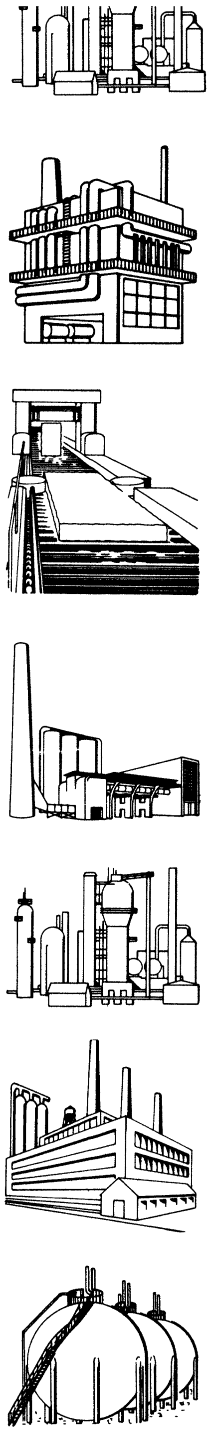
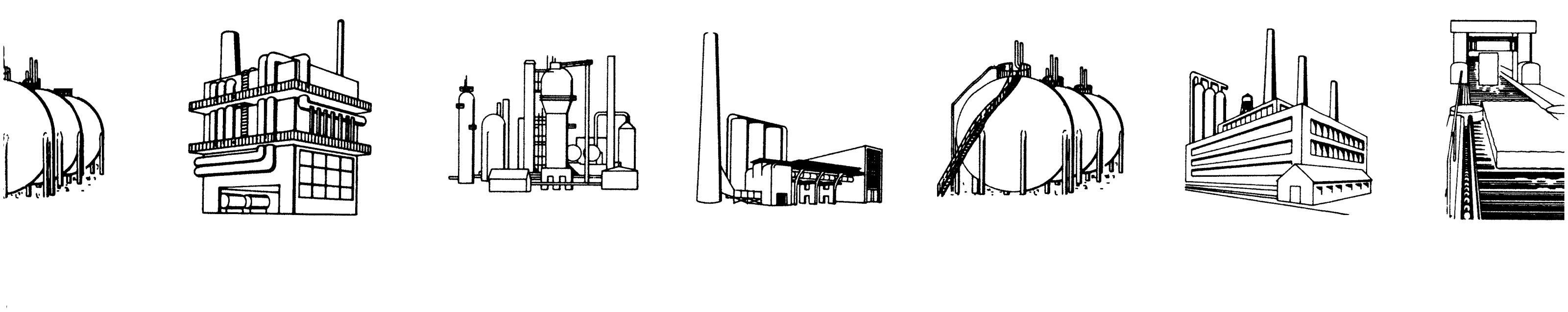


PROCESS COMPUTER CONCEPT COURSE



PROCESS COMPUTER SECTION
INDUSTRY CONTROL DEPARTMENT
PHOENIX, ARIZONA

GENERAL ELECTRIC

GENERAL ELECTRIC
PROCESS COMPUTER CONCEPT COURSE

PROCESS COMPUTER SECTION
INDUSTRY CONTROL DEPARTMENT
PHOENIX, ARIZONA
GENERAL ELECTRIC

CONTENTS

I INTRODUCTION TO PROCESS CONTROL COMPUTERS

A. THE NEED FOR PROCESS CONTROL

1. Management is confronted with —

- Decreasing Profit Margins.
- Increasing Operating Costs.
- Increasing Complexity of Processes.

2. Automation can provide —

- Reduced Process Losses.
- Better Process Control.
- Smoother Plant Operation.
- Better Utilization of Personnel.
- Accurate Inventory Control and Scheduling.
- Automatic Alarming of Malfunctions.
- Improved Plant Safety.
- Increased Process Knowledge.
- Automatic Data Logging and Analysis.

B. THE PLACE OF PROCESS COMPUTERS

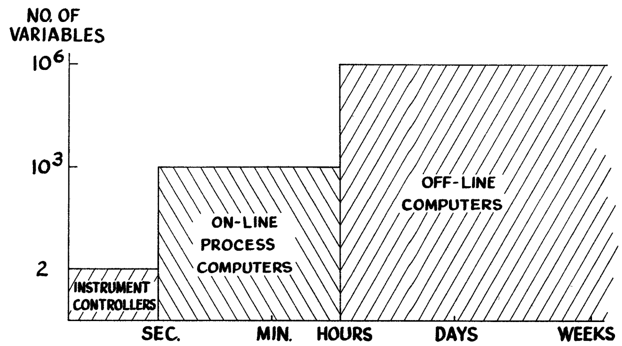
1. Digital Computer Control Vs. Analog Computer Control on Large Systems

FUNCTION	DIGITAL	ANALOG
Speed	Sampled Data	Continuous
Accuracy	No Loss	Tends To Drift
Reliability	Better Than 99%	Less Than 99%
Flexibility	Program Change	Hardware Change
Complexity	Standard Design	Special Design

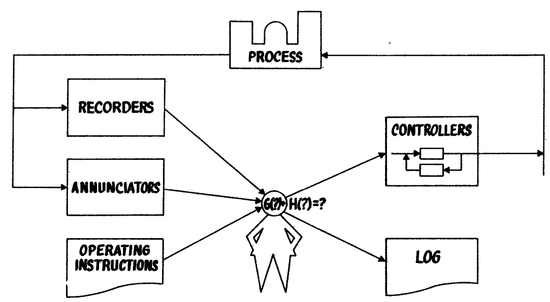
2. Area of Application of Process Computers

a. 10 - 1000 variables

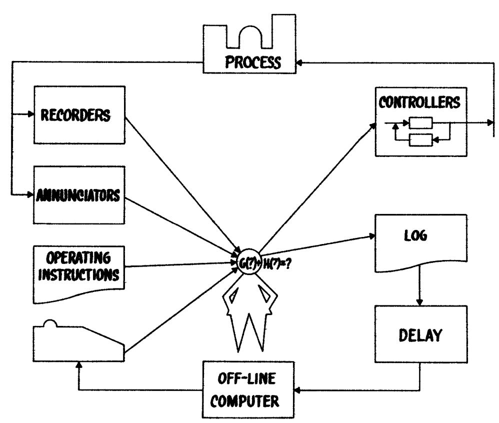
b. Time Scale of Seconds - Hours.



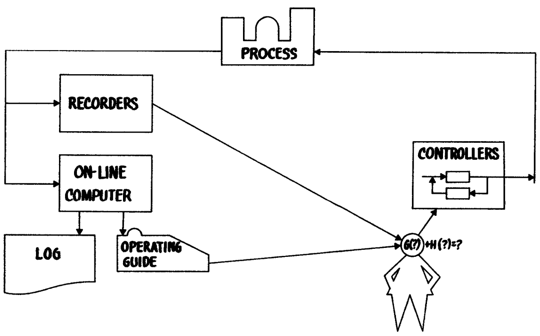
Conventional Process Operation



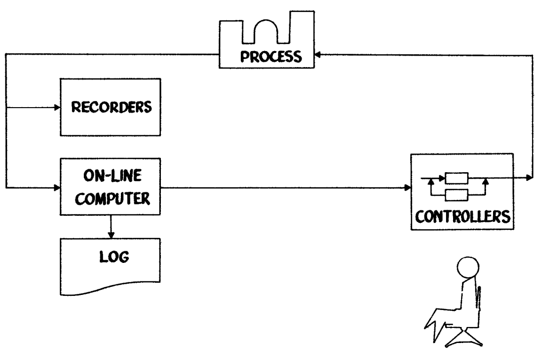
4. Process Operation Assisted By Off-Line Computer



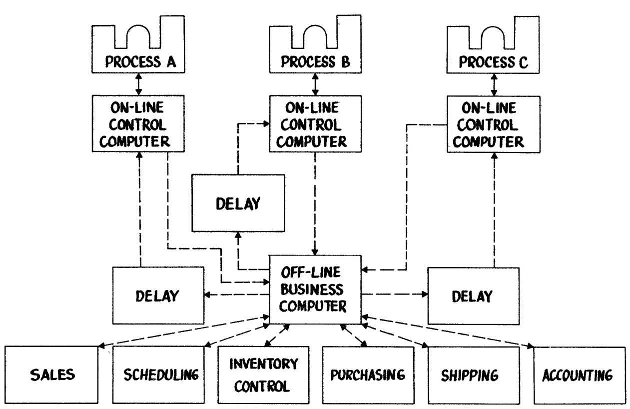
5. Process Operation Assisted By On-Line Computer



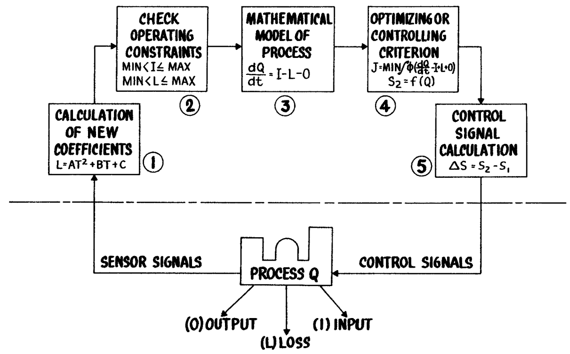
6. Process Operation Controlled By Computer



7. Integrated Process Business Automation



C. OPERATIONS PROVIDED BY PROCESS CONTROL COMPUTERS



(1) Scanning, calculating and storing new coefficients for the controlling equations.

(2) The physical and practical limitations of variables and equipment effecting the process are represented mathematically.

(3) A group of equations that describe the state of the process at any time.

(4) A mathematical equation representing the selected optimizing criterion or selection of the best set of operating variables for control.

(5) Control signals based upon present plant operating data, mathematical model, operating constraints, and controlling criterion.

D. FUTURE OF PROCESS CONTROL COMPUTERS

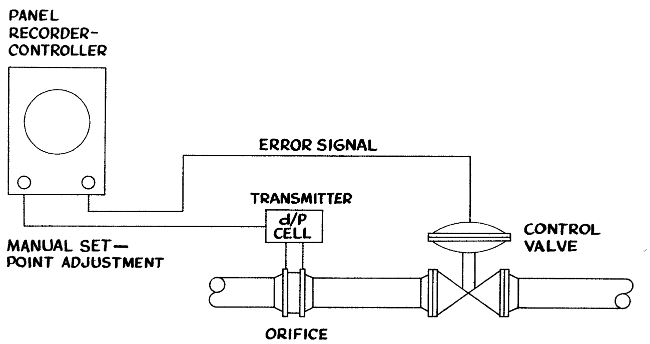
1. Direct Digital Control Functions

- Eliminate Analog Sub-loops.
- Automatic Calculation of Set Points.
- Checking and Alarming On Out-of-limits.
- Digital Display of Selected Values.
- Very Accurate Control of Process
- Computer Calibration of Sensors.
- Automatic Logging of Plant Variables.

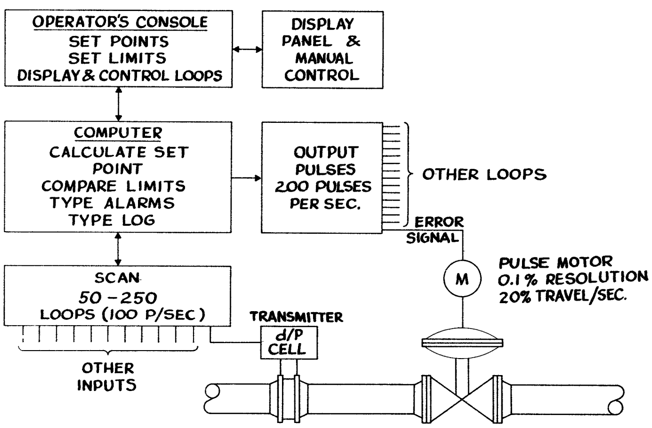
SAMPLING FREQUENCIES FOR DIRECT DIGITAL CONTROL

Variable	Speed
Flow	Every Second
Level & Pressure	Every 5 Sec.
Temperature	Every 20 Sec.
Composition	Every 20 Sec.
Operator Demands	Max. Every Sec.
Digital Inputs	Every Sec.

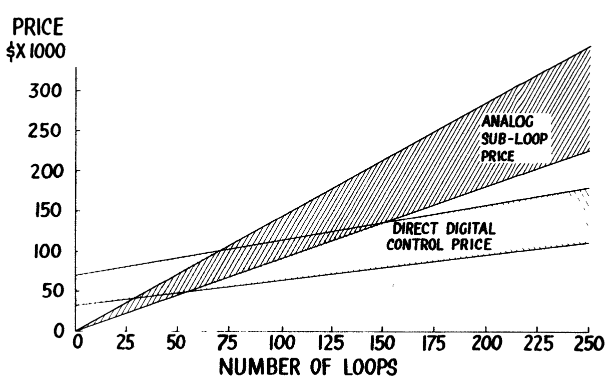
ANALOG CONTROL LOOP



DIRECT DIGITAL CONTROL



ECONOMIC EVALUATION OF DIRECT DIGITAL CONTROL



2. Digitized Input Signals

   a. High Accuracy Sensors

      - Digitized Flow Signals
      - Digitized Pressure Signals

   b. Digital Telemetering

      - Billing KWHR Meters
      - Pulse Accumulators

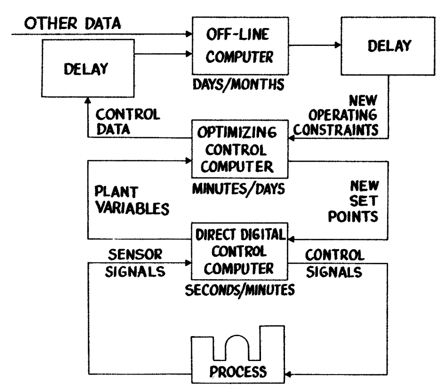
   c. Benefits

      - No Loss of Accuracy in Communication Channel.
      - No Loss of Accuracy in Analog-to-Digital Conversion.
      - High Speed Input into Computer.

3. Integrated Computer Systems

   - Small Direct Digital Control Computer
     Time Scale - Seconds to Minutes
   - Optimizing Control Computer.
     Time Scale - Minutes to Days
   - Off-Line Computer
     Time Scale - Days to Months.

INTEGRATED COMPUTER SYSTEMS



II APPLICATION AND JUSTIFICATION FOR CONTROL COMPUTERS

A. INDUSTRIAL APPLICATIONS

1. Cement Plants

- Kiln Control

  Control disturbances, such as changing kiln speed, feeder speed, draft, fuel, etc., affect kiln operation. Process disturbances, such as ambient temperature, wind velocity, rain, snow, or crystalline structure of raw material, affect the burnability and kiln operation.

- Raw Material Blending

  Manual blending is difficult because each of the raw materials contain some quantity of all the oxides normally found in cement.

- Computer Functions

  1. Automatic proportioning of raw material
  2. Automatic kiln start up
  3. Kiln control
  4. Ball mill load control
  5. Off-limit alarming
  6. Production logging

2. Direct and Hidden Benefits

   a. Kiln control

   - Reduced lining maintenance
   - Reduced fuel costs
   - Reduced power consumption in grinding
   - Smoother control and better uniformity of cement

   b. Raw material blending

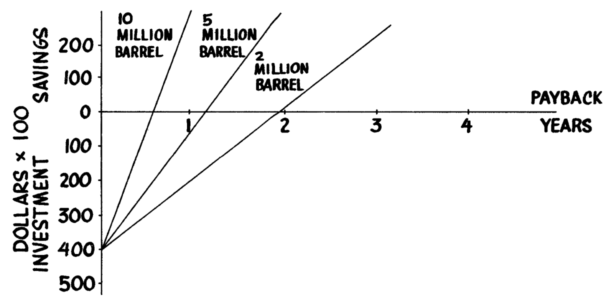
   - Extended quarry reserves
   - Minimize cost of additives
   - Increased uniformity of product
   - More "on target" mix

3. Typical Payouts and Justification for Medium (2 million barrel per year) Plant

Function	Estimated Annual Savings
Extend Quarry Reserves	$ 50,000
Minimize Cost of Additives	15,000
Increased Uniformity of Product	15,000
More "On Target" Mix	35,000
Reduced Power in Grinding	20,000
Reduced Lining Maintenance	30,000
Reduced Fuel Cost	25,000
Production Logging & Alarming	15,000
Total	$205,000

Estimated payback between 2-4 years. Initial cost depends on degree of existing centralized control.

CEMENT PLANT
CASH FLOW CHART FOR VARIOUS SIZE PLANTS



When automation is justified, it is being paid for by either a capital investment or by reduced efficiency.

4. Chemical and Petrochemical Applications

- Alcohol Plants
- Ammonia Plants
- Chloride Plants
- Chemical Plants
- Ethylene Plants
- Natural Gas Purification
- Paper-making Process
- Primary Oxidation Units
- Styrene Butadiene Rubber
- Vinyl Chloride

5. Direct and Hidden Benefits
   - Increased "Through Put"
   - Reduced Operating Cost
   - Improved Quality Control
   - Better Control
   - Accurate Operating Data
   - Reduced Maintenance
   - Better Technical Information About Process

6. Typical Payouts and Justification
   - Market-limited Conditions — tangible benefits have shown a payback time of 2-4 years
   - Production-limited Conditions — tangible benefits have shown a payback time of 1-2 years

Evaluation of hidden benefits along with the tangible benefits has shown payback times approximately one half those indicated by the tangible benefits alone.

7. Steel - Basic Oxygen Furnace

   - Charge Calculation

     Instantaneous process disturbances in metallurgy and temperature of hot metal (cast iron) and scrap make precalculation of each heat impossible. Unwieldy heat material balance equations require computer.

   - Computer Functions
     1. Charge/Reblow Calculations
     2. Set point Control of Raw Materials (metal, scrap, lime, ore, spar, oxygen)
     3. On-Line Raw Data Collection/Conversion (temperatures, analysis, material weights, and flows)
     4. Scan, Log, Alarm
     5. Production Logging

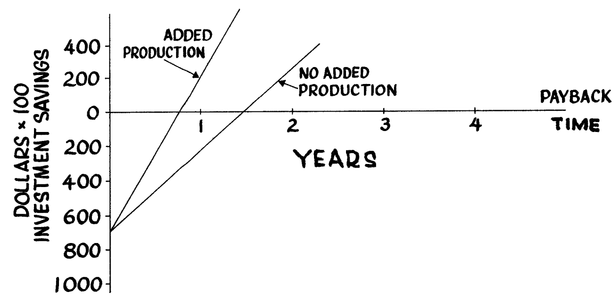
   - Economic Justification

     Assumptions

     - Two 250-ton Vessels - one in operation full time
     - Normal cycle time - 60 minutes
     - Average reblow time - 4 minutes
     - 50% of heats "on target" manually
     - 75% of heats "on target" with computer control
     - Hot metal market - $80/ton
     - BOF process costs - $50/ton

Function	Estimated Annual Savings
Reduced scrap loss (.5%) due to overweight heats	$106,000
Reduce iron loss in slag (5%)	170,000
Reduce flux requirements (5%)	133,000
Reduce oxygen overblow	10,000
Reduced labor	45,000
TOTAL	$464,000

BASIC OXYGEN FURNACE
ESTIMATED PAYBACK



The time and energy spent improvising for a tool that is needed will pay for the tool itself.

8. Steel - Hot Strip Mills

   - Computer Functions
     1. Horizontal Setup - Screws are set to produce finished gage at head end of bar within limitation of stand proportional loading; stand speeds are set in finish mill to maintain constant mass flow.
     2. Vertical Setup - Edger screws are set to produce target width in rough mill; edger speeds are set to compensate for draft.
     3. Mill Pacing - Slabs called for, holding tables speed controlled to fix gaptime at finish mill entry.
     4. Strip Temperature Control - Mill setup function modified to set strip head end on finished temperature; run-out table sprays controlled to fix coiling temperature.
     5. Production Logging - Quality and productivity logs per coil and on summary basis.

- Direct Economic Benefits
  1. Increased Productivity - Proper pacing utilizes mill up to electromechanical limitations, reducing work turns or increasing productivity.
  2. Reduced Cobble Loss - The large percentage of total cobbles due to operator setup errors are virtually eliminated.
  3. Reduced Reject Loss - Improved mill setup reduces off gage/width coils sold as seconds or scrapped.
  4. Reduced Slitter Scrap - Closer width tolerance scheduling allows reduction in overwidth strip.
  5. Roll Maintenance - Less cobbles, closer load distribution improves roll and spindle life.
- Indirect Economic Benefits
  1. Scheduling Flexibility - Large gage changes outside normal operator limits are allowed.
  2. Metallurgical Properties - Improved due to closer finishing/coiling temperature control.
  3. Reduced Mill Damage - From cobble reduction.
  4. Timely Production Reports - Allow prompt reschedule to fill high-priority orders; provide permanent quality (gage, width, temperature log) for each delivered coil.
- Typical Payout and Justification for:
  - 1.2 million ton/year mill
  - Carbon/silicon product only
  - Typical Product mix
  - Production limited by orders available

Function	Estimated Annual Savings
Reduce work turns	$108,000
Reduced cobbles/cold bar removal	210,000
Roll maintenance	54,000
Reduced rejects as seconds	240,000
Reduced scrap rejects	120,000
Reduced slitter scrap	240,000
TOTAL	$972,000

Estimated payback - 2 to 4 years as function of existing control based upon direct economic benefits.

9. Steel - Additional Automation Areas
   - Cold Rolling Mills
   - Tinning Lines
   - Annealing Lines
   - Power Demand Calculations and Fuel Selection
10. Petroleum, Applied Research and Miscellaneous Applications
    - Alkylation Unit
    - Catalytic Polymerization Unit
    - Crude Still
    - Fluid Catalytic Cracker
    - Lubricants and Waxes
    - Oil Refinery
    - Thermofor Catalytic Cracker
    - Pilot Plants
    - Weather Forecasting
    - Jet Fuel Storage Tank Monitor
    - Air Traffic
    - Surface Traffic
    - University Process Studies
    - Medical Research - Heart Hospital
    - Missile Checkout Systems
    - TV Network Switching
    - Toll Registration
    - Engine Testing

B. UTILITY APPLICATIONS

The average relative annual costs of electric utility operations are given below:

Fixed Costs	(Capitalization)	50%
Fuel		40%
Wages		6%
Maintenance		4%

These figures show that the large savings in automation will not only come from a reduction in operating personnel, but also from optimization to reduce fuel costs and capitalization expenditures.

1. Steam Electric Power Plants

- Data Logging
- Performance Calculations
- Monitoring and Performance
- Sequence Monitoring
- Turbine-Generator Control
- Boiler Optimization
- Automatic Start-up and Shut-down

2. Direct and Hidden Benefits

- Operator Labor Savings (1-2 operator/shift)
- Clerical and Engineering Labor Savings
- Fuel Savings (1/2 to 1%)
- Reduced Possibility of Major Damage (1/3 to 2/3 fewer major outages)
- Reduced Outage Time for Inspection (1 to 2 days per year)
- Improved Centralized Control
- Reduced Number of Recorders and Panel Space
- Improved Plant Safety
- Better Plant and Equipment Logs

3. Typical Payouts and Justification - Assumed Typical Utility Constants:

- 15% Annual Carrying Charge
- 30¢ per Million BTU Fuel Cost
- Unit Boiler, Turbine, Generator Scheme
- Total Cost of One Operator per Year $10,000
- Plant Maintenance of $1.60 per Kilowatt per Year. A 1% Maintenance Savings of $16/MW/Yr.
- Plant Load Factor of 65%

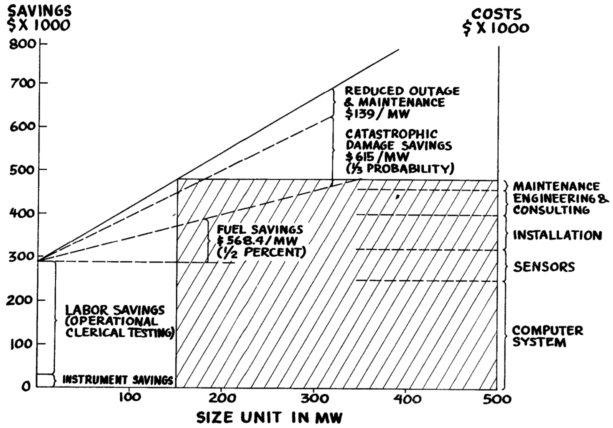
The following tabulations in Table I represents the assumed savings available for System I and II:

Savings	Instrument & Control Panel	Labor (Operating Clerical Testing)	Fuel	Major Damage and Outage	Reduced Outage & Maintenance
Function of Size	Constant	Constant	Varies Directly	Varies Approximately Directly	Varies Directly
System I "GARDE"	Reduced Trend Recorders and Multipoint Recorders	Equivalent to One Operator Per Shift. 4 - Total	Approx. 1/2% or 50 BTU/KWHR	1/3 of $1846/MW Max. Available or $615/MW	1% Maintenance and 1 day/yr $139/MW
System II "Automatic Plant"	Reduced Recorders and Control Panel Size	Equivalent to Two Operators Per Shift. 8 - Total	Approx. 3/4% or 75 BTU/KWHR	2/3 of $1846/MW Max. Available or $1230/MW	3% Maintenance and 1-1/2 day/yr $378/MW

Table I

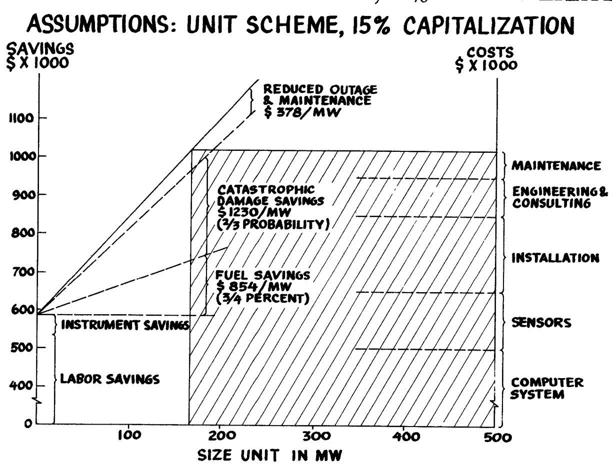
SYSTEM I GARDE COMPUTER SYSTEM EVALUATION

ASSUMPTIONS: UNIT SCHEME, 15% CAPITALIZATION



SYSTEM II AUTOMATED PLANT SYSTEM EVALUATION

ASSUMPTIONS: UNIT SCHEME, 15% CAPITALIZATION



4. Electrical Power Dispatching

   a. Operations Control

      - Load and Frequency Control
      - Economic Dispatching
      - Hydro-Thermal Optimization
      - Kilovar Supply Switching
      - Automatic Data Logging

   b. Operations Accounting

      - Interchange Billing
      - System and Unit Production Statistics

   c. Operations Planning

      - Interchange Transactions
      - Unit Commitment
      - Selection of Fuels
      - Maintenance Scheduling
      - Spinning Reserve Calculations

5. Direct and Hidden Benefits

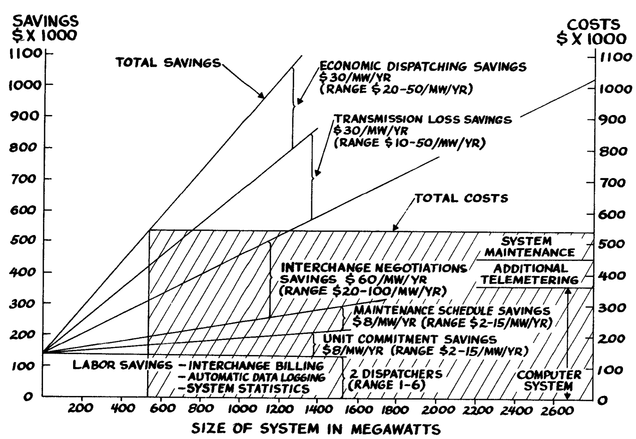
System studies have shown the following range of savings.

Function	Annual Savings $/MW/Yr.
Economic Dispatching	10-50
Transmission Loss Consideration	10-50
Interchange Negotiations	20-100
Maintenance Scheduling	2-15
Unit Commitment Scheduling	2-15
Hydro-Thermal Coordination	50-150*
Labor Savings	
Interchange Billing	
Automatic Data Logging	
System Statistics	(1-6 men)

*Based upon peak hydro capacity

6. Typical Payouts and Justification Based Upon Average Savings.

ECONOMIC EVALUATION OF DIGITAL DISPATCHING BASED UPON AVERAGE RANGE OF SAVINGS AND 15 PER CENT CAPITALIZATION (SAVINGS FOR HYDROELECTRIC OPTIMIZATION NOT INCLUDED)



7. Nuclear Power Plants

   - Rod Position Monitoring
   - In Core Ion Chamber Monitoring
   - Automatic Logging and Alarming
   - Plant Performance Calculations

- Core Performance Calculations
- Control Rod Pattern Calculations
- Fuel Burnup Optimization
- In-core Chamber Calibration
- All Normal Power Plant Computer Functions

8. Direct and Hidden Benefits

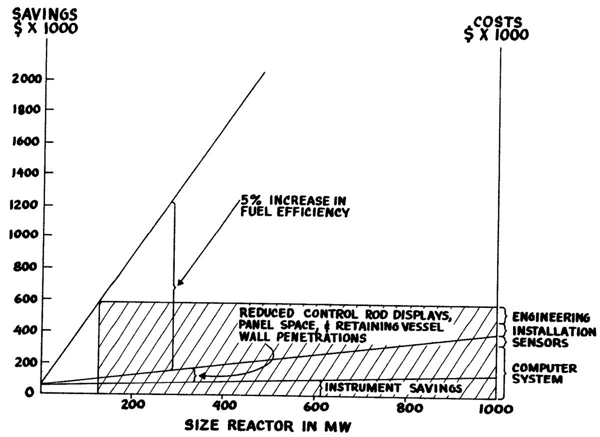
- Reduced Retaining Vessel Wall Penetrations and Wiring
- Reduced Display Equipment and Panel Layout
- Automatic Logging and Alarming
- Fuel Burnup Optimization
- Reduced Refueling Outages
- Reduced Possibility of False Scrams
- All Benefits Associated with Power Plant Computer Applications

9. Typical Payouts and Justifications - Assumed Typical Utility Constants:

- 15% Annual Carrying Charge
- 18¢ per Million BTU Fuel Cost
- 11,000 BTU/KWHR Heat Rate
- Plant Load Factor of 65%
- Possible Gain in Average Exposure 5 to 8 Percent

BOILING WATER REACTOR
15% CAPITALIZATION

MONITORING & CORE CALCULATIONS



III THE DIGITAL COMPUTER

A. COMPUTER LANGUAGE - THE BINARY NUMBER SYSTEM

Electronic Digital Computers most commonly employ the binary number system as an internal language because it is most compatible with electronic circuitry (bi-stable). Electronic elements can represent binary information directly but have to use special coding systems to represent decimal and alphabetic information.

1. The Basis of all Number System - The Pattern of Numbers

All number systems have a basic pattern of representation which can be stated mathematically by the following polonomial.

$$\text{Number} = (a_n r^n + a_{n-1} r^{n-1} + \ldots + a_0 r^0 + \ldots + a_{-n} r^{-n})$$

Where a_n is a coefficient and equal to any admissible symbol, n is an integer referring to the position of the nth term with respect to the decimal point, and r is the radix or base of the number system.

2. The pattern of numbers applied to the decimal number system, where a = 0 through 9, and r = 10

$$(937.5)_{10} = (9\text{x}10^2)+(3\text{x}10^1)+(7\text{x}10^0)+(5\text{x}10^{-1})$$
$$= 900 + 30 + 7 + .5$$

3. The pattern of numbers applied to the binary number system, where a = 0 or 1, and r = 2

$$(10110.11)_2 = (1\text{x}2^4)+(0\text{x}2^3)+(1\text{x}2^2)+(1\text{x}2^1)+(0\text{x}2^0)+(1\text{x}2^{-1})+(1\text{x}2^{-2})$$
$$= 16 + 0 + 4 + 2 + 0 + .5 + .25$$
$$= (22.75)_{10}$$

4. The pattern of numbers applied to the Octal Number system, where a = 0 through 7, and r = 8

$$(1651.4)_8 = (1\text{x}8^3)+(6\text{x}8^2)+(5\text{x}8^1)+(1\text{x}8^0)+(4\text{x}8^{-1})$$
$$= 512 + 384 + 40 + 1 + .5$$
$$= (937.5)_{10}$$

The Octal Number system has special significance when used in association with the binary number systems because it is an ideal abbreviation of binary numbers. This is possible because

$$2^n = 8^{n/3}$$

Therefore the octal numbers may be translated directly into binary numbers with a group of three binary digits equal to each octal digit. For example:

$$(1651.4)_8 = (001\ 110\ 101\ 001\ .\ 100)_2$$

Notice the grouping of binary characters.

5. Coded Numbers and Letters - The basic form of representation in a computer system is binary and it is necessary to use special binary coding systems to represent decimal numbers or alphabetic characters.

a. Binary-Coded-Decimal (BCD) - BCD employs 4 binary digits (bits) to represent each decimal character as shown in the following table.

Decimal	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

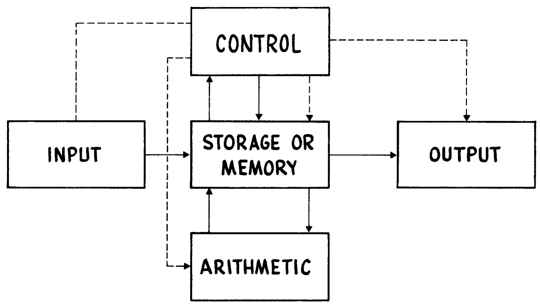
This coded system is less conservative of bit combinations than true binary, but directly compatible with decimal numbers. An example of BCD is as follows:

$$(937.5)_{10} = (\underset{9}{1001}\ \ \underset{3}{0011}\ \ \underset{7}{0111}\ \ \underset{.}{.}\ \ \underset{5}{0101})_{\text{BCD}}$$

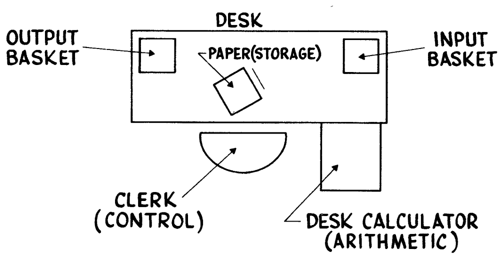
b. Binary-Coded-Alphanumeric - Using 6 binary bits in coded combinations all alphabetic, numeric, and special symbols may be represented in a binary coded form. This binary-coded system is normally used in business type computers.

B. BASIC DIGITAL COMPUTER ORGANIZATION

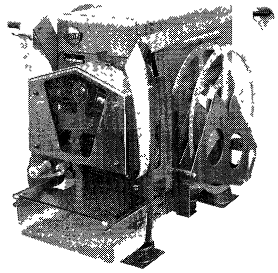
1. Five Basic Components



- Input - introduces information in electronic digital form.
- Output - translates electronic information into external form.
- Memory - Stores data and instructions electronically.
- Arithmetic - Performs basic arithmetic operations.
- Control - Interprets instructions to control the other 9 components.

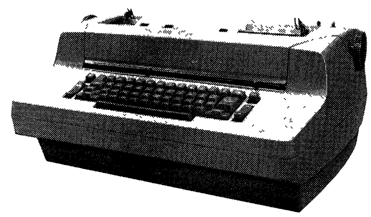


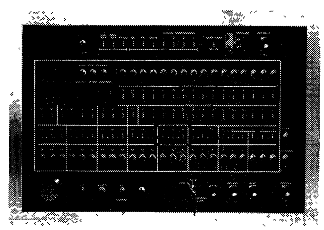
Analogy of the Five Basic Components



Paper Tape Reader & Punch

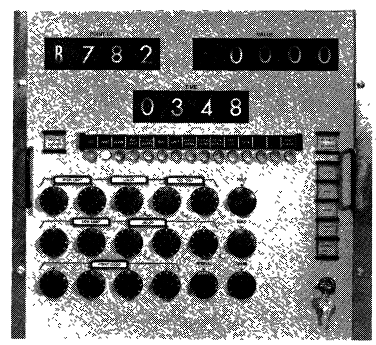
Electric Typewriter



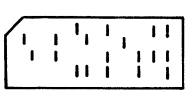


Programming and Maintenance Console

Operators Console

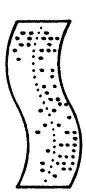


a. & b. Input/Output Communication with the Computer



Punched Card developed by Dr. Herman Hollerith in 1889 for use of U. S. Census Bureau.

Translates holes in cards into electronic pulses. Eighty columns provide 80 characters.



Punched Paper Tape

Translates holes in tape into electronic pulses.

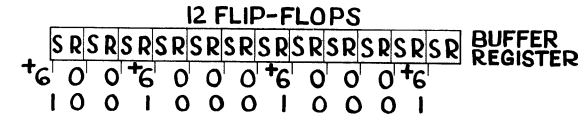
Each group of holes across the tape represents a character.

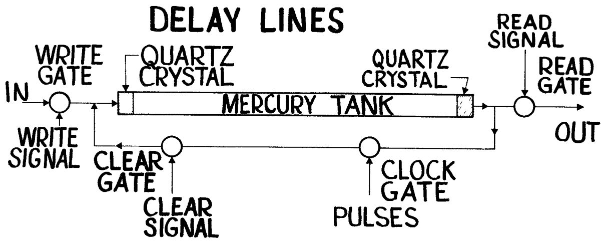
c. Memory or Storage

A basic function that must be performed in digital computers is the retention of information until it is used; this is referred to as "memory" or "storage".

The first memory devices were single word (a group of binary bits) vacuum tube or transistor "registers". These were called buffers for temporary storage.

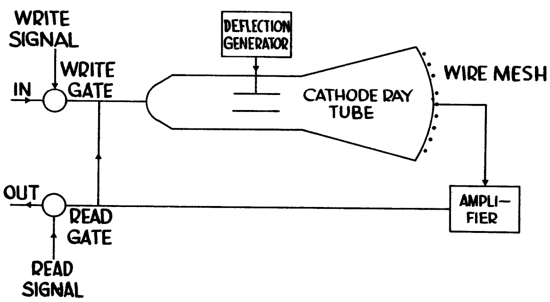
The most commonly used circuit for 1 bit storage device is the "flip-flop", sometimes referred to as a "bistable multivibrator" or "toggle". A vacuum tube toggle circuit was developed by Eccles and Jordan in 1919 and termed a trigger relay or flip-flop.





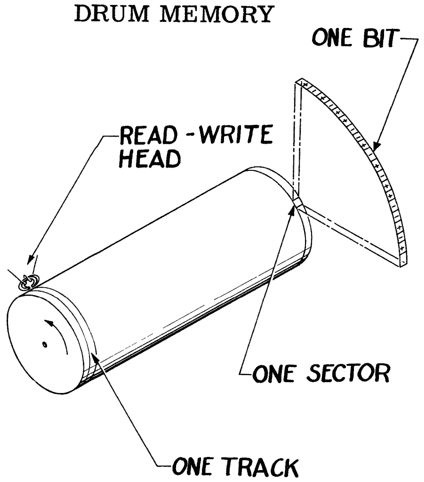
1949 - EDSAC (Electronic Delay Storage Automatic Computer)
1951 - UNIVAC 1
(both used delay storage tanks)

ELECTROSTATIC STORAGE TUBES

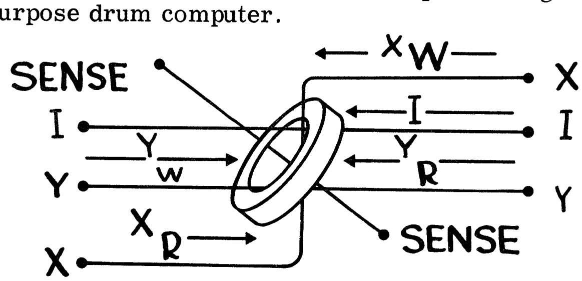


IBM 701 & 702 in 1953 and UNIVAC SCIENTIFIC (ERA 1103) in 1952 used electrostatic memories.

DRUM MEMORY



IBM 650 in 1954 was the first mass produced general purpose drum computer.

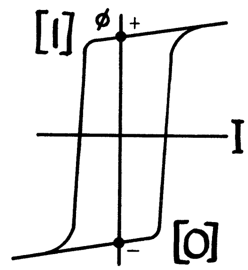


Memory Core with Associated Windings

The Whirlwind I developed at the Mass. Institute of Technology in 1954 was the first core memory computer.

(1) Magnetic Core Memory - uses many electromagnetic cores, .050 inches in diameter, each retaining one bit of information. Storage is accomplished by inducing the proper magnetic polarity in each core.

(a) The magnetic cores have a square hysterysis loop that insures bi-stable polarity. Applying one-half of the necessary current required to change the state of the core has no effect on the core.

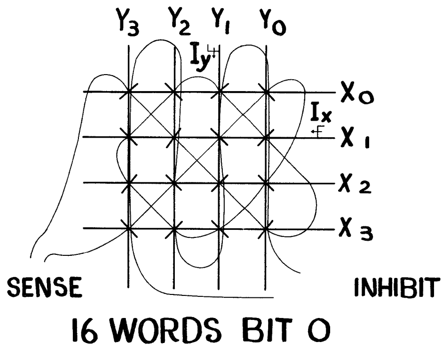


(b) Reading is accomplished by driving the polarity of the selected core to a given reference polarity and then sensing (with the sense wire) for a change or no change in the state of the core. In this fashion bi-stable information may be retained in a core, but the information is destroyed by the read-out operation and must subsequently be restored to the proper polarity.

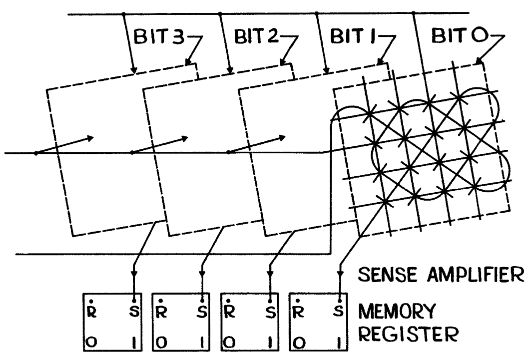
(c) Writing is always accomplished directly after read-out and is done by attempting to drive the polarity of the core to the opposite state and discriminately allowing or inhibiting this operation to take place. If an output is sensed during reading, then the writing phase causes the state of the core to be driven back to its original condition. If no output is sensed, the writing operation is inhibited and the core remains in its original state. The inhibiting is accomplished by allowing minus one-half of the current required to be placed on the inhibit wire, which counteracts the effect of the x wire current. This results in only one-half of the required current and thus inhibits the change in state of the core.

(d) Magnetic Core Plane - is the basic unit of a core memory system. Each x and y wire is common to all cores in the respective row (x) or column (y). The sense and inhibit wires are common to all cores in the plane. As a given x and y wire is activated, the resulting sense for change in the core at the intersection is done by the sense wire. Control of writing operation is accomplished with the inhibit wire.

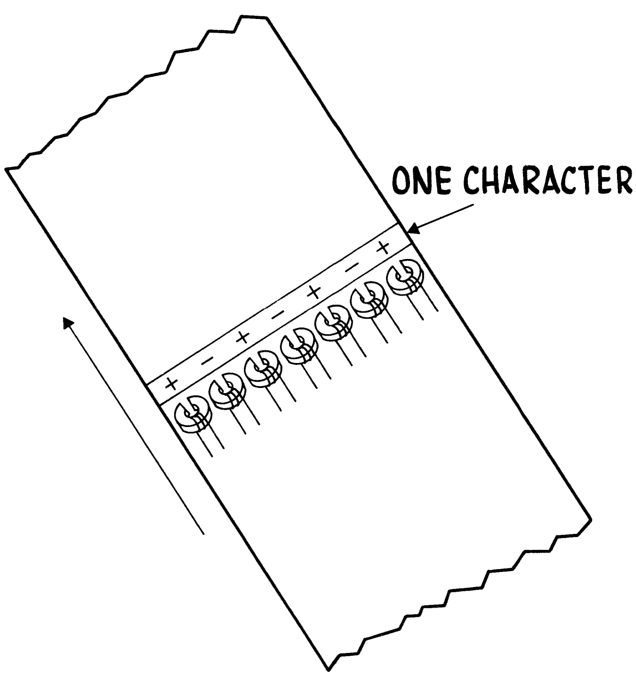
CORE PLANE



ONE WORD 4 BITS/WORD



MAGNETIC TAPE



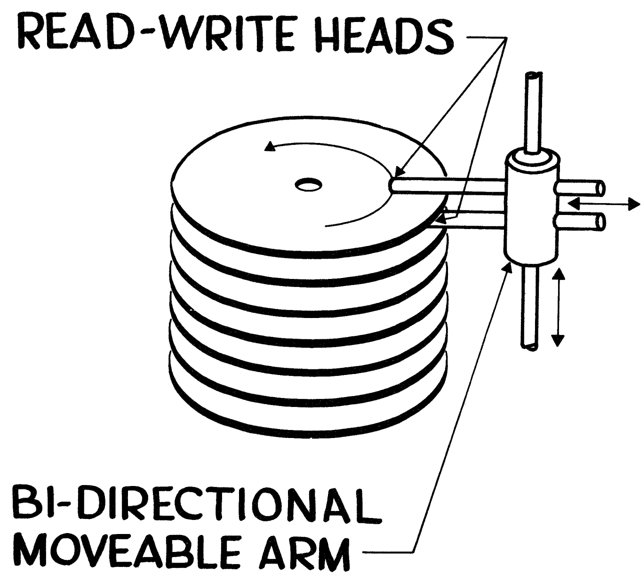
Large sequential file storage for high activity of records. An activity of 100% means that all records are updated at the same time. Speeds of 15,000 to 96,000 characters/second, packing density of 200 to 550 characters/inch, tape lengths of 1200, 2400, 3600 ft. giving 2 to 14 million characters/reel. Information is stored sequentially by record number and is not addressable.

DISK FILES

Large random access file storage for low activity of record or information.

(2) Magnetic Disk - are rotating disks stacked one on top of the other coated with a magnetizable substance. Polarized spots retain the information and the reading and writing is accomplished the same as discussed for magnetic drums.

(a) Read-write heads are mounted on movable arms which move the heads from place to place on the disk and also to the other disks in the system. Multiple arms may be employed to minimize access time.

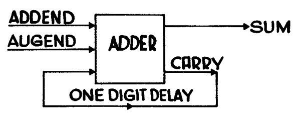


(b) Addressing is accomplished by numbering the disks, tracks and sectors.

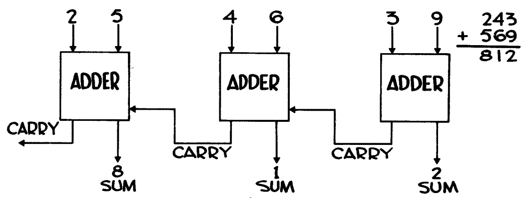
Speeds of 10,000 to 62,500 characters/sec. Head positioning times of 1500 to 150 milliseconds. Access or latency times of 500 to 50 milliseconds. Storage of 1 to 6 million words.

d. Arithmetic Unit

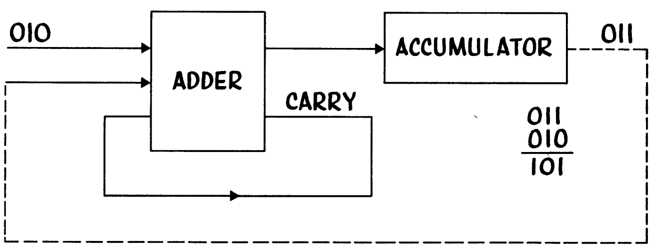
The heart of the arithmetic unit is the adder. A serial adder has one adder stage, circuits for handling one pair of digits at a time.



A parallel adder has one stage for every digit in the word.



In a pure binary machine the adder is very simple.



Binary Arithmetic - is the basic form of arithmetic in any electronic digital computer and it employs logical circuits to perform these functions.

(1) Addition

(a) Rules:

0	0	1	1
+0	+1	+0	+1
0	1	1	10

(b) Examples:

	1 1 carries		111111 carries
$(13)_{10}$	$+(01101)_2$	$(27)_{10}$	$(0011011)_2$
$+(09)_{10}$	$(01001)_2$	$+(39)_{10}$	$+(0100111)_2$
$(22)_{10}$	$(10110)_2$	$(66)_{10}$	$(1000010)_2$

(2) Subtraction

(a) Rules:

0	0	1	1
-0	-1	-0	-1
0	1*	1	0

(*Borrow one from next column)

(b) Examples:

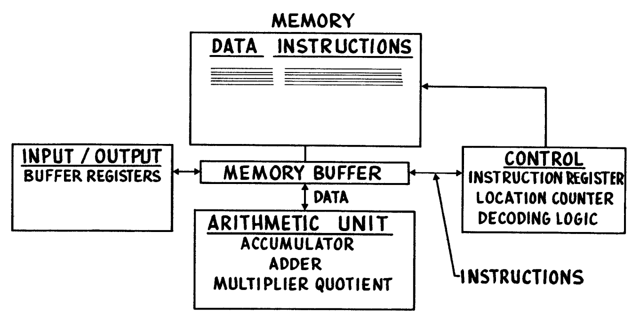
45	(101101)	44	(101100)
-20	-(010100)	-34	-(100010)
25	(011001)	10	(001010)

(3) Multiplication and Division - As in many digital calculating devices these functions are accomplished by shifting and adding or subtracting.

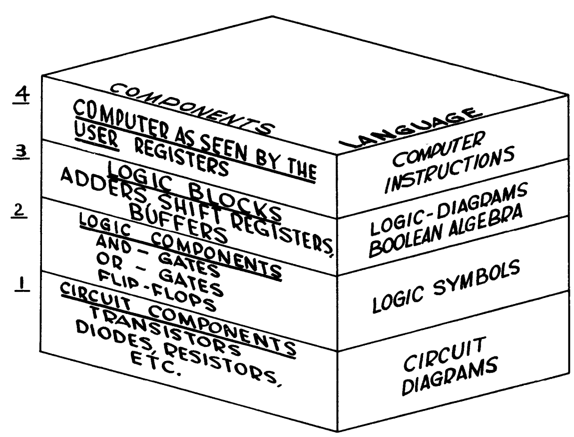
e. Control

Types of Control	Speed	Flexibility
External	No	Yes
Wired	Yes	No
Internal	Yes	Yes

Stored - Program concept (internal control) was suggested by Dr. J. Van Nuemann in 1945. The EDSAC, Electronic Delay Storage Automatic Computer, was built at the Cambridge University in 1949 as the result of Dr. Van Nuemann's work.



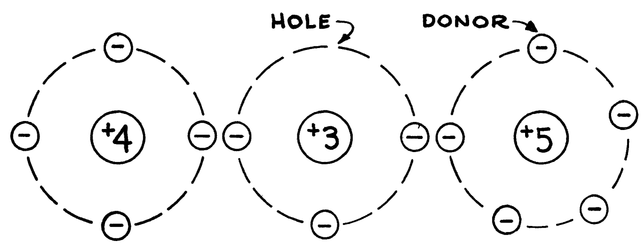
C. BASIC COMPUTER STRUCTURE HARDWARE



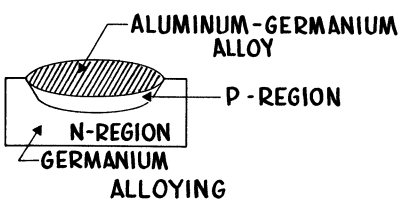
1. Circuit Components - Semiconductors

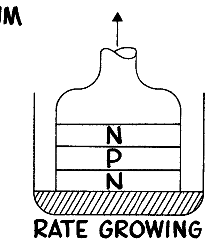
Category	Resistance Ω /cm^3	Material
Conductor	10^{-6}	Silver
Semiconductor	50-60	Pure Germanium
	50,000-60,000	Pure Silicon
Insulator	10^{12}-10^{13}	Mica

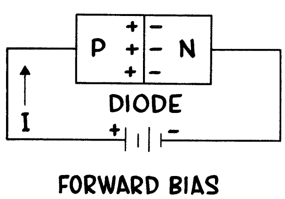
VALENCE BAND CHARACTERISTICS

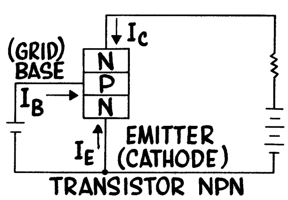


Semiconductor	Positive P-type Semiconductor doped with: Boron, Aluminum	Negative N-type Semiconductor doped with: Arsenic, Antimony





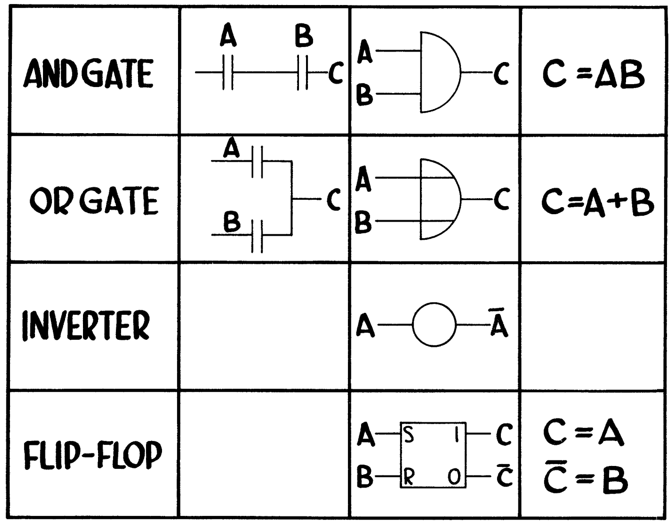




Transistor Junction Temperatures where transistor action ceases

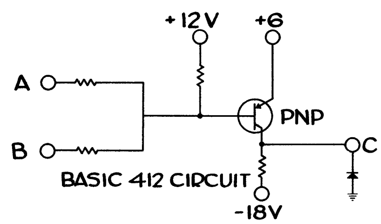
Germanium - 85°C
Silicon - 150°C

2. Logic Components



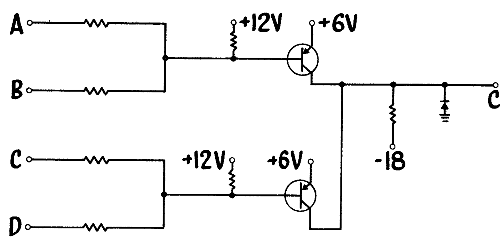
NOR LOGIC - NEGATIVE LOGIC SYSTEM

TRUE LOGIC STATE = 0V
FALSE LOGIC STATE = +6V

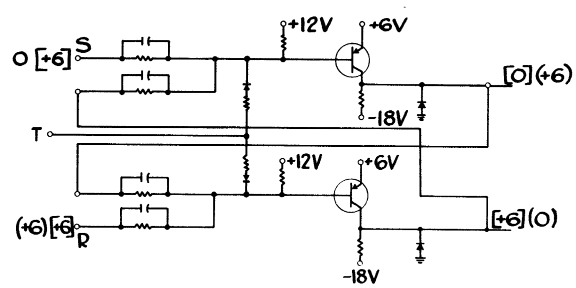


A AND B = 6V C = 0V
A OR B = 0V C = 6V

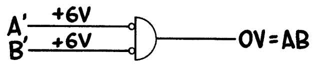
USES TRANSISTOR GATES FOR ADDITIONAL INPUTS



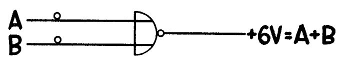
BASIC FLIP-FLOP CIRCUIT USED IN 412 COMPUTER



NOR GATE SYMBOLS

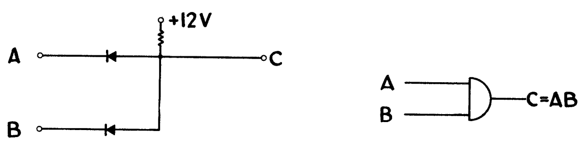


NOR AND

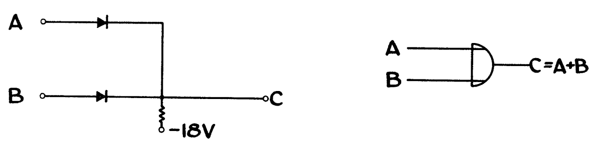


NOR OR

DIODE LOGIC



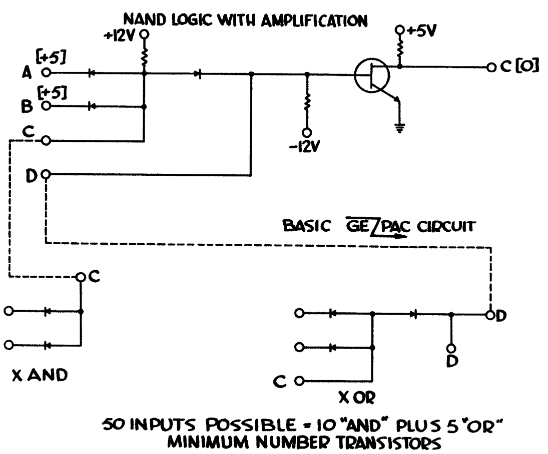
POSITIVE AND GATE



POSITIVE OR GATE

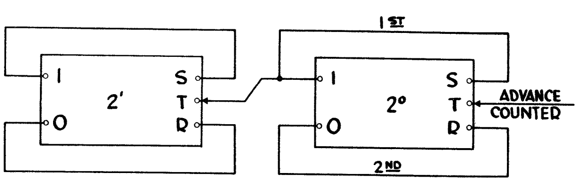
NAND LOGIC - NEGATIVE DIODE LOGIC

Positive "AND" becomes negative "OR"
Positive "OR" becomes negative "AND"



3. Logic Blocks and Control

TWO-STAGE BINARY COUNTER

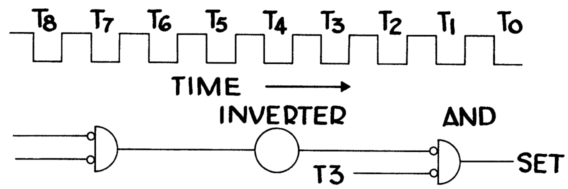


TRUTH TABLE

2^1	2^0
0	0
0	1
1	0
1	1

0 = Reset
1 = Set

TIMING CONTROL - 400KC OSCILLATOR



Control - The control component of the computer system automatically controls the operation of the other four components of the system.

(1) Stored Program - is a sequence of instructions stored in memory which are decoded by the control unit into actions to be performed by the input, output, memory, or arithmetic unit. One computer instruction (stored in one location - in memory) usually consists of two parts: (1) An Operation Code (i.e. ADD, SUBTRACT, etc.) and (2) and Operand Address (location in memory that contains the data to be used to execute the operation code).

(2) Execution of the Stored Program - is accomplished in a very definite way. It consists of repeating the following three operations for each instruction that is executed.

(a) An instruction is extracted from the location in memory indicated by the instruction counter in the control component and decoded in the control component.

(b) The instruction is executed during which time data may be extracted from or put into storage.

(c) The instruction counter of the control unit is advanced to indicate where the next sequential instruction is located in memory.

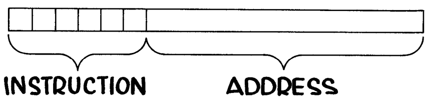
(3) Advantages of a stored program

(a) Speed - The speed at which instructions may be executed is limited only by the access speed of the memory device employed.

(b) Flexibility - Since the program is stored in memory as information (just as data is stored), changing the program is accomplished by simply loading a new sequence of instruction on top of (destroying) the old program.

(c) Versatility - Since each instruction is stored in memory in the form of binary information, the program can self-modify portions of the program to adjust to changing conditions. The program senses for these changes, and then treats certain of its own instructions as data to modify these instructions to fit the changing situation.

DECODING INSTRUCTIONS AND ADDRESS



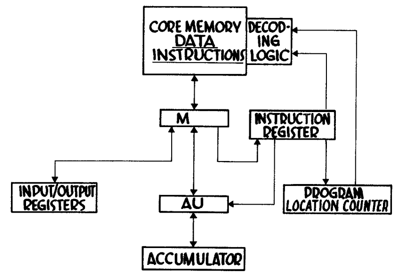
Instruction Code

001010 = Add M Register to A Register
001110 = Subtract M Register from A Register

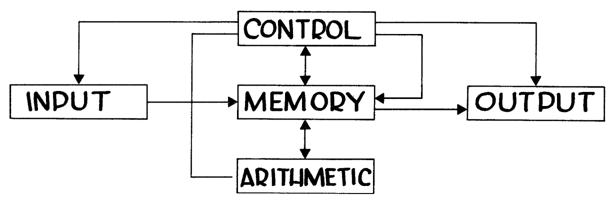
Address Code

0000000100100 = Address Location 36

4. Computer As Seen By User



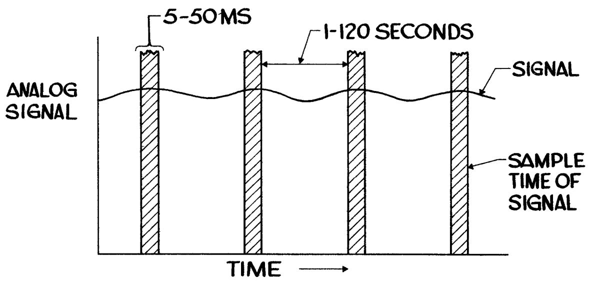
D. DIFFERENCES IN DIGITAL COMPUTER APPLICATIONS



	Business	Scientific	Process
INPUT	High Speed Large Volume Off-Line Preparation	Moderate Speed Small Volume	High Speed (Flexible) Large Volume On-Line Real Time
OUTPUT	High Speed Large Volume Off-Line Preparation	Slow Speed Small Volume On-Line Preparation	Flexible-Moderate Speed Small Volume On-Line Preparation Real Time
MEMORY	Small Volume	Large Volume	Large Volume
ARITHMETIC	Slow Speed Decimal	High Speed Binary	Moderate Speed, Flexible Bit Logic Binary
CONTROL	Read/Write/Compute Simultaneously	Compute Only	Input/Output/Compute Simultaneously
SYSTEM	Office Environment Daily Maintenance	Office Environment Daily Maintenance	Plant Environment Monthly Maintenance

IV ON-LINE SYSTEM INPUTS AND OUTPUTS

A. ANALOG SCANNING CONCEPTS



DATA SAMPLING

Scanning Speeds

Multipoint Recorder	1/3 Point/Sec.
Computer Scanning	20-40 Points/Sec.

B. RELAY MATRIX SWITCHING

Type	Speed	Noise Level
Mercury	Slow	Good
Reed	Medium	V. Good
Solid State	V. Fast	Poor

C. NOISE AND ACCURACY CONSIDERATIONS

1. Process Noise

   Solution - Computer averaging

2. Noise Pickup

   a. Instrument Power Supplies

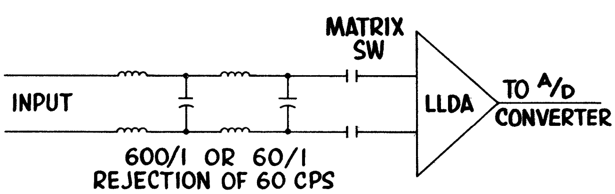
   b. 60 Cycle Noise

   c. Thyratron & Ignitron Equipment

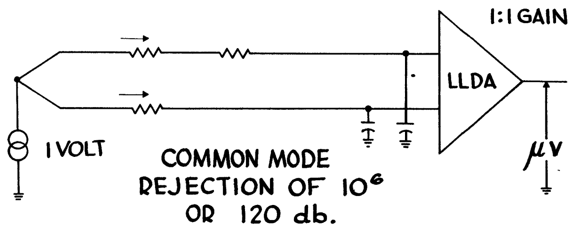
   d. Annunciator Systems

   e. Power Switching Equipment

Solution - Good intercabling practices, shielding and filtering of 60 cycle noise.



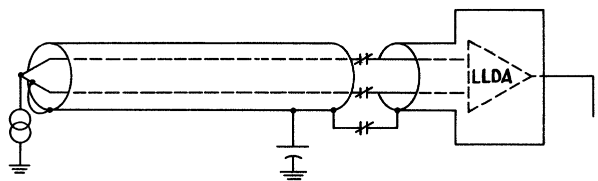
3. Common Mode Noise



Error with 120V AC on Sensor = .120 mV rms
726°F Cu. Constantan = 20 mV

Error = $\frac{.12}{20}$ x 100 = 0.6% rms Error

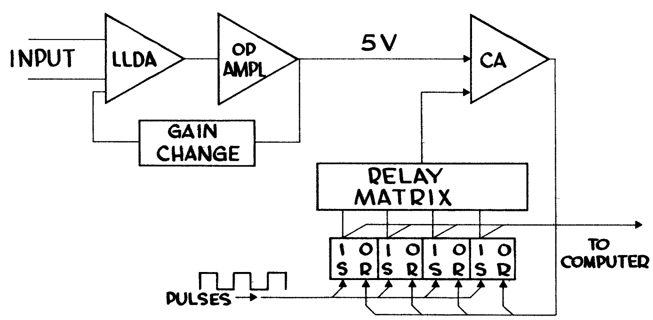
SOLUTION IS 3-WIRE SWITCHING



Amplifier and Input Wiring Float to Common Mode Voltage.

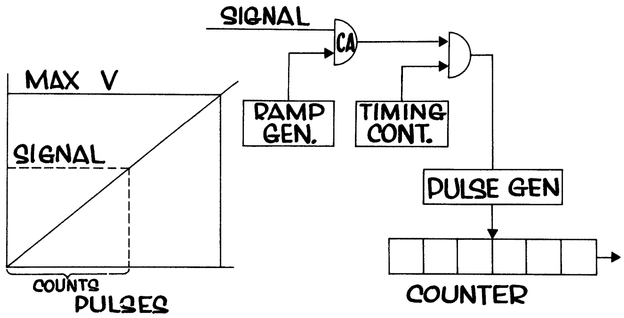
D. ANALOG-TO-DIGITAL CONVERSIONS

1. Successive Approximation

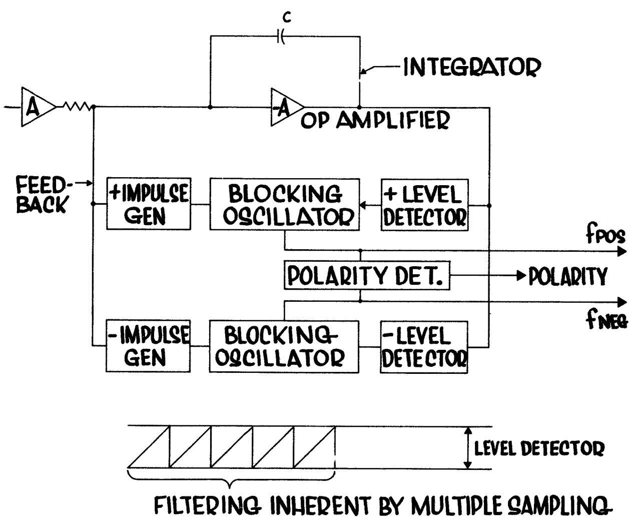


1st Try	1000 = 8V	RESET
2nd Try	0100 = 4V	
3rd Try	0110 = 6V	RESET
4th Try	0101 = 5V	COMPLETE

2. Ramp Method

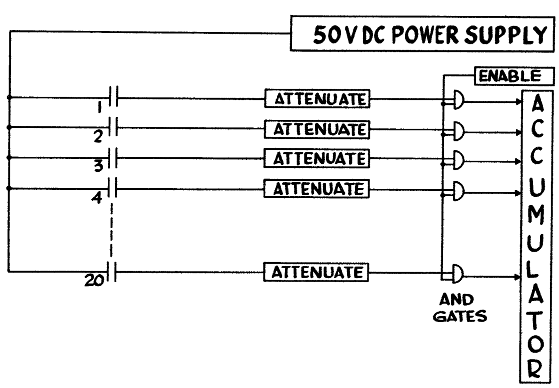


3. Integrating Amplifier Voltage to Frequency Conversion

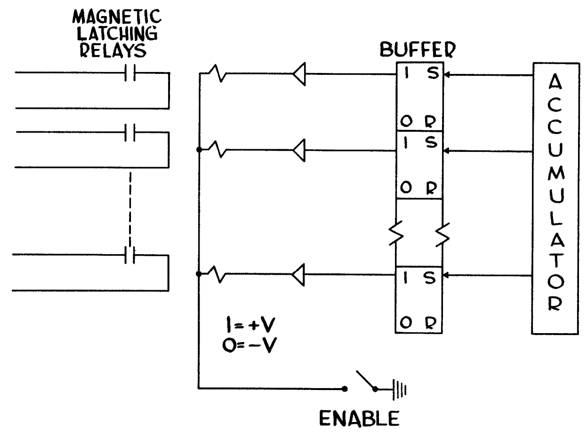


E. DIGITAL INPUTS AND OUTPUTS

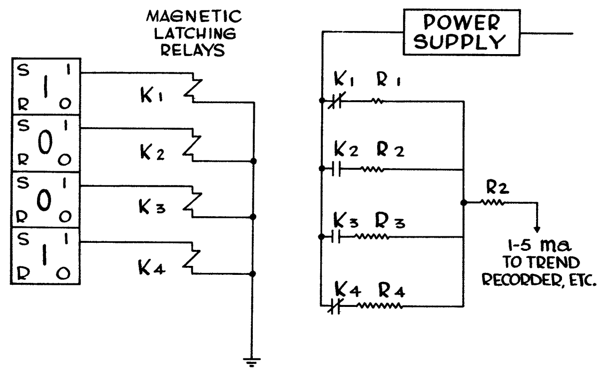
1. Contact Inputs



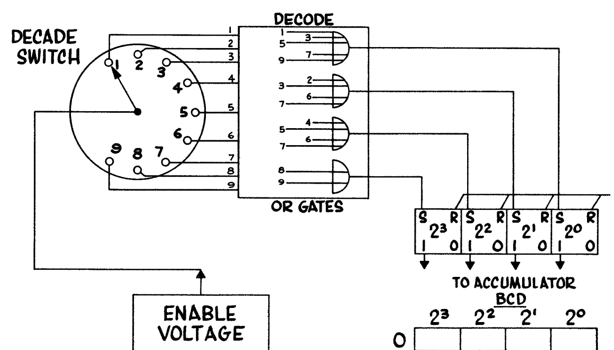
2. Contact Outputs



3. Digital-To-Analog Outputs



4. BCD Inputs

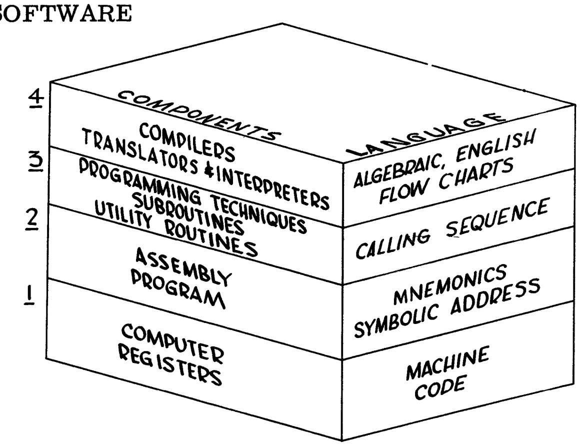


BCD

	2^3	2^2	2^1	2^0
0				
1				1
2			1	
3			1	1
4		1		
5		1		1
6		1	1	
7		1	1	1
8	1			
9	1			1

V BASIC PROGRAMMING STRUCTURE

SOFTWARE



1. MACHINE CODING

Every digital computer must be controlled by providing the precise logical instructions in the proper sequence. Each instruction in turn must be made up of the proper binary bit configuration. The instruction contains two basic pieces of information in a single address machine; the instruction (an action verb) and the operand address (the object) as shown below.

INSTRUCTION	ADDRESS	INSTRUCTION REGISTER
(1 1	0 0 1 0 0 0)$_8$	ADD CONTENTS OF (1000)$_8$ TO THE ACCUMULATOR
(32	0 0 0 1 0 0)$_8$	STORE CONTENTS OF ACCUMULATOR IN LOCATION (100)$_8$

The octal instruction 11 means add and 32 means store. Every instruction in the computer memory will appear as numbers as shown above.

Programming aids (Software) have been developed to relieve the programmer from having to have intimate knowledge of the computer hardware and to lessen the burden of the many bookkeeping type functions.

2. ASSEMBLY PROGRAMS

The first refinement to machine coding was the development of a method of allowing a computer program to assign the addresses. Each portion of a program was written relative to a starting address. When the program was completed, the assembly program translated the relative addresses into absolute addresses. This allowed several programmers to code portions of large programs without assigning actual addresses in advance.

The second refinement to machine coding was the use of mnemonics (aids to memory) to write instructions. For example, ADD would be written instead of 11 and STA (Store accummulator) instead of 32. The assembly program translated these mnemonic codes into machine instructions.

The third refinement was the ability to carry remarks along with each instruction without disturbing the operation of the assembler.

The use of symbolic addressing, i.e., symbols or names rather than numbers, and the ability of assembler to perform simple arithmetic to obtain an address were further refinements to assembly programs.

An assembly program is by definition a one-to-one translation of a written instruction into the computer code (sometimes called the object program). Some assembly programs go beyond this definition to provide compiler like instructions. These instructions are called pseudo instructions because there is not a machine instruction to substitute for the pseudo command. A pseudo instruction is recognized by the assembler and several machine instructions are used to implement the command.

ASSEMBLY PROGRAM EXAMPLES

Symbolic Location of Instruction	Operation Code	Symbolic Address	Remarks
START	LDA	VALUE	Initial Value = 100
	ADD	CON	
	SUB	FIVE	
	STA	WS	Working Storage
	BRU	START +17	
VALUE	DEC	100	Pseudo Inst. to Define Constants and Memory Space
CON	DEC	32	
FIVE	DEC	5	
WS	BSS	10	

3. PROGRAMMING TECHNIQUES

Programs that are common to most computer applications, such as BCD to binary conversion, binary to BCD conversion, mathematical routines, etc., are supplied by the computer manufacturer in the form of library subroutines. The use of these subroutines can greatly reduce the over-all programming effort. Each library subroutine specifies a calling sequence for bringing the proper information to the subroutine, and specifies the location of the return to the main program.

Subroutines

a. Common function (i.e., square root, sin, etc.)

b. Used many times during one large program.

c. Is written and checked out once, but used many times.

d. Conserves programming effort.

e. May be used on similar computers with no change.

f. Large Library of subroutines is available to choose from.

Utility Routines

a. Program Loaders

(1) Load program into memory from punched cards or paper tape.

(2) Simplifies loading memory.

b. Memory Dump Routines

(1) Records contents of memory permanently on paper tape.

(2) Tapes may then be used to reload memory.

c. Trace Routines

(1) Prints out contents of important registers after each program step.

(2) Aid in debugging programs.

d. Diagnostic Routines

(1) The computer program helps find where computer components are marginal in operation.

(2) Aid in preventative maintenance.

4. TRANSLATORS, INTERPRETERS AND COMPILERS

A program that will translate between one computer code to another computer code is called a translator program. In some business and scientific computer applications, the cost of reprogramming the existing routines for use on a newer computer has represented a major investment. The use of a translator program will automate the conversion from one machine to the next. Unfortunately translator programs are difficult to write for computers with large differences in the command structure, it is difficult to obtain an optimum program in both memory space and running time, and some manual coding is usually required.

Another approach to automatic coding is the use of interpretive coding. An interpretive program can be viewed as a large number of subroutines. Each time the subroutine is executed, the subroutine must go back to the calling sequence to determine, or interpret, the information contained there as to what information it is to operate upon, where to place the results, etc. This interpretation must be carried out each time the subroutine is executed, and the calling sequence is written in a format which resembles that of instructions. The TASC program for the GE 412 and GE/PAC 4000 is an interpretive program.

The GE/PAC 4000 makes use of the interpretive features along with special hardware features to provide quasi instructions. These instructions, such as the floating point arithmetic instructions, are short subroutines that are addressed by the instruction.

Compilers are programs that allow the programmer to write English statements or mathematical expressions. The program is defined in a language much closer to that of the problem being solved. The use of compilers provide the following:

a. Translates mathematical statement, accounting type sentence that defines the problem into an actual computer program.

b. Minimizes time required to get an operating program.

c. Makes programming much easier.

d. Usually more expensive to write, and less efficient in use of computer memory and computer time.

e. Provides good documentation of the program because English statements are used.

The compiler programs automatically produce the object (machine code) program. It is only necessary to compile a program once, and the compiler computer system need not be the actual computer on which the program is to be run.

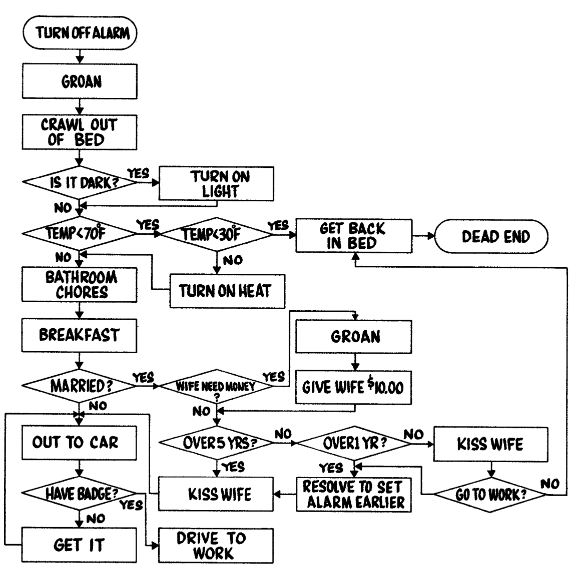
VI CONCEPTS OF PROGRAMMING

A. FLOW CHARTING

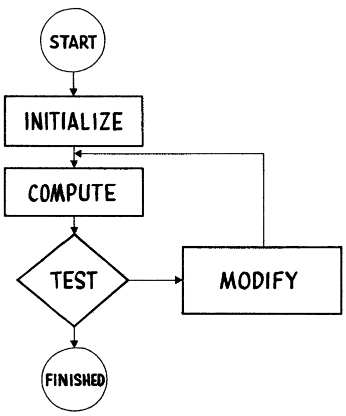
1. Advantages of Flow Charting
   a. Common Format for Complete Problem Definition
   b. Logical Method of Defining the Problem
   c. Easily Read and Explained
   d. Used to make computer memory requirement estimates
2. Symbols

SYMBOL REPRESENTATION	RECOMMENDED FOR ASSOCIATION OF COMPUTER MACHINERY (ACM)	COMMENTS
CARD		
TAPE		MAGNETIC OR PAPER
PRINTED DOCUMENT (OUTPUT)		
OPERATION FUNCTION		
DIRECTION OF FLOW		FOLLOW THE ARROWS MAY READ AS "GOES INTO"
FIXED CONNECTOR		SAME SYMBOL
VARIABLE CONNECTOR		ONE IN, MANY OUT
COMPARISON DECISION TEST		2 OR 3 WAY SPLIT
CLOSED SUBROUTINE OR LIBRARY SUBROUTINE		
ALARM		INDICATES ALARM PRINTOUT IN PROCESS COMPUTER CONTROL SEQUENCES
EXCEPTIONS ERROR, ERROR HALT		ONE SYMBOL CAN EASILY REPRESENT ALL THESE EXCEPTIONS- ERROR, ERROR HALT, ENTER, EXIT, START, STOP
STOP		
START		
FLAG EXPLANATION ASSERTION		A BROKEN LINE FROM A FLAG TO THE FLOW LINE IS SUFFICIENT TO REPRESENT AN ASSERTION OR EXPLANATION

3. Types of Flow Charts
   a. General
      (1) Getting Up in the Morning

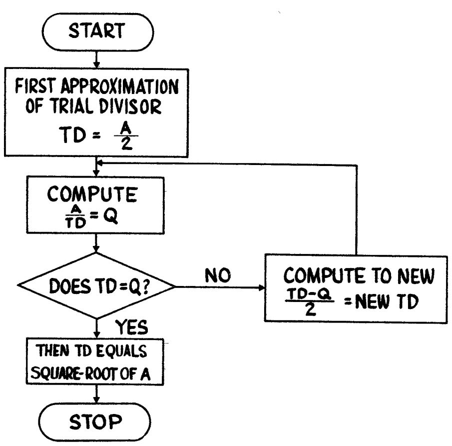


B. BASIC ELEMENTS OF A LOOP



Square Root

Symbol Definition
A = Argument (the number to take the square-root of)
TD = Trial Divisor
Q = Quotient



C. CODING EXAMPLE

Find Square Root of A

DATA

Location	Contents
1000	A
1001	TD
1002	Q
1003	Constant (2)10

PROGRAM IN MACHINE CODE

Location Instruction	Operation Code	Operand Address	Comments
2000	LDA	1000	Load Argument
2001	DVD	1003	Divide A by Z = TD
2002	STA	1001	Store TD
2003	LDA	1000	
2004	DVD	1001	A/TD = Q
2005	STA	1002	Store Q
2006	SUB	1001	Q - TD
2007	BZE	2011	Test Difference = ZERO
2008	LDA	1001	If Not Zero Then
2009	ADD	1002	$(TD + Q)/_2$ = New TD
2010	BRU	2001	Return To Beginning To Store TD
2011	LDA	1001	If Difference = O Then TD = A
2012	ETC.		Continue

PROGRAM IN ASSEMBLY LANGUAGE

Symbolic Location Of Instruction	Operation Code	Symbolic Operand Address	Comments
A	EQU	1000	Pseudo Instructions Are
TO	EQU	1001	Used To Define Location
Q	EQU	1002	Of Data And Constants
TWO	EQU	1003	
TWO	DEC	2	Defines Contents Of Location TWO
	ORG	2000	Pseudo Instruction Defines Beginning Location Of Program
SQROOT	LDA	A	Load Argument
LOOP	DVD	TWO	Divide By 2 = TD
	STA	TD	Store TD
	LDA	A	Load Argument
	DVD	TD	Divide By TD
	STA	Q	Store Q
	SUB	TD	Q - TD = Difference
	BZE	END	Test To See If Difference = O
	LDA	TD	If Not Then Try Again
	ADD	Q	TD + Q/2 = New TD
	BRU	LOOP	Return To Location Loop
END	LDA	TD	TD = A
	ETC.		

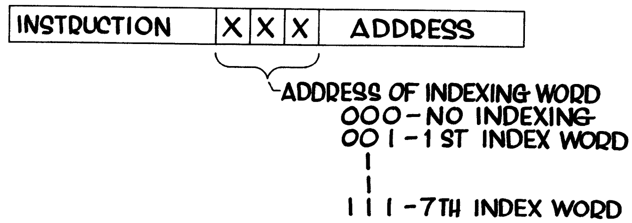
PROGRAM IN COMPILER LANGUAGE

Statement Number	Statement
1	TD = A/(2)
2	Q = A/TD
3	If TD = Q, GO TO STATEMENT 6
4	TD = (TD + Q) / (2)
5	GO TO STATEMENT 2
6	SQUAREROOT OF A = TD
7	END

D. INDEXING

One of the most powerful features of a stored program digital computer is the ability to change or perform arithmetic operations on instructions as well as data. The ability to address modify by indexing an instruction is an example of this feature. Special index registers (or words) are provided to modify the address of any instruction. The contents of the specified index register is added to the instruction address before execution. Thus, an indexed instruction produces an effective address.

INSTRUCTION FORMAT



EXAMPLE OF INDEXING

Add 500 Words Beginning At Location "DATA" and Store Answer in "SUM".

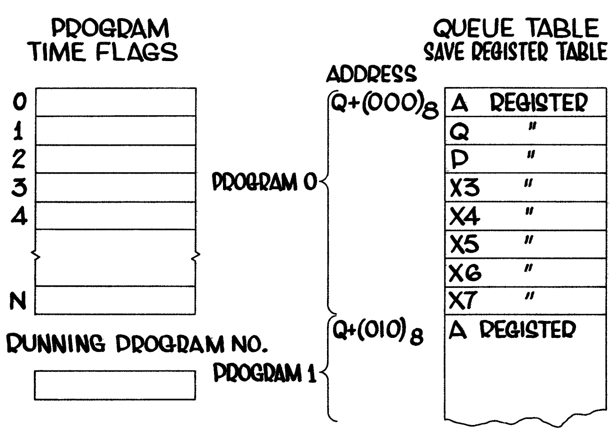
Symbolic Location	Operation Code	Symbolic Address	(Index) Tag	Comments
START	LDX	ONE	1	Load X1 With 1
	LDA	DATA		Bring 1st Value To Accumulator
	ADD	DATA	1	Add (DATA) + (DATA +1)
	BXH	499	1	Branch X1 $\geq$ 499
	BRU	START + 7		X1 $\geq$ 499, Branch
	INX	1	1	X1 < 499, Store Answer
	BRU	START +2		Add Again
	STA	SUM		Store Answer
	BRU	ECP		
ONE	DEC	1		
DATA	BSS	500		
SUM	DEC	0		

VII CONCEPTS OF REAL-TIME PROGRAMMING

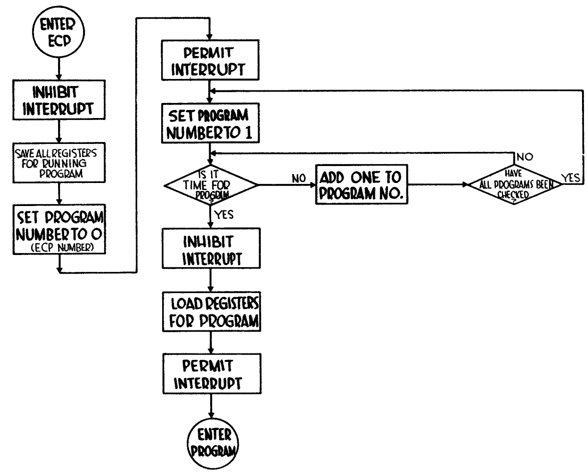
A. PROGRAM CONTROL WITH EXECUTIVE CONTROL PROGRAM.

Real-time programming involves the coordinated operation of many programs on a time-shared basis. A priority system must be established so that the most important functions are executed first. All the programs must be capable of queuing up so that they will eventually be serviced. The executive control program (ECP) is the controller for the use of "real-time" in any process computer application. It provides the queue for all programs waiting for service, and decides which program should be executed first.

Each functional program is designed to run on a fixed-time basis or on demand. Upon the completion of a functional program, the program updates the time flag to indicate when the program is to again be executed. A "real-time" clock within the computer memory is also updated on a second-by-second basis. Each second, or more often, the ECP program is entered and the registers are saved for the running program in the proper location of the queue table. The ECP now searches for a higher priority program to be executed by comparing the present time to the time flags for each program. If it is time to execute a higher priority program, the registers are loaded from the proper locations in the queue table and the running program number is changed. If there are no higher priority programs to be executed, the registers of the running program are again loaded from the queue table and execution continues on the running program.



EXECUTIVE CONTROL PROGRAM



B. PRIORITY INTERRUPTS

Process control computers make use of special hardware to automatically interrupt the running program in order to service a function of higher priority. In a data processing system, the slow input/output devices become the system bottleneck. Therefore, the priority interrupt levels are set first with the program requiring highest priority and using the slowest peripheral device. Thus, the priority interrupts will keep the input/output devices running at their maximum speed.

EXAMPLE OF PRIORITY INTERRUPT LEVELS

DEVICE	TIME OF DEVICE	FREQUENCY OF EXECUTION
1. Analog Scanner	50 Milliseconds	Continuous
2. System Time Counter	1 Second	Continuous
3. Alarm Printer	200 Milliseconds	Non-continuous
4. Log Typewriter	100 Milliseconds	Non-continuous

The order of events that take place upon a priority interrupt is shown below.

1. The control senses an automatic interrupt.

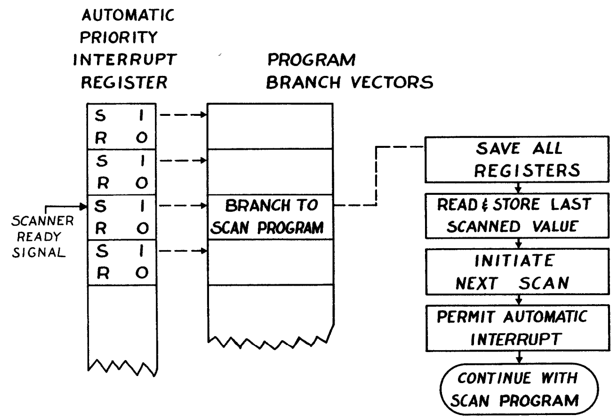
2. Further interrupts are inhibited until permitted by the program instruction.

3. The program location counter must be stored automatically to insure a proper return to the interrupted program. Additional registers may be stored by the hardware automatically.

4. Search is made for the device that caused the interrupt. This search is from the highest priority to the lowest.

5. A program branch instruction directs the location counter to the proper program to be executed. The interrupt is reset.

6. Additional registers are stored in the ECP program area if required by the interrupting program.



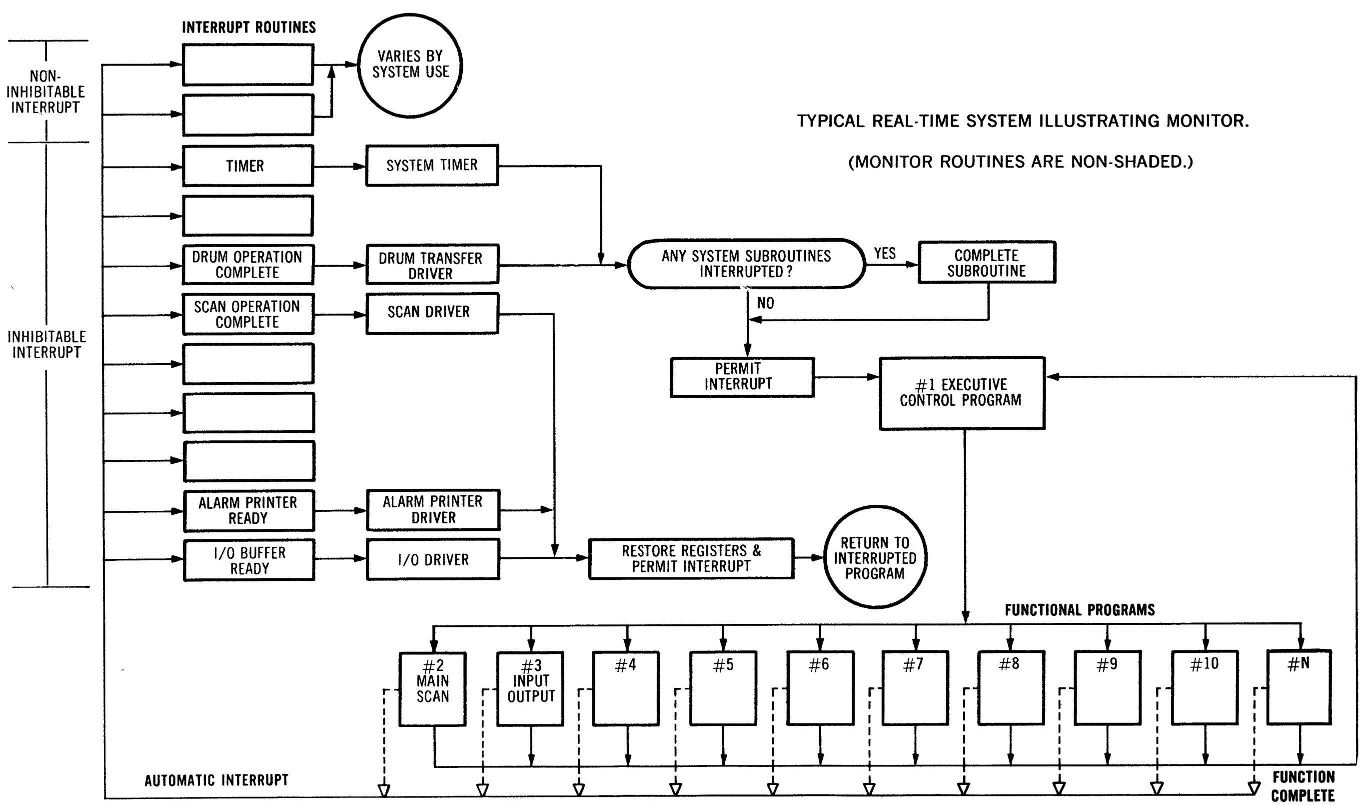
C. MONITOR SYSTEM

The monitor system is an operating system program which provides the basis for a process computer system. It accomplishes the timing and scheduling operations through the ECP program, and the basic input and output functions through the automatic priority interrupt system. A functional diagram of the Monitor system is shown below. Monitor provides:

1. Real-time scheduling
2. Priority assignment of computer functions
3. Scanning on analog inputs
4. Limit checking
5. Alarming
6. Converting to engineering units
7. Averaging
8. Input/output of digital information
9. Diagnostic checking of peripheral equipment.

With the monitor system, a functional program such as performance calculations, etc. can easily be inserted into the systems without disturbing the other functions.

A SAMPLE MONITOR SYSTEM



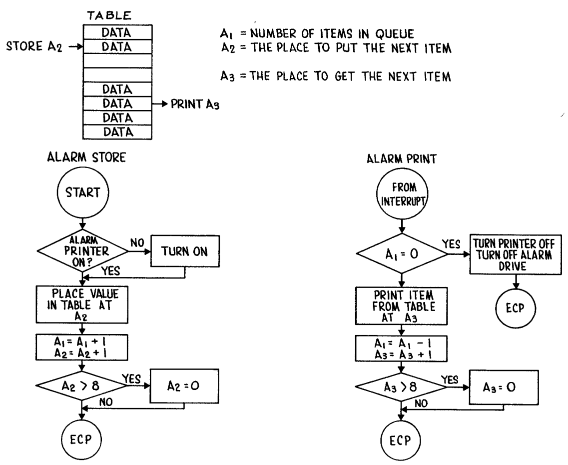
D. REAL-TIME PROGRAMMING TECHNIQUES

Several important concepts are involved in real-time programming as follows:

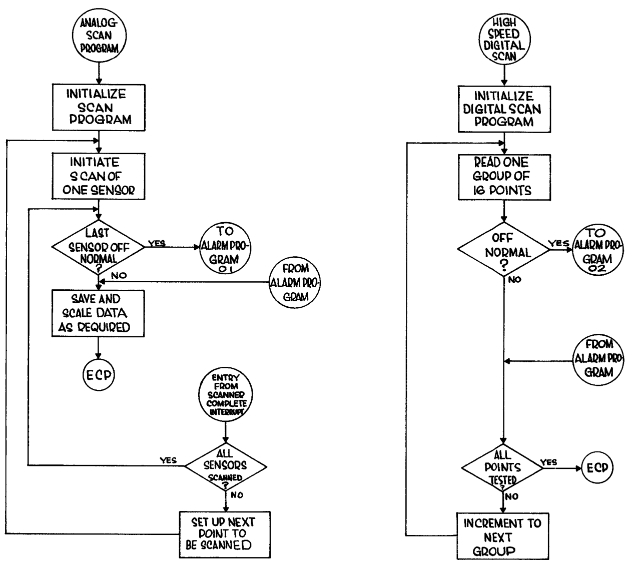
1. The system program never stops.
2. All programs must operate together in a system.
3. Debugging is simplified if each program is short and functional.
4. Programs must share core memory areas and may have to be segmented.
5. Circular queuing tables are used to prevent overflows and obtain latest alarms, etc.

A sample flow chart for a circular queue is shown below:

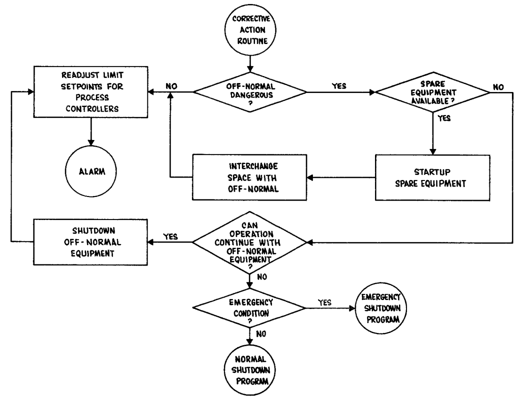
CIRCULAR QUEUE



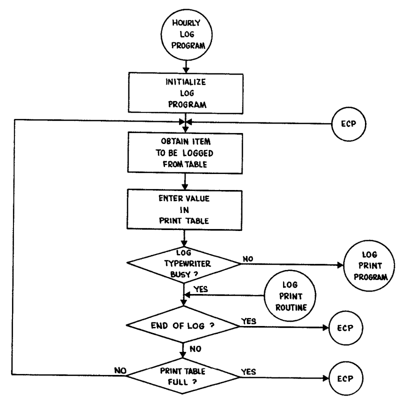
SCAN ROUTINES



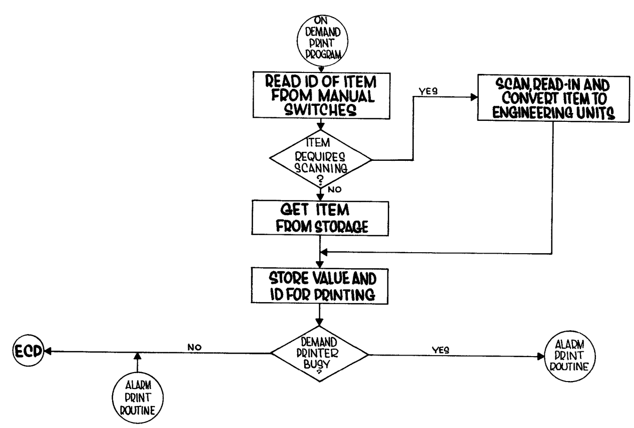
CORRECTIVE ACTION ROUTINE



HOURLY LOG ROUTINE



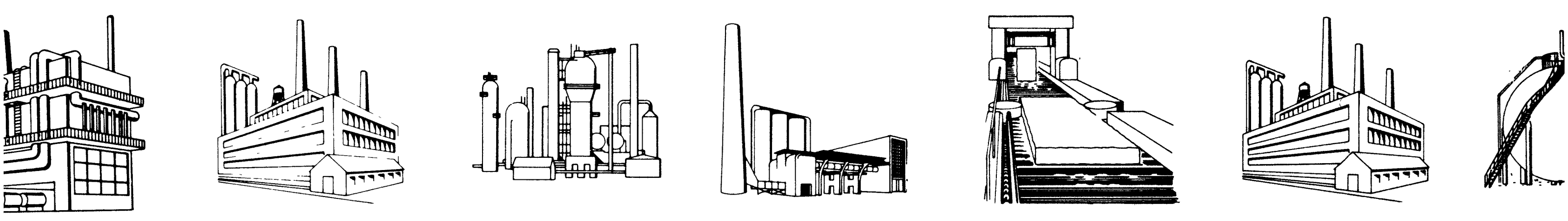
ON-DEMAND PRINT ROUTINE

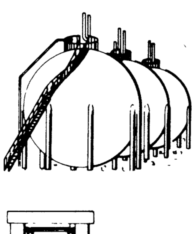


FOR QUALITY...
PRODUCTIVITY
... PROFIT

PROCESS COMPUTERS
FOR AUTOMATION

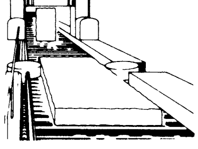
PROCESS COMPUTER SECTION PHOENIX, ARIZONA





Progress Is Our Most Important Product

GENERAL ELECTRIC



10-64 (2.5M)

Litho in U.S.A.