ZASTOSOWANIA MATEMATYKI
APPLICATIONES MATHEMATICAE
XII, 3 (1971)

J. LEWOC (Wroclaw)

SIMULATION OF A COMPUTER CONTROL SYSTEM

0. Introduction. When designing any computer control system, one of the major problems encountered consists in the analysis of the computing capacity with regard to the requirements imposed by the process to be controlled. Such an analysis may have the following purposes:

1. To determine whether a planned system configuration meets the requirements imposed.

2. To compare various possible configurations and choose the optimum one.

3. To evaluate the degree of system utilization and determine if and in what extent the functions of the system may be enlarged (e.g. if and how long user's programs may be run in the central processor).

The most important purpose is the first one. It is, of course, senseless to compare different system configurations if some of them do not meet the requirements put on the computing capacity. But, if each configuration meets the requirements, it is very usefull to compare them and choose e.g. that of the best functions (cost performance or minimum costs). In the paper a computer control system will be described and analysed by means of Monte Carlo methods.

1. Computer control system for a steel mill. The system considered in the present paper is designed for and installed in a steel mill producing steel bars for numerous small orders.

1.1. Description of the production process and system functions. The steel mill in consideration is a plant producing bars from individual steel billets. Billets are taken from storage places and conveyed to the heating furnaces according to daily production schedules. In the furnaces they are heated up to milling temperature (above 1000°C). Hot billets are pushed down on to the roll table and brought to several working stands where they are milled and cut. The final product (bars) is stored in the cooling room or conveyed to the hot output stand for subsequent treatment.

The production line is long and the equipment installed at its final part enables to transfer bars at a very high speed.

The production process is fully automatized and the only actions to be taken by operators of the individual stands are changes of working programs when the production process is to be modified.

The still mill is, however, intended to produce bars for small orders and production process changes may be relatively frequent so that operators are to intervene frequently. Any operator should recognize the end of a batch of billets (or bars) being processed and set his plant for processing the next portion. It is, still, difficult to recognize the end of a batch since hot billets or bars of different steels are visually identical; this fact forced millers to make relatively large time delays between processing sequential batches.

Such a solution decreased the steel mill output and it was decided to provide the mill in consideration with an automatic system following up the milling process and recognizing ends of portions.

Various technological and economical considerations (which are beyond the scope of the present paper) show the most effective system to be the one based on a real time computer. The system should perform the following functions:

To hold informations referring to the milling process, input once per shift via a paper tape reader.

To pick up signals from photorelays or operators and determine current material distribution along the milling line.

To display all necessary informations for operators according to current material distribution.

To measure the temperature of billets at the output of the heating furnaces and before the second milling station and inform the dispatcher of any deviation from required values.

To measure and store weight of raw material and final product.

To prepare production reports.

1.2. Computer control system. In this section we describe the actual computer control system designed and built for the steel mill. The schematic diagram of the system is shown in Fig. 1.

The Odra 1204 processor is the Central Processor Unit (CPU) of the system. The CPU is linked with necessary input (output units via five input) output channels.

Channel No. 0 (called *operation channel*) is an integral part of the CPU. The channel contains an internal clock which is used in the system being described. The function of the clock is to suspend material distribution updating routines for some period of time.

Such a time delay is necessary because of the material sensing method employed in the system. Individual billets or bars are detected by means of photorelay couples mounted at specific locations along the production line; such a solution increases the reliability of sensing and enables us to recognize the direction of movement (in abnormal process conditions material may be withdrawn from the line; if this occurs, the material

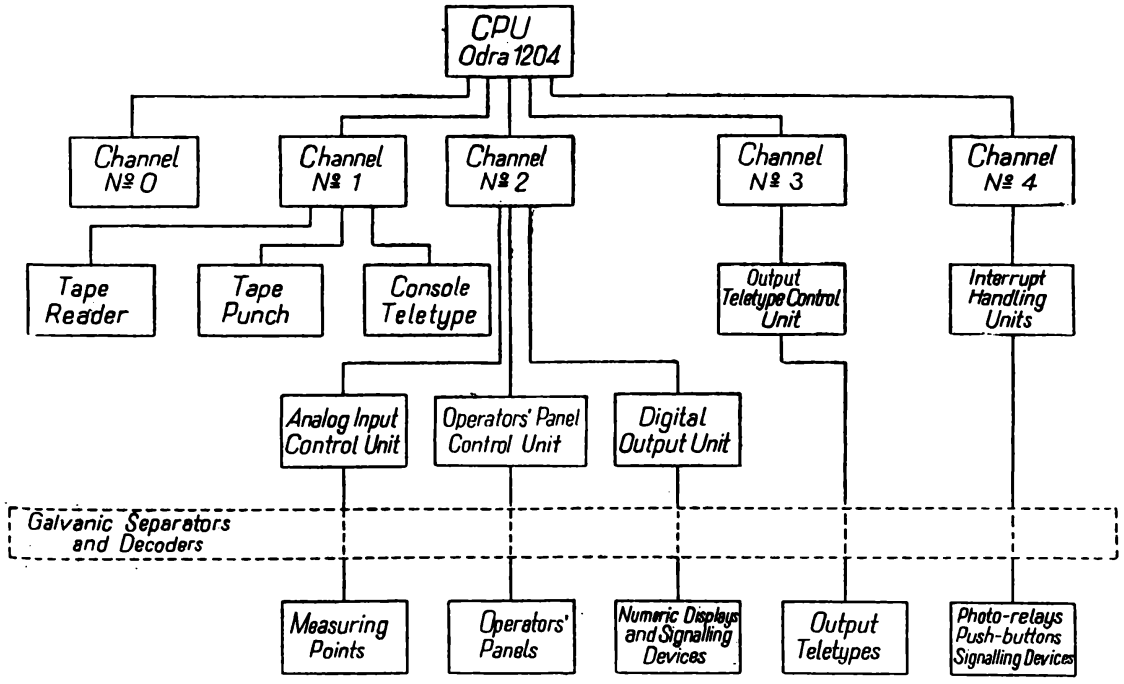

Fig. 1. Schematic diagram of the computer control system

distribution updating algorithm must be modified). When one of any two photorelays is activated, it sends an interrupt signal to the CPU, switching on the internal clock. After approximately 0.1 sec. the clock generates an interrupt signal which results in checking whether the second relay has been activated or not and in entering a proper distribution updating subroutine or the error message subroutine. Since the clock is shared between all pairs of photorelays, demands for service are arranged in a queue; the maximum queue length is an interesting characteristic of the system.

Channel No. 1 is used to connect three basic peripheral units (paper tape reader (TR), tape punch (TP) and console teletype (CT)) to the CPU. The TR and TP are to be employed when the system is controlling the process, and interrupt signals to be processed by the CPU include the operator demand to read in data from the TR and the end of transfer when punching or reading is completed.

Channel No. 2 is the interface between the CPU and the three process control units, i.e. the analog input unit, operators' panel unit and digital output unit.

The first unit controls the operation of reed relay scanner, A/D converter and digital clock.

When an interrupt signal Read Weight (generated by a balance operator) or Read Temperature (from a pyrometer) arrives, the CPU sends a proper analog input number to the scanner which, in turn, initializes the converter after having connected the required sensor to the later. When the measurement and conversion are completed, the converter sends to the CPU the interrupt signal End of Measurement resulting in entering a suitable processing routine.

The converter is used to measure several independent values so that queueing for the device may occur. The maximum queue length enables us to determine whether measurements can be adequately fast. Beside the above-mentioned signal from the converter, the digital clock gererates regular signals causing the CPU to read actual time.

The second unit connects the CPU with the four operator panels used to input some data and / or orders from specific process stands. Via the unit the operators may send to the CPU interrupt signals Read Panel Setting. Moreover, the transfer is of block type and, on the completion, the channel generates the End of Transfer interrupt. As the unit is shared between four operator panels, queuing for it may sometimes occur and it is interesting to check the maximum queue length.

The digital output unit enables us to display appropriate process data at the process operators' stands; the unit generates no interrupt signals.

Through the channel No. 3 reports are written on four output teletypes; the channel generates End of Transfer interrupt signals. Queuing for the channel may occur and the maximum gueue length is of interest when analyzing the real time operation of the system.

Channel No. 4 is the interrupt channel, i.e. it allows to receive and test various signals from push-buttons, photorelays and pyrometers installed in the mill. When considering the photorelays, it is of most importance to service a signal informing that a piece of material has just passed from a process station before the next piece is to pass therefrom; if that fails, a billet or bar may be "lost" and all information upset. It should be checked that such a situation will not occur.

Interrupt signals arriving in normal operation of the system are summarized in Table 1.

TABLE 1. Summary of interrupt signals and processing requirements

No.	Signal description	Arrivals	Processing time (T_p)
1	2	3	4
0	Internal clock interrupt	Approximately in 0.1 sec. after a photorelay interrupt routine requests clock service	Depending upon a number of photorelay routines waiting for clock service $T_p = \sum_{i=1}^{n} C_i + C$ (n — number of photorelay routines waiting for clock action, C_i — processing time of the i-th material distribution updating routine, C — constant)
1	Operator's Demand "Read from TR"	Poisson type	
2	End of transfer (channel 1)	Two signals in regular intervals after processing Read from TR signal or one signal at the end of transfer on TP initialized by some interrupt routines	Various constants (depending on transfer having been completed)
3	End of measurement	Signal delayed by a constant period with respect to request for analog input unit service	$T_p = C_1 + RC_2$ (C_1, C_2 — constants, R — random number with uniform distribution)
4	Digital clock interrupt	In regular intervals	Constant
5-8	Read from operators' panels	Poisson type	Various constants
9	End of transfer (channel 2)	Delayed by a given period with respect to start of reading from any operator's panel	$T_p = C_1 + RC_2$, (C_1, C_2 — constants, R — random number with uniform distribution)
10	End of transfer (channel 3)	In a constant period after starting transfer to any of the output teletypes	Constant

TABLE 1 (ctd.)

1	2	3	4
11-54	Process signals including: — Photorelay interrupts	Interrupt strings arriving from the first relays of pairs are of Poisson type with some given minimum delays Interrupt signals from the second relays of pairs are delayed by a certain interval with respect to those above-mentioned	Various constants depending on the photorelay in consideration
	— Push-button interrupts and pyrometer interrupts	Poisson type of various intensities	Constant or of type as for No. 9

1.3. Interrupt service mechanism. For the system in consideration most of the interrupts are of equal importance and the process being controlled does not require them to be assigned with absolute priorities. Moreover, we may say that interrupting some material distribution updating routines may result in system errors since all routines operate upon the single distribution model. Hence, it was not worthwile to use absolute priorities when the software structure was being planned.

It was, therefore, decided to use relative priorities, i.e., when any interrupts occur, they are processed in a preset order and interrupt programs are run until completion.

2. Problem formulation. Let us consider the above-described system as a multiphase stochastic service system including the CPU, internal clock, analog input unit, operators panel unit and output teletype unit. The system services a set of customers (interrupt signal lines) with assigned constant relative priorities. The customers send demands for service to the CPU. Some of the customers may request service from another block of the system (e.g. internal clock or output teletype unit); in such cases the CPU services the demand partially, transfers it to the requested device and, after required action of the later, which is signalled by an appropriate interrupt, completes service of the demand.

Strings of demands arriving from individual customers are those summarized in Table 1; service times are also described there.

The problem consists in verifying whether the system under investigation is able to service the customers successfully, i.e. whether service of any customer may be completed in certain time limits after the demand has been sent.

The answer to this question should specify the distribution of service time for each customer; deriving such a solution is not easy and worthwile. Instead, we will use another approach to the problem; namely, certain measures will be introduced which characterize the overall behaviour of the system. The measures include:

(i) The number of "lost" signals coming from photorelays, i.e. the number of situations when a demand sent from a photorelay arrives before the previous one has been accepted.

It is of most importance for the system that such situations must not occur since "lost" signals always result in an erroneous information of material distribution along the milling line and, consequently, in the system failure to fulfill the required functions.

(ii) The maximum busy time, i.e. the maximum period during which the CPU is servicing requests from customers (interrupts).

The quantity allows to evaluate the maximum time period between an arrival of a demand and completion of service.

(iii) The mean busy time, i.e. the ratio of total time during which the CPU is busy servicing the requests and the time of system operation. The characteristic is a measure of system utilization.

(iv) The maximum queue lengths for the internal clock, analog input unit, operators' panel unit and output teleprinter unit. The numbers allow to determine if demands are serviced in periods not exceeding limits put on by the production process.

Thus, the problem is reduced to deriving the upper described measures for the system in consideration.

3. Solution. The structure of the system and the demand string distributions make it impossible to derive the required measures by means of analytic methods employed in queueing theory.

It was, therefore, decided to investigate the systems by means of simulation on a digital computer. A program was written describing the above-described service system and run on an Odra 1204 computer. Several runs were performed and the foresaid measures were derived for different input data describing different intensities of the request (interrupt) strings.

3.1. Simulation program. In the discussion we shall use the following symbols:

n — the number of customers for the system in investigation, $n \in \{0, 1, \ldots, 54\}$,

t — time simulated,

$t_r(n)$ — arrival moment of the request of the n-th customer,

Θ – random variable with exponential distribution,
$t_s(n)$ – the service time for the n-th customer,
t_b – the current busy time of the server (CPU),
t_{bmax} – the maximum busy time of the server,
t_{btot} – the total busy time of the server,
t_{bm} – the mean busy time of the server

$$t_{bm} = \frac{t_{b\,tot}}{t},$$

l_c – the internal clock queue length,
l_{cmax} – the maximum queue length of the internal clock,
$n_l(n)$ – the number of lost requests for the n-th customer,
$c(n), c_1(n), c_2(n)$ – tables of constants determined by the input data.

3.1.1. The main loop of the simulation program. The flow diagram of the main loop is shown in Fig. 2. Having been initialized, the program reads the input data specifying the input process rates, the parameters for processing time calculations and the initial state of the system (i.e. the arrival moments for the first requests of all customers). On the basis of the input data, the program calculates tables for generation of the Poisson process (the algorithm of the generation will be described later on) and starts the proper simulation.

The table of arrival moments $t_r(n)$ is looked up for the earliest request to be served by the CPU. The value found is substituted for the time simulated and the subroutine corresponding to the entrance is called. The subroutines normally generate values of next arrival moments of the customer requests $t_r(n)$ and service times $t_s(n)$, but some of them are more complex and generate the next arrival moments for other customers or calculate required queue lengths. When exit is made from the subroutine, the values of time simulated and of busy time are updated, namely:

$$t := t + t_s(n), \qquad t_b := t_b + t_s(n).$$

This moment corresponds to the completing of the service of the n-th customer when the server becomes free to serve other requests (if there are any). Now, all customers are checked in ascending order of their number (descreasing priorities) if they have sent requests for service during the busy time of the server, i.e. if the condition

$$\sum_{n \in \{0,1,\ldots,54\}} t_r(n) \leqslant t$$

is fulfilled. If so, the program branches back and enters the subroutine corresponding to the suctomer requesting service.

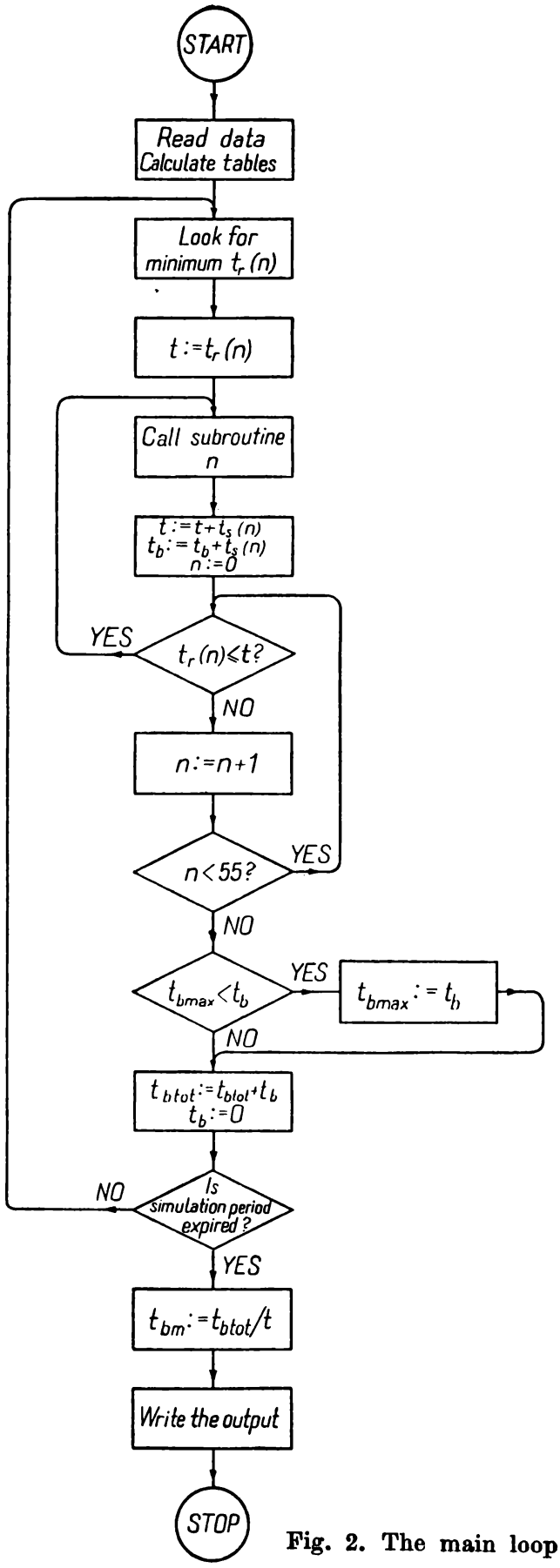

Fig. 2. The main loop

Otherwise, i.e. if

$$\prod_{n\in\{0,1,\ldots,54\}} t_r(n) > t,$$

which means that the busy period of the server has just been finished, the values of the maximum and total busy time of the server are updated and the current busy time is reset to zero.

Then the program checks if the simulation period has been attained.

If not, the branch to the very beginning of the simulation is executed; otherwise, the mean busy time of the server is calculated, the results are output and the program stops. It follows that the main loop simulates a general mass service system in which one server handles requests of a set of customers represented by subroutines. The measures calculated by the loop include the maximum and mean busy time of the server.

3.1.2. An example of the customer simulating subroutines. Let us consider a set of three related subroutines, namely: the internal clock subroutine, the first-of-photorelay pair subroutine and the second-of-photorelay pair subroutine; the block diagrams are shown in Figs. 5, 3 and 4, respectively. The measures to be derived by the set include the maximum queue length of the internal clock and the number of photorelay signals "lost"; moreover, the subroutines should obviously calculate the arrival moment $t_r(n)$ and service times $t_s(n)$ required by the main loop. Having been entered, the subroutine of the first-of-photorelay pair checks if the next request has arrived before starting to serve the current one. If so $(t \geqslant t_{r_1}(n))$, the subroutine will increase the number of lost signals $n_l(n)$ by one and substitute the $t_{r_1}(n)$ for $t_r(n)$. Then the subprogram generating the value of the random variable Θ of the exponetial distribution is called.

The substitution

$$t_{r_1}(n) := t_{r_1}(n) + \Theta + c(n) \tag{1}$$

(where $c(n)$ is a constant determined by the input data) is the final operation of that program path.

The next stage starts from substituting $t_{r_1}(n)$ for $t_r(n)$; then a new value of $t_{r_1}(n)$ is calculated by means of formula (1).

The subsequent box represents determining the moment of the next-request arrival for the second photorelay of the same pair $t_r(n-1)$, the $c_1(n)$ being a constant set by the input data.

The service time $t_s(n)$ for the request in consideration is taken from table $c_2(n)$ given by the input data.

The final stage of the subroutine simulates queueing for the internal clock. Checking the queue length for zero value, it is tested whether the

clock is busy. If so, the clock is started, $t_r(0) = t + 0.1$ sec.; $l_c := 1$ and the service time for the clock request is set to $c_0 + c_3(n)$, where c_0 is the time for servicing the clock request itself and $c_3(n)$ is the time required to process the n-th material distribution updating routine.

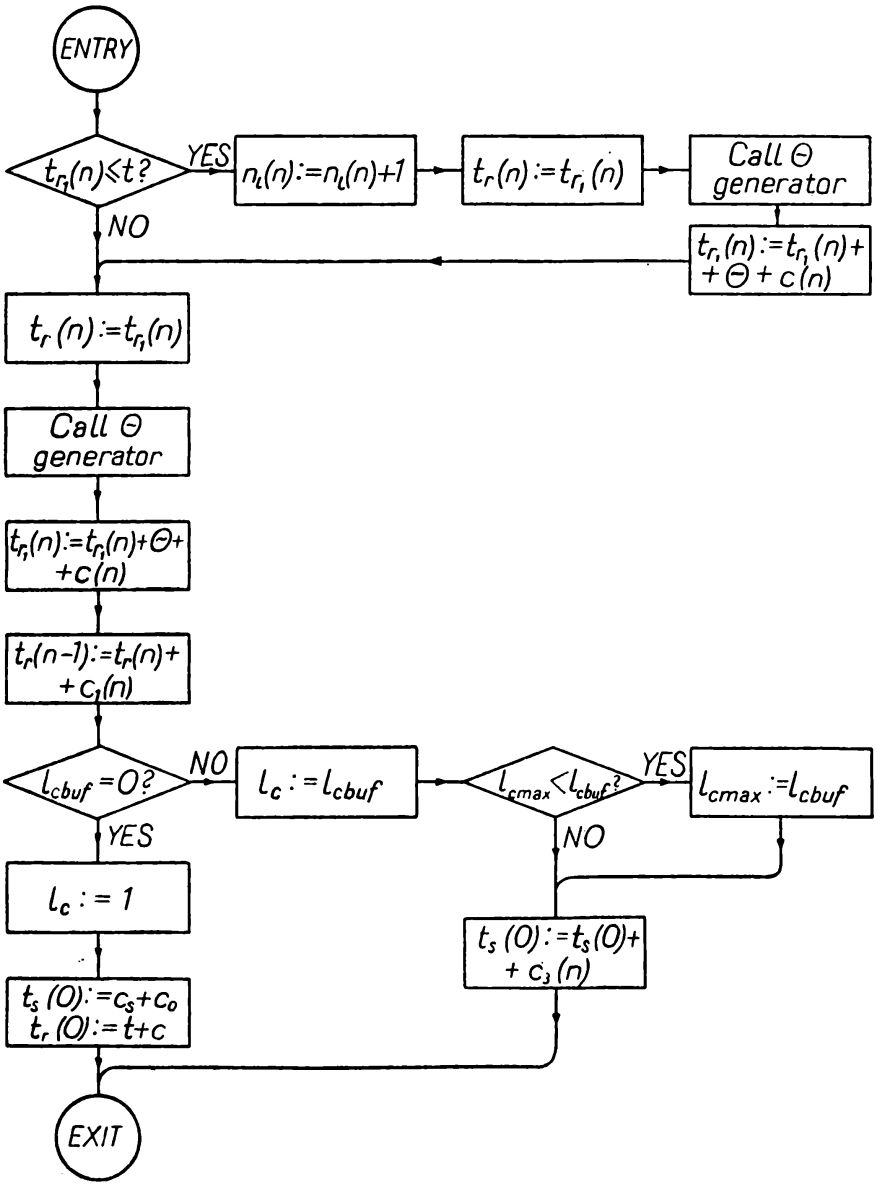

Fig. 3. Subroutine for the first photo-relay of a pair

If the clock is busy at the moment, the subprogram increases the buffer queue length by 1, updates the value of the maximum queue length and sets the service time for the next clock request to

$$c_0 + \sum_i c_2(i) + c_2(n),$$

where $\{i\}$ is a set of photorelays waiting for service from the clock.

Referring to the second-of-photorelay pair subprogram (see Fig. 4), it only substitutes a value of $c_2(n)$ for the $t_s(n)$ and a value greater than the simulation period for the $t_r(n)$ (the actual $t_r(n)$ will be set by the $(n+1)$-th subprogram). The check for "lost" signals is not made since the arrangement of priorities eliminate the possibility to lose here any signal if all signals from the preceding one have been serviced properly.

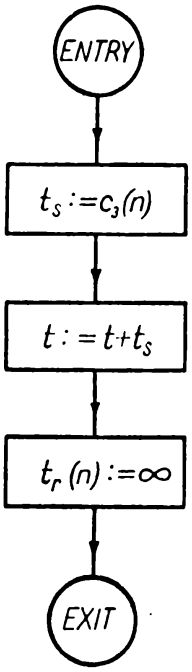

Fig. 4. Subroutine for the second photorelay of a pair

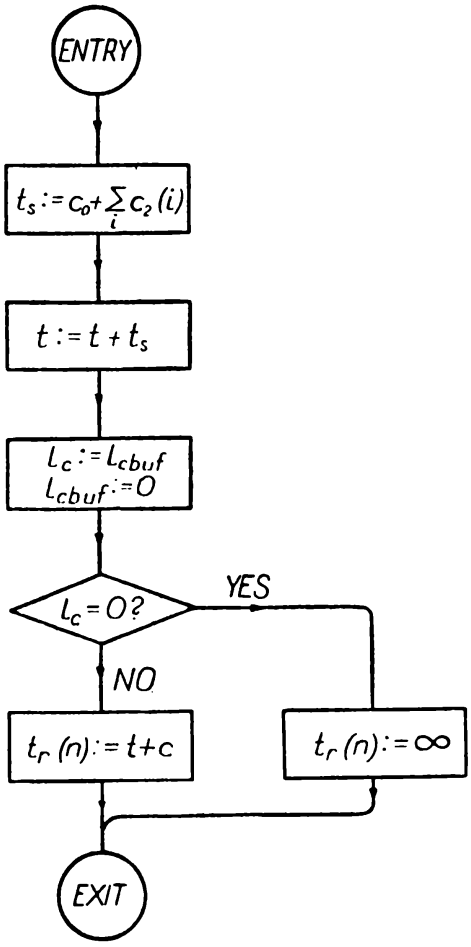

Fig. 5. Internal clock subroutine

The internal clock subprogram substitutes for $t_s(0)$ the value

$$c_2(0) + \sum_i c_2(i),$$

where $\{i\}$ is a set of numbers corresponding to the set of material distribution updating routines to be run, and checks if the buffer queue is empty. If so, the $t_r(0)$ is set to a value greater than the simulation period; if not, the buffer queue length is substituted for the actual queue length and set to zero, $t_r(0)$ is set to $t+0.1$ sec., and the $t_s(0)$ for the next clock request is set to

$$c_0 + \sum_i c_2(i),$$

where $\{i\}$ corresponds to photorelay requests having formed the buffer queue.

3.1.3. The generator of random variables with exponential distribution. When constructing the stochastic model of the system it was assumed that interarrival times of some customer requests are sums of certain constants $c(n)$ and exponentially distributed random variables $\Theta(n)$. The assumption seems to be realistic for the signals from the mill in consideration; the $c(n)$ are the lower limits determined by the maximum speed of the milling plants and $\Theta(n)$ reflects the action of operators controlling the former. It is, thus, necessary to generate such variables.

To this end, let us divide a finite interval of the values of the random variable into n sections, so that the probability of lying in the interval (a_k, a_{k+1}) equals to $1/n$ $(k = 0, \ldots, n-1)$. If the distribution has the density function $f(x)$, the condition is equivalent to the following formula:

$$\int_{a_k}^{a_{k+1}} f(x)\,dx = \frac{1}{n}. \tag{2}$$

For a random variable with the exponential distribution we have $f(x) = \lambda e^{-\lambda x}$, so that formula (2) may be written as

$$\int_{a_k}^{a_{k+1}} \lambda e^{-\lambda x}\,dx = \frac{1}{n}, \qquad a_0 = 0. \tag{3}$$

It follows from (3) that

$$a_{k+1} = -\frac{\ln(e^{-\lambda a_k} - 1/n)}{\lambda}. \tag{4}$$

The values of a_k are calculated from (4) at the beginning of any simulation run. The table is then used by the generator of the random variable with exponential distribution. The generator is a subroutine of the flow diagram shown in Fig. 6. Having been entered, it generates a value of the random variable R with uniform distribution. The six most significant bits of the number are taken as an address for a table entry. Then another value of R ($R \in (0, 1\rangle$) is calculated and the exponentially distributed variable will be generated on the base of the following formula:

$$t_r(n) = a_k + (a_{k+1} - a_k)R.$$

The values of R are determined by a subprogram complementing the following operations:

$$X := X0 + 2 \times X1;$$
$$X0 := X1;$$
$$X1 := X.$$

The generated sequence is a modified Fibonacci sequence and is featured by a very high period and little correlation between sequential values.

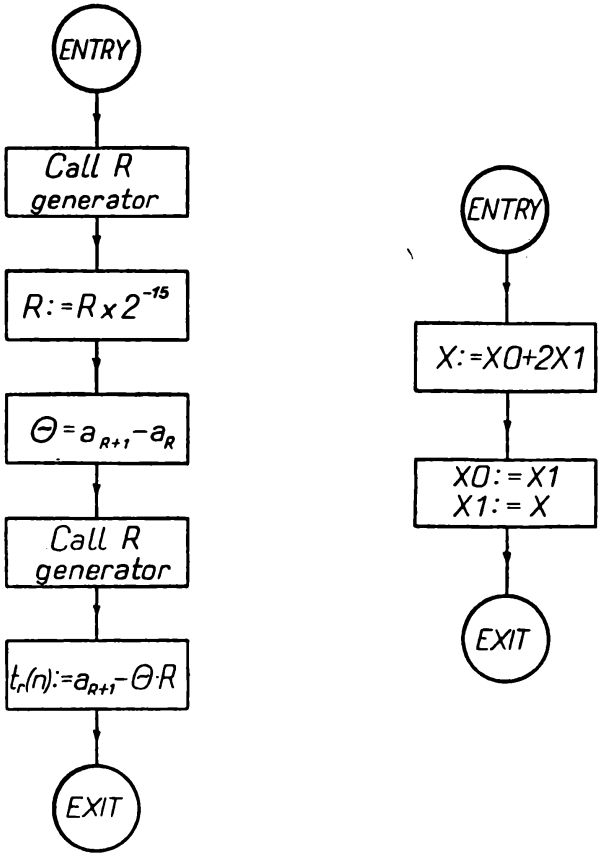

Fig. 6. θ nad R generators

3.2. Results. In order to investigate the system in consideration, several program runs were performed for different sets of the input data. The results showed the following:

(i) For the distributions and intensities of the requests, which are likely to occur in the actual environment, the computing capacity of the system meets the requirements put on by the milling process. The mean busy time of the CPU is a few percent. The maximum busy time is relatively small. There is no evidence for loosing any interrupt signals. The maximum queue length of the peripheral units and the internal clock shared between various interrupts are small and there is no danger that any of the units may introduce inadmissible delays.

(ii) The intensities of the input processes that result in a saturation of the system are far above the maximum ones that may be achieved in the actual plant.

(iii) When the intensites are increased, the maximum busy time increases faster than the mean time. From this it follows that while investigating any digital control system, the maximum times should be recorded since the mean times are not good measures for such systems.

The simulation proved to be an efficient method of investigating a digital control system. We now summarize some of the advantages of the method.

(i) The simulation allows to model a stochastic service system relatively precisely. It is not necessary to make overassumptions for structures and the service mechanism and the actual ones may be directly investigated.

(ii) Various measures may be derived by means of this method.

This is of special importance in consideration of digital control systems in which measures other than those derived for stochastic service systems in general are required (e.g. the maximum waiting and for service times are of interest rather than the mean times).

(iii) It is relatively easy to overcome the problem of conditional events. This feature is very important for investigation of digital control systems since very often signals which are sent from a process in control are strictly correlated reflecting the interactions between individual process stages.

(iv) A variety of system configurations may be analysed and compared in a relatively simple way. Having established a service mechanism, which is often determined by the process in control or by a general organization of a control system, a program may be written which simulates the mechanism, and any modifications of the structure need only to change numbering or to modify the subprograms simulating the individual customers (interrupts).

(v) Sequences of random variables with complex distributions may easily be generated, so that it is not necessary to make overassumptions with respect to input processes and service times.

PRACOWNIA PROJEKTOWO-TECHNOLOGICZNA
WROCŁAWSKIE PRZEDSIĘBIORSTWO AUTOMATYZACJI „ELAM"
WROCŁAW

Received on 9. 12. 1970

J. LEWOC (Wrocław)

SYMULACJA PEWNEGO PROCESU STEROWANIA

STRESZCZENIE

W pracy przedstawiono przykład analizy konkretnego systemu sterowania, dotyczącej spełnienia wymagań pracy systemu w czasie rzeczywistym. Opisano obiekt i proces kierowany za pomocą systemu wykorzystującego maszynę cyfrową Odra 1204. Ponadto została omówiona konfiguracja wyposażeniowa oraz wymagania dotyczące przetwarzania poszczególnych sygnałów odbieranych przez system. Dla przeprowadzenia analizy, przedstawiono go jako wielofazowy system masowej obsługi i określono parametry charakteryzujące jego pracę. Wartości tych parametrów wyznaczono metodą symulacji na maszynie cyfrowej. Praca zawiera opis pętli głównej programu symulacji konkretnego systemu oraz wnioski dotyczące symulacji podobnych obiektów.