Digital Control of Industrial Processes

CECIL L. SMITH

Louisiana State University, Baton Rouge, Louisiana*

The application of computers for controlling process units is now about ten years old. The applications have progressed from the experimental stages of early development to a well-established (economically speaking) control practice. This has not been without some anxious moments and a few outright failures. These have not been forgotten, but current successes are proving beyond any doubt that digital computers are capable of generating an attractive economic return on the investment required for their installation.

This paper has three objectives: first, to acquaint those outside the process industries of the general process control problem in order that they may appreciate some of the requirements imposed on a control computer; second, to describe the computer hardware normally considered for process control; third, to present the general philosophy of the real-time software systems required for process control.

Key words and phrases: industrial process control, digital control, process control, computer control, real-time systems, control systems, supervisory control, hierarchy control systems, computer-process interface, computer control software, computer control hardware

CR categories: 1.3, 3.22, 3.82

CONTROL

The Process Control Problem

Industrial processes for which successful computer control systems have been reported include blast furnaces, petroleum and petrochemical plants, paper machines, textile mills, etc. Each has its unique problems, so this discussion must be fairly general. In all of these processes, the variables are divided into these four categories, as illustrated in Figure 1:

- Manipulated variables. These are variables such as input raw material flow rate, steam pressure in a vessel, etc., whose values can be adjusted by the control system, whether analog (conventional) or digital.
- Disturbances. These are variables whose values affect the operation of the process but which are not subject to adjustment by the control system. Examples include composition of raw material, ambient air temperature, etc. Some variables in this category can be measured, while others cannot.
- Controlled variables. These are the variables whose values really measure the performance of the plant, and thus are those which the control system must keep at some target value (often called the set point). Examples include production rate, product quality, etc. The general control problem is to adjust the manipulated variables so as to maintain the controlled variables at their target values in face of disturbances. Some controlled variables can be measured directly, but some must be inferred from other measurements, a task at which digital computers far excel their analog counterparts.

---

* Department of Chemical Engineering. This work was supported in part by a Project THEMIS contract administered by the US Air Force Office of Scientific Research for the US Department of Defense.

CONTENTS

- Intermediates. These variables appear at some intermediate point in the process. The control system can often use them advantageously in determining what control action should be taken. Examples include temperature of a water jacket, composition of an intermediate stream, etc.

As a typical plant has several variables in each of the above categories, it is apparent that the control of process units is no simple matter. This is further complicated by the difficulty in deriving a mathematical model of the process from process characteristics. The problem in this regard is that the process characteristics depend, first, on the level of plant operation (i.e. the plant is usually highly nonlinear), and second, even at a constant operating level the plant's characteristics change with time (i.e. the plant is nonstationary).

Although complicating the job of installing the computer control system, these aspects are really the basic reasons such a sophisticated control system can be justified. The ability of the digital computer to collect large quantities of data, analyze it, and make logical decisions based upon the results makes it attractive for such applications.

Conventional Control Systems

Before delving into the characteristics of digital control systems, an appreciation of the conventional approach to process control is a helpful background [1.3, 1.4, 1.12, 1.15]. The basic control loop in conventional (analog) systems is the simple feedback loop illustrated in Figure 2. The value of the controlled variable is detected by a sensor or transmitter, e.g. a thermocouple for measuring temperature. This value is compared to the desired value or set point to generate the error. The control law generates a change in the manipulated variable so as to drive this error to zero. The controller output is imposed upon the process by an actuator, which is an automatic positioning valve in most cases.

The control law commonly used is the proportional-integral-derivative (PID) relationship or some simplification thereof. That is, the manipulated variable $m(t)$ is related to the error $e(t)$ by the equation

$$m(t) = K_c\left\{e(t) + \frac{1}{T_i}\int_0^t e(\tau)d\tau + T_d\frac{de(t)}{dt}\right\} + m_R, \qquad (1)$$

where

K_c = proportional gain,
T_i = reset or integral time,
T_d = derivative time,
m_R = reference value at which the control action is initialized.

The adjustments K_c, T_i, and T_d appear generally as adjustments on the rear of the controller. The selection of their proper values is normally a trial-and-error procedure called "tuning," although some systematic approaches have been suggested [1.12, 3.13, 3.48, 3.54, 3.64, 3.70–3.72, 3.80, 3.81, 3.83].

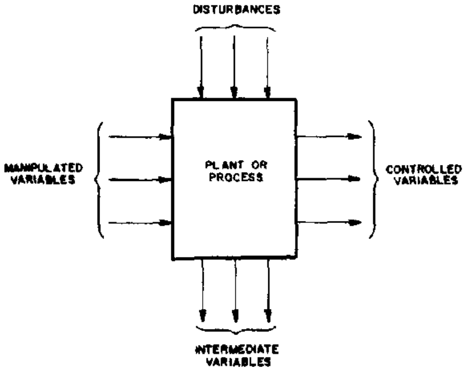


FIG. 1. General representation of a process

In well over 75 percent of the applications, only the proportional-integral (PI) terms are used, primarily because of the difficulty of tuning the general PID (or three-mode) controller.

In a typical plant, there may be anywhere from a few of these devices to upwards of a hundred or more. Until around the late 1950s, these devices were invariably pneumatic (operated on air pressure). Aside from being much more reliable than their vacuum-tube electronic counterparts, they had the added advantage of safety when used in areas in which explosive gases might be encountered. Only with the introduction of solid-state electronic controllers in the late 1950s have pneumatic controllers gradually begun to be replaced.

No matter whether pneumatic or electronic, conventional analog control systems basically suffer from inflexibility. There must be almost a one-to-one correspondence between control loop functions and hardware to perform these functions. This places several burdens upon the designer of the control system: (a) his strategy must be such that it can be implemented with analog hardware; (b) subsequent modifications of the control strategy require modifications of the analog hardware.

In the mid-1950s, control system designers began to look toward the digital computer as a means for circumventing these problems. Virtually any control strategy is

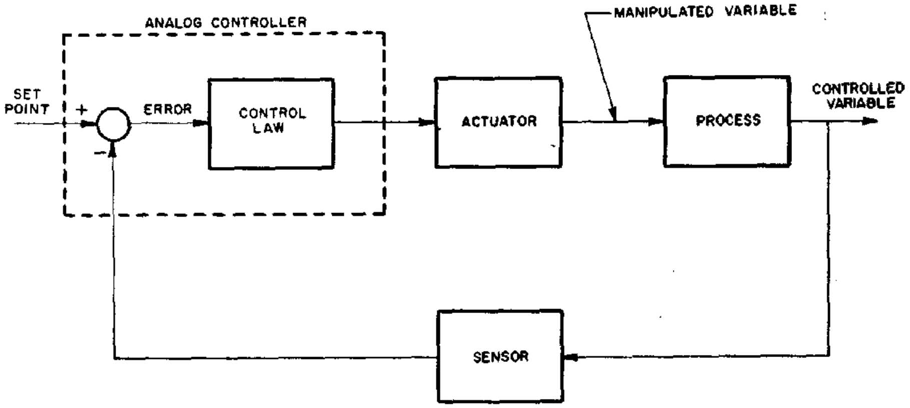


FIG. 2. Basic conventional feedback control loop

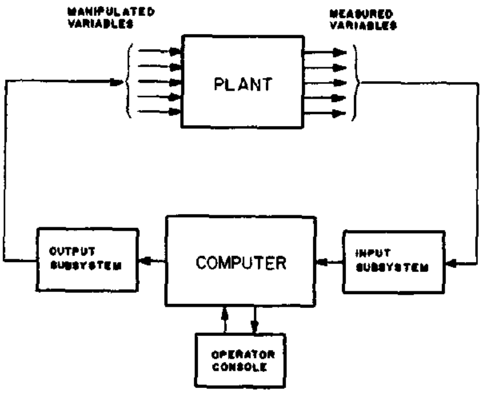


FIG. 3. Direct digital control

programmable, and most modifications in the strategy are simply program changes.

There have been two distinct approaches to digital control—direct digital control versus supervisory digital control [1.7, 1.14, 2.11, 2.19]. Although most control systems are combinations of the two, they shall be treated distinctly here.

Direct Digital Control

In direct digital control [1.14, 2.6–2.8, 2.12, 2.16, 2.20, 2.24, 2.25, 2.28] the computer calculates the values of the manipulated variables (e.g. valve positions) directly from the values of the set points, controlled variables, and other measurements on the process. The decisions of the computer are applied directly to the process, and hence the name direct digital control. This control arrangement is illustrated in Figure 3.

As the values of the manipulated variables are calculated by the computer, the conventional three-mode controllers described above are no longer needed. Their functions are instead performed by the equations, called algorithms, by which the computer calculates the manipulated variable from the set point and controlled variable. An example of a control algorithm is the discrete (finite difference) equivalent to eq. (1) for the continuous controller [3.4, 3.10, 3.12, 3.14, 3.17, 3.19, 3.22, 3.28, 3.65]:

$$m_n = K_c e_n + (K_c T/T_i) \sum_{i=0}^{n} e_i + (T_d K_c/T)(e_n - e_{n-1}) + m_R, \tag{2}$$

where

m_n = value of manipulated variable at the nth sampling instant,

e_n = value of the error at the nth sampling instant,

T = sampling time.

Other parameters are defined as before. This equation is called the position form of the control algorithm, as its result is the actual value of the manipulated variable, typically a valve position. If eq. (2) is written for m_{n-1} and subtracted from eq. (2) as given above, the result is

$$\Delta m_n = K_c(e_n - e_{n-1}) + (K_c T/T_i)\, e_n + (T_d K_c/T)(e_n - 2e_{n-1} + e_{n-2}), \tag{3}$$

where $\Delta m_n = m_n - m_{n-1}$ is equal to the change in the manipulated variable; hence the name "velocity algorithm." The significant difference between eqs. (2) and (3) is that eq. (3), the velocity algorithm, does not contain the term m_R, the value of the manipulated variable when the loop is placed on automatic (i.e. calculations are begun). If eq. (2) is used, the computer must be able to read the valve position to insure a smooth transition from manual to automatic, called "bumpless transfer." Using eq. (3) accomplishes the same objective without having to read m_R.

Being the digital equivalent of the three-mode analog controller, the above algorithms are used in by far the majority of the applications. The adjustment or tuning has been investigated extensively [3.4, 3.15, 3.49, 3.50, 3.57–3.60, 3.74]. There have been other algorithms proposed, some of which exceed the performance of the above algorithms. A field of mathematics known as z-transform theory has been developed especially for digital control systems [1.5, 1.6, 1.8, 1.11, 1.13, 1.16, 1.17, 3.35, 3.43, 3.73]. If the system being controlled can be described reasonably well mathematically, algorithms can be designed to give almost any desired performance.

One of the prime control considerations in DDC is selecting the sampling time. In general, the performance of the system improves as the sampling time is decreased. However, this increases the computational load on the digital processor, eventually limiting the number of control loops a given processor can service. There are no hard-and-fast rules for selecting sampling times, one rule of thumb being one second for flow loops, five seconds for pressure loops, and twenty seconds for temperature loops [2.1].

One of the first incentives suggested for DDC systems was economic savings. The basic idea was that since one computer could provide the same functions as several analog controllers, there must be some point at which the cost of these several analog controllers would equal the cost of the digital system. Some early estimates on the number of loops to be replaced ranged as low as fifty, but this unfortunately was not proven to be correct. Two problems seemed to have been underestimated:

—Programming costs. With no prior experience and without DDC software packages or proven monitors, the programming effort far exceeded that anticipated.

—Backup hardware. This problem stems from the fact that operating personnel must be able to exercise effective control over the plant in event of total computer failure. In many cases this backup was a complete analog system, thus eliminating any hardware savings.

In regard to the question of reliability, the first Users' Workshop on Direct Digital Computer Control [2.1] indicated that a computer availability of 99.95 percent (translates into about four hours downtime per year) would be required for DDC. As this downtime included diagnosis and repair of the problem, it is obvious that the machine should be easily repaired as well as dependable. This consideration has promoted plug-in modules and retarded the trend toward compact machines. In case of computer failure, the system is normally designed so that all outputs are frozen at their most recent valid values. This should occur for a single loop whenever a component fails as well as for the total system when the computer fails.

Current thoughts seem to be that DDC systems simply cannot be justified on the basis of savings in analog hardware. Instead, the justification must come from the application of control techniques that are either impractical or impossible to implement with analog hardware. Such techniques include:

1. *Compensation of process variables.* A prime example is correcting gas flow readings for pressure and temperature changes.

2. *Feedforward control* [1.12, 1.15, 3.9, 3.21, 3.32, 3.52, 3.67]. As illustrated in Figure 4(a), the feedforward control system calculates a change in the manipulated variable to offset the effect of a measured disturbance upon the process. This control strategy is typically used in addition to the usual feedback strategy.

3. *Cascade control* [1.12, 1.15, 3.18, 3.24, 3.30, 3.77]. As illustrated in Figure 4(b), the cascade control system comprises two simple feedback control loops, one of which provides the set point for the other. For example, the outer (master) control loop might calculate the temperature of the water jacket to maintain the reactor temperature at its desired value. The inner (slave) controller then adjusts the water flow to the jacket to give the desired jacket temperature.

4. *Multivariable (noninteracting)* [1.15, 3.6, 3.16, 3.23, 3.82]. In many processes two (or more) variables are to be controlled by adjustig two (or more) manipulated variables. In most cases, adjusting each manipulated variable affects both controlled variables, and the process is said to be coupled. If two separate analog controllers are employed, the two loops affect each other, and are said to be interacting. This interaction can frequently be minimized by using a "decoupler" in conjunction with these feedback loops, as illustrated in Figure 4(c). The decoupler is designed so that adjusting one of its inputs will produce a change in only one of the controlled variables, thus apparently eliminating the interaction.

5. *Adaptive systems* [1.2, 3.1, 3.2, 3.8, 3.26, 3.39, 3.63]. These systems modify the control strategy in response to changing characteristics of the process. They generally attempt to measure one or more parameters in a process model and change the control strategy accordingly.

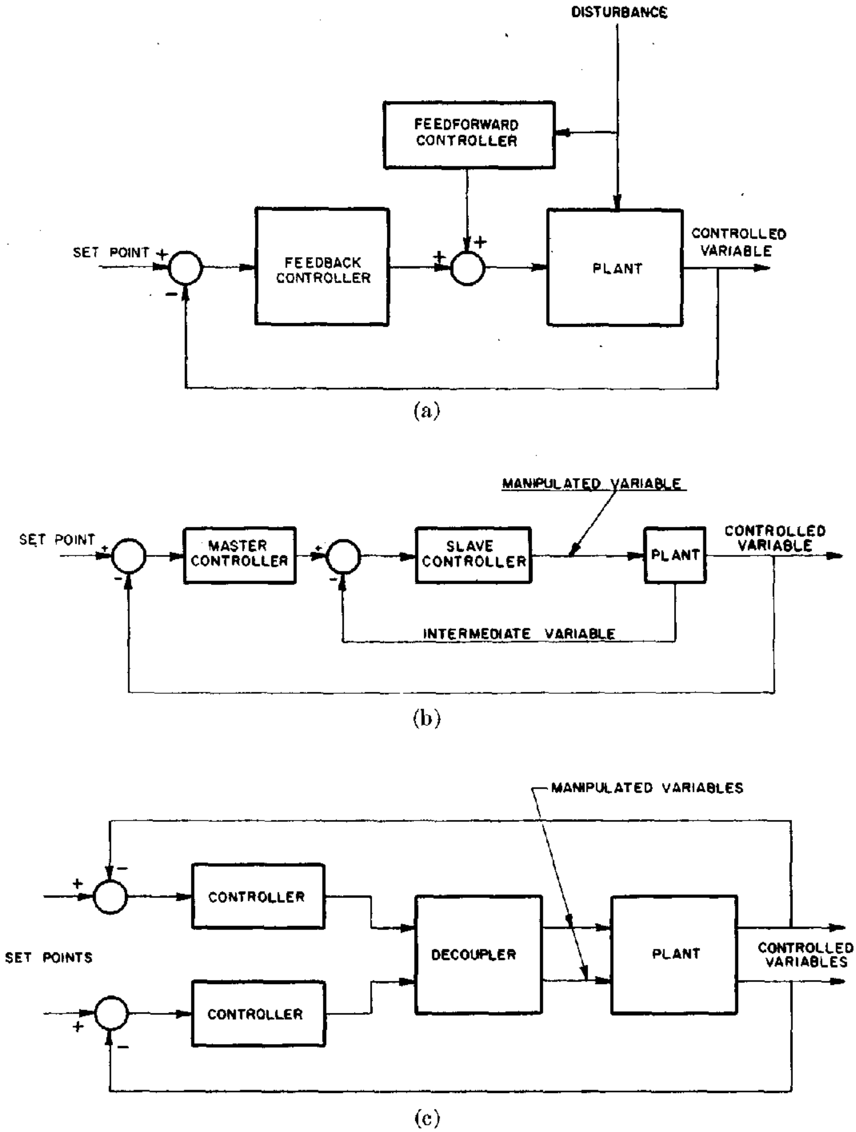


Fig. 4. Advanced control concepts: (a) feedforward, (b) cascade, (c) multivariable

6. *Automatic tuning* [3.25]. As mentioned above, one of the real problems is the tuning of the three-mode controllers. This same problem exists with control algorithms, but progress is being made on procedures that enable the computer to perform the tuning.

7. *Dead-time compensation* [3.2, 3.51]. Most process systems have significant dead-times (time delays or transportation lags, i.e. the time required for material to move from one point to another). Such systems are difficult to control with the conventional three-mode controller or corresponding algorithms, but special compensation techniques are attractive for digital systems.

8. *Optimal control.* Dynamic optimization of process units has not been applied on a widespread basis to date. Several reasons account for this, such as a large number of state variables, the requirement that

the control law be feedback in nature, availability of only relatively poor mathematical models of most processes, etc. Estimation theory may answer the last problem, and at least one investigation has been reported [3.78]. However, the main reason appears to be that there is simply not enough economic incentive in most processes to justify the effort to apply optimal control theory.

One especially promising area for DDC is in the control of batch operations. Control of these units can become quite intricate, especially when one piece of equipment (usually a chemical reactor) is used to manufacture a variety of different products, each of which requires a different operating procedure. The bookkeeping becomes very tedious, resulting in less efficient operation of the unit under manual supervision of a conventional control system. The computer's ability to store operating procedures for all possible batches, to flawlessly progress each batch according to this schedule, and in general provide reproducible and consistent operation gives it a distinct advantage. Standard software designed specifically for batch operations is becoming more widely available.

Supervisory Computer Control

The basic objective of a process operation is to optimize the financial return on investment. The economic return on an operation depends upon a number of factors, one of the significant ones being the day-to-day operating strategy. It is frequently not obvious to the operating personnel what the optimum operating strategy should be. A plant is a complex, interacting entity, and the optimum operating strategy can only be ascertained after considering the combined effect of many different options.

The obvious approach is to use the digital computer to perform just such an analysis. Typical input information needed might include:

- cost of raw materials, utilities, etc.
- value of products;
- composition of raw materials and products;
- current values of variables within the process;
- constraints on the operation, e.g. safety limitations;
- specifications on products.

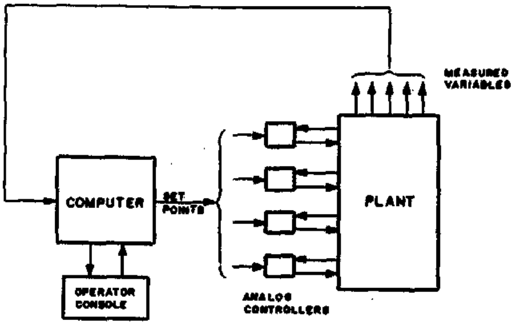


FIG. 5. Supervisory control system

A process model is needed to relate all of these various factors to the economic return on the operation. The optimum operating strategy is that which optimizes this return.

Although the computer determines the optimum operating strategy, the analog control system still implements the decisions. Thus, in many cases, the control computer simply provides the set points for the analog control loops, as illustrated in Figure 5. The computer system does not replace any analog hardware. The backup problem is not as critical, as in the case of computer failure the set points simply remain at their last settings.

The concept of supervisory control actually dates back to the development of autopilots for aircraft. Early development was conducted by the Bunker-Ramo Corporation and by Hughes Aircraft, the latter holding a patent, called the Exner patent [2.2], covering the supervisory concept (specifically, using a digital computer to provide the set point for an analog controller). Bunker-Ramo entered the process computer field and pioneered much of the early development.

The existence of the Exner patent has been of particular importance in supervisory control projects. Although some moves toward litigation have been considered, the current trend appears to be that computer manufacturers are negotiating licensing agreements covering the use of their products in a supervisory environment.

The economics [1.7, 1.14, 4.1–4.14] of supervisory systems are based on the prospect of the system producing sufficient improvement in process operation to justify the financial investment in the computer control system. This is seldom easy to verify beforehand, but processes that are likely to give such a return generally exhibit one or more of the following characteristics:

- high throughput, so small improvements will generate large returns;
- high complexity, thus making it difficult for operating personnel to consistently make correct decisions;
- frequent changes in disturbances, raw materials, or economic market position, requiring frequent changes in the goals of plant operation.

Perhaps the best indication of the acceptance of the supervisory concept lies in the number of repeat orders for such systems.

The main obstacle to the installation of supervisory systems is that mathematical models of plants are seldom available beforehand. Thus, the project must justify the expenditure of funds for this effort, which is by no means minimal. This effort typically requires the work of several engineers for periods of a year or more, plus plant tests and additional laboratory data. In many cases the computer is installed early in order to promote this work. It can easily amount to 25 percent or more of the total project cost.

The Hierarchy Concept

In the immediately preceding sections two distinct approaches toward the application of computer control to process units were presented. In reality the final system is usually a hybrid between the two, containing parts of both. It seems that most applications have been supervisory in nature, but still incorporating a few of the real attractive DDC loops.

The above discussion presents supervisory control as a digital computer providing the set point for analog control loops. However, the same principle applies to the case in which the supervisory digital computer supplies the set points to the DDC computer. In fact, DDC becomes even more attractive in these cases for these reasons:

—Few if any additional analog inputs are required. Thus, this cost only appears once in the total configuration.

—The communication between the two digital computers is considerably superior to the communication between the digital and analog systems.

—The backup problem in DDC can be solved by allowing the supervisory computer to assume control of critical loops in case of failure of the DDC computer.

—The Exner patent apparently does not cover this application.

This concept has certainly been promoted by some of the new features of third-generation computers, such as multiple-ports to memory.

The extension of these concepts to higher and higher levels gives the "hierarchy concept" or the "automated company," as illustrated in Figure 6 [2.15, 2.19, 5.12, 6.7]. The lowest level is occupied by the DDC computer, being responsible for the control of a single plant. On the next step is the supervisory computer, which may be responsible for several individual plants.

The next step up the ladder is to a computer responsible for coordinating an entire complex of several plants. In such a complex the shipping of materials from one plant to another means that the operations in one are highly dependent upon the operations in another. It also follows that what is best for an individual plant in the complex may not be best for the complex as a whole. As the company is only interested in the total return on investment from all the plants, there may be enough incentive to install a computer to ascertain what operating goals in each plant provide the maximum return from the whole complex. These goals are forwarded to the supervisory computer, which in turn calculates the set points for the DDC computer.

At the top of the ladder is the corporate level control computer, which should logically be a part of the management information system. This computer makes available to management an up-to-date status report on the operation of the entire company. On the basis of current market information, it could have the ability to make some policy decisions, which are then communicated to the computers on the lower levels. It could also receive other policy decisions from management, and likewise pass them along. In fact, one of the incentives to install such a system is that decisions at the corporate level could be disseminated to the individual plants in a matter of hours instead of the customary weeks or months. In this way, large companies could be as responsive as small ones to market conditions.

HARDWARE

The Computer System

Now that we have discussed current approaches to implementation of computer systems, we shall examine the computer hardware usually selected. We shall concentrate on the first two levels (i.e. DDC and supervisory) of computer control systems, as that is where most of the effort is being expended at this time. These computers tend to be specialized in one way or another, whereas the computers at the higher levels will probably be very much like those in current scientific data processing centers except for a few real-time features.

For a typical process control application, four approaches could be considered:

1. Use a very large, general purpose computer, for which process control would be only one of its many functions.

2. Use a general purpose computer for which process control would be its primary function, although it could in some cases provide a limited amount of other services.

3. Use a general purpose computer but dedicate it to the sole purpose of process control.

4. Use a small special purpose computer capable of providing only a limited number of control functions.

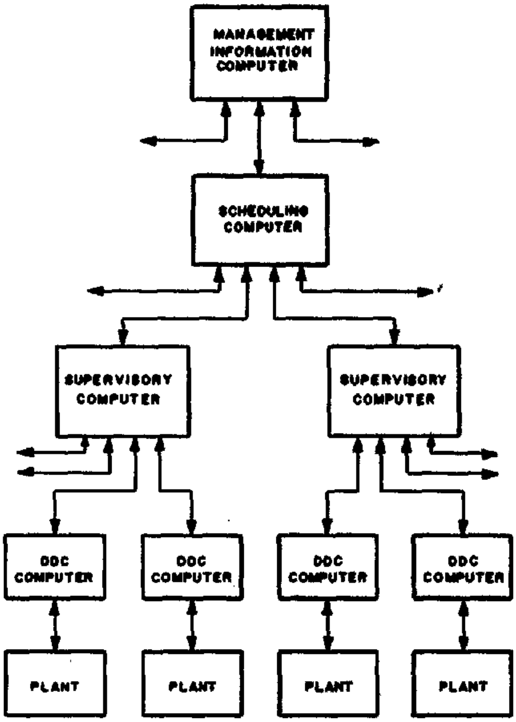


FIG. 6. Hierarchy of control system

As present trends seem to favor the second and third approaches, subsequent discussion will apply only to these. The first is rarely used, but the fourth is becoming more attractive as small computers in the $10,000–$30,000 price range become available. These can be used to capitalize on cases when one or two applications in a plant offer large returns, while all other aspects are marginal.

Virtually all control computers presently being installed are fixed word length, binary machines. The two popular word lengths are 16 bits or 24 bits. Factors that must be considered in choosing between these two are [5.20, 5.40]:

—The maximum resolution of input data from the process is typically 14 bits, and often less. Thus, a 16-bit word length is quite adequate.

—With a 16-bit machine, two words are required to store a floating-point number.

This forces the user to store his data insofar as possible in integer form (one word). With a 24-bit machine, a floating-point number can be stored in one word. Although this word only allows about 4 digits of precision, this is adequate for most process control work.

—A 24-bit machine can direct-address up to at least 16K words of core. This means that indirect addressing will be used much less than for a 16-bit machine. Furthermore, many control computers have 16K or less core storage, which means that 24-bit machines can direct address all of core storage in these cases.

—A 24-bit machine can offer a far more powerful instruction set which leads to more efficient programming and core utilization. Also, most instructions can be single words, while in a 16-bit machine a good many instructions must be double words, i.e. 32 bits versus 24 bits.

—The core storage is more expensive for 24-bit machines.

The proper word size is certainly a function of the specific duty the machine is to perform. For DDC, 16 bits (or even 12) are adequate for most situations. On the other hand, for supervisory control where floating-point numbers are used freely, the 24-bit machines become more attractive.

Other features of the CPU and memory worthy of note are:

- Parity. Parity checking is used with every word in storage.
- Storage protect. In current machines a protect bit is provided with each word in storage to provide the capability of protecting certain areas of core from "runaway programs." Future machines are expected to incorporate paging hardware and fencing schemes.
- Operations monitor. Often called a "watchdog timer," this timer must be frequently reset by the operating programs. If a program gets "hung-up" and the timer is not reset, an alarm is sounded.
- Priority interrupts [5.7, 5.52]. These can divert the CPU's attention from the normal program execution sequence in order to attend to other duties. Upon completion of these, provision is made for the resumption of the normal program execution sequence. Two types of interrupts exist:

a. Interrupts originating within the computer system synchronize various functions, frequently I/O operations.

b. Interrupts originating in the process synchronize the computer system with the outside world, notify the computer of emergency situations, or in general request the computer's immediate attention.

- Real-time clock. This clock's primary function is to coordinate the machine's operation with the real world's time.
- Integral I/O processor, also called "cycle-stealing" I/O. With this feature, the CPU does not have to be devoted to I/O chores.
- Contact closures. Discrete inputs in the form of contact closures can enter directly into the CPU. For a 16-bit machine, 16 of these enter in one word, normally requiring only one instruction cycle to be read. As this is much faster than for analog inputs, these are used to the maximum extent possible.
- Power fail safe. This enables the contents of the working registers to be stored so that operations may be resumed from that point when power is restored.
- Hardware floating point arithmetic. Not available on many computers designed for process control, it is probably not necessary for strictly DDC systems. In supervisory systems where mathematical operations such as linear programming are employed, this item may decrease program execution time significantly.
- Instruction repertoire. This varies considerably from one machine to another, ranging from the twenties to more than a hundred.

The same peripheral devices available for data processing machines are also available for control computers. Considerations in the selection of these devices are:

- Card read/punch versus paper tape read/punch. Card I/O is considerably more expensive than paper tape I/O. However, program changes are much easier to insert into a card deck than into paper tape.

Many early systems were paper tape oriented, but the current trend is toward cards.

- Teletype. Most process control systems have a teletype or a 10- to 15-cps typer for systems messages in the computer room, and often one or more additional units in the field for presenting information or messages to the plant operating personnel.
- Disk or drum storage. Essentially all supervisory systems and some DDC systems use these devices to store programs and data used only intermittently by the control system. Typical mass storage capabilities range from a million words up. Average access times range from on the order of 10 msec for fixed head devices to about a half-second for movable head units. In cases where program excution must begin at once, the access time determines whether the program can reside on disk or must reside in core. The frequency of use of a program is a second factor to consider when deciding where a program is to reside.
- Line printer. Essentially no DDC systems and few supervisory systems produce enough printed output to justify these devices. However, during the initial programming stages, the volume of output is much higher than normal, making it attractive to lease a line printer initially with intention of removing when the volume of output declines.
- Magnetic tapes. Disk or drum storage is virtually always selected over magnetic tape units for control computers, primarily because of the shorter access time with disk or drum.

This list does not include some special output devices such as incremental X-Y plotters found on some systems. Nor are devices such as operator stations or cathode-ray tube devices included, as they will be discussed subsequently.

The Analog Front End

In order to function in a proper manner, it is necessary for the control system to receive an adequate amount of reliable information from the process. The bulk of this information originates in the process in the form of continuous (analog) signals. Before information in this form can be entered into core storage, the continuous signal must be sampled (i.e. read at a discrete instant of time) and quantized (i.e. converted to digital form). These important functions are performed by the analog front end.

The information transfer from process to computer begins with a sensor (or transducer). This device senses a process variable such as temperature, pressure, flow rate, etc., and outputs a voltage signal proportional to its value. The computer control system, whether supervisory or DDC, generally demands more accuracy than does a conventional analog system. The rationale here is quite simple. To justify the expenditure for the digital system, its resulting performance must be superior, which in turn requires better data from the process. In some cases this necessitates better instruments; in others a regular maintenance and calibration effort. In fact, it is often beneficial to incorporate features in the hardware and software systems to aid in calibrating instruments.

Figure 7 gives a schematic diagram of a typical analog front end configuration. Two types of analog signals originate from the sensors:

(a) High level signals. Usually defined as any signal greater than about one volt, these signals originate from transmitters (with process sensing elements) capable of outputting a significant amount of power with the signal.

(b) Low level signals. Typically down in the millivolt range, these signals originate from sensors such as thermocouples or resistance thermometers. These signals are especially prone to distortion, requiring special shielding from noise and careful handling by the analog front end. Although perhaps the majority of the analog signals in present control systems fall into this category, this will surely diminish as integrated circuit amplifiers begin to be incorporated into sensing elements.

Depending upon the type of signal, the analog front end configuration varies, as will be discussed subsequently.

The function of each device in the analog

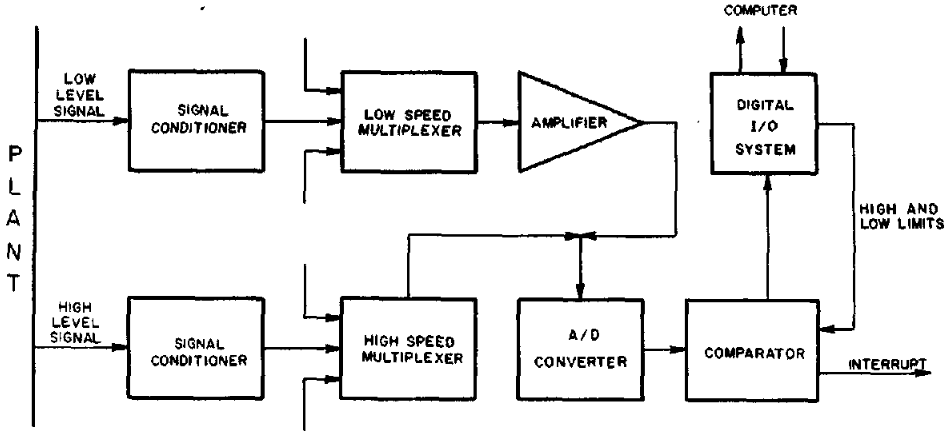


FIG. 7. Analog input system

front end illustrated in Figure 7 is [5.3, 5.11, 5.18, 5.24, 5.25, 5.33, 5.39, 5.41, 5.44, 5.47]:

Signal conditioning. As virtually all signals are to some extent influenced by noise during their transmission, it is usually desirable to remove some of this upon entering the analog front end. This is usually accomplished by a passive network such as an RC circuit.

Multiplexer. This device is effectively the sampler in the digital control loop. It selects the proper signal for conversion to digital form for entry into the computer. The number of inputs to a single multiplexer ranges from about 256 up to about 2048. The switching speed of the multiplexer depends upon the type of analog signal:

(a) For high level signals the switching is accomplished by solid state field effect transistors (FET). Input rates on the order of 10,000 points per second (10 kc) or higher (perhaps 250 kc in the near future) are possible with such devices.

(b) For low level signals, the distortion of field effect transistors cannot be tolerated. Conventional mercury-wetted or reed delays must be used, reducing the input rate to a few hundred points per second or less.

In most cases, all high level inputs are connected to a high speed multiplexer, and all low level inputs connected to a separate low speed multiplexer. Further switching is provided prior to the A/D converter to select the proper multiplexer.

Amplifier. The amplifier converts low level signals to high level prior to the A/D converter. Normally the amplifier follows the multiplexer so that it may be shared to reduce equipment cost. However, if a low level signal is to be sampled rapidly, an amplifier may be inserted in the signal lines prior to the high speed multiplexer.

A/D converter. This device converts the signal from the amplifier or multiplexer into digital form. The multiplexer selects a given signal, which must remain connected to the input of the converter long enough for its output to settle to a constant value. This time can be as low as 20 μsec for 5- to 10-v analog signals.

Comparator. This device compares the input signal with high and low limits stored in the computer's memory. If either is violated, an interrupt to the computer is generated. This comparator is a hardware device external from the CPU, thus freeing the arithmetic unit from this duty.

The control of the analog front end may reside with the CPU, or some of the control may reside with control modules external to the CPU, thus relieving it of some duty. With cycle stealing I/O or multiple ports to memory, it is feasible for the control module to assume all control function, thus freeing the CPU.

Digital Inputs

Two types of digital inputs, namely priority interrupts and discretes, have already been mentioned. Another commonly used digital input is to a pulse counter. This is a hardware device that can operate in either of two fashions:

(a) The register in the counter is initialized to zero, and then incremented at each pulse. At the end of a specified time, the counter is read by the computer.

(b) The register is initialized to a given value, and then "downcounted" (i.e. subtracted by one) at each pulse. When the register reads zero, an interrupt is generated.

These devices are commonly used in conjunction with magnetic flow meters, tachometers, and other similar measuring devices.

Outputs to the Process

Outputs from the computer can generally be classified into three categories: (a) voltage outputs, (b) pulse outputs, and (c) analog outputs [5.15, 5.35, 5.38]. Each of these have their individual uses. The first two types of outputs are found on practically all process control installations, some being able to operate without analog outputs.

Voltage outputs are simply high or low level discrete outputs that can be used to position or operate a relay. These are perhaps the simplest of all the outputs. Uses are typically for controlling on-off devices such as electrical motors, solenoid valves, etc. These are the least expensive, and are utilized whenever possible.

Pulse outputs are typically used for driving stepping motors. These are widely used in the automatic set point stations for the analog loops in a supervisory control system, but are also found on motor-driven potentiometers, valve positioners, and other final control actuators. The pulse generator is normally external to the CPU. A typical unit consists of a register which is loaded by the CPU with the number of pulses to be generated. The pulse generator then generates a pulse at regular intervals of time, down counting the register until it reads zero. The times between pulses and pulse duration is preset in the less expensive units, although some offer a choice of timing sequences. These units may also be shared between several stepper motors by multiplexing the output of the pulse generator.

One of the primary considerations in process control installations is safety in case of computer failure. As the stepper motors maintain their present position until another pulse is received, their positions remain at their last valid settings in case of computer failure. It is also convenient to arrange for a manual station to operate in conjunction with the computer set point station. These considerations frequently make these devices more attractive than digital-to-analog converters.

Analog outputs are obtained from digital-to-analog converters, also commonly called digitally set pots. This device consists of a resistor lattice network with relays or solid-state devices driven by the bit pattern of the input word. These are high impedance devices, providing an accurately proportioned output voltage by dividing a stable reference supply voltage. Although the output of these devices may also be multiplexed, a hold device must be supplied on the output of each channel of the multiplexer to maintain the output voltage between sampling instants, which makes multiplexing economically unattractive.

Operator's Console

One of the most important parts of the computer control system is the operator's console, the device through which the plant operating personnel communicate with the computer [5.23, 5.27, 5.31, 5.49, 5.50]. Human engineering must take the spotlight in designing this device, because information transfer must take place in both directions in a clear, easy-to-use manner. Through this console comes information to the computer such as desired operating conditions, results of laboratory analyses, etc. Similarly, the operating personnel must be able to request information such as values of specific variables in engineering units, current trends, predictions of the effect of a certain proposed action, etc.

The design must everywhere emphasize the convenience of use by the operating personnel. Few of these personnel are engineers, programmers, or equivalent, which increases the burden of designing the console so that the transfer of information is effective in both directions. Extensive coding of information is not readily accepted by these people, nor are intricate mechanisms for obtaining requests or inserting information.

To meet the design objectives of the operator's console, cathode ray tube (CRT) display units are ideally suited. These devices can display trends, present possible choices, and in general present information of all types in a much superior fashion than economically feasible otherwise. For example, a schematic diagram of the entire process could be displayed to the operator. He could then (with a light pen) select a portion to be displayed in greater detail, perhaps showing all sensors and control devices. Those sensors in alarm could perhaps blink to attract his attention. Again, with the light pen he could select specific variables to be displayed or other variables (such as set points) he would like to specify. In general, the future of CRT devices in this area is very promising [5.46, 5.49, 5.50].

SOFTWARE

Software Requirements

One of the main question marks in practically every digital control venture to date has been the software. Costs of user written software have been consistently underestimated; computer configurations have been selected that could not support the software needed (a typical situation is not enough core storage); and users have not always recognized the capabilities and limitations of vendor supplied software.

The record has been so consistent in this area that it has become the focal point in the analysis of every computer control venture (see [4.1–4.14]).

There are several reasons for this. First, every process is unique, requiring a considerable amount of custom software development. Second, the software must be revised as more and more information is learned about the process itself. This has negated the concept of doing a once-for-all programming job. Third, the real-time nature of the computer control task requires software support over and beyond that of conventional software.

At this point it should be understood that the terms software and programming (coding) are not used interchangeably. The coding aspect is indeed quite similar to coding for other applications, as will be discussed briefly in a subsequent section. Although the coding task should not be downgraded, the development of the philosophy and general operating characteristics of the complete set of computer programs is a major and extremely important task. The success of the entire project hinges upon software design and development.

To some extent, the desirable features of the software systems differ between DDC and supervisory systems. In DDC systems, the tasks to be performed are customarily relatively simple, but must be performed at very frequent intervals. Furthermore, while the total number of tasks may be large, many of these are identical or only minor variations of other tasks. On the other hand, a supervisory system usually contains a lesser number of tasks, but each is more complex than a typical task at the DDC level. As the complexity is higher, few of them even approach being similar.

While the demands on the software packages differ between DDC and supervisory systems, both must operate in real-time. In the ensuing discussion, the features of software for supervisory systems will be examined, followed by discussion of the points where DDC systems differ.

Software for Supervisory Control

The heart of a software package for a supervisory control system is a real-time executive or monitor system. This executive must, among other things, be capable of processing interrupts, scheduling program execution, and utilizing auxiliary storage (disk or drum) in an efficient manner. Executive systems of this type are available from several vendors.

For sake of discussion, this executive can be divided into three parts: (1) interrupt servicing, (2) real-time sequencing, and (3) off-line monitor. A simplified flowchart of such a monitor is shown in Figure 8. Note that there are two modes of operation: a normal mode for executing real-time programs and an interrupt mode for servicing interrupts. We shall now discuss briefly the three parts of this monitor.

Interrupts. A typical process control computer could have from two to about sixteen levels of interrupts. With each level of interrupt is associated a priority. That is, an interrupt that occurs on a higher level of priority takes precedent over all others. As real-time programs being executed in the normal mode can be interrupted from any interrupt level, they effectively operate on the lowest priority (aside from off-line work under the off-line monitor). To illustrate the sequence of events under this type of arrangement, consider the sequence of interrupts under the two-level-of-interrupt system illustrated in Figure 9. The first pair of interrupts illustrates an interrupt on the highest level interrupting the servicing of a lower level interrupt. The second pair of interrupts illustrates that when an interrupt on a lower level occurs while a higher level interrupt is being serviced, the lower level interrupt is not serviced until completion of servicing all higher level interrupts. Similarly, the last pair of interrupts illustrates that an interrupt cannot interrupt the servicing of an interrupt on the same level.

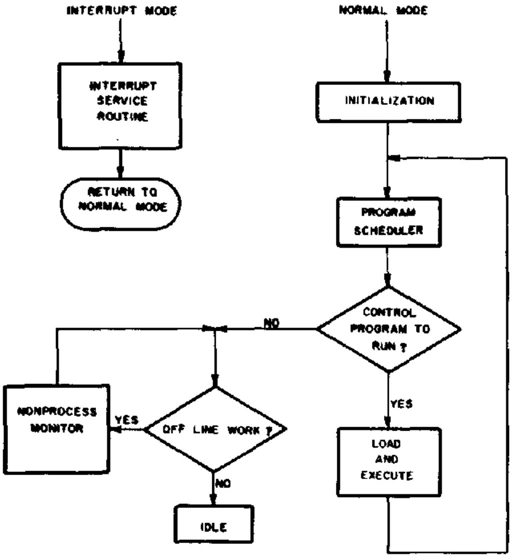


FIG. 8. Abbreviated representation of a real-time executive [1.14]

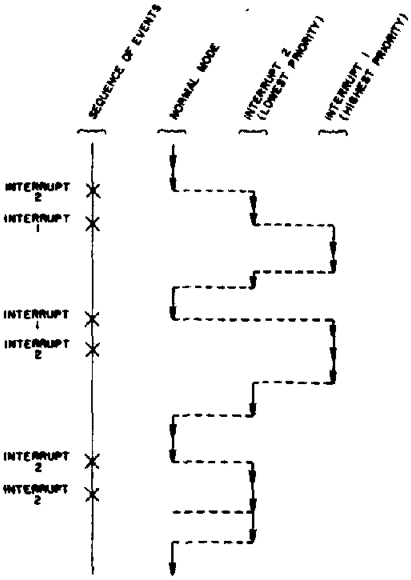


FIG. 9. Typical interrupt sequence

In general, the arrangement of the interrupts and their service routines is not a simple matter. In most cases, interrupts occurring within the computer system occupy the highest levels. Process interrupts may generally be arranged in a variety of ways, and only careful, simultaneous consideration of process characteristics and software development aspects can lead to the most successful operation. The complexity obviously increases as the number of interrupt levels increases. It is conceivable that using fewer levels of interrupts can simplify the software and program logic to the point that a more successful overall operation is obtained. The problem of choosing which interrupts should occupy which levels is also complicated by the fact that, in some systems, the manner in which the interrupts are wired on a given level determines a subpriority for that level. That is, suppose that two interrupts occur on a lower level while a higher priority interrupt is being

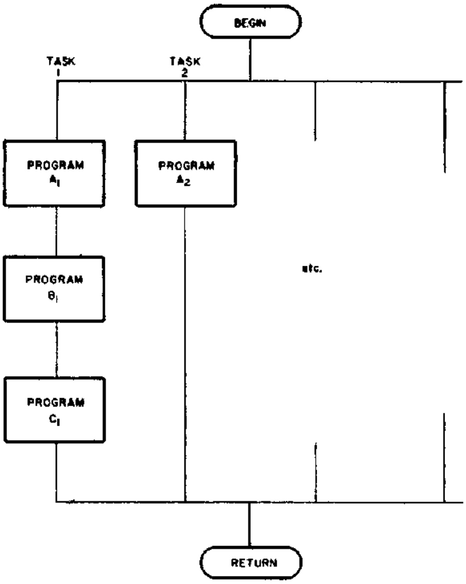


FIG. 10. Organization of tasks

serviced. Which one of the lower level interrupts is serviced first depends upon the manner in which they are wired, not on which occurs first in time.

With each interrupt level, there is set aside sufficient storage in a prespecified location for storing the contents of the working registers so that the program being executed at the time the interrupt occurred can be resumed. If this function is not hardware mechanized, the executive system must accomplish this with software. Program control is then transferred to some designated location in core storage to identify the specfic cause or origin of the interrupt. At this point, a specific program that dictates a specific action to be taken is executed. This action may perform all functions needed, or may simply specify a program to be executed by the real-time monitor. Upon completion of all of these functions, the interrupt is said to have been serviced. Control then reverts back to the program being executed.

It is, of course, conceivable that a certain sequence of events should not be interrupted at all, or perhaps only by extremely urgent interrupts. This condition is obtained by the use of MASK and UNMASK routines. The effect of a MASK is to prohibit the servicing of interrupts on a given level(s) until this condition is removed by an UNMASK routine. This gives the programmer a degree of control over the servicing of interrupts.

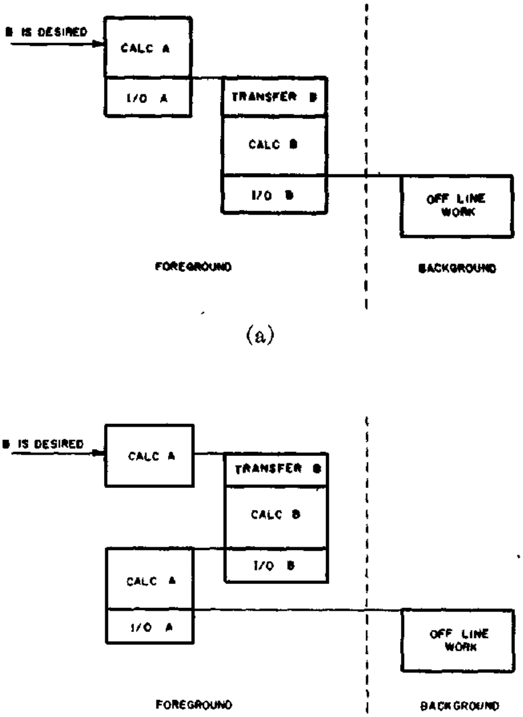
Program sequencing. The execution of programs by the process control computer generally follows no set pattern. Instead it is highly dependent upon process conditions, operator requests, etc., although some programs are executed repetitively at certain intervals of time. To some extent, the operating programs for a process control computer can be organized into groups to perform certain tasks, as illustrated in Figure 10. The completion of a single task may require one or more individual programs. Similarly, the same program may appear in the sequence of programs to perform two different tasks. Each individual program within a given task is assigned its own individual priority, i.e. it is not necessary for all programs within a given task to be assigned the same priority.

In a real-time environment, the executive must be capable of scheduling program execution such that all desired tasks are accomplished. The programmer dictates which individual programs within each task take precedence by assigning them a priority, usually under program control (i.e. not necessarily assigned when the program is written). This sequencing of program execution is accomplished with the aid. of a table called QUEUE. The names of programs to be executed are entered into QUEUE under program control, their priority usually being assigned at this point.

In many cases, servicing an interrupt consists of nothing more than placing the name of some program into QUEUE. This program is usually the first program in a sequence to perform a given task, thus initializing the task. At the completion of this program, the final instructions place the name of its successor in QUEUE, thus propagating the chain. It is also possible for a program in one task to call for another task to be accomplished, thus providing another mechanism for initializing a task.

As all interrupts take priority over programs being executed from QUEUE (unless the program has executed a MASK instruction), it is conceivable that an interrupt service routine could insert into QUEUE the name of a program of higher priority than the program currently being executed. Some process control computer executives, particularly the early ones, do not terminate the execution of a lower priority program that is in progress at the time a higher priority program is inserted into QUEUE. In many applications this is quite satisfactory, although executives are now appearing with the capability of automatic suspension, i.e. the termination of lower priority programs when higher priority entries are made into QUEUE.

In supervisory systems, the executive must also be responsible for effective utilization of mass storage. In most cases, the executive must transfer a program from disk to core before it can be executed. This is illustrated in Figure 11 for executives with and without automatic suspension. In each case, the computing system has cycle stealing I/O, allowing I/O operations (including transfer from mass storage) to be overlapped with program execution. As illustrated in Figure 11(b), the automatic suspension feature reduces the time between the point that the execution of program B is required and the execution actually begins. However, if program A were executing some task that should not be interrupted once in progress (even though the task may be of low priority), the automatic suspension feature may cause some problems unless this was recognized and prevented by the programmer.



(a)

(b)

FIG. 11. Programming sequencing: (a) without automatic suspension; (b) with automatic suspension

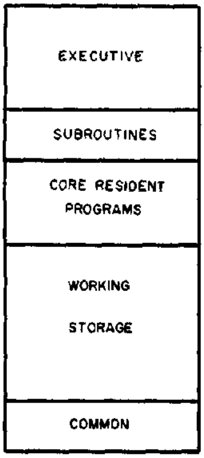


FIG. 12. Allocation of core storage

Allocation of core storage typically resembles that illustrated in Figure 12. The real-time executive typically occupies the lower portion of core storage. Frequently used subroutines, e.g. floating-point subroutines, I/O subroutines, etc., are also located in core storage. Frequently used programs whose execution must proceed quickly are also located in core. These are called core resident programs. All subroutines called by core resident programs must usually be core resident also. A section of core storage is also assigned as a common area for storage of variables whose values are required frequently. Long tables or seldom-required data are stored in files in mass storage. The remaining core storage is called working storage or variable core.

Depending upon the particular executive, this working area may contain only one program or may contain several. Systems that may contain only one program in working storage at any one time usually store these programs on disks as core loads, which include all subroutines not located in permanent core storage. Thus, locating more subroutines in core storage decreases the size of the core loads and, consequently, the time for transfer from mass storage to working storage. Executives with the automatic suspension feature usually permit more than one program to occupy working storage at any given time. If room is not available for the highest priority program, lower priority ones must be removed to the disk to be resumed when space is available.

One desirable feature not always available on real-time executives is the ability to add, modify, or delete core resident programs while the computer is performing its normal control functions. If this capability is not available, control must be transferred to the backup system, and the time required to modify the system is charged as down time against the computer control system. Obviously, this prevents the programmer from inserting frequent changes, although they may be highly desirable, into the control programs. It also impairs the control engineer's freedom to experiment with new control strategies.

Off-line monitor. The free time when the process control computer is not busy with control functions is utilized for off-line (noncontrol) functions under the off-line monitor. Although this is usually referred to as time-sharing by the process control computer manufacturers, this is not time-sharing in the context used for large computer systems. Instead, it is closer to a foreground/background operation, with real-time programs operating in the foreground and off-line work in the background. Except for the cycle stealing I/O, the processor's attention is devoted to a single program at any given time. This was illustrated in Figure 11.

Programs such as the FORTRAN compiler, the assembler, disk utility routines, and other supporting functions to the process control system are operated under the off-line monitor. The core loads for execution in real-time are built under the off-line monitor and stored on disks for subsequent use. Another important program operated under the off-line monitor is the simulator, which allows the testing of a real-time program under a simulated control environment prior to being incorporated into the system. Of course, programs in no way associated with the control system can also be executed under the off-line monitor.

Software for DDC

To some extent, many of the previous comments about software for supervisory systems also apply to DDC systems. But being first level in nature, DDC systems require software packages that emphasize speed, quick response, reliability, and efficiency. To be competitive with analog systems, this must typically be accomplished with a minimum expenditure on computer hardware. This frequently prohibits the addition of mass storage devices such as disks, which often cannot meet the quick response requirements anyway. This simplifies the executive system to some extent, as the disk routines are not required and all programs, subroutines, and data are core resident.

DDC software systems must typically accomplish relatively simple tasks (for example, control algorithms) at frequent intervals of time. To some extent, the duties to be performed vary little from one application to the next. As will be described in a subsequent section, this makes standard software packages in the form of interpreters quite attractive. In fact, perhaps ninety percent of the software required for DDC will be available in standard packages in the near future.

As DDC software is analogous to software for supervisory systems in other aspects, no further discussion seems appropriate.

TABLE I. ASSEMBLY VERSUS COMPILER LANGUAGES*

Language	*Advantage*	*Explanation*
Assembly	Fast object code	Fewer instructions to convert into machine code decreases execution time
	Efficient memory utilization	Assembly code can take advantage of memory-conserving features of modern control computers
	Control over program and data location	Assembly code offers more flexibility in specifying program layout and data storage
	Access to all computer functions and instructions	Programmer can take advantage of his detailed computer knowledge to write more effective control programs
	Efficient program linkage	Calling up subroutines and shifting control parameters is simpler
	Ability to use different classes of codes	Reentrant routines for servicing priority interrupt are facilitated
Compiler	Machine independent and standardized	A limited advantage
	Self-documenting	Yes, but must be supplemented
	Easier to learn	Yes, for a scientist or engineer
	Quicker, less tedious to write or modify	Yes, provided the program writer knows when to provide control alternatives
	Easier to debug—self-checking	Prevents some programmer errors

* From [6.17]; reproduced by permission.

Programming Languages

Assembly. Although used almost exclusively on earlier process control computers, the proportion of control programs written in assembly is decreasing in both supervisory and DDC applications. However, as the assembly language is the most efficient from execution time, it is used more frequently in DDC, where speedy response and fast action are more important than in supervisory systems. The use of assembly language will surely continue to decline, as the high personnel requirements for this type of programming cannot be tolerated in a market where this talent is already in short supply. Instead, the trend will be toward larger, faster systems where programming efficiency is not as critical. Table I gives a summary of the advantages of assembly coding versus use of a compiler such as FORTRAN [6.17].

FORTRAN. The bulk of programming of supervisory systems is currently being done at the FORTRAN level. Although programming in this language is not as efficient as in assembly, it is understood by most engineers graduating today. One solution to permitting FORTRAN to be used most of the time and assembly only when the rewards are high is the provision of a FORTRAN compiler that accepts in-line assembly statements. The FORTRAN compiler converts the FORTRAN statements to assembly, with the assembler taking over from there. This type of operation also permits the programmer to "dress up" the assembly code without having to prepare the entire program in assembly.

FORTRAN compilers available on process control computers are not as general as on larger machines, although they are FORTRAN IV as compared to FORTRAN II. The features generally supported include COMMON, DATA, EQUIVALENCE, DIMEN-

TABLE II. STANDARD LANGUAGE CHARACTERISTICS*

1. Relates to English in defining variable names
2. Defines the solution of problems in the same form as the implementation of the problems
3. Efficiently gathers process data and regulates variables with several algorithmic options
4. Incorporates routine decisions and optimization
5. Data outputting capability and transmission to other computers
6. Free-time calculation capability
7. Formulation capability of the coordinator of all functions
8. System configuration generation
9. Ease of debugging in English
10. Ease of maintenance
11. Self-documentation
12. Machine independent until the machine implementation stage of the project
13. Flexible
14. Must run in a "reasonable" size computer
15. Free formatting
16. Capability of assembly language statement insertion
17. Usable in a management information system
18. Modular flexibility
19. On-line data update capability
20. Easy interface with operator's consoles
21. Interrupt manipulation capability
22. Short execution time and easy modification of programs
23. Reflection of process variable failures throughout the whole system
24. On-line program correction capability
25. Arithmetic instruction capability
26. Program linking
27. Flexible data structures
28. Dynamic storage allocation
29. Time delay capability

* From [6.25]; reproduced by permission.

SION, REAL, INTEGER, DEFINE FILE, READ, WRITE, arithmetic IF, FORMAT, CALL, RETURN, FUNCTION, SUBROUTINE, plus the usual arithmetic operations. Notable omissions are logical IF, double precision, complex, and other special features. Although not selective within a program, double or extended precision can be used for *all* floating-point operations within a program. Similarly, *all* integer variables may be stored in either one word or two words, the former being selected for most cases. Most of these restrictions arise because process control computers are small relative to their large cousins in data processing centers. Similarly, their FORTRAN compilers do not optimize the code as well as their larger cousins. This could potentially be circumvented by the availability of a more general, more efficient compiler that operates on a large system but produces a code executable by the process control computer.

The real-time features are frequently inserted into the FORTRAN programs by callable subroutines, although some compilers accept special statements for real-time functions. Examples of such functions are reading the real-time clock, setting interval timers, masking or unmasking interrupts, placing programs in QUEUE, etc.

Higher level languages. Currently, there is considerable interest in improving and standardizing the programming languages used on process control computers [6.5, 6.25, 6.29, 6.35, 6.36]. Table II gives the characteristics of such a language as they were formulated in a conference held at Purdue [6.5] at which both users and vendors were represented. These characteristics appear to indicate a procedural language such as FORTRAN, but obviously they go far beyond the capability of current FORTRAN'S. There is, however, another school of thought that advocates a problem-oriented or fill-in-the-forms language resembling the interpretive digital simulation languages currently available.

Problem-oriented languages. These are general programs capable of performing many of the control functions common to various applications. They take care of such functions as the analog scan, conversion of input data to engineering units, high and low limit alarm violations, and other common functions. Thus, they can indeed reduce the programming task in many cases.

Of course, there is a penalty. Being general in nature, they will be less efficient than programs written for specific objectives. This means that the computer system will probably need more core and mass storage. If timing is critical, the problem-oriented programming system may not be able to perform as well as custom programs. Further, much of the flexibility of assembly and FORTRAN languages is lost at this level. This disadvantage is minimized when routines specific to a given application can be readily added to the problem-oriented programming package.

Because practically the same basic functions are required of all DDC control systems, the future looks brightest at the DDC level. For example, the block diagram in Figure 13 illustrates the functions required to calculate a control algorithm. The actual calculations or control functions are accomplished within the blocks, their connection pattern dictating the overall task to be accomplished. The input data consists of the connection patterns to the various blocks and the constants or parameters required by each. The problem-oriented programming package consists of routines written to accomplish the task corresponding to each block plus a type of executive to keep track of the interconnection patterns [6.18].

The available blocks are generally sufficient to accomplish all common control functions, such as cascade, feedforward, etc. For functions specific to a given application, special blocks may generally be added. Sections also take care of the analog input scan, the operators console, report writing, and other common requirements of a DDC system so that the programming demands upon the user are minimized.

Although problem-oriented languages are available at the supervisory level [6.4], the differences between individual supervisory systems are such that it is more difficult to provide all the functions required. Again, however, the problem-oriented programming system can at least take care of the analog scan, engineering conversions, operator's console, report writing, and similar functions. (Many computer manufacturers also provide software for these functions, although they may not provide a problem-oriented language as such.)

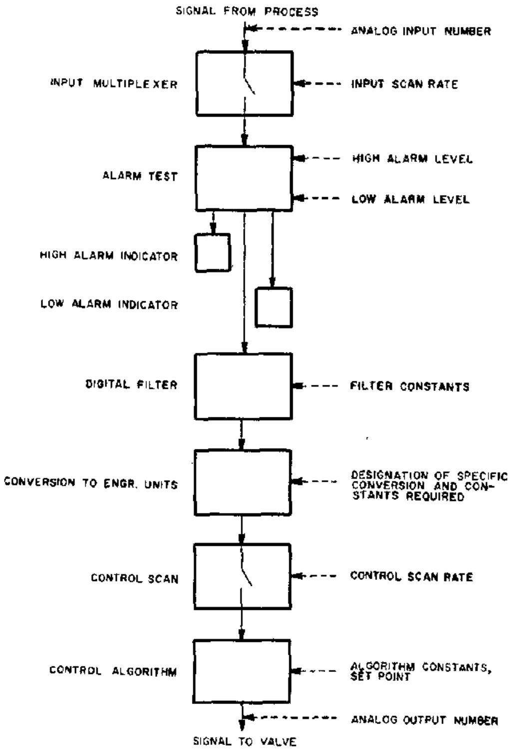


FIG. 13. Typical algorithm calculation

SUMMARY

This article has attempted to familiarize the reader with the current state-of-the-art and requirements for process computer control systems. At the moment, the future of digital computers in this area is especially bright, the rate of installation of new systems being limited primarily by the personnel available. Certainly not all aspects have been covered in this article, and several important ones were mentioned only casually. The following literature references should contain some entries on almost any subject about which additional information is desired [2.19].

BIBLIOGRAPHY

BOOKS

1.1. DESMONDE, W. H. *Real-Time Data Processing Systems: Introductory Concepts.* Prentice-Hall, Englewood Cliffs, N. J., 1964.

Discusses aspects of real-time computer systems, but is not specific to process control.

1.2. Eveleigh, E. E. *Adaptive Control and Optimization Techniques.* McGraw-Hill, New York, 1968.

Advanced text describing practical aspects for application of adaptive control, identification techniques, and optimization.

1.3. Harriott, P. *Process Control.* McGraw-Hill, New York, 1964.

Basic text on analog process control.

1.4. Johnson, E. F. *Automatic Process Control.* McGraw-Hill, New York, 1967.

Basic text on analog process control.

1.5. Jury, E. I. *Theory and Application of the Z-Transform Method.* Wiley, New York, 1964.

Devoted to the mathematics (z-transforms) specifically applicable to sampled-data or digital control.

1.6. Kuo, B. C. *Analysis and Synthesis of Sampled-Data Control Systems.* Prentice-Hall, Englewood Cliffs, N. J., 1963.

Devoted to the theory of designing digital control loops, but not specific to process control.

1.7. Lee, T. H., Adams, G. E., and Gaines, W. M. *Computer Process Control: Modeling and Optimization.* Wiley, New York, 1968.

Primary consideration is digital process control; more theoretical than Savas [1.14].

1.8. Lindorff, D. P. *Theory of Sampled-Data Control Systems.* Wiley, New York, 1964.

Devoted to the mathematics (z-transforms) specifically applicable to sampled-data or digital control.

1.9. Martin, J. T. *Programming Real-Time Computer Systems.* Prentice-Hall, Englewood Cliffs, N. J., 1965.

Devoted to programming considerations of real-time systems, but not specific to process control.

1.10 Miller, W. E. (Ed.). *Digital Computer Applications to Process Control.* Plenum Press, New York, 1965.

Collection of papers describing various aspects of computer control systems.

1.11. Monroe, A. J. *Digital Processes for Sampled-Data Systems.* Wiley, New York, 1962.

Devoted to theory of designing digital control loops, but not specific to process control.

1.12. Murrill, P. W. *Automatic Control of Processes.* International Textbook Co., Scranton, Pa., 1967.

Basic text in analog process control.

1.13. Ragazzini, J. R., and Franklin, G. F. *Sampled-Data Control Systems.* McGraw-Hill, New York, 1958.

Devoted to theory of designing digital control loops, but not specific to process control.

1.14. Savas, E. S. *Computer Control of Industrial Processes.* McGraw-Hill, New York, 1965.

Primary consideration is digital process control; includes section on applications.

1.15. Shinskey, F. G. *Process Control Systems.* McGraw-Hill, New York, 1967.

Basic text in analog process control.

1.16. Tou, J. T. *Digital and Sampled-Data Control Systems.* McGraw-Hill, New York, 1959.

Devoted to theory of designing digital control loops, but not specific to process control.

1.17. ——. *Optimum Design of Digital Control Systems.* Academic Press, New York, 1963.

Reflects inroads of modern control theory on the design of digital control systems.

ARTICLES

General

2.1. Guidelines and general information on user requirements concerning direct digital control (Users' Workshop on Direct Digital Computer Control, Princeton, N. J., April 3–4, 1963); Questions and answers on direct digital computer control (Users' Workshop on Direct Digital Control, Princeton, N. J., May 6–7, 1964). In Williams, T. J., and Ryan, F. M. (Eds.), *Progress in Direct Digital Control,* Instrument Society of America, Pittsburgh, 1969, pp. 251–258 and 259–278.

The minutes of meetings of representatives of users and vendors to formulate the requirements for DDC computers, both hardware and software.

2.2. Aronson, A. Hughes-owned patent looms large in process industries. *Control Eng. 12,* 3 (March 1965), 21.

Discusses possible effect of Exner patent on supervisory digital control.

2.3. Ast, P. A., Cugini, J. C., and Davis, R. S. New trends in handling computer control projects. *Chem. Eng. 76,* 16 (July 28, 1969), 128–132.

Discusses several aspects of the computer control project.

2.4. Bernard, J. W. Plan control at the right level. *Control Eng. 13,* 9 (Sept. 1966), 95–98.

Discusses philosophy in designing digital control systems to meet objectives of plant control.

2.5. ——, and Cashen, J. F. Direct digital control—questions that must be answered. 19th Annual ISA Conf. and Exhibt, New York, Oct. 1964, Preprint 5.4-1-64.

An evaluation of problem areas in DDC as of 1964.

2.6. —— and ——. Direct digital control. *Instruments and Control Systems 38,* 9 (Sept. 1965), 151–158.

Introductory article.

2.7. Bruce, R. G., and Fanning, R. J. How direct digital control works. *Hydrocarbon Processing 43,* 12 (Dec. 1964), 83–86.

An early application of DDC to a fractionator.

2.8. Clarke, S. L. H. An introduction to direct digital control. *Control 9*, 81 (March 1965), 127–129.

Introductory article.

2.9. Drewry, H. S. Project planning pays off. *Control Eng. 13*, 9 (Sept. 1966), 91–94.

Discusses successful planning of computer control projects.

2.10. Eisenhart, R. D., and Williams, T. J. Closed loop computer control at Luling. *Control Eng. 7*, 11 (Nov. 1960), 103–114.

Describes the results of a joint venture by Monsanto and Bunker-Ramo in computer control.

2.11. Gardner, N. F. Computer control classifications. Proc. Third Annual Workshop on the Use of Digital Computers in Process Control, Baton Rouge, La., Feb. 21-23, 1968, Rimbach Pubs., Philadelphia, 1968, pp. 100–111.

Separates applications of computer control systems into various categories.

2.12. Guisti, A. L., Otto, R. E., and Williams, T. J. Direct digital computer control. *Control Eng. 9*, 6 (June 1962), 104–108.

Describes pertinent facets of a direct digital computer control system.

2.13. Harper, S. D. Next steps in computer control. *Chem. Eng. 72*, 12 (June 7, 1965), 171–176.

Describes expected future developments in computer control systems.

2.14. Hendrie, G. C. Consider digital computer process control. *Automation 11*, 11 (Nov. 1964), 78–83.

Advocates consideration of computer control systems for process units.

2.15. Hodge, B. Company control via computer. *Chem. Eng. 72*, 12 (June 7, 1965), 177–180.

Advances the concept of hierarchical computer control systems.

2.16. Kloch, H. F., and Schoeffler, J. D. Direct digital control at the threshold. *Electronics 37*, 12 (March 23, 1964), 49–55.

Describes concepts of direct digital computer control and enumerates its advantages.

2.17. Madigan, J. M. Computer controlled processing. *Chem. Eng. Progr. 56*, 5 (May 1960), 63–67.

Discusses the applicability of computer control systems to the chemical process industries.

2.18. Ostberg, W., Fuchigams, T., and DiCisco, D. Case histories in computer control. *Control Eng. 13*, 9 (Sept. 1966), 136–142.

Discusses prior efforts in the digital computer control area.

2.19. Pike, H. E. Direct digital control—a survey. 23rd Annual ISA Conf. and Exhibit, New York, Oct. 1968, Preprint 68-840.

Survey of digital computer control, including a good bibliography.

2.20. Rosenbrock, H. H., et al. Process control and computers. *Control 9*, 81 (March 1965), 125–127.

Discusses the use of a digital computer for first-level plant control.

2.21. Rudisill, E. L. DDC-steppingstone to process optimization. *Automation 12*, 5 (May 1965), 84–87.

Advocates DDC as a first step to effective plant operation.

2.22. Ryan, F. M. (Ed.). Process computer scorecard. *Control Eng. 13*, 9 (Sept. 1966), 73–82.

Survey of digital computers in use for process control.

2.23. Stout, T. M. Process control. *Datamation 12*, 2 (Feb. 1966), 22–27.

A survey of the status of process control.

2.24. VanderSchraaf, E., and Strauss, W. I. Direct digital control: An emerging technology. *Oil and Gas J. 62*, 46 (Nov. 16, 1964), 167–172.

An introduction to DDC.

2.25. Webb, J. C. Representative DDC systems. *Instruments and Control Systems 40*, 10 (Oct. 1967), 78–83.

Discussion of typical direct digital computer control systems.

2.26. Weiss, M. D. Instrument engineer's guide to digital computer control. *Instrumentation Technol. 14*, 9 (Sept. 1967), 41–54.

An introduction to DDC from the instrument engineer's viewpoint.

2.27. Williams, T. J. Systems engineering. *Oil and Gas J. 57*, 33 (Aug. 10, 1959), 93–97.

Discusses an approach to the analysis of operating process units.

2.28. ——. What to expect from direct-digital control. *Chem. Eng. 71*, 5 (March 2, 1964), 97–104.

Discusses the economic considerations for DDC systems,

2.29. ——. Process control today. *ISA J. 12*, 9 (Sept. 1965), 76–81.

A survey and evaluation of the status of process control.

2.30. Woodley, G. V. Modeling and programming for direct digital control. *ISA J. 13*, 3 (March 1966), 48–54.

Discusses an approach to development of a DDC strategy.

2.31. Yetter, E. W., and Sanders, C. W. A time-shared digital process-control system. *ISA J. 9*, 11 (Nov. 1962), 53–56.

Discusses an application of a computer control system.

Control

3.1. Bakke, R. M. Adaptive gain tuning applied to process control, 19th Annual ISA Conf. and Exhibit, New York, Oct. 1964, Preprint 3.2-1-64.

Presents a method of adapting controller gain to a changing process.

3.2. ——. Direct digital control with self-adjustment for processes with variable dead time and/or multiple delays. 20th Annual ISA Conf. and Exhibit, Los Angeles, Oct. 1965, Preprint 30.1-1-65.

Advances the concept of dead time compensation for digital control systems.

3.3. ——. Computer helps operators tune process controller. *Instrumentation Technol. 15,* 9 (Sept. 1968), 52–56.

Presents guidelines for manual on-line tuning.

3.4. Bernard, J. W. Advanced control algorithms for DDC systems. *ISA J. 13,* 4 (April 1966), 54–55.

Presents high performance control algorithms for DDC systems.

3.5. Bertram, J. E. The effect of quantization in sampled-data feedback systems. *Trans. AIEE 77,* 4 (Sept. 1958), 177–182.

Theoretical treatment of the effect of quantization in computer control systems.

3.6. Bollinger, R. E., and Lamb, D. E. Multivariable systems analysis and feedforward control synthesis. *Ind. and Eng. Chem. Fundamentals 1,* 4 (Nov. 1962), 245–252.

Theoretical analysis of multivariable control systems.

3.7. Boxenhorn, B. Using Kalman filtering to estimate control parameters. *Control Eng. 16,* 7 (July 1969), 69–72.

Illustrates applicability of Kalman filtering to industrial control problems.

3.8. Bristol, E. H. A simple adaptive system for industrial control. *Instrumentation Technol. 14,* 6 (June 1967), 70–74.

Discusses a practical approach to adaptive control systems.

3.9. Calvert, S., and Coulman, G. Feedforward control: Its future role in the chemical industry. *Chem. Eng. Progr. 51,* 9 (Sept. 1961), 45–48.

Review and evaluation of the concept of feedforward control.

3.10. Chang, S. S. L. Discrete systems and digital computer control. *Appl. Mech. Revs. 20,* 5 (May 1967), 429–437.

Discusses the influence of such factors as quantization on computer control systems.

3.11. Chien, G. K. L. Role of computers in industrial process control. *Automation 10,* 8 (Aug. 1963), 52–59.

Discusses various ways in which computers can be used in a process control environment.

3.12. Clough, J. E. DDC algorithms in multipurpose computer control systems. 22nd Annual ISA Conf. and Exhibit, Chicago, Sept. 11–14, 1967, Preprint 10-5-ACOS-67.

Discussion and evaluation of DDC control algorithms.

3.13. Coles, H. E. Controller settings from process frequency response. *Instruments and Control Systems 39,* 6 (June 1966), 137–140.

Design of control systems using frequency response.

3.14. Cox, J. B., et al. A practical spectrum of DDC chemical-process control algorithms. *ISA J. 13,* 10 (Oct. 1966), 65–72.

A survey of proven algorithms for direct digital control.

3.15. Cundall, C. M., and Latham, V. Designing digital computer control systems. Pt. I, *Control Eng. 9,* 10 (Oct. 1962), 82–86; Pt. II, *Control Eng. 10,* 1 (Jan. 1963), 109–113.

Presentation of basic concepts used to develop digital control systems.

3.16. Dahlin, E. B., et al. Designing and tuning digital controllers. Pt. I, *Instruments and Control Systems 41,* 6 (June 1968), 77–83; Pt. II, *Instruments and Control Systems 41,* 7 (July 1968), 87–91.

Use of z-transforms to design algorithms for single- and multivariable control systems.

3.17. Davies, W. D. T. Control algorithms for DDC. *Instrument Practice 21,* 1 (Jan. 1967), 70–77.

A survey of control algorithms for DDC.

3.18. Day, R. L. The design of cascade control systems. *Trans. Soc. Instrument Technol. 10,* 2 (June 1958), 75–84.

Discusses relevant factors in designing cascade control loops.

3.19. DeBolt, R. R., and Powell, B. E. A "natural" 3-mode controller algorithm for DDC. *ISA J. 13,* 9 (Sept. 1966), 43–47.

Presents a three-parameter DDC control algorithm.

3.20. Denn, M. M. The optimality of an easily implementable feedback control system: An inverse problem in optimal control theory. *A.I.Ch.E. J. 13,* 5 (Sept. 1967), 926–931.

Given the control law, this article discusses problems for which it is optimal.

3.21. Distefano, G. P., May, F. P., and Huckaba, C. E. Transient response and feed-forward control of a distillation tower subject to a sequence of upsets. *A.I.Ch.E. J. 13,* 1 (Jan. 1967), 125–131.

Application of feedforward control.

3.22. Fertik, H. A., and Ross, C. W. Direct digital control algorithm with anti-windup feature. 22nd Annual ISA Conf. and Exhibit, Chicago, Sept. 11–14, 1967, Preprint 10-1-ACOS-67.

Presents a modification of the control algorithm to prevent reset wind-up.

3.23. Foster, R. D., and Stevens, W. F. A method for the noninteracting control of a class of linear multivariable systems. *A.I.Ch.E. J. 13,* 2 (March 1967), 334–340.

Theoretical article on multivariable control.

3.24. Franks, R. G., and Worley, C. W. Quantitative analysis of cascade control. *Ind. and Eng. Chem. 48,* 6 (June 1956), 1074–1079.

Discusses design and evaluation of cascade control system.

3.25. Gallier, P. W., and Otto, R. E. Self-tuning computer adapts DDC algorithms. *Instrumentation Technol. 15,* 2 (Feb. 1968), 65–70.

Presents a model-reference approach to adaptive tuning.

3.26. Garden, M. Learning techniques adapt DDC for valve actuators. *Instrumentation Technol. 15,* 1 (Jan. 1968), 39–45.

An approach to compensate for nonlinearities in actuators.

3.27. Gardenhire, L. W. Selecting sampling rates. *ISA J. 11,* 4 (April 1964), 59–64.

Discusses factors pertinent to selection of DDC sampling rates.

3.28. Goff, K. W. Dynamics in direct digital control. Pt. I, *ISA J. 13,* 11 (Nov. 1966), 45–49; Pt. II, *ISA J. 13,* 12 (Dec. 1966), 44–54.

Discusses a variety of factors pertinent to DDC, including actuator selection, filter selection and design, selection of sampling rate, etc.

3.29. ——. Programming computers for control—DDC algorithms, today and tomorrow. *Leeds & Northrup Tech. J. 3* (Spring 1968), 2–9.

Presents approaches to successful programming of DDC control algorithms.

3.30. Gollin, N. W. Cascade control systems. *Control Eng. 3,* 7 (July 1956), 94–98.

Survey article on cascade control systems.

3.31. Gupta, S. C., and Ross, C. W. Simulation evaluation of a digital control system. *ISA Trans. 3,* 3 (July 1964), 271–279.

Illustrates use of simulation to design and evaluate digital control systems.

3.32. Harris, J. T., and Schecter, R. S. The feedforward control of a chemical reactor. *Ind. and Eng. Chem. Process Design and Develop. 2,* 3 (July 1963), 245–252.

An application of feedforward control.

3.33. Hartwigsen, C. C., et al. Analysis of sampling rates and control settings for direct digital control. 20th Annual ISA Conf. and Exhibit, Los Angeles, Oct. 1965, Preprint 30.1-4-65.

Discusses interrelationship between control settings and sampling rate in DDC systems.

3.34. Higgins, T. J., and Howard, G. W. An exact analysis of sampled-data systems with finite sampling time. *ISA Trans. 2,* 4 (Oct. 1963), 350–358.

Theoretical treatment of noninstantaneous sampling systems.

3.35. Huber, C. I., and Kermode, R. I. Linear programming z-transform design of digital controllers for regulator systems. *Ind. and Eng. Chem. Fundamentals 7,* 1 (Feb. 1968), 158–164.

Design of control algorithms using z-transforms coupled with linear programming.

3.36. Jones, C. A., et al. Design of optimum dynamic control systems for nonlinear processes. *Ind. and Eng. Chem. Fundamentals 2,* 2 (May 1963), 81–89.

Discusses selection of control laws for nonlinear systems.

3.37. Kalman, R. E., and Bertran, J. E. A unified approach to the theory of sampling systems. *J. Franklin Inst. 267,* 5 (May 1959), 405–436.

A highly theoretical article on digital and sampled-data control systems.

3.38. ——, Lapidus, L., and Shapiro, E. Computer control of processes—mathematics is the key. *Chem. Eng. Progr. 56,* 2 (Feb. 1960), 55–61.

Illustrates key role of mathematics in development of computer control systems.

3.39. Kershow, W. V. Adaptive control by plant identification. *Control Eng. 12,* 9 (Sept. 1965), 103–110.

An adaptive control strategy based on updating a plant model.

3.40. Knowles, J. B., and Edwards, R. The effect of a finite word length computer in a sampled-data feedback system. *Proc. IEEE 112,* 6 (June 1965), 1197–1207.

Discusses effect of quantization in digital control systems.

3.41. —— and ——. Computational error effects in a direct digital control system. *Automatica 4,* 1 (May 1966), 7–30.

Analysis of the effects of computer error in DDC systems.

3.42. Koepcke, R. W. A discrete design method for digital control. *Control Eng. 13,* 6 (June 1966), 83–87.

Advances the use of a discrete model for designing digital control algorithms.

3.43. Koppel, L. B. Operational methods in sampled-data process control. *ISA J. 13,* 10 (Oct. 1966), 52–61.

A review of z-transform theory as applicable to digital process control.

3.44. ——. Optimum control of distributed-parameter processes. *Ind. and Eng. Chem. Fundamentals 6,* 2 (May 1967), 299–303.

Theoretical article treating optimal control of units such as heat exchangers.

3.45. Lapidus, L., Luus, R., and Rothenberger, B. R. The control of nonlinear systems. *A.I.Ch.E. J. 13,* 1 (Jan. 1967), 101–118.

Theoretical article treating the development of control strategies for nonlinear systems.

3.46. Latour, P. R., Koppel, L. B., and Coughanowr, D. R. Time-optimum control of chemical processes for set-point changes. *Ind. and Eng. Chem. Process Design and Develop. 6,* 4 (Oct. 1967), 452–460.

Discusses the design of a bang-bang controller for effecting set point changes.

3.47. ——, ——, and ——. Feedback time-optimum process controllers. *Ind. and Eng. Chem. Process Design and Develop. 7,* 3 (July 1968), 345–353.

Develops optimal control equation for systems with dead time.

3.48. Lopez, A. M., et al. Tuning controllers with error-integral criteria. *Instrumentation Technol. 14,* 11 (Nov. 1967), 57–62.

Presents tuning equations for analog controllers.

3.49. ——, Murrill, P. W., and Smith, C. L. Optimum tuning of proportional digital controllers. *Instruments and Control Systems 41,* 10 (Oct. 1968), 97–102.

Presents tuning equations for proportional digital control algorithms.

3.50. ——, ——, and ——. Tuning PI and PID digital controllers. *Instruments and Control Systems 42,* 2 (Feb. 1969), 89–95.

Presents tuning relationships based on integral criteria.

3.51. Lupfer, D. E., and Oglesby, M. W. Applying dead-time compensation for linear predictor process control. *ISA J. 8,* 11 (Nov. 1961), 53–57.

Presents a technique for improved control of processes with dead time.

3.52. Master, R. C. A programmable process controller. 19th Annual ISA Conf. and Exhibit, New York, Oct. 1964, Preprint 11.4-3-64.

Presents a control algorithm for digital control.

3.53. ——. Feedforward control. *Instruments and Control Systems 39,* 9 (Sept. 1966), 107–112.

Survey article on feedforward control.

3.54. Miller, J. A., et al. A comparison of controller tuning techniques. *Control Eng. 14,* 12 (Dec. 1967), 72–75.

Presents performance comparisons for analog controller tuning methods.

3.55. ——, Murrill, P. W., and Smith, C. L. How to apply feedforward control. *Hydrocarbon Proc. 48,* 7 (July 1969), 165–172.

Survey article on feedforward control; includes comprehensive bibliography.

3.56. Min, H. S., and Williams, T. J. Chemical process control in the presence of both transport lag and sampled-data control. Process Dynamics and Control, *Chem. Eng. Progr.* Symp. Series No. 36, 1961, pp. 100–108.

Discusses discrete control of systems with dead time.

3.57. Moore, C. F., Murrill, P. W., and Smith, C. L. Modifying digital control dynamics for controller tuning and hardware lag effects. *Instrument Practice 23,* 1 (Jan. 1969), 45–49.

Presents use of an effective dead time to account for effect of sampler.

3.58. Mosler, H. A., Koppel, L. B., and Coughanowr, D. R. Sampled-data proportional control of class of stable processes. *Ind. and Eng. Chem. Process Design and Develop. 5,* 3 (July 1966), 297–309.

Evaluates performance of discrete proportional control laws.

3.59. ——, ——, and ——. Application of conventional loop tuning to sampled-data systems. *Ind. and Eng. Chem. Process Design and Develop. 6,* 1 (Jan. 1967), 101–105.

Discusses tuning of digital control loops.

3.60. ——, ——, and ——. Sampled-data, proportional-integral control of a class of stable processes. *Ind. and Eng. Chem. Process Design and Develop. 6,* 2 (April 1967), 221–225.

Discusses performances of discrete PI algorithms.

3.61. ——, ——, and ——. Process control by digital compensation. *A.I.Ch.E. J. 13,* 4 (July 1967), 768–778.

Advances an effective algorithm for slow-sampling digital control systems.

3.62. Neumann, L. P., Smith, C. L., and Murrill, P. W. Time domain specifications of digital controllers. *Instruments and Control Systems 42,* 5 (May 1968), 97–100.

Presents algorithm design techniques to attain specified rise time, overshoot, and damping ratio.

3.63. Pemberton, T. J. Automatic tuning of nonlinear control loops. *Instruments and Control Systems 42,* 5 (May 1968), 123–126.

Discusses adapting loop gain for nonlinear valve characteristics.

3.64. Pessen, D. W. How to "tune-in" a three-mode controller. *Instrumentation 12,* 2 (Second Quart. 1954), 29–34.

Discusses tuning of a three-mode controller.

3.65. Pounds, J. G. A method for evaluating 3-mode algorithms. 20th Annual ISA Conf. and Exhibit, Los Angeles, Oct. 1965, Preprint 30.2-1-65.

Discusses evaluation of digital control algorithms.

3.66. Rowton, E. E. Sampled-data control of pH. *Instrumentation Technol. 15,* 6 (June 1968), 63–65.

Discusses an application of sampled-data control.

3.67. Shinskey, F. G. Feedforward control of pH. *Instrumentation Technol. 15,* 6 (June 1968), 65–69.

An application of feedforward control.

3.68. Slaughter, J. B. Quantization errors in digital control systems. *IEEE Trans. AC-9,* 1 (Jan. 1964), 70–74.

An analysis of the effect of quantization errors.

3.69. ——. Compensating for dynamics in digital control. *Control Eng. 11,* 5 (May 1964), 109–114.

Discusses design of digital control loops.

3.70. Smith, C. L., and Murrill, P. W. Controllers—set them right. *Hydrocarbon Processing 45,* 2 (Feb. 1966), 105–124.

Survey article on tuning relationships.

3.71. —— and ——. A more precise method for tuning controllers. *ISA J. 13,* 5 (May 1966), 50–58.

Advances new tuning relationships.

3.72. —— and ——. Analytical tuning of underdamped systems. *ISA J. 13,* 9 (Sept. 1966), 48–53.

Discusses tuning of systems with oscillatory response.

3.73. —— and ——. The dynamics of spot samples. Pt. I, *Hydrocarbon Processing 46,* 12 (Dec. 1967), 109–114; Pt. II, *Hydrocarbon Processing 47,* 1 (Jan. 1968), 155–160.

Review article on z-transform mathematics.

3.74. Spitz, D. A. Optimal process control of a single variable. *Instruments and Control Systems 42,* 8 (Aug. 1968), 87–90.

Discusses the control performance in light of frequency content of the signals.

3.75. Terao, M. Quantization and sampling selection for efficient DDC. *Instrumentation Technol. 14,* 8 (Aug 1967), 49–55.

A theoretical analysis resulting in guides to selection of sampling rate and required accuracy in control.

3.76. Thayer, D., and Cohen, E. M. Tuning direct digital control. *Instruments and Control Systems 40,* 10 (Oct. 1967), 85–88.

Presents approaches to the tuning of DDC algorithms.

3.77. Vander Grinten, P. M. E. M. Control effects of instrument accuracy and measuring speed. *ISA J. 12,* 12 (Dec. 1965), 48–50.

Advances the concept of control efficiency to analyze the effect of instrument errors.

3.78. Wells, C. H. Application of modern estimation and identification techniques to chemical process. Joint Automatic Control Conf. [Amer. Automatic Control Council], Boulder, Colo., Aug. 5–7, 1969 (Am. Inst. of Chem. Engineers, New York, 1969), pp. 473–481.

Illustrates the applicability of estimation techniques to a stirred chemical reactor.

3.79. —— and Larson, R. E. Combined optimum control and estimation of serial systems with time delay. Joint Automatic Control Conf. [Amer. Automatic Control Council], Boulder, Colo., Aug. 5–7, 1969 (Am. Inst. of Chem. Engineers, New York, 1969), pp. 23–33.

Discusses application of estimation theory to process-type units.

3.80. Wills, D. M. Tuning maps for three-mode controllers. *Control Eng. 9,* 4 (April 1962), 104–108.

Illustrates the effect of control parameter on system's time response.

3.81. ——. A guide to controller tuning. *Control Eng. 9,* 8 (Aug. 1962), 93–95.

Discusses tuning of analog controllers.

3.82. Zalkind, C. S. Practical approach to noninteracting control. *Instruments and Control Systems 40,* 4 (March 1967), 89–94; *40,* 4 (April 1967), 111–116.

Discusses approaches to the design of multivariable control systems.

3.83. Ziegler, J. G., and Nichols, N. B. Optimum settings for automatic controllers. *Trans. ASME 64,* 11 (Nov. 1942), 759–768.

One of the first articles on controller tuning.

Economics

4.1. Eliot, T. Q., and Longmire, D. R. Dollar incentives for computer control. *Chem. Eng. 69,* 1 (Jan. 8, 1962), 99–104.

Indicates areas where computer control can be attractive.

4.2. Cotter, J. E. Justifying direct digital control. *Chem. Eng. Prog. 65,* 5 (May 1969), 52–55.

Devoted to the economic justification of DDC systems.

4.3. Hall, C. R. Computer control of processes—is it worth the cost? *Chem. Eng. Progr. 56,* 2 (Feb. 1960), 62–66.

Considers the cost versus return from computer control.

4.4. Jakubik, R. F., Kader, D., and Perillo, L. B. Justifying process control computers. *Automation 11,* 3 (March 1964), 81–84.

Analysis of factors pertinent to the economics of computer control systems.

4.5. Lane, J. W. Four examples where process computers pay off. *Instrumentation Technol. 15,* 7 (July 1968), 46–52.

Gives specific examples where control computers are justified.

4.6. Madigan, J. M. How managers see computer control. *ISA J. 10,* 1 (Jan. 1963), 49–50.

An analysis of the computer control venture from the managerial viewpoint.

4.7. Silva, R. Plant savings and the control hierarchy. *Instruments and Control Systems 41,* 6 (June 1968), 85–88.

Shows how various levels of computer control affect return from plant.

4.8. Snow, R. H. Conditions for successful computer control. *Chem. Eng. 72,* 12 (June 7, 1965), 181–185.

Gives certain conditions that must be met for attractive economic return.

4.9. Stout, T. M. Evaluating control system payout from process data. *Control Eng. 7,* 2 (Feb. 1960), 93–97.

Shows how an analysis of current process operating conditions can indicate possible returns.

4.10. ——. Computer control economics. *Control Eng. 13,* 9 (Sept. 1966), 87–90.

Devoted primarily to optimizing or supervisory systems.

4.11. ——. Estimating plant profits for process computer control. *Instrumentation Technol. 16,* 6 (June 1969), 56–61.

An article on economics with some equations to give expected returns.

4.12. Webb, M. S. Justification for control computers. *Chem. Eng. 61,* 10 (Oct. 1965), 83–86.

General discussions of ways to justify process control computers.

4.13. Wherry, T. C., and Parsons, J. R. Guide to profitable computer control. *Hydrocarbon Proc. 46,* 4 (April 1967), 179–182.

Discusses economic philosophies the authors feel will lead to economically successful control installations.

4.14. Williams, T. J. Economics and the future of process control. *Automatica 3,* 1 (Oct. 1965), 1–13.

Discusses the impact of economic incentives on the growth of computer control applications.

Hardware

5.1. Bailey, S. J. Faster computer control with read-only memories. *Control Eng. 14,* 8 (Aug. 1967), 65–68.

Discusses impact of read-only memories on process control computers.

5.2. Ball, J. Tying computers together. *Control Eng. 13,* 9 (Sept. 1966), 119–121.

Discusses some of the aspects of computer-to-computer communication in hierarchical computer systems.

5.3. Cerni, R. H. Transducers in digital process control. *Instruments and Control Systems 37,* 9 (Sept. 1964), 123–126.

Discusses role of sensors in computer control systems.

5.4. Charlton, K. W., and Spracklen, S. B. Control from chromatographs. *Control Eng. 10,* 3 (March 1963), 93–96.

Discusses the use of chromatographs in closed loop control systems.

5.5. Copeland, J. R., and Jackson, S. P. Minicomputers for the control system. *Control Eng. 16,* 8 (Aug. 1969), 90–94.

Discusses impact of minicomputers on process control systems.

5.6. Cundall, C. M. Backup methods for DDC. *Control Eng. 12,* 4 (April 1965), 93–103.

Presents techniques for switching from digital to analog control in case of computer failure.

5.7. Dinman, S. B., and Sonnenfeldt, R. W. Priority interrupt in control computers. *Control Eng. 8,* 5 (May 1961), 127–131.

Discusses the role of interrupts in digital computer control systems.

5.8. Eckman, D. P., Bublitz, A., and Holben, E. A satellite computer for process control. *ISA J. 9,* 11 (Nov. 1962), 57–64.

Discusses computers for on-line control of process units.

5.9. Feldman, H. L., et al. Minimizing process computer maintenance. *Instrumentation Technol. 15,* 1 (Jan. 1968), 60–66.

Suggests precautions to minimize process computer downtime.

5.10. Feldman, R. Selecting input hardware for DDC. *Control Eng. 14,* 8 (Aug. 1967), 75–79.

Reviews techniques and types of equipment available for the front end of time-shared computer systems.

5.11. Flynn, G. J. Special report: digital instrumentation. *Electronics 37,* 16 (May 18, 1964), 57–72.

A survey of digital instrumentation.

5.12. Gaines, N. W., et al. Union Carbide integrates multi-computer process control. *Instrumentation Technol. 14,* 3 (March 1967), 49–51.

A description of Carbide's Texas City computer control unit.

5.13. Green, P. J. Direct digital control of a nuclear reactor. *Instruments and Control Systems 38,* 6 (June 1965), 85–88.

An application of DDC.

5.14. Hall, C. R. Can computers control your process? *Chem. Eng. 67,* 6 (March 21, 1960), 153–156.

Advocates process improvements as a supplement to or in lieu of a computer control system.

5.15 Harris, R. A. The computer-actuator interface. *Control Eng. 13,* 9 (Sept. 1966), 110–113.

Considers various aspects of the transfer of signals from computer to process.

5.16. Harrison, T. J. Hardware: A matter of logic, memory, and timing. *Control Eng. 14,* 11 (Nov. 1967), 74–79.

Presents basic building blocks of a computer to promote its understanding by control engineers.

5.17 ——. How hardware responds to software. *Control Eng. 14,* 12 (Dec. 1967), 65–70.

Describes software-hardware interaction in a typical process control computer application.

5.18 Haynes, J. L. Time division multiplexing for industrial applications. *Instruments and Control Systems 37,* 4 (April 1964), 98–104.

Examines the utility of multiplexing in industrial control systems.

5.19. Hendrie, G. C., and Sonnenfeldt, R. W. Computer reliability—what's that? *ISA J. 10,* 1 (Jan. 1963), 51–56.

Examines the computer reliability question.

5.20. —— and ——. Evaluating control computers. *ISA J. 10,* 8 (Aug. 1963), 73–78.

Several computers are compared on the basis of key performance characteristics.

5.21. —— and ——. Reliability still means backup. *Control Eng. 13,* 9 (Sept. 1966), 131–135.

Suggests that current computer reliability is not adequate for removal of backup systems.

5.22. Hoffman, H. T., and Ragle, R. D. Multiprocessor computers for better on-line control. *Control Eng. 15,* 2 (Feb. 1968), 75–78.

Suggests that a multiprocessor arrangement could improve the response of process control computers.

5.23. Hubberth, W. F. Shrinking the man-computer interface. *Control Eng. 12,* 8 (Aug. 1965), 63–66.

Discussion of the man-machine communication problem.

5.24. Ida, E. S. Reducing electrical interference. *Control Eng. 9,* 2 (Feb. 1962), 107–111.

Suggests approaches to reducing input noise to digital computer control systems.

5.25. Jursik, J. Rejecting common mode noise in process data systems. *Control Eng. 10,* 8 (Aug. 1963), 61–66.

Outlines precautions for reducing noise on analog input systems.

5.26. Kaiser, V. A. New configurations in computer control. *Instrumentation Technol. 15,* 10 (Oct. 1968), 69–74.

Predicts effect of hardware improvements on digital control systems of the future.

5.27. Kern, J. L. The computer-operator interface. *Control Eng. 13,* 9 (Sept. 1966), 114–118.

Discussion of the man-machine communication problem.

5.28. Kompass, E. J., and Young, L. H. Expanding reliability to system effectiveness. *Control Eng. 5,* 4 (April 1958), 105–112.

Discusses relationships between component reliability and control system performances.

5.29. Lapidus, G. Minicomputers—what all the noise is about. *Control Eng. 15,* 9 (Sept. 1968), 73–80.

A survey on the available small computers.

5.30. ——. Digital proliferation—minicomputers revisited. *Control Eng. 15,* 11 (Nov. 1968), 72–78.

Describes some new entries into the minicomputer market.

5.31. Larsen, M. J., and Glahn, G. L. A user-designed operator console. *Instrumentation Technol. 15,* 4 (April 1968), 45–51.

Describes Humble Oil's operator's console.

5.32. McMath, P. The third generation—what does it offer? Proc. Third Annual Workshop on the Use of Digital Computers in Process Control, Baton Rouge, La., Feb. 21–23, 1968, Rimbach Pubs., Philadelphia, 1968, pp. 54–58.

Discusses impact of third generation computers on process control systems.

5.33. Maher, J. R. Cable design and practice for computer installations. *Tappi 48,* 5 (May 1965), 90A–93A.

Presents wiring practices intended to reduce noise on analog input systems.

5.34. Marks, C. H., and Skillern, J. A. Transmitters for computer information systems. *ISA J. 11,* 11 (Nov. 1964), 73–77.

Discusses various aspects of transmitters usable with computer systems.

5.35. Marson, G. B. Output and standby equipment for direct digital control. *Control 9,* 81 (March 1965), 137–140.

Devoted to hardware accompanying DDC systems.

5.36. Master, R. C. AC pickup on signal transmission lines. *Control Eng. 10,* 7 (July 1963), 123–124.

Discusses the problem of pickup of stray AC signals on analog input lines of control computers.

5.37. Mergler, H. W., and Hubbard, K. H. Digital control for single loop. *Control Eng. 12,* 2 (Feb. 1965), 61–64.

Discusses a single-loop digital controller.

5.38. Moore, C. F., Murrill, P. W., and Smith, C. L. Gauging performance costs of control loop hardware. *Instrumentation Technol. 15,* 11 (Nov. 1968), 69–72.

Discusses effect of control system hardware on loop performance.

5.39. Neblett, J. B., and Mears, F. C. Linking computers to analyzers in real-time process control. *ISA J. 9,* 1 (Jan. 1962), 44–47.

Discusses interface between process computer and analyzers such as chromatographs.

5.40. Pando, V. A., and Vance, J. M. Guide to computer specifications. *Control Eng. 13,* 9 (Sept. 1966), 99–104.

Discusses the specifications for a process control computer.

5.41. Rhodes, J. C. The computer-instrument interface. *Control Eng. 13,* 9 (Sept. 1966), 110–113.

Discusses the analog front end of a computer control system.

5.42. Ryan, F. M. Supervisory control systems. *Control Eng. 10,* 1 (Jan. 1963), 77–86.

Survey article of existing supervisory computer control systems.

5.43. Staff. Survey of small computers. *Instruments and Control Systems 42,* 8 (Aug. 1969), 69–83.

Gives characteristics, options, peripherals, and other information about minicomputers.

5.44. Stewart, L. E. How to handle computer input noise. *Power 107,* 10 (Oct. 1963), 77–79.

Suggests routes to minimizing the noise on analog signals from process sensors.

5.45. Stout, T. M., and Turner, E. B. A practical approach to multiple computer control. *Control Eng. 14,* 10 (Oct. 1967), 81–83.

Brings out the horizontal as well as hierarchical interactions in multiple computer control systems.

5.46. Theis, D. J., and Hobbs, L. C. Low cost remote terminals. *Datamation 14,* 6 (June 1968), 22–29.

Gives characteristics, features, and approximate cost of inexpensive CRT terminals.

5.47. Thurman, C. H. How to modify plant instruments for digital-computer control. *ISA J. 10,* 2 (Feb. 1963), 73–75.

Suggests instrument changes when switching from analog to digital control.

5.48. Ware, W. E. Direct digital control. *Instruments and Control Systems 38,* 6 (June 1965), 79–84.

Discusses DDC hardware.

5.49. Weisberg, D. E. Man-machine communication and process control. *Data Proc. Mag. 9,* 9 (Sept. 1967), 18–24.

Advocates CRT displays for man-machine communication.

5.50. ——. Graphic displays matching man to machine for on-line control. *Control Eng. 15,* 11 (Nov. 1968), 79–82.

Advocates CRT displays for man-machine interaction in control systems.

5.51. Whitman, K. A. Testing computer hardware. *Instrumentation Technol. 15,* 6 (June 1968), 59–61.

Discusses aspects of computer hardware testing at the factory.

5.52. Williams, M. E. Computer priority interrupt. *Instruments and Control Systems 36,* 8 (Aug. 1963), 91–95.

Discusses priority interrupt systems in process computer control systems.

Software

6.1. Aiken, W. S. Programming digital computers for on-line process control. *ISA J. 6,* 11 (Nov. 1959), 62–65.

Discusses aspects of programming real-time computers.

6.2. GE-PAC 4020 programming manual. General Electric Manual GET-3471, 1967.

A typical manual for a process control computer.

6.3. Fundamentals of process computers. General Electric Manual PCP-108, 1967.

Discusses basics of process control computers.

6.4. 1800 process supervisory program. IBM Manual H20-0261-0, 1966.

Description of Prospro.

6.5. Minutes, Workshop on Standardization of Industrial Computer Languages, Purdue U., Lafayette, Ind., Feb. 1969.

Results of a meeting working toward a standard process control language.

6.6. Bailey, S. J. Pushbutton programming for on-line control. *Control Eng. 15,* 8 (Aug. 1968), 76–79.

Describes Ferranti's CONSUL software system.

6.7. Baldridge, B. H. Parallel cascade processing, an approach to direct digital supervision. 1966 IEEE National Convention Record, New York, March 1966, Pt. 3, pp. 20–28.

Discusses a dual-computer control system.

6.8. Biles, W. R., and Sellars, H. L. A computer control system. *Chem. Eng. Progr. 65,* 8 (Aug. 1969), 33–35.

Describes Shell's MUSIC system.

6.9. Brooks, M. E. Process-oriented language compilers. Joint Automatic Control Conf. [Amer. Automatic Control Council], Minneapolis, Minn., June 19–21, 1963 (Am. Inst. of Chem. Engineers, New York, 1963), pp. 184–190.

An effort at software designed to readily permit specified functions.

6.10. Clough, J. E., and Westerberg, A. W. FORTRAN for on-line control. *Control Eng. 15,* 3 (March 1968), 77–81.

Shows how Fortran compilers can be designed for efficient use in on-line computing tasks.

6.11. Drewry, H. S. For successful computing control—start simple. *Control Eng. 12,* 8 (Aug. 1965), 76–81.

Suggests a building approach to successful computer control applications.

6.12. Frank, W. L., Gardner, W. H., and Stock, G. L. Programming on-line systems. *Datamation 9,* 5 (May 1963), 29–34.

Discusses aspects of programming real-time systems.

6.13. Gasper, T. G., Dobrohotoff, V. V., and Bangress, D. R. New process language uses English terms. *Control Eng. 15,* 10 (Oct. 1968), 118–121.

Explains an extension of Fortran to give an English-language compiler for on-line control systems.

6.14. Glass, T. J. Current trends in process computer software. 22nd Annual ISA Conf. and Exhibit, Chicago, Sept. 11–14, 1967, Preprint D2-3-DAHCOD-67.

Discusses new software developments.

6.15. Hansen, M. H., Manghan, B. Y., and Cornell, J. D. Scanning and processing plant variables. *Chem. Eng. Progr. 65,* 8 (Aug. 1969), 36–41.

Describes Shell's MUSIC system.

6.16. Hovorka, R. B., et al. The control system in action. *Chem. Eng. Progr. 65,* 8 (Aug. 1969), 46–52.

Describes Shell's MUSIC system.

6.17. Kipiniak, W., and Quint, P. Assembly vs. compiler languages. *Control Eng. 15,* 2 (Feb. 1968), 93–98.

Gives some of the tradeoffs in preparing control programs in assembly versus using a compiler.

6.18. Kuehrmann, O. V. A practical approach to direct digital control. Proc. Third Annual Workshop on the Use of Digital Computers in Process Control, Baton Rouge, La., Feb. 21–23, 1968, Rimbach Pubs., Philadelphia, 1968, pp. 85–88.

Discusses Leeds and Northrup's DDC package.

6.19. Lewtas, R., Taylor, J., and Neblett, J. B. Executive software. Proc. Third Annual Workshop on the Use of Digital Computers in Process Control, Baton Rouge, La., Feb. 21–23, 1968, Rimbach Pubs., Philadelphia, 1968, pp. 14–23.

Discusses Honeywell's Olert system.

6.20. McIrvine, E. C. Planning software for a manufacturing line. *Control Eng. 15,* 4 (April 1968), 100–103.

Discusses the software needs of a system which includes on-line process control computers as well as higher level administrative computers.

6.21. Markham, G. W. Fill-in-the-form programming. *Control Eng. 15,* 5 (May 1968), 87–91.

Suggests fill-in-the-form programming as one approach to the standardization of on-line process control software.

6.22. Nikiforuk, P. N., and Tinker, E. B. Computer applications in chemical process control. *British Chem. Eng. 9,* 1 (Jan. 1964), 17–23.

Discusses problems and approaches to successful applications of computer control.

6.23. Ochtman, E., and Sullivan, P. R. DDC software for a paper machine. *Control Eng. 15,* 11 (Nov. 1968), 90–95.

Describes some software problems specific to a computer control system for a paper machine.

6.24. Oerter, G. W. Relating real-time programs to systems hardware. *ISA J. 9,* 12 (Dec. 1962), 55–58.

Shows relationship between software and hardware in process computer control systems.

6.25. Ostfeld, D. M. A standard control language —how and when? *Instrumentation Technol. 16,* 9 (Sept. 1969), 32–34.

Discusses results of Purdue conference on standardization of computer control programming languages [6.5].

6.26. Patel, N. R. A DDC software package for a float glass process. *Control Eng. 15,* 7 (July 1968), 68–71.

Describes the software package for an application of DDC.

6.27. Pistole, D. G., et. al. Acceptable standards for process control software. 23rd Annual ISA Conf. and Exhibit, New York, Oct. 1968, Preprint 68-731.

Presents a view on a standard process control software package.

6.28. Puser, W. F. C. System timing for on-line computer control. Pt. I, *Instrumentation Technol. 15,* 12 (Dec. 1968), 41-46; Pt. II, *Instrumentation Technol. 16,* 1 (Jan. 1969), 51–56.

Discusses timing aspects of data transfer, program execution, etc.

6.29. Schoeffler, J. D. Process control software. *Datamation 12,* 2 (Feb. 1966), 33–42.

State-of-the-art article on software.

6.30. Sellars, H. L., et al. Program scheduling and data handling. *Chem. Eng. Progr. 65,* 8 (Aug. 1969), 42–45.

Describes Shell's MUSIC system.

6.31. Shannon, J. H. Executive control routines for process computers. *Control Eng. 10,* 4 (April 1963), 85–91.

Discusses components of a real-time executive.

6.32. Spooner, J. C. Real-time operating system for process control. 22nd Annual ISA Conf. and Exhibit, Chicago, Sept. 11–14, 1967, Preprint D1-1-DAHCOD-67.

Presents characteristics of a real-time executive.

6.33. Turner, G. On-line programs from control logic diagrams. *Control Eng. 15,* 9 (Sept. 1968), 102–104.

Describes the use of logic diagrams in developing computer control software.

6.34. Weaver, S. Importance of manufacturer software. *Control Eng. 15,* 1 (Jan. 1968), 60–64.

Points out the role of vendor-supplied software in a process computer control system.

6.35. Weiss, E. A. Needed: a process language standard. *Control Eng. 15,* 12 (Dec. 1968), 84–87.

Points out the need to standardize programming languages for process control computers.

6.36. ——. Conflict rages over process language standards. *Control Eng. 16,* 7 (July 1969), 77–81.

Describes some results of the Purdue conference on developing standard languages [6.5].

6.37. Whitman, K. A. Organizing the program for control. *Chem. Eng. 73* (Dec. 5, 1966), 135–138.

Discusses organization of process computer control software.

6.38. ——. Starting a pump under computer control. *Chem. Eng. 74* (Jan. 2, 1967), 81–92.

Discusses software required for this task.

6.39. Williams, T. J. Software: more standard programs. *Control Eng. 13,* 9 (Sept. 1966), 126–130.

Points out the need for more standard programming packages.

6.40. —— and Bailey, S. J. Software-critical factor in computer usage. *Control Eng. 14,* 10 (Oct. 1967), 64–72.

Describes the role of software in process computer control systems.

6.41. Woodley, G. V. Standard software blocks ease DDC system design. *Instrumentation Technol. 15,* 4 (April 1968), 57–62.

Advocates a modular software system.