P. M. Grant

Interleaving Slow- and Rapid-data-rate Experiments with a Time-sharing Laboratory Automation System

Abstract: A technique is described for accommodating rapid-transfer-rate experiments within an IBM 1800 Time Share Executive (TSX) laboratory automation monitor designed primarily for slow-scanning, low-drift apparatuses, each having the computer control its independent variable. The slow-scan experiments may be delayed for substantial periods to allow break-in by, and dedication of the computer to, those tasks requiring acquisition of short bursts of high-speed data. The system is structured so that each user can implicitly and dynamically specify the current maximum time interval for which his experiment may be interrupted.

The break-in on a slow-scan experiment is done on a demand-response basis through the use of interrupt coreloads and masking of all other interrupts that are likely to interfere with a particular high-speed scan that has been initiated. When data acquisition is completed, control is returned to the time-sharing system by unmasking the interrupts and an appropriate data analysis task is queued for later execution.

Introduction

Papers have previously appeared in this journal[1] discussing on-line data acquisition from various types of laboratory instruments. Many laboratory automation time-sharing systems, for example the one employed by Cole[2] and Okaya[3], slave the computer to the experiment by means of interrupt service subroutines. On the other hand, the system developed by Gladney[4] that is now in operation at the IBM San Jose Research Laboratory, takes the opposite approach and permits the computer to determine the course of a given experiment. The system tradeoffs involved in each philosophy have been discussed elsewhere[5] and are not at issue here. We note, however, that Gladney's technique, in contrast to the former approach, is quite unique and not nearly as ubiquitous. Moreover, this technique allows great simplicity in interface design by eliminating the necessity for the experiment to interrupt the computer periodically and by removing the need for local timing hardware. In the present experiments, however, we were faced with the problem of bringing on-line some experiments in which the basic data rate exceeded the limits of this time-sharing system.

In the present paper, then, we describe the method whereby such high-data-rate applications were integrated into the latter system, which was designed primarily for, and is servicing mostly, low-data-rate experiments that are slaved to the computer. At the same time we also provide an introduction to the two following papers[6,7], which cover actual embodiments.

The concept of time-sharing is difficult to define objectively. From an operational standpoint, one might say that time sharing exists when a community of users, each individually requiring only a fraction of the computer's resources, are collectively unaware of each others' presence within the time frame of their own experiments. If such is not the case, as when certain experiments must be suspended for noticeable and possibly detrimental periods in order that others may proceed, we will say that *interleaving* operations are taking place. Let us then begin by identifying and classifying those factors of an experiment which require appreciation from the viewpoint of a laboratory automation system.

Consideration of experiment types

Table 1 summarizes from a systems aspect the four types of experiments encountered in automation applications. Any given time-sharing system has an ultimate data acquisition rate f_{TS} in points/sec above which our definition is violated. In addition to scan rate, however, process controllability must be considered. One example of an "uncontrollable" process might occur when the independent variable of an experiment is time, such as in transient photodecay and gas-liquid chromatography. On the other hand, the more profound question is really whether the measurement, once initiated, must proceed without undue interruption. See, e.g., the paper by Raimondi et al.[7] on the automation of a residual gas analyzer, where in principle the mass sweep of the instru-

Table 1 Identification of experimental types in terms of acquisition speed and controllability of process. The rate f_{TS} denotes the highest acquisition rate attainable within a given time-sharing operating system, where f is the required scan rate of a given experiment.

Type	*Data acquisition rate*	*Process regulation*	*Examples*
I	Slow, $f < f_{TS}$	Controllable	Optical and rf spectrometers
II	Slow, $f < f_{TS}$	Uncontrollable	Chromatographs
III	Fast, $f > f_{TS}$	Controllable	Rapid scan spectrometers
IV	Fast, $f > f_{TS}$	Uncontrollable	Transient photodecay and mass spectroscopy

ment can be controlled. Nevertheless, in many of the measurements made with this apparatus, the ambient gas is gettered very rapidly, necessitating a fast scan rate with no delays. In some cases where time is one of the independent variables, as in differential scanning calorimetry, it can be "controlled" in the sense that the heating rate can be regulated. Thus the question of the controllability of the independent variable can actually relate to the *regulation of the initiated physical process* under measurement.

The time-sharing system described in Ref. 4 explicitly attacks experiment Types I, II and III. If one foregoes the partially subjective requirement that no noticeable delays be introduced in the data acquisition rate of any experiment, then the logical distinction between Types I and II disappears. The distinction is not entirely subjective because one can further divide Type I experiments into those in which dependent variable drift is a factor and those in which it is not. If we define a time interval over which this drift becomes intolerable, say t_y^{drift}, implicitly assuming $t_y^{\text{drift}} > 1/f \gg 1/f_{TS}$, then we would desire the total scan time T_S of the Type III experiment to satisfy the condition $T_s \lesssim t_y^{\text{drift}}$. In addition, it is worth pointing out that many experiments that are actually Type I may be primitively treated as Type II. A case in point is that of the optical spectrometer described in Ref. 8. However, such a procedure is generally undesirable in the presence of Type IV experiments.

Let us then consider the problems posed by Type IV experiments when they must be interleaved with Type II operations. These have been discussed in a preliminary manner by Gladney[4]. The most critical factor is the size of the time window that can be given by the Type II experiment for Type IV data acquisition. This time interval is a strong function of the Type II toleration of serial information loss and the effectiveness of subsequent data analysis techniques in compensating for such loss. For example, the analysis of gas-liquid chromatographic data may be carried out by analytic-function-fitting methods, in which case the density of points accumulated can be made sufficiently large so that some loss of data can be withstood. On the other hand, should the Type II experiment be in a highly critical state at the moment of interruption, as might occur in a stress-strain measurement at the point of sample rupture, even small data losses might prove highly injurious and not subject to smoothing by numerical techniques. The method to be described here attempts to resolve these conflicts by restricting, through imposition of a "gentlemen's agreement," the time available to Type IV experiments according to the nature of Type II operations currently under way.

Break-in and masking operation

The time-sharing system employed at San Jose resides in an IBM 1800 Data Acquisition and Control Computer operating under Time Share Executive (TSX) and consists of a set of SKELETON subroutines which dynamically create, assign and update software timers needed by various experiments as they come on-line[4,9]. These timers are triggered by hardware timer interrupts operating on the highest possible priority. This system permits six simultaneous experiments to time-share process I/O resources at an upper limit of around 20 accesses per second per experiment. Thus in the context of Table 1, $f_{TS} = 20$ Hz for this implementation.

However, the experiments discussed in the following papers must utilize data rates on the order $f \approx 2$ to 8 kHz with uncontrollable independent variables and hence fall into the Type IV category discussed above. These experiments are accommodated simply by breaking in on current Type I and Type II experiments and masking those interrupts that service them for an acceptable period of time, during which the total resources of the 1800 are dedicated to the Type IV operation. The establishment of an "acceptable" interrupt period is evoked by interrogating the TSX INSKEL COMMON tables, created by the time-sharing system, for information on those Type II experiments in process[4]. This interrogation is done prior to interrupt masking within the Type IV user's coreload by calling subroutine LIMSK[9]. This subroutine returns to the calling coreload an estimate of a tolerable masking interval which the Type IV user is then under obligation to follow. The interval is calculated from a knowledge of f for the fastest Type II experiment active at the time. On the assumption that the Type II user can tolerate the loss of four consecutive data, $4(1/f)$ is passed back to the Type IV coreload as the maximum acceptable masking period. In essence,

each Type II user thus specifies implicitly and dynamically to the Type IV user what the former considers to be his maximum tolerable exposure to masking. We believe this technique to be particularly well suited to our laboratory environment and the task at hand.

Figure 1 contains a flow chart of a typical Type IV user coreload under the San Jose TSX system. This coreload may be of either the queued or the interrupt variety, interrupt coreloads being preferred if prompt commencement of the experiment is desired. Note that one of two options is open in the event the initially requested masking time is greater than the time returned by LIMSK. The user can either exit for the purpose of waiting or manually re-entering a different time interval, or he can have his coreload re-configure the run initialization parameters to adapt the experiment to run within the available period. Under the scheme presented in Fig. 1, Type II operations could possibly be damaged if they were to be initiated between the time allocation check and the masking execution; however, the probability of this occurrence is almost vanishingly small and has never been observed to happen in practice. Finally, note that any data analysis is deferred by queueing appropriate coreloads to the background in order to allow waiting process or nonprocess jobs to continue.

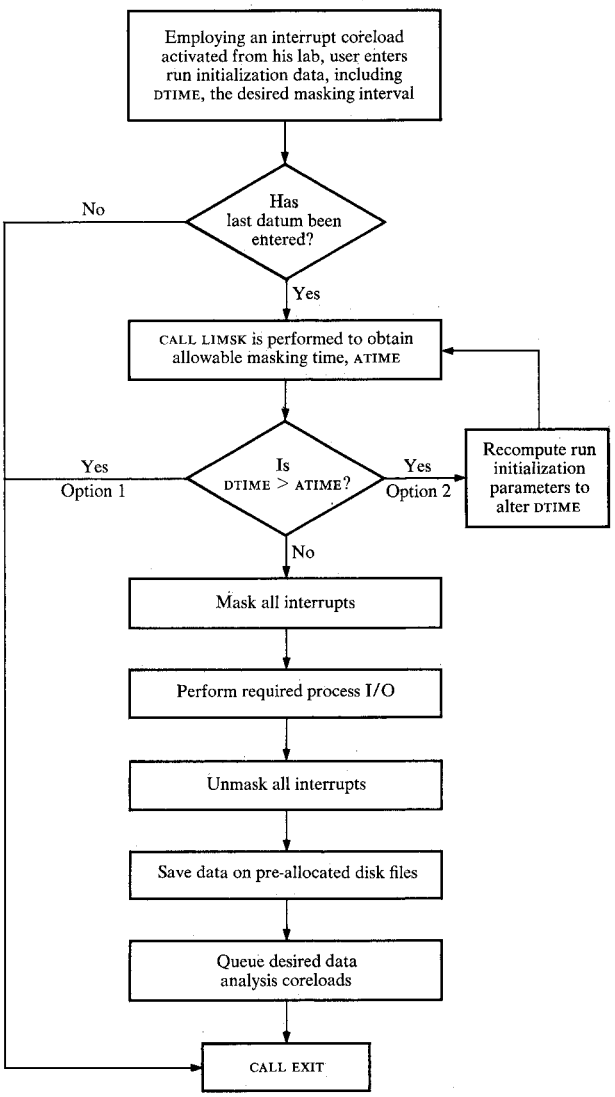

Figure 1 Flowchart of typical Type IV user interrupt coreload.

Summary

A method of operation has been described which permits the interleaving of uncontrolled, high-data-rate experiments in conjunction with a time-sharing system operating in a master-slave relationship towards controlled slow-speed scanning experiments. The method has been so designed that experiments can be classified both as to their mean data rate and as to the controllability of the macrostructure of the rate.

The approach has worked successfully for the past two years under the 1800 TSX implementation used in the IBM San Jose laboratory. Examples of its use will be discussed in Refs. 6 and 7 in this issue.

Acknowledgments

The author wishes to thank D. C. Clarke for many discussions and for advice and help in 1800 Assembler coding problems and express his appreciation to E. Kay for his support of these laboratory automation efforts.

References

1. See the group of 17 papers published in *IBM J. Res. Develop.* **13,** (January 1969). Also, Gayles, Honzik and Wilson, *IBM J. Res.* Develop. **14,** 25 (1970).
2. H. Cole, *IBM J. Res. Develop.* **13,** 5 (1969).
3. Y. Okaya, *ibid.*, p. 126.
4. H. M. Gladney, *J. Comp. Phys.* **2,** 255 (1968).
5. J. D. Swalen, *IBM J. Res. Develop.* **13,** 2 (1969). See also the extensive bibliography included with the foreword.
6. B. H. Schechtman and P. M. Grant, *IBM J. Res. Develop.* **15,** 296 (1971, this issue).
7. D. L. Raimondi, H. F. Winters, P. M. Grant and D. C. Clarke, *IBM J. Res. Develop.* **15,** 307 (1971, this issue).
8. P. M. Grant, *IBM J. Res. Develop.* **13,** 15 (1969).
9. Coded by H. M. Gladney.

Received February 26, 1971

The author is located at the IBM Research Laboratory, San Jose, California 95114.