

*The direct electric analogy computer, developed largely through research in Caltech's Computing Center.*

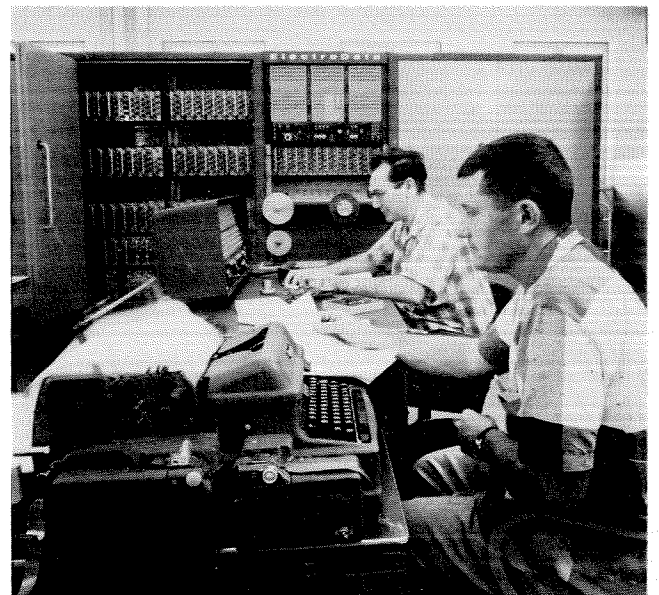
# Caltech's Computing Center

The importance of large-scale machine computing techniques to engineering and science has been growing steadily for the past ten or twelve years. These techniques have become an essential element of our modern technology, making possible the rapid analysis and development of the complex systems and machines of today.

This has placed new emphasis on a number of basic disciplines — including such theoretical subjects as Boolean algebra and symbolic logic, and their application to switching theory and the logical design of computers. The need for higher-speed, more compact methods of physical instrumentation has spurred research in such basic areas of solid state physics as semiconductors, ferromagnetics and the superconductivity of materials at low temperatures. Of particular importance at present is the study and application of the physical properties of very thin films, a few hundred angstroms in thickness. The similarity between computer logic and human brain functions has brought together mathematical and biological research in this area, and such fields of mathematics as numerical analysis and engineering analysis have also taken on greater importance.

Caltech's interest in this general field became significant towards the end of World War II, and actually started in two areas. The physical chemists developed a need for a digital computing facility more comprehensive than the conventional desk calculator, and a small IBM computing facility was set up. Also, the engineering division became interested in the development and application of machine computing

*by G. D. McCann*

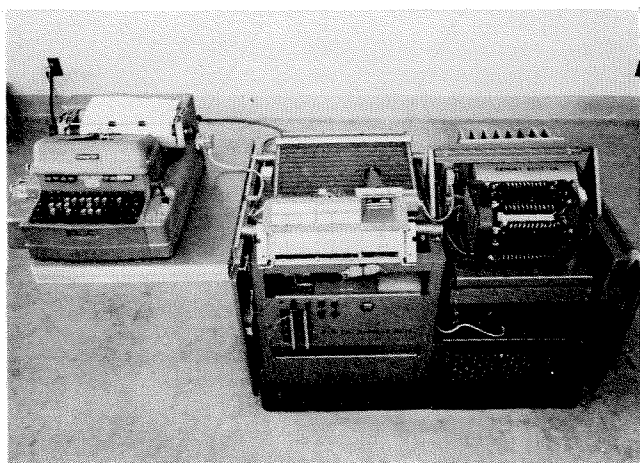


*The ElectroData Datatron — 205 digital computer.*

techniques to complex system design problems, and an analysis laboratory was established in 1946. The initial efforts of this laboratory centered around analog computing techniques. From 1946 to 1956, extensive work was conducted in electric circuit synthesis and instrumentation, which resulted in the development of a new computer, the direct analogy computer. Applications of this computing technique were made to design analysis problems in the fields of solid mechanics, fluid mechanics, and heat transfer. Methods of structural stress analysis, vibration analysis, and the aeroelastic analysis of airframes were developed. This research was carried out primarily by Charles H. Wilts, Richard H. MacNeal, B. N. Locanthi and by the writer.

By 1956, the need for a more comprehensive computing facility that would include a large-scale digital computer had grown to the point where plans were made for the addition, as a general campus facility, of the ElectroData Datatron 205 computer. This, along with a Librascope LGP-30 computer developed from research here at the Institute, and the direct analog computer, now constitutes Caltech's Computing Center facility, on the first floor of the Spalding Laboratory of Engineering.

Among the professors of mathematics who have recently joined the faculty to work in these areas are John Todd, in the mathematics department; and J. N. Franklin, in applied mechanics. Others engaged in this program are Robert Nathan, Kendrick Hebert, Charles Ray, and Robert Harder. They, along with Charles Wilts and the writer, are engaged in research and instruction in the new courses which have been developed in numerical analysis, machine design, and in engineering analysis and synthesis. The computing facility has come to be widely used on the campus, and each year, through coding courses, about 75 to 100 students and staff learn the Datatron programming techniques. The scope of its applications to general campus research is indicated in the table at the right.



*The Librascope LGP-30 digital computer.*

## CAMPUS RESEARCH USING THE COMPUTING CENTER

### Astrophysics

- Satellite orbit calculations
- Reduction of photoelectric spectrum observations
- Calculation of the rate of expansion of the universe

### Biology

- Mathematical modeling of plant growth
- Simulation of brain functions

### Chemistry

#### *Physical Chemistry*

- Crystallography, chemical-molecular structure determination of compounds
- Calculations for X-ray and electron diffraction
- Chemical kinetics through classical chemistry and through shock wave reaction
- Molecular orbital calculations

#### *Chemical Engineering*

- Chemical kinetics
- Simulation of pilot plant operations

### Applied Physics and Engineering

- Network synthesis
- Noise and auto-correlation
- Switching circuit design and analysis
- Information theory
- Pattern recognition
- Servomechanism analysis
- Solid mechanics – dynamics and vibration analysis
- Thermal heating – radiation and conductivity transport
- Aeroelasticity – flutter and aircraft design analysis
- Thermal – aeroelasticity in high speed flight
- Fluid mechanics and cavitation
- Linear and nonlinear ion diffusion analysis
- Microwave tube analysis and design – calculation of electromagnetic field problems with interacting electron beams
- Plasma calculations and magnetohydrodynamics
- Antenna and artificial dielectric problems

### Mathematical and Numerical Analysis

- Development of new methods for aeroelasticity stability calculations
- Development of new methods for solving linear and nonlinear diffusion equations
- Monte Carlo methods for nuclear physics
- Determination of largest circles within which the basic special functions of mathematical physics are univalent

### Physics

- Electromagnetic field calculations
- Nuclear partial path calculations
- Molecular distribution in centrifuges

Since machine computing has become so important to most fields of engineering and science, the basic principles of computers and their applications should become more commonly known. Here is a brief description of the more important analog and digital computers.

### Analog Computers

Analog computers are based on the general principle of measuring the performance of some physical system whose properties can readily be changed to simulate a variety of other physical systems, or their defining mathematical equations. These are usually algebraic or differential equations. The more common analog computers today use electrical systems whose electrical parameters can be quickly varied, and whose circuits are easily hooked up in different forms. Currents and voltages represent the exciting functions and are measured as solutions to the problem under study. Two of the most important of our modern electric analog computers are the electronic differential analyzer and the direct electric analog computer which was developed largely through research in Caltech's Computing Center.

In the electronic differential analyzer, electronic voltage amplifiers with simple resistor-capacitor feedback circuits can be used to produce the integral of a voltage with respect to time, or to sum several voltages. The diagram below illustrates how a circuit can be built up with these two elements in order to simulate a simple ordinary differential equation with constant coefficients. In this circuit the voltage  $e$  at the output of the right hand integrating amplifier simulates the function  $y$  in the original differential equation. Differential analyzers are in extensive use today, both as analytical tools for the development of (or as actual component parts of) complex systems—for guidance control, for example, or for commercial

process control. The annual production of such equipment runs into the hundreds of millions of dollars.

The direct analog computer is an extension of the concept of synthesizing mathematical equations with electrical components, using many passive circuit components in addition to amplifiers. The more important of these are inductors, transformers, resistors, and capacitors. Since this concept provides direct analogies between both voltages and currents in the computer and such variables as velocity and force in a mechanical system, the comprehensive extension of the method to the use of amplifiers for simulating non-conservative elements also requires a current generator. This type of computer has been used extensively at Caltech and in the aircraft industry for the design analysis of airframes.

### Digital Computers

In distinct contrast to analog computers, digital computers deal directly with symbols which may be both alphabetic and numeric. They process the symbols at high speeds in various ways, including such logical operations as arithmetic, "bookkeeping," and "decision making." To understand their operation, consider a relatively simple general purpose computer, the Librascope LGP-30, in the Institute's Computing Center. This machine is based on research carried out in our laboratory several years ago under the direction of Stanley Frankel.

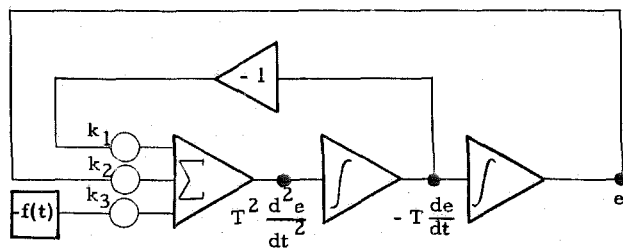
It is described as "a 4096 word magnetic drum memory, 32 'bit' (binary digit) word, single address, fixed decimal point, stored program computer using vacuum tube flip-flops and diode logic circuits." Its input-output device is a Flexowriter and paper tape.

The basic principles of this computer are illustrated on page 27. The magnetic drum has 64 tracks of main storage memory, each track having a magnetic head which can write into the track with current pulses or read out the contents of the track as voltage pulses which control flip-flops. Sixty-four 32 bit words can be stored on each track, which may be either a number or a command word. The computer is serial because each word is presented in time sequence, one bit at a time. On the left end of the drum there are four permanently magnetized timing tracks for controlling its operation. One of these produces "clock pulses" for timing each individual bit. On the other end of the drum there are three recirculating registers, each using two separate heads to permit recirculatory storage of a word. The functions performed by these registers are indicated in the diagram, and by the command list on page 27.

It is known as a single address-stored program computer because command words consisting of a 4 bit command and a single 12 bit reference address (6 bits for track number and 6 bits for sector number) are taken in sequence out of memory and executed. As shown by the four modes of operation

Simple Ordinary Differential Equation

$$\frac{d^2 y}{dx^2} + A \frac{dy}{dx} + By = f(x) \quad \text{or} \quad \frac{d^2 e}{dt^2} = - \left[ A \frac{de}{dt} + k_2 e - k_3 f(t) \right]$$



$$T^2 \frac{d^2 e}{dt^2} = - \left[ k_1 T \frac{de}{dt} + k_2 e - k_3 f(t) \right]$$

How amplifiers are used for transferring between routines.

in the diagram of the digital computer at the right, the computer searches for the memory address given in the C register, places the command found there in the R register (also adding one to the C register number), performs the command indicated in the R register, and then repeats with the new C register address.

All commands involve the word whose address is in the R register and the word in the A register. These commands are of three general classes: arithmetic, transportation or bookkeeping, and decision making. A typical example should show how a comprehensive set of commands can perform useful computing.

Let us develop a program for the LGP digital computer which will enable it to compute the stopping time and distance of a vehicle which is traveling at a series of velocities, for a fixed reaction or delay time in applying the brakes, and for decelerations that are some function of the vehicle velocity.

## COMMANDS FOR THE LGP-30 DIGITAL COMPUTER

### Instruction

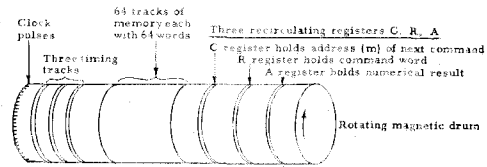
- |               |  |
|---------------|--|
| A* <i>m</i> + | Add contents of <i>m</i> to that of accumulator <i>A</i> and retain results in <i>A</i> .  |
| S <i>m</i>    | Subtract contents of <i>m</i> from that of <i>A</i> and retain results in <i>A</i> .   |
| M <i>m</i>    | Multiply number in <i>A</i> by that in <i>m</i> retaining the most significant 30 bits of results in <i>A</i> .  |
| D <i>m</i>    | Divide number in <i>A</i> by that in <i>m</i> retaining the rounded quotient in <i>A</i> .   |
| B <i>m</i>    | Place the contents of <i>m</i> in <i>A</i> .   |
| H <i>m</i>    | Store contents of <i>A</i> in location <i>m</i> but also retain it in <i>A</i> .   |
| C <i>m</i>    | Store contents of <i>A</i> in <i>m</i> and clear <i>A</i> .  |
| R <i>m</i>    | (Return address) - Add "one" to address held in <i>C</i> register and record this in the address portion of instruction held in memory location <i>m</i> . |
| U <i>m</i>    | Transfer control to <i>m</i> unconditionally or get next instruction from <i>m</i> . This <i>m</i> is put in <i>C</i> .                                    |
| T <i>m</i>    | Test or conditional transfer. Transfer control to <i>m</i> only if number in <i>A</i> is negative, otherwise continue in sequence.                         |
| E <i>m</i>    | Extract. Clear contents of <i>A</i> to zero in those bit positions for which <i>m</i> has zeros.   |

\* Symbol for command defined by "4 bits"

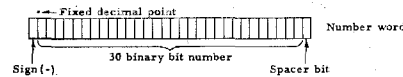
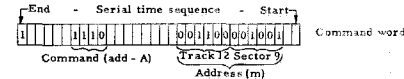
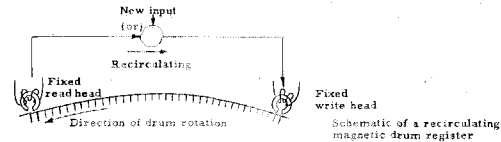
A=1110

+ Address or location in memory for which word involved in command is to be found

## DIAGRAM OF A SIMPLE "GENERAL PURPOSE" DIGITAL COMPUTER



A register. Its number and the number given by address in R will be operated on by command in R. Input to and output from computer flows through A.



### Four normal modes of operation

1. Search for word in memory whose address (*m*) is held in *C* register.
2. Transfer this command word to *R* register and add "1" to *C*.
3. Search for word whose address is held in *R* register.
4. Perform command indicated in *R* register on this word and contents of *A* register.

Let:  $t = \text{time}$   
 $d = \text{fixed time delay}$   
 $a = \text{acceleration (negative in this case)}$   
 $v = \text{velocity}$   
 $s = \text{distance traveled}$   
 $a = f(v)$

$$v(t) = v_0 + \text{Integral} (adt) \Big|_d^t$$

$$s(t) = v_0 t - \text{Integral} (vdt) \Big|_d^t$$

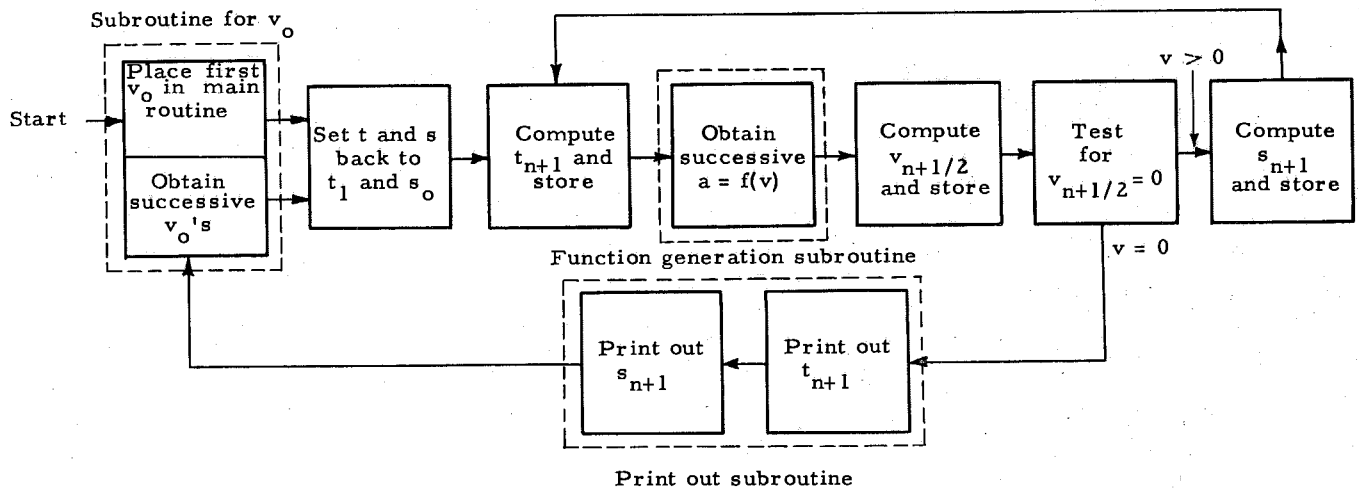
These are integral equations of calculus. The process of converting equations of this type to a form suitable for the computer, which only knows how to do arithmetic, falls in the realm of numerical analysis. In this simple problem a fixed time interval will be assumed and the equations for  $t$ ,  $v$  and  $s$  approximated as follows:

$$t_{n+1} = t_n + \Delta t; \quad t_1 = d$$

$$v_{n+1} = v_n + a \Delta t$$

$$s_{n+1} = s_n + (v_{n+1}) \Delta t; \quad s_1 = v_0 d$$

## FLOW CHART INDICATING THE COMPUTER PROGRAM FOR VEHICLE DECELERATION CALCULATIONS



$t$  = time  
 $d$  = fixed time delay  
 $a$  = acceleration (negative)  
 $a$  = a function of velocity  
 $v$  = velocity  
 $s$  = distance

$$\begin{aligned}
 t_{n+1} &= t_n + \Delta t & ; t_1 &= d \\
 a_n &= f(v_{n-1/2}) \\
 v_{n+1/2} &= v_{n-1/2} + a_n \Delta t \\
 s_{n+1/2} &= s_n + v_{n+1/2} \Delta t ; s_1 = v_0 d
 \end{aligned}$$

The operations of successively solving these equations are shown in the flow chart above, which illustrates some of the essential commands of such a machine.

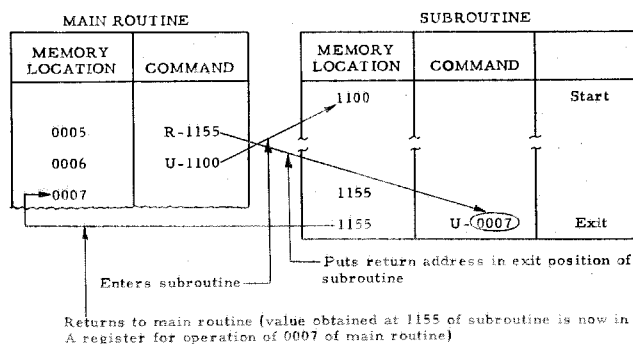
In preparing a program from this chart, it will be assumed that the processes of obtaining each successive value of the initial velocity  $v_0$ , the acceleration  $a$  as a function of  $v$ , and printing out the desired answers, are "subroutines" already prepared and stored in tracks 11, 12, 13. Therefore, a program will be given only for the main routine of this specific problem, and this will be written to be stored in track 00.

There are two commands (page 27) for the specific operation of splicing subroutines into a main routine. These are the R and U commands and their application to this type of program transfer is illustrated at the left below.

The complete program of commands for the main portion of this problem is given on page 32. The application of the one decision-making command given on page 27 is that of either repeating the  $t.v.s.$  calculations if  $v$  is still positive, or printing out the final  $t$  and  $s$  if  $v$  has reached zero. This is performed in steps 17 and 18 on page 32. In step 17, the smallest positive number is subtracted from  $v$ . If still positive, the test command of step 18 is ignored and step 19 is executed next. If the resulting number is zero, the command is transferred to step 24 to start the print-out operations.

The fact that this is a binary computer offers little complication to the user because it is quite practical to devise a routine to interpret and translate numbers written on the Flexowriter in decimal form, and commands written in alphabetic form. Such a translation code can either be kept permanently in part of the memory or stored on paper tape for input during translation. Permanent storage of a given type of routine in binary code can easily be affected by punching out on tape the actual routine placed in the machine by the translating program.

### HOW R AND U COMMANDS ARE USED ON THE LGP-30



*continued on page 32*

*Larger digital computers*

The LGP-30 has a total of only 16 commands but is capable, in principle, of doing any problem possible on any digital computer. It is used extensively for student training. However, large problems such

as complex partial differential equations would require a long computation time. The Datatron 205 is actually used for most of the service computing performed for the Institute. This, like almost all of the existing larger, higher-speed computers, is also of the stored program type. These computers achieve greater power because they have:

1. Higher operating speeds
2. Larger command lists
3. More memory
4. Higher-speed input and output

## MACHINE LANGUAGE ROUTINE

for Solving Vehicle Deceleration Problem  
on the LGP-30

Memory Location	Command	Explanation
0000	R-1155	Place first $v_0$ in
0001	U-1100	A register
0002	C-0041	Store $v_0$ in main routine
0003	B-0047	
0004	C-0042	Store $d=t_1$ as first $t_n$
0005	B-0041	Bring $v_0$
0006	M-0047	Compute $s_1=v_0d$
0007	C-0043	Store $s_1$ as first $s_n$
0008	B-0042	Bring $t_n$
0009	A-0044	Compute $t_{n+1}=t_n+\Delta t$
0010	C-0042	Store new $t_{n+1}$
0011	B-0041	Bring $v$
0012	R-1395	Place $a=f(v)$ in A
0013	U-1300	
0014	M-0044	$a\Delta t$
0015	A-0041	New $v_{n+1}=v_{n-1}+a\Delta t$
0016	H-0041	Store and hold $v_{n+1}$
0017	S-0046	Test for $v_{n+1}=0$
0018	T-0024	
0019	B-0041	Bring $v_{n+1}$
0020	M-0044	$(v_{n+1})\Delta t$
0021	A-0043	$s_{n+1}=s_n+(v_{n+1})\Delta t$
0022	C-0043	Store $s_{n+1}$
0023	U-0008	Return to calculation of new $t$
0024	B-0042	Bring $t_{n+1}$
0025	R-1261	
0026	U-1200	Print out $t_{n+1}$
0027	B-0043	Bring $s_{n+1}$
0028	R-1261	
0029	U-1200	Print out $s_{n+1}$
0030	R-1155	
0031	U-1106	Place next $v_0$ in A
0032	U-0002	
o		
o		
o		
0041	-	Storage space for $v$
0042	-	Storage for $t$
0043	-	Storage for $s$
0044	-	Storage for $\Delta t$
0045	-	$a$
0046	+0.0000001	Smallest number greater than zero
0047	-	Storage for $d$

*Automatic Indexing*—The Datatron has several important additional features not found in the LGP-30. One of these is the B register, a device for providing automatic indexing. By adding to the complexity of the command structure, it is possible to have any command which is processed to have its address adjusted at the same time by one or more other numbers. Thus, without using any extra commands, such operations as successive calculations can be stored successively in different locations, or successive numbers for a repetitive calculation can be brought up for the calculation. The Datatron can also perform “floating point” arithmetic in addition to fixed decimal arithmetic and it has a supplementary magnetic tape memory for the storage of 800,000 words.

*Automatic Coding*—The simple program on vehicle braking at the left gives some indication of the tedious and complex jobs of preparing a suitable program for each problem. Including the three subroutines, that problem requires about 160 commands. Truly comprehensive programs frequently have as many as a thousand or even ten thousand commands. The coding problem is therefore an important part of the operation of a practical computing center, and in the past many man-hours of coding time have been required for each computer-hour of calculation. Along with numerical analysis, coding research has constituted an important part of the research activity of our Computing Center. This work has been conducted largely by J. N. Franklin and Kendrick Hebert. They have been concerned with the development of methods for making the computer do the bulk of the actual detailed coding. Two general methods have been developed at a number of research centers for accomplishing this. One of these, developed by Franklin and Hebert, is known as a compiler routine.

*A Compiler Routine*—The most complex part of preparing a program such as the vehicle deceleration problem is in specifying the exact location of each word being processed. Each machine command must refer to a specific memory location. If a set of commands could be devised which pertain to a given

set of symbols that can represent variables in a given problem, and the computer could be made to keep track of these quantities by internal programming, this difficulty could be overcome. Suppose we devise a set of compiler commands such as:

$$c = a \times b$$

$a$ ,  $b$  and  $c$  are quantities to be defined for the computer, and it always knows where to find them. The operator does not concern himself with this. For the LGP-30 the following "machine" commands could produce this compiler command.

- 1 - Bring (address of  $a$ )
- 2 - Multiply (address of  $b$ )
- 3 - Clear (address of  $c$ )

For such an operation, a compiler routine must be devised where the computer first takes each symbol such as  $a$ , determines a space for its storage and then remembers it. It will next prepare a subroutine of machine commands as shown above for each abstract compiler command. The compiler routine developed by Franklin and Hebert has 1200 machine commands. It is stored permanently on the magnetic tape memory of the Datatron together with the subroutine library of our Computing Center. Some of the 18 abstract commands it recognizes and processes are listed below.

### PARTIAL COMMAND LIST FOR A SIMPLE COMPILER ROUTING

Command	Explanation
1 $x = a + b$	$a$ , $b$ and $y$ must be defined by input data or in previous commands. Computer has thus already assigned memory locations for them. In these operations mem. loc. is assigned for results which are stored there.
2 $x = ab$	
3 $x = a/b$	
4 $x = y$	
<hr/>	
5 Go to $s^*$	Unconditional transfer
6 $x; y; s; s'; s''$	If $x < y$ go to $s$ ; $x = y$ go to $s'$ ; $x > y$ go to $s''$
<hr/>	
7 Read input	Reads next block of input tape
8 Car. ret.	Move typewriter carriage to left-hand edge and advance roller one row
9 Car. ret., print $x$	Also prints word in location of $x$
10 Tab. and print $x$	Moves carriage to next tab stop and prints $x$
11 $y = f(x)$	Places in $y$ its value as function of $x$ for those function subroutines stored in magnetic tape such as $y = \sqrt{x}$ , $y = \tan x$ , etc.

\* In preparing the final machine language routine the compiler routine assigns a number ( $s$ ) to each compiler command identifying its location in the program.

### COMPILER ROUTING FOR VEHICLE DECELERATION PROBLEM

#### Program

- 1  $s = tv$
- 2  $t = t + T$
- 3  $a = \sqrt{v}$
- 4  $a = ka$
- 5  $b = aT$
- 6  $v = v + b$
- 7  $v; o: 11, 11, 8$
- 8  $c = vT$
- 9  $s = s + c$
- 10 go to 2
- 11 Cr print  $t$
- 12 tab print  $s$
- 13 read input

#### Input Data

The constant  $k$   
 $t$  = magnitude of  $d$ , the delay time  
 $T = \Delta t$   
 $v$  = initial velocity—successive values to be used are in subsequent blocks of input

An actual program, as prepared for the Datatron, will be put on two paper tapes. One will have a numbered list of the commands describing the problem. The other will list the initial values of all given quantities in one block of data, and subsequent new values of data to be used in successive blocks. The person originally devising the code will write it from the partial command list at the left. This is turned over to someone who punches the two Flexowriter tapes using a simple standard format. The code tape is put into the computer with the compiler preparation routine stored in its main memory. The computer then prepares its own "machine language" program which can be printed out on tape ready for use at any time.

*Compiler Routine for the Vehicle Deceleration Problem*—To compare this method with that of the direct preparation of a code in machine command language, the vehicle deceleration problem described before has been written in compiler form above. To illustrate the format for the function command (No. 11 in the partial command list) the specific square root function has been assumed for the acceleration. The list above shows that only 13 commands need be written by the user instead of the 160 needed for the complete machine code which is prepared automatically by the digital computer.

This is but one small example of the various areas of research being conducted in the Caltech Computing Center.