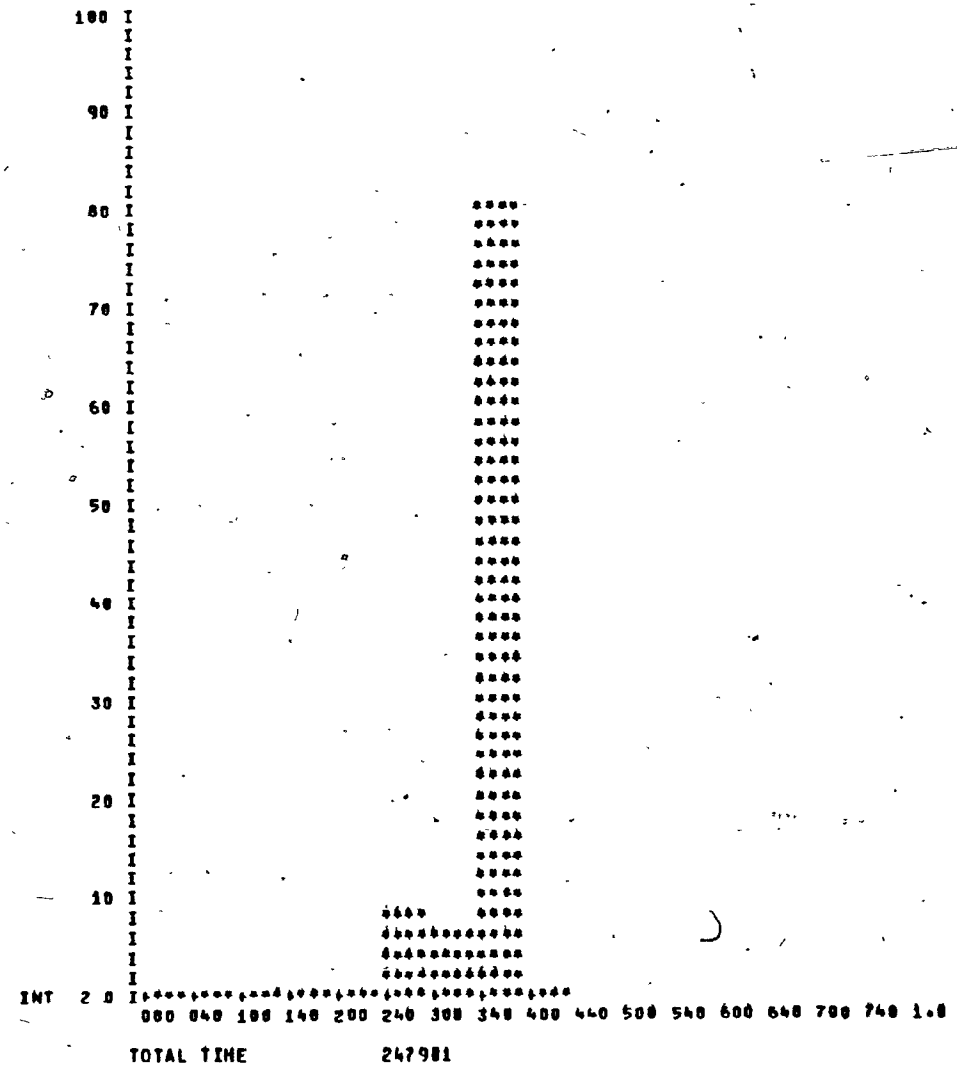
RUN 7 MIST 77



TOTAL TIME 247700

MEMORY UTILIZATION BY PERCENT OF TOTAL TIME (PARTIONS OF 40,000 (OCTAL))

RUN 7 HIST 78

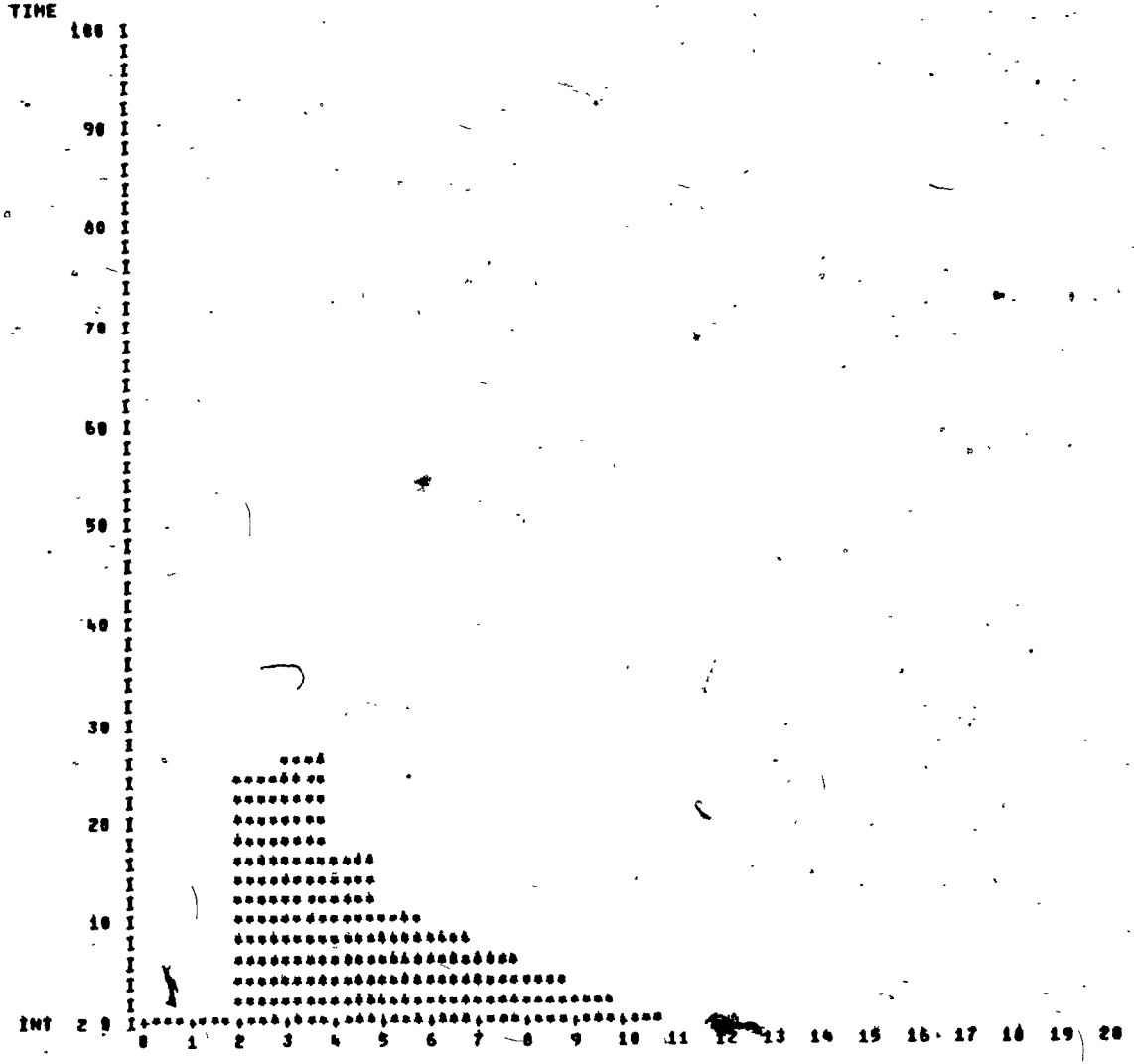


TOTAL TIME 247981

NUMBER OF PPS IN USE 247788 SECONDS

PERCENT OF TIME

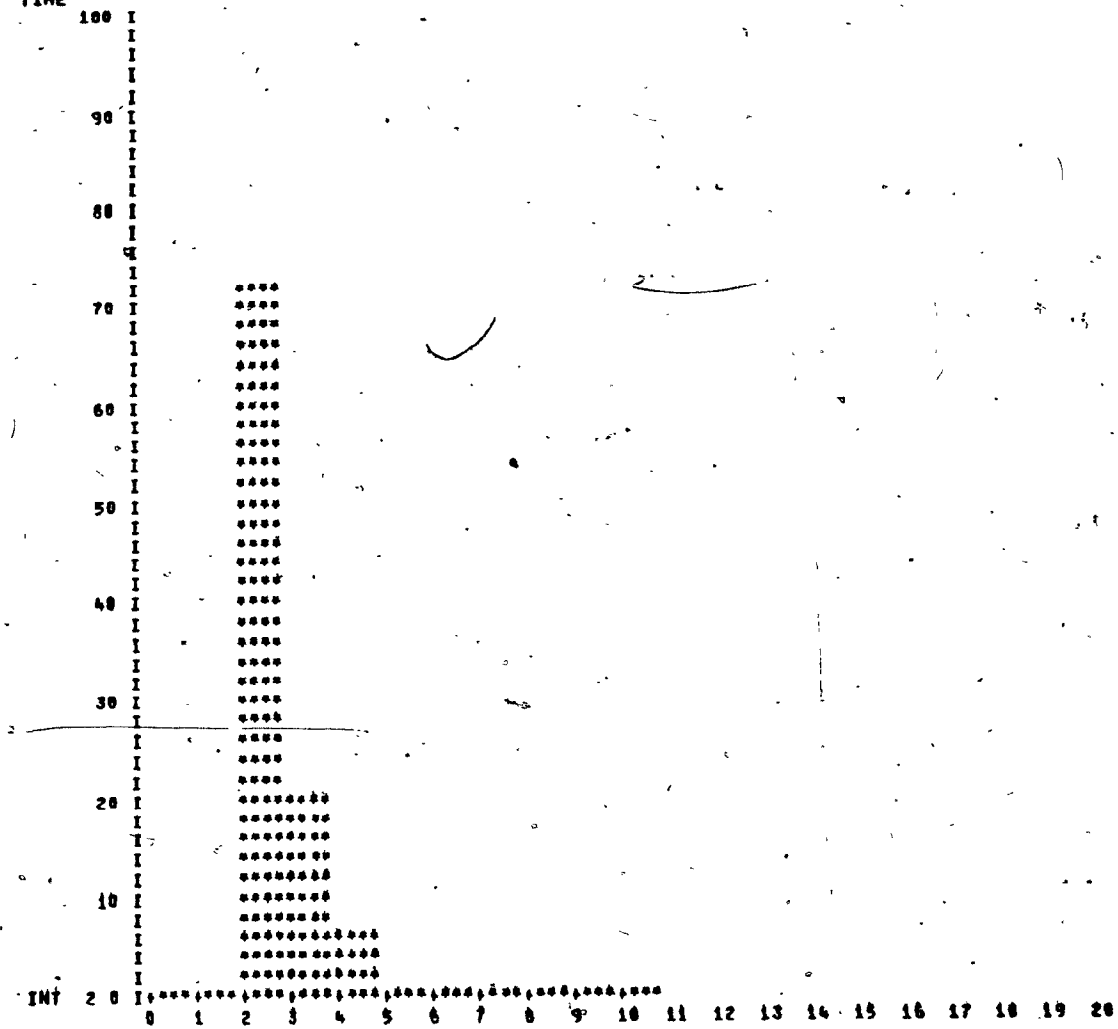
RUN 7 HIST 79



TOTAL TIME	247788	247788
AVERAGE PPS	4.89	

PP USE BY ORIGIN 00

RUN 7 HIST 00

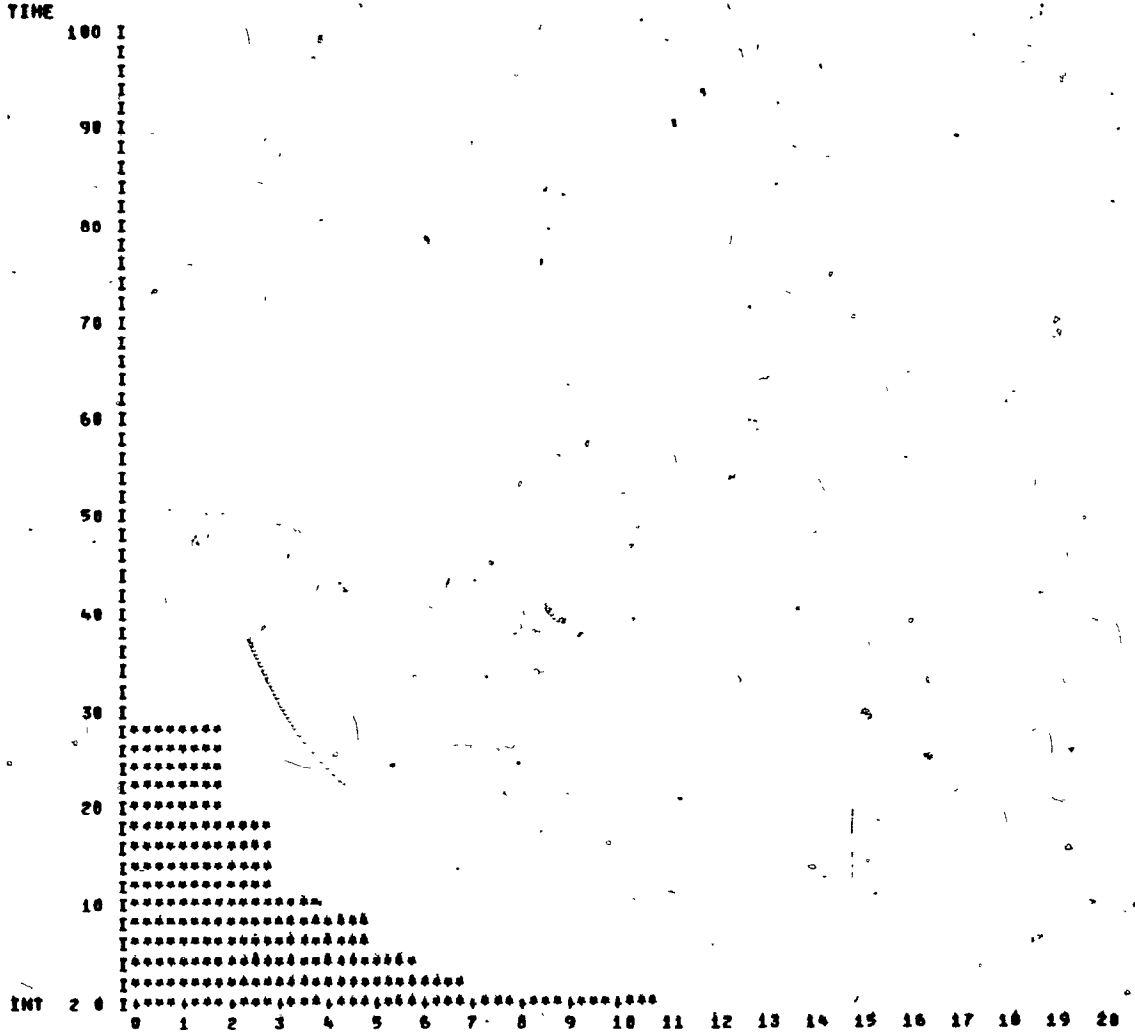


TOTAL TIME 247788 247745

AVERAGE PPS 2.38

PP USE BY ORIGIN 01

RUN 7 HIST 01

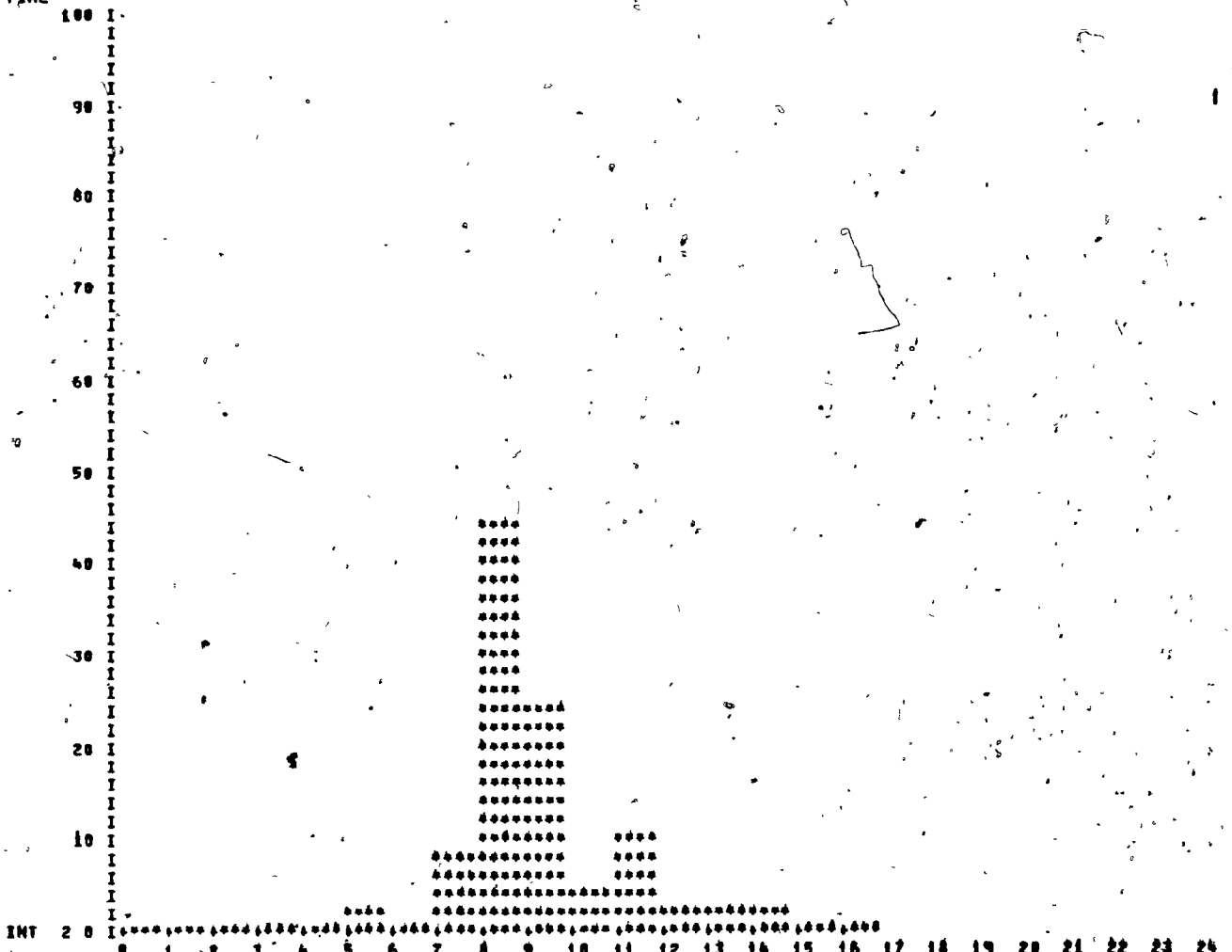


TOTAL TIME	247788	247788
AVERAGE PPS	1.70	

NUMBER OF CONTPOL POINTS IN USE 247.788 SECONDS

PER CENT OF TIME

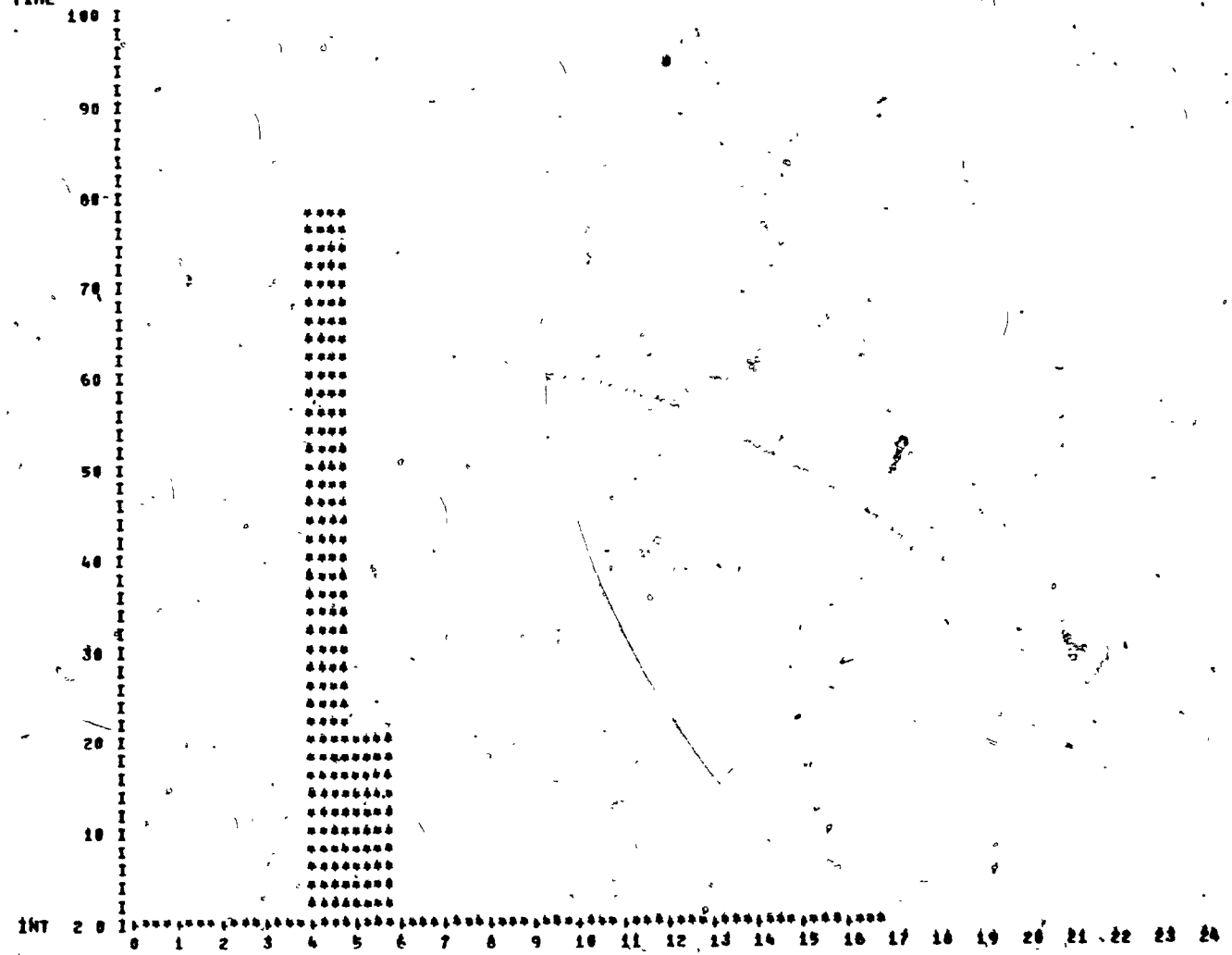
RUN 7 HIST 02



TOTAL TIME	247788	247788
AVERAGE CPS	9.82	

CONTROL POINT USE BY ORIGIN 88

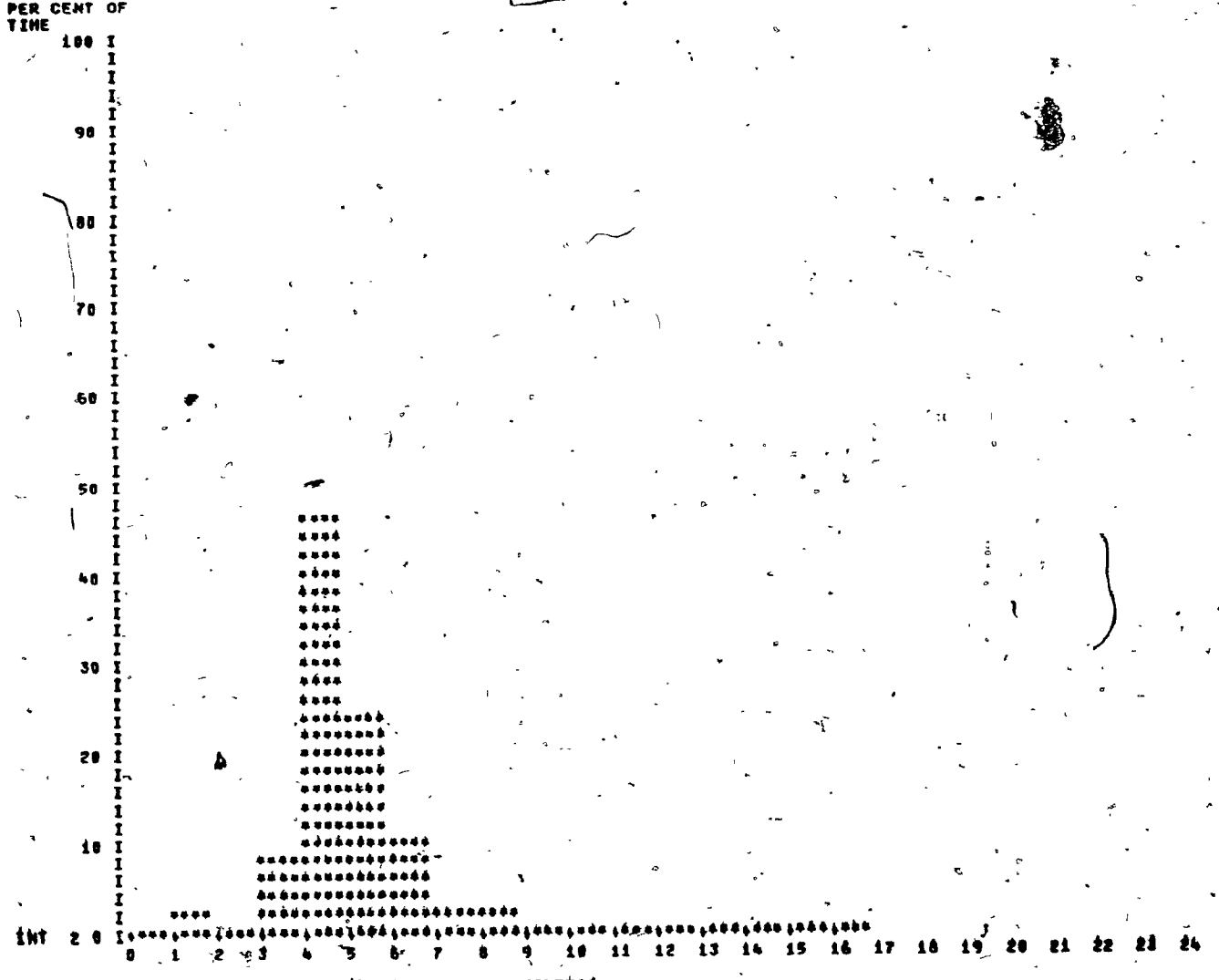
RUN 7 HIST 83



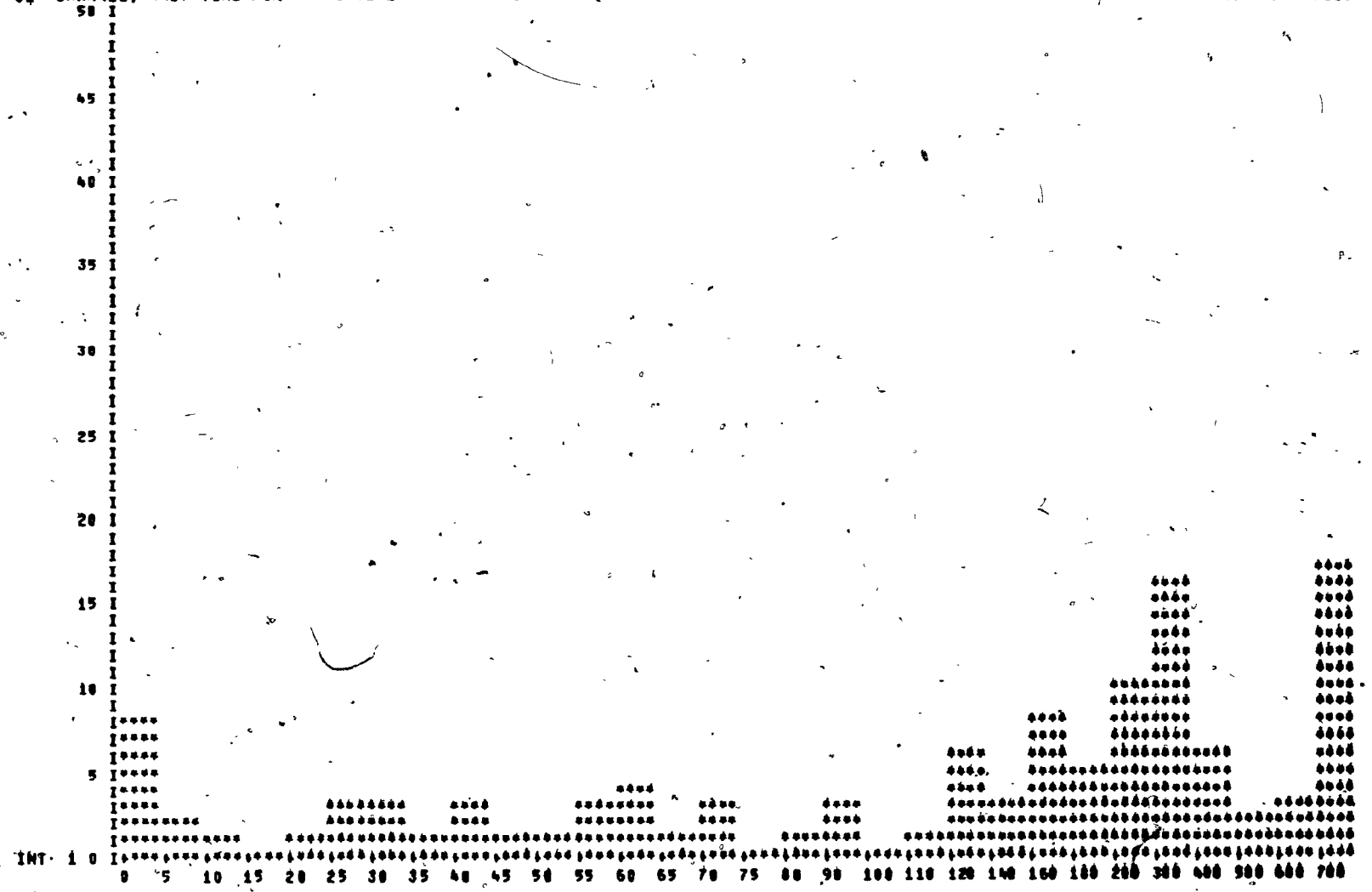
TOTAL TIME 247788 247788
AVERAGE CPS 4.22

CONTROL POINT USE BY ORIGIN 01

RUN 7 HIST 84



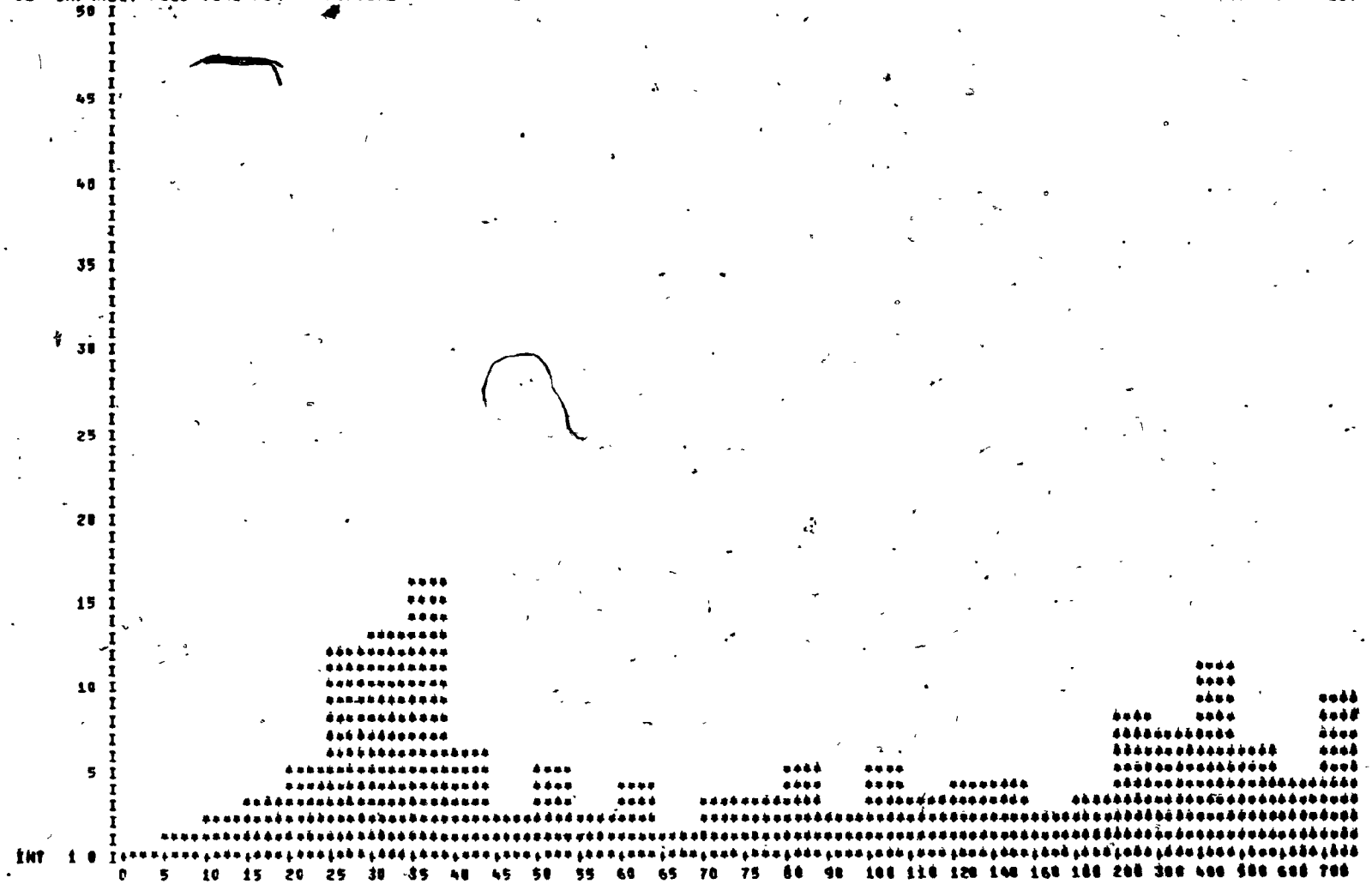
02 CHANNEL, WAIT TIME FOR PP ROUTINE QFM RUN 7 HIST 65



X Y-AXIS	7	2	1	0	1	3	3	1	3	1	1	3	3	1	3	0	1	3	0	1	5	3	7	4	9	14	5	2	3	19
X X-AXIS	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	0	2	1	3	2	6	14	7	3	5	53
GROUPED PERCENTAGES (X)					.32						1.31							2.16						8.98						87.22	

TOTAL Y-AXIS	116	MAXIMUM Y-AXIS	17
TOTAL X-AXIS	40686	MAXIMUM X-AXIS	21692
AVERAGE (X-AXIS)	350.74	STANDARD DEVIATION	412.37
MAXIMUM OBSERVED	2973		

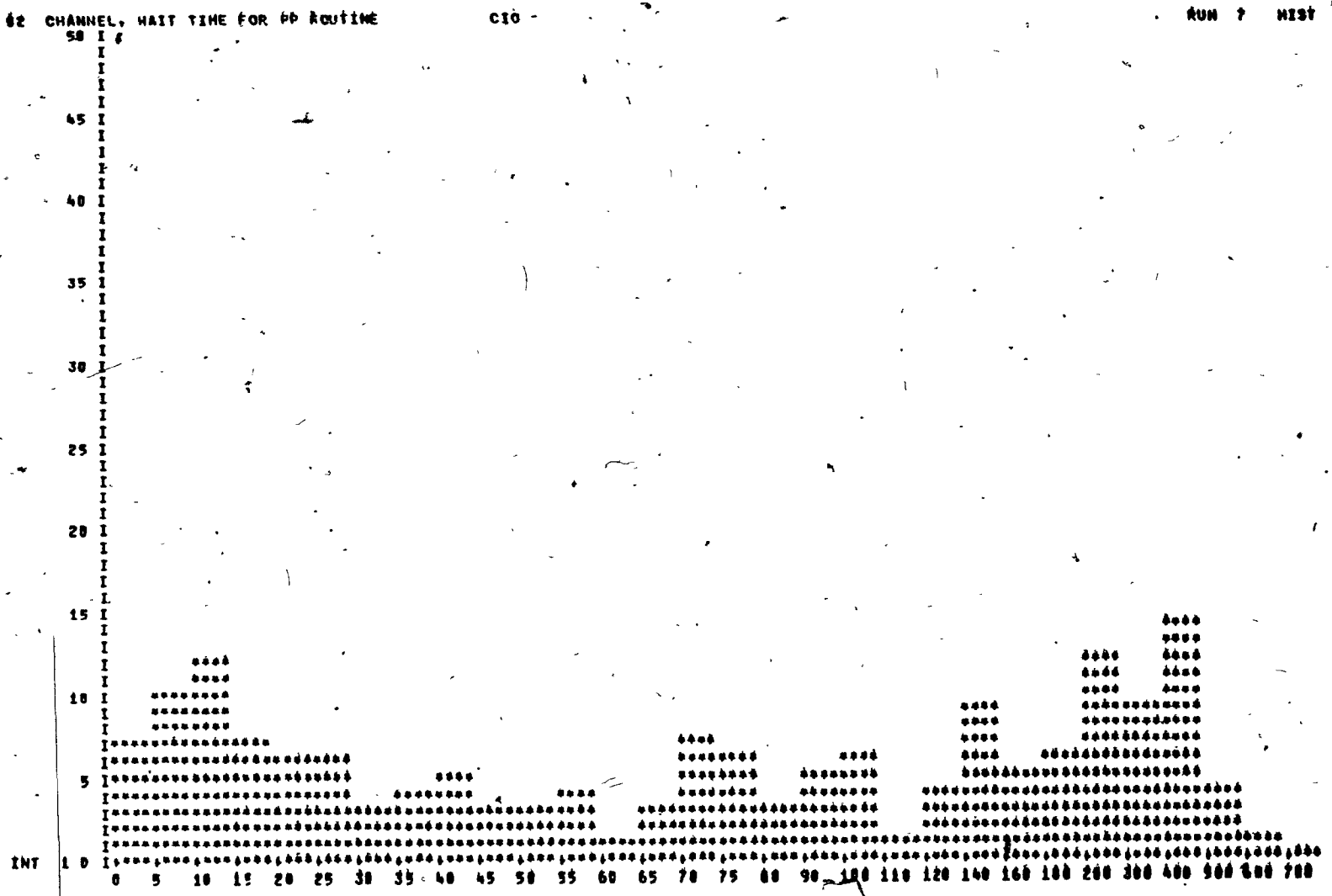
02 CHANNEL, HOLD TIME FOR PP ROUTINE QFM RUN 7 HIST 06



% Y-AXIS	0	1	1	2	3	8	9	11	4	1	3	1	3	1	2	2	3	1	3	2	3	3	1	2	5	5	7	4	3	6
% X-AXIS	0	0	0	0	0	1	1	2	1	0	1	0	1	0	1	1	1	1	2	1	2	2	1	2	7	8	16	11	9	27

GROUPED PERCENTAGES (X) 1.77 5.77 4.55 9.64 70.20

TOTAL Y-AXIS	151	MAXIMUM Y-AXIS	16
TOTAL X-AXIS	29905	MAXIMUM X-AXIS	8045
AVERAGE (X-AXIS)	198.05	STANDARD DEVIATION	243.05
MAXIMUM OBSERVED	1040		

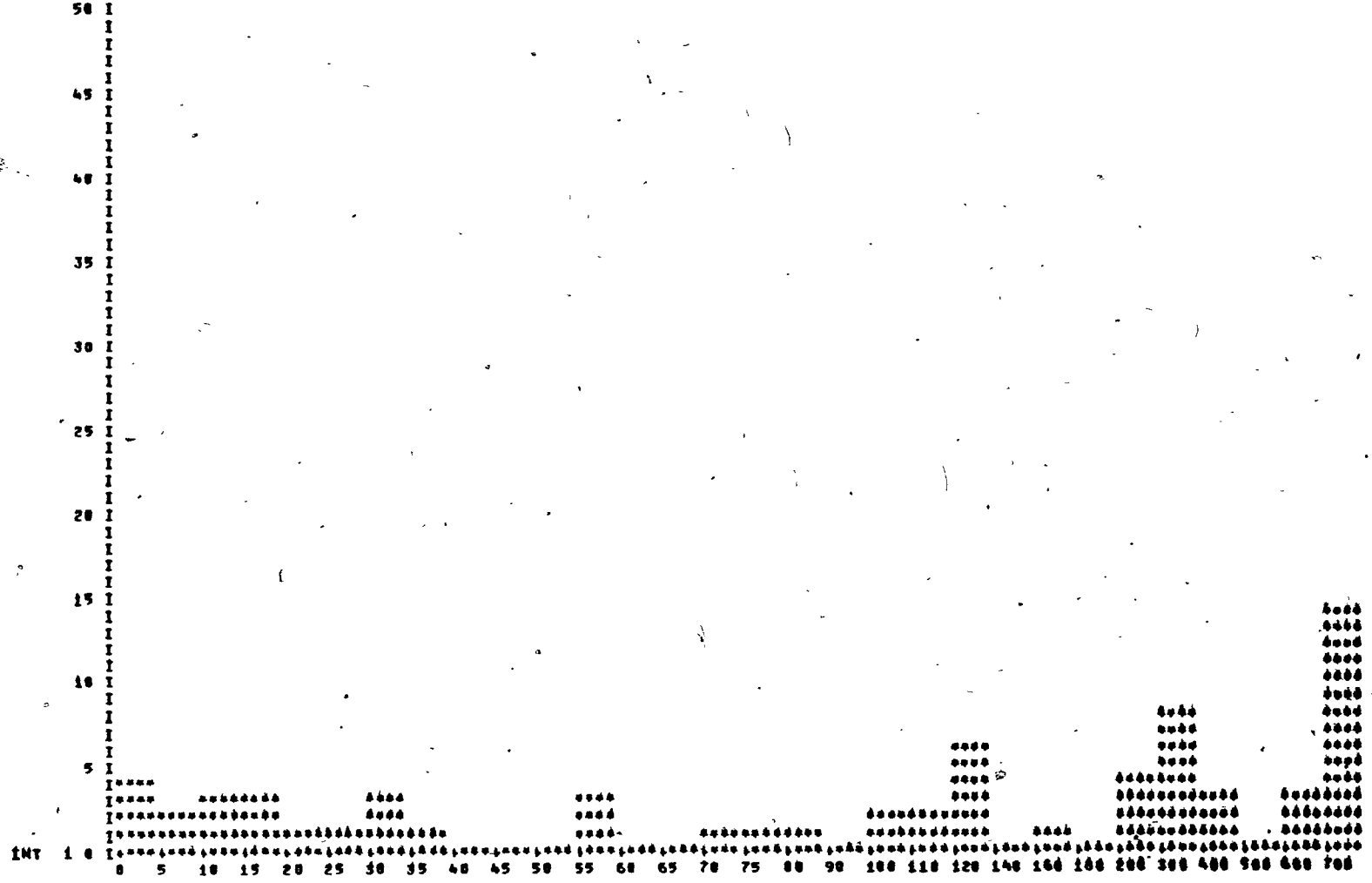

62 CHANNEL, WAIT TIME FOR PP ROUTINE
CIO -
RUN 7 HIST 87
INT 1 0
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 90 100 110 120 140 160 180 200 300 400 500 600 700
X Y-AXIS 4 6 7 4 4 4 2 2 3 2 2 2 1 2 4 4 2 3 4 1 2 5 3 4 7 8 8 2 1 0
X X-AXIS 0 0 1 1 1 1 0 1 1 1 1 1 0 1 2 2 1 2 3 1 2 6 4 8 12 13 27 10 3 0
GROUPED PERCENTAGES (X) 2.00 4.28 8.35 19.87 84.71
TOTAL Y-AXIS 166
TOTAL X-AXIS 23195
AVERAGE (X-AXIS) 139.73
MAXIMUM OBSERVED 635
MAXIMUM Y-AXIS 14
MAXIMUM X-AXIS 6214
STANDARD DEVIATION 151.49

02 CHANNEL, HOLD TIME FOR FP ROUTINE CIO RUN 7 HIST 88

% Y-AXIS	1	3	8	11	13	13	10	6	5	4	4	3	2	2	2	2	3	2	1	1	1	1	1	0	0	1	0	0	0	0	
% X-AXIS	0	1	2	4	6	8	7	5	5	4	5	4	3	3	3	3	7	5	2	2	4	3	2	1	2	5	3	2	0	0	
GROUPED PERCENTAGES (X)					20.90						29.21						22.41						14.42							13.06	

TOTAL Y-AXIS	1183	MAXIMUM Y-AXIS	151
TOTAL X-AXIS	53304	MAXIMUM X-AXIS	4064
AVERAGE (X-AXIS)	45.06	STANDARD DEVIATION	52.47
MAXIMUM OBSERVED	595		

82 CHANNEL. WAIT TIME FOR PP ROUTINE IAJ RUN 7. HIST 89



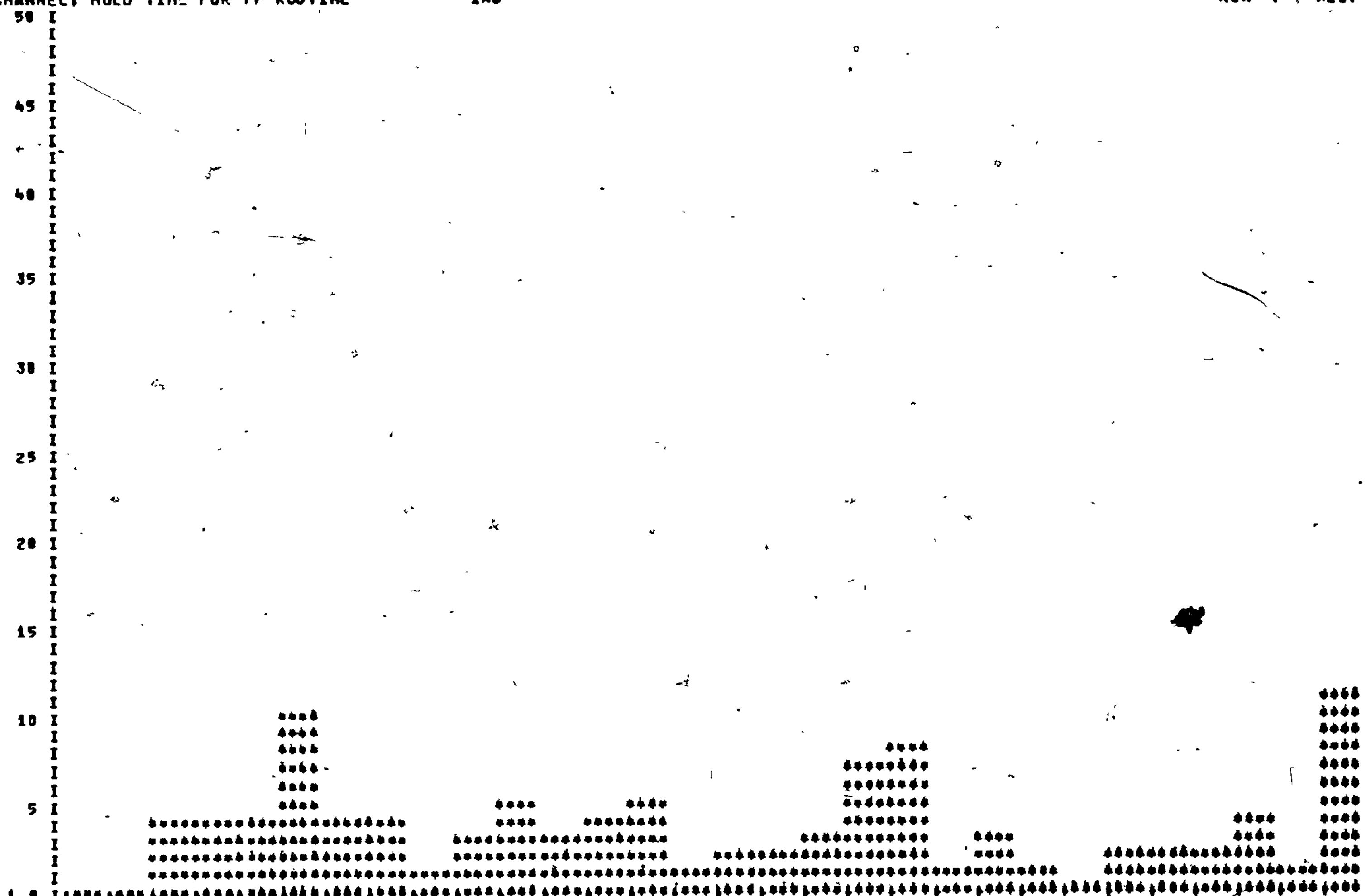
% Y-AXIS	6	3	6	6	1	1	6	1	0	0	0	6	0	0	1	1	1	0	3	3	9	0	1	0	6	12	6	0	6	21
% X-AXIS	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	3	0	1	0	3	9	9	0	6	69

GROUPED PERCENTAGES (X) .53 1.81 .78 4.98 93.10

TOTAL Y-AXIS	67	MAXIMUM Y-AXIS	14
TOTAL X-AXIS	30805	MAXIMUM X-AXIS	28790
AVERAGE (X-AXIS)	447.84	STANDARD DEVIATION	956.58
MAXIMUM OBSERVED	2973		

144

82 CHANNEL, HOLD TIME FOR PP ROUTINE 1AJ RUN 7 , HIST 90

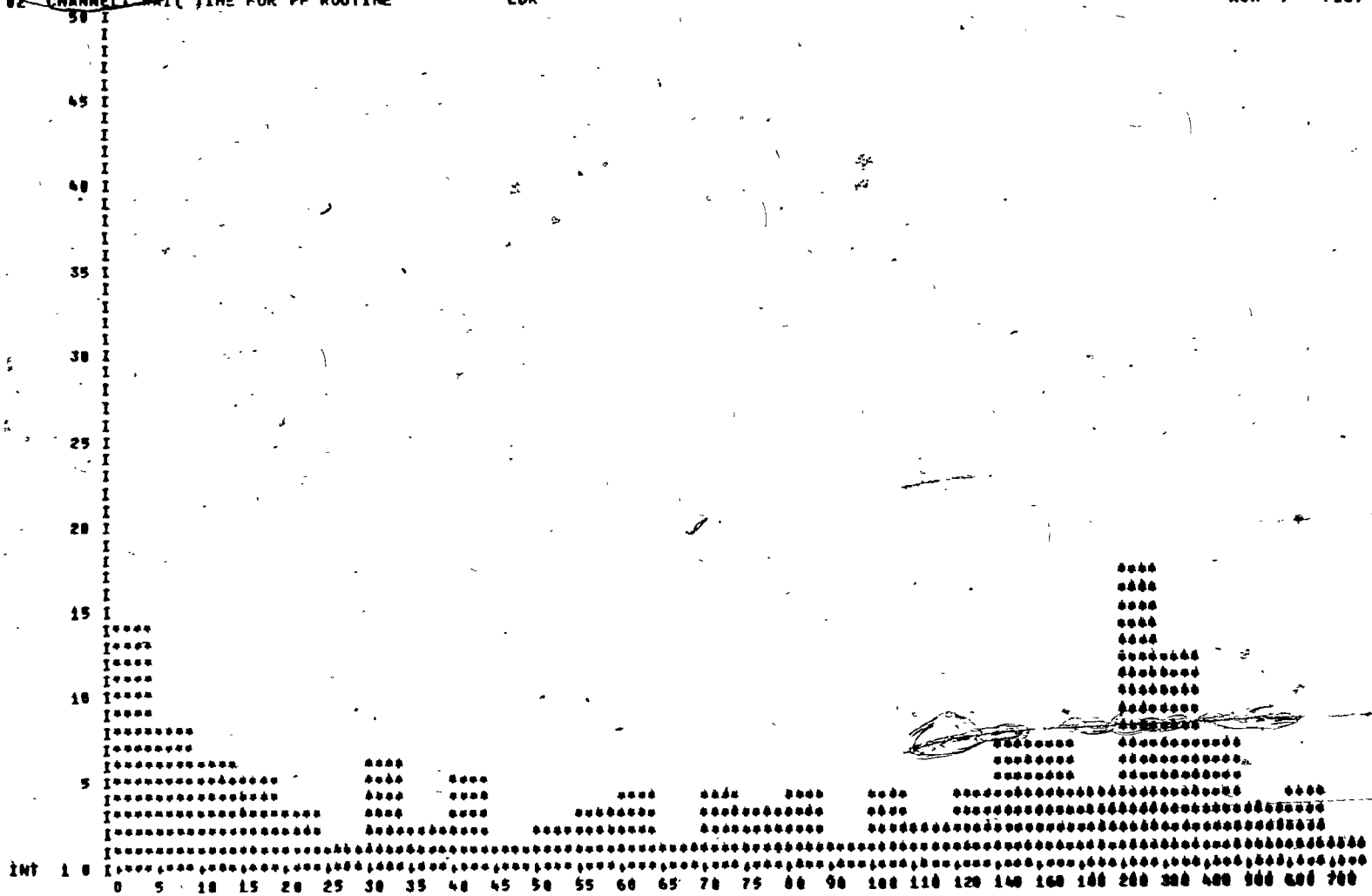


INT	0	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	90	100	110	120	140	160	180	200	300	400	500	600	700
% Y-AXIS	0	0	4	4	4	10	4	4	1	3	5	3	4	5	1	2	2	3	7	8	1	3	1	0	2	2	2	4	1	11
% X-AXIS	0	0	0	0	0	1	1	1	0	1	1	1	1	2	0	1	1	1	3	4	1	2	1	0	2	3	4	10	3	54

GROUPED PERCENTAGES (X) 2.23 4.18 5.92 11.05 76.02

TOTAL Y-AXIS	101	MAXIMUM Y-AXIS	11
TOTAL X-AXIS	21556	MAXIMUM X-AXIS	11650
AVERAGE (X-AXIS)	213.43	STANDARD DEVIATION	323.12
MAXIMUM OBSERVED	1255		

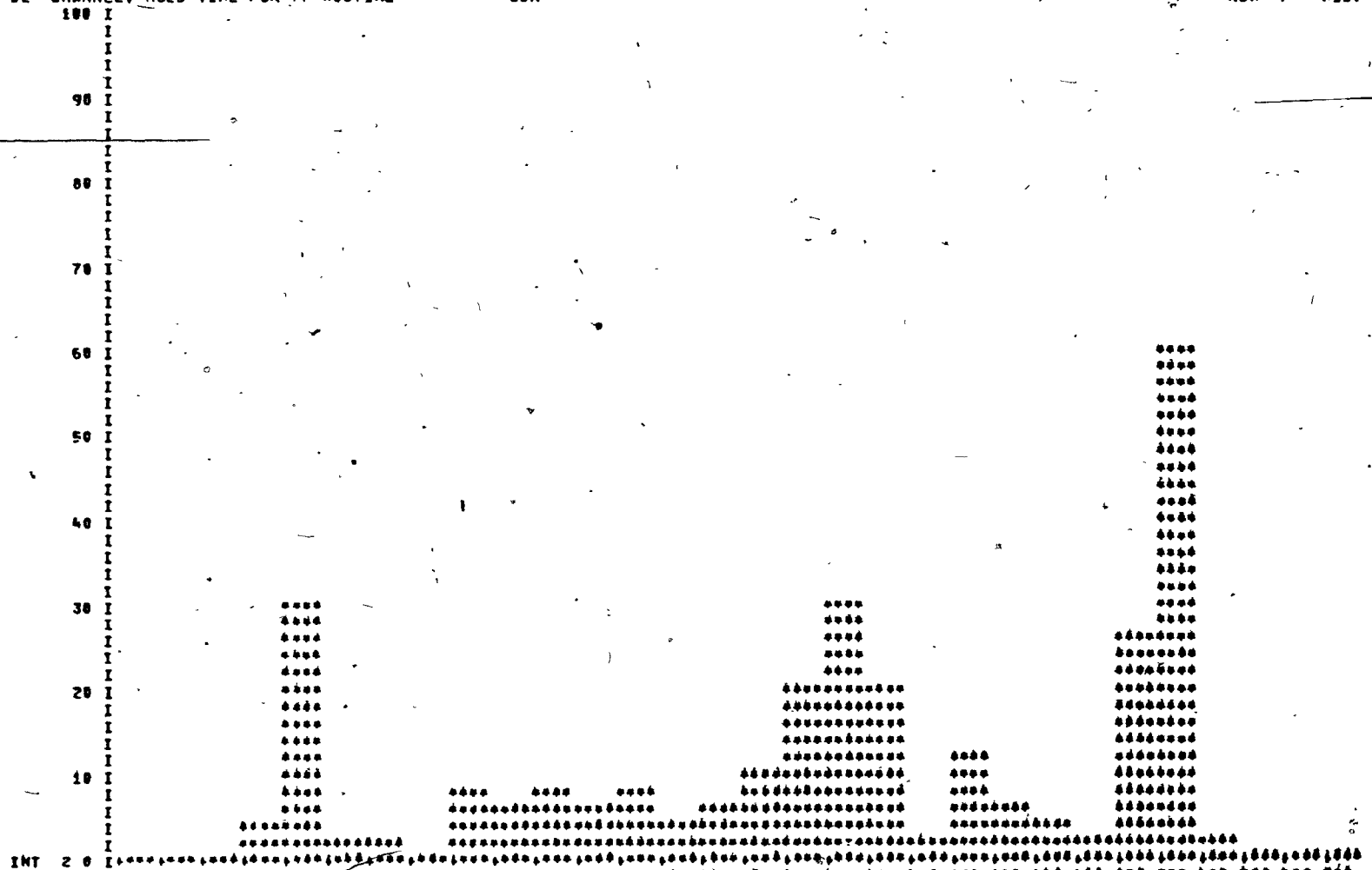
02 CHANNEL WAIT TIME FOR PP ROUTINE LDR RUN 7 HIST 03



% Y-AXIS	10	6	4	3	2	1	4	1	3	1	1	2	3	1	3	2	3	1	3	1	3	5	5	3	12	6	5	2	3	1
% X-AXIS	0	0	0	0	0	0	1	0	1	0	0	1	1	0	1	1	1	0	2	1	2	5	5	3	10	10	10	7	11	3
GROUPED PERCENTAGES (%)				1.39							3.46							5.52						18.04					71.59	

TOTAL Y-AXIS	145	MAXIMUM Y-AXIS	17
TOTAL X-AXIS	23039	MAXIMUM X-AXIS	4333
AVERAGE (X-AXIS)	158.89	STANDARD DEVIATION	168.43
MAXIMUM OBSERVED	744		

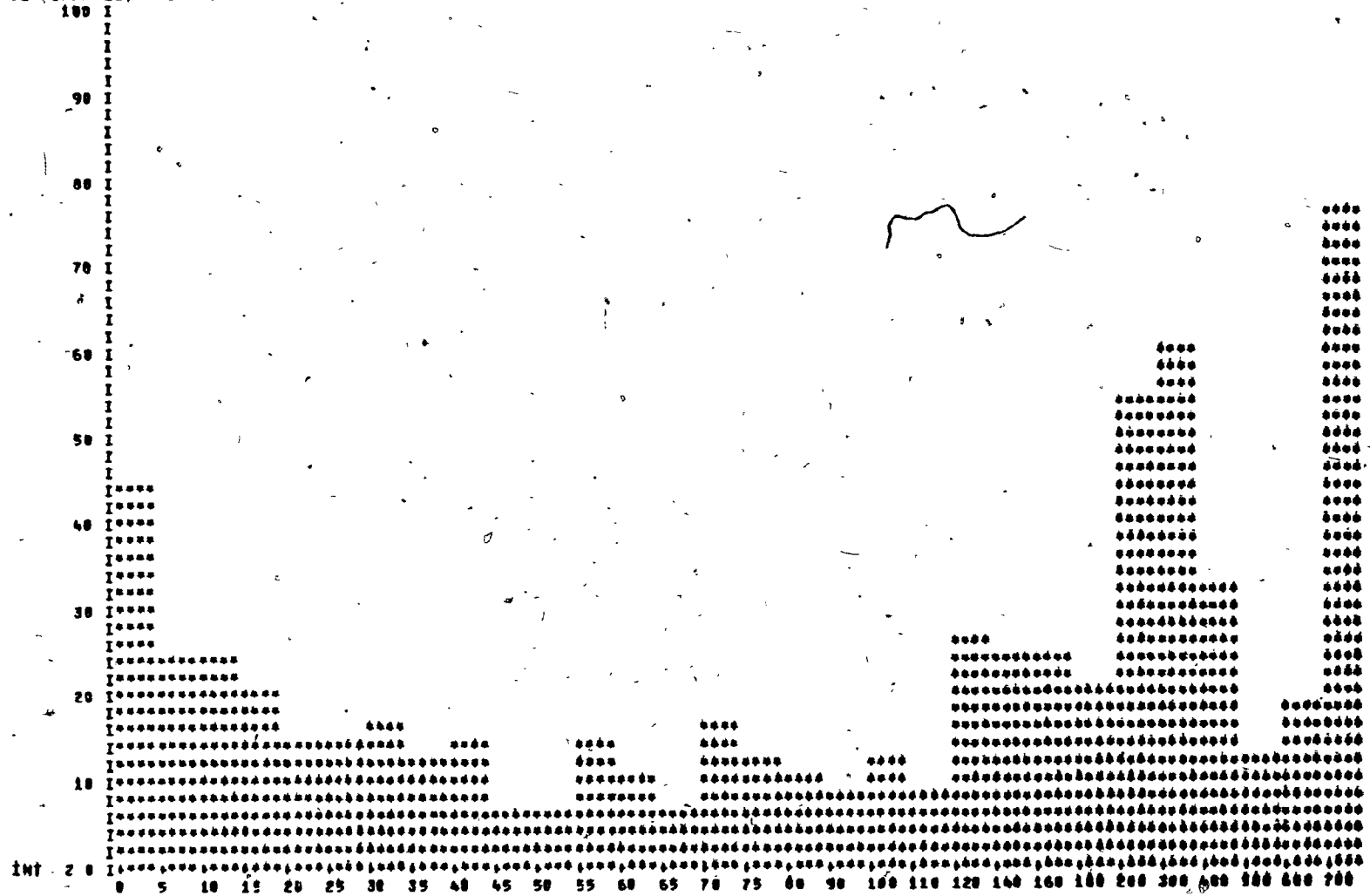
02 CHANNEL, HOLD TIME FOR FP ROUTINE LDR RUN 7 HIST 94



X Y-AXIS	0	0	0	2	11	1	1	0	3	2	3	2	3	1	2	4	7	11	7	1	4	2	2	1	9	21	1	0	0	0
X X-AXIS	0	0	0	0	1	0	0	0	1	1	1	1	1	1	1	2	4	6	5	0	3	2	2	1	17	46	2	0	1	0
GROUPED PERCENTAGES (X)					1.77						3.60						14.96						13.86						69.82		

TOTAL Y-AXIS	294	MAXIMUM Y-AXIS	61
TOTAL X-AXIS	45211	MAXIMUM X-AXIS	28794
AVERAGE (X-AXIS)	153.78	STANDARD DEVIATION	123.27
MAXIMUM OBSERVED	648		

02 CHANNEL, WAIT TIME FOR PP ROUTINE PRU RUN 7 HIST 118

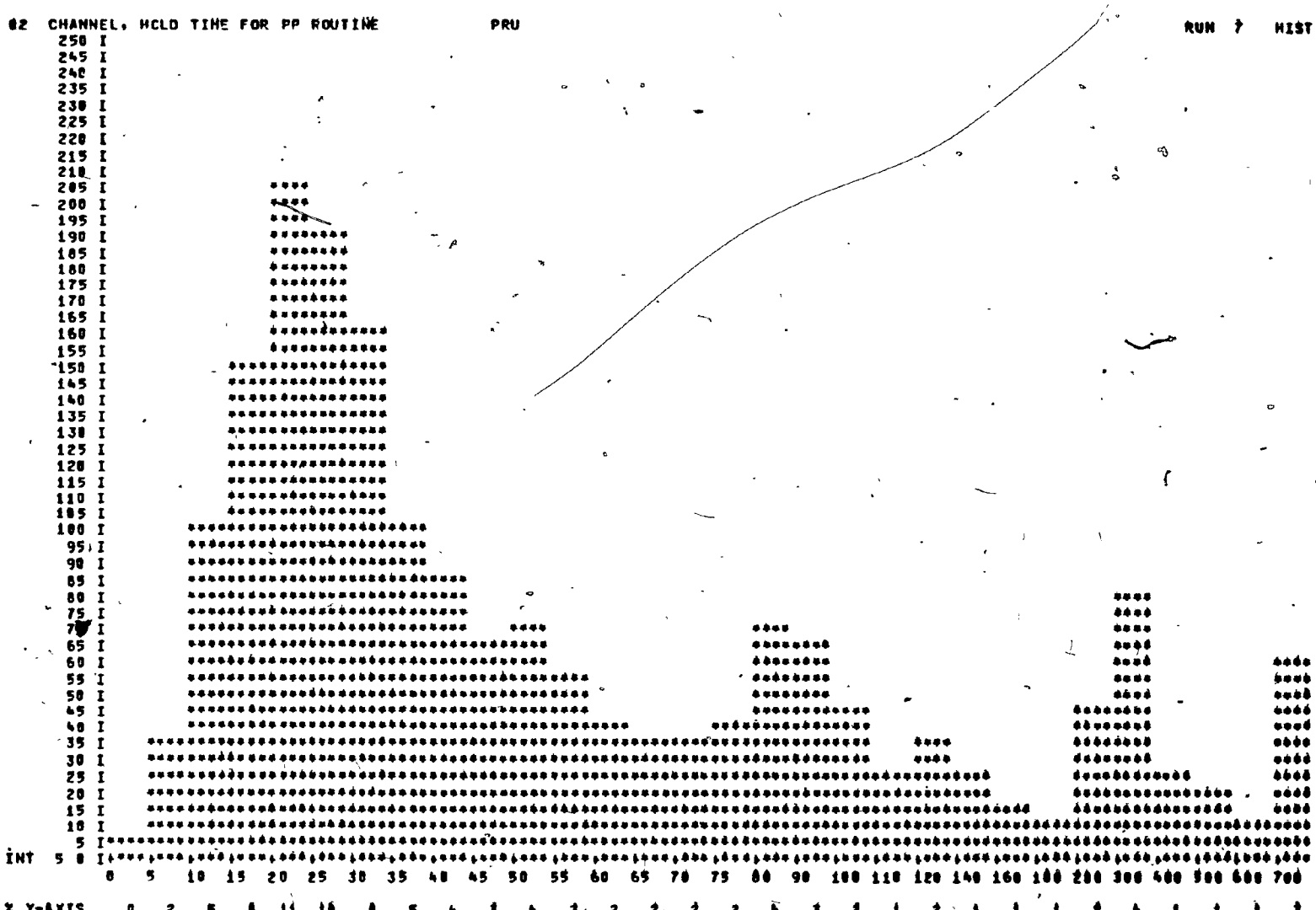


X Y-AXIS	7	4	4	3	2	2	3	2	2	1	1	2	2	1	3	2	2	1	2	1	4	4	4	3	8	9	5	2	3	12
X X-AXIS	0	0	0	0	0	0	0	0	3	0	0	0	0	0	1	0	0	0	1	0	2	2	2	2	6	10	7	3	6	56

GROUPED PERCENTAGES (X) .79 1.48 2.36 8.39 87.82

TOTAL Y-AXIS	651	MAXIMUM Y-AXIS	76
TOTAL X-AXIS	207648	MAXIMUM X-AXIS	113832
AVERAGE (X-AXIS)	318.96	STANDARD DEVIATION	452.75
MAXIMUM OBSERVED	3136		

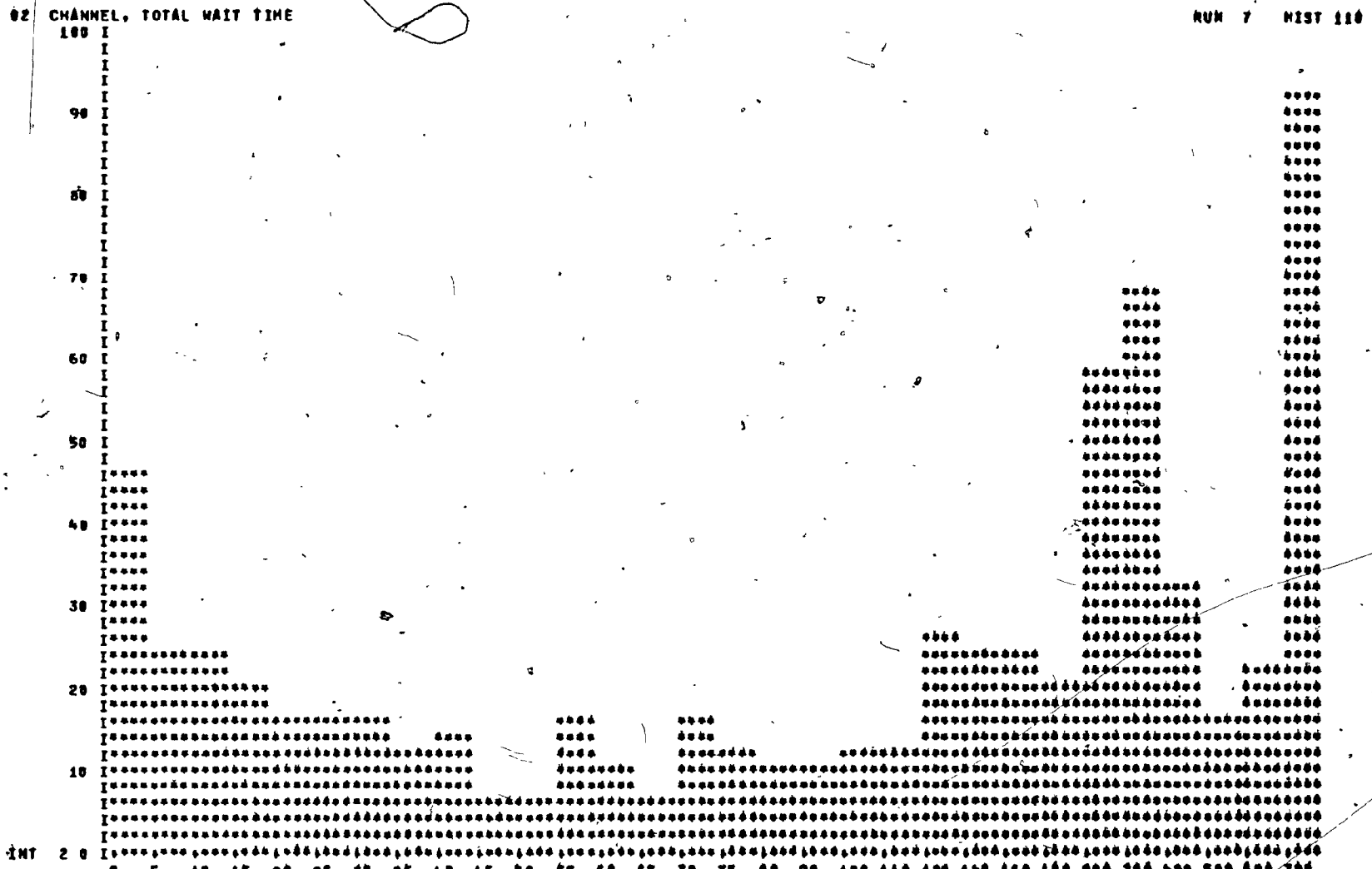
148



% Y-AXIS	0	2	5	8	11	10	8	5	4	3	4	3	2	2	2	2	4	3	2	1	2	1	1	1	2	4	1	1	1	3
% X-AXIS	0	0	1	1	2	2	2	2	2	1	2	1	1	1	1	1	3	3	2	1	2	2	1	1	6	13	5	6	4	28

GROUPED PERCENTAGES (X) 6.49 10.58 18.81 9.93 62.19

TOTAL Y-AXIS	1956	MAXIMUM Y-AXIS	208
TOTAL X-AXIS	214228	MAXIMUM X-AXIS	59388
AVERAGE (X-AXIS)	109.52	STANDARD DEVIATION	192.87
MAXIMUM OBSERVED	1623		

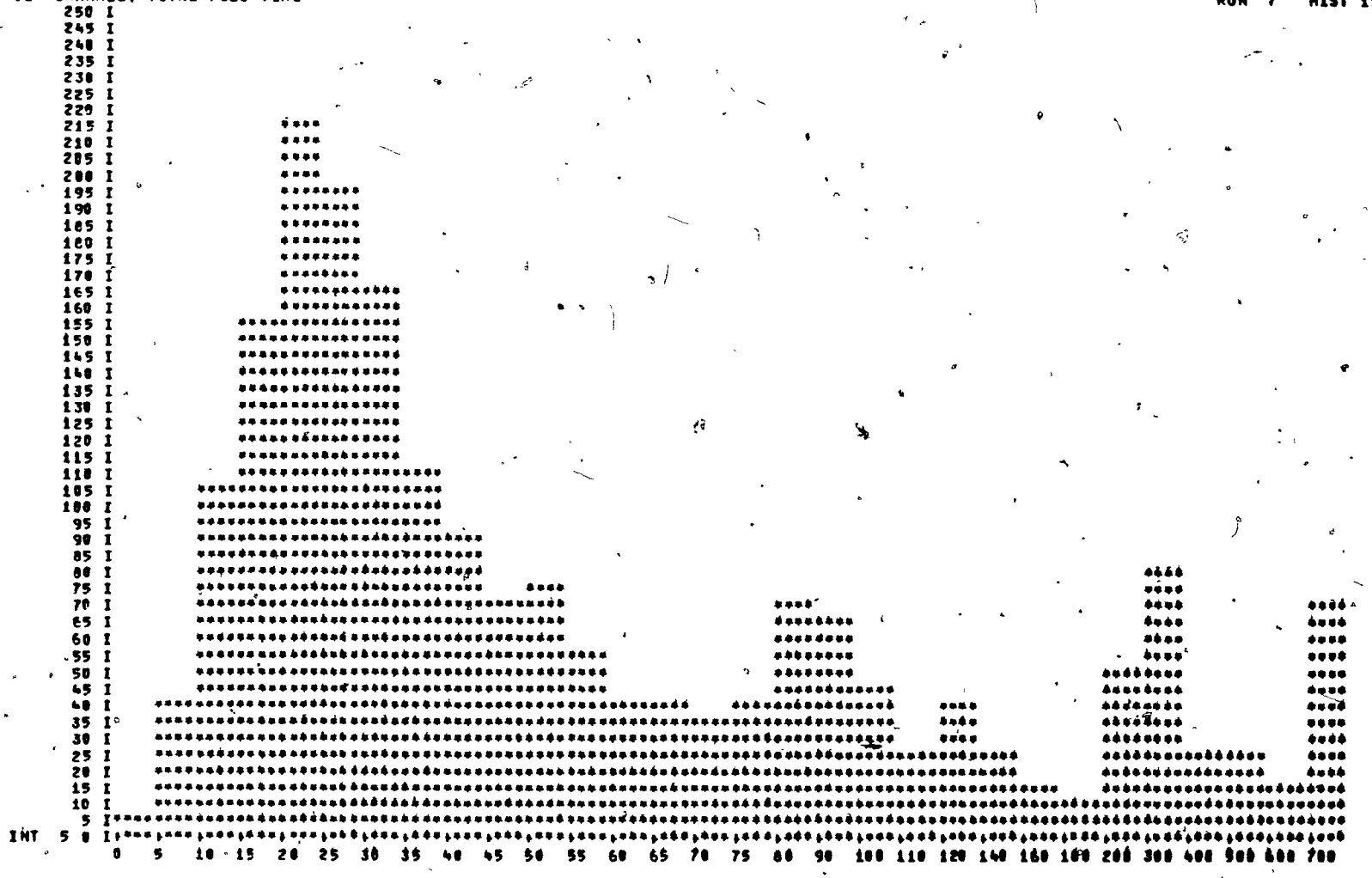


% Y-AXIS	7	4	4	3	2	2	2	2	2	1	1	2	1	1	2	2	2	1	2	2	4	3	4	3	8	16	8	2	3	11
% X-AXIS	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1	1	2	2	2	6	10	6	4	6	37

GROUPED PERCENTAGES (X) .71 1.32 2.11 7.48 88.48

TOTAL Y-AXIS	788	MAXIMUM Y-AXIS	92
TOTAL X-AXIS	240210	MAXIMUM X-AXIS	136373
AVERAGE (X-AXIS)	343.16	STANDARD DEVIATION	471.34
MAXIMUM OBSERVED	3136		

02 CHANNEL, TOTAL HOLD TIME RUN 7 HIST 119

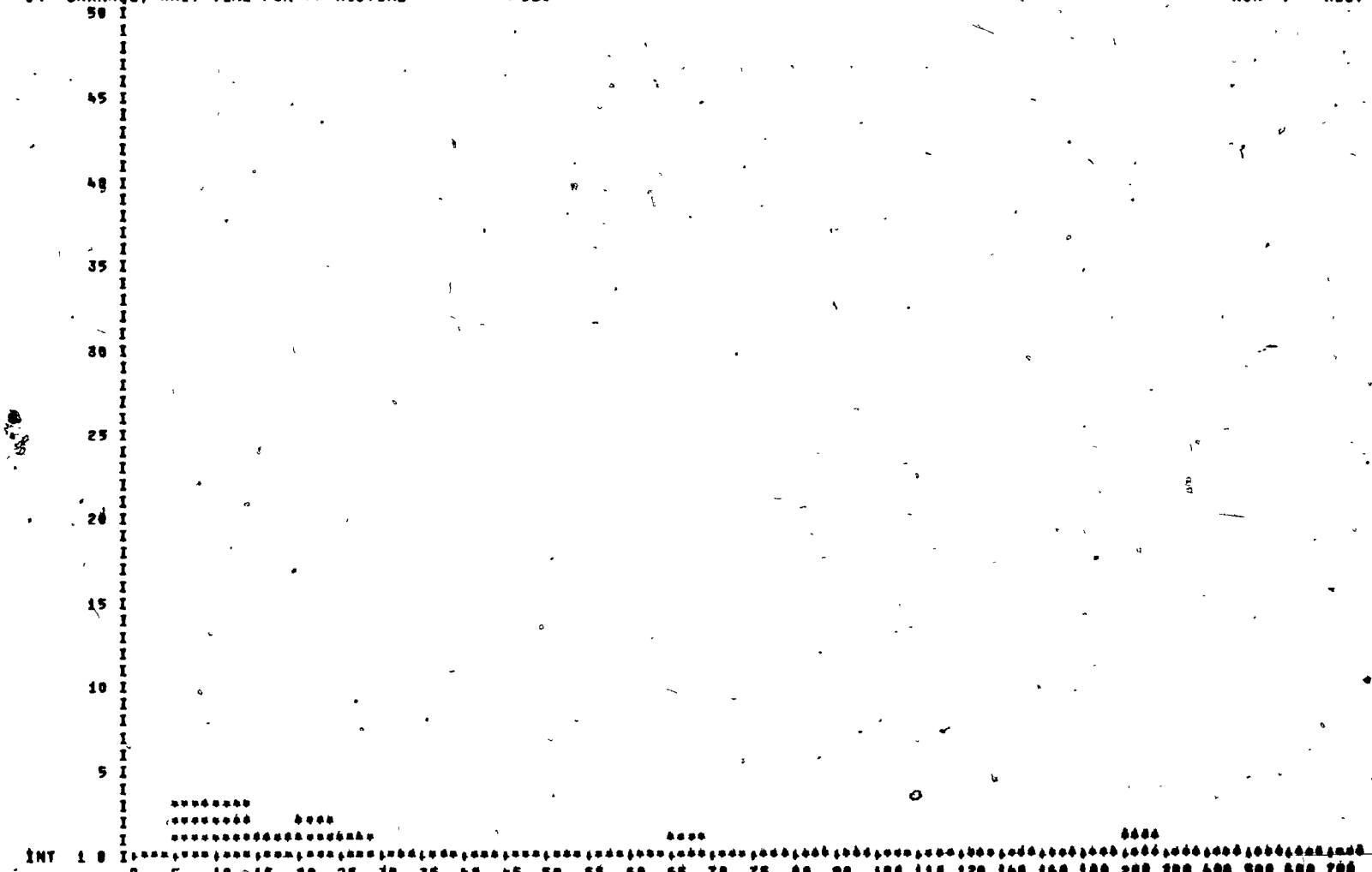


% Y-AXIS	0	2	5	8	11	10	8	5	4	4	4	3	2	2	2	2	4	3	2	1	2	1	1	1	2	4	1	1	1	4
% X-AXIS	0	0	1	1	2	2	2	2	2	1	2	1	1	1	1	1	3	3	2	1	2	2	1	1	1	6	12	5	7	5	10

GROUPED PERCENTAGES (X) 6.01 9.96 9.85 9.29 64.90

TOTAL Y-AXIS	2052	MAXIMUM Y-AXIS	216
TOTAL X-AXIS	239868	MAXIMUM X-AXIS	72855
AVERAGE (X-AXIS)	116.50	STANDARD DEVIATION	284.38
MAXIMUM OBSERVED	1623		

04 CHANNEL, WAIT TIME FOR PP ROUTINE CIO RUN 7 HIST 120

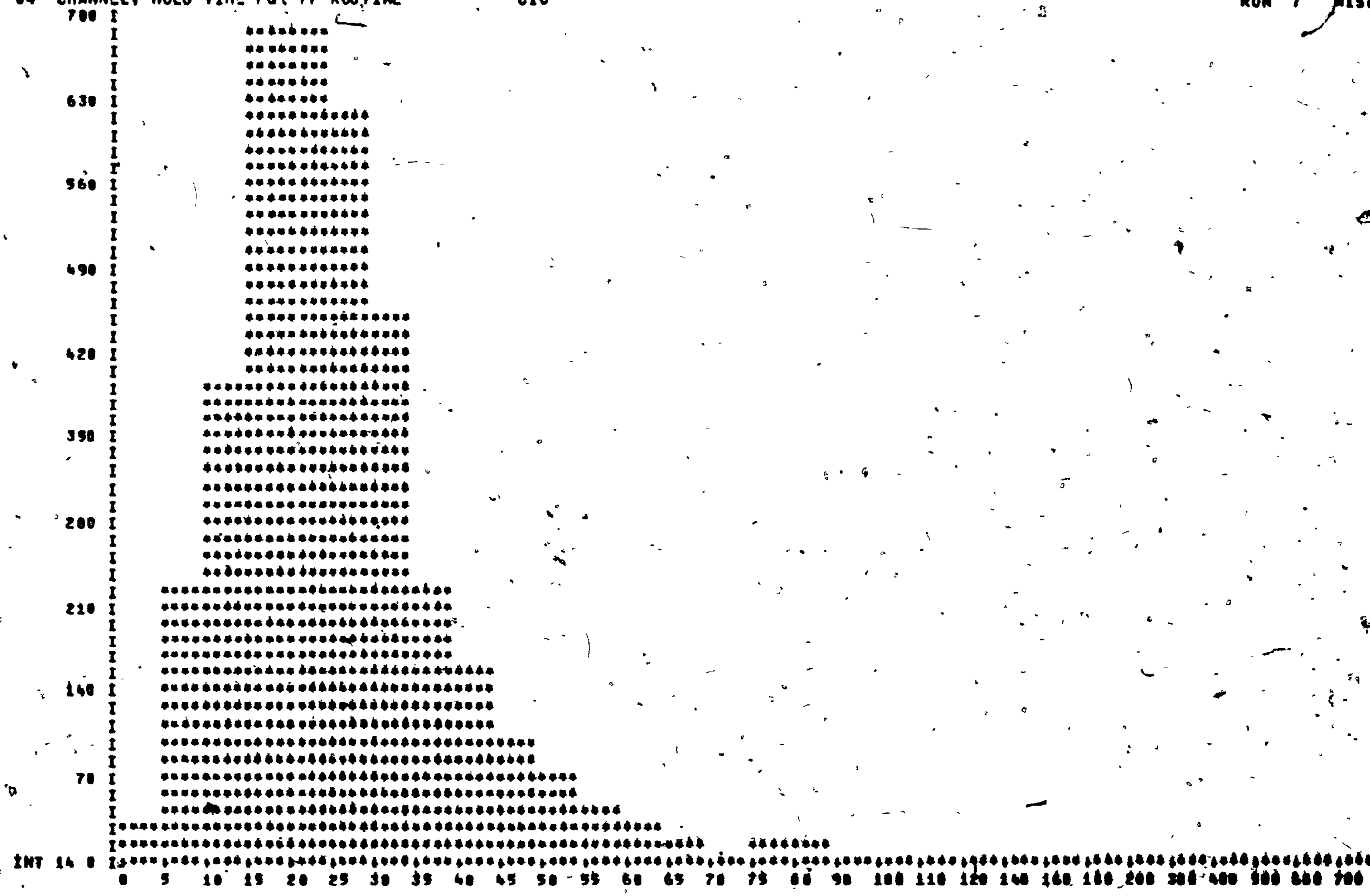


% Y-AXIS	0	25	25	8	17	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	0	5	8	3	0	5	0	0	0	0	0	0	0	15	0	0	0	0	0	0	0	0	0	0	55	0	0	0	0	0

GROUPED PERCENTAGES (X) 30.17 0.00 14.56 0.00 55.27

TOTAL Y-AXIS	12	MAXIMUM Y-AXIS	3
TOTAL X-AXIS	474	MAXIMUM X-AXIS	262
AVERAGE (X-AXIS)	39.51	STANDARD DEVIATION	68.96
MAXIMUM OBSERVED	262		

04 CHANNEL, HOLO TIME FOR PP ROUTINE CIO RUN 7 HIST 121

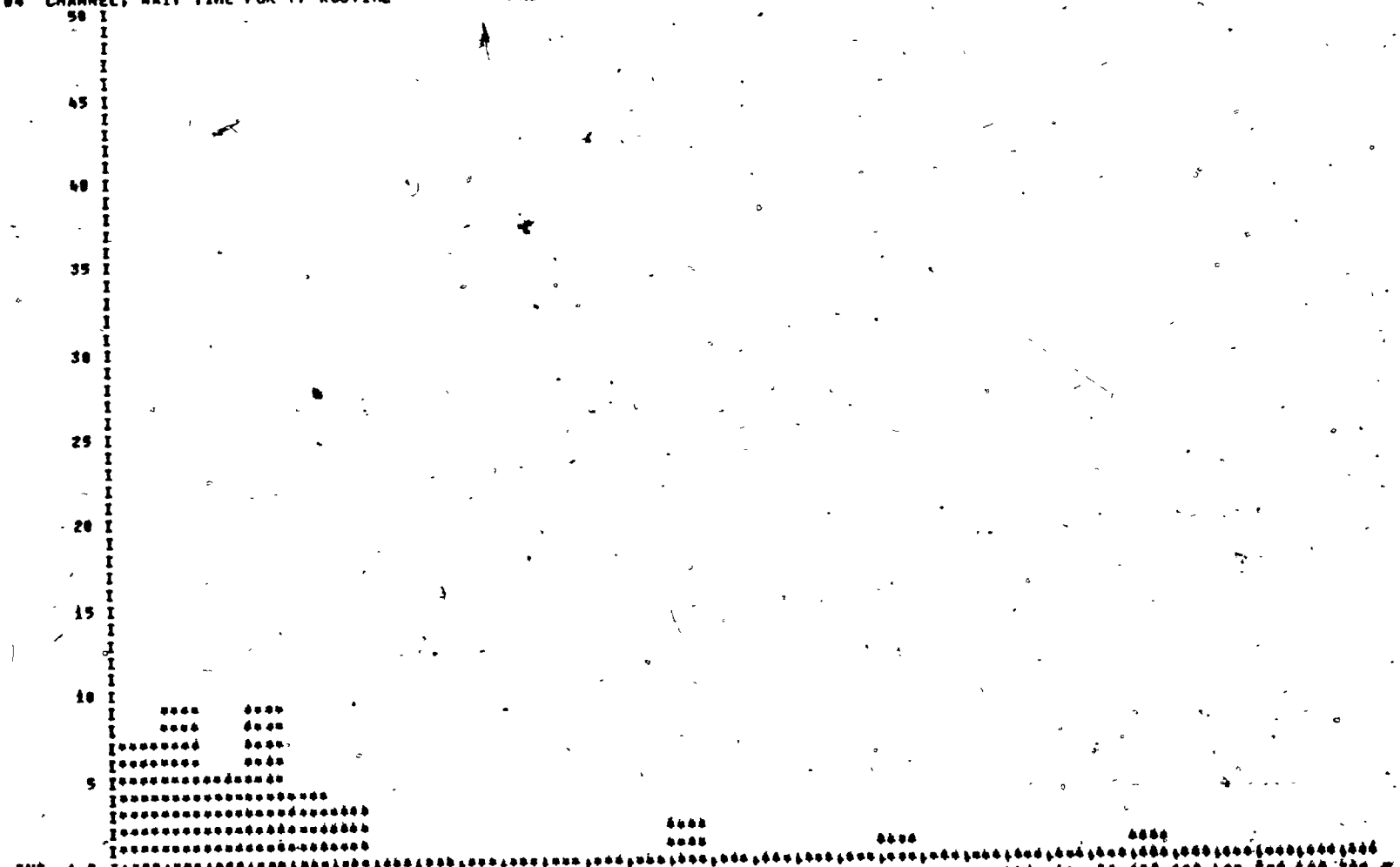


	0	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	90	100	110	120	140	160	180	200	300	400	500	600	700
% Y-AXIS	1	6	10	18	18	16	12	6	4	3	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	0	2	5	11	15	16	14	8	7	5	4	3	2	1	1	1	1	1	1	0	0	0	1	0	0	2	0	1	0	0

GROUPED PERCENTAGES (X) 40.30 40.00 6.70 2.04 2.90

TOTAL Y-AXIS	3897	MAXIMUM Y-AXIS	692
TOTAL X-AXIS	104726	MAXIMUM X-AXIS	16022
AVERAGE (X-AXIS)	26.87	STANDARD DEVIATION	22.73
MAXIMUM OBSERVED	539		

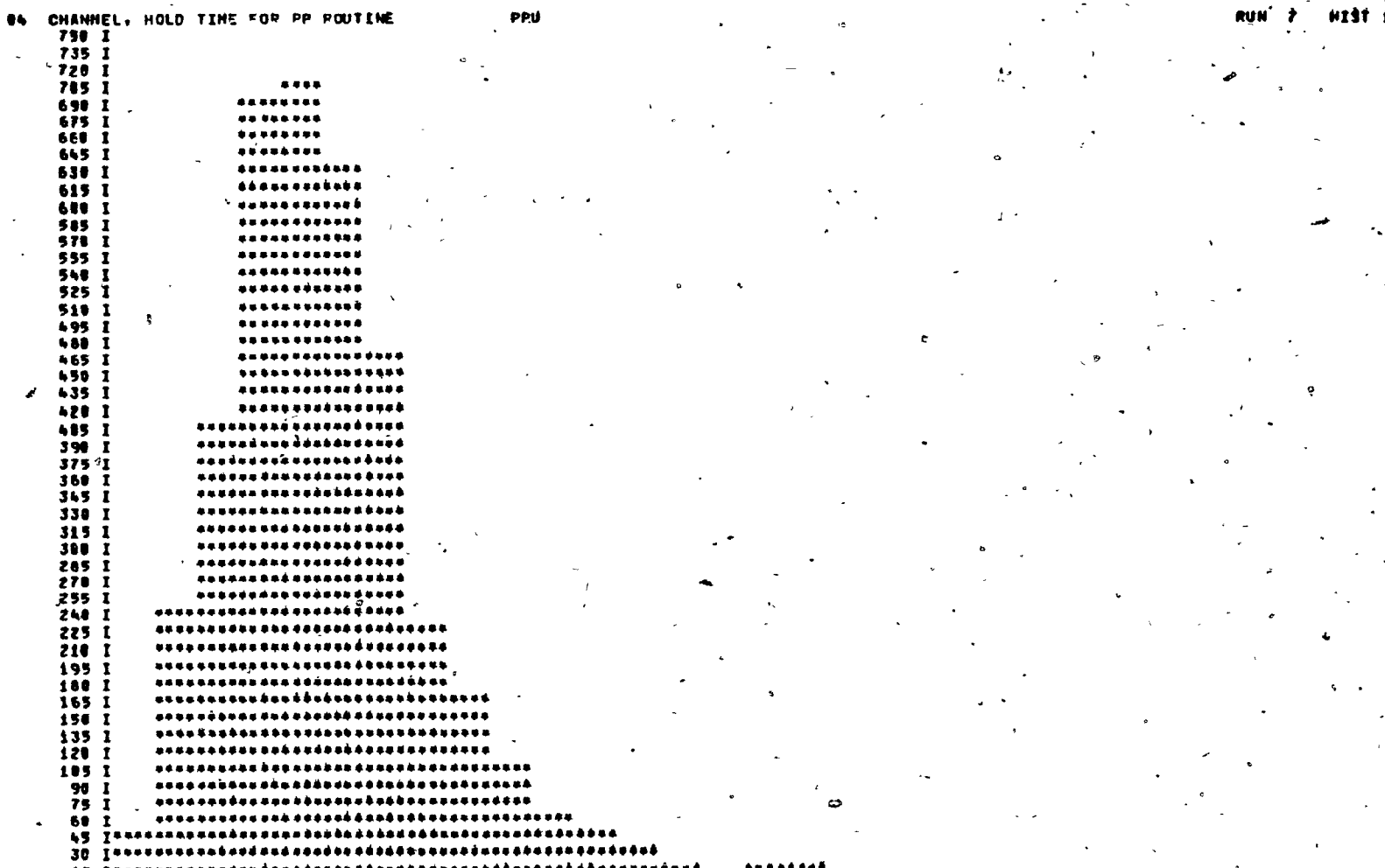
04 CHANNEL, WAIT TIME FOR FP ROUTINE PRU RUN 7 HIST 129



% Y-AXIS	17	22	12	22	10	7	0	0	0	0	0	0	0	5	0	0	0	0	2	0	0	0	0	2	0	0	0	0	0	
% X-AXIS	2	7	6	16	8	6	0	0	0	0	0	0	0	14	0	0	0	0	11	0	0	0	0	27	0	0	0	0	0	

GROUPED PERCENTAGES (X) 47.47 8.08 14.24 11.25 27.84

TOTAL Y-AXIS	41	MAXIMUM Y-AXIS	5
TOTAL X-AXIS	969	MAXIMUM X-AXIS	262
AVERAGE (X-AXIS)	23.63	STANDARD DEVIATION	42.72
MAXIMUM OBSERVED	262		

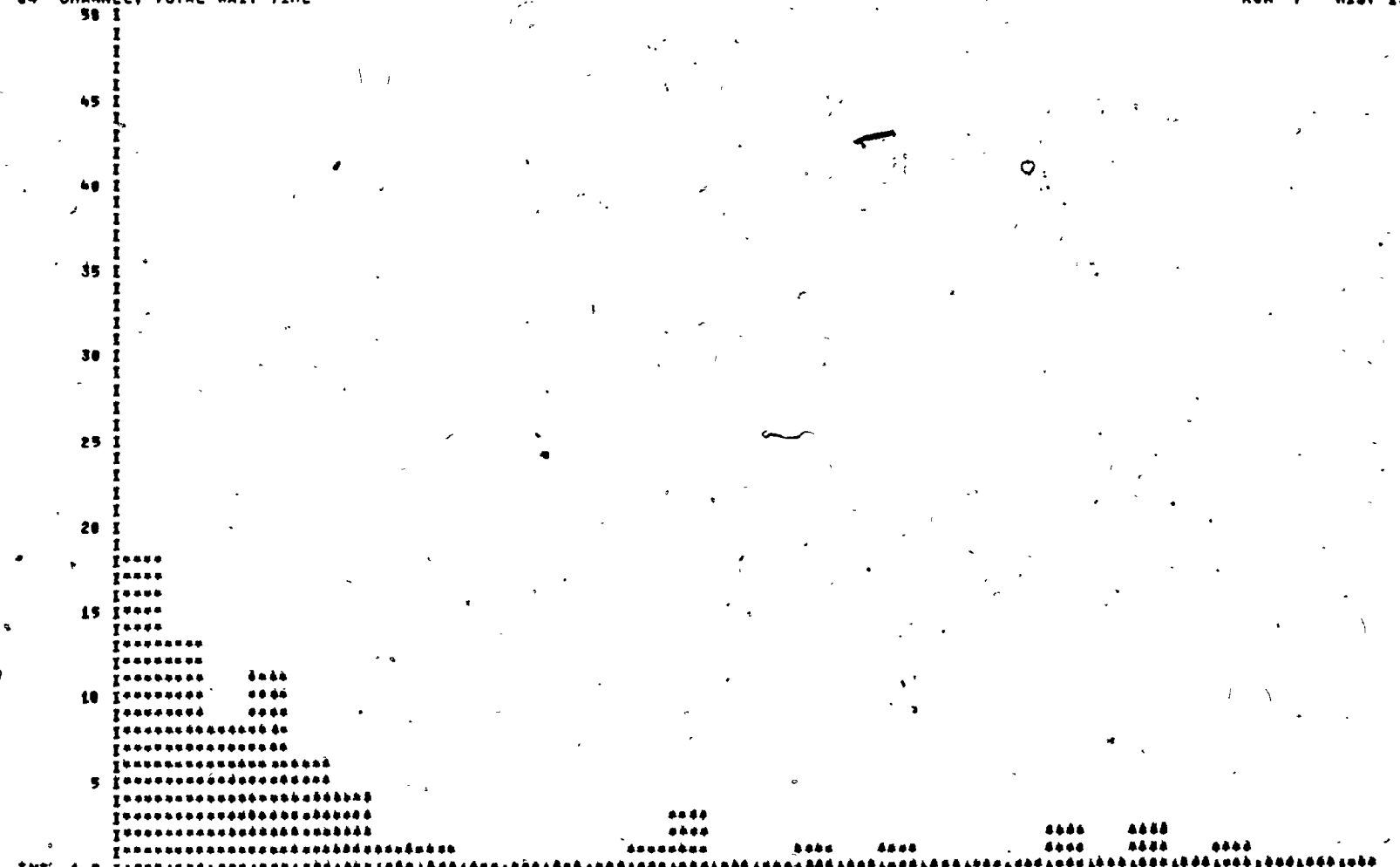


% Y-AXIS	1	6	10	17	18	16	12	6	4	3	2	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	0	2	5	11	15	16	14	8	7	5	4	3	2	1	1	1	1	1	1	0	0	0	1	1	0	2	0	1	0	0

GROUPED PERCENTAGES (X) 47.76 39.43 7.18 2.84 2.87

TOTAL Y-AXIS	3998	MAXIMUM Y-AXIS	715
TOTAL X-AXIS	108082	MAXIMUM X-AXIS	16991
AVERAGE (X-AXIS)	27.06	STANDARD DEVIATION	23.11
MAXIMUM OBSERVED	539		

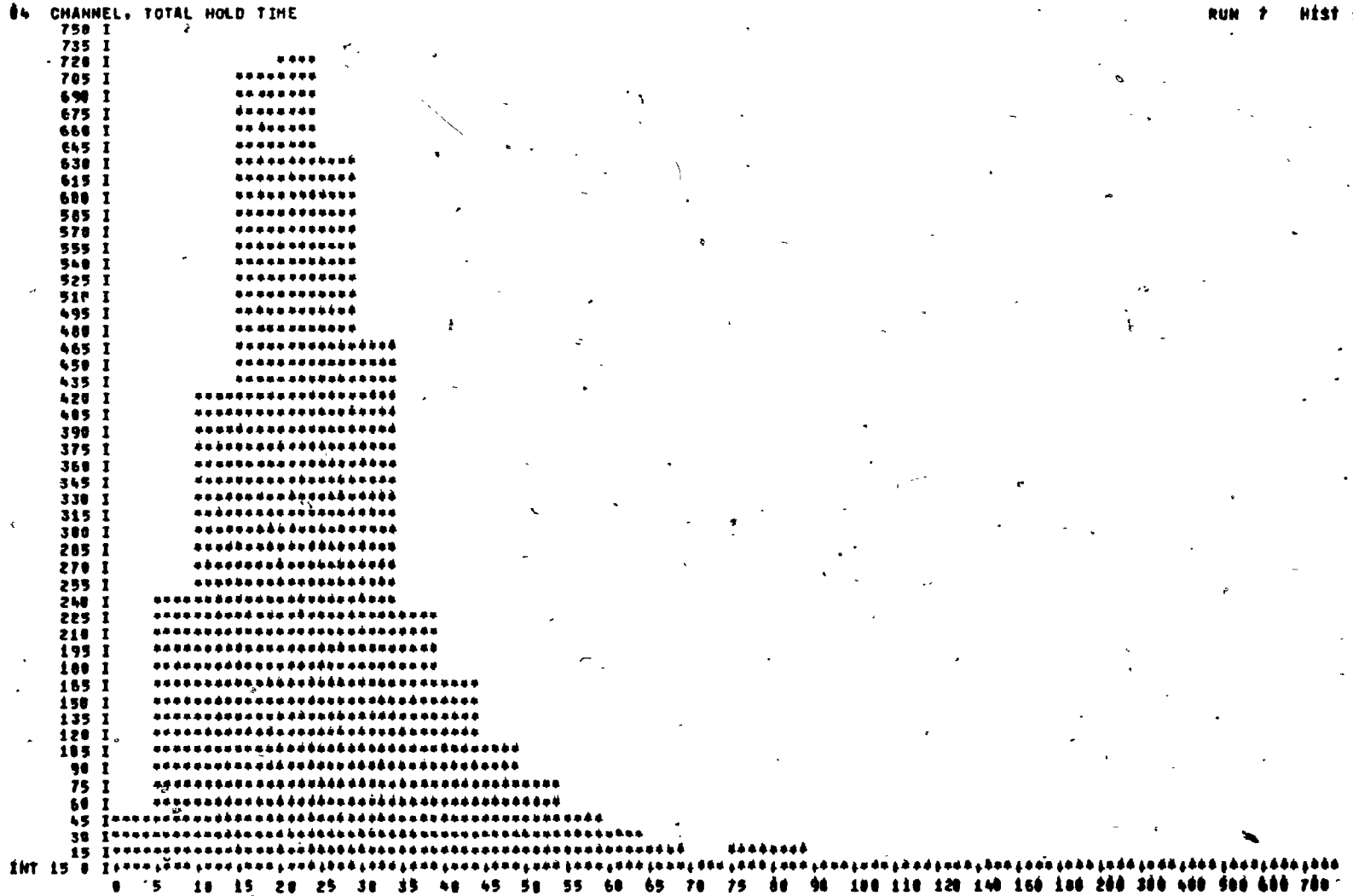
04 CHANNEL, TOTAL WAIT TIME RUN 7 HIST 137

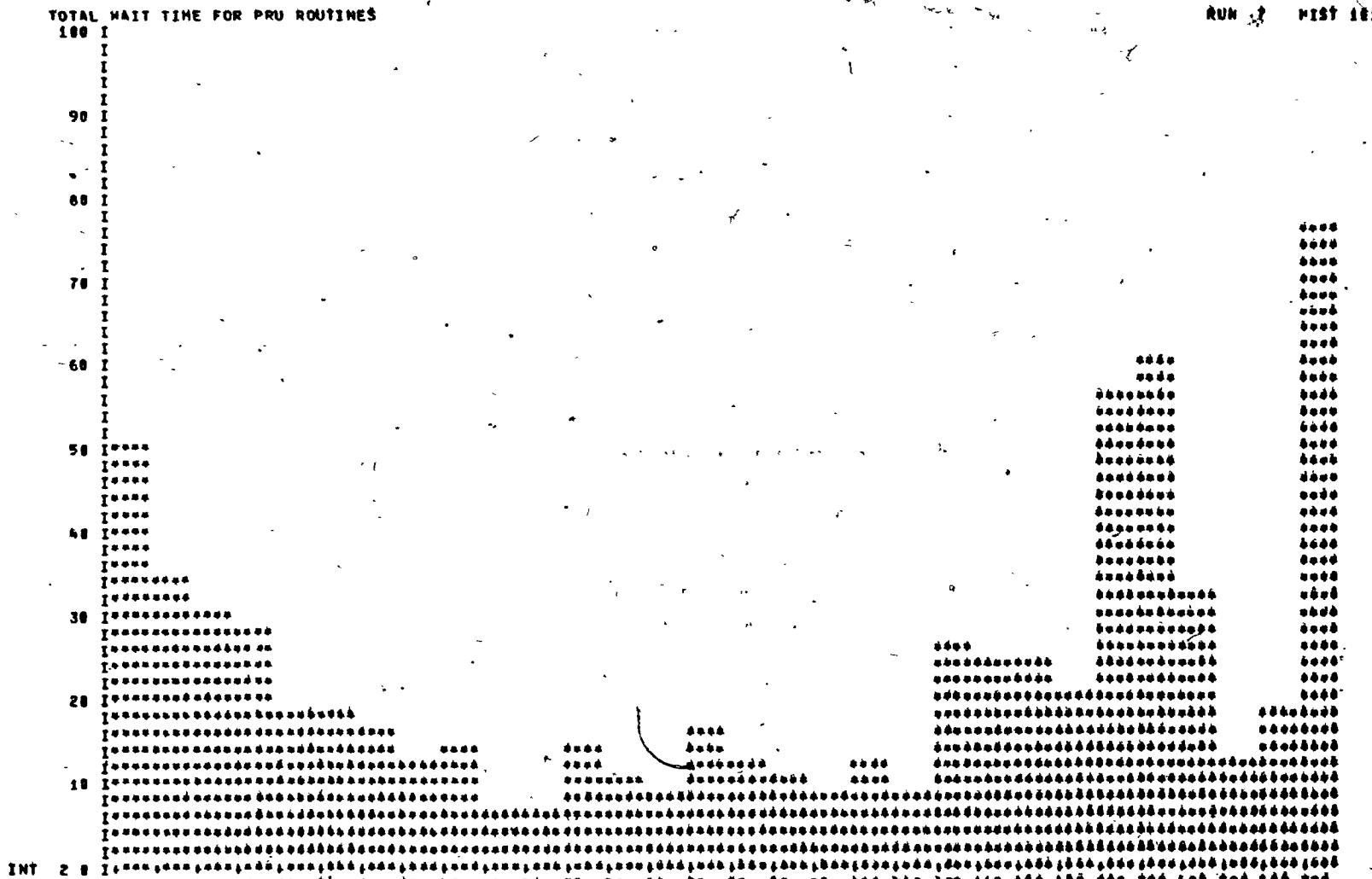


% Y-AXIS	25	10	11	15	0	5	1	1	0	0	0	0	1	4	0	0	1	0	1	0	0	0	3	0	3	0	1	0	0	0
% X-AXIS	1	4	4	8	5	4	1	1	3	0	0	0	3	8	9	0	3	0	4	0	0	0	14	0	21	0	17	0	0	0

GROUPED PERCENTAGES (X) 26.11 2.62 14.29 18.18 38.62

TOTAL Y-AXIS	73	MAXIMUM Y-AXIS	18
TOTAL X-AXIS	2478	MAXIMUM X-AXIS	524
AVERAGE (X-AXIS)	33.95	STANDARD DEVIATION	69.16
MAXIMUM OBSERVED	433		


04 CHANNEL, TOTAL HOLD TIME
RUN 7 HIST 130
INT 15 0
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 90 100 110 120 140 160 180 200 300 400 500 600 700
X Y-AXIS 1 6 10 17 18 16 12 6 4 3 2 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
X X-AXIS 0 2 4 10 14 15 13 7 6 4 3 3 2 1 1 1 1 1 1 0 0 1 1 1 1 0 2 1 1 1 0
GROUPED PERCENTAGES (X) 45.11 37.40 7.56 3.60 6.13
TOTAL Y-AXIS 4090
TOTAL X-AXIS 116153
AVERAGE (X-AXIS) 28.40
MAXIMUM OBSERVED 632
MAXIMUM Y-AXIS 722
MAXIMUM X-AXIS 17252
STANDARD DEVIATION 31.32

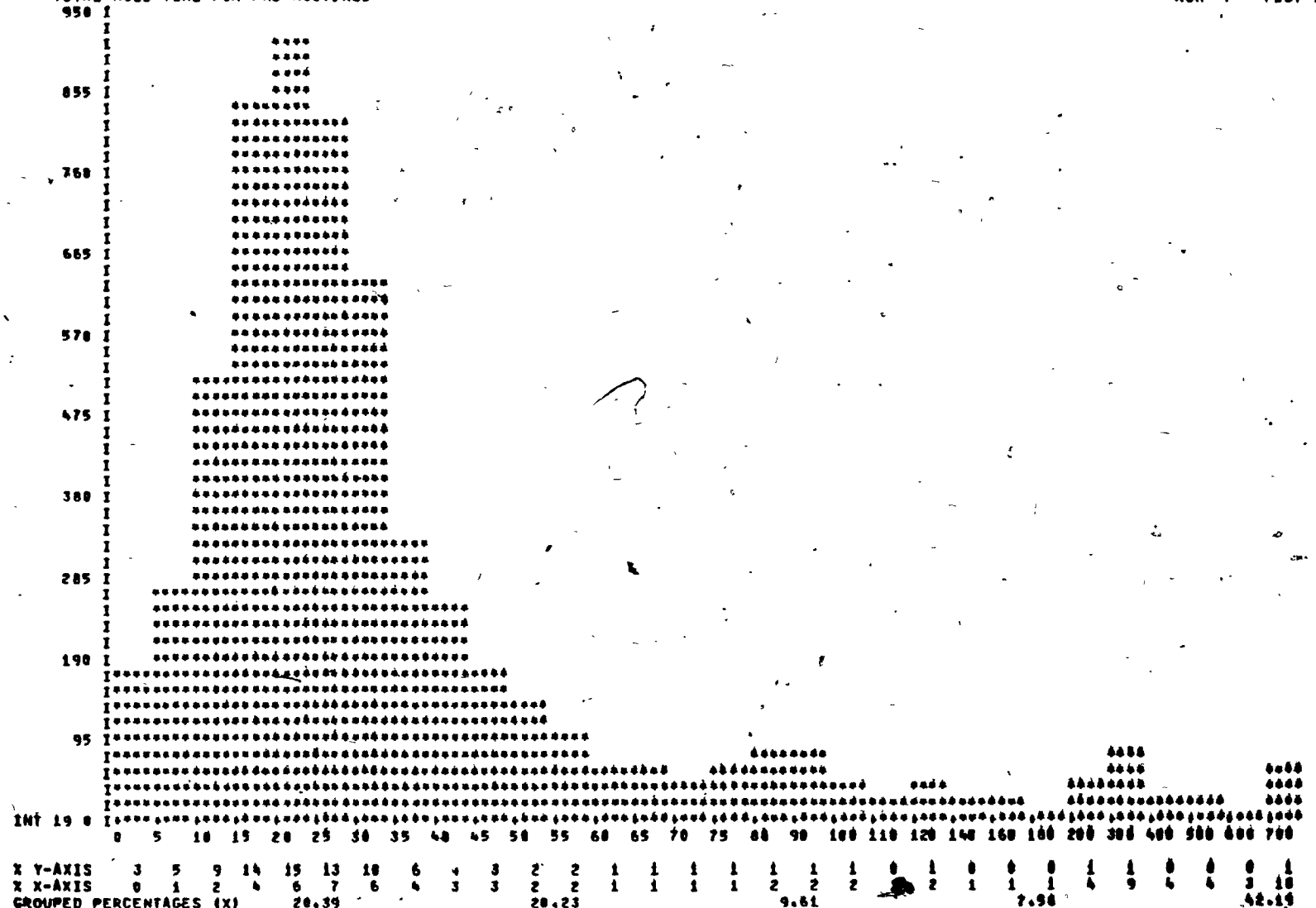


X Y-AXIS	7	5	4	4	3	3	2	2	2	1	1	2	1	1	1	2	2	2	1	2	1	4	3	4	3	8	9	5	2	3	11
X X-AXIS	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	2	2	2	2	7	10	7	3	5	54
GROUPED PERCENTAGES (%)				1.88							1.48							2.42						8.36						66.74	

TOTAL Y-AXIS	692	MAXIMUM Y-AXIS	78
TOTAL X-AXIS	208689	MAXIMUM X-AXIS	113032
AVERAGE (X-AXIS)	301.46	STANDARD DEVIATION	644.75
MAXIMUM OBSERVED	3136		

TOTAL HOLD TIME FOR PRU ROUTINES

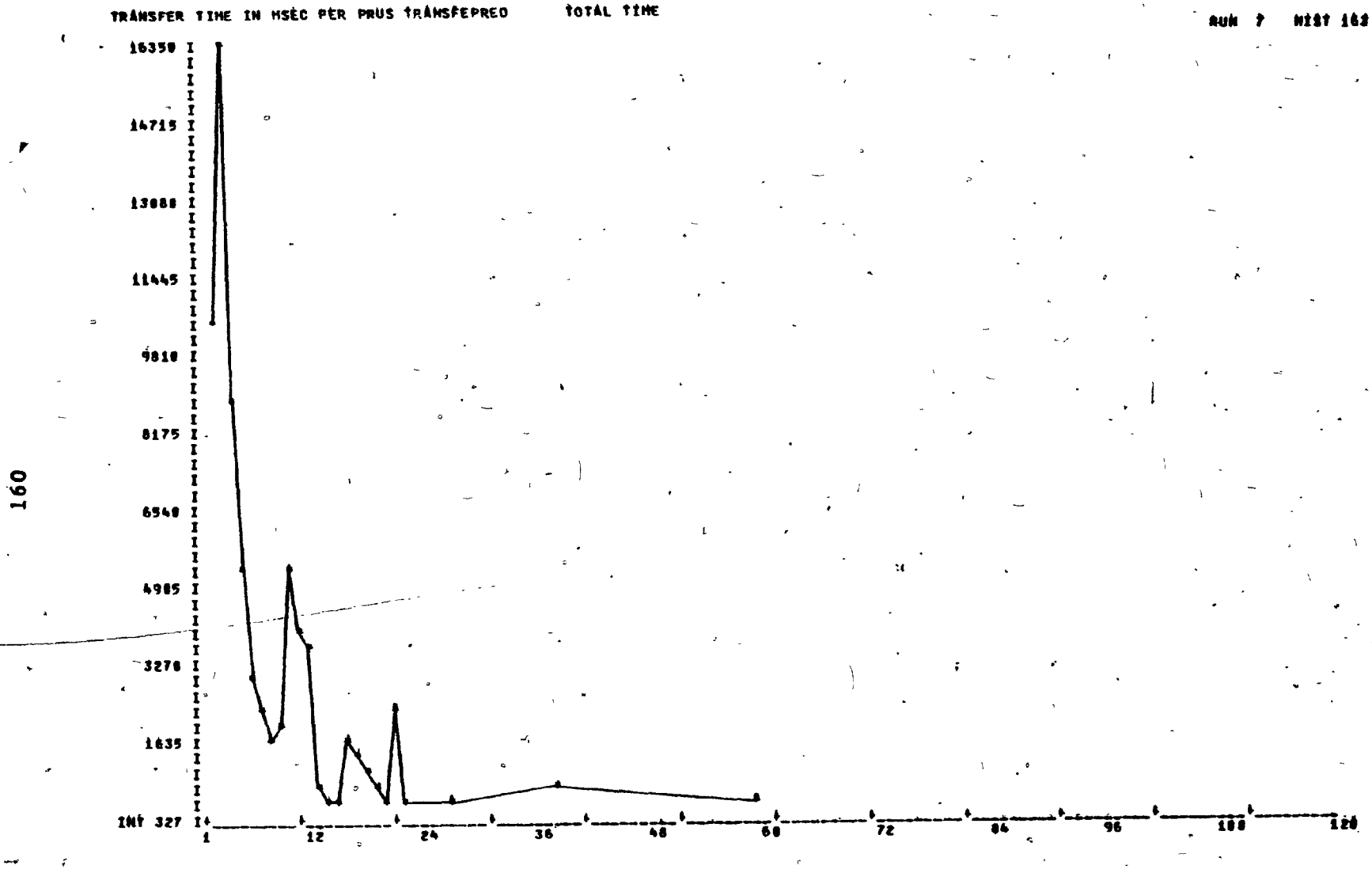
RUN 7 HIST 162

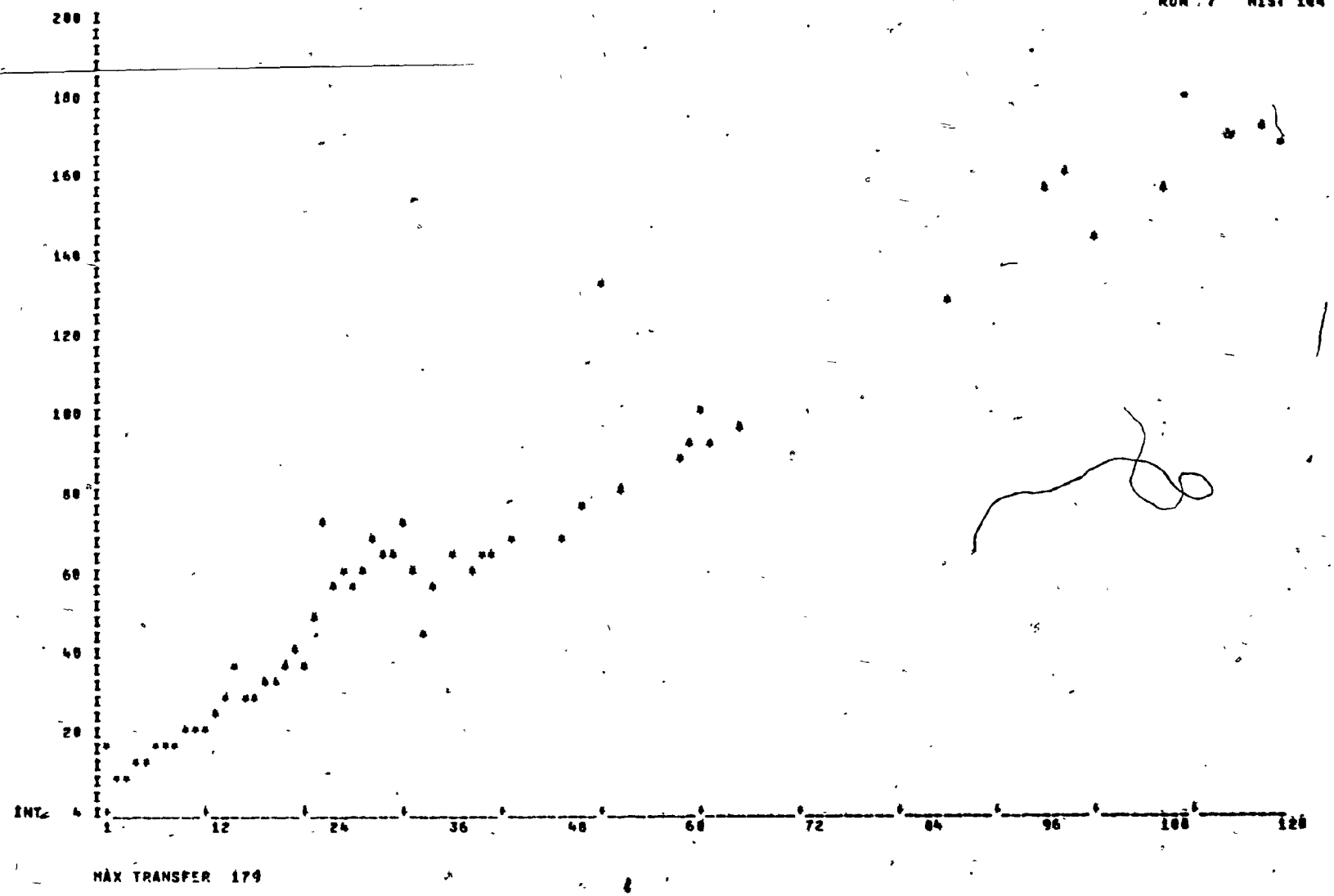


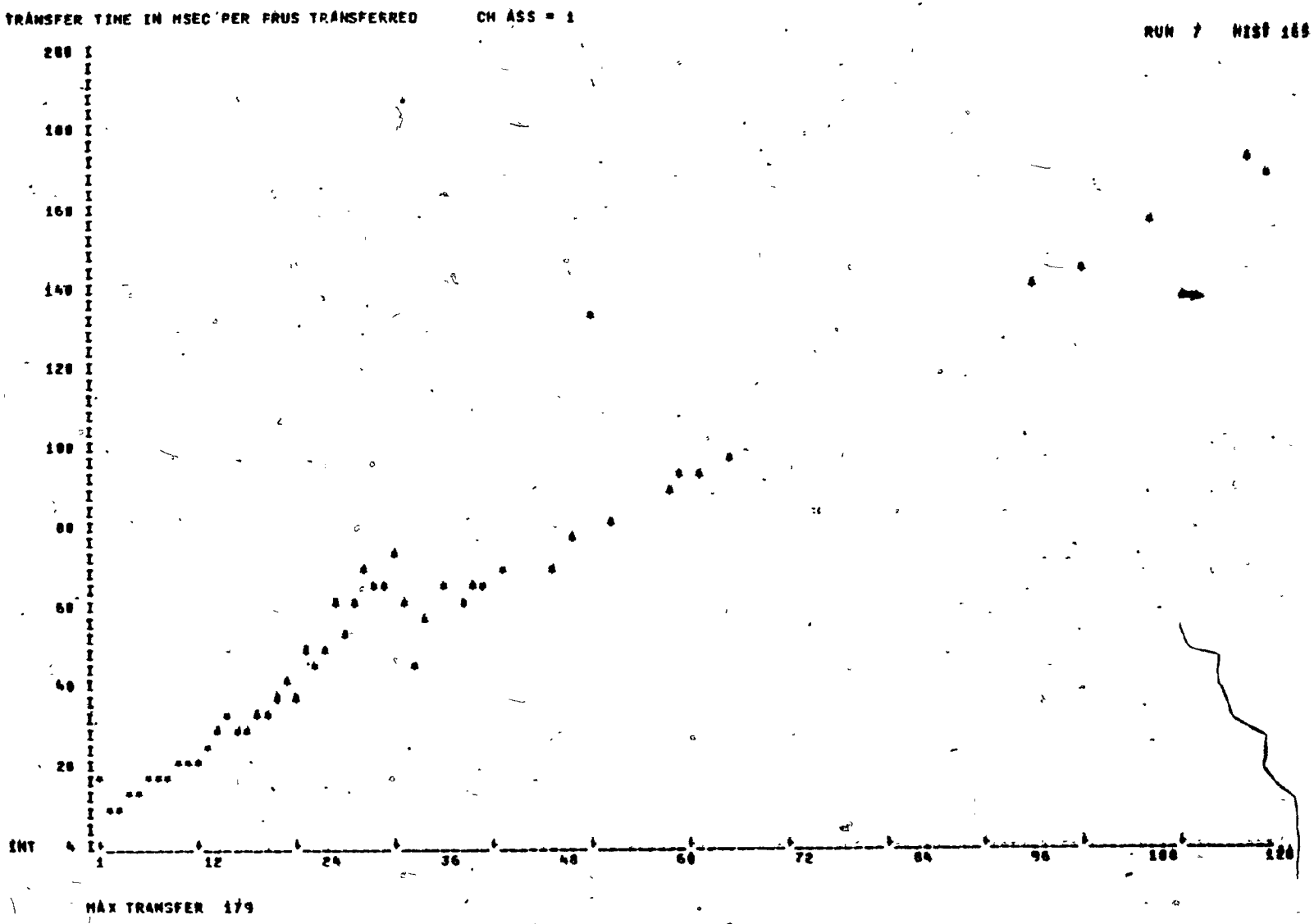
% Y-AXIS	3	5	9	14	15	13	10	6	4	3	2	2	1	1	1	1	1	1	1	0	1	0	0	0	1	1	0	0	0	1
% X-AXIS	0	1	2	4	6	7	6	4	3	3	2	2	1	1	1	1	2	2	2	2	2	1	1	1	4	9	4	4	3	10

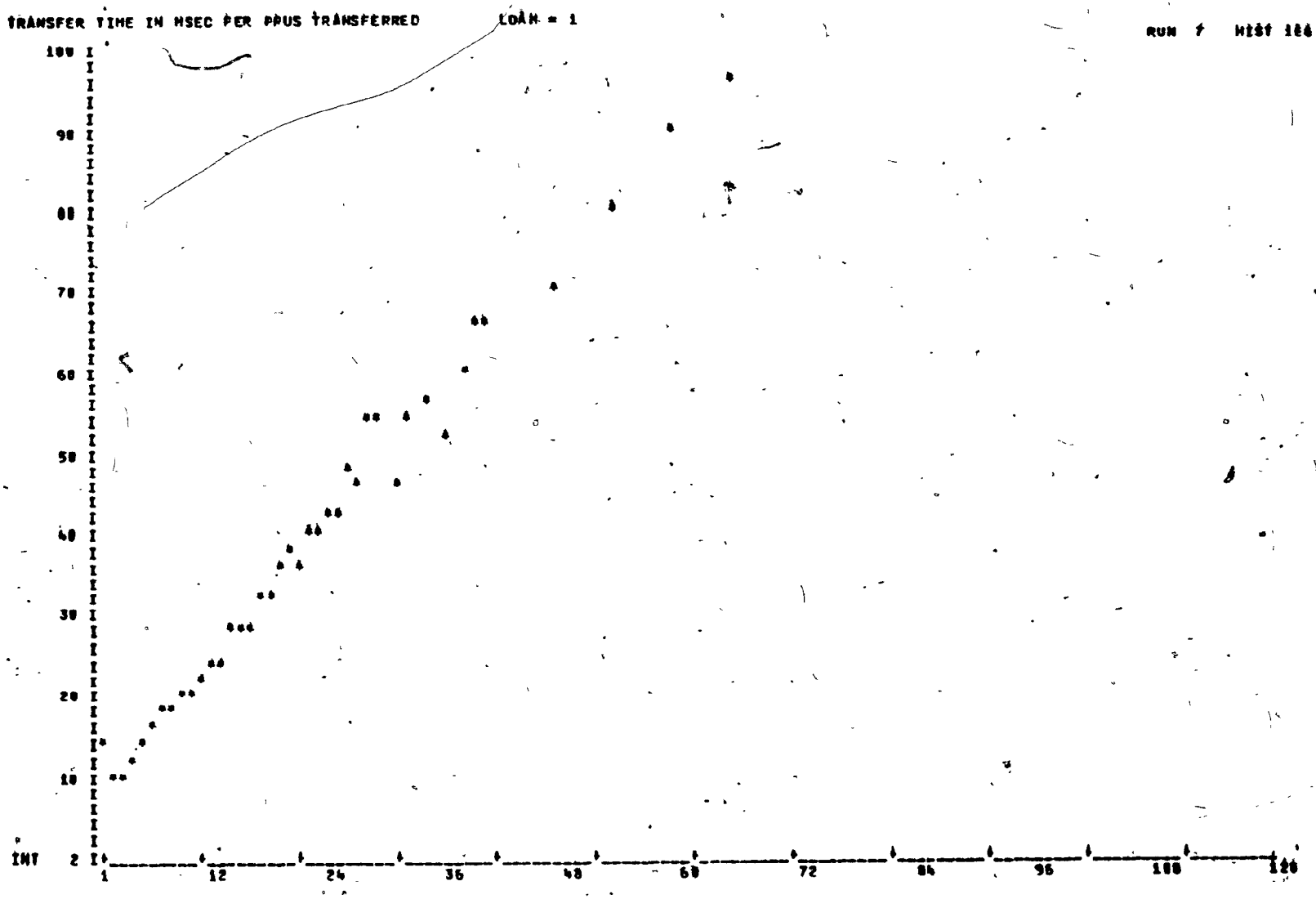
GROUPED PERCENTAGES (X) 20.39 20.23 9.61 7.98 42.15

TOTAL Y-AXIS	6892	MAXIMUM Y-AXIS	923
TOTAL X-AXIS	323110	MAXIMUM X-AXIS	99300
AVERAGE (X-AXIS)	53.04	STANDARD DEVIATION	117.13
MAXIMUM OBSERVED	1623		

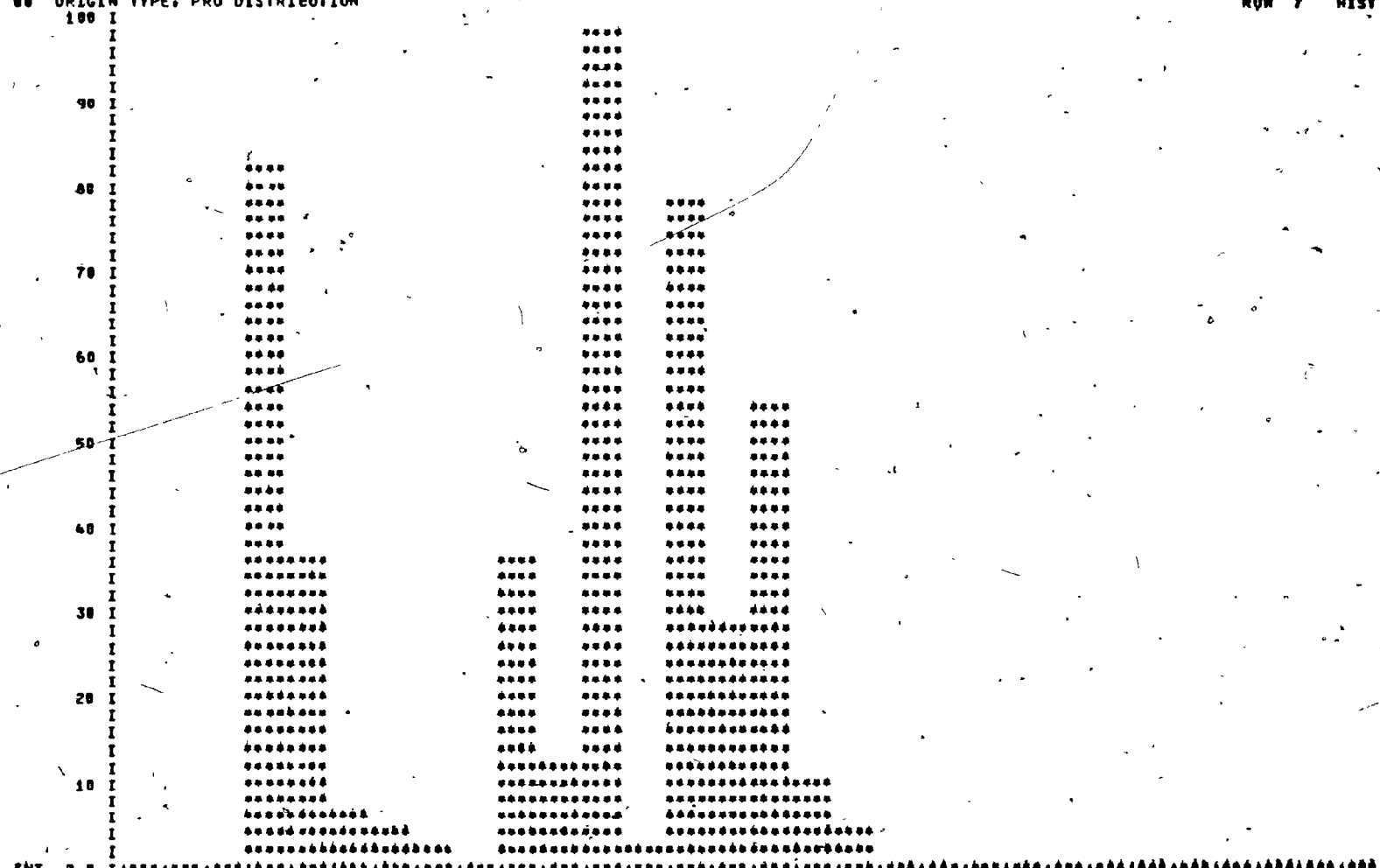

TRANSFER TIME IN MSEC PER PRUS TRANSFEPRED
TOTAL TIME
RUN 7 MIST 163
16358
14715
13088
11445
9810
8175
6540
4905
3270
1635
INT 327
1
12
24
36
48
60
72
84
96
108
120
160


TRANSFER TIME IN MSEC PER PRUS TRANSFERRED
CH ASS >=1
RUN . 7 NIST 164
200
180
160
140
120
100
80
60
40
20
INT= 4
1
12
24
36
48
60
72
84
96
108
120
MAX TRANSFER 179


TRANSFER TIME IN MSEC PER PRUS TRANSFERRED
CH ASS = 1
RUN 7 HIST 165
200
180
160
140
120
100
80
60
40
20
INT 4
1
12
24
36
48
60
72
84
96
108
120
MAX TRANSFER 179


TRANSFER TIME IN MSEC PER PPUS TRANSFERRED
LOAN = 1
RUN 7 HIST 188
100
90
80
70
60
50
40
30
20
10
INT 2
1
12
24
36
48
60
72
84
96
108
120
MAX TRANSFER 103

00 ORIGIN TYPE, PRU DISTRIBUTION RUN 7 HIST 171

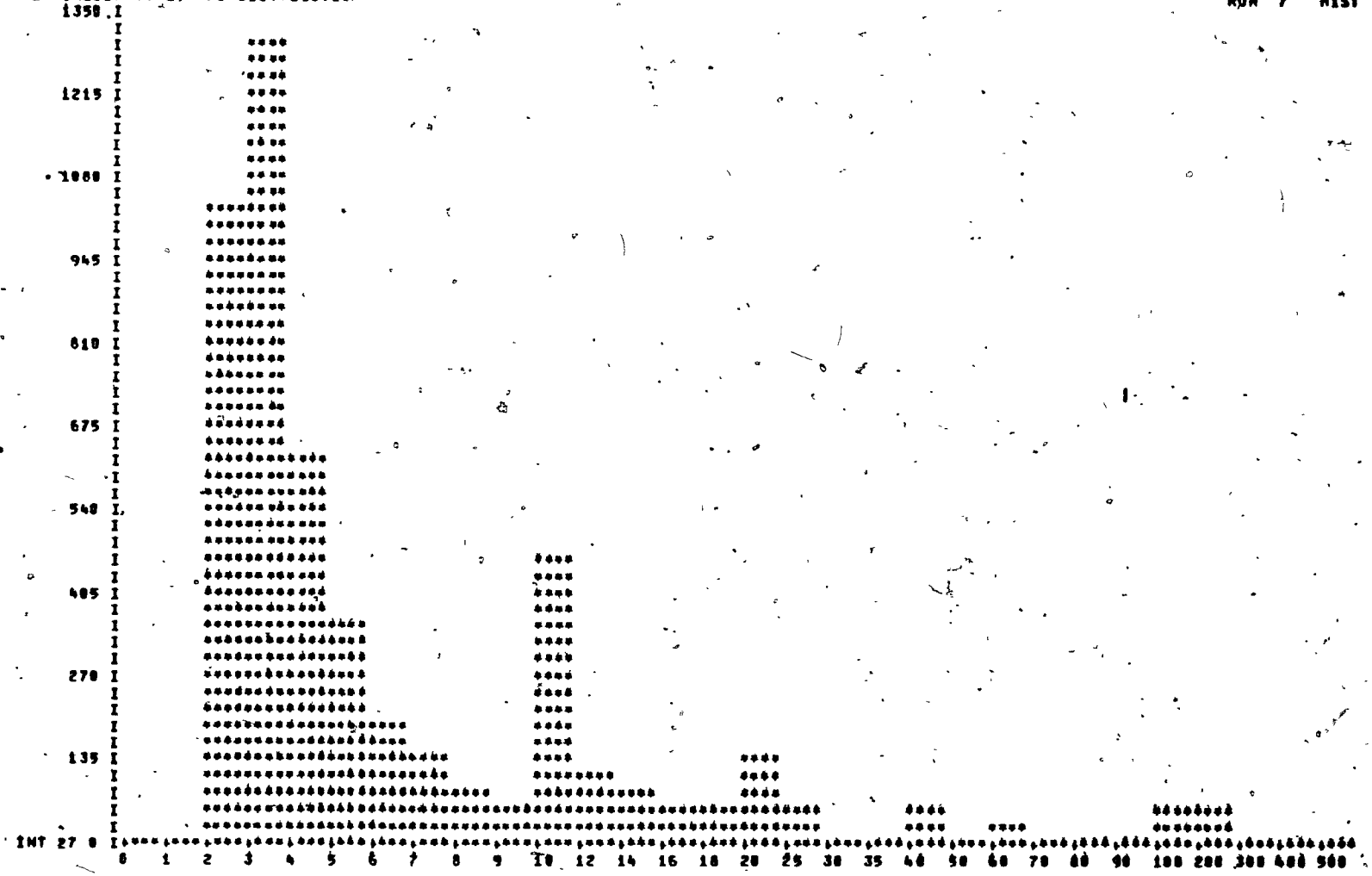


% Y-AXIS	0	0	0	10	6	1	1	1	0	6	3	21	1	17	6	12	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	0	0	0	4	3	1	1	0	0	6	2	21	1	22	9	22	5	3	0	0	0	0	0	0	0	0	0	0	0	0	0

GROUPED PERCENTAGES (X) 7.44 30.89 62.48 0.00 0.00

TOTAL Y-AXIS	465	MAXIMUM Y-AXIS	99
TOTAL X-AXIS	5727	MAXIMUM X-AXIS	1203
AVERAGE (X-AXIS)	12.32	STANDARD DEVIATION	7.14
MAXIMUM OBSERVED	34		

01 ORIGIN TYPE, PPU DISTRIBUTION RUN 7 HIST 172

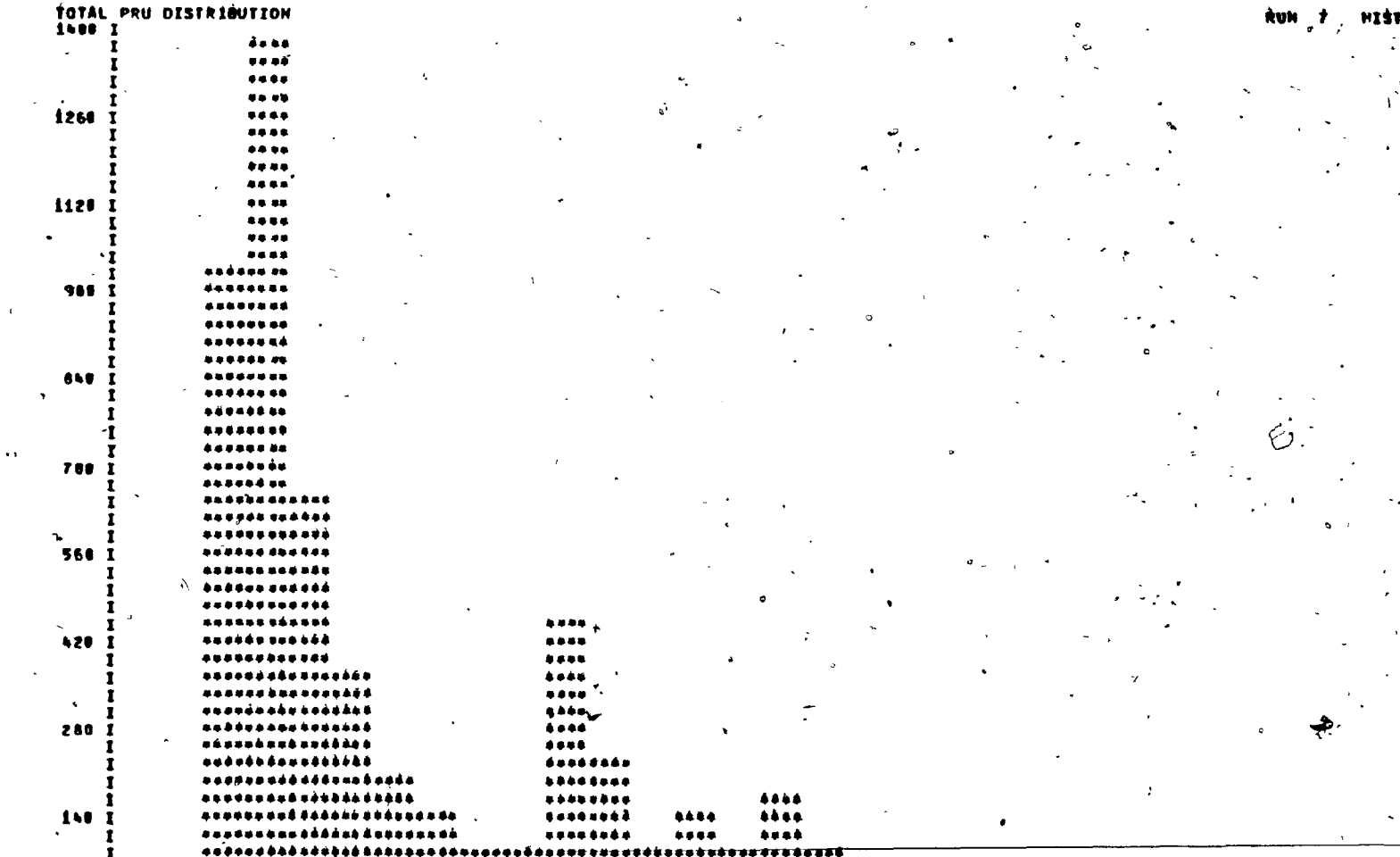


% Y-AXIS	8	8	20	25	12	7	4	3	2	1	9	2	2	1	1	3	2	0	0	1	0	1	0	0	0	1	1	0	0	0
% X-AXIS	0	0	3	6	4	3	2	2	1	1	8	2	2	2	2	5	3	1	1	6	1	3	0	0	1	17	24	0	1	0

GROUPED PERCENTAGES (X) 16.27 15.79 13.88 10.82 43.24

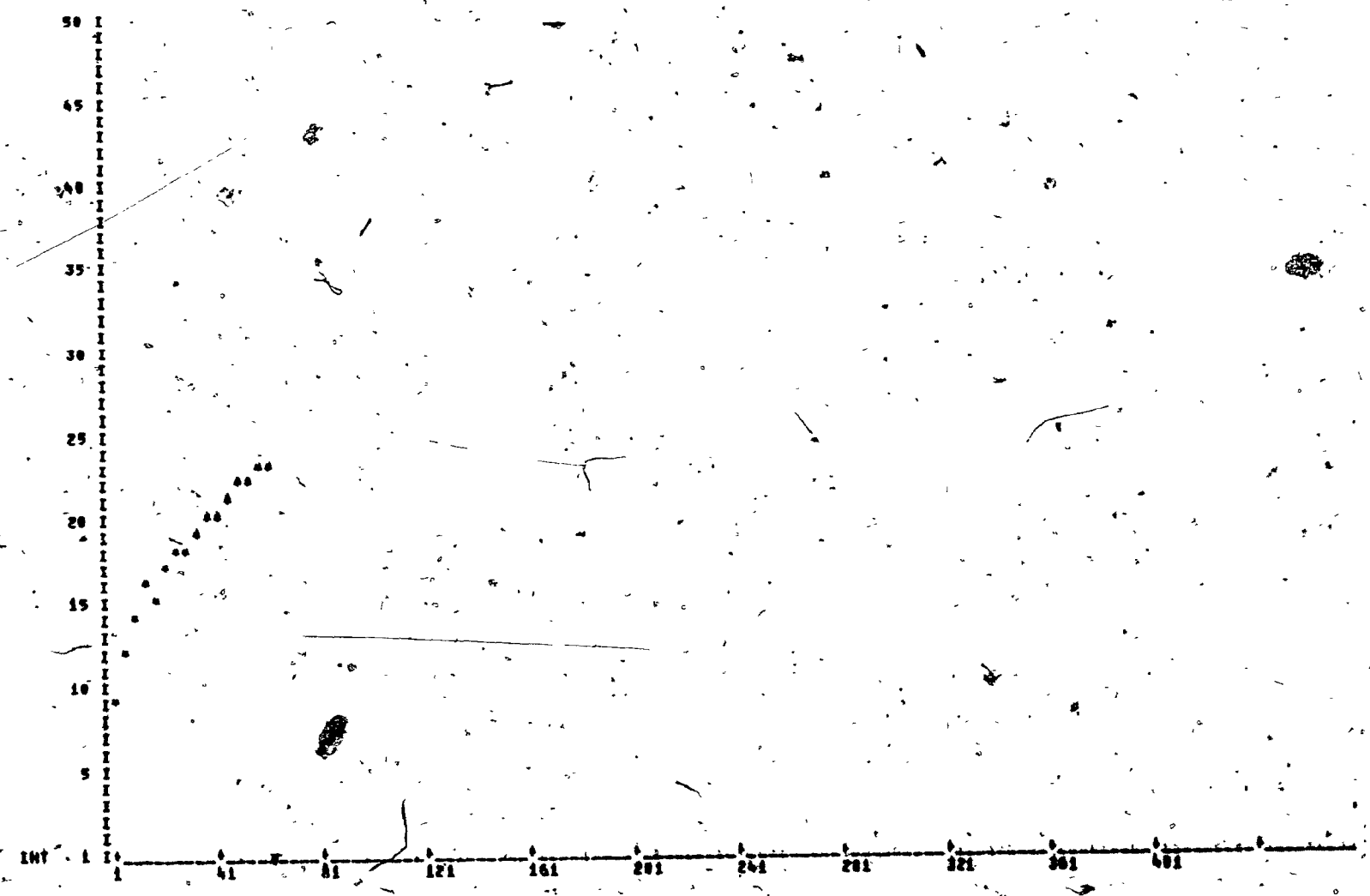
TOTAL Y-AXIS	5162	MAXIMUM Y-AXIS	1312
TOTAL X-AXIS	63575	MAXIMUM X-AXIS	15376
AVERAGE (X-AXIS)	12.32	STANDARD DEVIATION	32.79
MAXIMUM OBSERVED	436		

RUN 7, HIST 173



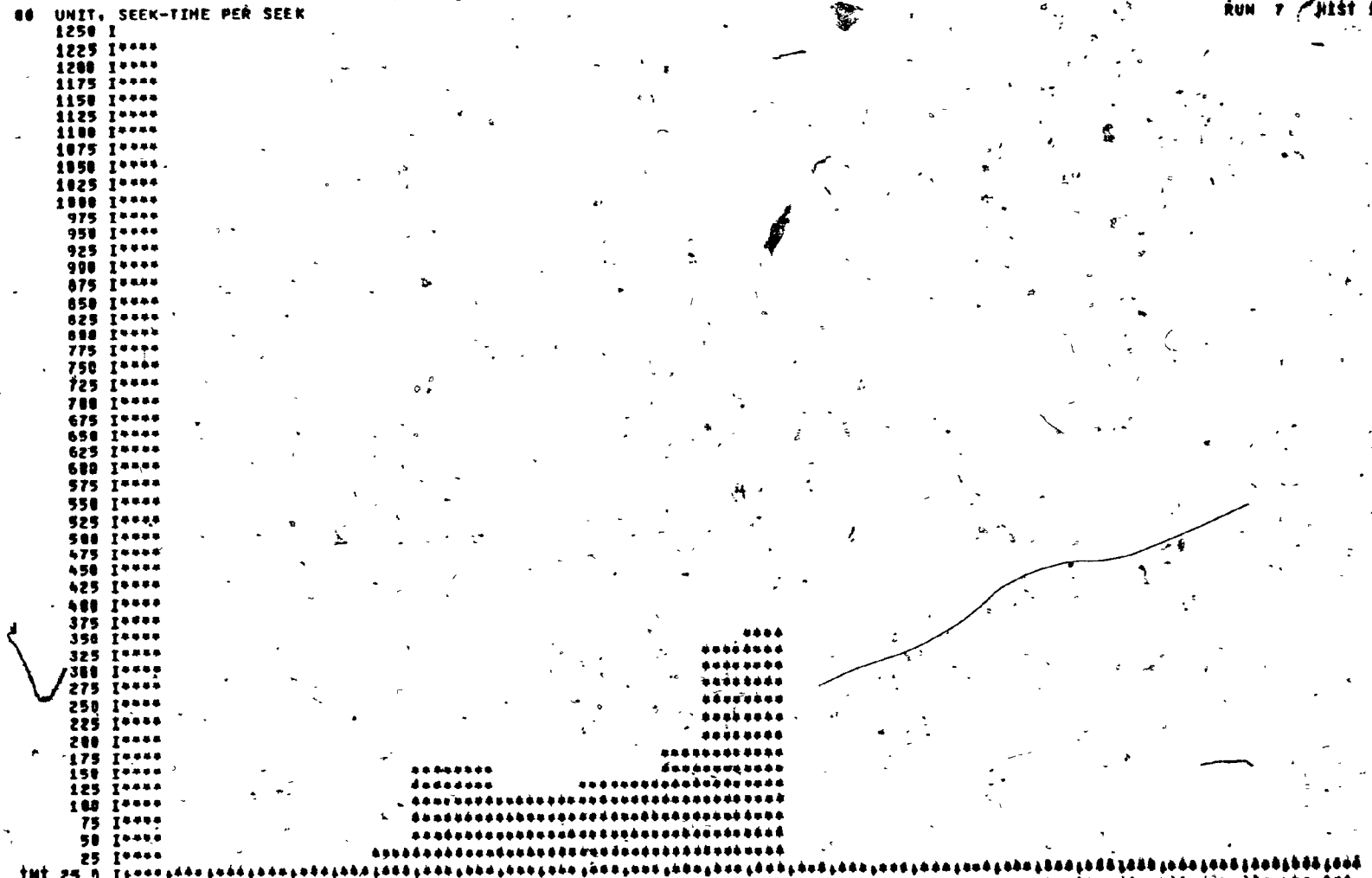
% Y-AXIS	0	0	10	25	12	7	4	3	2	2	0	4	2	3	2	3	2	0	0	1	0	1	0	0	0	1	1	0	0	0
X X-AXIS	0	0	3	6	4	3	2	1	1	1	7	4	2	3	2	6	3	1	1	5	1	3	0	0	1	10	22	0	1	0
GROUPED PERCENTAGES (X)					16.54						16.97						17.90						9.93						39.67	

TOTAL Y-AXIS	5627	MAXIMUM Y-AXIS	1394
TOTAL X-AXIS	69382	MAXIMUM X-AXIS	15376
AVERAGE (X-AXIS)	12.32	STANDARD DEVIATION	31.47
MAXIMUM OBSERVED	436		


88. SEEK TIME IN MSEC PER SEEK LENGTH-CYLINDERS
RUN 7 HIST 174
50
45
40
35
30
25
20
15
10
5
INT 1
1
41
81
121
161
201
241
281
321
361
401

00 UNIT, SEEK-TIME PER SEEK

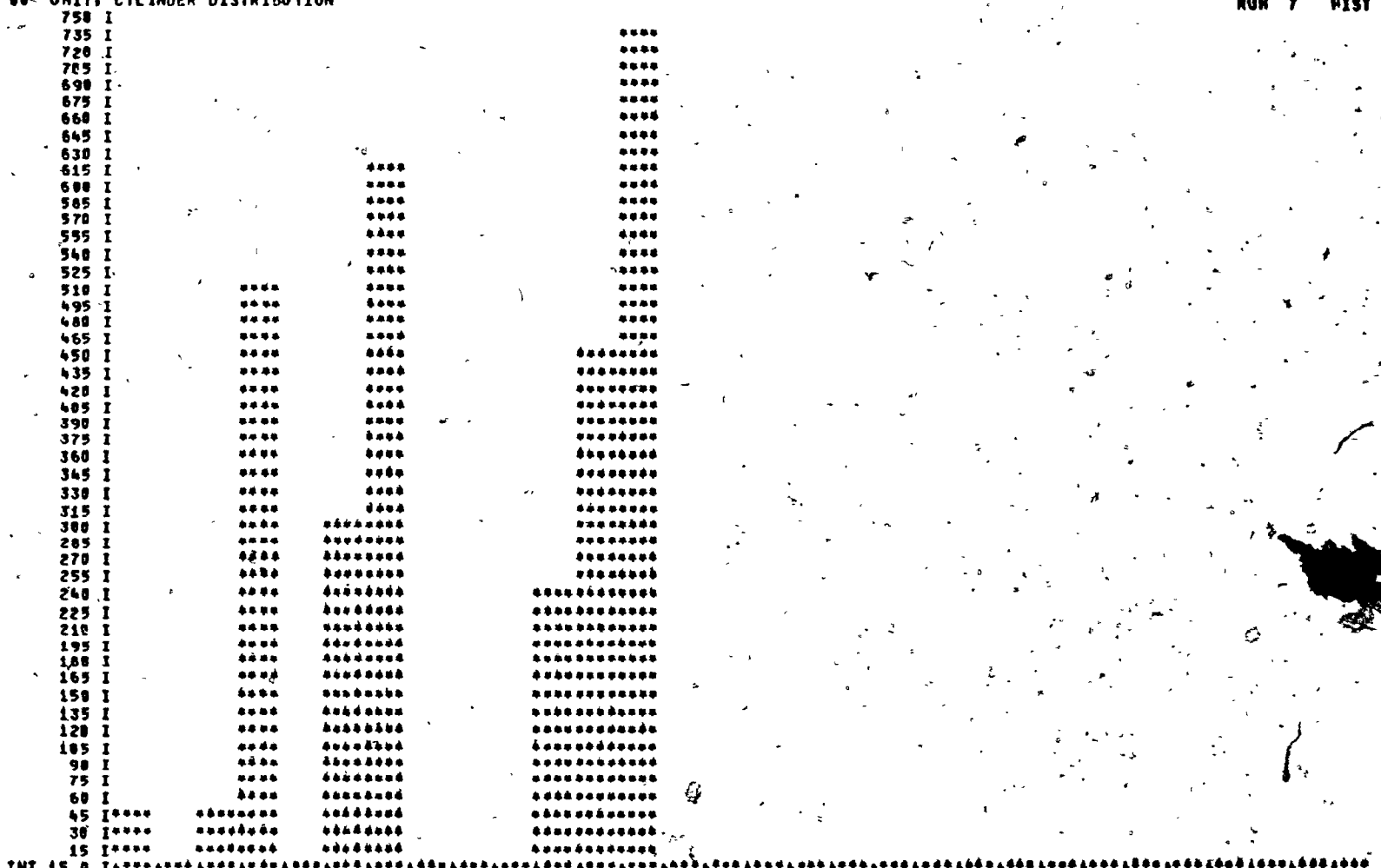
RUN 7 HIST 179



% Y-AXIS	41	0	0	0	0	0	1	6	6	4	3	4	5	7	11	12	1	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	0	0	0	0	0	0	1	5	5	4	4	6	8	12	24	28	2	0	0	0	0	0	1	0	0	0	0	0	0	0
GROUPED PERCENTAGES (X)			.09								24.77						74.99					.55							0.00	

TOTAL Y-AXIS	3824	MAXIMUM Y-AXIS	1229
TOTAL X-AXIS	26471	MAXIMUM X-AXIS	7446
AVERAGE (X-AXIS)	6.75	STANDARD DEVIATION	8.45
MAXIMUM OBSERVED	73		

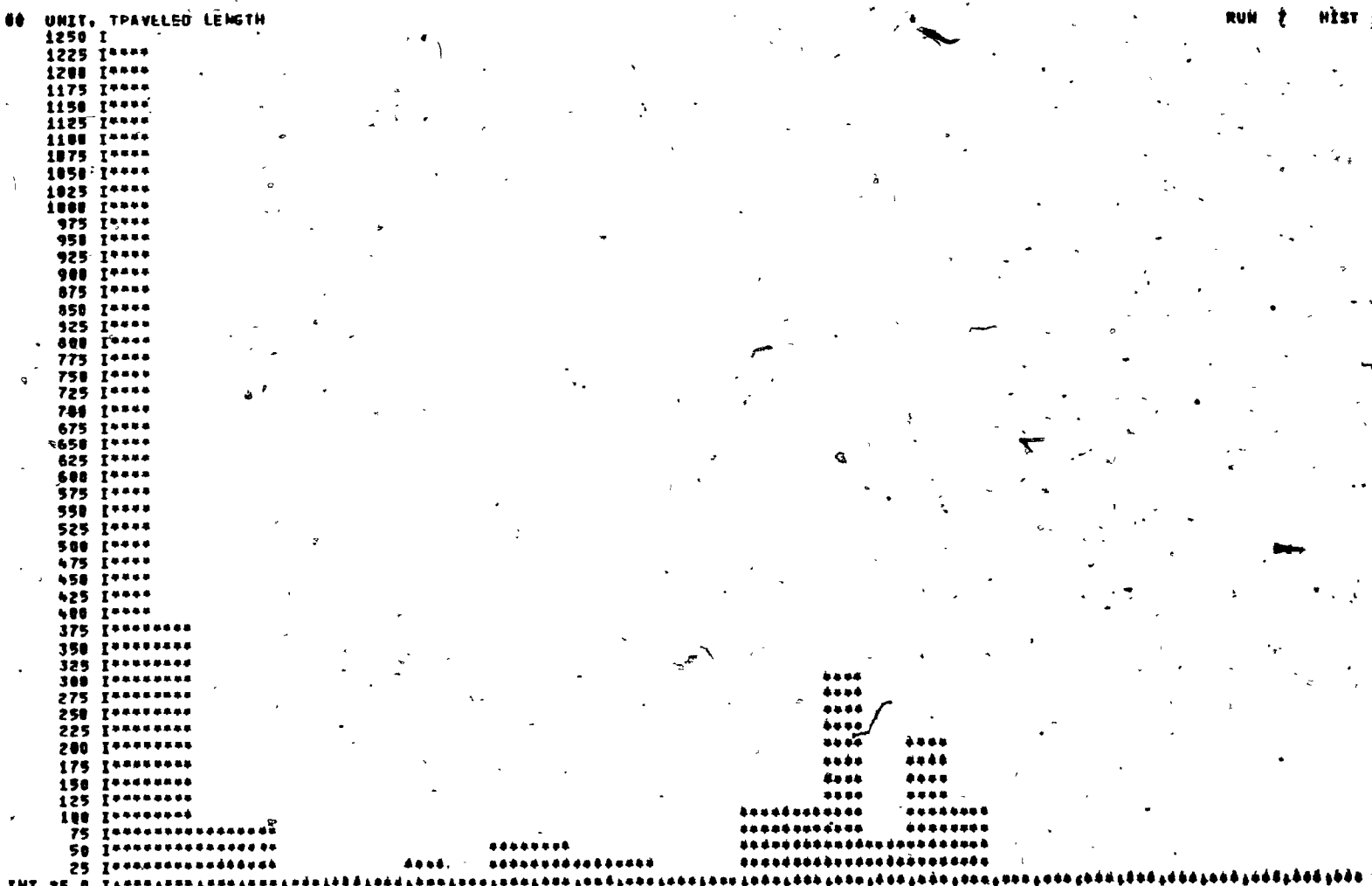
00- UNIT, CYLINDER DISTRIBUTION

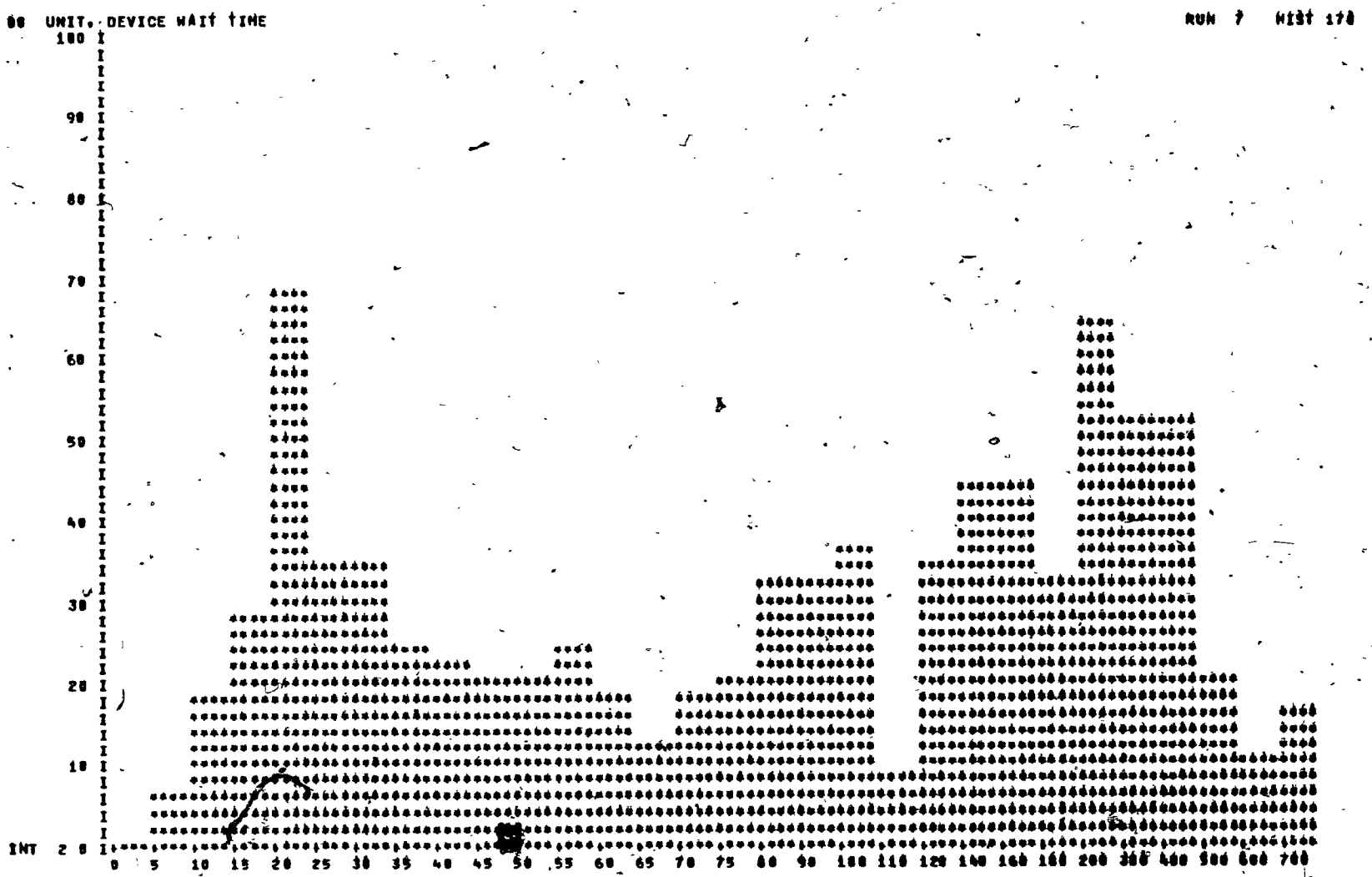


X Y-AXIS	2	0	2	17	0	10	21	0	0	0	0	15	24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X X-AXIS	0	0	0	7	0	7	16	0	0	0	10	22	30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

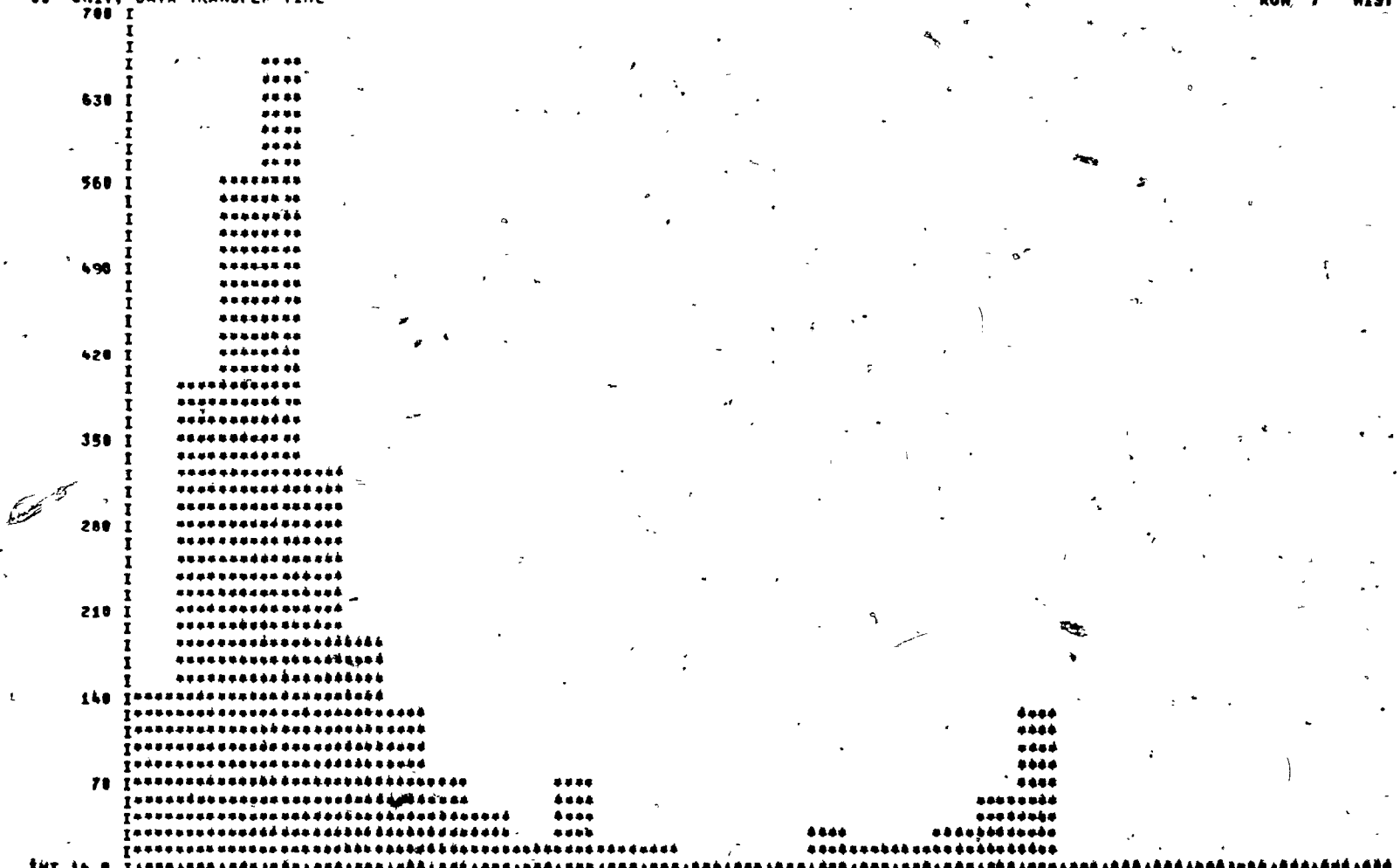
GROUPED PERCENTAGES (X) 14.37 48.03 37.60 0.00 0.00

TOTAL Y-AXIS	3824.	MAXIMUM Y-AXIS	74
TOTAL X-AXIS	120808	MAXIMUM X-AXIS	45451
AVERAGE (X-AXIS)	39.98	STANDARD DEVIATION	18.83
MAXIMUM OBSERVED	356 ON CYLINDER	30	


00 UNIT, TRAVELED LENGTH
RUN 1 HIST 177
1250 I
1225 I****
1200 I****
1175 I****
1150 I****
1125 I****
1100 I****
1075 I****
1050 I****
1025 I****
1000 I****
975 I****
950 I****
925 I****
900 I****
875 I****
850 I****
825 I****
800 I****
775 I****
750 I****
725 I****
700 I****
675 I****
650 I****
625 I****
600 I****
575 I****
550 I****
525 I****
500 I****
475 I****
450 I****
425 I****
400 I****
375 I*********
350 I*********
325 I*********
300 I*********
275 I*********
250 I*********
225 I*********
200 I*********
175 I*********
150 I*********
125 I*********
100 I*********
75 I*****************
50 I*****************
25 I*****************
INT 25 0 I
0 1 2 3 4 5 6 7 8 9 10 12 14 16 18 20 25 30 35 40 50 60 70 80 90 100 200 300 400 500
% Y-AXIS 41 13 3 3 1 0 1 1 0 2 2 1 1 0 1 4 4 11 2 7 3 0 0 0 0 0 0 0 0 0
% X-AXIS 0 1 0 1 0 0 0 1 0 1 2 1 1 1 1 6 9 27 5 25 15 2 0 0 0 0 0 0 0 0
GROUPED PERCENTAGES (X) 2.67 5.23 45.10 46.93 0.100
TOTAL Y-AXIS 3824
TOTAL X-AXIS 37147
AVERAGE (X-AXIS) 12.28
MAXIMUM OBSERVED 63
MAXIMUM Y-AXIS 1220
MAXIMUM X-AXIS 10079
STANDARD DEVIATION 16.94


80 UNIT, DEVICE WAIT TIME
RUN 7 HIST 178
100
90
80
70
60
50
40
30
20
10
INT 2 8
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 90 100 110 120 140 160 180 200 300 400 500 600 700
% Y-AXIS 0 1 2 3 8 4 4 3 3 2 2 3 2 2 2 2 4 4 4 1 4 5 5 4 7 6 6 2 1 2
% X-AXIS 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 2 2 3 1 3 5 8 4 11 13 17 8 5 10
GROUPED PERCENTAGES (X) 2.30 4.63 7.84 21.58 53.72
TOTAL Y-AXIS 896
TOTAL X-AXIS 138826
AVERAGE (X-AXIS) 161.25
MAXIMUM OBSERVED 989
MAXIMUM Y-AXIS 68
MAXIMUM X-AXIS 23599
STANDARD DEVIATION 169.17

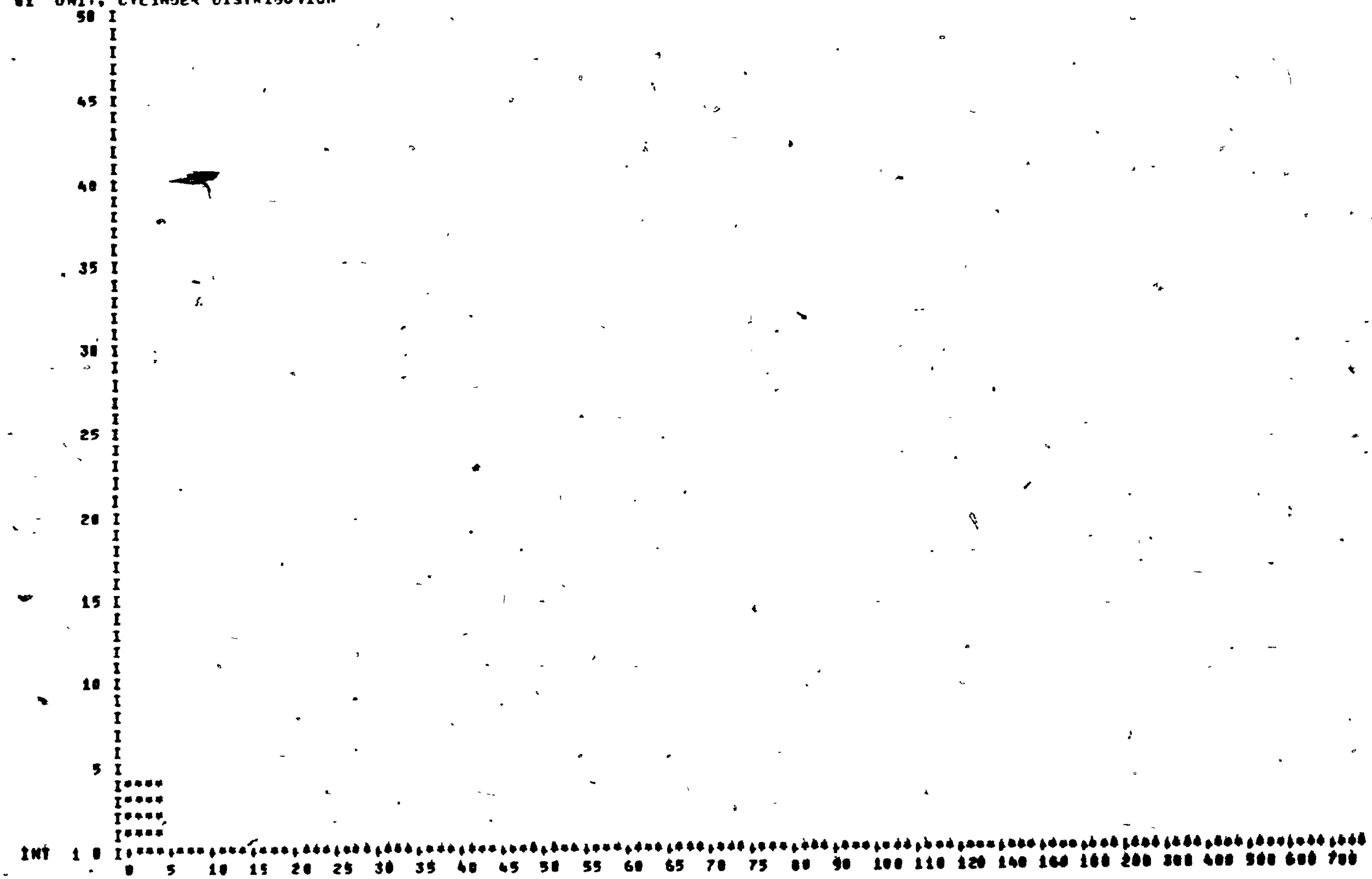
00 UNIT, DATA TRANSFER TIME RUN 7 HIST 174



% Y-AXIS	5	13	19	22	11	6	3	3	2	0	2	1	0	0	0	0	1	1	1	1	2	3	0	0	0	0	0	0	0	0
% X-AXIS	1	3	7	12	8	5	5	3	2	1	4	1	1	1	0	1	3	2	2	5	10	23	1	0	0	0	0	0	0	0
GROUPED PERCENTAGES (X)					39.98						15.94							7.14					48.93						0.00	

TOTAL Y-AXIS	3824	MAXIMUM Y-AXIS	661
TOTAL X-AXIS	93899	MAXIMUM X-AXIS	21457
AVERAGE (X-AXIS)	38.79	STANDARD DEVIATION	37.77
MAXIMUM OBSERVED	165		

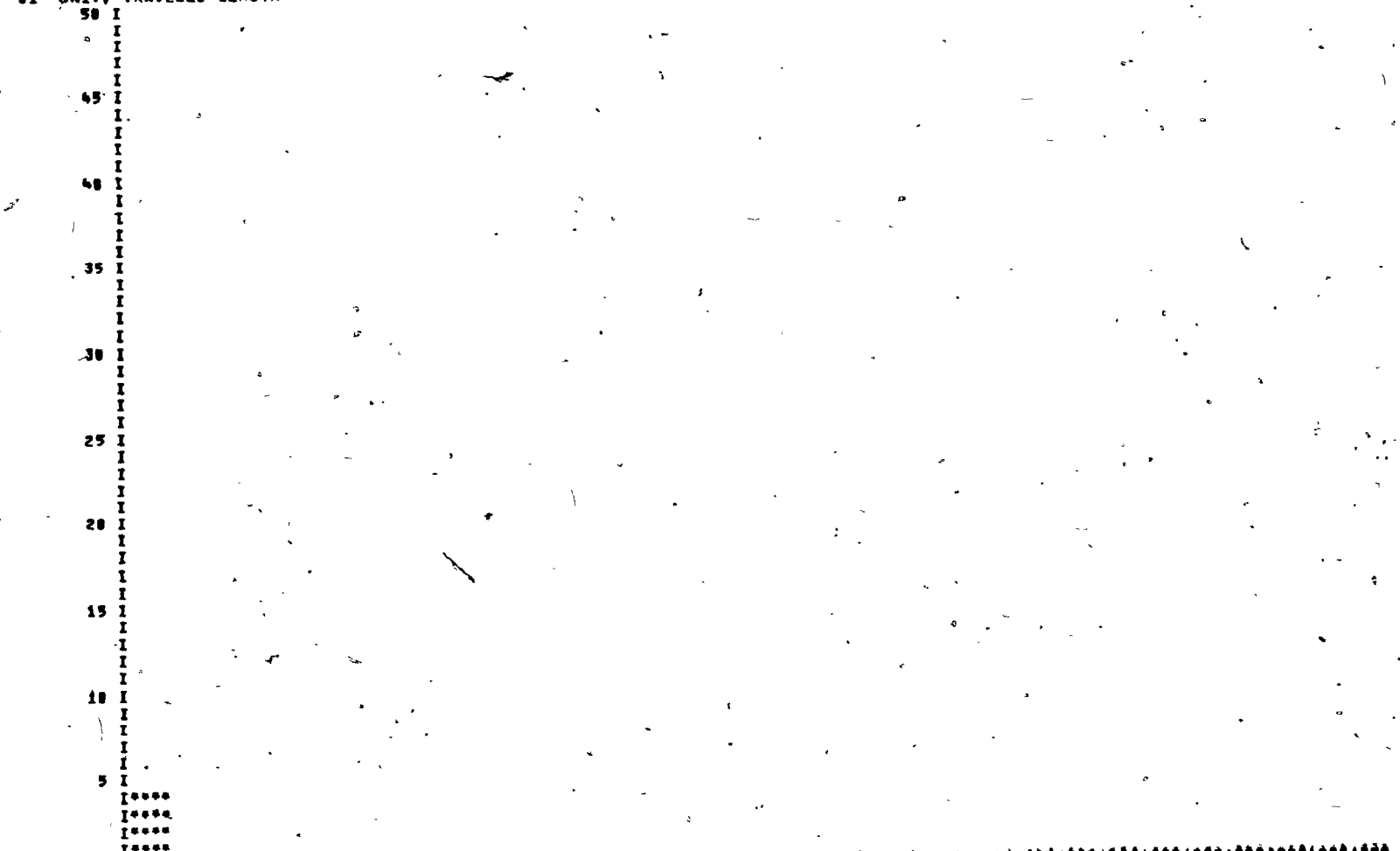
61 UNIT, CYLINDER DISTRIBUTION RUN 7 HIST 100



```
% Y-AXIS  100  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
% X-AXIS    0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
GROUPED PERCENTAGES (X)  0.00   0.00   0.00   0.00   0.00
        TOTAL Y-AXIS             4        MAXIMUM Y-AXIS         4
        TOTAL X-AXIS             0        MAXIMUM X-AXIS         0
        AVERAGE (X-AXIS)         0.00     STANDARD DEVIATION     0.00
        MAXIMUM OBSERVED         4 ON CYLINDER   0
```

01 UNIT, TRAVELED LENGTH

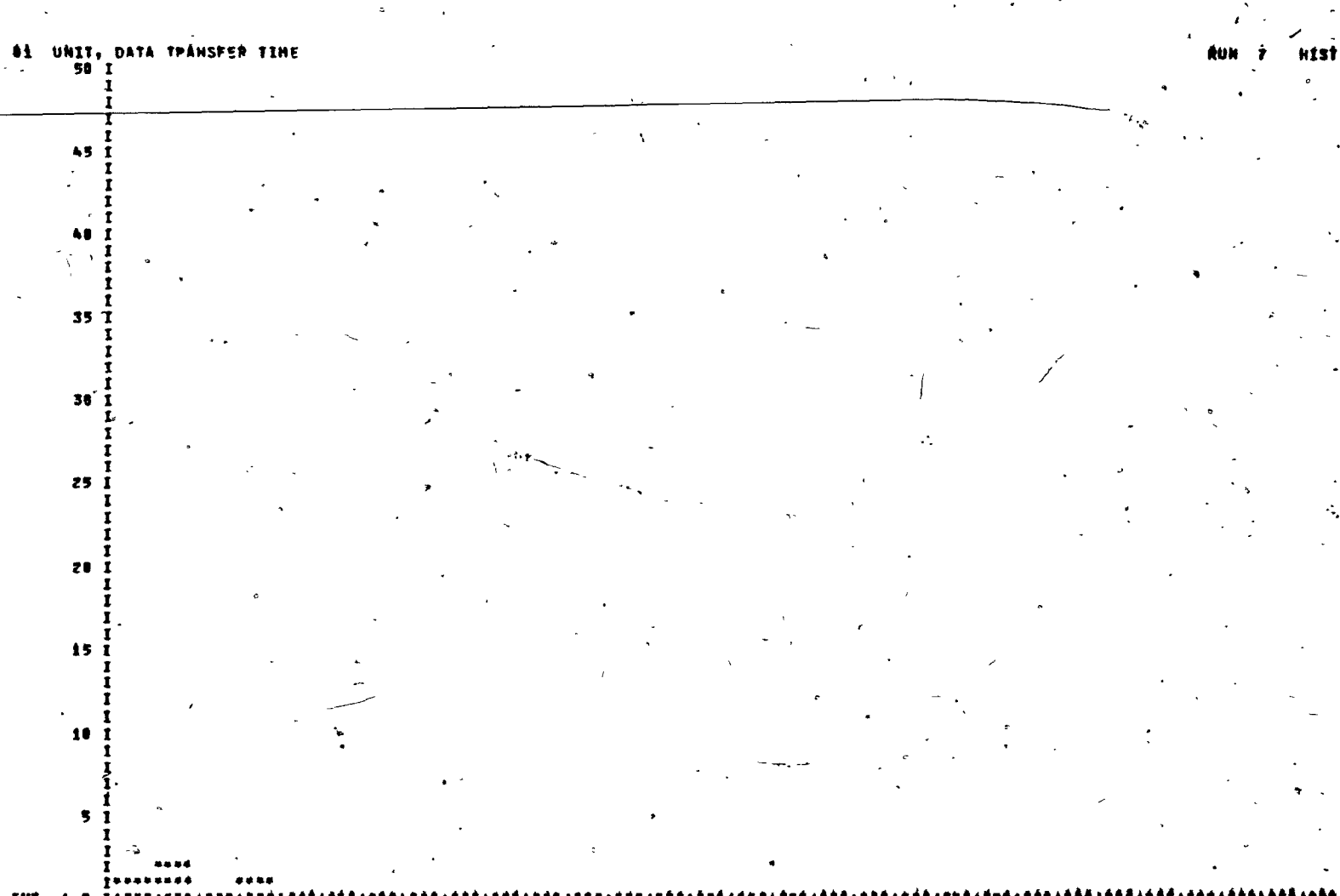
RUN 7 HIST 101



X Y-AXIS	100 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		
X X-AXIS	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0		
GROUPED PERCENTAGES (X)	0.00	0.00	0.00 0.00 0.00
TOTAL Y-AXIS	4	MAXIMUM Y-AXIS	4
TOTAL X-AXIS	0	MAXIMUM X-AXIS	0
AVERAGE (X-AXIS)	0.00	STANDARD DEVIATION	0.00
MAXIMUM OBSERVED	0		

01 UNIT, DATA TRANSFER TIME

RUN 7 HIST 102



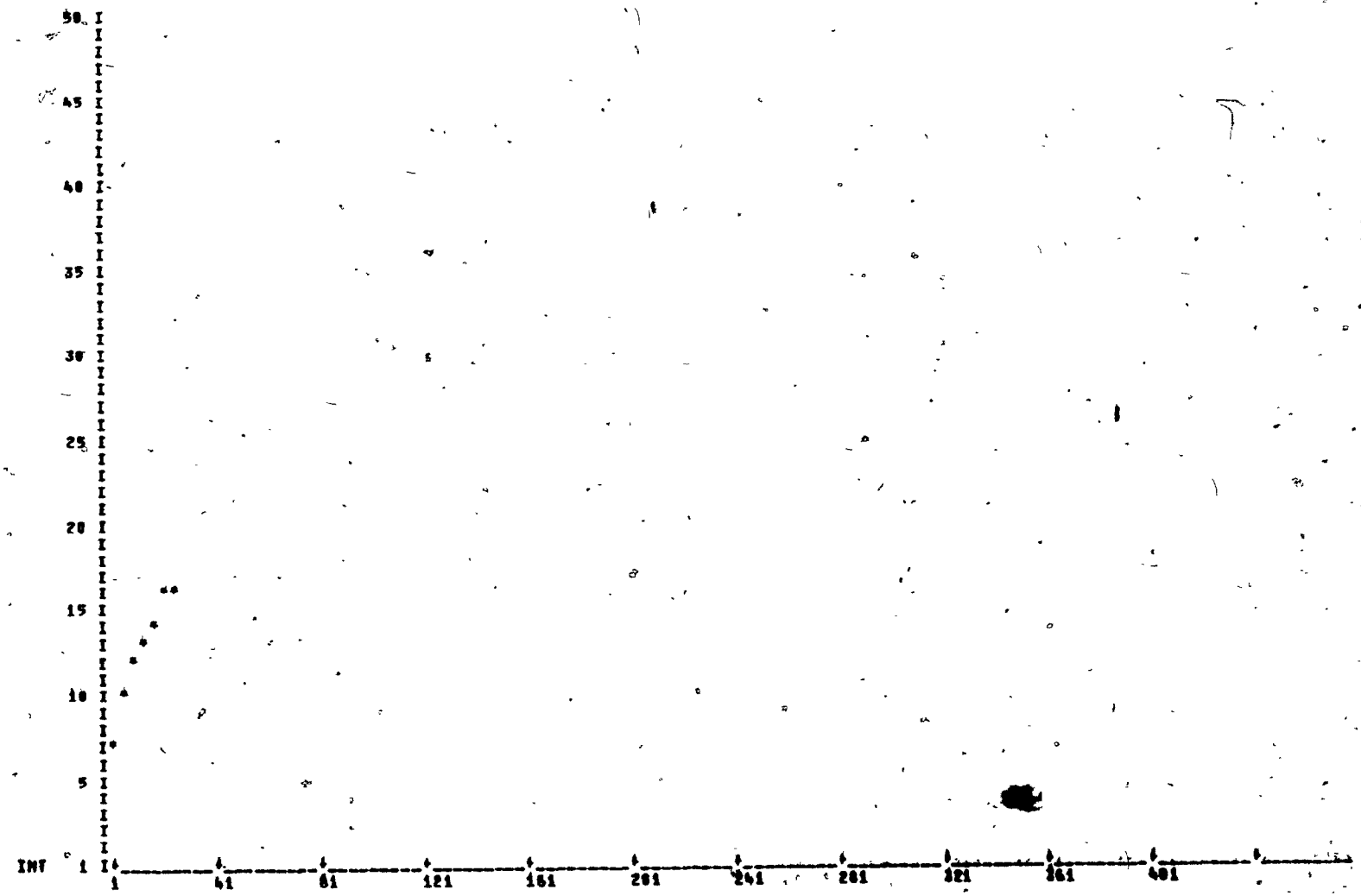
X Y-AXIS	25	50	0	25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X X-AXIS	12	42	0	45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GROUPED PERCENTAGES (X) 100.00 0.00 0.00 0.00 0.00 0.00

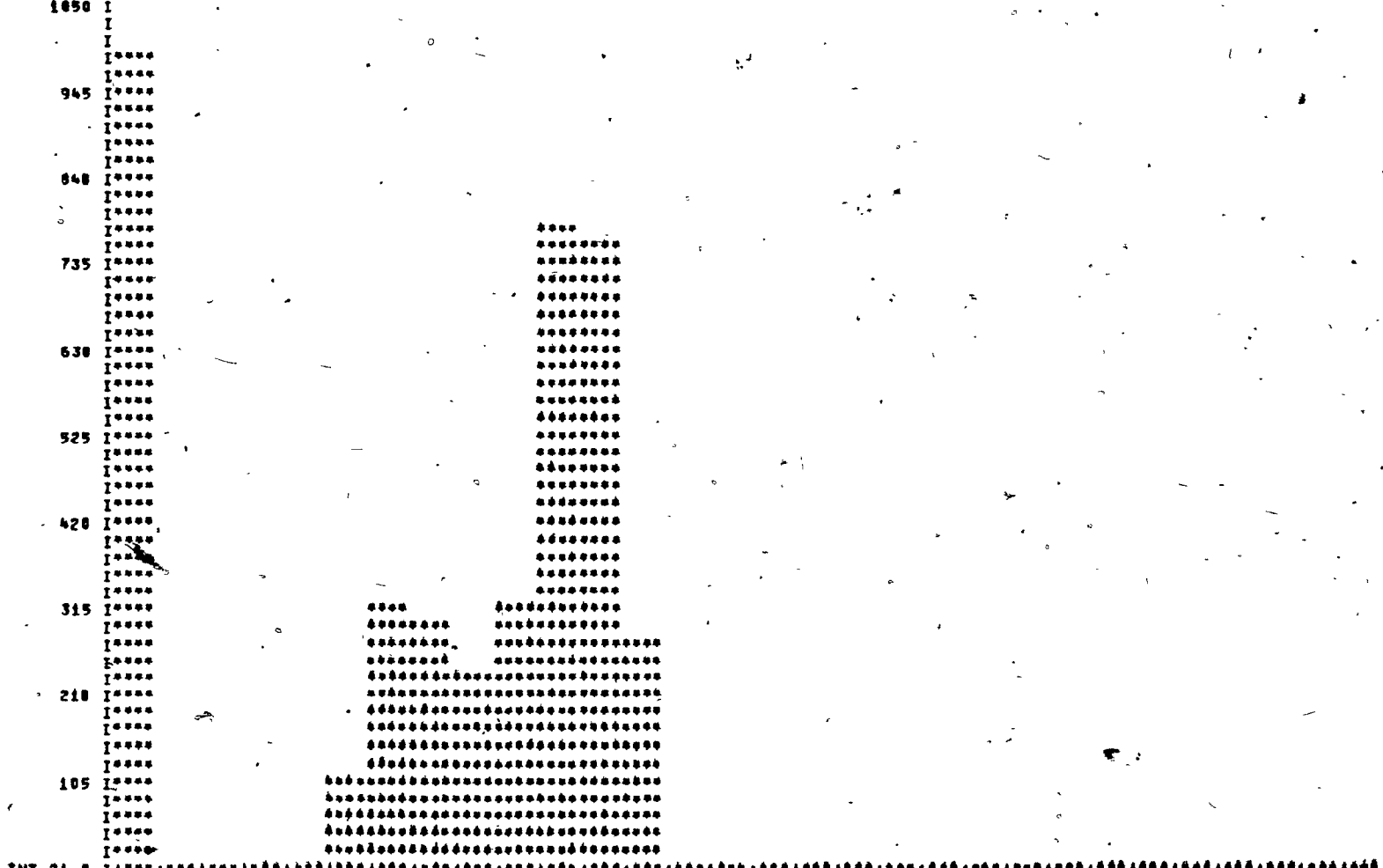
TOTAL Y-AXIS	4	MAXIMUM Y-AXIS	2
TOTAL X-AXIS	33	MAXIMUM X-AXIS	15
AVERAGE (X-AXIS)	8.25	STANDARD DEVIATION	4.09
MAXIMUM OBSERVED	15		

02. SEEK TIME IN MSEC PER SEEK LENGTH-CYLINDERS

RUN 1 HIST 183



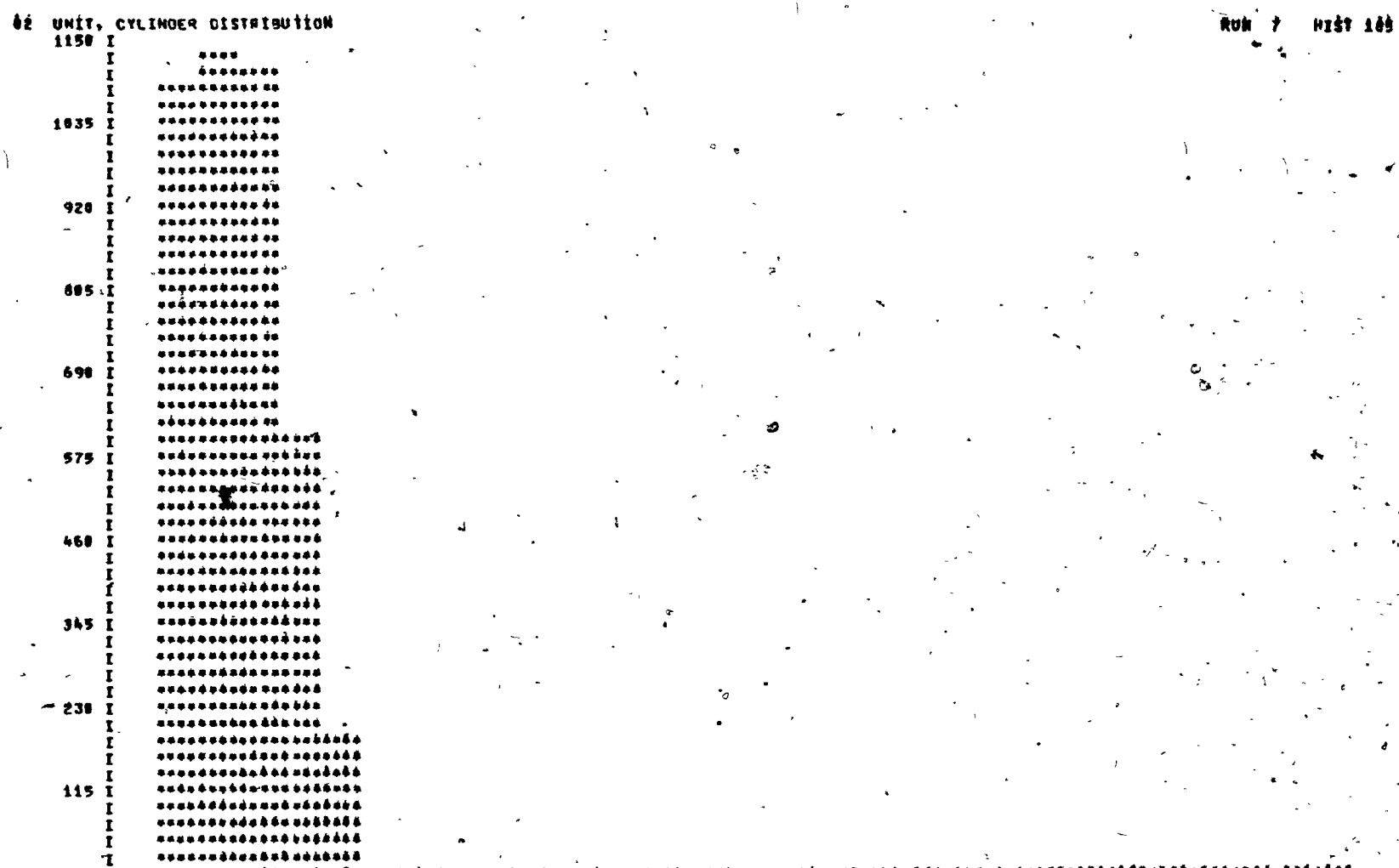
02 UNIT, SEEK-TIME PER SEEK RUN 7 HIST 184



% Y-AXIS	24	0	0	0	0	3	0	7	6	0	19	13	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	0	0	0	0	0	2	0	7	6	9	26	30	13	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GROUPED PERCENTAGES (X) 1.84 84.48 13.76 0.00 0.00

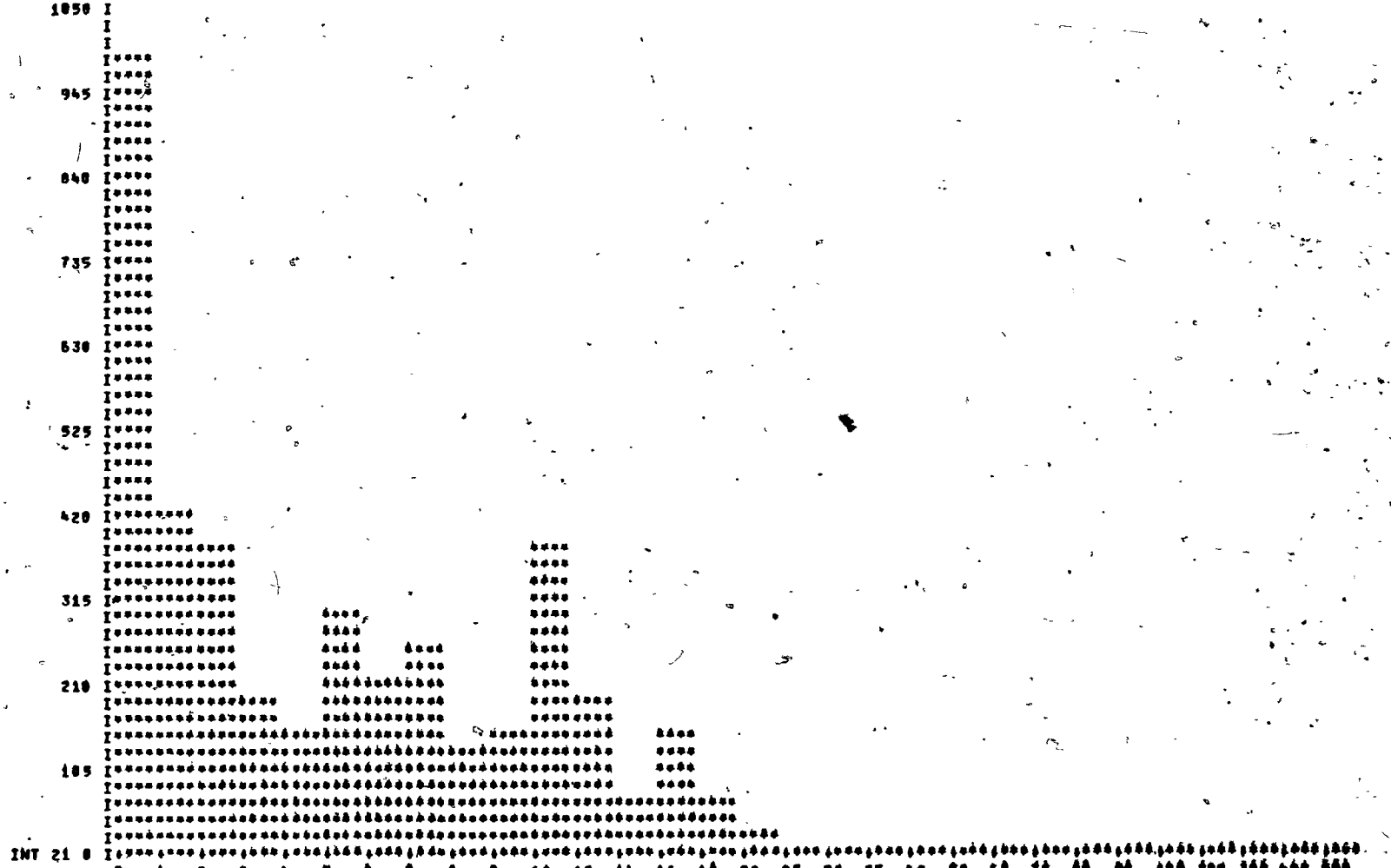
TOTAL Y-AXIS	4192	MAXIMUM Y-AXIS	1007
TOTAL X-AXIS	31698	MAXIMUM X-AXIS	9461
AVERAGE (X-AXIS)	7.56	STANDARD DEVIATION	4.85
MAXIMUM OBSERVED	17		



% Y-AXIS	0	26	27	27	15	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	0	14	23	32	23	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
GROUPED PERCENTAGES (X)					100.00						0.00						0.00						0.00						0.00		

TOTAL Y-AXIS	4192	MAXIMUM Y-AXIS	1149
TOTAL X-AXIS	59840	MAXIMUM X-AXIS	19108
AVERAGE (X-AXIS)	14.27	STANDARD DEVIATION	9.72
MAXIMUM OBSERVED	471 ON CYLINDER	7	

02 UNIT, TRAVELED-LENGTH RUN 7 HIST 186



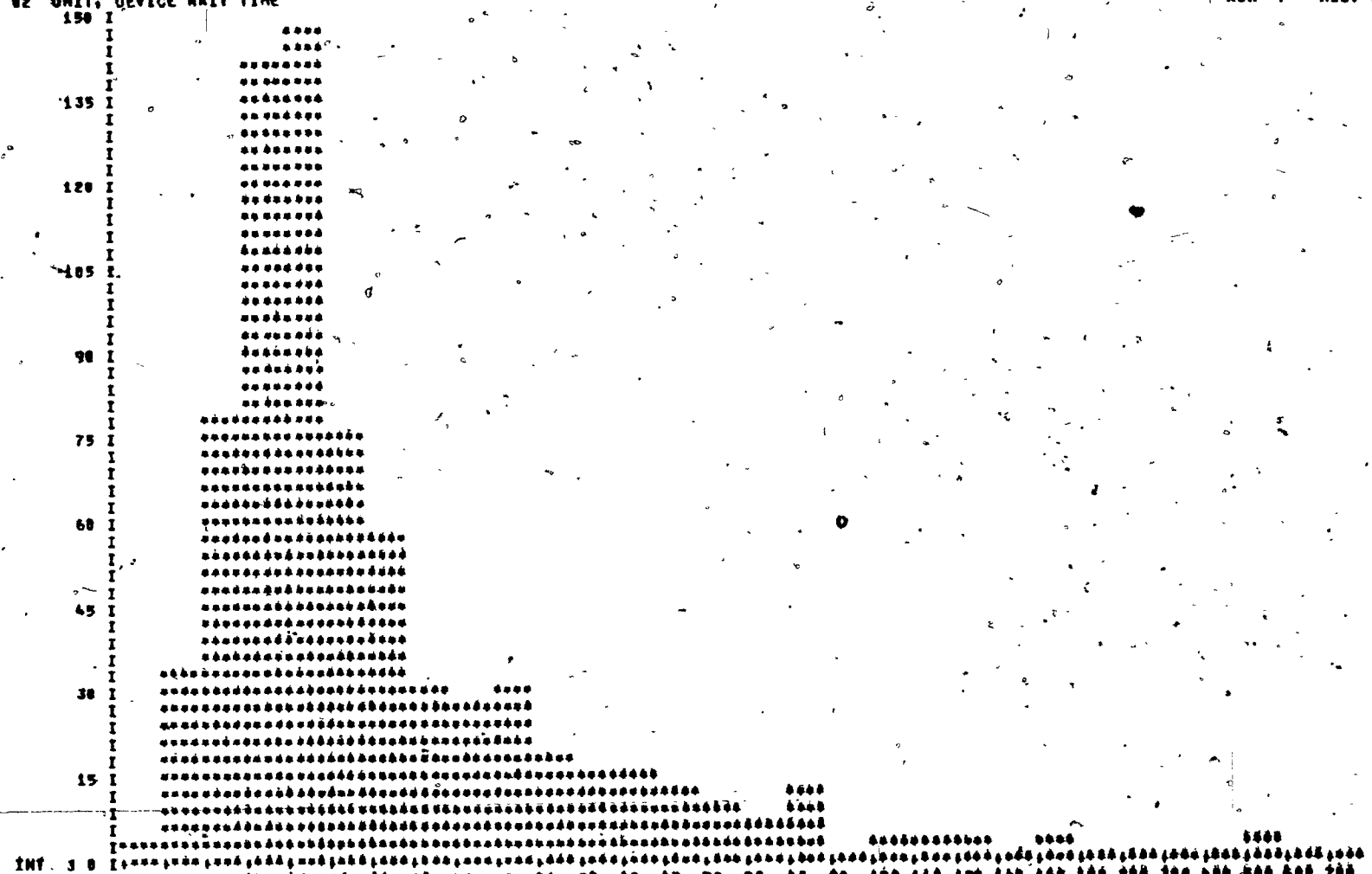
% Y-AXIS	24	10	9	5	4	7	5	6	3	4	9	5	2	4	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
% X-AXIS	0	2	4	3	3	7	6	8	5	6	19	11	5	13	6	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0

GROUPED PERCENTAGES (%) 17.94 54.96 27.10 0.00 0.00

TOTAL Y-AXIS	4192	MAXIMUM Y-AXIS	1006
TOTAL X-AXIS	22848	MAXIMUM X-AXIS	4142
AVERAGE (X-AXIS)	5.26	STANDARD DEVIATION	5.25
MAXIMUM OBSERVED	27		

02 UNIT, DEVICE WAIT TIME

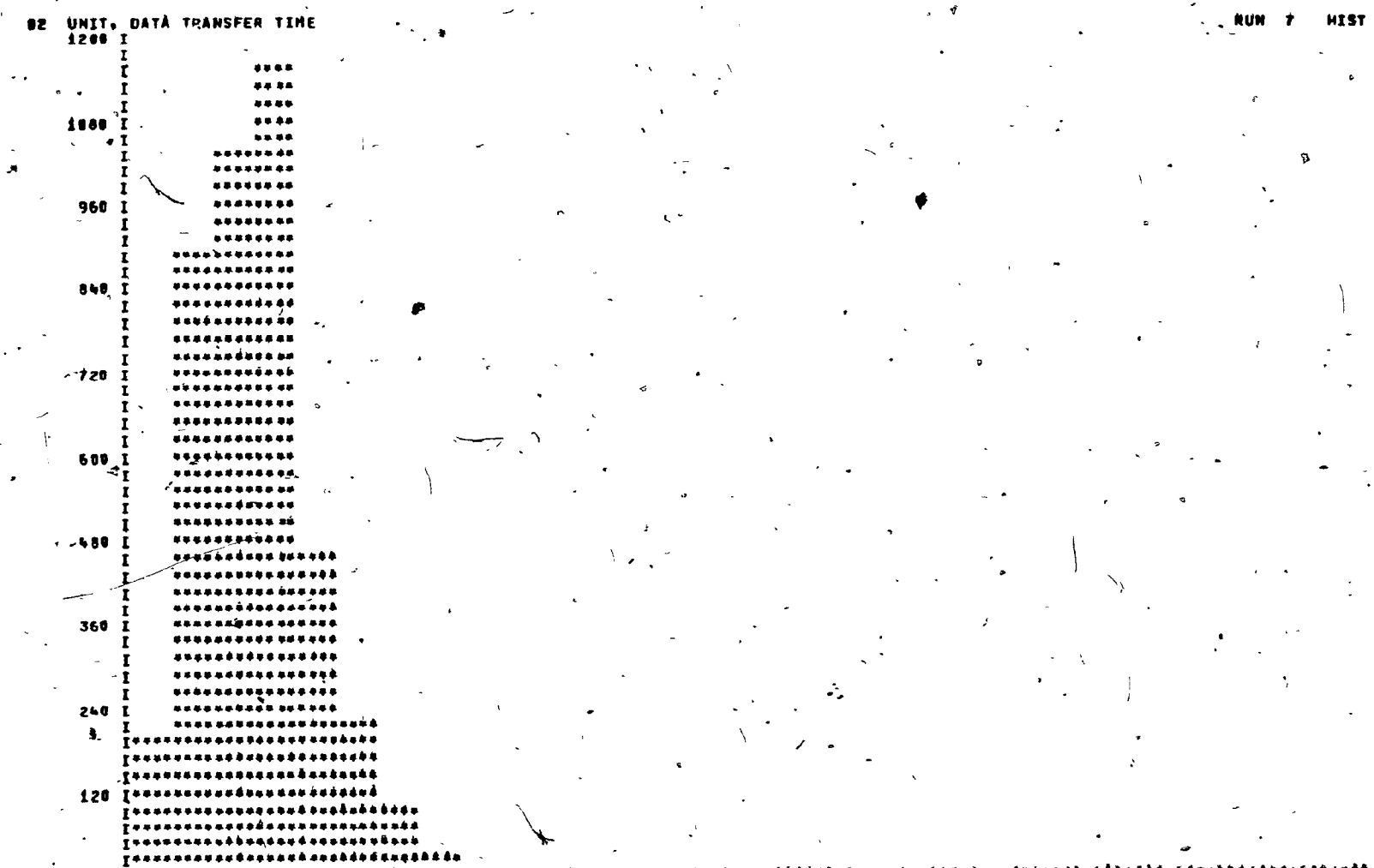
RUN 1 HIST 187



% Y-AXIS	1	5	10	19	20	10	8	4	4	4	3	2	2	2	1	1	2	0	0	0	1	0	0	0	0	0	0	0	0	0
% X-AXIS	0	1	4	9	12	8	7	4	4	6	4	3	4	3	2	2	4	1	1	1	2	1	2	0	0	0	0	6	5	3

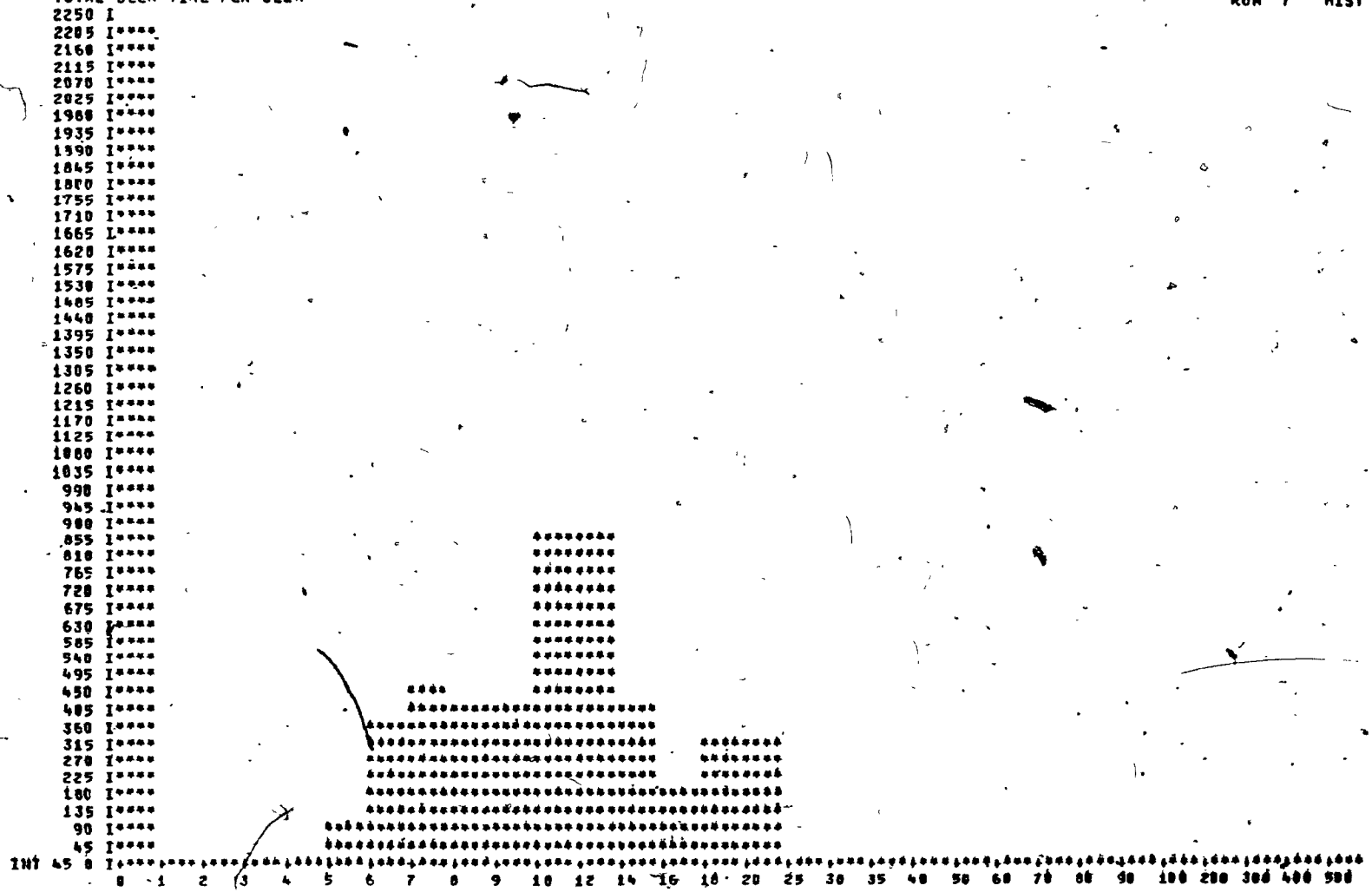
GROUPED PERCENTAGES (X) 33.00 26.47 18.58 7.35 13.00

TOTAL Y-AXIS	750	MAXIMUM Y-AXIS	147
TOTAL X-AXIS	26470	MAXIMUM X-AXIS	3223
AVERAGE (X-AXIS)	35.29	STANDARD DEVIATION	96.76
MAXIMUM OBSERVED	765		


82 UNIT, DATA TRANSFER TIME
RUN 7 HIST 128
1200
1080
960
840
720
600
480
360
240
120
INT 24 0
0 5 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 90 100 110 120 140 160 180 200 300 400 500 600 700
% Y-AXIS 5 21 25 28 11 6 2 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
X X-AXIS 1 10 19 30 16 10 5 2 1 0 0 0 0 0 0 0 0 0 0 0 1 4 2 0 0 0 0 0 0 0
GROUPED PERCENTAGES (X) 84.54 7.81 .40 7.25 0.00
TOTAL Y-AXIS 4192
TOTAL X-AXIS 66123
AVERAGE (X-AXIS) 15.77
MAXIMUM OBSERVED 165
MAXIMUM Y-AXIS 1167
MAXIMUM X-AXIS 19810
STANDARD DEVIATION 13.83

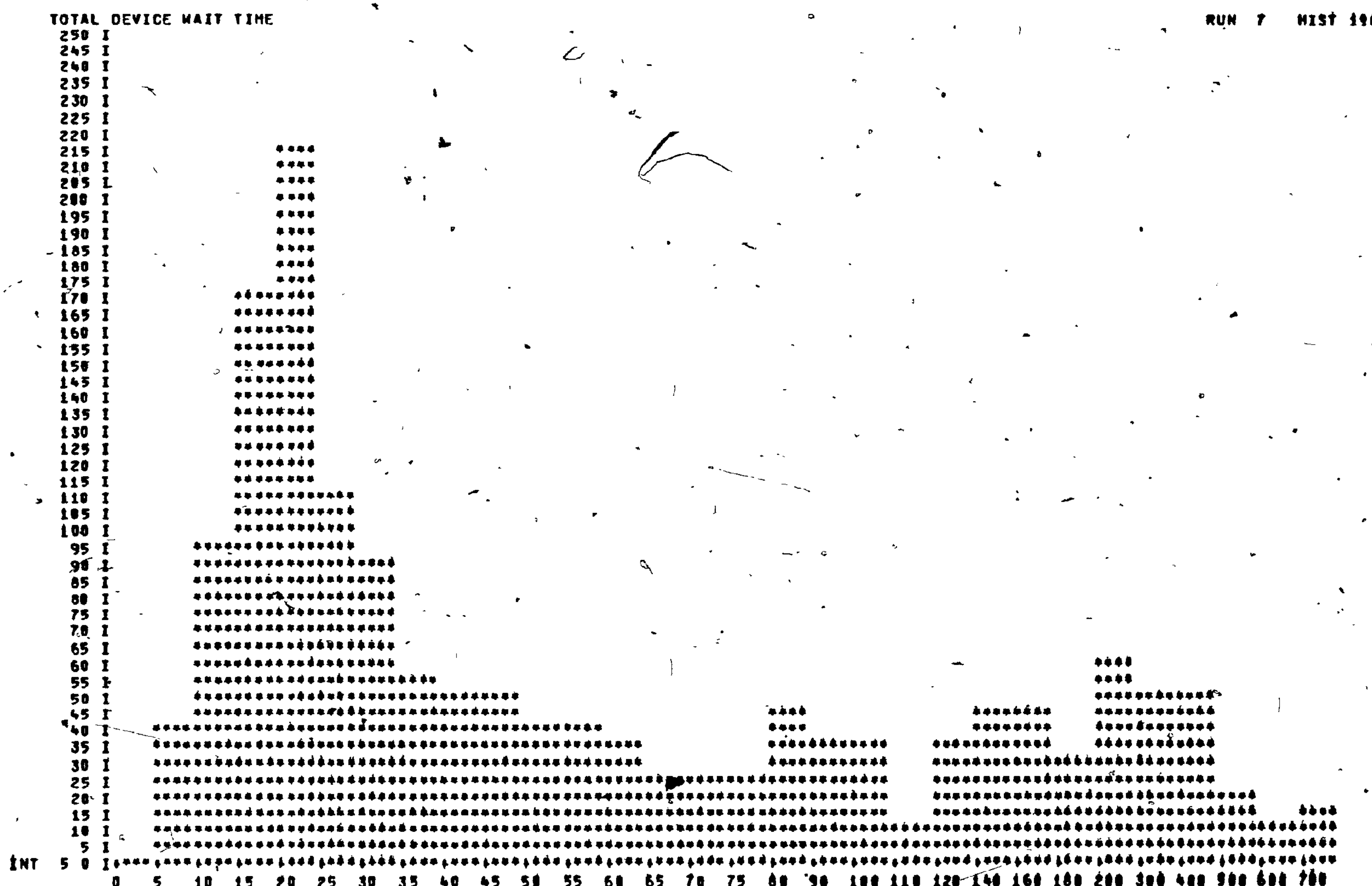
TOTAL SEEK-TIME PER SEEK

RUN 7 HIST 189



	0	1	2	3	4	5	6	7	8	9	10	12	14	16	18	20	25	30	35	40	50	60	70	80	90	100	200	300	400	500
Y-AXIS	31	0	0	0	0	2	5	7	6	6	12	12	6	3	5	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
X-AXIS	0	0	0	0	0	1	4	6	6	7	16	19	11	6	11	13	1	0	0	0	0	0	0	0	0	0	0	0	0	0
GROUPED PERCENTAGES (X)					1.65						57.26						41.44				.25									0.00

TOTAL Y-AXIS	7220	MAXIMUM Y-AXIS	2240
TOTAL X-AXIS	58169	MAXIMUM X-AXIS	11899
AVERAGE (X-AXIS)	8.06	STANDARD DEVIATION	6.63
MAXIMUM OBSERVED	73		

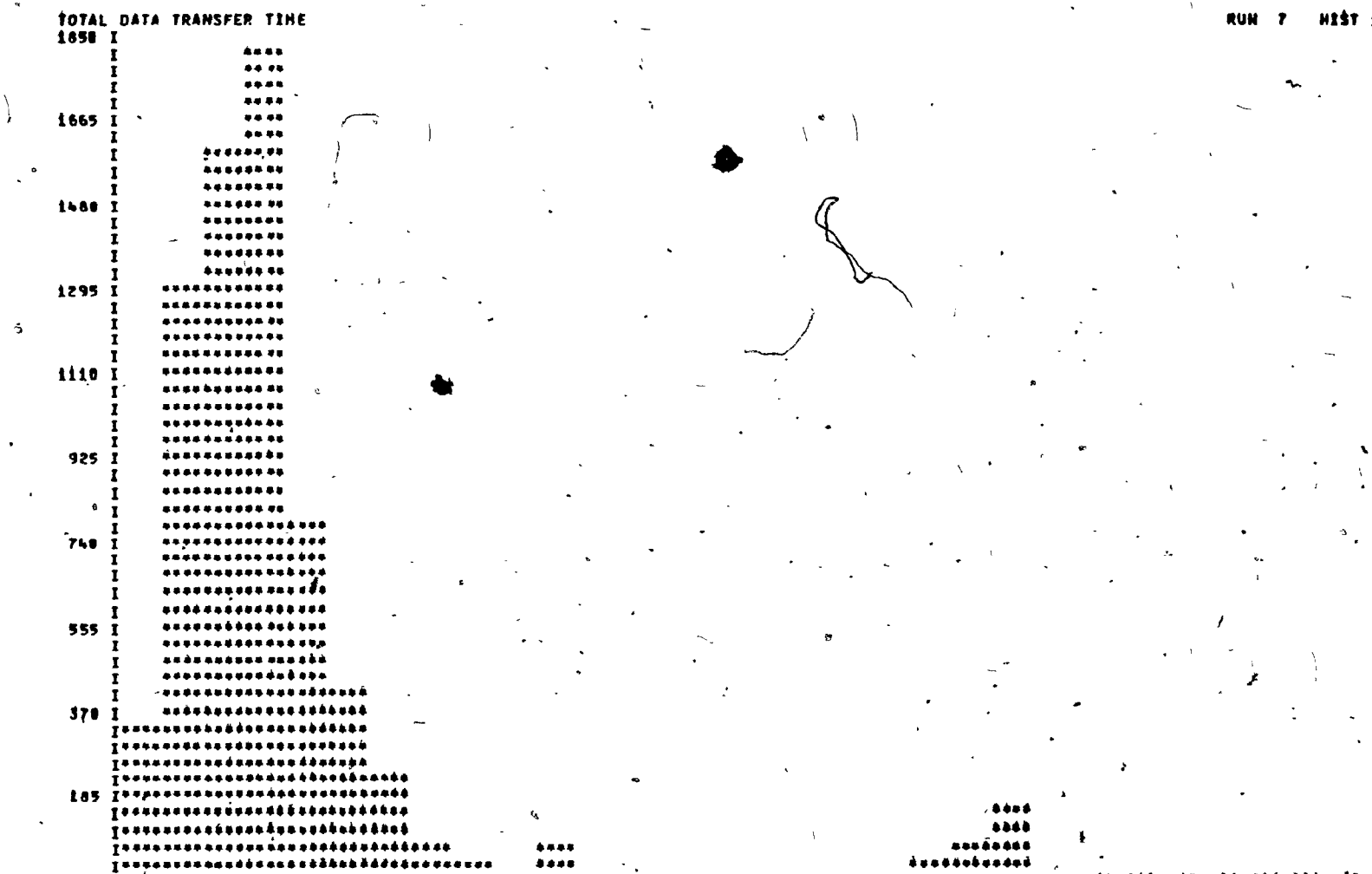


	0	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	90	100	110	120	140	160	180	200	300	400	500	600	700
% Y-AXIS	0	3	6	11	13	7	6	4	3	3	2	2	2	2	2	2	3	2	2	1	2	3	3	2	4	3	3	1	1	1
% X-AXIS	0	0	1	2	3	2	2	1	1	1	1	1	1	1	1	1	2	2	2	1	3	4	5	4	9	11	14	8	9	8
GROUPED PERCENTAGES (X)						7.37						8.46						9.25						19.23						55.70

TOTAL Y-AXIS	1606	MAXIMUM Y-AXIS	215
TOTAL X-AXIS	164496	MAXIMUM X-AXIS	23595
AVERAGE (X-AXIS)	102.43	STANDARD DEVIATION	143.89
MAXIMUM OBSERVED	909		

TOTAL DATA TRANSFER TIME

RUN 7 HIST 191



% Y-AXIS	5	18	22	25	11	6	3	2	1	0	1	0	0	0	0	0	0	0	0	1	1	2	0	0	0	0	0	0	0	0
% X-AXIS	1	6	12	19	11	7	5	3	2	1	2	1	1	0	0	0	2	1	1	3	6	15	2	0	0	0	0	0	0	0

GROUPED PERCENTAGES (X) 56.16 12.56 4.34 26.96 0.00

TOTAL Y-AXIS	7228	MAXIMUM Y-AXIS	1829
TOTAL X-AXIS	159255	MAXIMUM X-AXIS	38684
AVERAGE (X-AXIS)	22.06	STANDARD DEVIATION	27.63
MAXIMUM OBSERVED	165		

C H A P T E R V

ANALYSIS AND RESULTS

5.1 Introduction

Several runs have been executed from KRONOS 2.1 at level 388 to NOS 1.1 at level 419. It was necessary to stop at NOS 1.1 because in subsequent releases of NOS there was no space left in MTR to insert the code to record the I/O activity. This chapter describes the results from the last four runs.

The extremely large number of histograms generated by the analysis program (about 200 for each run), raises the question of how to present them in a comprehensible manner? To aid understanding of these histograms, they are divided into four classes: Job statistics, PP statistics, Ch statistics, and Disk statistics. These classes are not independent, however the separation is necessary for better clarity. The data were gathered from approximately six minutes of trace for each run. The description of the workload used to drive the system in each case has been given in section 3.7.

Each run ran with the following configuration:

- 131K CM,
- 10 PPs,
- 15 CPs,
- Batchio and Magnet enabled,

- 12 Chs connected as follows:
  - Ch 2, 3 and 4 to the disks (see next paragraph),
  - Ch 10 (octal) to the Console,
  - Ch 12 (octal) to the unit record equipment (Card Reader, Card Punch, and Line Printer),
  - Ch 13 to the tape units (7-track and 9-track),
  - Ch 0, 1, 5, 6, 7, 11 unused.

For each of the four runs the disks were connected via channels 2, 3 and 4 as follows:

- Run 6: Ch 2 and 3 to Unit 0,
- Run 7: Ch 2 to Unit 0,
  Ch 4 to Unit 2,
- Run 8: Ch 2 and 3 to Unit 0,
  Ch 2 and 4 to Unit 2,
- Run 9: Ch 2 to Unit 0.

In addition, Unit 1 (removable pack) was connected to channel 2 in all experiments, but access to it by workload programs was disabled. The system configuration for each run is shown in appendix D, as output of the PREPROCESSOR program.

Run 6, because of the very large channel activity (see table 5.3-3), did have a few table overflows, nevertheless for the large number of events it does not really make a noticeable difference. It might have some effect in routines with a very small number of calls.

In some tables the time units are not specified due to lack of space, however in all tables times are in msec. unless otherwise specified.

5.2 PP Routines

In order to promote a better understanding of the PP programs found in the following pages, a list and a summarized definition of these PP program that have been used by the system during our experiments is given below.

CIO - Combined Input/Output. CIO processes input/output requests for CPU programs.

CMS - Check Mass Storage. CMS surveys all MS devices and verifies that proper physical devices are mounted or makes them available for user access if possible. This verification takes place every 60 1SP cycles.

CPM - Control Point Manager. CPM is a general PP program to be used by CPU programs to reference or alter job control information in the CP area.

DIS - Job Display. DIS provides a convenient means to alter the running of a job, or if called to a blank CP to initiate the operation of utility programs. DIS provides an interpreted display of the exchange area for the job, as well as the status of the job. Keyboard entries are provided to allow the user to alter CM in several formats,

and to execute control cards as if they had entered the system with an input file.

DSD - Dynamic System Display. DSD is loaded in PP1 at deadstart time and remains there throughout execution of the system. DSD provides an overall status display for all currently running jobs via the 6612 display. The keyboard of the 6612 is monitored by DSD and is used for operator communication to the system.

D00 - Extract Error Text. D00 is a routine that will extract messages from specially created system text decks to aid in analyzing error situations returning from a product set. By using an error number and proper system text deck an error diagnostic will be transmitted to the dayfile and/or to a specified CM buffer. All system text decks to be used must be MS resident.

LDR - Overlay Loader. LDR loads overlays in response to CPU program requests.

LFM - Local File Manager. LFM performs various file managing tasks for a job.

PFM - Permanent File Manager. PFM is a permanent file driver capable of creating permanent files on any system MS device. Files may be of any length and are allocated on the device according to their length. PFM will perform all the necessary tasks to complete the permanent file

request. These tasks are search, create or modify users file catalog and transfer file from MS to MS if required. PFM may be called by any routine that sets up the proper call block in CM.

QFM - Queue File Manager. QFM performs tasks for queue management programs.

RPV - Reprieve CP Program. RPV provides the ability for a CPU routine to get control back after a specified normal or abnormal termination condition. There are two cases in which RPV may be called: one is to initialize, the other is to reset.

STS - Status Processor. STS provides MS devices status, file status, or the PRU count of a file.

TCS - Translate Control Statement. TCS translates control statements.

1AJ - Advance Job Status. 1AJ is called to advance an active job (i.e., to read the next control card and determine what is to be done next for this job).

1BA - Batchio Auxillary Processor. 1BA is called by 1CD to process MS transfers and special functions impossible or inconvenient to perform in 1CD.

1CD - Batchio Combined Driver. 1CD is the BATCHIO driver for up to 8 devices of 3 types:

3256/501/505 - 3555/512 - 580 Line Printers,

3446/415 Card Punch,

3447/405 Card Reader.

Any combination up to 8 of these devices may be driven. Each of the device types has its own conversion table which is read out of CM, any time it is not loaded. 1BA is called to perform MS transfers and other functions such as accounting.

1CJ - Complete Job. 1CJ performs all of the job's termination procedures.

1CK - System Check Point. 1CK may be called to process system checkpoints.

1DL - Display Overlay Loader. 1DL may be called to load overlays from MS for display programs (e.g., DSD).

1DS - DSD Request Processor. 1DS processes functions for DSD which are not possible for DSD to process. 1DS is also called to enter jobs for 1AJ in certain cases.

1IO - Batchio Manager. 1IO performs scheduling of all processes operating at the BATCHIO CP. This includes:

- searching for the highest priority OUTPUT and PUNCH files,
- checking for "ready" status on any card readers,
- managing of buffer storage for the above,
- posting of error condition messages for the above.

1MA - Monitor Auxillary Processor. 1MA is called by the system monitor to perform functions which can not be processed immediately by the monitor.

1MT - PPU Magnetic Tape Executive. 1MT processes various tape functions for MAGNET (the CPU magnetic tape executive).

1RI - Rollin Job. 1RI performs job rollin in response to a calling program such as the job scheduler or the system display.

1RO - Rollout Job. 1RO performs job rollout in response to a calling program such as the job scheduler or the system display, or a dump field length function from 1AJ.

1SJ - Job Scheduler. 1SJ performs scheduling of jobs from the input and rollout queues.

1SP - Evaluate Priorities. 1SP is called periodically by 1SJ to perform the following functions:

- evaluate priorities of files in the various queues;
- check CM time slices for jobs at CPs. If a time slice is expired, a job's priority is lowered and if it is TXOT with output available, it is rolled out;
- check for device checkpoint requests and call 1CK if any found;
- check for device initialize requests and call CMS if any found;
- check device MS tables for any disk errors which the

driver could not place directly in the errlog. If any of these found, 2SP is loaded to place the error message in the errlog;

- all timed event jobs are made eligible for scheduling when the desired event has occurred or if their time has expired.

5.3 Overall and Job Statistics: Histograms 58-71,79-84

Table 5.3-1 summarizes the job statistics. It can be seen in this table that the CPU time for Batch Origin Type (BCOT = Origin Type 1) is 221 sec. and it is the same for the four experiments, they should not be different! Thus these figures verify to some extent the accuracy of our experiments. Also, the number of batch origin jobs are equal, the number of RA changes match the number of storage moves, necessarily an RA change is issued only for a storage relocation.

Notice that the number of System Origin jobs (SYOT = Origin Type 0) is bigger than the four expected (MAGNET, BATCHIO, SYSTEM and the Sample program), this is caused by the system job CMS which comes in every 60 seconds. Table 5.3-2 shows the CPU utilization for the four runs. Runs 7 and 8 show a better CPU utilization. These workloads did have however a larger idle time than expected, I believe it is because of the restricted number of channel and disk units. Similar studies undertaken by the author (LOCIC 78,79) show a smaller idle time, however two channels and six disks were

used. It can be seen in table 5.3-2 that increasing the number of disk units the idle time decreases as much as 60% (Run 7 and Run 9). In table 5.3-3 it can be noticed that the proportion of DSWM functions increase drastically as soon as there are more channels than disks.

Runs 7 and 8 (with 2 or 3 Chs and 2 Disks) did have the best turnaround, with shorter elapsed time and smaller system CPU time. The system has been used more efficiently; more PP requests but less PP hold time; more storage moves, more rollouts, higher multiprogramming level, thus shorter real time.

The figures for PP library search account only for primary overlays such as: 1MA, CIO, 1SP, etc. [After a call has been issued (CIO, QFM, PFM), there is a sequence of loads for different PP routines with different level of overlays, and only those beginning with a letter or a 1 (one) have been recorded.]

Table 5.3-1 Job Statistics

	Run 6	Run 7	Run 8	Run 9
Total Number Of Jobs	40	40	40	41
SYOT Jobs	9	9	9	10
BCOT Jobs	31	31	31	31
Elapsed Time sec.	283	248	250	269
Total SYOT Time "	1151	1045	1052	1171
Total BCOT Time "	1630	1434	1328	1400
CPU Time "	244	238	239	241
SYOT CPU Time "	23	17	18	20
BCOT CPU Time "	221	221	221	221
CPU Time Between RO "	5.52	3.07	2.16	5.39
No. Rollouts SYOT	0	0	0	0
" " BCOT	20	27	25	18
No. PP Requests SYOT	2346	2038	2077	2225
" " " BCOT	5552	6322	6383	5759
No. PP Library Search	8049	8544	8662	8144
No. RA Changes	282	387	336	295
No. Storage Moves	282	387	336	295
Max. No. Jobs In The System At One Time	15	17	17	14
Multiprogr. level SYOT	4.12	4.22	4.22	4.37
" " BCOT	4.15	4.63	4.60	4.22

Table 5.3-2 CPU Utilization

	Run 6	Run 7	Run 8	Run 9
CPU Utilization	86.22%	95.97%	95.60%	89.59%
CPU Idle Time	13.78%	4.03%	4.40%	10.41%
CPU Time used by System	9.43%	7.14%	7.53%	8.30%

Table 5.3-3 System Activity Gathered by the Probe

	Run 6	Run 7	Run 8	Run 9
Ch activity (# events)	197,748	61,921	98,756	71,564
DSWM functions	91.74%	73.16%	82.89%	77.53%
PP actvity (# events)	31,000	33,082	33,452	31,303
Percent of equal time(*)	19.68%	26.65%	24.29%	28.69%

(*) Equal time events occur when a sample in CDUMP has the same time field as a sample in FDUMP.

5.3.1 Memory Utilization: Histograms 72-78

Table 5.3.1-1 summarizes the memory utilization. The requests for the same FL can be explained knowing that the dynamic loader issues an FL request for the next step automatically, thus it could request an FL which the job is already running. Some jobs are rolled out after two to six msec. of CPU time, this is also due to the dynamic loader. The loader will request memory and if there is not enough memory the job is necessarily rolled out. Since the system was quite heavily loaded, memory was necessarily in shortage (78% to 80% of the time memory was full).

Table 5.3.1-1 Memory Utilization (FL sizes are in octal)

	Run 6	Run 7	Run 8	Run 9
Total FL changes	210	214	211	205
FL changes SYOT(*)	1	0	0	0
FL changes BCOT(*)	97	89	89	96
Total FL requests (**)	184	216	205	190
Requests for same FL	8.33%	9.09%	9.09%	8.33%
Avg. job size SYOT	7.2K	7.2K	7.2K	7.1K
Avg. job size BCOT	53K	53K	53K	53K
Avg.size FL change BCOT(*)	21.6K	16.4K	17.1K	21.4K
max. FL change "	74K	70K	100K	100K
Avg. wait time FL "	169.67	274.47	175.52	144.59
Avg.CPU time betw.FL "	735.86	798.22	792.08	740.67
Memory utilization	340-400K	340-400K	340-400K	340-400K
by percent of time	60%	80%	78%	62%
FL size by	0-20K	0-20K	0-20K	0-20K
percent of time SYOT	100%	100%	100%	100%
FL size by	240-320K	260-300K	260-300K	260-300K
percent of time BCOT	20%	52%	62%	38%

(*) These numbers exclude roll-in, roll-out, beginning-of-job and end-of-job FL changes.

(**) Indicates requests coming from the Ch activity.

5.4 PP Statistics: Histograms 1-5.7

The histograms generated for the PPs are quite detailed: there have been up to 26 different PP programs used by the system, where 19 of them used data channels to perform the specific function.

The following is a summary description for each PP routine, as observed using the program QUERYFILE (see appendix D).

CIO - has been called, necessarily, by everyone who did read/write to the Mass Storage.

CMS - has been called to check the Mass Storage (removable packs) every 60 sec.

CPM - has been called by every program to get the CPU time used by each job step, to be written to the dayfile.

DIS - has been used to invoke the SAMPLE program and to submit the workload (the workload was on tape and has been submitted through the control card LDI).

DOO - has been used only by COBOL jobs to send dayfile messages, usually error messages.

LDR - has been called by every job which needed to load a compiler.

LFM - has been used by every job for file handling.

QFM - has been used by the submitting job to enter the benchmark's jobs into the input queue.

RPV - has been used in the termination process of all Fortran programs.

STS - has been used by the Cobol programs only.

TCS - is generally called by 1AJ as its overlay, but for some obscure reason it has been called by two jobs to translate Return, Writef, Rewind, and Sortmrg control cards.

1AJ - has been called by every job to process the next step.

1CJ - has been used to terminate each program.

1CK - was called by the system every 15 sec. (approximately).

1DL - has been used by the submitting job and by the system.

1DS - has been called by the system only.

1IO - has been invoked by BATCHIO every second to check if any activity in the unit record equipment.

1MA - has been called by everyone who needed it, for requesting storage and sending dayfile messages.

1MT - has been used by MAGNET and the submitting job. When MAGNET is idle or activity is very low, 1MT is called every 2 sec. to check if there is anything to do.

1RI - was called by jobs to be rolled-in.

1RO - was called to roll-out jobs.

1SJ - was called by the system and all jobs (1SJ is called only by the system, and then moves to a new CP), to enter a job from the input queue or the roll-out queue.

1SP - has been called by the system and a few jobs (1SP is called only by the system, and then moves to a CP) which needed a priority evaluation.

Table 5.4-2 shows the benefit of having 2 disks for I/O operations. Also it can be seen that the number of occurrences for PRU is smaller than that for CIO number, it might be because a CIO is issued and no transfer done; one of the reasons could be a premature exit from CIO, such as: a file not found, a permanent file busy, a wrong file name, etc.

The CPU time shown in the next and following tables is considered to be the time that the CPU did execute in parallel for the same CP (i.e., CIO call or other with no recall bit set).

Table 5.4-1 PP Utilization

	Run 6	Run 7	Run 8	Run 9
Total PP assigns	7909	8366	8464	7989
Percent PP0 assgn.	.01	.01	.01	.01
" PP1 "	.01	.01	.01	.01
" PP2 "	14.41	9.91	14.66	14.47
" PP3 "	14.82	14.85	6.82	8.49
" PP4 "	14.38	17.40	9.62	12.72
" PP5 "	3.69	7.78	12.16	11.63
" PP6 "	15.86	5.00	15.76	16.02
" PP7 "	8.62	15.14	11.35	13.28
" PP8 "	13.81	14.80	17.82	14.44
" PP9 "	14.39	15.10	11.79	8.92
Total PP hold time sec.	1821.13	992.72	1016.87	1450.08
" wait " "	60.57	3.03	5.63	6.93
" CPU " "	106.69	57.43	58.38	97.78
Avg. PP hold time ms.	230.26	118.66	120.14	181.51
" " " " (*) "	158.81	59.42	61.17	114.04
" " wait " "	7.66	0.36	0.67	0.87
" " CPU " "	13.49	6.86	6.90	12.24
Avg. PPs used by SYOT(**)	3.34	2.38	2.40	2.58
" " " BCOT	3.04	1.70	1.66	2.79
CPU time betwn req. SYOT	9.59	8.58	8.78	8.74
" " " " BCOT	40.72	35.75	35.46	39.17

(*) Average PP hold time excluding PP0 and PP1 which were the dedicated PPs for MTR and DSD.
(**) Including PP0 and PP1 (MTR and DSD).

Table 5.4-2 Occurrences and Average Hold Time per PP Routine

	Run 6		Run 7		Run 8		Run 9	
PRU(*)	(5273)	191.1	(5819)	77.1	(5943)	79.9	(5327)	157.6
CIO ms.	(5355)	140.8	(5906)	46.7	(6025)	49.1	(5405)	114.5
CMS "	(4)	1074	(4)	225.5	(4)	428.5	(5)	963.0
CPM "	(62)	464.6	(62)	358.4	(62)	309.9	(62)	378.0
DIS "	(1)	5376	(1)	4295	(1)	1712	(1)	4131
DSD "	(1)	282569	(1)	247788	(1)	249564	(1)	269488
DOO "	(10)	406.4	(10)	163.0	(10)	331.4	(10)	338.6
LDR "	(294)	363.2	(294)	267.2	(294)	286.0	(294)	320.7
LFM "	(85)	5.1	(85)	4.9	(85)	4.9	(85)	4.9
MTR "	(1)	282569	(1)	247788	(1)	249564	(1)	269488
QFM "	(63)	749.6	(63)	506.7	(63)	532.4	(63)	716.6
RPV "	(28)	4.3	(28)	2.4	(28)	2.2	(28)	2.1
STS "	(31)	3.1	(31)	3.2	(31)	3.1	(31)	3.1
TCS "	(3)	4.7	(3)	4.7	(3)	4.0	(3)	4.3
1AJ "	(113)	342.1	(123)	177.5	(120)	195.4	(112)	288.3
1BA "	(2)	1231		---		---		---
1CD "	(1)	175699		---		---		---
1CJ "	(37)	768.1	(36)	344.5	(36)	326.3	(37)	840.1
1CK "	(18)	447.8	(16)	321.9	(16)	281.0	(18)	361.9
1DL "	(10)	308.6	(10)	118.0	(12)	150.6	(13)	167.3
1DS "	(15)	192.9	(15)	94.7	(16)	149.4	(24)	186.3
1IO "	(151)	13.2	(231)	6.3	(237)	6.5	(255)	7.4
1MA "	(162)	10.3	(162)	7.2	(162)	6.5	(162)	4.6

Table 5.4-2 Occurrences and Average ... (continued)

	Run 6		Run 7		Run 8		Run 9	
lMT ms.	(933)	12.6	(710)	14.1	(699)	14.2	(818)	13.2
lRI "	(19)	470.5	(27)	319.5	(25)	285.6	(18)	470.2
lRO "	(20)	510.3	(27)	253.6	(25)	174.2	(18)	405.7
lSJ "	(476)	22.7	(505)	21.6	(494)	20.5	(509)	21.5
lSP "	(14)	1.6	(14)	1.5	(14)	1.6	(16)	1.6

(*) PRU accounts for all PP request which transferred PRUs.

Table 5.4-3 shows the CPU time used while executing the PP routine, i.e., CPU:I/O and CPU:PP overlap as explained in section 5.4.

Table 5.4-3 Average CPU Time (if occur.) per PP Routine

	Run 6	Run 7	Run 8	Run 9
PRU msec.	19.79	9.51	9.47	17.93
CIO "	19.48	9.30	9.26	17.58
CPM "	1.26	1.40	1.31	1.26
DIS "	2.00	---	77.00	88.00
DOO "	1.00	1.20	1.50	1.10
LDR "	1.29	1.41	1.44	1.48
LFM "	2.84	3.06	2.93	3.06
QFM "	1.08	1.13	1.05	1.02
RPV "	1.11	1.00	.93	.93
STS "	1.42	1.45	1.52	1.35
TCS "	.67	.67	.67	.67
1AJ "	.75	.58	.74	1.20
1CK "	8.17	9.94	9.25	18.00
1DL "	1.50	1.30	1.42	7.38
1DS "	2.07	2.07	7.13	1.71
1MA "	.82	.86	.86	.88
1MT "	1.05	1.37	1.45	1.11
1RI "	.11	.07	.72	.22
1SJ "	.27	.40	.18	.14

Table 5.4-4. Average Wait Time (if occur.) per PP Routine

	Run 6	Run 7	Run 8	Run 9
PRU msec.	47.36	2.88	4.93	6.62
CIO "	51.06	3.73	5.63	7.51
CPM "	11.06	1.00	11.24	8.07
DOO "	1.00	1.00	1.00	1.00
LDR "	30.99	1.02	2.01	9.17
LFM "	37.58	1.00	3.43	1.45
QFM "	26.46	1.00	5.37	1.67
RPV "	1.00	1.00	1.00	1.00
STS "	54.41	1.00	1.00	10.39
TCS "	1.00	---	1.00	1.00
1MA "	22.93	1.00	11.41	3.09

In collecting the wait time some problems have been encountered, in several instances the wait time was equal to the last four digits of the real time. Thus, for several routines such as: 1AJ, 1SJ, 1RO, 1RI, 1CJ, 1MT, 1IO, etc., the wait time has been discarded, therefore they do not show in this table. However, for the routines shown in this table the time is accurate.

5.5 Channel Statistics: Histograms 85-162

The histograms generated for the data channels are very detailed, there are as many as 19 PP programs that use some of the channels, and with 8 channels there are a very large number of histograms. The first 3 channels (2,3 and 4) were used for the disk subsystem. Channel 10 (octal) is the console channel and it is normaly used by the operator to monitor the system. Channel 12 (octal) is shared by the unit record equipment. Channel 13 (octal) is connected to the magnetic tape controller. Physical channel 14 (octal) is the clock, and is accessed only by MTR; when a PP in the pool makes a request for Ch 14, it is really interlocking on the MST. Channel 15 (octal) is used by the system as an interlock for the FNT.

During the validation of the data, some hidden problems have been discovered. The Ch hold time, which is computed as for the PP hold time, is not the true hold time since a Ch can be dropped and reassigned to some other PP while the disk arm is moving for the first request.

Something really astonishing happens when there are more channels than disk units. It has been found that more than one seek was moving simultaneously on the same unit! Impossible hardware-wise but possible software-wise, here is the reason: after a CIO call has been issued the only way to know if the unit is busy, is to issue an LDAM function. LDAM

will then interrogate the status bit in the DILL word of the MST table, which is set by the PPMTR function DSWM. However, when DSWM is called the logical unit reserved bit is cleared during the function execution. Thus an LDAM may be issued and interrogate the bit, finding it not set; therefore a seek function will be issued for that unit. Because the unit is really hardware busy the request seems to be waiting for a seek completion, but in reality it is waiting for the unit.

Table 5.5-1 Channel Statistics

		Run 6	Run 7	Run 8	Run 9
Total Ch assigns		6853	7670	7814	7316
Percent Ch2 assigns		48.27	26.75	32.05	76.79
" Ch3 "		31.91	---	7.21	---
" Ch4 "		.06	53.34	40.98	.07
" Ch10 "		.09	.07	.04	.08
" Ch12 "		4.96	9.04	9.10	10.46
" Ch13 "		7.76	4.56	4.52	6.01
" Ch14 "		3.33	3.10	3.02	3.14
" Ch15 "		3.60	3.16	3.10	3.44
Max. req. of Chs before the PP drop	Ch2	5	5	6	7
" " " " "	Ch3	4	-	4	-
" " " " "	Ch4	1	5	5	1
" " " " "	Ch10	4	3	2	4
" " " " "	Ch12	3	3	3	3
" " " " "	Ch13	1	1	1	1
" " " " "	Ch14	1	1	1	1
" " " " "	Ch15	10	13	10	13

Table 5.5-2 Time Distribution per Ch

	Run 6	Run 7	Run 8	Run 9
Total hold time sec.	1,097.8	617.7	611.1	761.5
Avg. hold time ms.	160.19	80.53	78.20	104.09
Avg.hold time(PRU) "	105.03	53.04	52.04	75.26
hold time Ch2	32.71%	38.70%	33.41%	62.45%
" " Ch3	24.46%	---	8.91%	---
" " Ch4	.01%	18.81%	14.50%	.01%
" " Ch10	25.74%	40.12%	40.84%	35.39%
" " Ch12	(*)16.06%	.74%	.72%	.73%
" " Ch13	.89%	1.41%	1.38%	1.24%
" " Ch14	.07%	.11%	.11%	.09%
" " Ch15	.07%	.12%	.12%	.10%
Total wait time sec.	262.1	242.9	174.2	548.9
Avg. wait time ms.	38.24	31.67	21.84	72.13
Avg.wait time(PRU) "	35.68	34.24	25.01	77.67
wait time Ch2	81.89%	98.90%	98.50%	99.98%
" " Ch3	17.19%	---	.68%	---
" " Ch4	.00%	1.02%	.60%	.00%
" " Ch12	.79%	.00%	.00%	.00%
" " Ch13	.13%	.08%	.22%	.02%

(*) As explained in section 5.1, Run 6 did have a few table overflows, and Ch12 is one of the consequences.

Table 5.5-3 Ch Time Distribution per OT

	Run 6	Run 7	Run 8	Run 9
Ch2 was used by SYOT	14.94%	15.26%	11.83%	12.00%
Ch2 " " " BCOT	85.06%	84.74%	88.17%	88.00%
Ch3 " " " SYOT	14.13%	---	8.17%	---
Ch3 " " " BCOT	85.87%	---	91.83%	---
Ch4 " " " SYOT	100%	8.61%	11.65%	100%
Ch4 " " " BCOT	---	91.39%	88.35%	---
Ch10 " " " SYOT	100%	100%	100%	100%
Ch12 " " " SYOT	100%	100%	100%	100%
Ch13 " " " SYOT	100%	100%	100%	100%
Ch14 " " " SYOT	15.79%	14.71%	14.83%	15.65%
Ch14 " " " BCOT	84.21%	85.29%	85.17%	84.35%
Ch15 " " " SYOT	100%	100%	100%	100%

Table 5.5-4 Ch2 Average Wait and Hold Time per PP Routine

		Run 6		Run 7		Run 8		Run 9	
		wait	hold	wait	hold	wait	hold	wait	hold
Total	ms.	64.9	108.6	117.1	116.5	68.5	81.5	97.7	84.7
PRU	"	50.2	100.1	106.2	109.5	63.3	78.7	79.8	77.0
CIO	"	9.0	71.6	19.6	45.6	19.5	44.7	11.9	43.7
CMS	"	665.2	385.5	563.8	381.0	233.7	169.6	766.6	317.7
CPM	"	355.0	262.7	468.6	282.1	237.0	127.6	503.6	276.2
DOO	"	399.7	365.5	309.2	298.7	603.2	468.9	975.0	584.4
LDR	"	48.2	221.7	78.4	153.8	86.3	172.5	124.3	147.0
QFM	"	444.4	245.0	269.4	198.1	254.5	136.5	522.7	206.0
1AJ	"	336.3	298.9	297.1	213.4	139.8	132.2	520.7	317.1
1BA	"	258.0	44.0	---	---	---	---	---	---
1CD	"	194.2	116.8	---	---	---	---	---	---
1CJ	"	628.3	391.1	482.3	320.3	331.7	211.4	791.7	361.5
1CK	"	271.8	260.7	599.9	427.1	224.6	204.3	1084.1	598.6
1DL	"	74.3	308.0	46.6	59.5	71.3	68.8	102.9	52.7
1DS	"	39.9	79.9	38.7	43.5	57.2	69.1	595.0	277.0
1IO	"	199.0	381.0	---	---	---	---	---	---
1MA	"	25.0	36.0	12.3	35.7	---	42.0	---	48.0
1MT	"	70.3	57.0	75.4	93.4	91.6	156.8	145.0	43.2
1RI	"	306.5	395.9	310.0	312.0	311.9	217.2	809.3	509.2
1RO	"	---	319.2	---	150.5	---	37.8	---	126.1
1SJ	"	---	70.5	6.3	22.3	3.3	25.3	16.0	33.0

Table 5.5-5 Ch3 Average Wait and Hold Time per PP Routine

	Run 6		Run 7	Run 8		Run 9
	wait	hold		wait	hold	
Total ms.	20.6	122.8	---	2.1	96.8	---
PRU "	16.0	118.5	---	2.2	93.5	---
CIO "	6.9	102.9	---	---	91.1	---
CMS "	125.4	253.1	---		---	---
CPM "	45.8	223.4	---	.2	135.6	---
DOO "	19.3	130.0	---		---	---
LDR "	42.6	232.7	---		---	---
QFM "	84.5	199.1	---	18.4	136.4	---
1AJ "	63.1	185.6	---	53.6	88.5	---
1BA "	665.5	360.0	---		---	---
1CJ "	186.8	229.3	---	1.7	134.1	---
1CK "	16.7	114.6	---	---	86.0	---
1DL "	20.7	143.6	---		---	---
1DS "	19.8	107.0	---		---	---
1MA "	.5	74.5	---	---	100.0	---
1MT "	63.5	84.0	---		---	---
1RI "	295.8	233.2	---	---	64.7	---
1RO "	---	316.0	---	---	197.1	---
1SJ "	88.0	75.5	---		---	---

Table 5.5-6 Ch4 Average Wait and Hold Time per PP Routine

		Run 6		Run 7		Run 8		Run 9	
		wait	hold	wait	hold	wait	hold	wait	hold
Total	ms.	---	14.5	.6	28.4	.3	27.7	---	9.6
PRU	"	---	14.5	.2	27.1	.1	26.3	---	9.6
CIO	"	---	---	.1	26.9	.0	26.2	---	---
CMS	"	---	14.5	12.5	68.4	3.1	52.1	---	9.6
DOO	"	---	---	---	25.0	---	13.5	---	---
QFM	"	---	---	2.7	34.8	1.4	28.8	---	---
1AJ	"	---	---	6.1	22.6	5.8	20.0	---	---
1CJ	"	---	---	20.1	81.2	8.5	64.8	---	---
1CK	"	---	---	6.1	42.2	4.1	33.2	---	---
1RI	"	---	---	---	85.0	---	68.8	---	---
1RO	"	---	---	---	89.0	---	119.0	---	---

Table 5.5-7 Ch10 Average Wait and Hold Time per PP Routine

		Run 6		Run 7		Run 8		Run 9	
		wait	hold	wait	hold	wait	hold	wait	hold
Total	ms.	---	47094	---	49557	---	83183	---	44914
DIS	"	---	2116	---	1164	---	1712	---	1572
DSD	"	---	69582	---	81818	--	123924	---	66584

Table 5.5-8 Ch12 Average Wait and Hold Time per PP Routine

	Run 6		Run 7		Run 8		Run 9	
	wait	hold	wait	hold	wait	hold	wait	hold
Total ms.	6.1	518.4	---	6.6	---	6.2	---	7.2
PRU "	9.0	7.5	---	---	---	---	---	---
ICD "	7.1	5827	---	---	---	---	---	---
IIO "	6.0	4.8	---	6.6	---	6.2	---	7.2

Table 5.5-9 Ch13 Average Wait and Hold Time per PP Routine

	Run 6		Run 7		Run 8		Run 9	
	wait	hold	wait	hold	wait	hold	wait	hold
Total ms.	.66	18.3	.6	24.8	1.1	23.9	.2	21.4
PRU "	---	199.3	---	74.7	---	63.8	---	166.1
IMT "	.66	18.3	.6	24.8	1.1	23.9	.2	21.4

Table 5.5-10 Ch14 Average Wait and Hold Time per PP Routine

	Run 6		Run 7		Run 8		Run 9	
	wait	hold	wait	hold	wait	hold	wait	hold
Total ms.	.0	3.1	---	3.0	---	3.0	---	3.1
PRU "	---	3.0	---	2.6	---	2.6	---	3.0
CIO "	.0	3.4	---	3.3	---	3.2	---	3.3
LFM "	.1	3.2	---	3.3	---	3.2	---	3.3
1AJ "	---	2.3	---	3.5	---	3.0	---	3.0
1DS "	---	3.8	---	3.3	---	3.3	---	3.4
1RI "	---	.8	---	.5	---	.7	---	.7

Table 5.5-11 Ch15 Average Wait and Hold Time per PP Routine

	Run 6		Run 7		Run 8		Run 9	
	wait	hold	wait	hold	wait	hold	wait	hold
Total ms.	---	3.2	---	3.0	---	3.0	---	3.1
PRU "	---	2.8	---	2.5	---	2.7	---	2.8
1CD "	---	1.0	---	---	---	---	---	---
1CK "	---	3.4	---	3.3	---	2.9	---	3.2
1DS "	---	2.3	---	2.2	---	2.6	---	2.5
1SJ "	---	3.3	---	3.1	---	3.1	---	3.2

5.6 PRU Statistics: Histograms 163-173

Table 5.6-1 shows, that about 90% of PRUs transferred and 90% of the number of transfers were for BCOT. The number of PRUs transferred by CIO calls represent only 46%, however the CIO calls represent 90% of all calls. Table 5.6-2 shows that 55% of the total transfers were for 2-4 PRUs only. Generally (over 50%) an I/O request has been issued for transferring 2-3 PRUs.

The PP routine names shown in table 5.6-3 transferred PRUs at least once in the specific run, but not necessarily in every call. For example in Run 7 1MA (histogram 13) did about 10 transfers over the total of 162 calls.

In table 5.6-3 1RO does not show, not because 1RO did not transfer PRUs, but because when the information was collected the number of PRUs, with several other items, were not available. This is due to the fact that 1RO does not drop the PP, but overlays itself with other routines, which then drop (see section 3.2.2.2.1).

The transfer time versus the number of PRUs transferred is shown in hist.# 163-166. Histogram # 163 shows the total time per each type of transfer; i.e., total time that has been used to transfer 2 PRUs, total time that has been used to transfer 3 PRUs, etc. Histogram # 166 shows that no missed revolutions have been encounterd. In this graph only averages are shown, however all transfers have been scanned one by one and only 17

missed revolution cases were found. Table 5.6-4 shows a list of missed revolutions for each run.

These missed revolutions might also be some minor hardware problems, i.e., the hardware mechanism which provides the arm movement, provides also a security check, to see if the latest move has reached the right cylinder; if not, it returns the arm to the cylinder 0 (zero) and reissue the seek again, thus this might cause the missed revolution syndrome.

Table 5.6-1 PRU Statistics

	Run 6	Run 7	Run 8	Run 9
Total PRUs transferred	69,874	69,302	70,000	68,837
transf. by BCOT	88.53%	91.74%	90.90%	91.58%
transf. by CIO calls	47.41%	45.77%	46.39%	45.39%
Total transf. occur.	5,023	5,627	5,757	5,125
occur. for BCOT	87.64%	91.74%	91.02%	90.60%
occur. for CIO calls	89.01%	89.96%	90.20%	88.80%

Table 5.6-2 PRU Distribution

	Run 6	Run 7	Run 8	Run 9
Total avg.	13.91	12.32	12.16	13.43
peak	(2-4) 52%	(2-4) 55%	(2-4) 55%	(2-4) 55%
Average SYOT	12.91	12.32	12.32	12.03
peak "	(12-17) 51%	(12-17) 39%	(12-17) 45%	(12-17) 41%
Average BCOT	14.05	12.32	12.14	13.58
peak "	(2-4) 57%	(2-4) 57%	(2-4) 59%	(2-4) 56%
Average Ch2	13.53	27.64	22.73	13.43
peak "	(2-4) 53%	(2-4) 27%	(2-5) 46%	(2-4) 55%
CIO "	7.24	8.91	8.35	6.87
peak "	(2-4) 60%	(2-4) 46%	(2-5) 59%	(2-4) 61%
Average Ch3	14.48	---	6.45	---
peak "	(2-4) 49%	---	(3-5) 62%	---
CIO "	7.66	---	5.61	---
peak "	(2-4) 56%	---	(3-5) 66%	---
Average Ch4	---	5.55	5.28	---
peak "	---	(2-4) 65%	(2-4) 70%	---
CIO "	---	5.46	5.15	---
peak "	---	(2-4) 66%	(2-4) 71%	---

Table 5.6-3 List of PP Routines which Transferred PRUs

Run 6	Run 7	Run 8	Run 9
CIO	CIO	CIO	CIO
CMS	CMS	CMS	CMS
CPM	CPM	CPM	CPM
DOO	DOO	DOO	DOO
LDR	LDR	LDR	LDR
LFM	LFM	LFM	LFM
---	---	---	PFM
QFM	QFM	QFM	QFM
1AJ	1AJ	1AJ	1AJ
1BA	---	---	---
1CK	1CK	1CK	1CK
1DL	1DL	1DL	1DL
1DS	1DS	1DS	1DS
1IO	---	---	---
1MA	1MA	1MA	1MA
1MT	---	---	---
1RI	1RI	1RI	1RI

Table 5.6-4 Missed Revolutions

Run 6		Run 7		Run 8		Run 9	
pru	diff.	pru	diff.	pru	diff.	pru	diff.
16	8.73	1	4.60	1	4.60	1	12.60
19	3.55	2	9.21	2	1.21	2	15.21
19	11.55	2	21.21	2	2.21	5	1.04
20	6.16	3	9.82	3	5.82	5	6.04
20	-16.16	4	1.43	4	1.43	17	1.34
		4	3.43	6	3.65	17	2.34
		5	3.04	7	6.25	19	15.55
		5	11.04	9	3.47	20	8.16
		5	37.04	14	16.51	20	14.16
		6	-1.65	18	7.95		
		7	3.25	19	3.55		
		7	5.25	36	20.90		
		10	1.08				
		17	6.34				
		19	4.55				
		19	4.55				
		20	15.16				

The time (diff.) shown, is the difference from the real transfer time and the time assuming it has been done in one revolution.

5.7 Disk Statistics: Histograms 174-191

From the problem explained in section 5.5, it has been necessary to distinguish whether a DSWM was for a wait for unit or for a seek wait. In table 5.7-1 the word "true" is for this distinction, true number of DSWM for a seek wait. It can be noticed that when there are more channels than disk units the true DSWM drop drastically.

The seek time shown in table 5.7-2 (Runs 7 and 8) seems quite small (about 10 msec.), but looking at table 5.7-3 it can be seen why. The arm movement has been on average 8.2 cylinders and 31% of the time the arm did not move.

Table 5.7-1 DSWM Statistics

		Run 6	Run 7	Run 8	Run 9
Total seeks issued(*)	Ch2	3,429	1,840	2,342	5,562
Total DSWMs issued	"	91,438	19,733	32,199	55,483
Avg. DSWM per seek	"	26.67	10.72	13.75	9.98
Percent true DSWMs	"	41.14%	95.66%	61.75%	92.92%
Max. DSWM per seek	"	380	22	211	25
true max.	"	38	22	21	24
Total seeks issued(*)	Ch3	2,421	---	538	---
Total DSWMs issued	"	90,170	---	28,112	---
Avg. DSWM per seek	"	37.24	---	52.25	---
Percent true DSWMs	"	29.78%	---	18.74%	---
Max. DSWMs per seek	"	509	---	521	---
true max.	"	33	---	19	---
Total seeks issued(*)	Ch4	---	3,246	2,296	---
Total DSWMs issued	"	---	25,567	21,500	---
Avg. DSWM per seek	"	---	7.88	9.36	---
Percent true DSWMs	"	---	97.04%	75.64%	---
Max. DSWMs per seek	"	---	14	236	---
true max.	"	---	14	13	---

(*) This counts only the first DSWM function issued, since a DSWM is issued only for a seek (besides when it is issued for unit busy, but this is not taken in account here).

Table 5.7-2 Seek Time Distribution

	Run 6	Run 7	Run 8	Run 9
Tot.seeks (incl. 0 seek)	6,653	7,220	7,319	6,695
Unit 0	6,649	3,024	3,454	6,690
Unit 1	4	4	4	5
Unit 2	---	4,192	3,861	---
Total seek time msec.	93,142	58,169	57,841	76,023
Unit 0 "	93,142	26,471	30,192	76,023
Unit 2 "	---	31,698	27,649	---
Percent of zero-seek	14%	31%	31%	19%
Unit 0	14%	41%	36%	19%
Unit 1	100%	100%	100%	100%
Unit 2	---	24%	27%	---
Avg. seek time if occurred				
Unit 0	16.09	14.74	13.57	14.07
Unit 2	---	9.95	9.73	---
Avg. seek time overall	14.00	8.06	7.90	11.36
Unit 0	14.01	8.75	8.74	11.36
Unit 2	---	7.56	7.16	---

Table 5.7-3 Cylinder Distribution

		Run 6	Run 7	Run 8	Run 9
Percent of accessed	U0	60%	47%	55%	61%
cylinders	"	80-119	50-64	50-64	60-79
most used & occur.	"	17 395	30 356	62 382	62 397
Avg. travelled length	"	23.40	12.28	11.60	13.77
zero trav. length	"	14%	41%	36%	19%
max. trav. length	"	115	63	63	84
Percent of accessed	U1	100%	100%	100%	100%
cylinders	"	0-4	0-4	0-4	0-4
most used & occur.	"	0 4	0 4	0 4	0 5
Avg. travelled length	"	0	0	0	0
zero trav. length	"	100%	100%	100%	100%
max. trav. length	"	0	0	0	0
Percent of accessed	U2	---	99%	99%	---
cylinders	"	---	5-29	5-24	---
most used & occur.	"	---	7 471	8 637	---
Avg. travelled length	"	---	5.26	4.03	---
zero trav. length	"	---	24%	27%	---
max. trav. length	"	---	27	19	---

Table 5.7-4 Device Wait Time

	Run 6	Run 7	Run 8	Run 9
Avg. wait time	111.15	22.78	28.41	80.73
Unit 0	111.22	46.64	48.04	80.79
Unit 2	---	6.31	10.88	---
Avg. wait if occur.	146.90	102.43	86.54	137.64
peak	15% 200-400	24% 15-24	40% 5-29	11% 200-300
Unit 0	146.90	161.25	127.06	137.64
peak	15% 200-400	8% 20-24	9% 200-300	11% 200-300
Unit 2	---	35.39	38.30	---
peak	---	29% 15-24	52% 5-24	---
Percent of waits	75.67%	22.24%	32.83%	58.66%
Unit 0	75.71%	28.31%	37.81%	58.70%
Unit 2	---	17.89%	28.41%	---

Table 5.7-5 Data Transfer Time

		Run 6	Run 7	Run 8	Run 9
Total, avg. time	msec.	24.82	22.06	21.47	22.92
max. observed	"	---	165	164	166
peak interval	"	5-19	5-19	5-19	5-19
peak percentage		61%	65%	66%	64%
Avg. Unit 0	msec.	24.83	30.79	28.31	22.93
max. observed	"	---	165	163	166
peak interval	"	5-19	5-19	5-19	5-19
peak percentage		61%	54%	58%	64%
Avg. Unit 1	msec.	14.25	8.25	10.50	9.40
max. observed	"	19	15	17	17
peak interval	"	10-19	1-19	1-19	1-19
peak percentage		100%	100%	100%	100%
Avg. Unit 2	msec.	---	15.77	15.36	---
max. observed	"	---	165	164	---
peak interval	"	---	5-19	5-19	---
peak percentage		---	74%	74%	---

Table 5.7-6 shows two different Ch hold times, they are (as explained in section 5.5): the first hold time is the difference between the Ch assign and Ch drop time. The second should be the true hold time plus the PP overhead, which is found by subtracting the total disk time from the PP hold time.

It is known that a Ch is up for at least the time to issue the seek function plus the data transfer time and the latency time. The time to issue the seek function is necessarily negligible. Therefore the Ch hold time and the PP overhead minus the transfer time should give us the PP overhead. For Runs 7 and 8 it seems very reasonable, but it is not for Runs 6 and 9. This large PP overhead is due to the Ch availability (Runs 6 and 9 do have very long wait times). In CIO (overlay 2CB) when the request for the Ch is made, if no channel is found free, the PP will pause a little while (pause for relocation), then will re-issue the function until a channel is found free. Thus the more the PP waits to get the channel, the more time is spent in the PP.

Table 5.7-6 shows a quite small average latency time (about 5 msec. instead of 8 msec.), but we have to consider that the system will allocate sectors starting from the first empty sector in the track, so it has not to wait for sector zero to start a read/write operation as in an IBM system.

Table 5.7-6

Overall Statistics for PP Routines Involving PRU Transfer

	Run 6	Run 7	Run 8	Run 9
Avg. PP hold time ms.	191.06	77.08	79.88	157.60
" CPU:I/O overlap "	19.79	9.51	9.47	17.93
" Ch hold time "	105.03	68.29	53.18	75.26
" Ch wait time "	35.68	34.24	25.61	77.67
" PRU transfered	13.91	12.32	12.16	13.43
" arm movement (cyl.)	23.40	8.20	7.60	13.77
Avg. total disk time ms.	153.97	52.90	57.78	115.01
wait time "	115.15	22.78	28.41	80.73
seek time "	14.00	8.06	7.90	11.36
transfer time "	24.82	22.06	21.47	22.92
avg. latency(*) "	6.05	5.51	5.15	4.82
data transfer(*) "	18.77	16.55	16.32	18.10
Avg. Ch hold plus PP O/H	37.09	24.18	22.10	40.98
" estimated PP O/H ms.	12.27	2.12	.63	18.06

(*) The data transfer time is computed assuming a time of 0.7 msec. per single PRU:

rotation time = 16.7 msec.,
24 sectors per track --> 0.7 msec. per sector,
2:1 interlace -> 2 sectors takes 2.1 msec.,
(two sectors plus one missed sector)
thus to tranfer n sectors -> (n * 2 - 1) * 0.7 msec.

Table 5.7-7 Overall Time for a CIO Call

			Run 6	Run 7	Run 8	Run 9
Avg.	PP hold time	ms.	140.84	46.73	49.05	114.46
"	CPU:I/O overlap	"	19.48	9.30	9.26	17.58
"	Ch hold time	"	83.99	31.11	38.24	43.68
"	Ch wait time	"	8.15	4.66	6.49	11.98
"	Ch2 hold time	"	71.58	44.85	44.73	43.68
"	Ch2 wait "	"	8.95	19.61	19.49	11.98
"	Ch3 hold "	"	102.85	---	91.05	---
"	Ch3 wait "	"	6.92	---	---	---
"	Ch4 hold "	"	---	26.87	26.19	---
"	Ch4 wait "	"	---	.12	.02	---
"	PRU transfered		7.41	6.27	6.26	6.87
"	PRU transfered	Ch2	7.24	8.91	8.35	6.87
"	" "	Ch3	7.66	---	5.61	---
"	" "	Ch4	---	5.46	5.15	---

The balance of utilization, as explained in section 1.1.3, is used to determine if there are bottlenecks in the system. The lower the value, the the longer the wait time (i.e., the longer is the queue for that device). In table 5.7-8 (Run 7) Ch2 and Unit0 have a compatible balance of utilization, so it is for Ch4 and Unit2; this should not be otherwise since they are connected together. Ch4 however had twice as many occurrences but the average transfer was five times smaller (see table 5.6-2), thus a smaller hold time and almost no wait time gives a very high balance of utilization, the best ratio. Runs 6 and 9 have very low balances of utilization.

Table 5.7-9 is another way to check whether disk units are a bottleneck. The busy time is computed summing the data transfer time, the seek time and the latency time divided by the elapsed real time. A disk unit is considered by some performance experts to be saturated when the busy time goes over 40%. Obviously in Runs 6 and 9 disk units were a bottleneck.

Table 5.7-8 Balance of Utilization for Ch and Disks

	Run 6	Run 7	Run 8	Run 9
Channel 2	.63	.50	.54	.46
Channel 3	.86	---	.98	---
Channel 4	---	.98	.99	---
Disk Unit 0	.26	.46	.44	.30
Disk Unit 2	---	.79	.67	---

Table 5.7-9 Busy Time for Disk Units

	Run 6	Run 7	Run 8	Run 9
Disk Unit 0	91.38%	48.25%	51.28%	85.15%
Disk Unit 2	---	39.48%	34.84%	---

5.7.1 Seek Overlap

The four experiments do not include a configuration with two disks connected to one channel. Thus, it is hard to show if there was any seek overlap. However, looking to the code there is a potential for seek overlap (see section 2.3.7). Also, it has been found possible to issue more than one seek (software-wise) for the same unit (see section 5.5).

This means that the channel is released after the seek is issued, thus another PP can request it and issue a seek to another free unit, then this PP will release the channel, so

another PP could request it, etc.

If it is possible to issue more than one seek, with one channel, to the same unit; it is necessarily possible to issue more then one seek, with one channel, to more than one unit.

To find if the seek overlap would have been useful, a program has been written to break down the disk utilization (Run 7) into three phases:

1- both units seeking,

2- both units transferring,

3- one unit seeking and one unit transferring.

This program counted instances where the two units were in one of the three phases. The results of the analysis have given the following:

1- seek/seek = 5.99%,

2- trnsf/trnsf = 12.56%,

3- seek/trnsf = 6.79%.

Since phase 1 and 3 are significant (12.78%), then DSWM is useful, i.e., both units were active at the same time and at least one was seeking, thus if DSWM was not implemented this overlapping could have not been possible. Phase two is also significant (12.56%), then two channels were necessary, otherwise a degradation of the service would have been expected. In total both units have been working together 25.34% of the occurences.

5.8 Conclusions

It is quite difficult to compare our results against the limited number of studies available about this subject; also there is a dissimilarity of system and machines.

From Hawthorn's (HAWTP 74) thesis, the CIO times for Ch4 and Ch5 could be compared with the results reported here, since they are connected to the 844 disks as in our study. However she has 8 disk units against our mere two (Run 7) but at a first glance, both figures seem quite competitive. She shows an average hold time of 30 msec. against 31.11 msec. and an average wait time of 25.6 msec. against 4.66 msec.; she does have necessarily a longer waiting queue for the disk or longer transfers. The balance of utilization for the two channels were of .73 and .68 compared to .50 and .98 in our study; they are not too close but the scratch file and system were on different packs in our study. Our configuration matches however Schwetman's (SCHWH 70) configuration, and his figures (.55 and .83) do comply with ours (.50 and .98) very well. Hawthorn's hypothesis about missed revolutions has been examined and found wrong. For all runs, only a few cases were found (see table 5.6-4), and these might well be due to data collection errors.

In Lynch's (LYNCW 72) study, comparison could be done for seek times, but here too, it is somehow difficult. He uses an IBM machine and we used a CDC machine, two complete different

architectures. On an IBM system you choose where to store your files; on CDC the system will choose for you, which I believe is a wise decision for non-production files; however a point may be made. We do have only an average of 31% (hist.197) where the arm did not move, against 63% in Lynch's study. The rest cannot be compared since he uses IBM 2314 drives, which are much slower than ours.

In Peltz's (PELTV 75) study, it is difficult to match figures since our scratch pack is his scratch, user and spool together. Overall figures seem to be compatible; however we seem to have smaller arm movement and he does have larger percentage for zero seeks (very close to Lynch's figures).

Summarizing, we found the "seek overlap" in the NOS operating system and we also know that there is no "latency overlap" as in the IBM systems. However, since the latency is much smaller we could not achieve much in that small time interval. Also it has been proven that there were no real missed revolutions in the NOS operating system.

Queue length for I/O could not be collected, however the I/O operations account for 76% (Run 7) of all channel activity; 1 PP was used 24%, 2 PPs 16%, and 3 PPs 10% of the time; thus queueing to disk were quite small. I believe that no other queueing algorithm would result in substantial improvement to our results.

5.9 Further Research

The next logical step for this study would be to create a model for the NOS operating system, and modify the data reduction program to prepare the data for the simulation model. Thus, the results of this study will be used as a validation for the modelled one.

The simulation model should not be too difficult to build, since this study already describes the system in quite enough detail to support a programmer to write a simulation program. The changes to the data reduction program, to modify the data to be acceptable to the simulation program, could be easily done since the data reduction program scans and analyzes the data one word at a time field by field, so, it would be a simple matter to insert some lines of code where the information is pertinent.

PART I. REFERENCES

ABATJ 69 PER TK7882 C5I2

Abate, J., and Dubner, H.; "Optimizing the Performance of a Drum-Like Storage" IEEE Trans on Computers, vol.C-18, no.11; (Nov.1969) pp.992-997.

The purpose of this paper is to analyze a particular strategy which attempts to optimize the performance of a drum-like memory. The present analysis is directed toward a system design for a head-per-track type storage which attempts to always execute first that request for a data transfer which will incur the shortest latency time relative to all the other possible waiting transactions.

ANDEJ 76

Anderson, J.W., and Browne, J.C.; "Graph Model of Computer Systems: Application to Performance Evaluation of an Operating System" Proceeding of the International Symposium on Computer Performance Modeling, Measurement and Evaluation; ACM SIGSIM; (March 1976) pp.166-178.

This paper defines and determines a graph model of a computer system in a form applicable to system performance analysis. This technique is demonstrated by application to modeling the UT-2 operating system for the CDC 6600. A generally applicable technique for extracting the graph representation of processes from event trace data is described. The technique is both complete and general and may be profitably applied for either partial or complete models of any type of complex computer system process.

ATWOJ 72

Atwood, J.W., and Grushcow, M.S.; "A Hierarchical Input/Output Supervisor for the IBM System/360" CIPS Session 72, Canadian Information Processing Society; (June 1972). pp.313401-313412.

This paper describes the I/O supervisor for project SUE, consisting of five major components: the kernel I/O manager, the channel and the control unit managers and the device allocator. Communication between these modules follows a strict hierarchy defined by the configuration of the I/O hardware connected to the central processor. The organization proposed here is easier to understand than the maze of interconnected routines that make up the present OS I/O supervisor.

ATWOJ 73 PER QA76 A1C6

Atwood, J.W.; "I/O Supervision in the Project SUE Operating System" Computer, vol.6; (Nov.1973) pp.19-23.

This paper describes a part of the project SUE: the I/O supervisor, for the IBM System/360. The design and construction of the project SUE I/O supervisor demonstrates the feasibility of a hierarchical approach to I/O supervision. The suggested improvements attempt to define an I/O architecture which is more convenient to use than presently utilized on the System/360.

ATWOJ 75

Atwood, J.W.; "Machine Architecture for Efficient Input/Output" Infotech State of the Art Report 22 on Input/Output, Infotech Information Ltd, England; (1975) pp.283-297.

This paper reviews current techniques in I/O supervision and then explores possible avenues for improvement. A short description of the IBM System/360/370 and CDC 6000 series is included. After discussing a hierarchical I/O supervisor that attempts to overcome some of the existing difficulties, the changes that are likely to come in the future are discussed.

ATWOJ 76 PER QA76 A1C6

Atwood, J.W.; "Concurency in Operating Systems" Computer, vol.9; (Oct.1976) pp.18-26.

This paper examines the idea of concurency as a mechanism for sharing. Two systems (CDC Cyber-70 and 6000 series, and IBM 360/370 series) are introduced to show the implementation of concurency through the use of ad hoc rules. Some recent proposals for language constructs which simplify and control the management of concurency are also introduced.

ATWOJ 79

Atwood, J.W., and LoCicero C.; "An Analysis of the CDC Network Operating System Using an Event-trace Monitor" Proceedings of 1979 CIPS Conference, Quebec; (June 1979) pp.108-113.

This paper reports some preliminary results of this thesis.

AUERP 74

Auerbach Publishers Inc.; "Computer System Performance Measurement", Data Processing Manual, Operations, 5-03-03; (1974) pp.1-15.

Too often the DP manager listens to a hardware salesman when confronted with a loaded machine schedule. Ordering additional equipment can be the line of least resistance. This paper explains what computer performance is, how computer performance is measured, the benefits of computer performance measurement, how to collect the data, and so on. It is a basic reading for someone who is asking: what is performance measurement for?.

AUERP 75

Auerbach Publishers Inc.; "Computer Performance Measurement Study" Data Processing Management, Case Studies, 7-08-03; (1975) pp.1-18.

This case study describes an actual computer performance measurement, tells why it was done, how it was planned, and how the data was collected; it gives analysis of the data, shows what actions were taken, and highlights the lessons to be learned. It is not necessarily typical of computer performance measurement activities but represents what was done in an actual case. The system measured was running with two large scale IBM 360/65s.

BAUEM 74

Bauer, M.J.; "A Simulation Approch to the Design of Dynamic Feedback Scheduling Algorithms for Time-Shared Computer Systems" ACM SIGSIM II Symposium; (June 1974) pp.165-173.

This paper describes the design of a dynamic feedback scheduling algorithm and the simulation in which it is being implemented, and presents some preliminary results. While the complete algorithm is not yet implemented within the simulation, preliminary results suggest that some improvement in response time may be possible. The computer system used is a DEC PDP-10, and the simulation is written in POOMAS.

BEDOR 74a

Bedoll, R.F.; "Performance Measurement on KRONOS" VIM-21 Proceedings; (Oct.1974) pp.225-229.

Like most other sites, BCS undertook performance measurement to gain a better understanding of the resources the computer system was using, and to allow intelligent decision regarding those resources. Their efforts were divided into two major areas: Low-level measurement such as channel utilization, PP usage, etc.; and High-level measurement which involves system job mix, job swapping, and scheduling, etc.

BERET 73

Beretvas, T.; "System Independent Tracing for Prediction of System Performance" Symposium ACM SIGSIM; (June 1973) pp.209-213.

The principles of a system-independent tracing facility are presented. A limited application of these principles is illustrated in the use of an existings tool, GTF. Shortcomings of the GTF-BASED trace are presented, and extensions for the purposes of virtual system modeling are discussed. The applicability of the Trace-Driven simulation technology for validation of an operating system (IBM 360) model is demonstrated.

BOYSJ 75 PER QA76.5 C617

Boyse, J.W., and Warn, D.R.; "A Straghtforward Model for Computer Performance Prediction" Computing Surveys, vol.7, no.2; (June 1975) pp.73-93.

This tutorial paper leads the reader through the development and use of an easily understood analytic model. They have introduced a queuing model and shown its application to computer performance prediction by means of a case study. A number of assumptions were made to simplify the analysis of the model. A very good paper for someone who wants to know how a model is built up and how to implement it.

BRICR 78 PER QA76 A772

Brice, R.S., and Browne, J.C.; "Feedback Coupled Resource Allocation Policies in the Multiprogramming-Multiprocessor Computer System" Comm ACM, vol.21, no.8; (Aug.1978) pp.678-686.

This study used a CDC 6600/6400 configuration driven by the UT-2D operating system. This paper presents model studies of some integrated, feed back-driven scheduling systems for multiprogrammed-multiprocessor computer systems. The basic control variables used are the data-flow rates for the processes executing on the CPU. Attention is given to the

amount of memory resource required for effective processing of the I/O activity (buffer space assignment). The model studies used both distribution-driven and trace-driven techniques. Even relatively simple dynamic schedulers are shown to improve system performance. The improvement is greatest under a heavy I/O demand workload.

BROWJ 72 PER TK7885 A1J6

Browne, J.C., Lan, J., and Baskett, F.; "The Interaction of Multiprogramming Job Scheduling and CPU Scheduling" Proc AFIPS FJCC, vol.41, pt.1; (1972) pp.13-22.

A system simulation of a CDC 6600 system is employed to assess the dependency of overall system performance on the effect of both job scheduling and CPU scheduling algorithms. This simulation study demonstrates some interesting interdependencies.

CALIP 67 PER QA76 A772

Calingaert, P.; "System Performance Evaluation: Survey and Appraisal" Comm ACM, vol.10, no.1; (Jan.1967) pp.12-18.

Measures of performance, such as throughput, turnaround, and availability are defined. The use of instruction mixes and kernels is presented. The role of analysis, simulation, and synthesis are discussed.

CAMPD 68 PER TK7885 A1J6

Campbell, D.J., and Heffner, W.J.; "Measurement and Analysis of Large Operating Systems During Sysyem Development" Proc AFIPS FJCC, vol.32; (1968) pp.903-914.

This paper describes a large number of measurement techniques that the authors have employed in developing their operating system. The number and variety of means demonstrate the many different problems faced by the system developer. The paper describes additional measurement techniques (to GECOS II), limitation of each, values of each, and specific lessons learned by applying these techniques to GECOS III.

CANTH 68 PER TK7885 A1J6

Cantrell, H.N., and Ellison, A.L.; "Multiprogramming System Performance Measurement and Analysis" Proc AFIPS SJCC, vol.32; (1968) pp.213-221.

The analysis of the performance of a multiprogrammed computer system and the programs which operate in the system are discussed, with particular reference to such an analysis of the GE 625/635 GECOS II operating system and the system software and user programs which operate in that system; A good analysis is given here.

CARLG 77

Carlson, G.; "The Use of Statistics in Performance Measurement-- Part 1 and Part 2" EDP Performance Review, vol.5, no.8 and no.9; (Aug-Sept.1977) pp.1-7, 1-7.

These two papers summarize different techniques on how to convert raw data in some meaningful output. The first report covers descriptive statistics such as the mean, standard deviation, min and max, and correlation. The second covers analitical statistics such as the t-test, analysis of variance, multiple linear regression, and factor analysis. Hopefully, the use of these analytical tools can help bring "science" to performance measurement.

CHENP 69 PER TA168 I5

Cheng, P.S.; "Trace-Driven System Modeling" IBM System Journal, vol.8, no.4; (1969) pp.280-289.

A Trace-Driven approach to computer system modeling is described. The work load of interest is first executed on a real system and trace data obtained by monitoring significant events in the course of this execution. The resultant data used as input to the desired simulation model, which may reflect changes in the configuration or operating system of the real system.

COFFE 69 PER QA76 A77

Coffman, E.G.Jr.; "Analysis of a Drum Input/Output Queue Under Scheduled Operation in a Paged Computer System", Journal ACM, vol.16, no.1; (Jan.1969) pp.73-90.

In this paper, magnetic drums in the role of auxiliary memories are studied in a context of modern multiprogramming systems featuring a paged environment. The purpose has been to provide an exact analysis of a queueing system of topical interest where only techniques of approximation have been successful thus far. The basic aim with regard to the analysis is to extend the methodology so far developed for assessing the performance of I/O devices viewed as servers in stochastic queueing systems.

COFFE 72 PER QA76 S555

Coffman, E.G., Klimko, L.A., and Ryan, B.; "Analysis of Scanning Policies for Reducing Disk Seek Times" SIAM J. Comput., vol.1, no.3; (Sept.1972) pp.269-279.

A number of recent studies have examined techniques for sequencing disk accesses to minimize or reduce seek times. The principal methods proposed have been called scanning policies. In this paper they formulate and analyze simple mathematical models of head motion in disk systems in which two different scanning policies are implemented.

CONTD 74

Conti, D.M.; "Computer Performance Measurement -- Tools, Problems, and Myths" VIM-21 Proceedings; (Oct.1974) pp.201-210.

This paper presents a survey of known hardware and software tools, past and present, that have been used to measure the performance of CDC 6000 and lower CYBER series machines. Software monitoring tools that are and have been available for measuring these machines are discussed in some detail. The relative advantages and disadvantages of each are also presented.

DENNP 67 PER TK7885 A1J6

Denning, P.J.; "Effects of Scheduling on File Memory Operations" Proc AFIPS SJCC, vol.30; (1967) pp.9-21.

This paper discusses some important aspects of file system organization relevant to maximizing the throughput, the average number of requests serviced per unit time. It investigates the proposition that the utilization of secondary memory systems can be significantly increased by scheduling requests so as to minimize mechanical access time. The importance of the results obtained here is that they are relatively independent of the behavior of the processes generating requests.

ESTR 76

Estel, R.G.; "How Fast is REAL-TIME?" Performance Evaluation Review, ACM SIGMETRICS, vol.5, no.3; (July 1976) pp.16-18.

A single bench mark test was compiled and run on the AN/UYK-7 computer and on a number of commercial computers in order to measure the relative thruput of the UYK-7, which is

the Navy's large scale real-time computer. The results indicate the speeds and accuracies of each host; however, general conlusions can be drawn only with some risk.

FRANH 69 PER QA76 A77

Frank, H.; "Analysis and Optimization of Disk Storage Devices for Time-Sharing Systems" Journal ACM, vol.16, no.4; (Oct.1969) pp.602-620.

In this paper, the transfer characteristic of disk storage devices are considered. Expected seek time and expected rotational latency are taken as measure of performance for the disk. The following aspects of disk files and their behaviour are considered: the speed profile of the positioning mechanism and its effect on seek time, effect of the probability distribution of information stored on track, and some others.

FRANM 74 PER TK7882 C5I2

Franklin, M.A., and Amitava, S.; "An Analitic Response Time Model for Single- and Dual-density Disk Systems" IEEE Trans on Computers, vol.C-23, no.12; (Dec.1974) pp.1269-1276.

This paper considers an equipment replacement problem. The question to resolve centers around replacing the disk storage devices currently used with other disks having twice the storage capacity. Queueing theory is used to obtain curves of response time versus arrival rate, and the results are compared with corresponding curves obtained by a simulation model.

FRASC 73 QA76.9 V5A18 C.2

Frasson, C.; "Simulation of Input-Output Units" ACM SIGARCH-SIGOPS Workshop on Virtual Computer System; (March 1973) pp.236-246.

This paper deals with the I/O aspect only. In order to simulate an I/O unit, it proposes to define a comprehensive language, easy to use, and capable of describing operations of the device on line with any computer.

FULLS 72 PER TK7882 C5I2

Fuller, S.H.; "An Optimal Drum Sheduling Algorithm" IEEE Trans on Computers, vol.C-21, no.11; (Nov.1972) pp.1153-1165.

The problem considered here is to find an algorithm that schedules the processing of these records with the minimal total amount of rotational latency, taking into account the current position of the drum. This problem is a special case of the traveling salesman problem. The algorithm that is developed has the attractive property of exhibiting a computational complexity on the order of N log N.

FULLS 74 PER QA76 A772

Fuller, S.H.; "Minimal-Total-Processing-Time Drum and Disk Scheduling Disciplines" Comm ACM, vol.17, no.7; (July 1974) pp.376-381.

This article investigates the application of minimal-total-processing-time (MTPT) scheduling disciplines to rotating storage units when random arrival of requests is allowed. Fixed-head drum and moving-head disk storage units are considered, and emphasis is placed on the relative merits of the MTPT scheduling discipline with respect to the shortest-latency-time-first (SLTF) scheduling discipline. The results of the simulation studies presented show that for intra-cylinder disk scheduling the MTPT discipline offers a distinct advantage over the SLTF discipline.

GOLDD 74 PER QA76 A772

Gold, D.E., and Kuck, D.J.; "A Model for Masking Rotational Latency by Dynamic Disk Allocation" Comm ACM, vol.17, no.5; (May 1974) pp.278-288.

This paper is one of a pair of papers dealing with masking rotational latency. The authors have presented some motivation for their problem and their definitions of terms. They have also given an algorithm for the permutation of blocks in a memory hierarchy. Several variations of the algorithm were presented and these make the ideas more useful in real applications.

GOTLC 73a

Gotlieb, C.C., and Metzger, J.K.; "Trace-Driven Analysis of a Batch Processing System" Symposium ACM SIGSIM; (June 1973) pp.215-222.

A Trace-Driven model was used to study a simple processing system (IBM 360/165 OS) which receives batches of jobs from a number of remote terminals. The development of the model is discussed in some detail. The utility of Trace-Driven modeling as a system tuning tool has been demontrated on a non-trivial problem.

GOTLC 73b QA76 A77

Gotlieb, C.C., and Mac Ewen, G.H.; "Performance of Movable-Head Disk Storage Devices" Journal ACM, vol.20; (Oct.1973) pp.604-623.

A queueing model of movable-head disk storage systems is developed so that the performance, as measured by the mean response time, can be calculated. Queue scheduling algorithms which improve the performance are considered. This analysis is extended to multimodule systems whereby tables of approximate mean response values time can be calculated over system parameters describing equipment characteristics, equipment configuration, system loading, file organization, and algorithm (SCAN or FIFO). The use of such tables is discussed and the applicability of the analysis to a recently marketed disk is noted.

HARDY 74

Hardy; "Monitoring of the Performance of the CYBER/72 at CANFARM" Session 48B, VIM-20 Proceedings; (Apr.1974) pp.447-482.

This paper attempts to describe the experience of the CANFARM DATA SYSTEM in using a series of monitors designed to aid in the evaluation of machine performance. It also attempts the rather more difficult task of assessing the relative merits of using different kinds of monitors (hardware and software).

HAWTP 74

Hawthorn, P.B.; "A Performance Evaluation of a CDC 6600 Computer" M.Sc. Thesis, Computer Science, University of Houston, Houston, Texas; (Dec.1974) 144 pp.

This thesis presents a performance evaluation of a CDC 6600 machine, under the KRONOS 2.0 operating system. The evaluation was done by use of an event-driven probe imbedded in the operating system. The results of the probe were written on disk and the disk files later analyzed.

HEILS 79

Heils, S.W.; "One Approach to the Management of Computer Performance Data" EDP Performance Review, vol.7, no.1; (Jan.1979) pp.1-10.

Telecommunication and Information Processing Operation (TIPO) has built a computer performance management information system which provides for the collection, integration, and dissemination of computer performance data. This report will explore some of the logic behind the System Growth Planning (SGP) system, how the system works, and some of the things that TIPO has been able to accomplish through this approach. The system used for this study was an Honeywell H6080.

HELLH 70 PER QA76.5 C617

Hellerman, H., and Smith, H.J.,jr,; "Throughput Analysis of Some Idealized Input, Output, and Compute Overlap Configurations" Computing Surveys vol.2, no.2; (June 1970) pp.111-118.

This tutorial approach to multiprogramming operation develops a flavor for system operations by deriving performance equations for idealized cases of input, output, and compute intervals. The performance evaluation parameter used is the system throughput (the reciprocal of job time spent in the system to complete). One can gain a good understanding of the concept of multiprogramming by trying to work through a few of the cases studied.

HUTCG 65 PER QA76 A772

Hutchinson, G.K.; "A Computer Center Simulation Project" Comm ACM, vol.8, no.9; (Sept.1965) pp.559-568.

This paper describes a macroscopic, job-shop-like simulation of a computer center. The center operations are described and performance measures are defined. Simulation experiments in job scheduling are described and the results of these experiments are presented. No descriptions of the simulator are given.

INGWL 74

Ingwersen, L.D.; "SCOPE 3.4 Scheduler Tuning" VIM-21 Proceedings; (Oct.1974) pp.213-219.

This paper reports the status of the SCOPE 3.4 Schduler tuning experiment currently in progress at Control Data's Sunnyvale, California facility. The project is a continuing effort. The following topics are discussed in the paper: general tuning concepts; description of the scheduler tuning approach; description of the SCOPE 3.4 scheduler; tuning guidelines; CDC tuning experiments; and how to do your own tuning.

KASPH 74a TA 168 H37A 1974

Kaspar, H., and Miller, D.S.; "Simulation of the TRW Timesharing System" Proc of the Seventh Hawaii International Conference on System Sciences; (1974) pp.155-157.

A discrete event language simulation of the TRW timesharing system, TRW/TSS, is described. The operating system is functionnaly modeled to the basic task level. The simulation is coded in SALSIM, a TRW developed discrete event simulation language written in FORTRAN which permits list processing, queueing and automatic event timing. Calibration and validation procedures are discussed. Results to date are presented. The simulated computer is a CDC 6500 and 6400 together with the TRW/TSS operating system.

KASPH 74b

Kaspar, H., and Miller, D.S.; "Job and Jobstream Modeling in the TRW Timeshare System Simulation" ACM SIGSIM II Symposium; (june 1974) pp.95-121.

The paper begins with a brief description of the TRW/TSS operating system where the components consist of a CDC 6500 and a CDC Cyber 74. Simulation objectives are given. An overview of the simulation is presented. The actual TRW/TSS user jobstream is given in detail. This is followed by two methods of modeling this user profile. Sections on calibration and validation of these jobstream models and a presentation of results and output to date conclude the article.

KIMBS 72 PER TK7885 A1J6

Kimbleton, S.R.; "Performance Evaluation: A Structural Approach" Proc AFIPS SJCC, vol.40; (1972) pp.411-416.

This paper is concerned with evaluating the performance of a computer system, including its operating system. Although directed towards evaluation for purposes of selection and comparison, the ideas presented are relevent with regard to the improvement or evaluation of a particular programming or operating system for a particular machine. In particular, the point is made that usually a few key variables drive the entire system.

KRAUO 71 TA168 H37+ 1971

Kraulis, O.E.; "Computer Systems Performance Evaluation" 4th Hawaii International Conference on System Sciences; (1971)

pp.437-439.

In the last few years, a whole technology has evolved for evaluating the performance of computer systems. At the University of Guelph, they have developed a number of tools and techniques for use in the task of systems evaluation and they have purchased a number of others. This report presents some of the results they have obtained and outlines the scope of their ongoing research into this subject.

LAZOC 77

Lazos, C.; "Functional Distribution of the Workload of a Linked Computer System and its Simulation" Performance Evaluation Review, ACM SIGMETRICS, vol.6, no.3; (Summer 1977) pp.5-14.

Consideration is given to a possible functional distribution of workload over two linked computers with separate channel access to a large disc store, into the resource utilisation of the linked system archived by simulation using a modified and re-entrant single processor simulator. Results suggest that the proposed distribution realises a high utilisation.

LEROJ 76

Leroudier, J., and Parent, M.; "Discrete Event Simulation Modeling of Computer Systems for Performance Evaluation" Laboria Report no.177, IRIA Laboria, France; (June 1976) 70 pp.

This report presents some of the problems which may be encountered by anyone who has to write a descrete event simulation model. It also describes the techniques currently available to solve these problems. This survey should not be considered as an introduction to descrete event simulation, the paper is addressed to users already familiar with simulation and who are facing a particular problem.

LINDD 76

Lindsay, D.S.; "A Hardware Monitor Study of a CDC KRONOS System" Proceeding of the International Symposium on Computer Performance Modeling, Measurement and Evaluation; ACM SIGSIM; (March 1976) pp.136-144.

This study was performed on a CDC 6600 running a KRONOS operating system. The measurements were taken with three different tools-- Dynaprobe, a small software probe, and Rand RMS probe. The parameters measured were-- CPU utilization,

disk and tape I/O, ECS and disk Rollout, terminal response time, CM lockout, and Exchange-jumps. The measurement results, for each parameter, are discussed in some detail and suggestions for some improvements are given.

LOCIC 78

LoCicero, C.; "NOS System Call Analysis for SBL, SIMBDP, IBL and IBL2" CDC Company Private, CDD Performance Evaluation Group; (Apr.1978) 850 pp.

This study was performed on a CDC CYBER 173 running a NOS 1.1 operating system. The report documents the initial results of a study on locality. Three different types of workload have been used: commercial, scientific and interactive. It also outlines in detail the monitor activity: monitor call distribution, frequency of monitor calls, average time between CPU switches, CPU utilization, CPUMTR overhead, PRU transfer, average time between monitor calls, estimated number of instructions between monitor calls and between CPU switches.

LOCIC 79

LoCicero, C.; "CPU Measurment for a NOS System" CDC Company Private, CDD Performace Evaluation Group; (Jan.1979) 82 pp.

This study was performed on a CDC Cyber 172 running a NOS 1.1 operating system. This report documents the initial results of a study to obtain detailed measurements on CPU activity during execution of several standard workloads. The CPU time has been broken down to: user mode, system and sub-systems, where system and sub-systems reflect the system overhead. A very detailed documentation for where the CPU spend its time, when not in user mode, is given.

LUCAH 71 PER QA76.5 C617

Lucas, H.C.; "Performance Evaluation and Monitoring" Computing Surveys vol.3, no.3; (Sept.1971) pp.79-91.

This often referenced categorization of efforts in the area of computer performance evaluation and monitoring performs a much needed function. In an area where a new methodology is advancing, Lucas attempts to classify the goals and techniques applied. For example, he cites three general purposes of system evaluation: selection evaluation, performance projection, and performance monitoring. Then he tabulates the various techniques on their suitability in achieving the said goals. The techniques are also explained in brief. The approach taken to unify such a broad body of

knowledge is appropriate.

LYNCW 72

Lynch, W.C.; "Do Disk Arms Move?" Performance Evaluation Review, ACM SIGMETRICS, vol.1, no.4; (Dec.1972) pp.3-16.

This paper describes some disk system experiments carried out at a university. The data resulting from the experiments, a possible model explaining the results, and some implication of the model are explained. The measurement of the disk arm movements yielded the unexpected result that the arms do not move in 63 per cent of the accesses and do move for an average of only 30 ms. in the remaining 37 per cent of the cases. The measurements were done on a IBM 360/67 under MTS.

MACDM 67 PER TK7885 A1J6

MacDougall, M.H.; "Simulation of an ECS-Based Operating System" Proc AFIPS SJCC vol.30; (1967) pp.735-741.

This paper describes the simulation of a CDC 6000 system with Extended Core Storage. The simulator is described in some detail and some simulation results are presented.

MACDM 70 PER QA76.5 C617

MacDougall, M.H.; "Computer System Simulation: An Introduction" Computing Surveys, vol.2, no.3; (Sept.1970) pp.191-209.

The underlying aspects of constructing a simulation model are uncovered by approaching the study of a disk-based multiprogramming system with a higher level (as opposed to simulation) language. Maintenance of variable-length event lists, queues for shared resources, and statistics gathering are all explained. This paper is a great place to begin if you are interested in system simulation principles. In addition, a fully annotated bibliogrphy provides references for more in-depth research.

MACDM 74

MacDougall, M.H.; "Simulating the NASA Mass Data Storage Facility" ACM SIGSIM II Symposium; (June 1974) pp.33-43.

This paper begins with an overview of the structure and facilities of the model. The methods employed in the internal and external design of the model to meet the wide range of requirements for its use are described, and the problems of

developing large scale computer system simulation models are discussed. The NASA/HOUSTON Mass Storage Facility comprises two CDC CYBER 70 with ECS, with facilities for communication between the MDSF computers and the IBM 360/75 computers of the RTCC.

MINSN 72 PER TK7885 A1J6

Minsky, n.; "Rotating Storage Devices as Partially Associative Memories" Proc AFIPS FJCC, vol.41; (1972) pp.587-595.

This paper points to several latent potentialities of rotating memories, and describes a method for utilizing them for a realization of "partial associativity". The method described here is by no means the only possible way for realizing associativity on cylinder memories. This paper should be treated, therefore, as an illustration of what can be done rather than as a definite proposal.

NIELN 67 QA76 A8 1967

Nielsen, N.R.; "Computer Simulation of Computer System Performance" Proc 22nd ACM National Meeting; (1967) pp.581-590.

This paper describes various types of simulations such as: Simulation of hardware logic, Simulation of hardware system performance, Simulation of software and system performance and Simulation and the third generation. Cited: "It has already been demonstrated that system performance simulations can be used effectively for this purpose -- if appropriate models can be developed. Thus, the third generation offers an exciting challenge".

NOE,J 71

Noe, J.D.; "A Petri Net Model of the CDC 6400" ACM SIGOPS Workshop on System Performance Evaluation, Harvard University; (Apr.1971) pp.362-378.

The paper describes a multiprocessor, multiprogramming system, the CDC 6400, in terms of Petri Nets and shows how this type of representation lends itself to planning system measurements. Petri Nets have been found useful for representing parallel processes. They can show the resources, the job states, and the service queues within the system. They are compatible with decision tables, or flowcharts representing the transition, and the transitions are readily interpreted as subprograms in the operating system, or as simulation routines. The machine described is a CDC 6400

under the SCOPE 3.2 operating system.

NOE,J 72 PER TK7885 A1J6

Noe, J.D., and Nutt, G.J.; "Validation of a Trace-Driven CDC 6400 Simulation" Proc AFIPS SJCC, vol.40; (1972) pp.749-757.

This paper describes validation of a simulation model of a multiprogramming system (SCOPE 3.2). The general approach taken in this study was to develop a simulation at the level of detail that shows the interaction between tasks and system resources. Provisions are included in the simulation model to represent the input job load either as individual jobs or as jobs classes. This work was done under a number of constraints that limited some of the things that would have been done. In spite of those constraints, a considerable amount of information was available through the DAYFILE and it can be seen how much can be done without the freedom to alter the system programs for measurement purposes.

NUTTG 73

Nutt, G.J.; "The Computer System Representation Problem" Symposium ACM SIGSIM; (June 1973) pp.145-149.

The need for a standardized notation for representing a wide class of computer systems has been discussed in this paper. Some attempts at a solution to the problem are briefly surveyed. The view taken by this paper is that such a representation can be derived, (perhaps already exist), but that one of the more serious hurdles to be overcome is in obtaining general acceptance of the method.

PELTV 75

Peltz, V.T.; "An Empirical Study of Direct Access Storage Device Performance" M.Sc. Thesis, Dept. Elect. Eng., University of Toronto, Toronto, Canada; (1975) 103 pp.

This thesis has conducted an empirical study of three commonly used direct access storage device queueing algorithms to obtain profiles of disk arm movement. An actual working system has been measured to see if significant changes in performance take place as a result of varying the queueing techniques. Measurements have been taken during regular production hours and under controlled repeatable conditions. The untruthfullness of certain assumptions often made in theoretical analysis of disk activity is demonstrated.

PIEPW 75 PER QA76 A772

Piepmeier, W.F.; "Optimal Balancing of I/O Requests to Disk" Comm ACM, vol.18, no.9; (Sept.1975) pp.524-527.

This ACM student award paper, determines a policy for efficient allocation and utilisation of a set of disk drives with differing operational characteristics. Using standard queueing theory, it finds results indicating that faster devices should have higher utilisation factors and that the number of different device types utilized tend to decrease with decreasing load.

REDDS 76 PER QA76.5 C617

Reddi, S.S. And Feustel, E.A.; "A Conceptual Framework for Computer Architecture" Computing Surveys, vol.8, no.2; (June 1976) pp.277-300.

This paper describes the concepts, definitions, and ideas of computer architecture and suggest that architecture can be viewed as composed of three components: physical organization; control and flow of information; and representation, interpretation and transformation of information. Architecture design problems and trade-offs are discussed in terms of the proposed framework. Computers such as the CDC 6600, TI ASC, Borroughs B6700, ILLIAC IV, IBM System/370 are described.

SCHEA 67

Scherr, A.L.; "An Analysis of Time-Shared Computer Systems" Res. Monograph no.36, MIT Press, Cambridge, Mass.; (1967).

This is a study of the MIT Compatible Time Sharing System (CTSS). Statistics on CTSS job behaviour are presented and the simulation model described in detail. The simulator employed the CTSS scheduling algorithm, and MAD listings for this algorithm as well as the simulator itself are given in appendices. Markov models of the system are developed, and the system performance as predicted by the simulation model and Markovian model is compared with actual system performance.

SCHWH 70

Schwetman, H.D.; "A Study of Resource Utilization and Performance Evaluation of Large Scale Computer System" Ph.D. Dissertation, TSN-12, Computation Center, The University of Texas, Austin, Texas; (July 1970) 115 pp.

This study of the performance of a large-scale computer system (the CDC 6600 and the UT-1 operating system) is based on empirical data gathered while the system was in operation. Extensive data detailing both the efficiency and balance of resource utilization in a variety of operating environments are presented as a basis of a quantitative evaluation of system performance. The definitions and techniques developed show promise of being transferable to other computer systems.

SCHWH 74

Schwetman, H.D.; "Simulation of Computer System Using Automatically Generated Load Descriptions" Proc Winter Simulation Conference, vol.2; (1974) pp.699-704.

This paper discusses some techniques and procedures which have been used to automatically generate the required load descriptions. The emphasis is on a request sequence generator which uses data found in a system event trace. One such technique, called Trace-Driven modelling, has been implemented at the Purdue University Computing Center. This implementation is discussed and its usefulness and accuracy are illustrated with actual data. The computer simulated is a CDC 6500 under the MACE-5 operating system.

SCHWH 78 PER QA76.5 S653

Schwetman, H.D.; "Job Scheduling in Multiprogrammed Computer System" Software-- Practice and Experience, vol.8, no.3; (May-jn.1978) pp.241-255.

The implications of partitioning of the scheduling function are studied by use of a simulation model. A system embodying this approach to job scheduling is discussed as an application of the approach to other type of systems. Three different fitting algorithms are described, and their operation illustrated via the output of a simulation model. A CDC 6500 was used for the testings.

SCHWH 78a PER QA76 A772

Schwetman, H.D.; "Hybrid Simulation Models of Computer Systems" Comm ACM, vol.21, no.9; (Sept.1978) pp.718-723.

This paper describes the structure and operation of a hybrid simulation model in which both discrete-event simulation and analytic techniques are combined to produce efficient yet accurate system models. Discrete-event simulation is used to model the arrival and activation of jobs, and a central-server. The accuracy and efficiency of the hybrid technique are demonstrated by comparing the result

and computational costs of the hybrid model of the example with those of an equivalent simulation-only model. This study uses a CDC 6500 for validation of these models.

SEAMP 69 PER TA168 I5

Seaman, P.H., and Soucy, R.C.; "Simulating Operating Systems" IBM System Journal, vol.8, no.4; (1969) pp.264-279.

The computer system simulator (CSS), which provides a language and a structure specifically designed for modeling computer systems, is described in this paper. The CSS language is a higher level language in which the subject system can be described in terms of its configuration, operating system programs, and job environment.

SHEMJ 69 PER 7885 A1J6

Schemer, J.E., and Heying, D.W.; "Performance Modeling and Empirical Measurements in a System Designed for Batch and Time-Sharing Users" Proc AFIPS FJCC, vol.35; (1969) pp.17-26.

In this paper the authors modelled a system, designed for both batch and terminal users. They validate their model by probing the system to determine if the response function was as they had modeled it to be, so they monitored response time to on-line demands, and percentage of CPU time available for batch jobs.

SHERS 72a

Sherman, S.W.; "Trace-Driven Modeling Studies of the Performance of Computer System" Ph.D. Dissertation, TSN-30, Computation Center, the University of Texas, Austin, Texas; (Aug.1972) 188 pp.

This study clearly demonstrates the feasibility of Trace-Driven modeling as a source of insight into the operation of real computer systems and as a method of investigating theoretical computer systems. He carried out the study on a large-scale computer, the CDC 6600, under two different operating systems, UT-1 and UT-2, and gathered the data used while the system was running its normal production load. He uses Trace-Driven modeling to analyze the effect of increasing the speed of the central processor, peripheral processors, and disks.

SHERS 72b PER QA76 A772

Sherman, S.W., Baskett III, F., and Browne, J.C.; "Trace-Driven Modeling and Analysis of CPU Scheduling in a Multiprogramming System" Comm ACM, vol.15, no.12; (Dec.1972) pp.1063-1069.

Also available under ACM (SIGOPS) Workshop on System Performance Evaluation, april 1971. This paper describes a Trace-Driven model done using microscopic level job stream data obtained in a production environment by an event-driven sofware probe used to drive a model of a multiprogramming computer system (CDC 6600 running under the UT-1 operating system). The scheduling method tested included the best possible and worst possible methods, the traditional methods, round-robin, FCFS, etc., and dynamic predictors. The relative and absolute performance of these scheduling methods are given. It is concluded that a successful CPU scheduling method must be preemptive and must prevent a given job from holding the CPU for too long a period.

SHERS 73

Sherman, S.W., and Browne, J.C.; "Trace-Driven Modeling: Review and Overview" Symposium ACM SIGSIM; (June 1973) pp.200-207.

A review of the simulations of computer systems which have used Trace-Driven modeling as the simulation tool is presented. The advantages and disadvantages of Trace-Driven modeling are examined and a comparison to conventional distribution driven simulation techniques is given.

SLOTD 70 QA76 A3 V.10

Slotnick, D.L.; "Logic per Track Devices" Advances in Computers, vol.10, Academic Press; (1970) pp.291-296.

This article describes a system which could not exist and would not even be worth thinking about in the absence of cheap, highly reliable electronic parts. Head per track disks are now state-of-the-art.

SPREH 73 QA76 I567

Spreen, H.; "High Performance Input/Output Channels" International Computing Symposium; Gunther, Levrat and Lipps Editors; (1973) pp.231-236.

The steady increase in revolution speed and information density of external storage units (that is, drums and disks) requires high performance I/O channels in current and future central processors. The basic results are that high performance I/O channels have to provide local buffering capabilities and that such I/O channels must be implemented by using individual logic for the most part of them.

STONH 73 PER QA76 A772

Stone, H.S., and Fuller, S.H.; "On the Near-Optimality of the Shortest-Latency-Time-First Drum Scheduling Discipline" Comm ACM, vol.16, no.6; (June 1973) pp.352-353.

This paper shows that the total time to access the entire collection for any SLTF schedule is never as much as a drum revolution longer than a minimum latency schedule.

STORT 77 PER QA76.5 S653

Storey, T., and Todd, S.; "Performance Analysis of Large Systems" Software-- Practice and Experience, vol.7; (jn-jl.1977) pp.363-369.

A hybrid analytic and experimental approach to the analysis of large systems is described. The approach is iterative under the assumption that a correct analysis will not be made first time. It is primarily aimed at the analysis of the performance effect of proposed changes prior to coding them. The requirement is for a fairly approximate analysis that is easy to make. The different reasons for analysing performance of a system are discussed and then the ways of approaching the analysis, with an outline of their approach. The details of their approach are given and illustrated by their experiment.

TEICD 66 PER QA76 A772

Teichroew, D., and Lubin, J.F.; "Computer Simulation -- Discussion of the Technique and Comparison of Languages" Comm ACM, vol.9, no.10; (Oct.1966) pp.723-739.

An early paper and excellent survey giving a detailed comparison of six simulation languages and software packages for digital computer: SIMSCRIPT, CLP, CSL, GASP, GPSS and SOL. The features of each language are compared in a series of tables, and the comparison expanded in the text. The systems for which these languages have been implemented are given. In addition to the six languages compared in details, a number of other simulation languages, both descrete and continous, are also listed.

TELFD 75

Telford, D.; "Sotware Monitoring" Session 51C, VIM-23 Proceedings; (Nov.1975) pp.146-150.

The purpose of this presentation is to discuss LMSC'S experience in software monotoring. In particular, he discusses the following 4 areas: 7054 Controller Study; Generalized Systems Monitoring; Program Residencey Studies; Program Performance Monitoring. This study was conducted on a CDC Cyber 171 under SCOPE 3.4.3.

TEORT 72 PER QA76 A772

Teorey, T.J., and Pinkerton, T.B.; "A Comparative Analysis of Disk Scheduling Policies" Comm ACM, vol.15, no.3; (March 1972) pp.177-184.

Five well-known scheduling policies for movable head disks are compared using the performance criteria of expected seek time and expected waiting time. Both analytical and simulation results are obtained. The selection and implementation of a maximum performance two-policy algorithm are discussed. This is a very complete discussion of current analysis technology in this area.

TEORT 78 PER QA76 A772

Teorey, T.J.; "General Equations for Idealized CPU-I/O Overlap Configurations" Comm ACM, vol.21, no.6; (June 1978) pp.500-507.

General equations are derived for estimating the maximum possible utilization of main storage partitions, CPU and I/O devices under different computer system. Examples are provided to illustrate the use of the equations to complete effective processing time per record and expected timesharing response time under both balanced and unbalanced resource utilization. A series of live test experiments were run with a Nova 1200 minicomputer system with a Caelus 303/2 movable head disk.

TOWSD 78 PER QA76 A772

Towsley, D., Chandy, K.M, and Browne, J.C.; "Models for Parallel Processing Within Programs: Application to CPU:I/O and I/O:I/O Overlap" Comm ACM, vol.21, no.10; (Oct.1978) pp.821-831.

Approximate queueing models for internal parallel processing by individual programs in a multiprogrammed system are developed in this paper. The models are formulated in terms of CPU:I/O and I/O:I/O overlap and applied to the analysis of these problems. The percentage performance improvement from CPU:I/O overlap is found to be greatest for systems which are in approximate CPU:I/O utilization balance and for low degrees of multiprogramming. The percentage improvement from I/O:I/O overlap is found to be greatest for systems in which the I/O system is more utilized than the CPU. These models were validated by running synthetic mixes on a CDC 6400.

WILHN 76 PER QA76 A772

Wilhelm, N.C.; "An Anomaly in Disk Scheduling: A Comparison of FCFS and SSTF Seek Scheduling Using an Empirical Model for Disk Accesses" Comm ACM, vol.19, no.1; (Jan.1976) pp.13-17.

A model for disk accesses based on published measurements is developed. The model is used to show that under highly probable conditions, FCFS seek scheduling is superior to SSTF scheduling in the sense of having a lower mean queue length. A simple example of an arrival sequence illustrating this anomaly is presented.

WILSC 75

Wilson, C.B.; "Software Monitoring" Session 51C, VIM-23 Proceedings; (Nov.1975) pp.138-145.

At the Naval Surface Weapons Center, they have been doing performance measurement since their first installation of a CDC 6400 in August 1970. An history of improvement done by monitoring the system, from their NOLOS (modified SCOPE 3.2) to the KRONOS operating system, is briefly described.

PART II. TEXTS AND INFORMATION ON RELATED TOPICS

BAUEF 73 QA76.6 A33 1972

Bauer, F.L., et al; "Advanced Course on Software Engineering" Lecture Notes in Economics and Mathematical Systems, vol.81; Springer-Verlag, Heidelberg, Germany; (1973) 545 pp.

The present book is a consolidated effort of a group of experts that have been carefully prepared in a two-week seminar to illustrate the use of the term: SOFTWARE ENGINEERING. As Bauer said: " it comes from the fact that people are forced to live with machines that they do not want. They have not constructed them, they simply receive them and have to make the best out of it. Sometimes, with the chance of buying a new machine, there is some hope that the situation will improve but, for simple market considerations, the manufacturer does everything he can do to make the customer stay with the product and this usually ends all hopes for improvements. Thus, Software Engineering, for the time being, is partly a defense stratagem. I hope one day Software Engineering considerations will dictate how machines are to be built and then to be used".

BRINP 73 QA76.5 B76

Brinch Hansen, P.; "Operating Systems Principles" Prentice-Hall, Inc., Englewood Cliffs, N.J.; (1973) 366 pp.

This book is an exellent overview of the problems and trade-offs inherent in operating system design; it also details the current state of their solution. Chapter One gives an overview of operating systems. Chapters Two and Three present an abstract view of computational processes, both sequential and asynchronous. Chapters Four through Six discuss techniques of implementing processes on present-day computers. Chapter Seven outlines the emerging field of protection, and chapter Eight describes and gives a critical analysis of the operating system for the RC4000. This book is required reading for anyone who would design a large system or would like to know how such a system should be put together.

DIJKE 68 PER QA76 A772

Dijkstra, E.W.; "The Structure of 'THE' Multiprogramming System" Comm ACM, vol.11, no.5; (May 1968) pp.341-346.

An overview of the design and development of a distributed operating system is presented with a personal touch. The system by nature implements multiprogramming among a group of

cooperating processes that are incorporated in a hierarchical software system. The different levels of the hierarchy handle specific tasks in the interest of facilitating logical design, efficient operation, and system verification. The basic nature of process synchronization is advanced by a description of semaphore operation. The paper proposes that these design techniques can be successful in large system efforts.

DRUMM 73 QA76.5 .D74

Drummond, M.E.; "Evaluation and Measurement Techniques for Digital Computer Systems" Prentice-Hall, Inc., Englewood Cliffs, N.J.; (1973) 338 pp.

A complete treatment of computer evaluation and measurement principles and their applications to design and installation management are seen. Chapters One and Two give an introduction and the historical evolution of evaluation and measurement. Chapter Three concentrates on the evaluation of CPU's. Chapter Four introduces the system approach through the use of formula timing. Chapters Five and Six are a brief introduction to simulation techniques and data acquisition. Chapter Seven presents the elementary forms of programmed measurement techniques. Chapters Eight through Ten present various forms of hardware monitoring, methods of presenting the results, and various techniques of evaluation and measurement applied to particular problems.

FERRD 78

Ferrari, D.; "Computer System Performance Evaluation" Prentice-Hall,Inc., Englewood Cliffs,N.J.; (1978) 540 pp.

This book shows performance evaluation techniques and their applications to various types of computer performance problems for all the major computers, plus a basic discussion of program performance evaluation. Concentrating on concepts and methods that have multiple applications, treatment is independent of any particular manufacturer's products or any particular machine. The book explains, step-by-step, how to apply measurement, simulation, and analytic techniques of evaluation in the study of computer performance. Included is detailed coverage of measurement studies and measurment tools, experiment design, model formulation and construction, model calibration and validation, deterministic and probalistic model of systems, and all the latest tools employed in these techniques.

GRIED 78

Gries, D., et al; "Programming Methodology" Texts and Monographs in Computer Science; Springer-Verlag, New York; (1978) 437 pp.

This book presents major papers by members of Working Group 2.3 of the International Federation of Information Processing. Suitable as a supplementary reader for courses or seminars on the subject. Part one examines programming, software engineering and similar topics. Part two deals with the use of correctness proof and the related topic of defining a programming language so as to facilitate proofs. Part three explores the problem of harnessing parallelism so that it can be used effectively. Part four and five feature papers on data types and software development.

HELLH 75

Hellerman, H., and Conroy, T.F.; "Computer System Performance" McGraw-Hill Inc.; (1975) 380 pp.

This book is concerned primarily with performance as opposed to function. The reader is assumed to be already familiar with functional aspects of machine organization and programming. The First chapter is an overview of major performance issues in computer systems. Chapters Two and Three cover selected topics in statistics and discrete mathematics. Chapters Four and Five deal with job processing and queuing models. Chapter Six discusses the relatively simple class of cases involving a single job. Chapter Seven describes major design options of operating systems. Chapter Eight discusses the IBM OS/360 operating system. Chapter Nine is devoted to time sharing systems, and chapters Ten and Eleven examine the theory and practice of virtual storage. A set of exercises for each chapter is supplied at the end of the book.

JENSK 74 QA76.73 P35J46

Jensen, K., and Wirth, N.; "PASCAL: User Manual and Report" Lecture Notes in Computer Science, Springer-Verlag, Berlin, Heidelberg, Germany; (June 1974) 170 pp.

This manual is based on 'The Programming Language PASCAL' (revised report).
The user manual: chapters 0 through 12 define the language PASCAL and serve as a standard for both the implementor and the programmer. Chapters 13 and 14 document the implementation of PASCAL on the CDC 6000 series machines.
The report: written by Wirth; it describes the few changes that have been made in the past few years and, in order to

provide a common base for implementation on various computers, the language defined, in this report, is called STANDARD PASCAL. This book is much better and more understandable that the first one (see WIRTN 71), but still there are few points where the description or the examples are not fully explanatory. These manual and report are important in operating systems as PASCAL is being used as a base language for many studies of operating system synchronization techniques.

MADNS 74 QA76.6 M33

Madnick, S.E., and Donovan, J.J.; "Operating Systems" McGraw-Hill, Inc.; (1974) 640 pp.

This book presents advanced software techniques, especially focusing on operating systems. It presents material that will enable the reader to design, use, and analyze not only current operating systems but future ones as well. Chapter One gives an overview of operating systems. Chapter Two describes I/O and interrupt programming. Chapters Three through Six describes the memory, processor, device and information management respectivily. Chapter Eight gives a small introduction to performance evaluation, and chapter Nine is case studies of some wide-spread models of the IBM 360/370. In addition this book has a very good annotated bibliography.

MILLE 72 QA76 A1C6

Miller, E.F.Jr.; "Bibliography on Techniques of Computer Performance Analysis" Computer, vol.5; (Sept/Oct.1972) pp.39-47.

An extensive bibliography of the entire area of computer performance can be found with some short annotations.

VANTD 74 QA76.6 V37

Van Tassel, D.; "Program Style, Design, Efficiency, Debugging and Testing" Prentice-Hall, Inc., Englewood Cliffs, N.J.; (1974) 256 pp.

This book is written for those who already know how to program but wish to increase their programming proficiency. The contents cover five subjects that are seldom discussed in beginning programming books: the style or readability of programs, program design, efficiency or optimization of programs, debugging, and testing. Not too many people will fail to profit from reading it.

WIRTN 71 PER QA76 A265

Wirth, N.; "The Programming Language PASCAL" Acta Informatica 1, Springer-Verlag Berlin, Germany; (1971) pp.35-63.

This paper describes the programming language PASCAL and provides a justification for its creation. Wirth intended the language as a teaching vehicle and as a tool for the creation of large programs. He tried to keep the language small, systematic, and efficient. PASCAL introduced several novel concepts and has inspired several other languages. In style, this paper is like the ALGOL 60 report but, unfortunately, it is neither as clear nor as complete.

WIRTN 73 QA76.6 W5713

Wirth, N.; "Systematic Programming: An Introduction" Prentice-Hall, Inc., Englewood Cliffs, N.J.; (1973) 167 pp.

This book introduces programming as the art or technique of constructing algorithms in a systematic manner, as a discipline in its own right. No specific area of application is emphasized. This book is tailored to the needs of people who view a course on systematic programming as a (possibly neglected) part of basic mathematical training. Few people beside Dijkstra will fail to profit from reading it.

WIRTN 76

Wirth, N.; "Algorithms + Data Structure = Programs" Prentice-Hall, Inc., Englewood Cliffs, N.J.; (1976) 360 pp.

This book presents a very systematic and scientific approach to the fundamental techniques associated with data composition and program development. The basic principles covered here are applicable to many scientific and fundamental data structures, interval and external sorting, recursive algorithms, dynamic data structures (recursive data types, pointers, list structures, tree structures, optimal search trees, multiway trees, and key transformations), and language structures and compiling (language definition and analysis, syntax graph, parser and translator contruction).

PART III. CDC SYSTEM MANUALS AND REFERENCES ON CDC

ABELV 70 TK7885 A1J6

Abell, V.A., Rosen, S., and Wagner, R.E.; "Scheduling in a General Purpose Operating System" Proc AFIPS FJCC, vol.36; (1970) pp.89-96.

This paper describes some aspects of an operating system (MACE) running at the Purdue University Computing Center on a CDC 6500 supported by an IBM 7094. The paper deals mostly with the scheduling mechanisms and strategies used in the system. It was the first time that scheduling and job movement techniques of the type described here have been implemented and used in a very large system.

ALLAR 64 PER QA76 A772

Allard, R.W., Wolf, K.A., and Zemlin, R.A.; "Some Effects of the 6600 Computer on Language Structure" Comm ACM, vol.7, no.2; (Feb.1964) pp.112-119.

The problem of compiling efficient 6600 codes prompted the development of an intermediate language reflecting the structure of the machine that is more easily manipulated in improving object program efficiency. The subject of this paper is the intermediate language and method of manipulating it. A simplified 6600 structure is presented to illustrate the compiling method.

BEDOR 74b

Bedoll, R.F.; "B I D -- BCS Interactive Debug System" VIM-21 Proceedings; (Oct.1974) pp.220-224.

BID is a system for the interactive debugging of FORTRAN and COMPASS programs. It is currently operational at the Boieng Computer Services on a dual-mainframe KRONOS 2.0 system, where it enjoys moderate usage by interactive users. BID has been available for about six months.

CDC 73a

Control Data Cyber 70 Model 73 Computer Systems; "System Description and Programming Inf. Ref. Manual vol.1" CDC Publication no.60347300; (1973).

This volume contains the system description and general programming information. It gives a general idea of how the hardware of the computer works.

CDC 73b

Control Data Cyber 70/6000 Series Computer Systems; "UPDATE Reference Manual" CDC Publication no.60342500; (1973).

This publication describes the UPDATE program for maintaining and updating source decks on libraries in compressed symbolic format.

CDC 74a QA76.5 C5925

Control Data Cyber 170/70, 6000/7600 Series Computer Systems; "COMPASS Version 3 Reference Manual" CDC Publication no.60360900; (1974).

This manual describes the principles, features, methods, rules, and techniques of producing a COMPASS language program. Essential tool for a systems programmer.

CDC 74b

Control Data Cyber 170/70, 6000/7600 Series Computer Systems; "MODIFY Reference Manual" CDC Publication no.60281700; (1974).

MODIFY is designed to maintain and update source decks that are on libraries in a compressed symbolic format. Essential tool to a systems programmer.

CDC 74c

Control Data Cyber 70/6000 Series Computer Systems; "SCOPE Reference Manual Ver.3.4" CDC Publication no.60307200; (May 1974).

This manual describes the SCOPE 3.4.3 Operating System. It was written for programmers who use all the source languages which operate under SCOPE 3.4, and it includes information of specific interest to those who write in Compass assembly language. The reference manual for CDC's, original (batch-oriented) operating system for the 6000 series.

CDC 75a

Control Data Cyber 70/6000 Series Computer Systems; "NOS Ver.1 Text Editor Reference Manual" CDC Publication no.60436100; (1975).

This manual contains the information a user must have to use the text editor. Its purpose is to effect character-oriented data manipulations from a remote terminal. A not very powerful text editor.

CDC 75b QA76.5 C59282

Control Data Cyber 70/6000 Series Computer Systems; "KRONOS 2.1 Reference Manual vol.1 and 2" CDC Publication no.60407000 and 60448200; (June 1975).

These manuals describe the external features of KRONOS 2.1.2 for the batch user. Vol.1 contains information for the applications programmer. Vol.2 contains information for those who write system or assembly language programs for use with KRONOS.

CDC 75c

Control Data Cyber 170 and 70/6000 Series Computer Systems; "NOS 1.0 Applications Programmer's Instant" CDC Publication no.60436000; (1975).

This manual provides condensed descriptions of system control statements, loader, product set, etc. Good tool for users.

CDC 76a

Control Data Cyber 70/6000 Series Computer Systems; "NOS Ver.1 Time-Sharing User's Reference Manual" CDC Publication no.60435500; (1976).

This manual describes communication between a remote time sharing terminal user and the NOS time-sharing systems. Necessary tool for systems programmers and other users.

CDC 76b

Control Data Cyber 70/6000 Series Computer Systems; "NOS Ver.1 Reference Manual vol.1 and 2" CDC Publication nos.60435400 and 60445300; (mar.1976).

These manuals describe the external features of the NOS operating system for the batch user. Vol.1 contains information for the applications programmer. Vol.2 contains information for those who write system or assembly language programs for use with NOS. The systems programmer's bible.

CDC 76c

Control Data Cyber 170 and 70/6000 Series Computer Systems; "NOS Ver.1 Systems Programmer's Instant" CDC Publication no.60449200; (Feb.1976).

This manual provides condensed descriptions of functions of the operating system for analysts, programmers, and operators. The pocket size systems programmer's bible.

CDC 76d

Control Data Cyber 70/6000 Series Computers; "KRONOS 2.1 Workshop Reference Manual" CDC Publication no.97404700; (April 1976).

The purpose of this KRONOS 2.1 Workshop manual is to provide the system analyst with detailed internal documentation readily available in a single manual. This workshop provides additional explanations for those areas which are not self-explanatory. The operating system anatomy.

CDC 76e

Control Data Corporation; "CDC Disk Storage Unit" CDC Publication no.70629500; (Sept.1976).

This manual has been prepared for customer engineers and other technical personnel directly involved with maintaining the disk storage unit (drive).

CDC 76f

Control Data Corporation; "CDC Disk Storage Subsystem" CDC Publication no.60363900; (Dec.1976).

This manual contains reference information for disk storage subsystems using one or more CDC 7x54 Series Disk Storage Controllers to handle CDC 844-2/21/41/44 Disk Storage Units.

CDC 77

Control Data Cyber 170/70, 6000 Series Computer Systems; "NOS Version 1 Operator's Guide" CDC Publication no.60435600; (mar.1977).

This manual contains information and procedures necessary to establish and control operation of a CDC NOS Operating System and is intended for use by the system operator.

CDC 77a

Control Data Corporation; "Intallation Handbook" CDC Publication no.60435700; (Jan.1977).

This handbook provides an analyst with the information to install the NOS Operating System. It contains information such as: procedure for deadstarting the system, general and specific installation information, information necessary to maintain the system once it is installed. A very helpful manual.

CDC 78

Control Data Corporation; "NOS Ver.1 Internal Maintenance Specification" CDC Publication no.60454300; (june 1978).

This internal maintenace specification (IMS) provides the system analyst with detailed internal documentation of the NOS system. Included are detailed descriptions of system routines and the system interfaces, tables and flowcharts of these routines. A remarkable manual in CDC history.

FRANW 74 PER QA76.5 S653

Franta, W.R., and Houle, P.A.; "On a Loose Communication Between Dissimilar CDC 6000 Operating Systems" Software--Practice and Experience, vol.4; (1974) pp.231-236.

This paper describes a system, known as LINK, which provides inter-machine communication between like machines operating under significantly different operating systems (CDC 6400 under KRONOS and CDC 6600 under MOMS). The methodology employed by LINK was selected from among several alternatives on the basis of effectiveness, complexity, and cost.

HARRM 67 PER QA76.A772

Harrison, M.C., and Schwartz, J.T.; "SHARER, a Time Sharing System for the CDC 6600" Comm ACM, vol.10, no.10; (Oct.1967) pp.659-665.

A time sharing system (SHARER) embedded within the standard batch processing system (SCOPE) for the CDC 6600 is described. The system is general purpose and file-based, providing facilities for file input, manipulation, editing, compilation, and conversational execution. It uses a simple scheme for system execution for a machine with only one relocation and memory bound register.

JAY,S 74

Jay, S.; "Measuring 3.4 Scheduler Performance" Session 41B, VIM-21 Proceedings; (Oct.1974) pp.211-212.

This paper describes briefly a standard, although somewhat hidden, feature of the SCOPE operating system which allows realistic measurement of scheduler performance. By use of this feature, installations can determine if their attempts to improve scheduler performance are succeeding.

THORJ 70 TK7889 C2T5

Thornton, J.E.; "Design of a Computer: The CDC 6600" Scott, Foresman and Co., Glenview, Illinois; (1970) 181 pp.

This book is offered as a Case Study of a major digital computer which has reduced to practice a number of interesting theories involving parallelism and concurency. It describes one of the early machines attempting to explore parallelism in electrical structure without abandoning the serial structure of the computer programs. One or maybe the only one, in CDC literature, which describes the CDC 6600 serie machines in full and understandable way (but maybe in a too hardware manner for the non-hardware guy).

APPENDIX A

Frequently Used Scheduling Algorithms

FCFS - First Come First Served, requests are honoured in the order they are generated (DENNP 67).

SSTF - Shortest Seek Time First, service the one having a track address closest to the current position of the head (DENNP 67).

SCAN - the disk arm acts as a shuttle as it sweeps back and forth across all the cylinders, servicing requests. It changes direction only at the inner and outer cylinders (DENNP 67).

LOOK - variation of SCAN, allows the disk arm to change direction when there are no further requests to service in the current direction (TEORT 72).

ESCHENBACH - is a deterministic scanning technique designed for message-switching systems which normally operate under heavy loading conditions; (TEORT 72)

- in order to allow the possibility of servicing all 'm' (total number of sector/track) sectors for the cylinder, the arm must remain in position for at least E revolutions;
- the arm may spend E revolutions on cylinder 0 before

moving to cylinder 1; and so until E revolutions are spent on cylinder N. The arm then moves directly back to cylinder 0 without stopping at any intermediate cylinders.

C-LOOK - combines the best characteristics of LOOK and the Eschenbach scheme.

N-SCAN - the first N requests are serviced before servicing more recent requests; (FRANH 69)

- first scan in the direction for which the farthest request distance is a minimum;
- then scan back until the remaining requests have been serviced.

FSCAN - takes the entire queue, at each decision point, and services the requests in a scan whose direction is determined by the minimum distance the current head position is from the outermost and innermost track addresses of the requests to be serviced. Thus the scan is preceeded by a move of the disk head to the nearer of the extreme track addresses (COFFE 72).

SLTF - Shortest Latency Time First (often called SATF) simply means that whenever the drum is free to read or write another record, it begins accessing the first record, among those remaining to be processed, to come under the read/write heads (DENNP 67).

MTPT - Minimal Total Processing Time (called in the original article an "Optimal Drum Scheduling Discipline"), has as its basis SLTF but tries to minimize the total latency time applying the traveling salesman problem. This is the most complicated scheduling algorithm which has a complexity of the order of N log N (FULLS 72).

PRIO - priority queueing, the I/O request of the highest priority job in the system is honoured first without regard for the position of the disk arm (PELTV 75).

APPENDIX B

In this appendix we show the various definitions necessary to assemble and execute the performance monitor subsystem and the program SAMPLE itself.

STORAGE ALLOCATION.

ADDRESS	LENGTH	BINARY CONTROL CARDS.
111	6648	IDENT SAMPLE,FETS,SAMPLE
6751		END

BLOCKS	TYPE	ADDRESS	LENGTH
PROGRAM*	ABSOLUTE	0	682
LITERALS*	ABSOLUTE	682	5
BUFFERS	ABSOLUTE	687	6142

ENTRY POINTS.

SAMPLE 160 RFL= 6751 SSJ= 153

```
                IDENT  SAMPLE,FETS,SAMPLE                                SAMPLE      1
                ABS                                                      SAMPLE      2
                SST                                                      SAMPLE      3
                SYSCOM B1                                                L419        1
                ENTRY  SAMPLE                                            L419        2
                ENTRY  RFL=                                              L419        3
                ENTRY  SSJ=                                              L419M1      1
          COMMENT 76/04/26. 77/06/29. SAMPLE - EVENT TRACE DUMP SUBSYSTEM.  L419     4


***             SAMPLE - EVENT TRACE DUMP SUBSYSTEM.                     L419        7
*               G.SPIX.          74/08/01.                               L419        8
*               G.MACK.          77/04/19.                               L419        9


***             SAMPLE IS A PROGRAM TO FLUSH EVENT TRACE DATA FROM THE CP L419       11
*               AND PP BUFFERS IN LOW CORE TO DISC FILES, FOR LATER USE   L419       12
*               BY ANALYSIS/MODELLING PROGRAMS.                           L419       13
*                  THREE FILES ARE CREATED BY SAMPLE DURING A RUN. THESE ARE- L419   14
*               MDUMP - DUMP OF LOW CORE AT START OF RUN,                 L419       15
*               CDUMP - DUMP OF PP TRACE BUFFER, AND                      L419       16
*               FDUMP - DUMP OF CP TRACE BUFFER.                          L419       17
*                                                                         L419       18
*               MDUMP FORMAT -                                            L419       19
*               ONE LOGICAL RECORD. THE FIRST WORD OF THE FILE IS THE REAL L419       20
*               TIME CLOCK AT THE START OF THE RUN. THE REMAINING WORDS   L419       21
*               ARE A COPY OF LOW CORE FROM ADDRESS 0 TO THE LAST WORD OF  L419      22
*               THE EQUIPMENT STATUS TABLE (EST).                         L419       23
*                                                                         L419       24
*               CDUMP AND FDUMP FORMAT -                                  L419       25
*               ONE LOGICAL RECORD ON EACH FILE.                          L419       26
*               FILES CONSIST OF TRACE ITEMS AS COPIED FROM THE BUFFER IN L419       27
*               LOW CORE. FOR FORMAT OF THESE DATA SEE THE COMMON DECK    L419       28
*               *COMSETD* - EVENT TRACE DEFINITIONS (LISTED BELOW).       L419       29


                LIST   X                                                 SAMPLE      0
                                                                         L419       31
```

```
                        CTEXT  CONSETD - EVENT TRACE DEFINITIONS.                    CONSETD    1


                  ***   CONSETD - EVENT TRACE DEFINITIONS.                           CONSETD    3
                  *                                                                  CONSETD    4
                  *     G.E.MACK          77/04/05.                                  CONSETD    5
                  *     DEPARTMENT OF COMPUTER SCIENCE,                            CONSETD    6
                  *     CONCORDIA UNIVERSITY,                                      CONSETD    7
                  *     MONTREAL, CANADA.                                          CONSETD    8


                  ***   CONSETD CONTAINS EQUIVALENCES NECESSARY TO ASSEMBLE THOSE   CONSETD   10
                  *     PORTIONS OF THE OPERATING SYSTEM WHICH IMPLEMENT THE EVENT  CONSETD   11
                  *     TRACE DUMP FEATURE.                                        CONSETD   12


                        IF     -DEF,QUAL$,1                                        CONSETD   14
                        QUAL   ETD                                                 CONSETD   15
                                                                                   CONSETD   16
                  D_H   BASE   MIXED                                               CONSETD   17


                  ***   BUFFER DEFINITIONS.                                        CONSETD   19
                  *                                                                CONSETD   20
                  *     TWO BUFFERS ARE DEFINED IN CHR. THESE ARE THE MTR OR PP    CONSETD   21
                  *     DUMP BUFFER, AND THE CPUMTR OR CP DUMP BUFFER. THE BUFFERS CONSETD   22
                  *     ARE DYNAMICALLY ALLOCATED AT DEADSTART BY THE ROUTINE      CONSETD   23
                  *     SET/ICM.                                                   CONSETD   24
                  *        A POINTER WORD IS MAINTAINED IN THE SYSTEM LOW-CORE      CONSETD   25
                  *     INSTALLATION AREA FOR EACH BUFFER. EACH POINTER WORD        CONSETD   26
                  *     CONTAINS TWO ITEMS. THE FIRST IS THE ABSOLUTE FIRST-WORD   CONSETD   27
                  *     ADDRESS OF THE BUFFER, AND THE SECOND IS A RELATIVE WORD    CONSETD   28
                  *     INDEX WHICH POINTS TO THE NEXT WORD TO STORE INTO. THE     CONSETD   29
                  *     INDEX IS RELATIVE TO THE FIRST-WORD ADDRESS OF ITS BUFFER, CONSETD   30
                  *     AND IS USED IN A SIMILAR FASHION TO THE *IN* POINTER OF A  CONSETD   31
                  *     CIO FILE ENVIRONMENT TABLE (FET).                          CONSETD   32
                  *        NOTE THAT THERE IS NO EQUIVALENT TO CIO'S *OUT* POINTER CONSETD   33
                  *     IN LOW CORE, AND A *LIMIT* POINTER IS NOT NECESSARY SINCE  CONSETD   34
                  *     THE BUFFER LENGTH IS AN ASSEMBLY CONSTANT.                 CONSETD   35
                  *                                                                CONSETD   36
                  *     NOTE THAT THE TWO INDICES ARE INCREMENTED WITH MODULO      CONSETD   37
                  *     ARITHMETIC, HENCE IT IS IMPERATIVE THAT THE BUFFER LENGTHS CONSETD   38
                  *     BE AN INTEGRAL POWER OF TWO (2**N WHERE N IS AN INTEGER).  CONSETD   39
```

```
                        **      BUFFER LENGTHS.                                          CONSETD  41
                        *                                                                CONSETD  42
                        *       WARNING - THESE MUST BE AN INTEGRAL POWER OF 2 (2**N).   CONSETD  43
                                                                                         CONSETD  44
      1000      YPBL    EQU     1000        PP (MTR) BUFFER LENGTH                       CONSETD  45
      2000      YCBL    EQU     2000        CP (CPUMTR) BUFFER LENGTH                    CONSETD  46


                        **      BUFFER POINTER WORDS.                                    CONSETD  48
                        *                                                                CONSETD  49
                        *       PP BUFFER POINTER WORD FORMAT.                           CONSETD  50
                        *                                                                CONSETD  51
                        *T YPBW 12/ INDEX,36/ UNUSED,18/ FWA                             CONSETD  52
                        *                                                                CONSETD  53
                        *       WHERE                                                    CONSETD  54
                        *                                                                CONSETD  55
                        *       INDEX  IS THE RELATIVE INDEX (SEE ABOVE), AND            CONSETD  56
                        *       FWA    IS THE BUFFER FWA.                                CONSETD  57
                        *                                                                CONSETD  58
                        *       NOTE THAT THE ABOVE INDEX IS 12 BITS, HENCE THE VALUE    CONSETD  59
                        *       OF YPBL MUST BE < 2**12.                                 CONSETD  60
                        *                                                                CONSETD  61
                        *       CP BUFFER WORD FORMAT.                                   CONSETD  62
                        *                                                                CONSETD  63
                        *T YCBW 24/ 0,18/ FWA,18/ INDEX                                  CONSETD  64
                        *                                                                CONSETD  65
                        *       THESE WORDS RESIDE IN THE LOW-CORE INSTALLATION AREA.    CONSETD  66
                                                                                         CONSETD  67
        16      YPBW    EQU     INSL+6      PP BUFFER WORD                               CONSETD  68
        17      YCBW    EQU     INSL+7      CP BUFFER POINTER WORD                       CONSETD  69


                        ***     USER PP REQUEST CONTROL.                                 CONSETD  71
                        *                                                                CONSETD  72
                        *       TWO CELLS ARE DEFINED IN THE CONTROL POINT INSTALLATION  CONSETD  73
                        *       AREA. THEY ARE USED TO SAVE THE USER'S LAST PP REQUEST AND CONSETD  74
                        *       THE TIME OF THIS REQUEST, FOR LATER USE IN CALCULATING   CONSETD  75
                        *       WAIT TIME FOR REQUEST SATISFACTION.                      CONSETD  76
                                                                                         CONSETD  77
                        **      YPRQ - LAST PP REQUEST (COPY OF INPUT REGISTER).         CONSETD  78
                        *       YPWT - REAL TIME OF REQUEST (FROM RTCL IN LOW-CORE).     CONSETD  79
                                                                                         CONSETD  80
        46      YPRQ    EQU     IN6W        LAST PP REQUEST                              CONSETD  81
        47      YPWT    EQU     IN7W        TIME OF LAST PP REQUEST                      CONSETD  82
```

```
                ***   FORMATS OF TRACE DATA ITEMS.                          COMSETD    84
                *                                                           COMSETD    85


H_D             BASE  *                                                     COMSETD    87
                IF    -DEF,QUAL$,1                                          COMSETD    88
                QUAL  *                                                     COMSETD    89
                                                                            COMSETD    90
                ENDX                                                        COMSETD    91
                LIST  *                                                     L419       33
                                                                            L419       34
                                                                            L419M1      2
                CTEXT COMCMAC - CPU SYSTEM MACROS.                          COMCMAC     1
```

```
                    **      EQUIVALENCES.
```

```
     100    WBFL    EQU    1000
    2400    CBUFL   EQU    24000
    3400    FBUFL   EQU    34000
    6000    MBUFL   EQU    CBUFL+FBUFL
```

```
L419      36
L419      37
SAMPLE    11
SAMPLE    12
SAMPLE    13
SAMPLE    14
SAMPLE    15
```

```
                          **      DATA AREA - FET DEFINITION.                                    SAMPLE   18
                          *                                                                      SAMPLE   19
                                                                                                 L419     39
  111                             ORG     111B                                                 L419     40
                                                                                                 SAMPLE   20
  111                     FETS    BSS     0                                                    SAMPLE   21
                                                                                                 SAMPLE   22
  111                     M       BSS     8           MEMORY INITIALIZATION DUMP               SAMPLE   23
  111 1504251520000000003 MDUMP   RFILE8  MBUF,MBUFL                                          SAMPLE   24
                                                                                                 SAMPLE   25
  121 777777777777777777  CBFP    CON     -0          CHANNEL BUFFER POINTER                   SAMPLE   26
  122 777777777777777777  CBFO    CON     -0          CHANNEL BUFFER OUT                       SAMPLE   27
  123 000000000000001000  CBFL    CON     /ETD/YPBL   PP BUFFER LENGTH                        L419     41
  124 777777777777777777  CBFW    CON     -0          CHANNEL BUFFER FWA                       SAMPLE   29
  125 000000000000000000  CBAC    CON     0           CHANNEL BUFFER ACTIVITY COUNT            SAMPLE   30
                                                                                                 SAMPLE   31
  126                     C       BSS     8           CHANNEL BUFFER DUMP FET                  SAMPLE   32
  126 030425152000000003  CDUMP   RFILE8  CBUF,CBUFL                                          SAMPLE   33
                                                                                                 SAMPLE   34
                                                                                                 SAMPLE   35
  136 777777777777777777  FBFP    CON     -0          FUNCTION BUFFER POINTER                  SAMPLE   36
  137 777777777777777777  FBFO    CON     -0          FUNCTION BUFFER OUT                      SAMPLE   37
  140 000000000000002000  FBFL    CON     /ETD/YCBL   CP BUFFER LENGTH                        L419     42
  141 777777777777777777  FBFW    CON     -0          FUNCTION BUFFER FWA                      SAMPLE   39
  142 000000000000000000  FBAC    CON     0           FUNCTION BUFFER ACTIVITY COUNT           SAMPLE   40
                                                                                                 SAMPLE   41
  143                     F       BSS     8           FUNCTION BUFFER DUMP FET                 SAMPLE   42
  143 060425152000000003  FDUMP   RFILE8  FBUF,FBUFL                                          SAMPLE   43
                                                                                                 L419M1    4
                    153   SSJ=    EQU     *           SSJ PARAMETER BLOCK                      L419M1    5
  153 000000077777                VFD     12/0,24/77777B      TIME LIMIT                       L419M1    6
                  0860            VFD     12/60B      CPU PRIORITY                             L419M1    7
                      7760        VFD     12/7760B    QUEUE PRIORITY                           L419M1    8
  154                   4         BSS     4           SSJL=(*-SSJ=)                            L419M1    9
```

```
                              **        MAIN PROGRAM.                                                  L419      44
                              *                                                                        L419      45
                              *         WHEN STARTED, THE PROGRAM DUMPS MEMORY TO FILE MDUMP, THEN     L419      46
                              *         BEGINS TO DUMP INFORMATION FROM THE CP AND PP BUFFERS TO       L419      47
                              *         FILES FDUMP AND CDUMP, RESPECTIVELY.                           L419      48
                              *         THE PROGRAM CONTINUES TO RUN UNTIL SENSE SWITCH 1 IS           L419      49
                              *         TURNED ON, AT WHICH TIME THE CIO BUFFERS ARE FLUSHED AND       L419      50
                              *         THE PROGRAM TERMINATES.                                        L419      51
                                                                                                       L419      52
                                                                                                       L419      53
 160  6110000001               SAMPLE   SB1     1            B1=1                                      SAMPLE    45
                0100000345              RJ      PRS          PRESET SAMPLE                             SAMPLE    46
 161  0100000273                        RJ      INI          INITIALIZE MEMORY DUMP                    SAMPLE    47
 162  13000                    SMP1     BX0     X0-X0        SET INACTIVE FLAG                         SAMPLE    48
          43601                         MX6     1                                                      SAMPLE    49
                5160000121              SA6     CBFP         RESET CHANNEL BUFFER POINTER              SAMPLE    50
 163  5160000136                        SA6     FBFP         RESET FUNCTION BUFFER POINTER             SAMPLE    51
                7160222302              SYSTEM RSB,WAIT,SMPA                                           SAMPLE    52
 167  5110000121                        SA1     CBFP         FETCH CHANNEL BUFFER POINTER              SAMPLE    53
                43614                   MX6     12           POSITION INDEX                            L419      54
                     11161              BX1     X6*X1                                                  L419      55
 170  20114                             LX1     12                                                     L419      56
           0100000250                   RJ      CBS          CHECK BUFFER STATUS                       SAMPLE    55
 171  7160222302                        SYSTEM RSB,WAIT,SMPB                                           SAMPLE    56
 174  5110000136                        SA1     FBFP         FETCH FUNCTION BUFFER POINTER             SAMPLE    57
                7211000000              SX1     X1+          ISOLATE INDEX                             L419      57
 175  0100000250                        RJ      CBS          CHECK BUFFER STATUS                       SAMRLE    58
 176  0310000162                        NZ      X0,SMP1      IF BUFFER ACTIVITY                        SAMPLE    59
                0100000462              RECALL                                                         SAMPLE    60
 177  0100000462                        RECALL                                                         SAMPLE    61
 200  0100000462                        RECALL               WAIT A WHILE                              SAMPLE    62
 201  0100000462                        RECALL                                                         SAMPLE    63
 202  0100000462                        RECALL                                                         SAMPLE    64
 203  56100                             SA1     B0           CHECK SENSE SWITCHES                      SAMPLE    65
          20165                         LX1     59-6                                                   SAMPLE    66
                0321000162              PL      X1,SMP1      IF SW1 NOT ON                             SAMPLE    67
 204  7120000126                        WRITER C,WAIT                                                  SAMPLE    68
 206  7120000143                        WRITER F,WAIT                                                  SAMPLE    69
 210  5110000125                        SA1     CBAC         FETCH COUNT OF CHANNEL ACTIVITIES         SAMPLE    70
                20106                   LX1     6                                                      SAMPLE    71
 211  0100000401                        RJ      COD                                                    SAMPLE    72
 212  5160000235                        SA6     SMPC                                                   SAMPLE    73
                7110000235              MESSAGE  SMPC                                                  SAMPLE    74
 214  5110000142                        SA1     FBAC                                                   SAMPLE    75
                20105                   LX1     5                                                      SAMPLE    76
 215  0100000401                        RJ      COD                                                    SAMPLE    77
 216  5160000241                        SA6     SMPD                                                   SAMPLE    78
                7110000241              MESSAGE  SMPD                                                  SAMPLE    79
                                                                                                       L419M1    10
 220  7160241115                        TIME    SMPE         CALCULATE CPU TIME USED BY SAMPLE.        L419M1    11
 223  5110000245                        SA1     SMPE                                                   L419M1    12
                5120000363              SA2     PRSA                                                   L419M1    13
 224  37112                             IX1     X1-X2                                                  L419M1    14
           0100000412                   RJ      CFD          CONVERT TO DISPLAY CODE                   L419M1    15
 225  5160000245                        SA6     SMPE                                                   L419M1    16
                7110000245              MESSAGE SMPE                                                   L419M1    17
                                                                                                       L419M1    18
```

```
  227  7110000602                    MESSAGE  (=C* SAMPLE COMPLETE.*)                    SAMPLE    80
  231  7160247021                    ENDRUN                                              SAMPLE    81
                                                                                         SAMPLE    82
                                                                                         SAMPLE    83
  233  0000001000016000121    SMPA   VFD      12/8,12/1,18//ETD/YPBW,18/CBFP             L419      58
  234  0000001000017000136    SMPB   VFD      12/8,12/1,18//ETD/YCBW,18/FBFP             L419      59
  235  5555555555555555555    SMPC   DATA     1H ,C* CHANNEL ACTIVITIES.*                SAMPLE    86
  241  5555555555555555555    SMPD   DATA     1H ,C* CENTRAL FUNCTIONS.*                 SAMPLE    87
  245  5555555555555555555    SMPE   DATA     1H ,C* CPU SECONDS USED.*                  L419M2    1
```

```
SAMPLE - EVENT TRACE DUMP SUBSYSTEM.                          COMPASS 3.3-433A.    79/03/13. 12.00.12.     PAGE    10
SUBROUTINES.

                                  **     CBS - CHECK BUFFER STATUS.                                    SAMPLE      89
                                  *                                                                    SAMPLE      90
                                  *      ENTRY   (X1) = CURRENT *IN*                                   SAMPLE      91
                                  *              (A1) = BUFFER DESCRIPTION TABLE.                      SAMPLE      92
                                                                                                       SAMPLE      93
                                                                                                       SAMPLE      94
   250  0000000000           CBS         PS      0               ENTRY/EXIT                            SAMPLE      95
   251  54211                            SA2     A1+B1           FETCH LAST *OUT*                      SAMPLE      96
             37312                       IX3     X1-X2           COMPARE IN - OUT                      SAMPLE      97
                  0333000253             NG      X3,CBS1         IF WRAP AROUND                        SAMPLE      98
   252  7233777677                       SX3     X3-WBFL                                               SAMPLE      99
                  0333000250             NG      X3,CBS          IF NOT THRESHOLD                      SAMPLE     100
   253  76010                CBS1        SX0     B1              SET ACTIVITY FLAG                     SAMPLE     101
             43601                       MX6     1               RESET BUFFERS                         SAMPLE     102
                  7272000100             SX7     X2+WBFL         INCREMENT BUFFER POINTERS             SAMPLE     103
   254  5160000687                       SA6     WBUF                                                  SAMPLE     104
                  54121                  SA1     A2+B1           FETCH BUFFER LENGTH                   SAMPLE     105
                       54311             SA3     A1+B1           FETCH BUFFER FWA                      SAMPLE     106
   255  37617                            IX6     X1-X7           COMPARE OUT TO LIMIT                  SAMPLE     107
             54720                       SA7     A2              UPDATE BUFFER POINTER                 SAMPLE     108
                  0316000257             NZ      X6,CBS2         IF NOT AT *LIMIT*                     SAMPLE     109
   256  54620                            SA6     A2                                                    SAMPLE     110
   257  5110000267           CBS2        SA1     CBSA            READ SKELETON                         SAMPLE     111
                  36223                  IX2     X2+X3           SET BLOCK FWA                         SAMPLE     112
                       54431             SA4     A3+B1           INCREMENT ACTIVITY COUNT              SAMPLE     113
   260  20222                            LX2     18                                                    SAMPLE     114
             73641                       SX6     X4+B1                                                 SAMPLE     115
                  12712                  BX7     X1+X2           SET RSB FUNCTION                      SAMPLE     116
                       54640             SA6     A4                                                    SAMPLE     117
   261  5170000270                       SA7     CBSB                                                  SAMPLE     118
                  7160222302             SYSTEM  RSB,WAIT,A7                                           SAMPLE     119
   264  6160000607                       WRITEW  A4+B1,WBUF,WBFL                                       SAMPLE     120
   266  0400000250                       EQ      CBS         .  DUMP AND EXIT                          SAMPLE     121
                                                                                                       SAMPLE     122
   267  00001100000000000607 CBSA        VFD     12/,12/WBFL,18/,18/WBUF                               SAMPLE     123
   270  00000000000000000000 CBSB        CON     0                                                     SAMPLE     124


                                  **     INI - INITIALIZE SAMPLE SESSION.                              SAMPLE     126
                                  *                                                                    SAMPLE     127
                                  *      RESETS CURRENT *IN* POINTERS.                                 SAMPLE     128
                                  *      DUMPS 0 TO LWA(EST) ON FILE MDUMP.                            L419        61
                                                                                                       SAMPLE     130
                                                                                                       SAMPLE     131
   271  7120000111           INI2        WRITER  M,WAIT          FLUSH BUFFER                          SAMPLE     132
                                                                                                       SAMPLE     133
   273  0000000000           INI         PS      0               ENTRY/EXIT                            SAMPLE     134
   274  01000004C2                       RECALL                                                        SAMPLE     135
   275  01000004C2                       RECALL                                                        SAMPLE     136
   276  01000004C2                       RECALL                                                        SAMPLE     137
   277  7160241115                       RTIME   INID            DUMP TIME                             SAMPLE     138
   302  6160000344                       WRITEW  M,INID,B1                                             SAMPLE     139
   304  43601                            MX6     1               RESET BUFFER POINTERS                 SAMPLE     140
             5160000122                  SA6     CBFO                                                  SAMPLE     141
   305  5160000127                       SA6     FBFO                                                  SAMPLE     142
```

```
SAMPLE - EVENT TRACE DUMP SUBSYSTEM.                        COMPASS 3.3-433A.    79/03/13. 12.08.12.      PAGE   11
SUBROUTINES.                                                INI

               5160000607             SA6     WBUF                                          SAMPLE     143
 306  7160222302                      SYSTEM  RSB,R,INIA   READ PP BUFFER POINTERS          L419        62
 311  7160222302                      SYSTEM  RSB,R,INIB   READ CP BUFFER POINTERS          L419        63
 314  7160222302                      SYSTEM  RSB,R,INIC   READ FIRST BLOCK OF LOW CORE     L419        64
 317  5110000122                      SA1     CBFO         SPLIT PP POINTERS                L419        65
            43014                     MX0     12                                            L419        66
                 73710                SX7     X1           FWA                              L419        67
 320  11E01                           BX6     X0*X1                                         L419        68
          5170000124                  SA7     CBFW                                          L419        69
                 20614                LX6     12           INDEX                            L419        70
 321  5120000137                      SA2     FBFO         SPLIT CP POINTERS                L419        71
            21606                     AX6     6            ROUND INDEX TO LOWER MULTIPLE OF 100B  L419  72
                 73720                SX7     X2           CP INDEX                         L419        73
 322  20606                           LX6     6                                             L419        74
          21706                       AX7     6            ROUND CP INDEX DOWNWARD          L419        75
               5061000000             SA6     A1+                                           L419        76
 323  20706                           LX7     6                                             L419        77
          10622                       BX6     X2                                            L419        78
               5072000000             SA7     A2+                                           L419        79
 324  21622                           AX6     18           CP FWA                           L419        80
          5160000141                  SA6     FBFW                                          L419        81
                 43060                MX0     -12                                           SAMPLE     160
 325  5110000614                      SA1     WBUF+ESTP    READ EST POINTER                 SAMPLE     161
            20130                     LX1     24                                            SAMPLE     162
                 15510                BX5     -X0*X1                                        SAMPLE     163
 326  7100000100                      SX0     WBFL                                          SAMPLE     164
 327  6160000607             INI1     WRITEW  M,WBUF,WBFL                                   SAMPLE     165
 331  43601                           MX6     1                                             SAMPLE     166
          37205                       IX2     X0-X5                                         SAMPLE     167
               0322000271             PL      X2,INI2      IF ALL DONE                      SAMPLE     168
 332  5160000607                      SA6     WBUF                                          SAMPLE     169
            10600                     BX6     X0                                            SAMPLE     170
 333  5110000343                      SA1     INIC         FETCH SKELETON                   SAMPLE     171
            20622                     LX6     18                                            SAMPLE     172
                 12661                BX6     X6+X1        SET ABSREAD FUNCTION             SAMPLE     173
 334  5160000344                      SA6     INIO                                          SAMPLE     174
               7160222302             SYSTEM  RSB,WAIT,INIO         READ NEXT BLOCK         SAMPLE     175
 340  7200000100                      SX0     X0+WBFL                                       SAMPLE     176
               0400000327             EQ      INI1         CONTINUE DUMP                    SAMPLE     177
                                                                                            SAMPLE     178
 341  00000001000016000122   INIA     VFD     12/0,12/1,18//ETD/YPBW,18/CBFO                L419        82
 342  00000001000017000137   INIB     VFD     12/0,12/1,18//ETD/YCBW,18/FBFO                L419        83
 343  00001000000000000607   INIC     VFD     12/,12/WBFL,18/0,18/WBUF                      SAMPLE     181
 344  00000000000000000000   INIO     CON     0                                             SAMPLE     182


                          **          PRS - PRESET SAMPLE.                                  L419M1      20
                          *                                                                 SAMPLE     187
                                                                                            SAMPLE     188
                                                                                            SAMPLE     189
 345  00000000000000000000   PRS      SUBR                 ENTRY/EXIT                       L419M1      21
 346  7110000363                      GETJO   PRSA         CHECK FOR SYSTEM ORIGIN          L419M1      22
 350  5110000363                      SA1     PRSA                                          L419M1      23
               0301000355             ZR      X1,PRS1      IF SYSTEM ORIGIN CALL            L419M1      24
                                                                                            L419M1      25
```

```
  351  7110000004                    MESSAGE (=C* ILLEGAL CONTROL CARD.*)                    L419M1   26
  353  7160041121                    ABORT                                                   L419M1   27
                                                                                            L419M1   28
  355  7160241115          PRS1      TIME   PRSA       SET STARTING CPU TIME                L419M1   29
  360  76110               .         OFFSN  1          ENSURE SENSE SWITCH 1 IN RUN POSITION L419M1   30
  362  0400000345                    EQ     PRSX       RETURN                               L419M1   31
                                                                                            L419M1   32
  363                 1    PRSA      BSS    1          O.S. CALL PARAMETER WORD             L419M1   33
                                                                                            SAMPLE   192
```

```
                                **      COMMON DECKS.                                                    SAMPLE    194
                                                                                                         SAMPLE    195
  364                                   CTEXT  COMCCIO - I/O FUNCTION PROCESSOR.                         COMCCIO     1
  376                                   CTEXT  COMCCDD - CONSTANT TO DECIMAL DISPLAY CODE CONVERSION.    COMCCDD     1
  410                                   CTEXT  COMCCFD - CONSTANT TO F10.3 CONVERSION.                   COMCCFD     1
  440                                   CTEXT  COMCCPM - CONTROL POINT MANAGER PROCESSOR.                COMCCPM     1
  444                                   CTEXT  COMCSYS - PROCESS SYSTEM REQUEST.                         COMCSYS     1
  476                                   CTEXT  COMCWTW - WRITE WORDS FROM WORKING BUFFER.                COMCWTW     1
                                                                                                         SAMPLE    200
                                                                                                         SAMPLE    201
                                        USE    BUFFERS                                                   SAMPLE    202
  607                  101    WBUF      BSS    WBFL+1                                                    SAMPLE    203
                       710    MBUF      EQU    *                                                         SAMPLE    204
  710                 2400    CBUF      BSS    CBUFL                                                     SAMPLE    205
 3310                 3441    FBUF      BSS    FBUFL+410                                                 SAMPLE    206
                      6751    RFL=      EQU    *                                                         L419       85


 6751                                   END                                                              SAMPLE    207

                  461000 CM  STORAGE USED          3850 STATEMENTS     504 SYMBOLS     000010 INVENTED SYMBOLS
                             MODEL 73 ASSEMBLY    11.006 SECONDS      222 REFERENCES
```

APPENDIX C

This appendix is a copy of all the modifications done to the operating system to gather the information needed in our study.

```
*IDENT    *ETD*  GEM.        77/05/09.
*/        ****   NOS 1.1 LEVEL 419 O.S.
*/        *****  IMPLEMENT EVENT TRACE DUMP FEATURE.
*/        AFFECTS THE FOLLOWING SYSTEM PROGRAMS -
*/        SET - INITIALIZE TRACE BUFFERS IN LOW CORE.
*/        MTR - DUMP TRACE DATA ON SELECTED FUNCTIONS.
*/        CPUMTR - DUMP TRACE DATA ON SELECTED OCCURRENCES.
*/
*DECK     SET
*I,55
*CALL COMSETD
          TITLE
*I,4185
          RJM    ITB          INITIALIZE TRACE BUFFERS
*I,4610
.ITB      SPACE  4,8
**        ITB - INITIALIZE TRACE BUFFERS.
*
*         ENTRY/EXIT - (CA - CA+1) = NEXT CM ADDRESS.
*
*         USES   CM - CM+4.
*
*         CALLS  NONE.


 ITB      SUBR                ENTRY/EXIT
          LDN    0            SET PP BUFFER POINTERS
          STD    CM           INDEX
          STD    CM+1         UNUSED
          STD    CM+2
          LDD    CA
          STD    CM+3         FWA BUFFER
          LDD    CA+1
          STD    CM+4
          LDN    /ETD/YPBW    STORE POINTER WORD
          CWD    CM
          LCD    CA           BUMP NEXT ADDRESS
          SHN    14
          ADD    CA+1
          ADC    /ETD/YPBL    BUFFER LENGTH
          STD    CA+1
          SHN    -14
          STD    CA
          SHN    14           SET CP BUFFER POINTERS
          ADD    CA+1
          SHN    -6           SET UPPER 12 BITS
          STD    CM+2
          LDD    CA+1
          LPN    77           SET BOTTOM 6 OF FWA/TOP 6 OF INDEX
          SHN    6
          STD    CM+3
          LDN    0
          STD    CM+4
          LDN    /ETD/YCBW    STORE IN CENTRAL MEMORY
          CWD    CM
          LCD    CA           INCREMENT NEXT CM ADDRESS
          SHN    14
          ADD    CA+1
          ADC    /ETD/YCBL    BY CP BUFFER LENGTH
```

```
          STD     CA+1
          SHN     -14
          STD     CA
          LJM     ITBX            RETURN
*/
*/
*/        *** HTR ***
*/
*DECK     HTR
*D,4
          SST     ON,HN,TH,TR,IA,OA,MA,CP,PPR,PPRA,FTN
*I,32
*CALL COMSETD
*/ REARRANGE DIRECT CELLS TO FACILITATE TRACE DUMPS
*I,84
 TD       EQU     *-2             TRACE DATUM (LEAST 24 BITS OF CLOCK)
 FN       CON     0               CURRENT FUNCTION NUMBER FOR TRACE
 CA       CON     0               CHANNEL OR OTHER PARM TO DUMP
 OA       CON     0               CURRENT PP OUTPUT REGISTER ADDRESS
*D,96
*D,98
*D,103
*I,106
 SE       CON     0               SECOND COUNT
 HN       CON     100             CONSTANT 100
*I,187

*         FOLLOWING LOOP ALSO PRESETS TRACE TABLES

*D,254
*         LJM     HTR             ENTER MAIN LOOP
*D,477
          SEM     SE
*D,483
          STM     SE
*D,489
          LDM     HN
*D,520
          LDM     HN              ADVANCE HOURS
*D,552
 AVT5     LDM     HN              ADVANCE DAYS
*D,566
          LDM     HN              RESET DAYS
*D,588
          LDM     HN              ADVANCE YEAR
*D,608
          LJM     AVT10
*D,897
 PPR1     LDD     OR              SAVE FUNCTION CODE
          STD     FN
          LDM     TPPR,OR         SET REQUEST PROCESSOR
*D,919
 FNR      LDC     0               NON-ZERO IF TRACE REQUESTED
 FNR8     EQU     *+1             (SET BY PP REQUEST PROCESSORS)
          ZJN     FNR0            IF NOT TO TRACE
          RJM     OPF             DUMP TRACE DATA

 FNR0     RJM     CCP             CHECK CENTRAL PROGRAM
*I,1025
```

```
OPF     SPACE  4,8
**      OPF - DUMP PP FUNCTION TRACE DATA.
*
*       ENTRY  (TO-OA) = DATA TO DUMP.
*              (OPFA-OPFA+4) = CURRENT BUFFER POINTER (YPBW).
*
*       EXIT   (OPFA) UPDATED.
*              (FNRB) = 0

OPF     SUBR               ENTRY/EXIT
        ACM    OPFA        INCREMENT BUFFER INDEX
        LFC    /ETO/YPBL-1 RESET IF OVERFLOW
        STM    OPFA
        ADC    *           ADD BUFFER FWA
OPFB    EQU    *-2         (ADJUSTED BY PRS)
        CWD    TO          DUMP DATA

        LDM    /ETO/YPBW   UPDATE POINTER
        CWM    OPFA,ON

        LDN    0           CLEAR REQUEST FLAG
        STM    FNRB
        UJN    OPFX        RETURN

OPFA    VFD    60/0        CURRENT POINTER WORD (SET BY PRS)
*I,1184
        STD    CA          SAVE FOR TRACE
        ACM    FNRB        SET TRACE REQUIRED
*I,1207
        STD    CA          SAVE FOR TRACE
*I,1229
        ACM    FNRB        REQUEST TRACE DUMP
*/
*/      DEPP - DUMP DROP CHANNEL
*/
*I,1304
        STD    CA          SAVE CHANNEL FOR TRACE
*I,1306
        ACM    FNRB        REQUEST TRACE
*/
*/      DSWM - DUMP COMPLETION OF WAIT.
*/
*D,1990
DSWM    LDD    OR+1        SET TRACE DATA
        STD    CA          SAVE CHANNEL NO. AND WAIT TYPE
        LDD    OR          SET *DSWM* FUNCTION
        STD    FN
        ACM    FNRB        REQUEST DUMP

        LDD    OA
*/
*/      RCHM - DUMP FIRST AND LAST OCCURRENCES ONLY.
*/
*I,2160
RCH2    LDM    TRCH,T1     CHECK REQUEST TABLE
        NJN    RCH4        IF ALREADY DUMPED
        LDC    4000
        RAD    FN          SET TRACE/GET FNC. CODE
        STM    TRCH,T1     SET REQUEST ALREADY DUMPED
```

```
                LDD     OR+1            SET CHANNELS REQUESTED
                STD     CA
 RCH3           ADM     FNRB            REQUEST DUMP

 RCH4           ACD     CF              SET CHANNEL TABLE WRITE FLAG
                LJM     FNR             EXIT
*I,2171
                LDD     OA              SET PP ORDINAL IN T1
                SBD     OF
                SHN     -3
                STD     T1
*D,2178
 RCH0           ZJN     RCH2            IF NOT SPECIFIED
*D,2183
                ZJN     RCH0            IF NOT SPECIFIED
*D,2188
                ZJN     RCH0            IF NOT SPECIFIED
*D,2190
                ZJN     RCH1            IF NOT BUSY
                LJM     RCH2

*I,2191
                STM     TRCH,T1         CLEAR REQUEST ENTRY
*I,2193
                STD     CA              SAVE FOR TRACE
*D,2196,2197
                LCD     T1              SET CHANNEL BUSY
*D,2199,2200
                LJM     RCH3
*/
*/              RSTM - DUMP STORAGE MOVES/REQUESTS
*/
*D,2463
                LDD     OA              CHECK IF TRACE NEEDED
                ADD     OR+1
                STD     T7
                LCD     T7              TRUNCATE
                LMM     TRST,T6         INDEXED BY CP NUMBER
                ZJN     RSTB            IF REPEAT
                LDD     T7              SET REQUEST ENTRY
                STM     TRST,T6
                LDD     OR+1            SET FL REQUESTED
                STD     CA
                RJM     DPF             DUMP DATUM

 RSTB           LDD     OR+1            CHECK REQUEST
*I,2506
                RJM     DMT             DUMP MOVE TRACE
*/
*/              RSYM - DUMP FIRST AND LAST OCCURRENCES
*/
*I,2547
                STD     CA              SAVE FOR POSSIBLE TRACE
*D,2557

 RSY7           LDM     TRCH,T4         CHECK IF TRACE NEEDED
                NJN     RSY7.1          IF ALREADY DUMPED
                LDC     4000            SET TABLE ENTRY
                RAD     FN              AND RS BIT IN TRACE
```

```
                STM     TRCH,T4
                AOM     FNRB            REQUEST DUMP

 RSY7.1         LJM     FNR             EXIT UNSATISFIED
*I,2566
                LDD     QA              SET PP ORDINAL IN T4
                SBD     OF
                SHN     -3
                STD     T4
*D,2570,2571
                ZJN     RSY8            IF NOT ALTERNATE DEVICE
                LJM     RSY6

 RSY8           STD     T3
*D,2594
                LJM     RSY7            IF NOTHING AVAILABLE
*I,2602
                STD     CA              SAVE FOR TRACE
                LDN     0               CLEAR REQUEST TABLE ENTRY
                STM     TRCH,T4
                AOM     FNRB            SET TRACE REQUESTED
*D,2605,2606
                LDD     T4              SET CHANNEL BUSY
*/
*/              DUMP STORAGE MOVE STATISTICS
*/
*I,3779
                RJM     DMT             DUMP MOVE TRACE DATUM
*I,3801
                RJM     DMT             DUMP MOVE TRACE DATUM
*I,3802
 DMT            SPACE   4,8
**              DMT - DUMP MOVE TRACE DATUM.
*
*               ENTRY   (MM - MM+4) STORAGE MOVE INFORMATION.


 DMT            SUBR                    ENTRY/EXIT
                LDD     MM              SET CONTROL POINT NUMBER
                SHN     -7
                ADC     2000            FLAG AS STORAGE MOVE DATUM
                STD     FN
                LDD     MM+1            SET MOVE BLOCK LENGTH
                STD     CA
                RJM     DPF             DUMP DATA TO BUFFER
                UJN     DMTX            RETURN
*D,3881
 INV6           RJM     DMT             DUMP MOVE DATUM-
                LDN     CMCL            STORE CONTROL WORD
*/
*/              TABLE MODS.
*/
*I,4422
 TRCH           SPACE   4,8
**              TRCH - TABLE OF PP CHANNEL REQUESTS.
*               ENTRY = 1 WORD.
*               INDEXED BY PP ORDINAL.
*
*T,             6/408,6/FN
```

```
*          FN      PP FUNCTION NUMBER OF OUTSTANDING REQUEST
*          12/0    IF NO OUTSTANDING REQUEST.


 TRCH      EQU     TUFL+40
 TRST      SPACE   4,8
**         TRST - TABLE OF PP STORAGE REQUESTS.
*          ENTRY = 1 WORD.
*          INDEXED BY CONTROL POINT NUMBER.
*
*T,        12/(DA)+(OR+1)
*          (DA)    PP OR ADDRESS.
*          (OR+1)  FL REQUESTED.


 TRST      EQU     TRCH+24
*D,4427
 TEVT      EQU     TRST+40
*/
*/         PRESET MODS.
*/
*D,4437
 PRS       LDN     /ETD/YPBW       READ PP BUFFER POINTER WORD
           CRM     DPFA,ON         STORE IN DPP
           LDM     DPFA+3          GET BUFFER FWA
           LPN     77
           ACC     ADDI            SET INSTRUCTION IN DPP
           STM     DPFB
           LDM     DPFA+4
           STM     DPFB+1

           LDN     NCPL            READ NUMBER OF CONTROL POINTS
*D,4641
           STM     SE
*/
*/
*/         ***** CPUMTR - ADD TRACE FEATURES.
*/
*DECK      CPUMTR
*/
*I,41
*CALL      COMSETD
*/
*/         MOD APP TO DUMP PP ASSIGNMENTS
*/
*D,864


*          DUMP PP ASSIGNMENT TRACE DATUM

           BX2     -X4             CHECK IF USER-ORIGIN REQUEST
           SX2     X2+B2
           ZR      X4,APP4         IF CONTROL POINT ZERO
           NZ      X2,APP4         IF NOT USER CALL
           SX7     0               ENSURE NO FPR FIELD
           SA2     X4+/ETD/YPWT    GET TIME OF REQUEST
           SA7     APPA
           SA7     A2              CLEAR TIME
           SA7     A7-B1           CLEAR REQUEST
```

```
APP1    MX7     -59             ENSURE UPPER BIT CLEAR IN DATUM
        BX6     -X7*X6
        AX7     23
        BX6     X7*X6           GET PPNAME, CP NUMBER AND RECALL BIT
        BX6     X4+X6           SAVE CPA TEMPORARILY
        BX4     X2
        SA2     RTCL
        ZR      X4,APP2         IF NO WAIT FOR PP
        IX2     X2-X4           CALCULATE DELAY

APP2    AX7     24
        BX2     -X7*X2          12-BIT DELAY TIME
        LX2     24
        BX6     X2+X6           ADD WAIT TIME
        MX7     0
        SA2     APPA            GET PPR (IF RFPM ONLY)
        SA7     A2+
        TX7     X1,-FP          GET PP ORDINAL ASSIGNED
        AX7     3
        BX7     X2+X7           SET PPR, PPA
        LX7     24
        SA2     X6+JNMW         GET JOB NAME AND ORIGIN
        MX4     24
        LX2     24
        BX2     X4*X2
        MX4     -59             ENSURE UPPER BIT CLEAR
        BX2     -X4*X2
        BX7     X2+X7
        SA2     X6+CPTW         GET JOB CPU TIME
        MX4     -24
        BX2     -X4*X2          USE LOW 24 BITS
        BX7     X2+X7
        SX2     76000B          CLEAR SAVED CPA
        BX6     -X2*X6
        SA2     RTCL            ADD REAL TIME CLOCK
        BX2     -X4*X2
        BX6     X2+X6

        SA2     /ETD/YCBW       GET BUFFER POINTER WORD
        SX4     B1+B1           INCREMENT BUFFER INDEX
        TA6     X2,TBFW         DUMP DATUM
        TA7     X2+1,TBFW
        IX7     X2+X4
        SX4     X7-/ETD/YCBL+2      CHECK INDEX OVERFLOW
        NG      X4,APP3         IF NOT AT LIMIT
        MX6     42              RESET INDEX
        BX7     X6*X7
APP3    SA7     A2              UPDATE INDEX WORD
        JP      B3              EXIT

*       FIND REQUEST TIME FOR NON-CPU/SYSTEM CALL

APP4    SA2     APPB+1          FAKE IT FOR NOW
        SX7     0               CLEAR WAIT TIME
        SA7     A2
        SA7     A7-B1           CLEAR REQUEST
        EQ      APP1

APPA    CON     0               PP ORDINAL OF REQUESTING PP (IF ANY)
```

```
 APPB     CON    0,0          FAKE WAIT TIME ENTRY
*/
*/       SFL
*/
*0,1588
*        DUMP TRACE DATUM
*0,1590
 SFL     SA1    87+JNMW      SET JOB NAME AND ORIGIN
         MX2    24
         LX1    24
         MX6    1
         BX7    X2*X1
         SX4    87           SET CP NUMBER
         BX7    -X6*X7       ENSURE BIT 59 CLEAR
         AX4    7            DIVIDE BY 2000
         LX6    43           SET DATA CODE = 1
         LX4    36
         SA1    RTCL         SET REAL-TIME
         AX2    12           EXTEND MASK
         BX6    X4+X6
         BX1    -X2*X1
         SA3    87+CPTM      SET CPU TIME
         BX6    X1+X6
         SA4    86+2         GET OLD FL
         BX1    -X2*X3
         AX4    36           RIGHT-ADJUST
         BX7    X1+X7
         IX3    X4+X0        GET NEW FL
         SA1    /ETD/YCBW    GET BUFFER INDEX
         LX4    18           POSITION OLD FL
         LX3    18           POSITION NEW FL
         SX2    2
         BX6    X4+X6
         BX7    X3+X7
         TA6    X1,TBFW      DUMP DATUM
         TA7    X1+1,TBFW
         IX6    X2+X1        UPDATE INDEX
         SX4    X6-/ETD/YCBL+2    CHECK INDEX OVERFLOW
         NG     X4,SFL0      IF NOT AT LIMIT
         MX7    42           RESET INDEX
         BX6    X7*X6
 SFL0    SA6    A1+          STORE INDEX WORD

*        UPDATE EXCHANGE PACKAGE AND CP STATUS

         SA4    86+2         READ EXCHANGE PACKAGE FL
*/
*/       APJ - SET PP REQUEST TIME
*/
*0,2023
 APJ1    SA1    87+/ETD/YPRQ      GET LAST REQUEST
         BX2    X5-X1        CHECK IF REPEATED REQUEST
         NO
         ZR     X2,APJ1.1    IF REPEAT
         SA2    RTCL         SAVE REQUEST AND TIME
         BX7    X5           IN CONTROL POINT AREA
         SA7    87+/ETD/YPRQ
         BX7    X2
         SA7    87+/ETD/YPWT
```

```
 APJ1.1   SA1    NP
*/
*/       MRAF - DUMP CHANGE OF RA.
*/
*I,3440

*        DUMP TRACE DATUM

         SA5    87+JNHW      SET JOB NAME AND ORIGIN
         MX2    24
         LX5    24
         MX6    1
         BX7    X2*X5
         SX4    87           SET CP NUMBER
         BX7    -X6*X7       ENSURE BIT 59 CLAER
         AX4    7            DIVIDE BY 2000
         LX6    43+1         DATA CODE = 2
         LX4    36
         SA5    RTCL         SET REAL TIME
         AX2    12           EXTEND MASK
         BX6    X4+X6
         BX5    -X2*X5
         SA3    87+CPTW      SET CPU TIME
         BX6    X5+X6
         SB4    36+6         SHIFT COUNT
         BX5    -X2*X3
         AX4    X1,B4        RIGHT-ADJUST RA/1000
         BX7    X5+X7
         IX3    X4+X0        GET NEW RA/1000
         SA5    /ETO/YCBW    GET BUFFER INDEX
         LX4    24           ADD OLD RA
         LX3    24           ADD NEW RA
         SX2    2
         BX6    X4+X6
         BX7    X3+X7
         TA6    X5,TBFW      DUMP DATUM
         TA7    X5+1,TBFW
         IX6    X2+X5        UPDATE INDEX
         SX4    X6-/ETO/YCBL+2    CHECK FOR OVERFLOW
         NG     X4,MRA1      IF NOT AT LIMIT
         MX7    42
         BX6    X7*X6
 MRA1    SA6    A5+          STORE INDEX

*        PROCESS REQUEST

*/
*/       CCAM DUMPS
*/
*I,3986
*
*        CALLS  OPT.
*I,3992
         BX6    X2           SAVE OLD INPUT REGISTER
         LX6    36-7         IN ORIGINAL POSITION
         SA6    CCAA
*D,4034
         BX5    X6
```

```
                SA4     CCAA            DUMP DROP TRACE
                MX6     1               SET DATA FLAGS FOR OPT
                MX7     1
                RJ      OPT
                SA4     A5-1            GET NEW INPUT REGISTER
                MX6     0               DUMP PP RE-ASSIGN
                MX7     1
                RJ      OPT
                LX5     59-58
*0,4036
                PL      X5,JAV          IF NOT *X* STATUS
*I,4048

 CCAA   CON     0               OLD INPUT REGISTER
*/
*/      DCPM - DUMP DROP PP ON ABTM CALL
*/
*I,4173
                SA4     A5-1            DUMP TRACE OF DROP
                MX6     1               SET DATA TYPE FLAGS FOR OPT
                MX7     0
                RJ      OPT
*/
*/      DPPM - DUMP DROP PP
*/
*I,4252
                SA4     A5-1            DUMP TRACE OF DROP
                MX6     1
                MX7     0
                RJ      OPT
*/
*/      JACM - DUMP DROP PP (OPTION 3 ONLY)
*/
*/      ALSO MOD JACM TO CLEAR WORD CPTW OF CONTROL POINT AREA WHEN
*/      OPTION 1 OR 3 IN EFFECT
*/
*I,4529
*       CPTW IS ALSO CLEARED
*I,4587
                SA7     B7+CPTW         CLEAR CP ACCUMULATOR
*I,4591
                BX1     X6              DUMP DROP PP DATUM
                SA4     A5-B1
                MX6     1
                MX7     0
                RJ      OPT
*D,4596
                IX6     X1-X5           DECREMENT PPU COUNT
*/
*/      LOAM - DUMP CALLS FROM 60I
*/
*D,4694

                MX2     6               DUMP TRACE DATUM
                SA1     A6+B1           GET (MB + 1)
                SA7     A5+             STORE PP OUTPUT REGISTER
                SA3     RTCL            GET TIME OF STORE OF (QR)
                LX6     6               GET 6-BIT FLAGS
                LX1     12
```

```
          BX5    X6*X2
          BX7    X1              CH, EQ, LT, LS, LU
          TX4    A5-1,-FP        SET PP ORDINAL
          LX5    47-59           POSITION FLAGS
          SX6    3               DATA CODE = 3
          LX4    58-4-3          POSITION PP ORDINAL
          BX7    X5+X7
          BX7    X4+X7
          LX6    59-17
          SA4    A5-B1           GET CP NUMBER FROM (IR)
          MX0    -24
          LX2    41-59
          BX3    -X0*X3
          BX4    X2*X4           ISOLATE CP NUMBER, RECALL BIT
          BX6    X3+X6
          SA1    /ETD/YCBW       FETCH BUFFER INDEX
          BX6    X4+X6
          SX0    2
          TA7    X1+1,TBFW       DUMP DATUM
          TA6    X1,TBFW
          IX7    X0+X1           INCREMENT INDEX
          SX4    X7-/ETD/YCBL+2    CHECK INDEX OVERFLOW
          NG     X4,ADI1         IF NOT AT LIMIT
          MX6    42              RESET INDEX
          BX7    X6*X7
ADI1      SA7    A1              UPDATE INDEX
          JP     PPRX            EXIT
*/
*/        RPPM - SET ORDINAL OF REQUESTING PP FOR APP
*/
*D,5839
          NZ     X1,RPP2         IF PP AVAILABLE
*I,5846
RPP2      TX7    A5-1,-FP        SAVE PP ORDINAL FOR APP TRACE
          LX7    3               IN FORMAT 48/0,6/PP+400,6/0
          SX7    X7+40000B
          SA7    APPA
          JP     APP             ASSIGN PP AND RETURN
*/
*/        SPLM - DUMP SEARCH REQUEST
*/
*D,5181
          SB6    SPL10           *SPL* RETURN ADDRESS
*I,5183
*         DUMP TRACE DATUM

SPL10     MX0    18              SAVE PACKAGE NAME
          BX1    X0*X6
          BX2    X0*X7           SAVE RESIDENCY
          SA3    A5-B1           READ INPUT REGISTER
          SA4    RTCL            GET REAL TIME
          SA7    A5+             STORE OUTPUT REGISTER
          SX6    4               DATA CODE = 4
          TX5    A3,-FP          SET PP ORDINAL
          LX6    53-11
          LX5    35-11-3
          LX1    17-59
```

```
          AX0     18
          BX7     X1
          LX2     23-47
          BX4     -X0*X4
          BX7     X2+X7
          MX0     6
          BX6     X4+X6
          LX0     41-59
          BX6     X5+X6
          BX3     X0*X3          ISOLATE RECALL BIT AND CP NUMBER
          SA1     /ETD/YCBW      FETCH BUFFER INDEX
          BX6     X3+X6
          SX0     2
          TA7     X1+1,T8FW      DUMP DATUM
          TA6     X1,T8FW
          IX7     X1+X0          INCREMENT INDEX
          SX4     X7-/ETD/YC8L+2     CHECK INCEX OVERFLOW
          NG      X4,SPL11       IF NOT AT LIMIT
          MX6     42             RESET INDEX
          BX7     X6*X7
SPL11     SA7     A1             UPDATE INDEX WORD
          JP      PPRX           EXIT
*/
*/        TIOM - SAVE PRU COUNT FOR OPT
*/
*I,5526
          BX7     -X4*X1         TAKE LOWER 12 BITS OF MT INCREMENT
          SA7     OPTA           SAVE FOR SUBSEQUENT OPT CALL
*/
*/        ACD SUBROUTINE OPT AND FIX IMPROPER SUBR ORGANIZATION
*/
*D,5720
          TITLE  PPU REQUEST PROCESSING SUBROUTINES.
*D,5731
 OPT      SPACE  4,8
**        OPT - DUMP PP TRACE DATUM.
*
*         DUMPS A DROP PP TRACE DATUM.
*         ALSO USED BY CCAM TO DUMP PP RE-ASSIGN TRACE.
*
*         ENTRY  A5 = OUTPUT REGISTER ADDRESS
*                X4 = 18/PPNAME,1/RECALL,5/CP,36/?
*                X6 = 1/FLAG 1,59/0
*                X7 = 1/FLAG 2,59/0
*                OPTA = 48/0,12/PRUS OR 0
*
*         EXIT   OPTA = 0.
*
*         USES   A - 4, 6, 7.
*                B - 5.
*                X - 0, 4, 6, 7.
*
*         CALLS  NONE.

 OPT      PS                    ENTRY/EXIT
          MX0     5             EXTRACT CP ADDRESS
          LX0     40-59
          BX0     X4*X0
```

```
          LX0     4-40+7
          S85     X0              (85) = CPA
          MX0     37              ISOLATE PPNAME,RECALL BIT AND CP NUMBER
          LX0     36              ENSURE NAME BIT 17 = 0
          BX4     -X0*X4
          BX6     X4+X6
          SA4     RTCL            SET REAL-TIME
          MX0     -24
          BX4     -X0*X4
          BX6     X4+X6
          SA4     85+CPTW         SET JOB CPU TIME
          BX4     -X0*X4
          BX7     X4+X7
          SA4     OPTA            ADD PRUS (IF ANY)
          LX4     12
          BX6     X4+X6
          SA4     85+JNMW         SET JOB NAME AND ORIGIN
          MX0     24
          LX4     59-35
          BX4     X0*X4
          BX7     X4+X7
          TX4     A5-1,-FP        SET PP ORDINAL
          LX4     24-3
          BX7     X4+X7
          SA4     /ETD/YCBW       FETCH BUFFER INDEX
          SX0     2
          TA6     X4,TBFW         DUMP DATUM
          TA7     X4+1,TBFW
          IX6     X4+X0           INCREMENT INDEX
          BX7     X7-X7           CLEAR PRUS COUNT
          SX4     X6-/ETD/YCBL+2  CHECK INDEX OVERFLOW
          SA7     OPTA
          NG      X4,OPT1         IF NOT AT LIMIT
          MX7     42              RESET INDEX
          BX6     X7*X6

OPT1      SA6     A4              UPDATE INDEX WORD
          JP      OPT             RETURN

OPTA      CON     0               PRUS TRANSFERRED COUNT
 HNG      SPACE   4,8
*/
*/        CPUNTR PRESET
*/
*I,9402
*CALL     COMSETO
*I,9571
 TBFW     CON     0               TRACE BUFFER FWA
*I,9886
          SA1     /ETD/YCBW       SET TRACE BUFFER FWA
          AX1     18
          BX6     X1
          SA6     TBFW
*/
*/        END OF MODSET.
*IDENT *ETD1*     JW.             78/05/20.
*/        ****    NOS 1.1 - 419
*/        ****    *ETD1*
*/        *****   CORRECT EVENT TRACE FEATURE.
```

```
*/        CORRECTIONS TO VARIOUS DECKS.
*/
*/        ****    CHECK *SPLH* FOR ONLY A-Z,1,2 LEVELS.
*DECK CPUMTR
*D *ETD*,246                       (5183)
 SPL10    MX0     6
          BX1     X0*X6            ISOLATE FIRST CHARACTER
          LX1     6
          SB3     X1
          SA1     SPLA             MASK A-Z,1,2
          LX1     X1,B3
          SA7     A5
          NG      X1,*+1           TRACE THIS CALL
          JP      PPRX
          MX0     18
*I *ETD*,278
 SPLA     VFD     1/0,26/-0,1/0,1/-0,31/0
*/
*/        ****    CORRECT LOSS OF 1 ST CHAR IN PP NAME.
*DECK CPUMTR
*I 831
          SA6     APPC
*I 851
          SA4     APPC
          ZR      X4,APP0          IF NO NAME STORED
          BX6     X4
 APP0     BSS     0
*I *ETD*,13
          SA7     APPC
*I *ETD*,70
          SA7     APPC
*I *ETD*,74
 APPC     CCN     0
*/
*/        ****    COUNT SECTORS IN CPA.
*DECK COMMMSD
*I 248
          RJM     306              (ADDRESS OF *CST* IN PPR)
*DECK COMSETD
*I 74
*         YPSC - SECTORS TRANSFERRED PER PP CALL
*I 80
 YPSC     EQU     IN4W             SECTOR COUNT
*DECK PPR
*I 36
*CALL COMSETD
*D 246
**        CST - COUNT SECTORS TRANSFERRED.
*
*         ENTRY - NONE.
*         EXIT  - (YPSC) UPDATED.
*                 (A) = 0

 CST      SUBR                     ENTRY/EXIT
          LDD     CP
          ZJN     CSTX             IF CTL PT ZERO
 CSTA     ACN     /ETD/YPSC
          CRD     CM
 CST8     AOD     CM               INCREMENT COUNT
```

```
                LCO     CP
CSTC            AON     /ETO/YPSC
                CWO     CM              RESTORE IN CPA
                LON     0
                UJN     CSTX

                BSSZ    5               COC,S LEFTOVERS
*I 597
                AON     1
                SBN     5
                MJN     *+5             IF IN LOWER 5 PP,S
                AOM     CSTA
                AOM     CSTC
                LOD     T1
                AON     1
                SBN     5
                PJN     *+2
                AON     5
                RAM     CSTB            ADD IN PP INCREMENT (MODULO 5)
                LOD     T1
*DECK CPUMTR
*D *ETO*.189                            (4252)
                MX7     -12
                SX6     A5              OR ADDR OF CALLER
                SA4     PPCP
                BX7     -X7*X4
                SX4     X4+B1
                IX6     X6-X7
                AX6     3               PPU NUMBER
                SB5     B1
                SX6     X6-5
+               PL      X6,*+2          IF UPPER 5 PPU NUMBER
                SB5     0
                SX6     X6+5
+               SB3     B7+/ETO/YPSC            PICK UP COUNTER WORD
                SB3     B3+B5
                SA4     B3
                SB3     B1+B1
                LX7     X6,B3           CALCULATE SHIFT COUNT
                LX6     3
                IX6     X6+X7
                SB3     X6
                LX6     X4,B3
                LX6     12              ROTATE COUNT TO LOWER 12 BITS
                MX7     -12
                BX6     -X7*X6          CLEAN UP THE COUNTER WORD
                SA6     CPTA            PLONK
                BX6     X7*X6
                SB5     60-12
                SB3     B5-B3           ROTATE WORD BACK TO ORIGINAL POSITION
                LX6     X6,B3
                SA6     A4
                SA4     A5-1            SET UP AND TRACE OPPM
*/
*/      FIX GARBAGE WORD IN BUFFER PROBLEM.
*DECK CPUMTR
*MODNAME *ETO*
*D 58,59                                        (863)
                SX4     X7-/ETO/YCBL            CHECK INDEX OVERFLOW
```

```
          NZ      X4,APP3       IF NOT AT LIMIT
                                (1586)
*O 106,107
          SX4     X6-/ETO/YCBL          CHECK INDEX OVERFLOW
          NZ      X4,SFL0       IF NOT AT LIMIT
                                (3440)
*O 159,160
          SX4     X6-/ETO/YCBL          CHECK INDEX OVERFLOW
          NZ      X4,MRA1       IF NOT AT LIMIT
                                (4893)
*O 229,230
          SX4     X7-/ETO/YCBL          CHECK INDEX OVERFLOW
          NZ      X4,ADI1       IF NOT AT LIMIT
                                (5183)
*O 273,274
          SX4     X7-/ETO/YCBL          CHECK INDEX OVERFLOW
          NZ      X4,SPL11      IF NOT AT LIMIT
                                (5758)
*O 337
          SX4     X6-/ETO/YCBL          CHECK INDEX OVERFLOW
                                (5750)
*O 339
          NZ      X4,OPT1       IF NOT AT LIMIT
*/
*/        FIX INCORRECT SHIFT COUNT.
                                (5750)
*O 321
          LX4     24
*/
*/        FIX INVALID CHANNELS IN *RSYM*
*DECK MTR
*I 2567
          LCN     0
          STO     CA            PRESET NO CHANNELS
*/
*/        FLAG WAIT TIME OVERFLOW IN PP REQUESTS
*DECK CPUMTR
*I *ETO*.23                     (863)
          SX7     X2-77768      CHECK UPPER LIMIT ON THE
          NG      X7,APP2       IF NO OVERFLOW
          SX2     77768         SET OVERFLOW NUMBER
*O *ETO*.25
 APP2     MX7     -12
*/        END OF MOOSET.
```

APPENDIX D

This appendix shows examples of output for the PREPROCESSOR program and other utility programs (QUERYFILE and DUMPBINFILE) used during the development of the performance evaluation. These programs have been entirely written by the author in PASCAL 2.2.

The PREPROCESSOR has been summarized in section 3.4.2.3. The following pages are the output for the four different configurations reported in the thesis. The outputs show: the actual name and version of the system, date and time of the test, the hardware mainframe configuration and all the device-types with their channel allocations.

```
KRONOS 2.1  TEST TRACE SYSTEM.          KRONOS 2.1.1-388

     DATE  TIME-OF-DAY   REAL-TIME   TIME-DUMP
 77/01/13.   04.30.12.  0001364202  0001364204

PP  PPCA  CP  CPCA  FEST  LEST  CMRS  MCMS
12  4200  17  0200  6100  6177  0461  3000

A.CH P.CH DEVICE-TYPE
 04   03   0411 DI
 00   10   0423 DS
 00   11   0322 CR
 00   11   0320 CP
 00   11   1421 LQ
 00   11   1421 LQ
 00   01   2424 TT
 00   04   2324 ST
 00   12   1503 MC
 00   13   1524 MT
 00   13   1624 MT
 00   13   5624 MT OFF-LINE
 00   13   1624 MT
 00   13   1624 MT
 00   00   2405 TE
 00   00   1605 NE
```

```
NOS 1.1 EVENT DUMP TEST SYSTEM.          NOS 1.1 419/T.

     DATE  TIME-OF-DAY   REAL-TIME   TIME-DUMP

 77/07/28.   03.12.10.  0003036114  0003036116

PP  PPCA  CP  CPCA  FEST  LEST  CMRS  MCMS

12  4200  17  0200  4400  4477  0537  3000

A.CH P.CH DEVICE-TYPE

 03   02   0411 DI
 02   04   0411 DI
 00   10   0423 DS
 30   12   0322 CR
 00   12   0320 CP
 00   12   5421 LQ OFF-LINE
 00   12   5421 LQ OFF-LINE
 00   13   1524 MT
 00   13   1624 MT
 00   13   1624 MT
 00   13   1624 MT
 00   13   1624 MT
 00   00   2405 TE
 00   00   1605 NE
```

NOS 1.1 EVENT DUMP TEST SYSTEM. NOS 1.1 419/T.

DATE	TIME-OF-DAY	REAL-TIME	TIME-DUMP
77/07/28.	03.34.30.	0000425552	0000425554

PP	PPCA	CP	CPCA	FEST	LEST	CMRS	MCMS
12	4200	17	0200	4400	4477	0537	3000

A.CH	P.CH	DEVICE-TYPE
06	02	0411 DI
02	04	0411 DI
00	10	0423 DS
00	12	0322 CR
00	12	0320 CP
00	12	1421 LQ
00	12	1421 LQ
00	13	1524 MT
00	13	1624 MT
00	13	1624 MT
00	13	1624 MT
00	13	1624 MT
00	00	2405 TE
00	00	1605 NE

NOS 1.1 EVENT DUMP TEST SYSTEM. NOS 1.1 419/T.

DATE	TIME-OF-DAY	REAL-TIME	TIME-DUMP
77/07/28.	03.58.09.	0002214643	0002214645

PP	PPCA	CP	CPCA	FEST	LEST	CMRS	MCMS
12	4200	17	0200	4400	4477	0537	3000

A.CH.	P.CH	DEVICE-TYPE
03	02	0411 DI
02	04	0411 DI
00	10	0423 DS
00	12	0322 CR
00	12	0320 CP
00	12	1421 LQ
00	12	1421 LQ
00	13	1524 MT
00	13	1624 MT
00	13	1624 MT
00	13	1624 MT
00	13	1624 MT
00	00	2405 TE
00	00	1605 NE

```
NOS 1.1 EVENT DUMP TEST SYSTEM.          NCS 1.1 419/T.

    DATE  TIME-OF-DAY   REAL-TIME   TIME-DUMP

 78/07/14.  01.50.58.  0000372100  0000372101

PP  PPCA  CP  CPCA  FEST  LEST  CMRS  MCMS

12  4200  17  0200  4400  4477  0537  4000


A.CH P.CH DEVICE-TYPE

 03   02   0411 DI
 00   04   0411 DI
 00   10   0423 DS
 00   12   0322 CR
 00   12   0320 CP
 00   12   5421 LQ OFF-LINE
 00   12   1421 LQ
 00   13   1524 MT
 00   13   1624 MT
 00   13   5624 MT OFF-LINE
 00   13   1624 MT
 00   13   1624 MT
 00   00   2405 TE
 00   00   1685 NE
```

```
NOS 1.1 EVENT DUMP TEST SYSTEM.          NCS 1.1 419/T.

     DATE   TIME-OF-DAY.   REAL-TIME    TIME-DUMP
 78/08/15.    02.42.06.  0001710712  0001710714

PP  PPCA  CP  CPCA  FEST  LEST  CMRS  MCMS
12  4200  17  0200  4400  4477  0537  4000

A.CH P.CH DEVICE-TYPE
 03   02   0411 DI
 00   04   0411 DI
 00   10   0423 DS
 00   12   0322 CR
 00   12   0320 CP
 00   12   5421 LQ OFF-LINE
 00   12   1421 LQ
 00   13   1524 MT
 00   13   1624 MT
 00   13   1624 MT
 00   13   1624 MT
 00   13   1624 MT
 00   00   2405 TE
 00   00   1605 ME
```

NOS 1.1 EVENT DUMP TEST SYSTEM.. NCS 1.1 419/T.

DATE	TIME-OF-DAY	REAL-TIME	TIME-DUMP
78/08/18.	00.48.25.	0001063565	0001063567

MP	PPCA	CP	CPCA	FEST	LEST	CMRS	MCHS
12	4200	17	0200	4400	4477	0537	4000

A.CH	P.CH	DEVICE-TYPE	
03	02	0411	DI
00	04	0411	DI
00	10	0423	DS
00	12	0322	CR
00	12	0320	CP
30	12	5421	LQ OFF-LINE
00	12	1421	LQ
00	13	1524	MT
00	13	1624	MT
00	13	1624	MT
00	13	1624	MT
00	13	1624	MT
00	00	2405	TE
00	00	1605	NE

```
NOS 1.1 EVENT DUMP TEST SYSTEM.          NOS 1.1 419/T.

     DATE  TIME-OF-DAY   REAL-TIME    TIME-DUMP

 78/08/18.   01.40.03.  0001351055  0001351057

PP  PPCA  CP  CPCA  FEST  LEST  CMRS  MCMS

12  4200  17  0200  4400  4477  0546  4000

A.CH P.CH DEVICE-TYPE

 00   02   0411 DI
 00   04   0411 DI
 00   04   0411 DI
 00   10   0423 DS
 00   12   0322 CR
 00   12   0320 CP
 00   12   5421 LQ OFF-LINE
 00   12   1421 LQ
 00   13   1524 MT
 00   13   1624 MT
 00   13   1624 MT
 00   13   1624 MT
 00   13   1624 MT
 00   00   2405 TE
 00   00   1605 NE
```

NOS 1.1 EVENT DUMP TEST SYSTEM. NOS 1.1 419/T.

DATE	TIME-OF-DAY	REAL-TIME	TIME-DUMP
78/08/18.	01.58.51.	0003732041	0003732043

PP	PPCA	CP	CPCA	FEST	LEST	CMRS	MCMS
12	4200	17	2400	4400	4477	0546	4000

A.CH	P.CH	DEVICE-TYPE	
03	02	0411	DI
00	04	0411	DI
02	04	0411	DI
00	10	0423	DS
00	12	0322	CR
00	12	0320	CP
00	12	5421	LQ OFF-LINE
00	12	1421	LQ
00	13	1524	MT
00	13	1624	MT
00	13	1624	MT
00	13	1624	MT
00	13	1624	MT
00	00	2405	TE
00	00	1605	ME

```
NOS 1.1 EVENT DUMP TEST SYSTEM.          NOS 1.1 419/T.

     DATE  TIME-OF-DAY   REAL-TIME   TIME-DUMP
 78/08/18.   02.15.16.  0000217673  0000217674

PP   PPCA  CP  CPCA  FEST  LEST  CMRS  MCMS
12  4200  17  3000  4400  4477  0537  4000

A.CH P.CH DEVICE-TYPE
 00   02   0411 DI
 00   04   0411 DI
 00   10   0423 DS
 00   12   0322 CR
 00   12   0320 CP
 00   12   5421 LQ OFF-LINE
 00   12   1421 LQ
 00   13   1524 MT
 00   13   1624 MT
 00   13   1624 MT
 00   13   1624 MT
 00   13   1624 MT
 00   00   2405 TE
 00   00   1605 NE
```

The DUMPBINFILE program runs in about 1.6K (3,150 octal) 60-bit words and is only 90 lines of self-documented program. The program dumps a binary file, in octal representation, splitting each word into five bytes of 12 bits each.

On each page a title, with the name of the file dumped, will appear. Each line of the dump is numbered, in octal and decimal, from a starting value given in input. The number of words per line desired must be specified in the input. If this is negative or zero, one is assumed; if greater than four, four is assumed. It is also possible to skip n lines at the beginning of the file (these lines are saved on a file named SCRATCH in case there is a need for, such as: reduce the size of the file for earlier testing); we can specify how many lines we want to print (this number will be rounded up to an end-of-page).

The input specifications are: fln skp sss nnn lll;
where:

- fln = file name (to be dumped),
- sss = starting position,
- skp = skip skp words at the beginning of the file,
- nnn = number of words per line,
- lll = number of line to be printed.

The QUERYFILE program, runs in about 2.6K (5,170 octal) 60-bit words and is a self-documented program of about 570 lines. The program reads the file FDUMP or CDUMP (renamed as BINFILE) and extracts pertinent information according to the input-query specifications.

It has been found the need for such a program when, debugging the analysis program, we were facing an enormous amount of output. Secondly the analysis program could only execute once in a day. Because of its demand of memory and time, the job could run only after 18 o'clock and, even then, we could wait hours to get the output, depending on how busy the system was. So, being frustrated by this situation, it has been decided to write this program which can give answers at any time in a fraction of a second (necessarily this increases depending on the amount of lines to be printed). In Pascal, a binary file of 41,000 60-bit words can be read in as few as 0.6 seconds.

A small query language has then been defined to achieve this (shown in fig. D-1):

```
<query>    --> <mne> <list>
           --> <mne>

<list>     --> <list> <explist>
           --> <unop> NAME
           --> <unop> NUM

<explist>  --> <binop> <list>
           --> null

<unop>     --> -
           --> null

<binop>    --> *
           --> +

<mne>      --> CHFN   (channel and function number, CDUMP)
           --> FN     (function number, CDUMP)
           --> MB     (move block length, CDUMP)
           --> BT1    (byte-1,channel or field length; CDUMP)
           --> CRT    (real-time clock, CDUMP)
           --> LDAM   (compute phys. addr. word, FDUMP)
           --> FL     (change of field length, FDUMP)
           --> W1BT2  (word-1 byte-2, time denied or prus or
                       old FL; FDUMP)
           --> W2BT2  (word-2 byte-2, PP requested and PP or
                       PP or new FL; FDUMP)
           --> CPN    (control point number, FDUMP)
           --> CPPP   (CP number and PP name, FDUMP)
           --> JBN    (job name, FDUMP)
           --> JBPP   (job name and PP name, FDUMP)
           --> PPCP   (PP name and control point, FDUMP)
           --> PPN    (PP name, FDUMP)
           --> PPOT   (PP name and origin type, FDUMP)
           --> RA     (change of reference address, FDUMP)
           --> RES    (PP resident library search, FDUMP)
           --> OTY    (origin type, FDUMP)
           --> RTM    (real-time clock, FDUMP)
```

NAME = PP name or Job name (3 characters),
NUM = control point, origin type, function, time, etc.
can be octal if ending with a $,
- = not,
+ = or,
* = and.

Fig. D-1 Definition of the Query Language

The program has been constructed in a way that new mnemonics can be added at any time, without disturbing the logic of the program, or without reprogramming half of the program. The addition will consist: add the new mnemonic in the mnemonic declaration type, then insert in the printing case statement, the new tests for printing the new line.

This program reads and analyzes the entire query-line. Then, if no errors occur, the evaluation of the query is done per every word (or two if FDUMP) of the file read and a line is printed according to this evaluation. In the query-line, one or more spaces can separate each token of the query-language. The mnemonic (<mne>) determines which field of a word is concerned in the query and determines also which file will be the input-file, thus it is a must. As we have seen from the grammar specification, it is a right-recursive grammar, this means that the last part (<explist>) can be repeated as many times as can be placed on one line.

Some examples of input are:

CHFN 13$ * -4 * -21$ * -12$

which will print each occurrence (in CDUMP) of channel 13 (octal) for any function appearing in the word except 4, 12 and 21 (octal);

CHFN 13$ * 4 + 12$ + 21$

this is the reverse of the previous. It will print each occurrence of channel 13 (octal) only if function 4 or 12 or 21 (octal) is present;

PPN 1SJ + 1AJ + 1RO + 1RI

it will print each occurrence (in FDUMP) where 1SJ or 1AJ or 1RO or 1RI is present;

PPN -1SJ * -1RO * -1AJ * - 1RI

this is the reverse of the previous one. It will print all the occurrences of PP names, except 1SJ, 1AJ, 1RO and 1RI;

FL

will print all occurrences of FL changes;

RA

will print all occurrences of change of reference address.

In the output, as first line, we will see the echo of the query; then, a title corresponding to which file choosen will be printed. The formatted output is of three different types: for CDUMP, one word per line separated into four parts: real-time, function, channel or field length, and the output register address; for FDUMP, two words per line split as: first word-- the flag (drop or assign), the PP program name, control point, field length or PRUs transferred, and real-time; second word-- flag, job name, origin type, PP number, and CPU-time; the last one is for the special case of the driver seek wait words, two words per line separated as: first word-- flag, code and real-time; second word-- output register address, sector, track, device and channel number. At the end of the search, the number of occurrences is also printed.

This program has been found a very useful tool in helping us to discover, in huge binary files, peculiar situations or

unexpected sequence of called PP program names, anomalies in the operating system or the event-trace such as: a PP is assigned to a job but not dropped at the end of the process; strange CPU times when a job enters or ends, in roll-in or roll-out; etc.

In my own opinion, such a program should be a requirement when dealing with enormous files (binary or not). The following pages show some examples of output for the QUERYFILE program.

CPPP 5 *-!!0 *-!!C *-CI0

F	PPN	CP	FL/S	REALTIME	F	JBN	OT	R PP	CPUTIME
0	1SJ	05	0000	01461362	1	!!!	00	0010	00000000
0	!!B	05	3674	01461507	0	!!!	00	3176	00000000
0	!!A	05	0030	01461507	0	!!!	00	0500	00000000
0	1AJ	05	1510	01461510	0	AAP	01	5006	00000000
1	1SJ	05	0000	01461510	1	AAP	01	0010	00000000
0	!!A	05	0500	01461547	0	AAP	01	0007	00000000
1	1AJ	05	0012	01462306	0	AAP	01	0006	00000000
0	CPM	45	0001	01462307	0	AAP	01	0006	00000000
1	CPM	45	0010	01462511	0	AAP	01	0006	00000002
0	CPM	45	0001	01462512	0	AAP	01	0006	00000002
1	CPM	45	0016	01463163	0	AAP	01	0006	00000003
0	1MA	45	0000	01463164	0	AAP	01	0006	00000003
1	1MA	45	0000	01463166	0	AAP	01	0006	00000003
0	1MA	45	0000	01463166	0	AAP	01	0006	00000003
1	1MA	45	0000	01463171	0	AAP	01	0006	00000003
0	1AJ	05	3172	01463172	0	AAP	01	0004	00000003
0	!!A	05	0007	01463213	0	AAP	01	0500	00000003
1	1AJ	05	0032	01463376	0	AAP	01	0004	00000003
0	LDR	45	0000	01463403	0	AAP	01	0004	00000004
1	LDR	45	0025	01463606	0	AAP	01	0004	00000005
0	RPV	45	0001	01463607	0	AAP	01	0004	00000005
1	RPV	45	0000	01463611	0	AAP	01	0004	00000006
0	LDR	45	0000	01463650	0	AAP	01	0002	00000011
1	LDR	45	0333	01464451	0	AAP	01	0002	00000012
0	!!B	05	3176	01465523	0	AAP	01	2276	00000454
0	LDR	45	0001	01465645	0	AAP	01	0006	00000517
1	LDR	45	0271	01466632	0	AAP	01	0006	00000520
0	LDR	45	0000	01467202	0	AAP	01	0006	00001033
1	LDR	45	0061	01467671	0	AAP	01	0006	00001036
0	1MA	45	0001	01470715	0	AAP	01	0002	00001236
1	1MA	45	0000	01470716	0	AAP	01	0002	00001237
0	1AJ	05	0717	01470717	0	AAP	01	0002	00001237
1	1AJ	05	0075	01472512	0	AAP	01	0002	00001241
0	LFM	45	0000	01472514	0	AAP	01	0002	00001241
1	LFM	45	0000	01472516	0	AAP	01	0002	00001245
0	LFM	45	0000	01472610	0	AAP	01	0002	00001270
1	LFM	45	0000	01472620	0	AAP	01	0002	00001273
0	LFM	45	0000	01477304	0	AAP	01	0007	00001737
1	LFM	45	0000	01477315	0	AAP	01	0007	00001741
0	!!B	05	2276	01500407	0	AAP	01	1576	00001742
0	!!B	05	1576	01501160	0	AAP	01	1567	00001762
0	!!B	05	1567	01503456	0	AAP	01	1541	00002137
0	LDR	45	0000	01510025	0	AAP	01	0010	00002704
1	LDR	45	0025	01510132	0	AAP	01	0010	00002705
0	!!A	05	0500	01511470	0	AAP	01	0172	00003120
0	1MA	45	0000	01512525	0	AAP	01	0006	00003251
1	1MA	45	0000	01512527	0	AAP	01	0006	00003251
0	1MA	45	0001	01512545	0	AAP	01	0003	00003251
1	1MA	45	0000	01512546	0	AAP	01	0003	00003252
0	1AJ	05	2547	01512547	0	AAP	01	0003	00003252
0	!!A	05	0172	01512551	0	AAP	01	0000	00003253
1	1CJ	05	0000	01514253	0	!!!	00	0003	00000000
0	1SJ	05	0000	01514431	1	!!!	00	0003	00000000
0	!!A	05	0000	01515005	0	!!!	00	0600	00000000
0	1AJ	05	5006	01515006	0	AAV	01	4304	00000000
1	1SJ	05	0000	01515006	1	AAV	01	0003	00000000

```
0 ::A 05 3600 01515036   0 AAV 01 0007 00000000
1 1AJ 05 0012 01515264   0 AAV 01 0004 00000000
0 CPM 45 1001 01515312   0 AAV 01 0005 00000000
1 CPM 45 0010 01515426   0 AAV 01 0005 00000002
0 CPM 45 0000 01515426   0 AAV 01 0005 00000002
1 CPM 45 0016 01516635   0 AAV 01 0005 00000002
0 1MA 45 0000 01516636   0 AAV 01 0005 00000002
1 1MA 45 0000 01516640   0 AAV 01 0005 00000002
0 1MA 45 0000 01516641   0 AAV 01 0005 00000002
1 1MA 45 0000 01516643   0 AAV 01 0005 00000002
0 1AJ 05 6644 01516644   0 AAV 01 0005 00000002
1 1AJ 05 0000 01516665   0 AAV 01 0005 00000003
0 1AJ 05 7747 01517747   0 AAV 01 4011 00000003
0 ::A 05 0007 01517764   0 AAV 01 0000 00000003
1 1AJ 05 0000 01517766   0 AAV 01 0011 00000003
0 1AJ 05 7766 01517786   0 AAV 01 0011 00000003
1 1RO 05 0000 01521221   0 ::: 00 0011 00000000
0 ::B 05 1541 01524375   0 ::: 00 2205 00000000
0 1SJ 05 0000 01525514   1 ::: 00 0011 00000000
0 ::B 05 2205 01526163   0 ::: 00 1711 00000000
0 ::A 05 0000 01526163   0 ::: 00 0500 00000000
0 1AJ 05 6163 01526163   0 AAX 01 5103 00000000
1 1SJ 05 0000 01526164   1 AAX 01 0011 00000000
0 ::A 05 0500 01526220   0 AAX 01 0007 00000000
1 1AJ 05 0012 01526304   0 AAX 01 0003 00000000
0 CPM 45 0000 01526540   0 AAX 01 0003 00000000
1 CPM 45 0010 01527141   0 AAX 01 0003 00000002
0 CPM 45 0000 01527142   0 AAX 01 0003 00000002
1 CPM 45 0016 01527430   0 AAX 01 0003 00000003
0 1MA 45 0000 01527431   0 AAX 01 0003 00000003
1 1MA 45 0000 01527432   0 AAX 01 0003 00000004
0 1MA 45 0000 01527433   0 AAX 01 0003 00000004
1 1MA 45 0000 01527435   0 AAX 01 0003 00000004
0 1AJ 05 7435 01527435   0 AAX 01 0003 00000004
0 ::A 05 0007 01527630   0 AAX 01 0500 00000004
1 1AJ 05 0032 01527766   0 AAX 01 0003 00000004
0 LDR 45 0001 01530033   0 AAX 01 0006 00000004
1 LDR 45 0025 01530155   0 AAX 01 0006 00000006
0 RPV 45 0001 01530202   0 AAX 01 0007 00000006
1 RPV 45 0000 01530204   0 AAX 01 0007 00000007
0 LDR 45 0000 01530354   0 AAX 01 0006 00000012
1 LDR 45 0333 01531123   0 AAX 01 0006 00000012
0 ::B 05 1711 01532042   0 AAX 01 1527 00000104
0 LDR 45 0000 01532711   0 AAX 01 0002 00000211
1 LDR 45 0046 01533521   0 AAX 01 0002 00000212
0 LDR 45 0000 01533522   0 AAX 01 0002 00000212
1 LDR 45 0271 01534473   0 AAX 01 0002 00000214
0 LDR 45 0001 01536706   0 AAX 01 0003 00000565
1 LDR 45 0061 01537377   0 AAX 01 0003 00000571
0 1MA 45 0000 01543322   0 AAX 01 0010 00000714
1 1MA 45 0000 01543324   0 AAX 01 0010 00000714
0 1AJ 05 3331 01543331   0 AAX 01 0010 00000715
1 1AJ 05 0075 01544434   0 AAX 01 0010 00000717
0 LFM 45 0000 01544444   0 AAX 01 0005 00000717
1 LFM 45 0000 01544447   0 AAX 01 0005 00000722
0 LFM 45 0000 01545665   0 AAX 01 0007 00000754
1 LFM 45 0000 01545676   0 AAX 01 0007 00000754
0 ::B 05 1527 01547150   0 AAX 01 2205 00001026
0 ::B 05 2205 01547571   0 AAX 01 1527 00001042
0 ::B 05 1527 01550147   0 AAX 01 1514 00001046
0 ::B 05 1514 01553266   0 AAX 01 2205 00001305
```

```
0 LFM 45 0000 01554321    0 AAX 01 0006 00001464
1 LFM 45 0000 01554331    0 AAX 01 0006 00001470
0 LDR 45 0000 01564614    0 AAX 01 0002 00002427
1 LDR 45 0025 01565170    0 AAX 01 0002 00002430
0 ::A 05 0500 01566766    0 AAX 01 0173 00002653
0 1MA 45 0000 01567273    0 AAX 01 0002 00002725
1 1MA 45 0006 01567372    0 AAX 01 0002 00002726
0 1AJ 05 7372 01567372    0 AAX 01 0002 00002726
0 ::A 05 0173 01567377    0 AAX 01 0000 00002727
1 1CJ 05 0000 01567771    0 ::: 00 0002 00000000
0 1SJ 05 0000 01576633    1 ::: 00 0004 00000000
0 ::A 05 0000 01577047    0 ::: 00 0500 00000000
0 1RI 05 7047 01577047    0 A80 01 4406 00000000
1 1SJ 05 0000 01577047    1 A80 01 0004 00000000
1 1AJ 05 0032 01577305    0 A80 01 0006 00000006
0 LDR 45 0000 01577323    0 A80 01 0006 00000006
1 LDR 45 0025 01577371    0 A80 01 0006 00000010
0 RPV 45 0000 01577373    0 A80 01 0006 00000010
1 RPV 45 0000 01577375    0 A80 01 0006 00000011
0 LDR 45 0000 01577436    0 A80 01 0007 00000013
1 LDR 45 0333 01600234    0 A80 01 0007 00000014
0 ::B 05 2205 01600463    0 A80 01 1514 00000017
0 ::B 05 1514 01601245    0 A80 01 2205 00000023
0 ::B 05 2205 01601670    0 A80 01 1514 00000034
0 LDR 45 0000 01603042    0 A80 01 0005 00000161
1 LDR 45 0046 01603204    0 A80 01 0005 00000162
0 LDR 45 0000 01603205    0 A80 01 0005 00000162
1 LDR 45 0061 01603621    0 A80 01 0005 00000163
0 1MA 45 0001 01603622    0 A80 01 0005 00000163
1 1MA 45 0000 01603623    0 A80 01 0005 00000164
0 1MA 45 0000 01605716    0 A80 01 0011 00000224
1 1MA 45 0000 01605720    0 A80 01 0011 00000224
0 1AJ 05 5720 01605720    0 A80 01 0011 00000224
0 LFM 45 0000 01606367    0 A80 01 0003 00000227
1 LFM 45 0000 01606371    0 A80 01 0003 00000231
0 LDR 45 0001 01606573    0 A80 01 0007 00000235
1 1AJ 05 0006 01606631    0 A80 01 0011 00000236
1 LDR 45 0025 01607124    0 A80 01 0007 00000236
0 1MA 45 0000 01607203    0 A80 01 0010 00000240
1 1MA 45 0000 01607204    0 A80 01 0010 00000241
0 1AJ 05 7513 01607513    0 A80 01 0010 00000243
0 ::A 05 0500 01607517    0 A80 01 0000 00000243
1 1CJ 05 0000 01610750    0 ::: 00 0010 00000000
0 ::B 05 1514 01612566    0 ::: 00 2205 00000000
0 ::B 05 2205 01616750    0 ::: 00 1514 00000000
0 1SJ 05 0000 01630104    1 ::: 00 0003 00000000
0 ::B 05 1514 01630313    0 M   00 1512 00000000
0 ::A 05 0000 01630313    0 M   00 0002 00000000
0 ::B 05 1512 01630524    0 M   00 1146 00000000
1 CMS 05 0027 01630632    1 M   00 0003 00000000
0 1AJ 05 0632 01630632    0 M   00 0003 00000000
0 ::A 05 0002 01630634    0 M   00 0000 00000000
1 1CJ 05 0000 01630641    0 ::: 00 0003 00000000
0 ::B 05 1146 01652010    0 ::: 00 1336 00000000
0 ::B 05 1336 01652613    0 ::: 00 1345 00000000
0 ::B 05 1345 01653151    0 ::: 00 2045 00000000
0 1SJ 05 0000 01653163    1 ::: 00 0003 00000000
0 ::B 05 2045 01653516    0 ::: 00 1354 00000000
0 ::A 05 0000 01653516    0 ::: 00 0500 00000000
0 1AJ 05 3517 01653517    0 A8N 01 4304 00000000
1 1SJ 05 0000 01653517    1 A8N 01 0003 00000000
```

```
0 ::A 05 0500 01653562    0 A8N 01 0007 00000000
0 ::8 05 1354 01653622    0 A8N 01 1045 00000000
1 1AJ 05 0012 01653662    0 A8N 01 0004 00000000
0 CPM 45 0001 01654201    0 A8N 01 0004 00000001
1 CPM 45 0010 01654353    0 A8N 01 0004 00000003
0 CPM 45 0000 01654355    0 A8N 01 0004 00000003
1 CPM 45 0016 01655461    0 A8N 01 0004 00000004
0 1MA 45 1001 01655462    0 A8N 01 0004 00000004
1 1MA 45 1000 01655463    0 A8N 01 0004 00000005
0 1MA 45 0000 01655464    0 A8N 01 0004 00000005
1 1MA 45 0000 01655466    0 A8N 01 0004 00000005
0 1AJ 05 5466 01655466    0 A8N 01 0004 00000005
1 1AJ 05 0000 01655507    0 A8N 01 0004 00000005
0 1AJ 05 5747 01655747    0 A8N 01 4003 00000005
0 ::A 05 0007 01655764    0 A8N 01 0000 00000005
1 1AJ 05 0000 01655765    0 A8N 01 0003 00000005
0 1AJ 05 5765 01655765    0 A8N 01 0003 00000005
1 1RO 05 0000 01656056    0 ::: 00 0003 00000000
0 ::8 05 1045 01656254    0 ::: 00 0654 00000000
0 ::8 05 0654 01732242    0 ::: 00 0636 00000000
0 ::8 05 0636 02017223    0 ::: 00 0640 00000000
0 ::8 05 0640 02024606    0 ::: 00 0636 00000000
0 ::8 05 0636 02025462    0 ::: 00 1027 00000000
0 ::8 05 1027 02044430    0 ::: 00 1627 00000000
0 ::8 05 1627 02044653    0 ::: 00 1636 00000000
0 ::8 05 1636 02046340    0 ::: 00 1336 00000000
0 ::8 05 1336 02050763    0 ::: 00 1214 00000000
0 ::8 05 1214 02052764    0 ::: 00 1445 00000000
0 ::8 05 1445 02060012    0 ::: 00 1436 00000000
0 ::8 05 1436 02112405    0 ::: 00 2376 00000000
0 ::8 05 2376 02206315    0 ::: 00 2400 00000000
0 ::8 05 2400 02233162    0 ::: 00 2376 00000000
0 ::8 05 2376 02367057    0 ::: 00 2436 00000000
0 ::8 05 2436 02371753    0 ::: 00 2476 00000000
0 ::8 05 2476 02375427    0 ::: 00 2500 00000000

OCCURRENCES   213
```

GLOSSARY of ACRONYMS

ABTM	- Abort CP, CPUMTR function
BCOT	- Batchio Origin Type
CCAM	- Change CP Assignment, CPUMTR function
CCHM	- Check Channel, MTR function
Ch	- Channel
CIO	- Combined Input/Output
CM	- Central Memory
CP	- Control Point
CPA	- Control Point Area
CMR	- Central Memory Resident
CPU	- Central Processor Unit
CPUMTR	- Central Processor Unit Monitor
DCHM	- Drop Channel, MTR function
DEPM	- Disk Error Processor, MTR function
DP	- Data Processing
DPPM	- Drop PP, CPUMTR function
DSC	- Disk Storage Controller
DSWM	- Driver Seek Wait, MTR function
DSU	- Disk Storage Unit
ECS	- Extended Core Storage
EOI	- End of Information
EOR	- End of Record

EP - Exchange Package

EPA - Exchange Package Area

EQ - Equipment

EST - Equipment Status Table

FET - File Environment Table

FL - Field Length

FNT - File Name Table

FST - File Status Table

I/O - Input/Output

IR - Input Register

JACM - Job Advancement Control, CPUMTR function

LDAM - Load Address for MS Drivers, CPUMTR function

MB - Message Buffer

MS - Mass Storage

MST - Mass Storage Table

MTR - Peripheral Processor Monitor (also PPMTR)

OA - Output Register Address

OR - Output Register

OT - Origin Type

P - Program Address

PFN - Permanent File Name

PP - Peripheral Processor (also PPU)

PPR - PP Resident

PRU - Physical Record Unit

RA - Reference Address

RCHM - Reserve Channel, MTR function

RMS - Rotating Mass Storage

RPPM - Request PP, CPUMTR function

RSYM - Request System, MTR function

SET - Initialize System

SYOT - System Origin Type

SYSTEXT - Systems Text

TRT - Track Reservation Table

TXOT - Telex Origin Type

UADM - Update Accounting and Drop, CPUMTR function

1AJ - Advance Job

1CJ - Complete Job

1RI - Roll-in Job

1RO - Roll-Out Job

1SJ - Job Scheduler

6DI - 7054/844-n Driver

( ) - Contents of