

- [54] **MICROPROCESSOR SYSTEM HAVING HIGH ORDER CAPABILITY**
- [75] Inventor: **Sydney W. Poland, Arlington, Tex.**
- [73] Assignee: **Texas Instruments Incorporated, Dallas, Tex.**
- [21] Appl. No.: **783,903**
- [22] Filed: **Apr. 1, 1977**

All-Semiconductor Memory", *Electronics*, Jun. 10, 1976, pp. 29-30.

*Primary Examiner*—Jerry Smith  
*Attorney, Agent, or Firm*—Stephen S. Sadacca; James T. Comfort

**Related U.S. Application Data**

- [63] Continuation-in-part of Ser. No. 714,464, Aug. 16, 1976.
- [51] Int. Cl.<sup>2</sup> ..... **G06F 3/08; G06F 15/02**
- [52] U.S. Cl. .... **364/706; 364/200; 364/900**
- [58] Field of Search ..... **235/156; 364/200, 900, 364/706**

**References Cited**

**U.S. PATENT DOCUMENTS**

|           |        |                         |           |
|-----------|--------|-------------------------|-----------|
| 3,593,313 | 7/1971 | Tomaszewski et al. .... | 340/172.5 |
| 3,760,171 | 9/1973 | Wang et al. ....        | 235/156   |
| 3,859,636 | 1/1975 | Cook ....               | 340/172.5 |
| 3,942,156 | 3/1976 | Mock et al. ....        | 340/172.5 |
| 3,953,833 | 4/1976 | Shapiro ....            | 340/172.5 |
| 3,971,925 | 7/1976 | Wenninger et al. ....   | 235/156   |

**OTHER PUBLICATIONS**

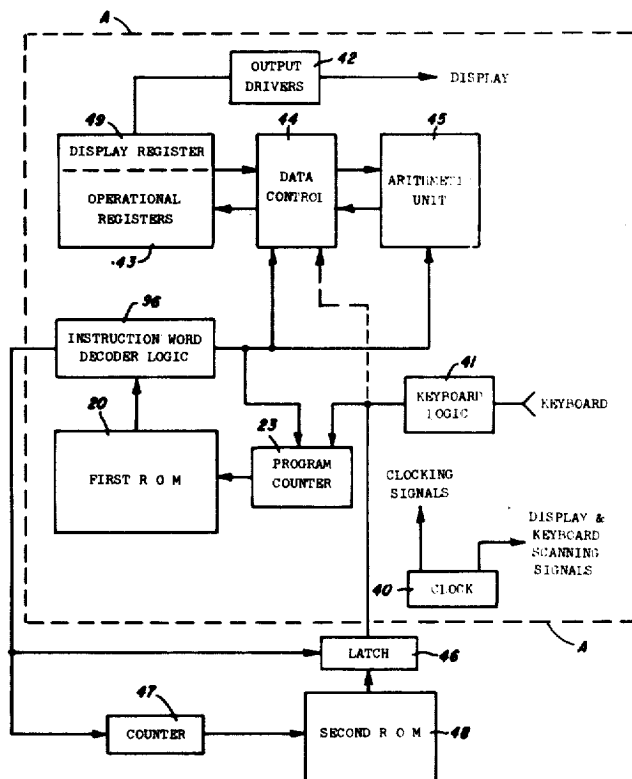
"National's Scientific Calculator HAS Long-Term

**27 Claims, 20 Drawing Figures**

[57] **ABSTRACT**

A microprocessor system with high order capabilities is provided with the two non-volatile memories which are read-only-memories (ROMs) in the disclosed embodiment. A first ROM stores the microcode for controlling the operation of the microprocessor circuits. The second ROM, which is preferably disposed in a module or cartridge, stores a plurality of program codes which are used to address the first ROM. The second ROM's module may be inserted into a receptacle for interconnecting it with the remainder of the microprocessor system. Preferably, a plurality of such second ROMs are available for selectively plugging into the microprocessor system.

Further, a particular embodiment of the microprocessor system with high order capabilities for use as an electronic calculator with high order capabilities is disclosed in great detail.



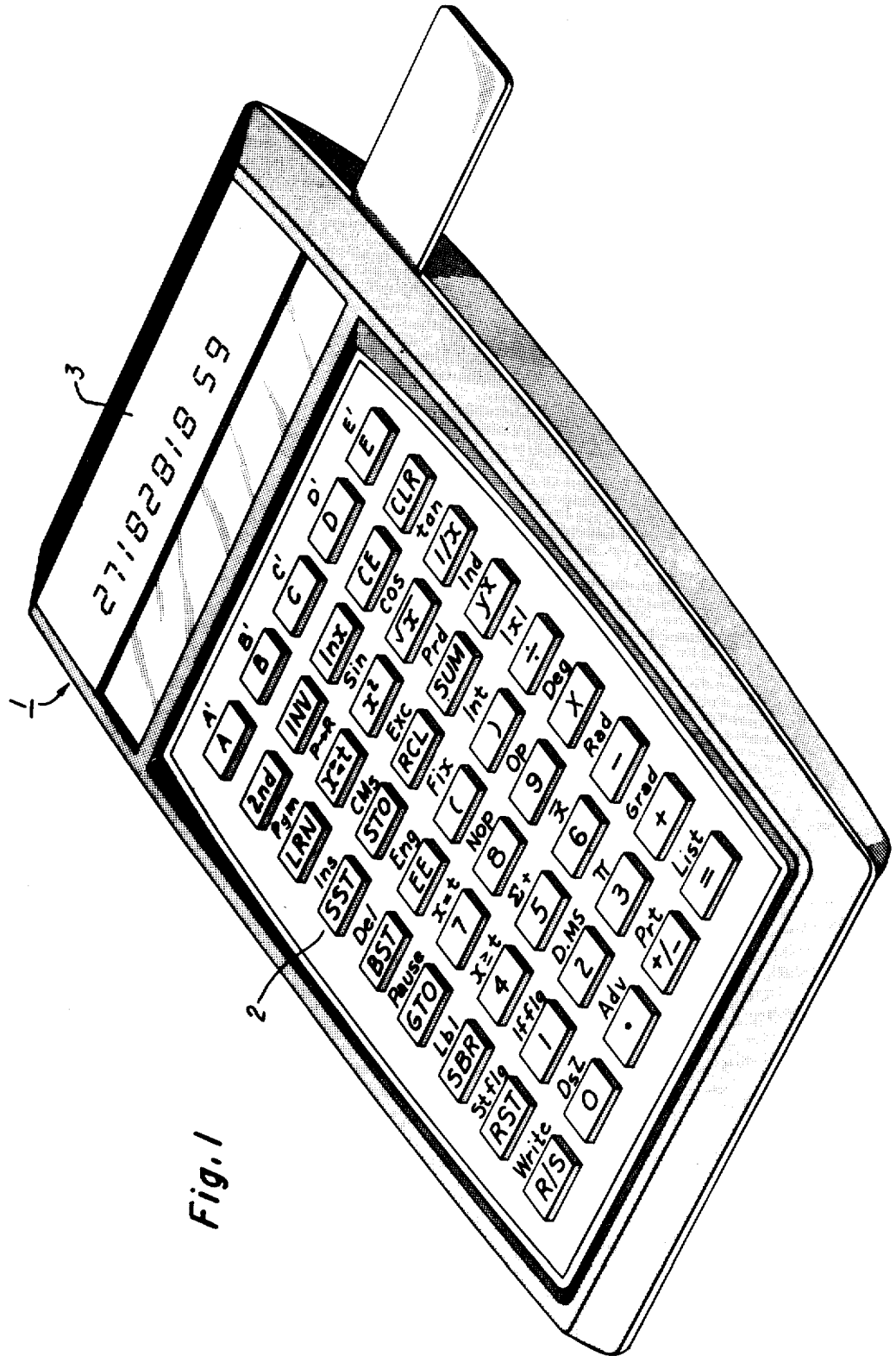


Fig. 1

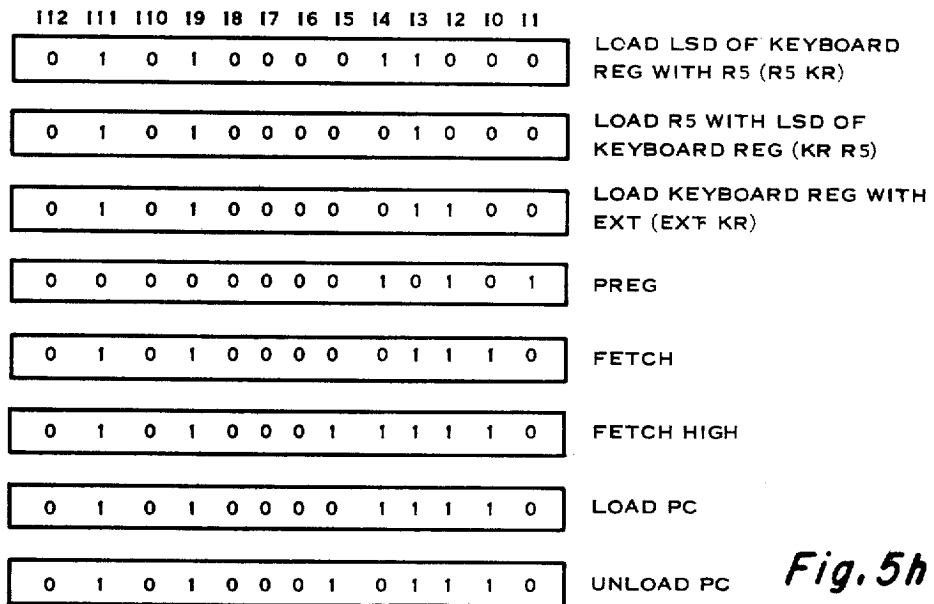


Fig. 5h

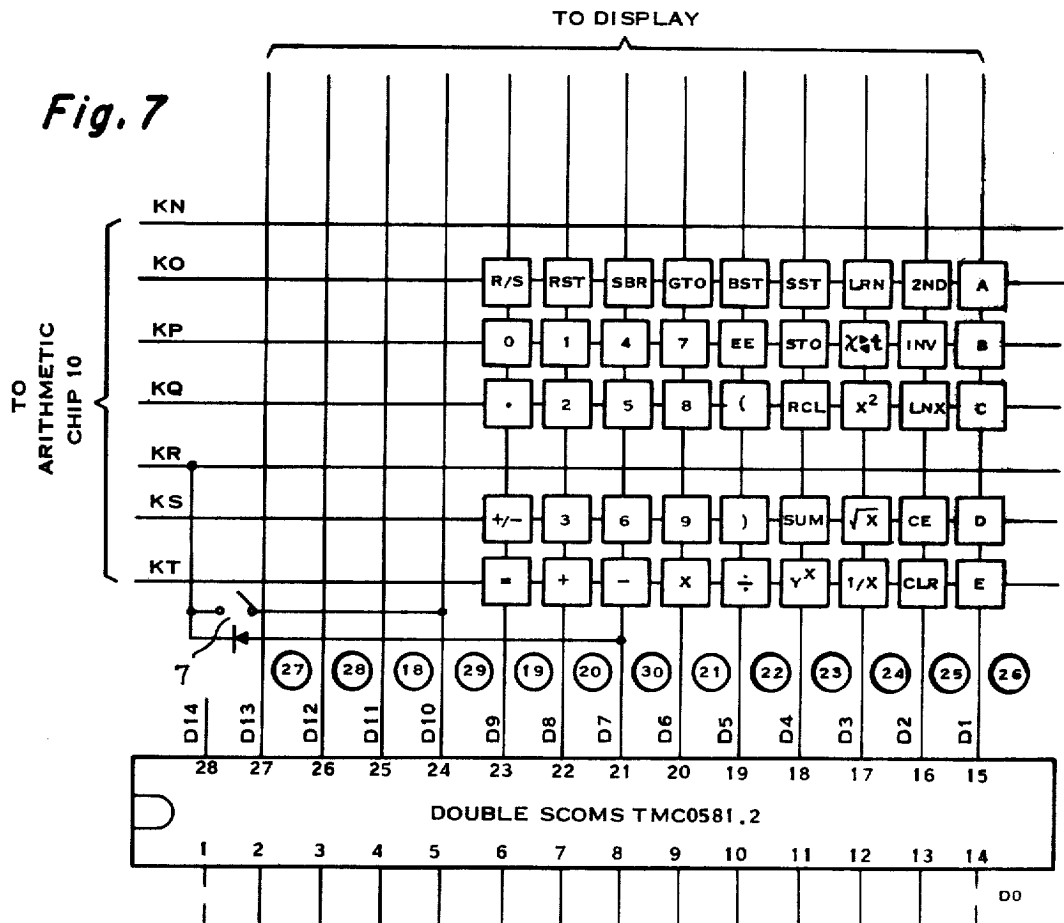
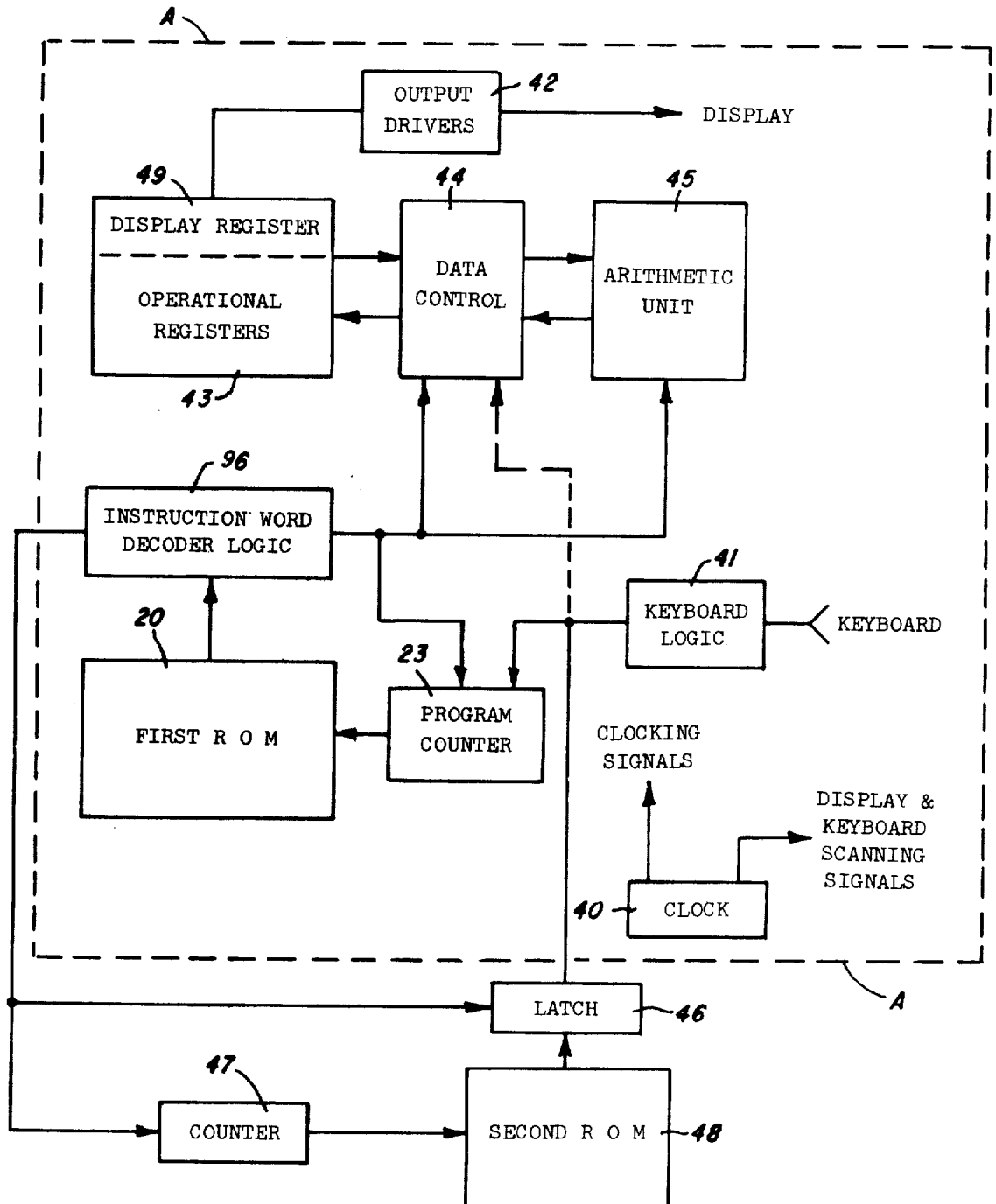
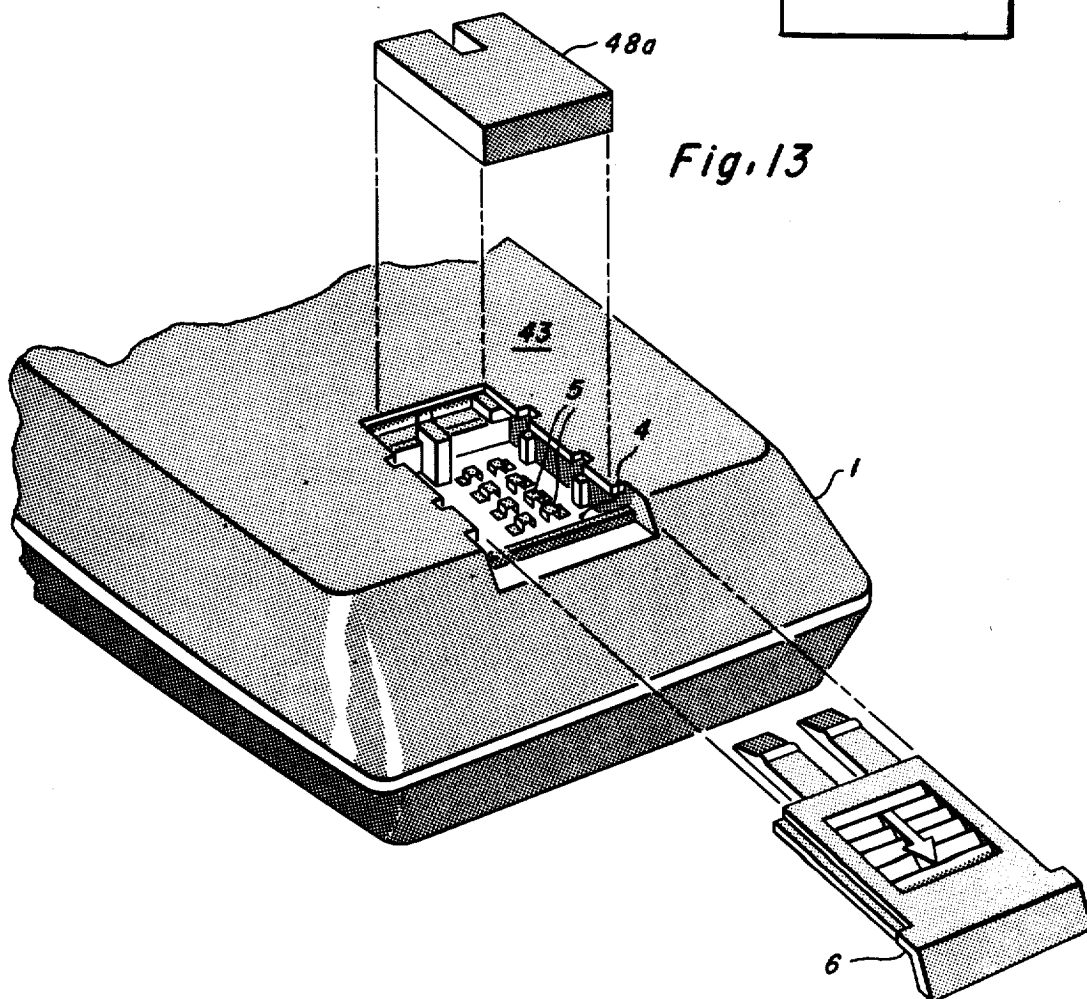
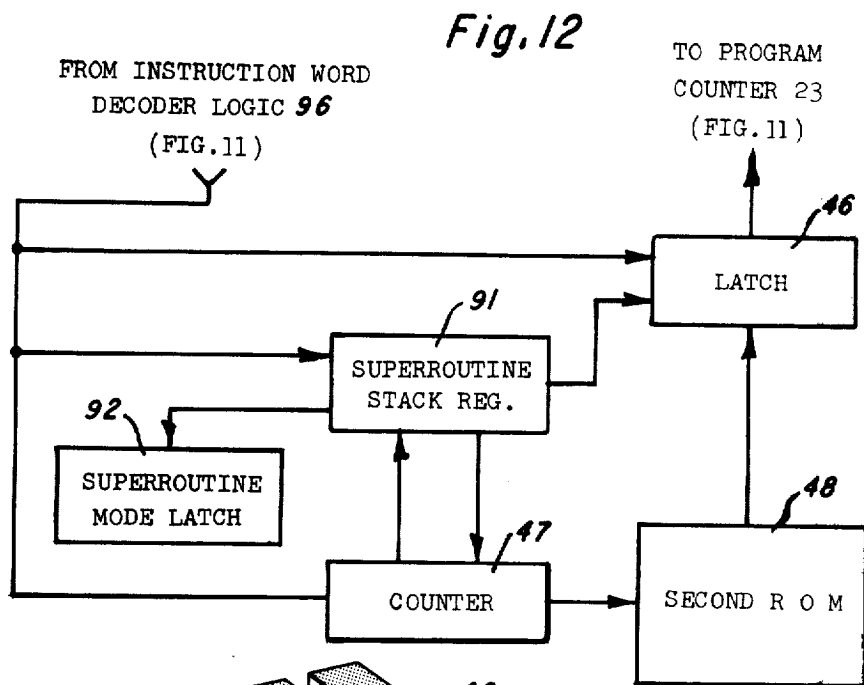


Fig. 11





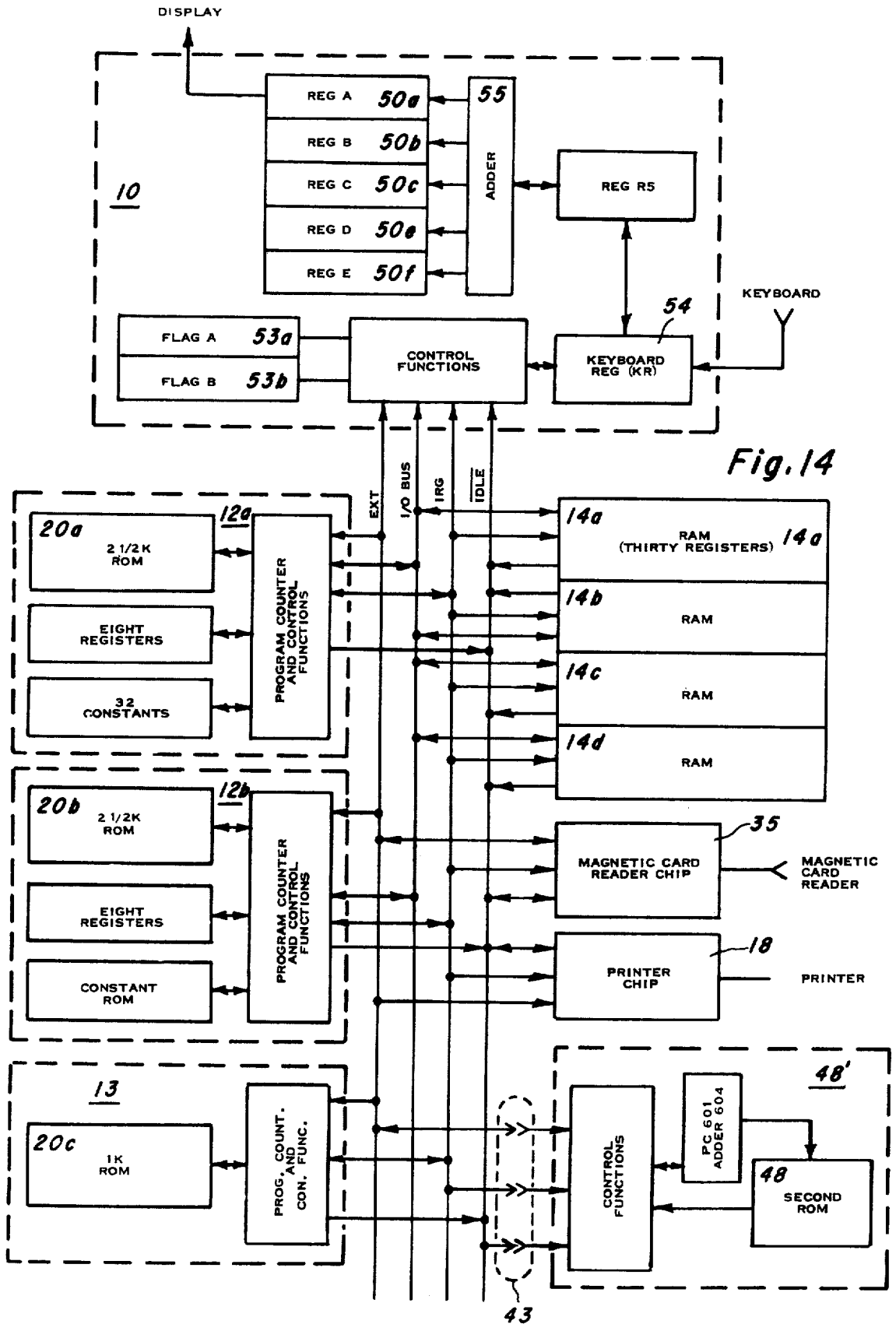


Fig. 14

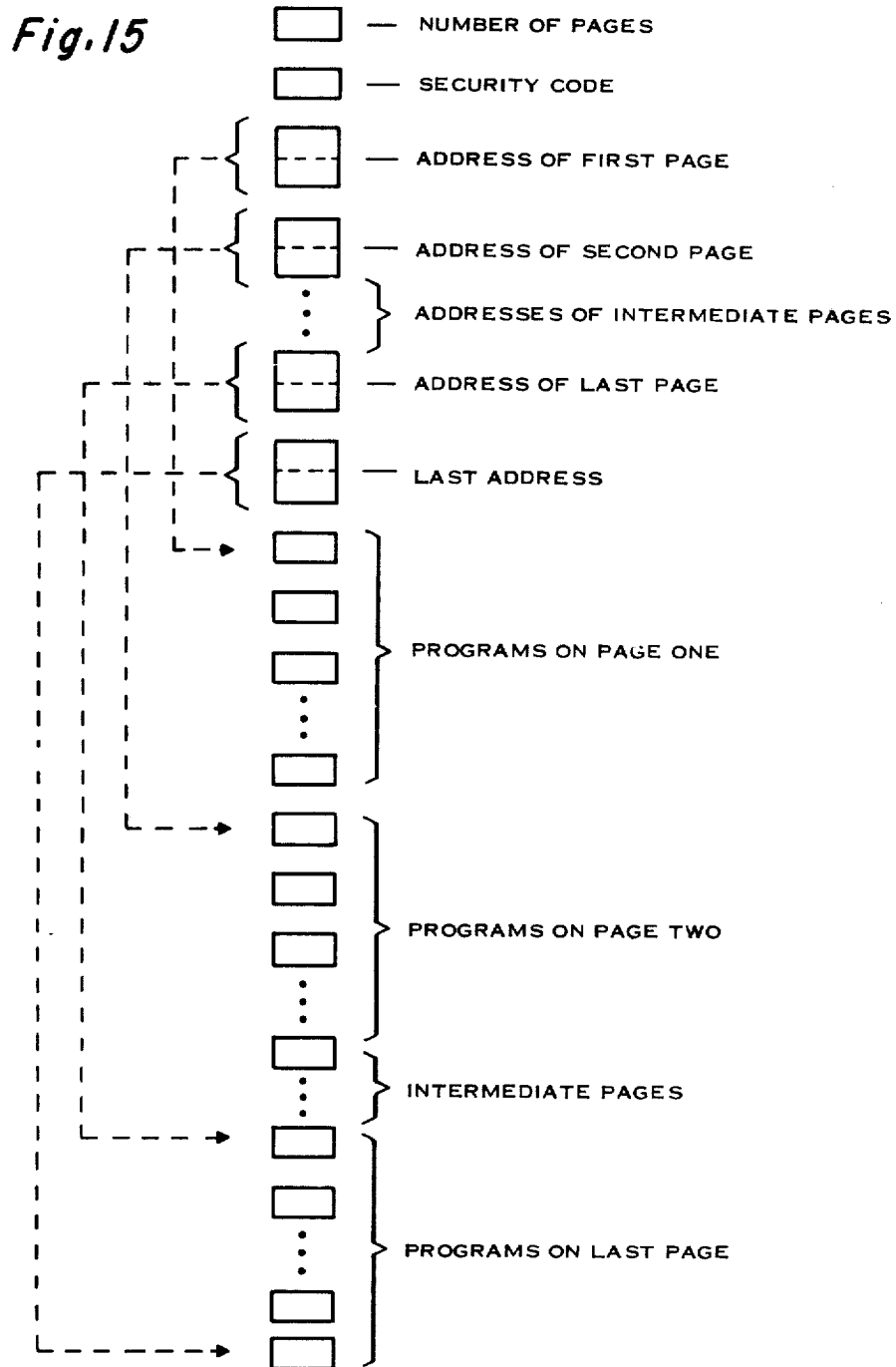
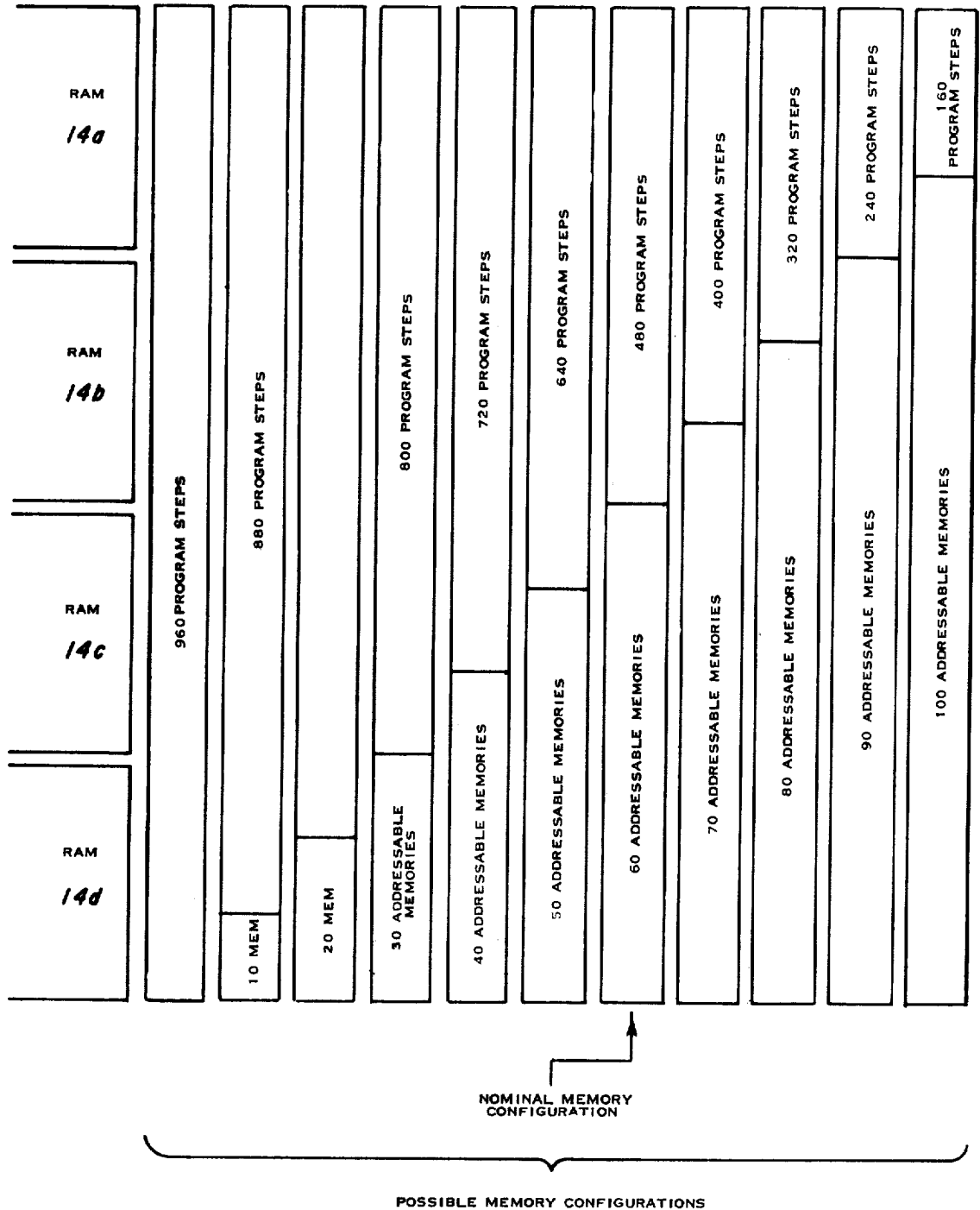


Fig. 16





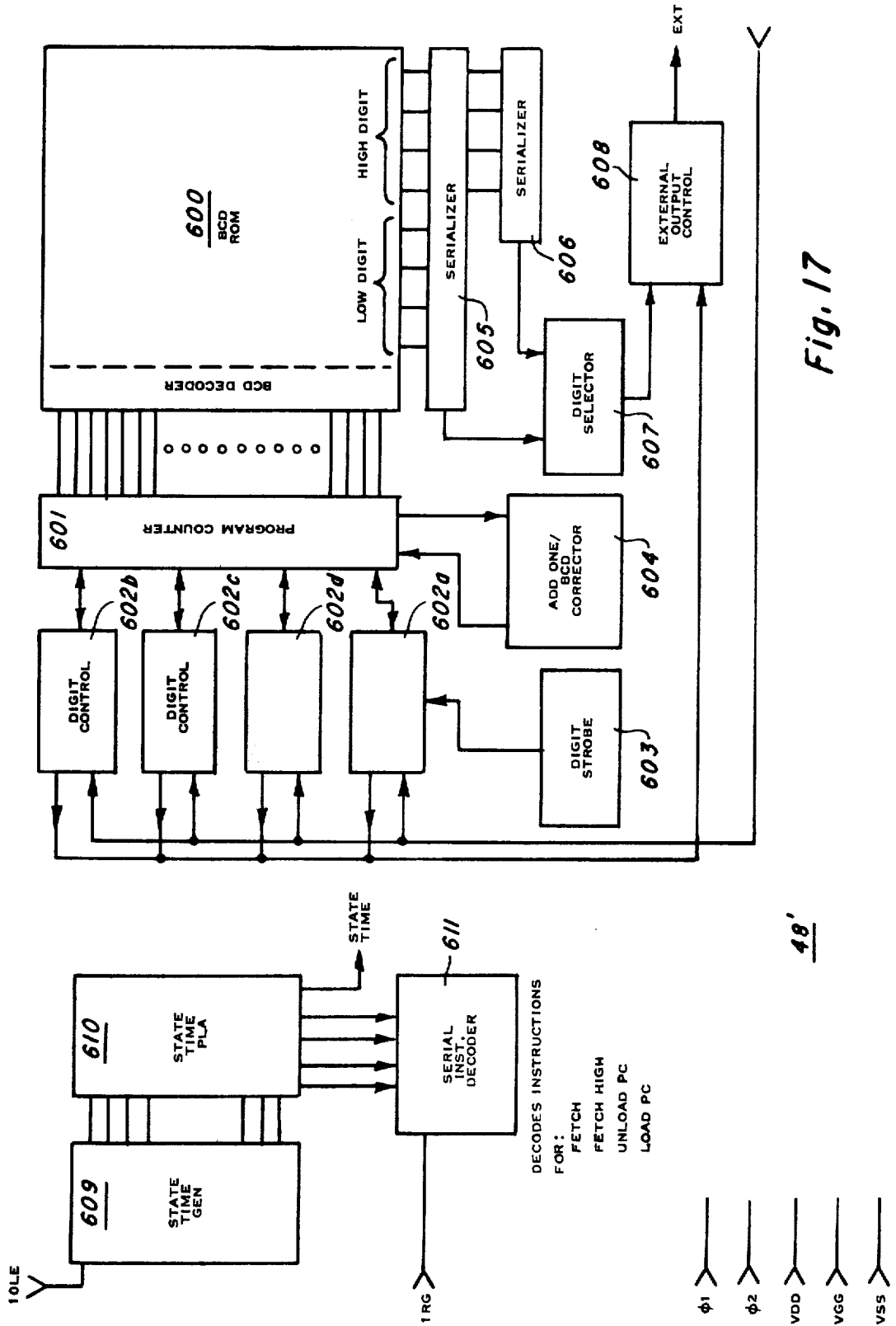


Fig. 17

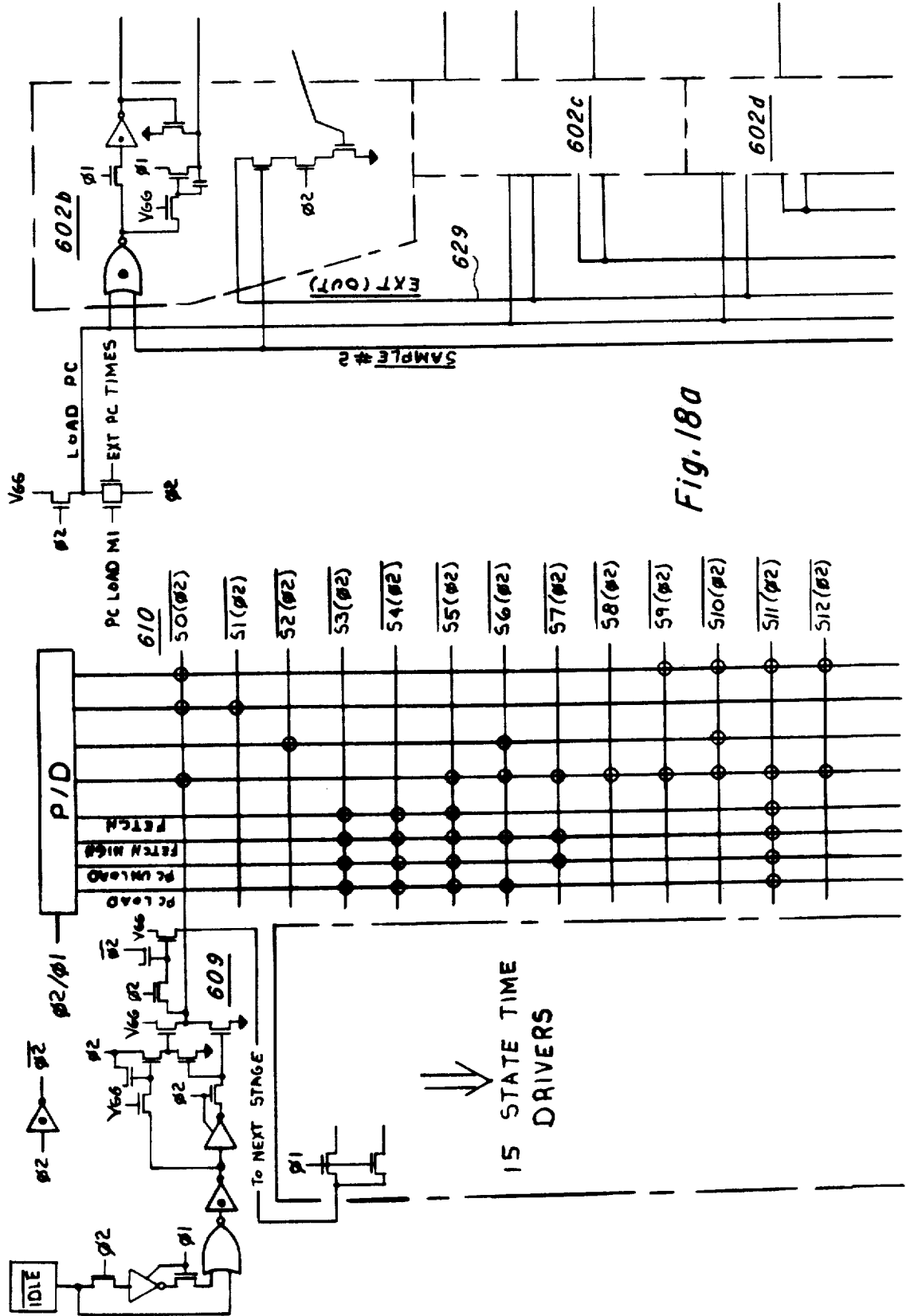
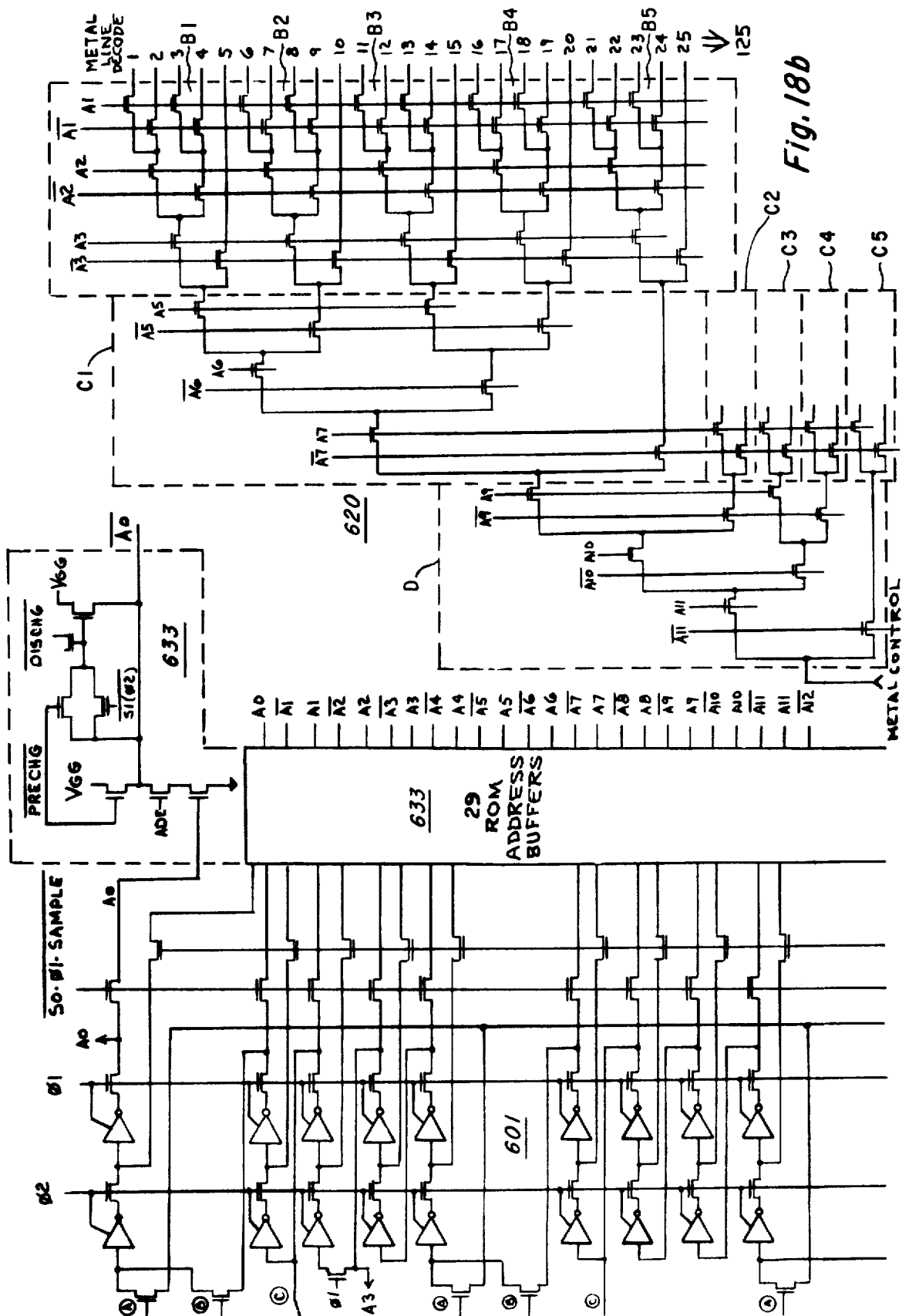
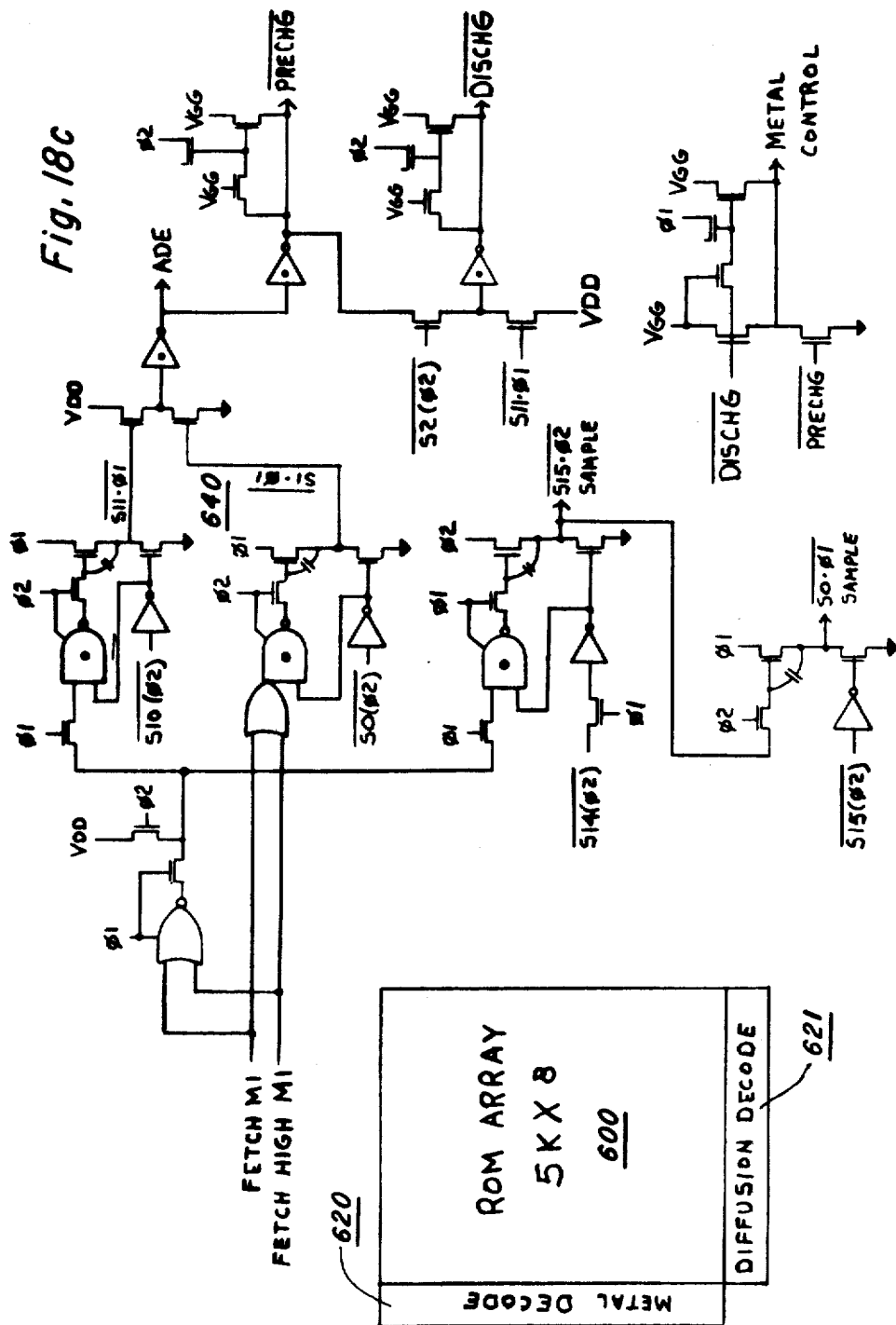


Fig. 18a





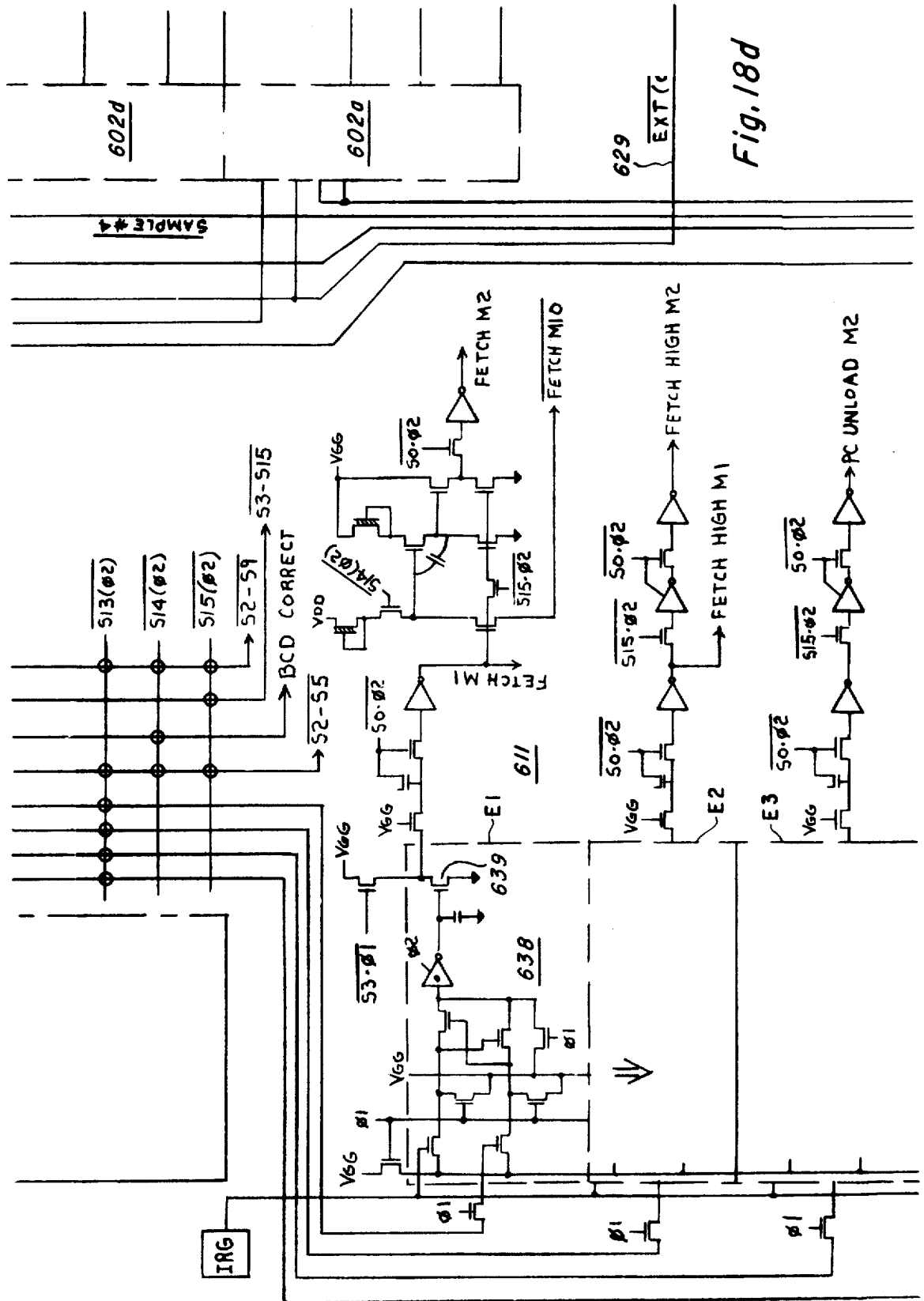
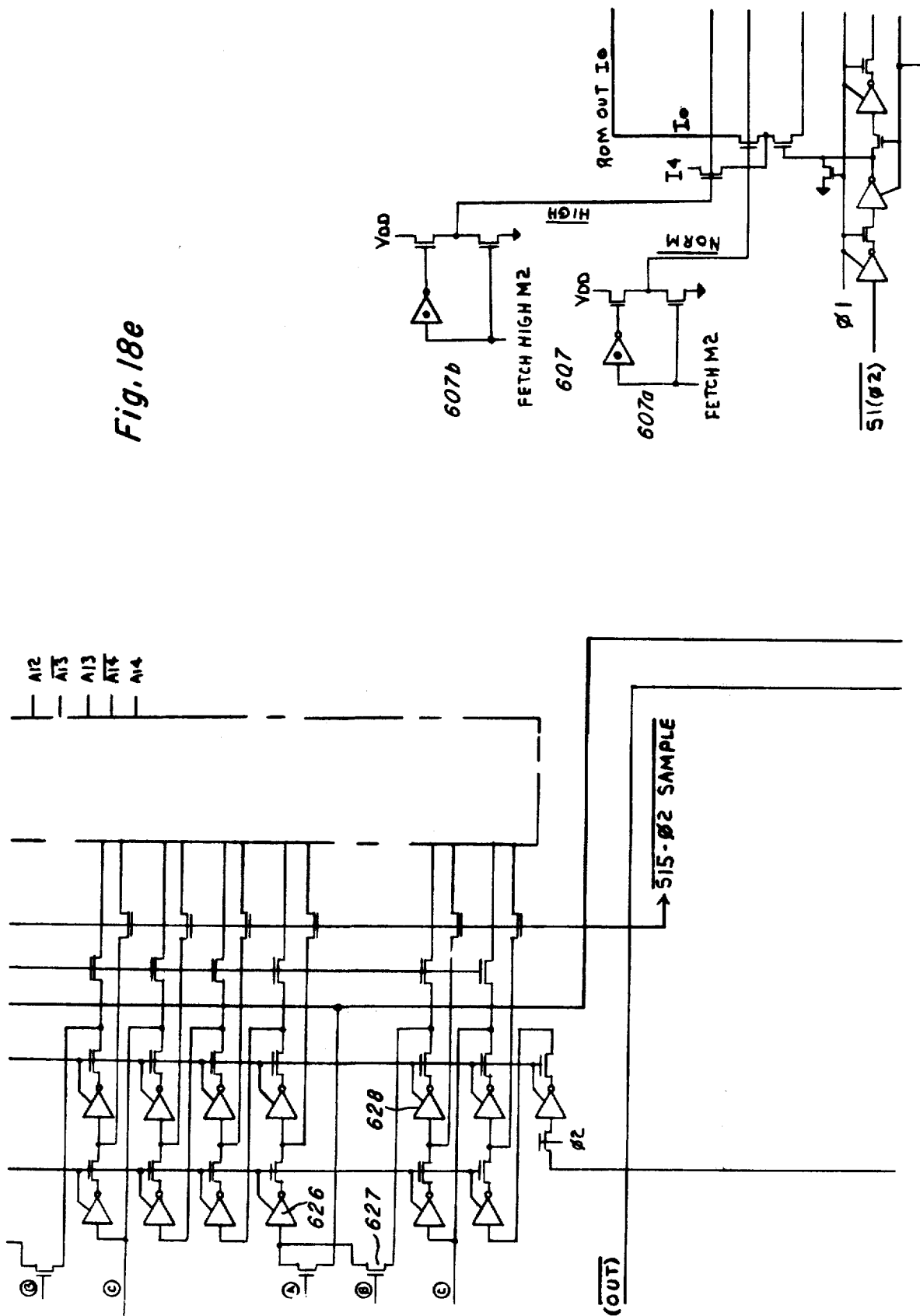


Fig. 18e



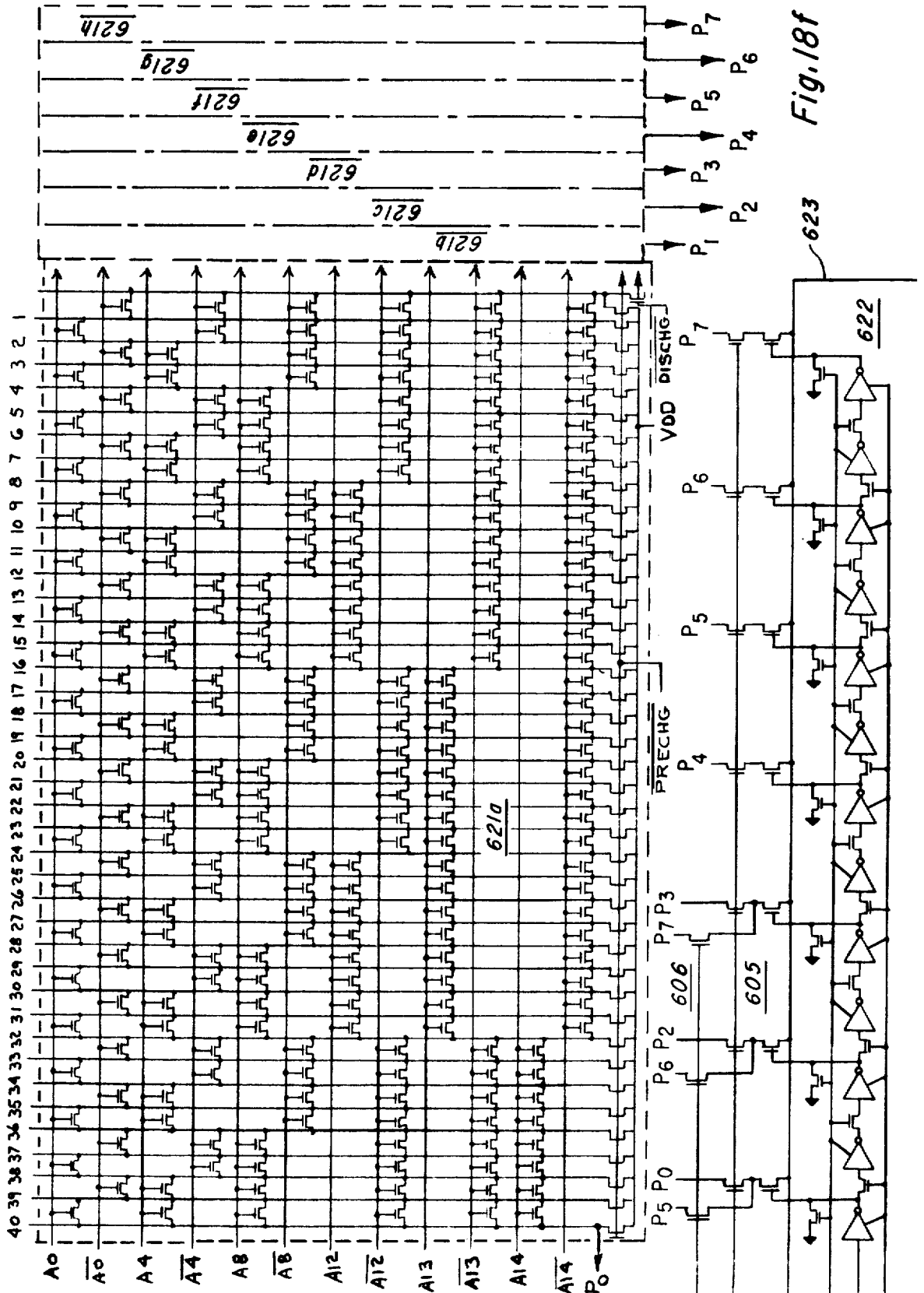
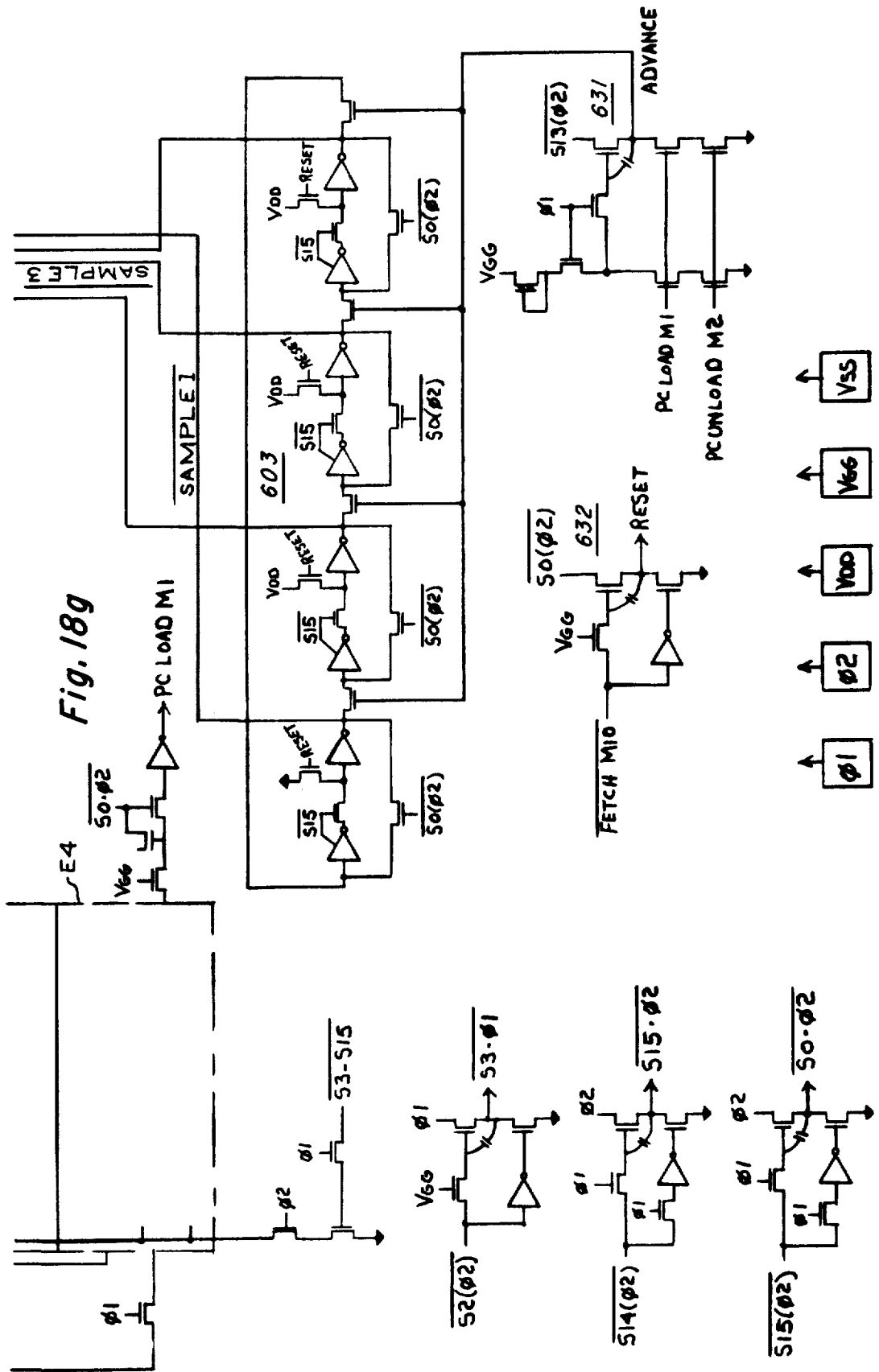
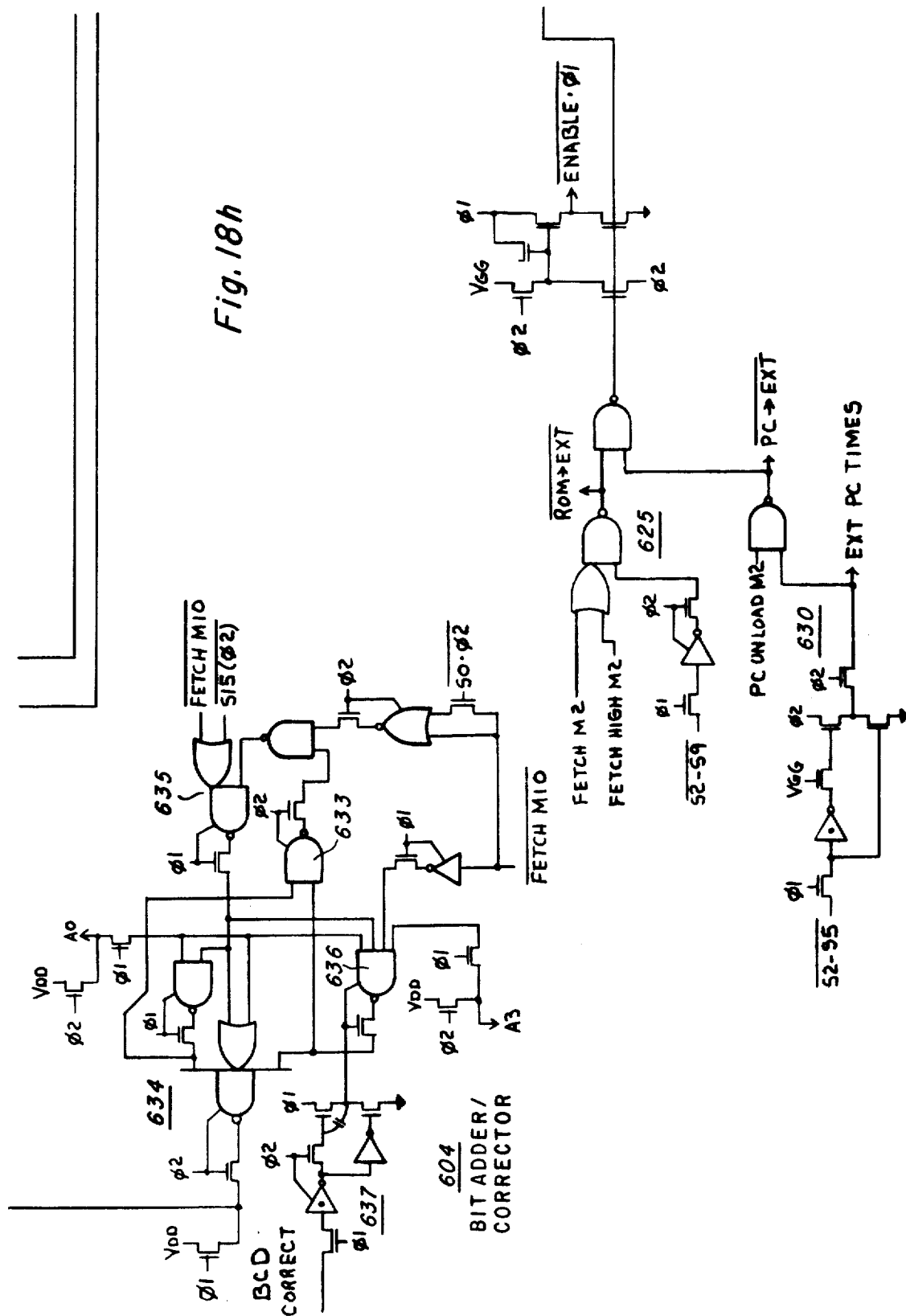


Fig. 18f







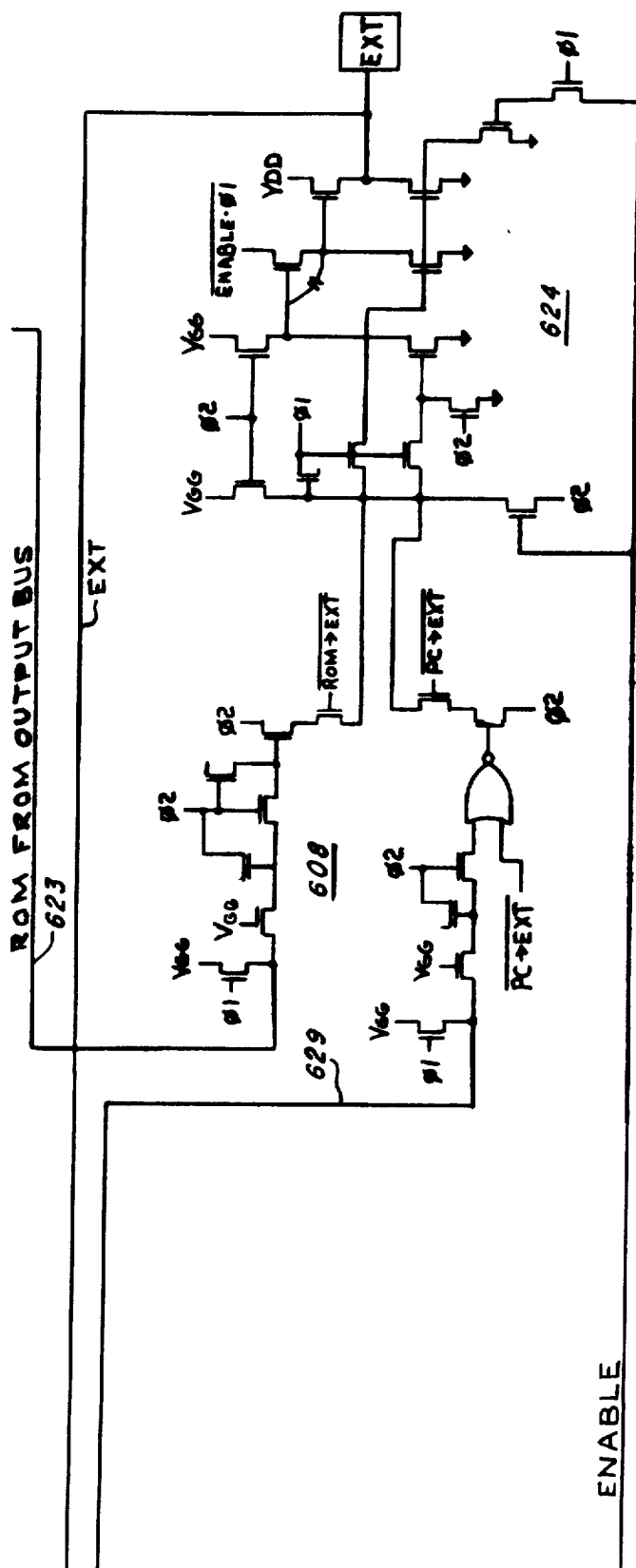
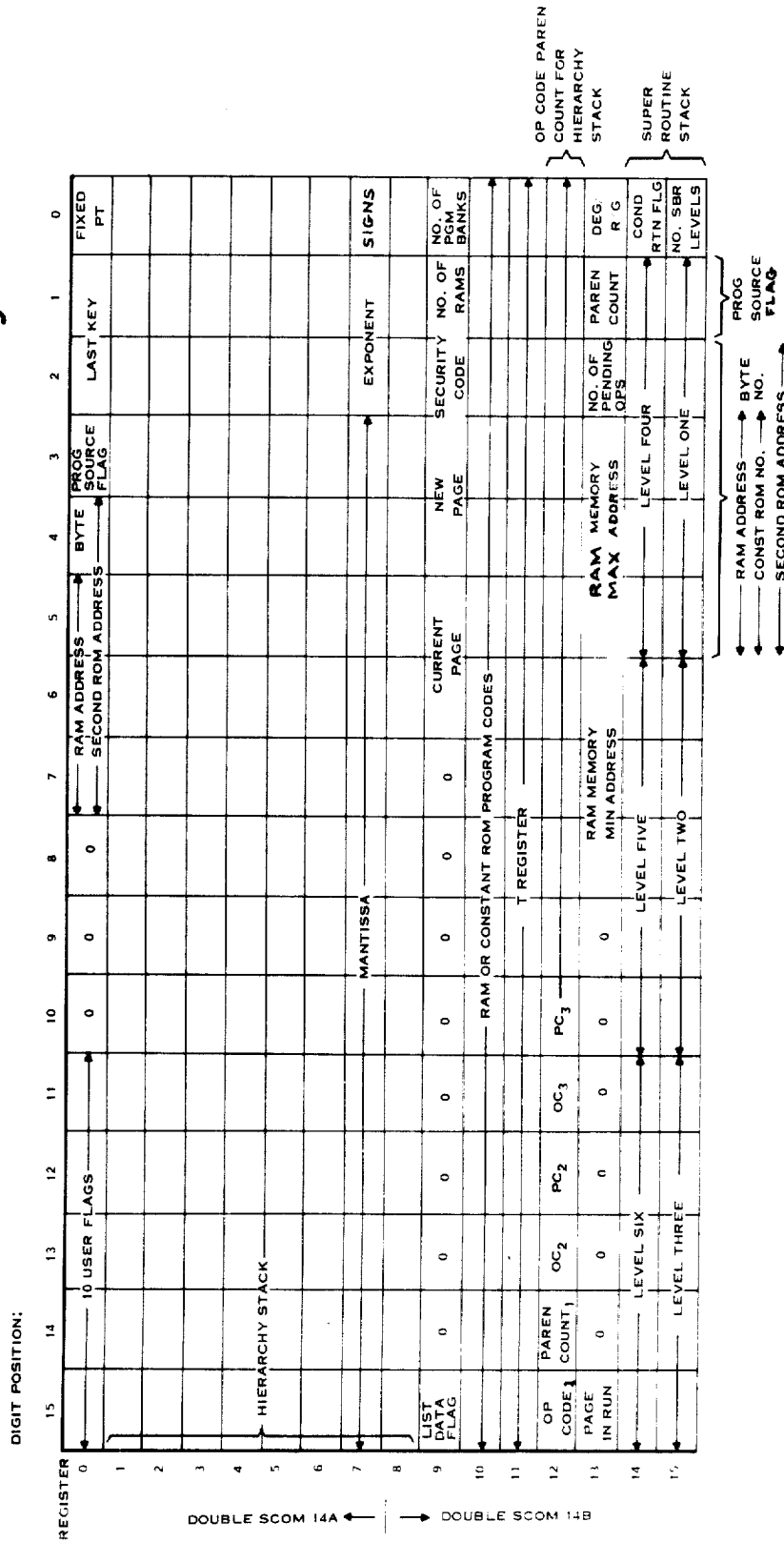


Fig. 18i

Fig. 19



## MICROPROCESSOR SYSTEM HAVING HIGH ORDER CAPABILITY

This application is a continuation-in-part of Ser. No. 714,464 filed Aug. 16, 1976.

### BACKGROUND OF THE INVENTION

This invention relates to microprocessor systems and more specifically to electronic calculators having the capability of solving higher order or complex mathematic problems. It should become evident, moreover, that my invention has utility in other applications making use of microprocessor technology, such as video games, to increase the level of sophistication of the functions performed by the microprocessor.

Electronic calculators have evolved from comparatively simple machines which add, subtract, multiply and divide the data entered into the calculator to machines which can perform sophisticated financial and mathematical operations such as, for example, changing polar coordinates to rectangular coordinates or solving compound or annuity interest problems. The calculator systems developed to date have had a single or multiple read-only-memory (ROM's) in which groups of instruction words are stored as microcode. The different groups of instruction words stored in the ROM cause the calculator's arithmetic unit, memory and display to cooperate to perform desired mathematical operations when instruction words are read out of the ROM. The ROM is addressed or controlled by a keyboard or other input means associated with the electronic calculator or microprocessor system. Thus, the depression of a key causes a selected group of instruction words to be read out of the ROM and these instruction words are provided to circuits controlling the inputting of data, the storage of data and the manipulation of data to perform the operation or function invoked by the key depressed. Thus, the instruction words generated by the ROM cause data to be entered, stored, and manipulated using, for instance, an arithmetic unit to perform such functions as adding, subtracting, multiplying dividing or performing higher order or complex mathematical operations on the data.

Each instruction word typically has a length of eight to sixteen binary bits, although microprocessors having longer or shorter instruction words are well within the state of the art. The performance of even a relatively simple operation, such as adding two floating point numbers, requires a group of many instruction words. For instance, the addition of two floating point numbers may require as many as 75 instruction words having thirteen bits each, thus absorbing 975 bits of ROM area to be able of performing such a simple operation. Similarly, the subtraction operation has a comparable set of instruction words and so on for other operations and functions. Of course, portions of such sets may be shared for certain operations.

Modern electronic calculators now perform sophisticated arithmetic and financial computations such as, for example, squaring, taking square roots, converting from polar to rectangular coordinates, computing logarithms and trigometric relationships, compounding interest, and other such computations. These computations have typically been implemented into the electronic calculator by increasing the size of the ROM to accomodate the larger number of instruction words associated with these higher order computations. While the size of com-

mercially available ROM's has increased during the past several years, the library of computational programs desired to be implemented in an electronic calculator has grown at even a faster rate. For example, it is desirable to have an electronic calculator capable of performing an entire library of computations related to electrical engineering, mechanical engineering, surveying, or the like. However, an electronic engineering library of computations could comprise, for instance, 45 computational programs or more, each of which require as many as 600 instruction words implemented in a read-only-memory. Thus an entire computational library would include for example, on the order of 27,000 instruction words of thirteen bits each which would require many conventional chips to implement the electrical engineering library in a conventional calculator. Of course, using a large number of chips can significantly increase the cost of an electronic calculator as well as making the packaging for hand-held use more difficult and unduly increasing power consumption.

In the prior art it is also known that an electronic calculator may be provided with the ability to perform higher order calculators by making the calculator programmable and storing the program in a Random Access Memory (RAM) or on a magnetic tape or card. But, in this case, the program is not permanently stored in the calculator, but must be read into the calculator's memories at least each separate time the calculator is energized; thus, such a program is not directly accessible from the calculator's keyboard.

It is an object, therefore, of this invention to improve electronic calculators and microprocessors.

It is another object of this invention to increase the number of computational programs stored in an electronic calculator without correspondingly increasing the size of the ROM(s) implemented in the electronic calculator.

It is a further object of this invention to make such computational programs directly accessible from the calculator's keyboard.

It is another object of this invention to store a large number of computational programs in the hand-held electronic calculator using a small number of chips.

It is yet another object of this invention to selectively equip an electronic calculator or microprocessor with different libraries of higher order function, which functions may be accessed from the calculator's keyboard and/or from a program stored in a programmable calculator.

It is still another object of this invention that the particular library with which a calculator is equipped may be changed by the end user thereof.

The aforementioned objects are satisfied as is now described. Generally, and in accordance with the preferred embodiment of the invention, an electronic calculator is equipped with first and second ROM's. The first ROM stores a plurality of groups of instruction words, each group effective for controlling an arithmetic unit to perform basic arithmetic operations such as adding, subtracting, multiplying and dividing, taking square roots, forming logarithmic and trigonometric operations and so forth in response to the operation of the first set of keys on a calculator keyboard. Each of these groups of these instruction words contains on the order of 75 to 200 instruction words. Thus, the first ROM is used as a main ROM as in a conventional calculator microprocessor. The second ROM stores a plurality of sets of program codes. Each set of program codes

are capable of performing a higher order mathematical program and are read out of the second ROM in response to operation of a second set of keys on the calculator keyboard. Each program code (which comprises eight binary bits in the embodiment disclosed) is effective for addressing a group of instruction words stored in the first ROM in much the same manner as depression of a key in the first set of keys. Thus, a set of program codes may mimic the depression of a plurality of keys in the first set of keys. Therefore, each higher order mathematical program is preferably a series of the basic arithmetic operations stored in the first ROM combined with operations for entry of data and/or constants. The second ROM reads out program codes which serve to address the groups of instruction words stored in the first ROM. By using the second ROM to store such higher order calculational programs, a second set of keys can be used to input commands triggering a long chain of basic arithmetic operations, including data entry operations, without the chance of human error and at a much greater speed than a human operator. Thus, the second ROM, in the preferred embodiment, stores sets of program codes, each program code effective for addressing the first ROM in much the same manner as a single depression of a key in the first set of keys; therefore, a set of such program codes may be advantageously utilized for a large number of higher order calculational programs in an electronic calculator. Since the second ROM must only store on the order of eight bits, or so, to select or address an entire group of instruction words, it should be evident to one trained in the art by utilizing the second ROM herein disclosed that great economies can be effected in total ROM area, when compared with prior art techniques. While I have referred to first and second sets of keys on the calculator keyboard, it is well known that a single physical key may be used to perform several functions and therefore the keys referred to in the first and second sets may be physically the same keys.

In a further aspect, the second ROM is preferably provided by a chip or chips which may be temporarily plugged into the calculator or microprocessor system by the end user thereof. Preferably, the calculator or microprocessor system is operated with one or more such second ROM chips, which chips are selected from a group of chips for operation in the calculator or microprocessor system by the end user thereof, according to the end user's particular needs at any given time. In the embodiment disclosed, a major portion of the second ROM is provided by a plugged-in chip and minor portion is provided by a permanently wired-in chip. Thus, in the embodiment disclosed, the end user may select which higher order functions are performable according to which particular second ROM chip is plugged into the calculator while a few high order functions stored in the permanently wired chip is inherently a part of the calculator system disclosed. The second ROM is disposed in a module for ease of handling by the end user of the calculator or microprocessor.

While a calculator system is disclosed herein in detail, it should be evident to those skilled in the art that my invention may also be used in applications other than calculators where a microprocessor with high order capability may be advantageously utilized.

#### BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as further objects and advantages thereof, will be best understood by reference to the following detailed description of an illustrative embodiment, when read in conjunction with the drawings, wherein:

FIG. 1 is a pictorial view of an electronic portable calculator of the type which may embody the invention;

FIG. 2 is a simplified block diagram of a multi-chip calculator system which may be utilized in practicing the present invention;

FIGS. 3a-3b are detailed block diagrams of the arithmetic chip featured in FIG. 2;

FIG. 4 is a detailed block diagram of the SCOM chip featured in FIG. 2;

FIGS. 5a-5e depict in representative form the instruction words decoded by the arithmetic and SCOM chips;

FIG. 5f depicts the origination of the EXT signal;

FIG. 5g depicts the first ROM address as stored in the address register;

FIG. 5h depicts the instruction words decoded on the second ROM chip and selected instruction words decoded on the arithmetic chip, but which may be conveniently employed in connection with the utilization of second ROM chip;

FIGS. 6a-6b are timing diagrams showing the timing of various parts of the multi-chip system;

FIG. 7 is a representation of the keyboard input matrix;

FIGS. 8a-8d are a composite schematic diagram of the arithmetic chip of FIG. 2;

FIGS. 9a-9e are a composite schematic diagram of the SCOM chip of FIG. 2;

FIGS. 10a-10r are schematics of certain circuits used in FIGS. 8a-8d and 9a-9e;

FIG. 11 is a block diagram of a modern electronic calculator equipped with one embodiment of the invention;

FIG. 12 is a block diagram of another embodiment of the invention which may be utilized with a modern electronic calculator of the type depicted in FIG. 11;

FIG. 13 is a pictorial view of an electronic calculator having an opening for removeably receiving a packaged second second ROM chip;

FIG. 14 is a simplified block diagram of a multi-chip calculator system utilizing the present invention;

FIG. 15 is a function diagram of the logical organization of data stored in the second ROM;

FIG. 16 depicts the variable boundary between data and program steps in the calculator's memory;

FIG. 17 is a block diagram of the second ROM chip;

FIGS. 18a-18i form a composite schematic diagram of the second ROM chip; and

FIG. 19 is a representation of data stored in the memory registers on the double SCOM chips.

#### LOCATION OF THE DRAWINGS

FIGS. 1, 5h, 7, and 11-19 accompany this patent. FIGS. 2, 3a-3b, 4, 5a-5g, 6a-6b, 8a-8d, 9a-9e and 10a-10r are hereby incorporated by reference from U.S. Pat. No. 3,900,722, entitled "Multi-Chip Calculator System Having Cycle and Subcycle Timing Generators", which issued on Aug. 19, 1975 to Michael J.

Cochran and Charles P. Grant, Jr. and which is assigned to the assignee of this invention.

### CONCEPTUAL DESCRIPTION

Referring to FIG. 1, an electronic portable calculator of the type which may employ features of this invention is shown in pictorial form. The calculator 1 comprises the keyboard 2 and the display 3. The display 3, in one embodiment, consists of twelve digits or characters, each provided by an array of light emitting diodes or characters, such provided by an array of light emitting diodes, a liquid crystal display, gas discharge tube or other display means. The display is preferably implemented to having eight mantissa digits, two exponent digits, and two character places for negative signs, etc., (one for the mantissa and one for the exponent), thereby permitting outputting the data in scientific notation for instance. Of course, the type of display and the number of digits displayed is a design choice. Ordinarily, the display would be of the seven segment or eight segment variety, with provisions for indicating a decimal point for each digit. The display 2 includes a number of keys (0-9), a decimal point key, the conventional plus (+), minus (-), multiply ( $\times$ ), divide ( $\div$ ), and equal (=) keys. Further the keyboard preferably includes keys for exponentiation ( $Y^x$  and inverse  $Y^x$ ) and trigonometric relationships (Sine X, Cosine X, and Tangent X). The calculator is further provided with OP Code Keys for performing special functions such as slope, intercept, plotting operations, alphanumeric operations and the like. Further, the calculator may be provided with keys for storing (STO) and recalling (RCL) data from memory, for clearing the calculator (CLR) and for clearing the last entry (CE). The keys used to access higher order functions will be described subsequently.

In FIG. 11, there is shown in block diagram form, the basic elements of a modern electronic calculator implemented on one or more semiconductor chips. It is to be understood that the block diagram of FIG. 11 is not intended to represent the block diagram of a detailed representation of electronic calculators, but is merely intended to indicate how the additional elements of an electronic calculator system having higher order capability are incorporated into a typical electronic calculator. Subsequently, it will be explained in detail how my invention may be practiced with the multi-chip calculator system depicted in FIGS. 2-19. The calculator of FIG. 11, is shown with a clock 40 which provides clocking signals for transferring data throughout the electronic calculator and provides scanning signals for scanning the display 3 and keyboard 2 or other data entry means. The inputs for the keyboard 2 are provided to keyboard logic 41 which provides an address in response to the depression of a particular key to program counter 23. It should be evident to one skilled in the art that keyboard logic 41, as well as other logic circuitry, may be implemented in the calculator as the elements described or may be implemented as a part of read only memory 20 and instruction word decoder logic 96.

The address received from keyboard logic 41 is inserted into program counter 23 and is utilized in addressing the First Read-Only-Memory (ROM) 20. First ROM 20 contains the microcode for performing basic arithmetic operations and outputs an instruction word in response to the address contained in program counter 23. Program counter typically includes an add-one circuit for incrementing the address in program counter

23. Thus, program counter 23 causes a group of instruction words to be read out of First ROM 20 in response to the incrementing of program counter 23, each instruction word being read out during an instruction cycle. The group of instruction words read out of First ROM 20 corresponds to the address received from keyboard logic 41.

The instruction words read out of First ROM 20 are decoded by instruction word decoder logic 96 to provide instruction commands to program counter 23, arithmetic unit 45 and data control 44. The instruction commands provide to program counter 23 enable branches to be executed by inserting a new address into program counter 23 in response to a branch instruction command stored in First ROM 20. Instruction commands provided to data control 44 and arithmetic unit 45 control the manipulation of numeric data in the calculator. Instruction word decoder 96 is also interconnected with a counter 47 and a latch 46 in my electronic calculator system having higher order math capability.

Data control 44 is interconnected with display register 49, operational registers 43 and with the arithmetic unit 45. Display register 49 stores the number displayed by the display 3 and has associated therewith a plurality of operational registers 43 which are used in conjunction with arithmetic unit 45 to perform arithmetic operations in response to particular instruction commands. Output drivers 42, interconnect display register 49 with a display 3 for decoding the electrical signal, stored in display register 49 and for driving display 3. Data control 44 comprises a series of selector gates for interconnecting the appropriate operational registers 43 and display register 49 with the arithmetic unit 45, with portions of instruction words, (if need be), or with logic signals from keyboard 2 (if need be).

Numeric data is inputted into display register 49 from keyboard 2 either by a data path from keyboard logic 41 via data control 44 under the control of appropriate instruction commands or by inputting selected portions of an appropriate instruction word in response to selected instruction commands. The electronic calculator system hereinbefore described, that being the portion shown within the reference a dashed line in FIG. 11, basically corresponds to the type of electronic calculators known in the prior art. Exemplary of the prior art calculators systems is the calculator system depicted in FIGS. 2-10.

Also in FIG. 11, there is shown a counter 47 and a latch 46 which is responsive to outputs from instruction word decoder logic 96. The counter 47 has an output for addressing a Second ROM 48. Second ROM 48 outputs a program code in response to the inputted address, the program codes being outputted via latch 46 to program counter 23. When keyboard logic 41 decodes keyboard outputs indicating that a higher order math calculation is to be executed, the higher order math calculation being preferably a series of basis arithmetic functions and operations of the type implemented in first ROM 20, keyboard 41 preferably input an address into program counter 23 which causes First ROM 20 to branch to a location therein for calling a program from Second ROM 48. When a program is called from Second ROM 48, instruction word decoder logic 96 first sets latch 46 to permit the program codes outputted by Second ROM 48 to be inputted into program counter 23. The program codes outputted by Second ROM 48 effectively transmit an address into program counter 23 for addressing First ROM 20. The first such

code preferably causes the First ROM 20 to branch to a location for performing the first basic arithmetic operation or function required by the Second ROM 48 program. The program codes may take the same logical format, for instance, as the output from keyboard logic 41. When calling a program from Second ROM 48, instruction word decoder logic 96 also transmits an address into counter 47, the address being the first location in Second ROM 48 of the called program. It should be evident, moreover, that counter 47 could be loaded with an address directly from keyboard logic 41 in lieu of from instruction decoder logic 96, this being essentially a design choice.

After the first program code is read out of Second ROM 48 via latch 46 and loaded into program counter 23 then First ROM 20 cycles through a group of instruction words to accomplish the indicated basic arithmetic operation or function. Of course, the number of instruction cycles required to accomplish the indicated operation or function depends on, for instance, a number of instructions contained for that basic operation or function in First ROM 20. As is well known, those operations or functions which are accessible via keyboard logic 41 from keyboard 2, usually contain instruction words for causing the display to be enabled at the end of the function or operation addressed in First ROM 20. Since, however, another program code is to be read from Second ROM 48 and inserted into program 23 upon accomplishment of the indicated function or operation, counter 47 includes an add-one circuit which is responsive to, for instance, a display command or other such commands located near or at the end of a group of instruction words in First ROM 20 for accomplishing a basic arithmetic operation or a function. When the display command or other such command is decoded by instruction word decoder logic 96, the add-one circuit in counter 47 increments and causes Second ROM 48 to read out the next program code of the called program via the set latch 46 to program counter 23, which in turn causes First ROM 20 to cycle through another group of instructions to accomplish the function or operation indicated by the outputted program code. Again, towards the end of this next basic arithmetic function or operation, a display code or other such code will be decoded in instruction word decoder logic 96 causing the add-one circuit in counter 47 to increment counter 47, the cycle repeating itself.

The advantages of Second ROM 48 and associated counter 47 and latch 45 should be evident to one trained in the art. This system permits equipping an electronic calculator with the capability of performing higher order calculational programs: for instance, changing polar coordinates to rectangular coordinates, doing financial calculations or solving complex engineering equations using significantly less total ROM area than would be required if such programs were implemented only in First ROM 20. Additionally, it should be evident that while the foregoing discussion has suggested that a program code read from Second ROM 48 mimics keyboard logic outputs from keyboard logic 41, the program codes read from Second ROM 48 could, in lieu thereof or in addition thereto, have codes which do not mimic the outputs from keyboard logic 41, but rather, for instance, would cause the program counter 23 to branch to locations in First ROM 20 which are not directly accessible from the keyboard. Thus an output from Second ROM 48 may cause program counter 23 to branch to a location in First ROM 20 which could not

be accessed directly from the keyboard 2 via keyboard logic 41. One purpose for such a program code would be a program code in a called program to indicate that the end of the program had been reached. This program code, which I shall refer to as the "return" program code, preferably causes program counter 23 to branch to an address location in First ROM 20 which would contain a group of instructions for resetting latch 46 and for displaying the contents of display register 49. The display instruction preferably follows the reset latch instruction, so that when the display command causes counter 47 to increment (if so used), no branching will occur in response thereto at program counter 23. Also latch 46 inhibits outputs from Second ROM 48 from being inserted into program counter 23 whenever a display instruction or other such instruction is decoded by instruction word decoder logic 96 incrementing the add-one circuit in counter 47 when the calculator has not called a program from Second ROM 48. Referring now to FIG. 12, there is shown a partial block diagram of a second embodiment of my calculator system having higher math capability. The latch 46 and counter 47 are interconnected with program counter 23 and instruction word decoder 96 as done in the embodiment shown in FIG. 11. In fact, this embodiment is similar to the embodiment in FIG. 11, except that a superroutine stack register 91 and an associated superroutine latch 92 have been interconnected with counter 23; stack 91 is responsive to outputs from instruction word decoder logic 96. Thus, the program codes outputted from Second ROM 48 are passed to program counter 23 via latch 46; counter 47 is used to address Second ROM 18 and is responsive to instruction word decoder logic 96 for inserting an initial address therein and for incrementing that address in response to decoded display commands, for instance. The superroutine stack 91 functions to either receive an address from counter 47 or to output an address to counter 47, both functions being in response to outputs from instruction word decoder logic 96. Superroutine stack 91 is a multi-level stack and functions in normal last-in-first-out mode. Superroutine stack 91 may be advantageously utilized in calculator systems with higher math capability so that the program codes stored in Second ROM 48, in addition to prescribing addresses for performing basic arithmetic operations and functions according to microcode stored in First ROM 20, may also use First ROM 20 for addressing Second ROM 48 itself. The advantages of this superroutine stack 91 may be best seen by example. For instance, Second ROM 48 may be implemented with program codes to perform the factorial function, and during the calculation of statistical programs, such as combinations and permutations, it is often advantageous to be able to use the factorial function. If a factorial function and the statistical combination function are both implemented in Second ROM 48, then the combination function program may call the factorial function, if a superroutine stack 91 is utilized. The point of exit from the combination function program must be stored so that the program can return thereafter accomplishing the factorial function. Thus, the address in counter 47 to which the program must return in Second ROM 48 after accomplishing the factorial function is stored in superroutine stack 91 because a new set of addresses will be loaded into counter 47 when the factorial function is executed. Whenever an address has been stored in superroutine stack 91, superroutine latch 92 is set. Further, as aforementioned, the factorial program will preferably have a

"return" program code loaded in Second ROM 48 at the end thereof. This return code normally causes latch 46 to be reset. However, the return code is inhibited from setting latch 46 when superroutine latch 92 has been set. Latch 46 is not reset at this time because the program codes being read from the Second ROM 48 must continue to be inserted in program counter 23 to carry out the main program, eg, the combination function program in the aforementioned example. Although the return code is not used to reset latch 16 if superroutine latch 92 is set, the return code is used to "pop" the address in the stack back into counter 47. Since stack 91 is a multi-level stack, several levels of "superroutines" may be utilized.

Preferably, the second ROM 48 is implemented by a ROM chip which is provided with a package permitting it to be plugged into the aforementioned electronic calculator. The second ROM chip is preferably packaged in a form to facilitate handling by the end user of the calculator to permit easy installation of that chip into the electronic calculator. Preferably, the electronic calculator system of this invention receives, at any given time, one second ROM chip, but the operator thereof selects which particular second ROM chip is plugged into the calculator system. The operator ROM chips each programmed to perform different types of high order functions. For instance, one or a plurality of second ROM chips may be provided for performing statistical problems while another ROM chip or chips is provided for performing surveying problems; still yet another second ROM chip or chips may be provided for performing aviation or navigational problems, for instance. Thus the end user of the calculator can configure a basic calculator to perform many different types of high order functions depending upon the particular library available in the particular second ROM chip or chips plugged into the electronic calculator of this invention.

Referring to FIG. 13, there is shown an electronic calculator having an opening 4 for exposing contacts 5, which are connected to the electronics of the calculator. Opening 4 is preferably provided on the rear side of the calculator case 1 as shown in the FIG. 1 and forms, with contact 5, a receptacle for receiving module 48a. Opening 4 is adapted to removably receive the second ROM 48, which is not shown in FIG. 13, but which is disposed in module 48a. Second ROM chip module 48a has contacts (not shown) which mate with contacts 5 for connecting the second ROM 48 therein to the electronic calculator. Door 6 may be closed to retain module 48a in opening 4 during normal operation.

#### THE SPECIFIC EMBODIMENT IN A PROGRAMMABLE CALCULATOR

Having described how the second ROM is advantageously used with an electronic calculator, a particular embodiment of the second ROM in a particular calculator is now described.

Referring now to FIG. 14, there is shown a detailed block diagram of an specific embodiment of a programmable electronic calculator employing the second ROM 48 of this invention. In FIG. 14, there is shown a plurality of chips (48', 10, 12a, 12b, 13, 14a-d and 18). Chips 10, 12a, 12b, 13, and 18 have heretofore been described in some detail in prior U.S. Patents and Patent Applications and therefore reference will be made to U.S. Patents or U.S. Patent Applications, as the case may be, for a detailed description of these chips. The

following discussion will basically relate to how chips 10, 12a, 12b and 13 cooperate with a second ROM chip 48'; which is described hereafter in detail, to implement a calculator having high order capability.

The calculator's arithmetic chip 10 has a plurality of Registers 50a-50e for storing numeric data, an Arithmetic Unit 55 for performing arithmetic operations on the data stored in Registers 50a-e, Flag Registers 53a-b for storing a plurality of flags, a keyboard register 54 which is (1) loadable with a decoded keyboard address derived from the calculator's keyboard, (2) loadable from a subroutine register or (3) loadable from the second ROM chip 48'. Arithmetic chip 10 is described in detail in aforementioned U.S. Pat. No. 3,900,722 which issued to Michael J. Cochran and Charles P. Grant, Jr. on Aug. 19, 1975 and which is assigned to the assignee of this invention. Line 21, column 4 through line 31, column 44 of U.S. Pat. No. 3,900,722 is hereby incorporated herein by reference.

U.S. Pat. No. 3,900,722 discloses a multiple chip calculator system employing the aforementioned arithmetic chip 10 and a scanning and read-only-memory (SCOM) chip. U.S. Pat. No. 3,900,722 discloses that eight SCOM chips may be utilized in a single calculator system. Referring again to FIG. 14, chips 12a and 12b are each double SCOM chips; a double SCOM chip is the equivalent to two SCOM chips of the type disclosed in U.S. Pat. No. 3,900,722 implemented on a single chip of silicon, with the F and G registers thereof replaced by a single eight register memory of the type disclosed in U.S. Patent Application Ser. No. 745,157 which was filed Nov. 26, 1976 and which is assigned to the assignee of this invention.

External ROM chip 13 provides for increased instruction word storage capacity. The ROMs 20a and 20b on double SCOM chips 12a and 12b and the ROM 20c on chip 13 provide the first ROM 20 for storing the microcode which controls the operation of the calculator system. The microcode stored in ROM's 20a-20c is listed in Tables IIa-IIc, respectively. ROM 20c is a 1 K×13 bit ROM while ROMs 20a-20b are each 2.5 K×13 bit ROMs.

Referring briefly to Tables IIa-IIc, the first column thereof is the hexadecimal address of the microcode instruction word appearing in the third column. The second column identifies the chip in which the microcode is stored. TMC-582 and TMC-583 are the two double SCOM chips 12a and 12b; TMC-571 is the external ROM chip 13. The fourth through nineteenth columns contain instruction words whose addresses are incremented by one for each column, reading from left to right. Thus, in Table IIa, the seventeen instruction words in the first row, columns three through nineteen are located at hexadecimal addresses 0000 through 0010. The instruction words are in hexadecimal format also and correspond to the instruction words identified in FIGS. 5a-5h.

As explained in U.S. Pat. No. 3,900,722, the arithmetic chip 10 and the double SCOM chips 12a and 12b are interconnected by lines for exchanging the following control signals: external (EXT), input/output (I/O), instruction words (IRG), and IDLE. External is a serial data channel which may be used, for instance, for addressing ROMs 20a-20c using an address stored in the keyboard register 54 when the PREG bit thereof is a logical one or for inputting or outputting serial data depending on the instruction word outputted on IRG (when the PREG bit is a logical zero). I/O is a four bit



parallel data channel for transferring data in bit parallel, digit serial fashion under control of construction words outputted from ROMs 20a-20c. IRG is a serial channel for transmitting the instruction word from the particular ROM 20a-20c controlling the operation of the system.

In FIG. 14, there are shown four multi-register chips 14a-14d which are connected to the I/O, IRG, and IDLE lines. These multi-register chips are essentially random access memory (RAM) chips which are utilized for storing the data used by the calculator system and programmed functions. It should be evident that the numbers of such chips as well as the size of the RAMS thereon is a design choice.

The magnetic card reader chip 35 is responsive to EXT, IRG and IDLE for inputting digital information to the calculator system from magnetic cards or outputting digital information from the calculator system to magnetic cards. Chip 35 is described in greater detail in U.S. Pat. Application 622,288 filed Oct. 10, 1975 and now U.S. Pat. No. 4,006,455. Of course, the use of a card reader is a design choice. If chip 35 is not utilized, the diode and switch 7 shown in FIG. 7 should be omitted. Switch 7 closes in response to a card being inserted into the card reading mechanism associated with chip 35.

Printer chip 18 may be used to provide the calculator of this invention with printing capability. It should be evident that the utilization of printer chip 18 is a design choice and further this chip may be either permanently installed in a printer calculator or may be installed in a print cradle, such as the PC 100a cradle manufactured by Texas Instruments Incorporated of Dallas, Texas which print cradle may be interfaced with a hand-held calculator provided with printing capability. Chip 18 is described in greater detail in U.S. Pat. Application Ser. No. 428,492 filed Dec. 26, 1973 and now U.S. Pat. No. 4,020,465.

The second ROM chip 48' is interconnected with the calculator system via external, IRG and IDLE. Chip 48' includes a second ROM 48 of the type heretofore discussed plus various control circuits for interfacing it with the remainder of the calculator system disclosed. As previously mentioned, second ROM chip 48' is preferably removable from the calculator of this invention and therefore a plug assembly 43 is provided for ease of removal and insertion. Preferably, the calculator of this invention is provided with a plurality of such second ROM chips 48', at least any one of which may be connected into the calculator system at any given time. This plurality of ROM chips 48' are programmed to provide different types of problem solving capabilities. For instance, one chip 48' might be implemented with programs for solving statistical problems while another might solve financial, surveying, navigation, medical, mechanical or electrical engineering problems, or the like. Moreover, it should become evident to those skilled in the art, that a plurality of such chips 48' might be interfaced with a calculator at one time if such chips were provided with a chip selection means for identifying which second ROM chip 48' is being addressed at any given time. Such chip selection circuits, while not used in the embodiment herein disclosed, are well known in the art.

While the second ROM of this invention is described as a read-only-memory, it should be evident to those skilled in the art that second ROM might be an electrically alterable device, such as an EPROM, or the like.

Similarly, a bubble memory or other such non-volatile memory means could also be utilized as a second ROM.

In Table VIII there is a listing of program codes used in a general purpose second ROM chip 48' to perform such operations as: performing a diagnostic checks, complex math operations, matrix math operations, matrix inversion, annuity and compound interest operations, permutation and combination calculations and the like. The program codes are listed in columns 3-19 of Table VII. The address of the program code in column three is given in column one and the addresses of the other program codes on the same line increment by one for each column reading from left to right.

#### ORGANIZATION OF PROGRAMS STORED IN THE SECOND ROM

As it has been previously discussed, the second ROM stores a plurality of program codes for performing high order functions. The organization of these program codes on chip 48' is now described in detail. In this embodiment, the program codes comprise a pair of four bit binary coded decimal (BCD) digits. Therefore, these codes may be any number between 00 and 99.

Referring now to FIG. 15, there is shown a functional diagram of how the program codes are organized on the second ROM of chip 48' and preferably a second ROM implemented in a pluggable package. In this embodiment ROM 48 stores on the order of 5,000 eight bit program codes. Referring now to FIG. 15, a rectangle thereon represents an eight bit code outputtable from ROM 48 in response to an address. Second ROM 48 stores a plurality of programs, which are for ease of addressing, arranged on "pages". Several programs are preferably allocated to each page. When a program is second ROM 48 is to be accessed from the calculator's keyboard, the operator depresses the "2ND" key and the "program" key (PGM) in this embodiment. The operator next enters a two digit number indicating the page upon which the program he or she wishes to access exists. For instance, if he or she wishes to access a program on page twelve, he or she would depress the one and two number keys. The operator knows upon which page the desired program exists because a program directory is preferably supplied along with a pluggable second ROM chip 48'. The operator then preferably enters a label to uniquely identify the particular program which is desired on the page previously entered. This is done by depressing either a particular label key A-E or A'-E' or the subroutine key (SBR) followed by a non-number key (e.g., SBR, =; or SBR, X<sup>2</sup>; or the like). Depressing the subroutine key and entering a three digit address preferably causes a branch to the location equal to the sum of the inputted address plus the address of the first program code on the inputted page.

The calculator is preferably permanently programmed to first read out the program code at location 0000 which indicates the number of pages stored on that particular second ROM 48. This number is compared with the inputting page number to assure that the inputted page number exists in second ROM 48. Next the security code at location 0001 is preferably outputted for examination, the function of which will be later described. The next step is to address second ROM 48 with the entered page location, the address being derived by multiplying the inputted page number by two. For example, if page 02 is entered, then the address to be used is 0004. At address 0004 is a top half of the

address (the thousands and hundreds digits) for the beginning point of the second page. At address 0005 is the bottom half of the address (the tens and units digits). The program codes at locations 0004 and 0005 define the address where the second page begins in second ROM 48. Locations 0006 and 0007 will also be read out to provide the address of the beginning point of the third page, which is indicative of the ending point of the programs stored on the second page. Thus the address of the second page derived from locations 0004 and 0005 is used as the starting point for a label search and the address of a third page is used to define the ending point of that search.

The program in second ROM 48 is caused to branch to the program code which occurs at the starting point of page two. At page two in second ROM 48, the label search is commenced by reading out program codes sequentially until either the label being searched for is detected or the beginning point of page three is encountered, indicating that the label being searched for does not exist on the page selected. The label being searched for is either a particular label program code (Table III code 10-19) or the label program code (Table III, code 76) followed by a particular non-numeral program code. When the last page is selected, then the address of the last page, as well as the last address on that page are read out to fulfill the function of reading out the addresses of pages 2 and 3 in the foregoing example. This sequence of events is also diagrammatically depicted in FIG. 15.

Referring now to Table III, there is shown a list of the program codes 00-99 preferably used in the calculator system of this invention along with the corresponding functions performed by these codes and the key sequences used to generate the codes when generated from the keyboard. As can be seen, certain program codes may not be directly generated from the calculator's keyboard. The functions performed by the program codes listed in Table III should be evident to those skilled in the art based on the description set forth in Table III. By way of further clarification, however, the inverse function key (INV) is used to perform the inverse of the function indicated for selected keys. For instance, the inverse function key when combined with the LN<sub>x</sub> key causes the calculator to take the number  $e^x$  in lieu of taking the natural logarithm of the number  $x$ . The indirect addressing key (IND, which must be used in combination with the 2ND key, of course) is used with the memory operation keys and "go to" or "conditional go to" keys (GTO,  $X \times T$  or  $X \geq T$ ) to indicate that the number following the program code does not describe either the memory used (if a memory operation) or the branching location (if a go to or conditional go to operation), but rather identifies the particular memory whose contents define either the particular memory to be used (if a memory operation) or the branch address (if a go to type instruction).

Referring again to Table III, program codes 00-09 define the ten numeral keys and the remaining program codes are defined according to the following convention. The first number thereof identifies the keyboard row in which the key is located and the second number defines the keyboard column in which the key is located, for the basic functions which may be accessed by a single key push. For functions accessed by multiple key push sequences, selected merged program codes are utilized. For instance, when the 2ND key is combined with another key to perform the operations indicated,

the number 5 is added to the basic program code (without a carry) to generate the merged program code. Thus, for example, the label A is stored as a program code "11" whereas the label A', which requires the 2ND key to be actuated before the A key, is stored as program code "16". Program codes which otherwise would define those keys performing the numeral functions (eg, 0-9), are reserved for selected merged program codes or for program code not directly generated at the keyboard. For example, program codes 62, 63, 64, 72, 73, 74, 83 and 84 are used for merged program codes wherein the IND key is used. Program codes 82 and 92, which are not defined according to the foregoing convention, define a hierarchy address function and the "return" function. The "return" function has already been mentioned and the hierarchy address function is used to address the eight registers on one of the double SCOM chips, which are set aside for hierarchy control purposes. This calculator system utilizes the algebraic operating system disclosed in U.S. Patent Application Ser. No. 708,958 filed July 26, 1976, for hierarchy control purposes. The hierarchy address code (82) is followed by another program code to define the hierarchy register and operation involved or to define a conditional return, whose function will be mentioned later. The meaning of the program code following the hierarchy address code is set forth in Table IV.

The use of such codes which are not directly accessible from the keyboard permit the accomplishment to special functions or entry in to date areas which are normally isolated from the operator.

The operation code (OP) is used with a following program code for calling the special functions identified in Table V. These routines are implemented in this calculator either in microcode alone or by using second ROM addressing techniques. The second ROM area for such operation code functions is located in the constant ROM areas of double SCOM chips 12a and 12b. Approximately half of the constant ROM in double SCOM chip 12a is used for storing constants in the manner contemplated by U.S. Pat. No. 3,900,722 while the other half of that constant ROM and all of the constant ROM in double SCOM 12b is used for storing program codes, as defined in Table III, in the manner generally set forth in U.S. Patent Application Ser. No. 714,464, filed Aug. 16, 1976. The contents of the constant ROMs are listed in Table VI hereof. Eight two digit program codes are stored in each constant storage area in the constant ROMs. The codes are stored from right to left; thus the first program code in constant area sixteen is an 82. The addresses of these program codes for discussion purposes will be: Constant number hyphen one of eight locations. Thus, the address of the first program code on constant area sixteen (82) is 16-0, while the third program code in constant area eighteen (43) is 18-2. Locations 16-0 through 24-2 contain a slope-intercept routine. The other routines in the constant ROM area are defined in Table VIa. As can be seen, the functions stored in the constant ROM areas are accessed either by OP codes or by normal keyboard entries; for example, the polar to rectangular conversion function is stored in the constant ROM areas and is accessed by depressing the P→R key on the keyboard.

The program codes in Table VI make use of the conditional return program code (82 followed by 20). The conditional return evokes a return function only when all the program codes for the accessed function have been read out. For instance, if the variance is being

calculated, the conditional return stored at locations 26-2 and 26-3 is ignored so that after having found the mean according to the program codes at locations 24-3 through 29-2 is performed. The hierarchy address function (program codes 82,-) used with several routines to address the hierarchy registers in order to maximize the addressable storage area available in RAM chips 14a-14d.

#### STORING A KEYED-IN PROGRAM

When the operator desires to utilize his or her own program in lieu of a program stored in the first or second ROM's, he or she may do so by an appropriate key sequence for storing keyboard enterable program codes of Table III in RAMs 14a-14d. RAMs 14a-14d may also be used for storage of numeric data, i.e. the results of the calculations performed by this electronic calculator. Normally, RAMs 14a-14d provide a storage for storing 480 program codes while RAMs 14c and 14d provide 60 addressable memory locations for storing numeric data. By depressing 2ND, OP, 1, 6, the operator may determine which data configuration RAMs 14a-14d are in; in this case, the number 479.59 would be outputted. The number to the left of the decimal point is the maximum address in RAMs 14a-14d for program steps, while the number to the right is the maximum address in RAMs 14a-14d of memory registers.

By inputting a number between one and ten, 2ND, OP, 1, 7; the inputted number is used as the number of decodes of memory registers set aside in RAMs 14a-14d and the resulting configuration is displayed in the aforementioned manner. For example, inputting 7, 2ND, OP, 1, 7 results in RAMs 14a-14d being repartitioned with seventy memory registers and 400 program step locations; also 399.69 would appear in the display.

RAMs 14a-14d can store up to 120 sixteen digit words and can be partitioned to store as many as 960 program codes with no addressable memory registers to as few as 160 program codes with 100 addressable memories in this embodiment. As can be seen from FIG. 16, the addressable memory locations may be traded at the rate of 10 for 80 program step locations when repartitioning takes place. Of course, the precise number of memories as well as the range of possible data configurations is a design choice.

The partitioning data is stored in digits 3-8 of register 13 on double SCOM chip 14b (see FIG. 19). Digits 8-6 holds the actual address in RAMs 14a-14d defining the location of memory 00 while digits 5-3 hold the largest number assigned to a memory. Thus, in the normal 479.59 configuration, 060 is stored in digits 8-6 while 059 is stored in digits 5-3. Attempting to branch to a program location equal to or greater than the contents of digits 8-6 causes an error condition, as does addressing a memory greater than the number in digits 5-3 of register 13.

Referring again briefly to FIG. 14, the down load operational code may be utilized to permit a page in second ROM 48 to be loaded into the program code storage area of RAMs 14a-14d. The security code in the second ROM 48 to be down loaded must be a 00 to permit the down loading to occur. A 01 security code inhibits the down loading operation, thereby helping to maintain the secrecy of the programs stored in second ROM 48 should that be desired. If the security code bit is set, then the programs in the second ROM may be utilized to perform the function indicated but the series of program codes may not be read out of the calculator

to the operator. After a program is down loaded into RAMs 14a-14d, the operator has free access for examining the program and altering it as he or she sees fit.

#### ALPHANUMERIC PRINTING OPERATIONS

Operational (OP) codes 00-07 are used for alphanumeric printing operations when the calculator of this invention includes a printer. For example, a thermal printer in combination with printing chip 18, such as that provided by the PC-100a disk unit manufactured by Texas Instruments Incorporated of Dallas, Tex. may be used. The twenty character position PC-100a printer may be utilized for printing a line of alphanumeric characters chosen by the operator by loading five two-digit character codes from the display register into four printing buffers. These two-digit alphanumeric character codes are listed in Table VII. The display register is first loaded with five two-digit codes from the keyboard, which are transferred to one of the buffers by OP01-OP04 codes; of course, each buffer stores one fourth of a line of characters.

The contents of the buffers are printed by a 2ND, OP, 0, 5 key sequence or by encountering 69, 05 program codes from either the RAMs 14a-14d or second ROM 48.

OP code 06 causes the printing of the contents of one buffer and the numeric contents of the display register. OP code 07 prints an asterisk in the column corresponding to the interger portion of the number then in the display register, provided that number is in the range of 0-19. Thus, OP code 07 is preferably used for plotting a series of answers obtained by the calculator. Of course, the range 0 to 19 is a design choice and, of course, to make better use of this plotting capability, the answers to be plotted are preferably first normalized to occur within the range 0 to 19.

#### DESCRIPTION OF THE SECOND ROM CHIP AND THE INTERFACE BETWEEN THE SECOND ROM CHIP AND THE OTHER CALCULATOR CHIPS

Referring now to FIG. 17, there is shown a block diagram of second ROM chip 48'. The second ROM 48 implemented thereon, is provided by binary coded decimal (BCD) ROM 600. The output of BCD ROM 600, which outputs the program codes listed in TABLE III, for instance, is connected to serializers 605 and 606. Serializer 605 converts both the high digit and low digit of the program code to serial format and supplies it to a digit selector. The serializer 606 converts only the high digit of the program code to serial and also supplies it to digit selector 607. Digit selector 607 provides the output of serializer 605 to external output control 608 in response to a decoded FETCH instruction or supplies the output of serializer 606 to external output control 608 in response to a decoded FETCH HIGH instruction.

BCD ROM 600 is addressed by an address in program counter 601. The address in program counter 601 preferably is a BCD four digit numeral. The address in program counter 601 is incremented each time a FETCH (but not a FETCH HIGH) instruction is decoded and is maintained in BCD format by an one bit/BCD corrector 604. Program counter 601 may be loaded one digit at a time with a digit appearing on External (EXT) via digit controls 602a-602d. Digit strobe 603 controls which one of the digit control 602a-602d is enabled; digit strobe 603 sequentially ena-

bles digit controls 602a-602d in response to a decoded UNLOAD PC (unload program counter) or a LOAD PC (load program counter) instruction. Digit strobe 603 is reset by a decoded FETCH instruction. The digit control 602a-602d enabled by digit strobe 603 inputs a single digit (four bits) of the four digit number from the EXT bus in response to a decoded LOAD PC instruction, the number being obtained from the bits occurring during state times S3-S6 on the EXT bus. Also, the enabled digit control 602a-602d outputs a digit from program counter 601 to the EXT bus in response to a decoded UNLOAD PC instruction.

State time generator 609 outputs an indication to state time PLA 610 of which one of the 16 possible state times of an instruction cycle the calculator is in. State time generator 609 is responsive to IDLE for maintaining the state times generated thereby in phase with state times generator on the arithmetic chip 10 or double SCOM chips 12a and 12b, for instance. State time PLA 610 outputs selected timing signals to the various logic circuits and also to serial instruction decode 611. Serial instruction decode 611 is also responsive to the serial instruction words appearing on IRG. Serial instruction decode 611 decodes the aforementioned FETCH, FETCH HIGH, UNLOAD PC and LOAD PC instructions.

External output control 608 outputs the serialized low and high program code digits or the serialized high digit alone from digit selector 607 in response to the FETCH and FETCH HIGH instruction, respectively, and outputs the digit from the enabled digit control 602a-602d in response to an UNLOAD PC instruction, the output from external output control being provided to the EXTERNAL bus.

Considering FIGS. 3a and 3b with FIG. 17, the following discussion examines the flow of data between second ROM chip 48', arithmetic chip 10 and the main ROMs 20a-20c. When a program is to be called from second ROM chip 48' according to the aforementioned 2ND, PGM, page number and label key sequence, the location 0000 is first loaded into the program counter 601. This may be done for instance by (1) zeroing a selected register 50a-50e, which loads the zero in register R5 65. As disclosed in the incorporated by reference U.S. Pat. No. 3,900,722, register R5 65 is automatically loaded with the least significant digit after some arithmetic operation involving the arithmetic unit and Register R5 65 may be loaded with the least significant digit in keyboard register 54 in response to a KRR5 instruction. Also the least significant digit of keyboard register 54 may be loaded with the contents of register R5 65 in response to a R5KR instruction. Now, the zero digit in register R5 65 is loaded into the least significant digit position of keyboard register 54. After having reset digit strobe with a FETCH instruction, the zero is loaded into each of the four digits of program counter 601 by four sequential LOAD PC instructions, thereby transferring the zero in the LSD of the keyboard register into all digit positions of program counter 601. Then a FETCH HIGH instruction is issued followed by a load keyboard with EXTERNAL (EXTKR) instruction for loading high digit of the program code outputted from ROM 600 into the keyboard register. The contents of keyboard register 54 is then loaded into register R5 65 and thence into a register 50a-50e using an R5→Adder instruction (FIG. 5b). This high digit may then be shifted in its register to the next more significant digit position by register shift instructions (FIG.

5b). A FETCH instruction then follows, which is followed by a EXTKR, KRR5 and R5→Adder instructions to load the low digit into the register in which the high digit had been previously loaded, thereby providing the high and low digits in one of the operational registers 50a-50e. This number may then be subtracted with the inputted page number to determine whether the inputted page number is available in ROM 600.

As can be seen, having an R5 register of a single digit length complicates the transferring of the low and high order digits of the program code into an operational register 50a-50e. It should be evident that an eight bit register R5 65 would simplify this process; the method herein described is used because register R5 in the pre-existing arithmetic chip 10 only has four bit positions. Modification to the above described instruction sequence for a calculator system having an eight bit R5 register 54 should be evident to those skilled in the art.

It should be also evident to those skilled in the art that this addressing sequence could be simplified if the I/O bus were used in lieu of EXT. Then, however, additional connections would be necessary to interface the second ROM chip 48' with the rest of the calculator system, thereby complicating the plug assembly for interfacing chip 48' with the remainder of the calculator system.

Assuming that the inputted page number is available in ROM 600, the inputted page number is multiplied by two and the resulting address is loaded one digit at a time into program counter 601 via register R5 65, keyboard register 54 and the EXT bus. As has been previously discussed, the two program codes at that location and the following location in ROM 600 are read out thereof one digit at a time using a sequence of FETCH HIGH, EXTKR, KRR5, KR→Adder, SHIFT, FETCH, EXTKR, KRR5, R5→Adder, SHIFT instructions and so forth in the manner previously set forth for four digits. At this time, the four digit address of the first program code for the inputted page number has been loaded into a selected register 50a-50e. This process is then repeated for the next two program codes, which are loaded into another register, this address defining the first program code on the page immediately following the inputted page number. The difference of these two numbers is taken and stored and the address of the program code on the inputted page is then loaded one digit at a time into program counter 601 by generally reversing the above sequence and substituting the FETCH and FETCH HIGH instructions with LOAD PC instructions, thereby loading program counter 601 with the address of the first program code on the inputted page number. The program codes are then read out sequentially and compared with the inputted label, the calculator testing for a match thereof. For each FETCH operation accomplished during this label search, the aforementioned difference is decremented by one. If the decrementing difference becomes equal to zero before a match is found, the calculator generates an error condition inasmuch as the label being searched for does not exist on the particular page inputted by the operator. Once the inputted label is detected, the outputted program codes are loaded into program counter 54 which in turn is used to address main ROMs 20a-20c when a PREG instruction is generated telling main ROMs 20a-20c to branch on the address being outputted on EXT. The microcode at that address is then read out in the usual manner to accomplish the function indicated by the outputted program code, that

sequence of events taking between fifty and several thousand instruction cycles, for instance. At the end of this sequence of instructions, a flag is tested to determine whether another program code is to be read from second ROM 48, or whether a program code is to be read from the program code storage area in RAMs 14a-14c, whether a program code is to be read from the aforementioned constant ROM area in double SCOM chips 12a or 12b or whether the calculator is simply to await another keyboard input by the operator. This flag is a memory on one of the double SCOM chips; the control of which the above-mentioned elements control calculator operation will be subsequently discussed. Assuming control is to be directed back to second ROM 48, FETCH and PREG instructions will again be issued to cause the calculator to branch to the location defined by the subsequent program code in second ROM 48. This process will be repeated until a RETURN program code is loaded into keyboard register 54, and main ROM 20a-20c branches to the location defined thereby; there instructions transfer control back to normal keyboard inputs and the display is activated with the contents of register A, unless of course, the above-mentioned 2ND, PRGM, page number, label key sequence occurred as a separate chain in a user inputted program in RAMs 14a-14d, in which case, control would be then returned to the program thereat.

It has previously been indicated that after the page number has been inputted, a label search is commenced on that page by inputting an appropriate label, i.e. either the labels A-E or A'-E' or the subroutine key (SBR) combined with a non-numeric key (indicating that such key is being used as a label) or a three digit number (which is treated as a relative address, as aforementioned). It should be appreciated that data may be entered into the memories, including the display register, after the page number is inputted and before the label or relative address is inputted. In fact, many of the programs in the embodiment of second ROM 48 programmed according to Table VIII assume that the number in the display at the time the label is inputted is to be used during the execution of the program identified by the label.

Referring now briefly to FIG. 19, there is set out the utilization of the sixteen registers on the two double SCOM chips 12a and 12b. Each register can store up to sixteen digits, each of which has four bits, of course.

Registers 1-8 and 12, as well as digits 1 and 2 in register 13 are reserved for heirarchy control, although registers 1-8 may be addressed from the second ROM using the aforementioned heirarchy address program code.

Register 0 is reserved for: (1) ten user flags, (2) the RAM/Constant ROM Program Counter and Program Source Flag, (3) last key entry and (4) a fixed point display indicator. The Program Source Flag in digit 3 and the digit 15 flag in Register B 53b (FIG. 3a) define where calculator control is to be passed after the present set of instruction words from first ROM 20 are executed. If flag B 15 is set, then the calculator is under control of a program either in (1) RAM's 14a-d, (2) second ROM 48 or (3) the second ROM portion of the constant ROMs on double SCOM chips 12a or 12b. If flag B 15 is reset, the calculator is under normal keyboard control. The program Source Flag is a 0 if control is to be returned to RAM 14a-d area; a 1 through 7 of control is to be passed to second ROM chip 48'; or an

8 or 9 if control is to be passed to the constant ROM area.

Registers 14 and 15 are utilized to permit superrouting of the programs in RAMs 14a-d or in second ROM chip 48'. Thus after control passes to the element specified by the current program control flag, the stack implemented by registers 14 and 15 is popped and the location and new program control flag previously in level one thereof is inserted into digits 7-3 of register 0. Of course, when the number of superroutine levels in the stack is one or greater, as indicated by the number in digit 0 of register 15. The stack is pushed (i.e., another level of superrouting is added) when (1) a function stored in the constant ROM area (e.g. P→R) is encountered; (2) a label is encountered (e.g. A-E or A'E'); (3) a subroutine program code or key depression followed by either a three digit address or a non-numeral label address (which initiates either a branch or a label search in the element presently controlling operation and if the second ROM chip 48' is the controlling element, the label search is limited to the present page); or (4) program codes or key depressions for 2ND, PGM, and then A-E' or SBR and label (e.g. SBR X<sup>2</sup> or SBR =) or SBR and a three digit relative address (initiating either a label search on the indicated page or a branch to a specific program code on that page whose address is determined by adding the relative address to the address of the first program code on that page). As can be seen, six levels of such routines may be employed in this embodiment, with digit 0 of Register 15 indicating the number of levels actually being utilized at any given time.

Register 13 stores the addresses of the useable data storage area in RAMs 14a-d, which may be varied or reconfigured by the operator inputting appropriate OP codes, as previously mentioned. Digit 0 of register 13 contains a flag indicating whether angular results are to be provided in degrees, radians or grads.

Register 11 is used as the T register, for the comparisons made in the "conditional go to" program codes, while register 10 stores eight program codes during operations under RAM 14a-d or constant ROM control, at which time the eight programs stored in a constant area or register in RAM being accessed are temporarily stored in register 10 to simplify the extraction of the program code to be used to address first ROM 20.

Register 9 contains the old and new page numbers which allows the user to change pages easily. Register 9 also contains information about the program size, size of the RAM and also the program's security flag, which may be stored in second ROM 48, if desired, as aforementioned.

Referring now to composite FIGS. 18a-18i, there is shown a detailed logic diagram of second ROM chip 48'. BCD ROM 600 is implemented as a conventional virtual ground type ROM of the type disclosed in U.S. Pat. No. 3,934,233, entitled "Read-Only-Memory For Electronic Calculator", which issued Jan. 20, 1976 and is assigned to the assignee of this invention. Decoders 620 and 621 used in addressing ROM 600 are important features of this invention which permit ROM 600 to be addressed using BCD data without wasted space within ROM 600. The decoders heretofore known in the prior art, such as those exemplified by U.S. Pat. No. 3,934,233, decode either binary, octal, or hexadecimal data, as the case may be. These decoders may be used with a ROM to decode BCD data, of course; however, in that case, large portions of the ROM would go un-

used inasmuch as hexadecimal numbers 11 through 16 would be decodable, but have no need to be decoded. Using the addressing scheme herein disclosed, permits the addressing of ROM 600 with BCD data without the wasted space within the ROM which would otherwise result with conventional decoders.

ROM 600 is implemented as a  $5000 \times 8$  bit array for storing 5000 eight bit program codes, the addresses thereof being the BCD encoded numerals 0000-4999. These four numerals are stored in program counter 601. Decoders 620 and 621 are able to decode these 5000 addresses without decoding the non-BCD codes often seen in the binary data contained in a register such as program counter 601. As will be seen, the ROM metal line decoder 620 uniquely decodes the one of 125 metal lines while the ROM diffusion line decoder 621 decodes one out of forty diffusion lines for each bit in a program code. Inasmuch as  $125 \times 40$  equals 5000, 5000 program codes in ROM 600 will be uniquely identified by metal decoder 620 and diffusion decoder 621.

Metal decoder 620 comprises a plurality of one of five decoders. These one of five decoders are provided in a three level cascaded arrangement inasmuch as  $5 \times 5 \times 5$  equals 125. Referring to dashed lines B1-B5, it can be seen that the dashed line B1 encompasses a one of five decoder, as do references dashed lines B2-B5. The inputs of one of five decoders B1-B5 are connected to the outputs of one of five decoder C1, all of which are used to perform a one out of twenty-five decode. One of five decoders C2-C5 are identical to one of five decoders B1-B5. The five outputs from one of five decoders C2-C5 are each coupled to one of five decoders like B1-B5 (not shown) for providing the decoding of metal lines 26-125. The inputs of one of five decoders C1-C5 are connected to the output of a third level one of five decoder identified by a reference line D. The B level decoders (e.g. B1-B5) are responsive to the A1-A3 bits outputted from program counter 601, while the C level decoders are responsive to the A7-A5 bits and the D level decoder is responsive to the A9-A11 bits from program counter 601. The remaining bits, (e.g. A0, A4, A8, A12, A13, and A14) are decoded by diffusion decoder 621, for providing a one out of forty decode. Diffusion decoder 621 is divided into eight sections 621a-621h for addressing the eight bits of the addressed program code. As can be seen decoder section 621a has one output line ( $P_0$ ) one ground line and 39 intermediate diffusions, inasmuch as ROM 600 is of the virtual ground type. Decoder sections 621b-621h are identical to section 621a, but output the  $P_1$ - $P_7$  bits of the program code.

It should be evident that decoder 621 performs a one out of two decode for address lines A0, A4 and A8 and performs a one out of five decode for address lines A12-A14. It should be evident to one skilled in the art, moreover, that other configurations of decoders 620 and 621 could be utilized. For instance, metal decoder 620 could be arranged to perform a one out of 250 decode by doubling the size of that decoder and adding a one out of two decoder of line A0, for instance, in front of the D level decoders, while diffusion decoder 621 would perform a one of twenty decode by deleting the A0 address line 0 and collapsing the size thereof by deleting the odd numbered diffusion lines.

It should further be evident to those skilled in the art that similar BCD only addressing schemes may be used with other sizes of ROMs. For instance, a 7000 word ROM may be BCD addressed with an one out of 250

metal decoder ( $2 \times 5 \times 5 \times 5$ ) and an one out of twenty-eight diffusion decoder ( $2 \times 2 \times 7$ ). As can be seen; the numbers in the parenthesis are the prime number factors of 7000.

Diffusion decoders sections 621a-621h output the  $P_0$ - $P_7$  bits of the program, respectively. The  $P_0$ - $P_7$  bits are outputted serially on external during state times  $S_3$ - $S_{10}$  in response to a decoded FETCH instruction while bits  $P_4$ - $P_7$  are to be outputted serially on external during  $S_3$ - $S_6$  in response to decoder FETCH HIGH instruction, as aforementioned. The  $P_0$ - $P_7$  bits from decoder 621 are connected to bus 623 when strobed by shift register 622 beginning at  $S_2\phi_2$  for the  $P_0$  bit and when enabled by logic 607a in response to a decoded FETCH instruction. Bits  $P_4$ - $P_7$  are conducted to bus 623 starting at state time  $S_2\phi_2$  when strobed by shift register 622 provided logic 607b receives a FETCH HIGH command. Bus 623 is coupled to the EXT line via external output buffer 624 when enabled by external output control 608. Control 608, in combination with logic 625, couples bus 623 to buffer 624 in response to either a FETCH or FETCH HIGH command. The data on bus 623 is one-half bit early due to a one-half bit delay in buffer 624.

Program counter 601 is provided by a  $15\frac{1}{2}$  bit shift register, comprising thirty-one inverters. The other half bit a delay occurs in the one bit adder/BCD corrector 604, thereby providing sixteen bits of storage for storing four BCD numerals. The least significant digit in program counter 601 is loadable from a four bit serial numeral appearing on EXT during state times  $S_3$ - $S_6$  via digit control 602a in response to a LOAD PC command. As can be seen from FIG. 18, digit control 602a inserts a new digit in the least significant digit position in program counter 601 by coupling the data on EXT to inverter 626 and open-circuiting transfer gate 627 connecting inverter 628 to inverter 626. Inverters 626 and 628 are two of the thirty-one inverters in program counter 601. Digit control 602a also outputs to bus 629 serial data stored in program counter 601 when enabled by digit strobe 603, the digit appearing on line 629 being one-half bit early compared to the time at which it will be outputted on EXT in response to a decoded UNLOAD PC instruction.

Digit controls 602b-602d are identical to digit control 602a, except that they are connected to program counter 601 at appropriate places in the shift register therein to interface with the next to the least significant digit through most significant digit positions during the  $S_3$ - $S_6$  state time period. Bus 629 is connected to external output control 608 which functions in combination with logic 630 for conducting the BCD digit outputted from the enabled digit control 602a-602d through to EXT during  $S_3$ - $S_6$  in response to a decoded UNLOAD PC instruction. Bus 629 is one-half bit early compared to the data on EXT, because of the one-half bit delay associated with output buffer 624.

Digit strobe 603 is comprised of a four bit ring shift register counter for sequentially enabling one of the digit controls 602a-602d. Shift register 603 advances in response to a LOAD PC or UNLOAD PC command by logic 631; shift register 603 for enabling the LSD digit decoder 602a in response to a decoded FETCH instruction by a logic 632.

The four digit BCD number stored in program counter 601 shifts through program counter 601 and from the output thereof back to the input thereof via the one bit adder/BCD corrector 604 each instruction cy-

cle. The contents of shift register 601 is coupled to thirty ROM address buffers 633 during  $S_{15}\phi_2$  through  $S_0\phi_1$ . The output of the thirty address buffers 633 provide the  $A_0$ - $A_{14}$  and  $\bar{A}_0$ - $\bar{A}_{14}$  outputs to the metal decoder 620 and diffusion decoder 621. Only fifteen stages of the shift register in program counter 601 are outputted to ROM 600 via buffers 633 and decoders 620 and 621, inasmuch as the most significant bit of the most significant digit in program counter 601 need not be decoded when the largest address is 4999. It should be evident to one trained in the art, however, to utilize the outputs of all stages and to increase the size of the aforementioned decoder 620 or 621 to accommodate extra address lines for using larger addresses than 4999.

One bit adder/BCD corrector 604 is a single bit, serial adder that adds one to the stream of four BCD digits circulating through it from program counter 601 in response to a decoded FETCH instruction. During NON-FETCH (including FETCH HIGH) instructions, the four BCD digits circulate through one bit adder/BCD corrector 604 without the add one operation being performed. The one bit adder/BCD corrector 604 does a "look ahead" at the serial stream of data exiting from program counter 601 in order to determine if BCD correction is necessary when an add one operation is to be accomplished. When corrector 604 receives the least significant bit of a digit, it is also provided with the most significant bit of that digit for the "look ahead" operation. Corrector 604 is also provided with a clock signal, BCD CORRECT, for indicating to corrector 604 when the least significant bit of a digit is being received thereby. If during this time period the least significant digit is being inputted and a FETCH instruction has been decoded, one will be simply added to the least significant digit unless the first and last bits of that digit are both a logical one, (i.e., a decimal nine has been outputted), then in lieu of adding one, which would form the illegal BCD code 1010, a 0000 is outputted from corrector 604 in the corresponding four state times and an add one operation is accomplished on the next more significant digit via carry circuit 633 in corrector 604. Should that digit contain a nine, the above operation repeats and, if not, a one is added to that digit.

In one bit adder/BCD corrector 604, adder gates 634 perform the add operation when a one is outputted from add one insertion gate 635 unless inhibited by gate 636. Gate 635 is responsive to a decoded FETCH instruction and state time  $S_{15}\phi_2$  for inserting a logical one into adder 634 in the least significant bit of the least significant digit or for inserting a one in the least significant bit when enabled by carry circuit 633. Gate 636 is responsive to a decoded FETCH instruction, the  $A_0$  and  $A_3$  bits from program counter 601 and the output of gate 635 for generating four logical zeros when a BCD nine is to be incremented during a FETCH instruction. Gate 636 is enabled by timing signal BCD correct at  $S_4\phi_1$ ,  $S_8\phi_1$ ,  $S_{12}\phi_1$  and  $S_0\phi_1$ , the BCD correct signal being delayed  $1\frac{1}{2}$  state times by a logic 637. Thus once gate 636 outputs a logical zero disabling the output from adder 634, that condition remains for four state times until the BCD correct signal is again generated from PLA 610. Carry circuit 633 is responsive to adder 634 for generating carries within a digit and to gate 636 for generating carries between digits.

The state time generator 609 comprises sixteen state time drivers, the first of which is responsive to a timing signal on IDLE for sequencing the state times generated thereby with the state times generated on the other chips in the calculator system. The outputs from the sixteen state time drivers are supplied to a PLA 610 for providing various timing signals used, for instance, for outputting the program code during  $S_3$ - $S_{10}$ , receiving and outputting single digit addresses during  $S_3$ - $S_6$  and the like. PLA 610 also outputs in serial fashion four serial trains of digits mimicking the LOAD PC, UNLOAD PC, FETCH HIGH and FETCH instructions which are outputted from main ROM 20a-20c or IRG. Of course, IRG also transmits many instructions which are not decoded on chip 48'.

These four serial bit trains are supplied to four serial instruction decoder circuits E1-E4 in decoder 611, along with the instruction words appearing on IRG. Each decoder in serial instruction decoder 611 performs an exclusive OR function on the data from IRG and one of the bit trains from PLA 610. If the exclusive OR function is satisfied for all bit positions of the instruction word (indicating that there was a match between the instruction word outputted and the bit train from PLA 610), then either a LOAD PC, UNLOAD PC, FETCH HIGH or FETCH instruction has been decoded, depending, of course, on which bit train is provided to the particular decoder for which the exclusive OR function was satisfied. Cross-coupled gates 638 perform exclusive OR function and discharge NODE 639 when a mismatch occurs anytime during the exclusive OR operation. The decoder circuitry encompassed by reference lines E2-E4.

It should be remembered that in the calculator system disclosed, an instruction is normally decoded during one instruction cycle and performed during the following instruction cycle. Inasmuch as the UNLOAD PC, FETCH HIGH AND FETCH operations output data on EXT and considering that the keyboard register 54 in arithmetic chip 10 must be sensitized to input this data by a EXTKR instruction, the data to appear on EXT from chip 48' must appear one instruction cycle later than the normal instruction cycle for executing decoded instructions. That is, the FETCH, FETCH HIGH and UNLOAD PC instructions result in data being outputted on external during the second instruction cycle following decoding. ROM 600 is a relatively large ROM, so for FETCH, or FETCH HIGH instructions the precharge cycle begins during the beginning of the instruction cycle immediately following the decoding of the instruction (by the FETCH M1 and FETCH HIGH M1 signals) whereas the conditional discharge occurs about one instruction cycle later. Thus ROM percharge circuitry 640 is responsive to FETCH M1 and FETCH HIGH M1 whereas the logic 625, 607a, 607b, add one/BCD corrector 604, logic 632 and the like are sensitized to delayed FETCH instructions FETCH M2 or FETCH  $\bar{M}10$ .

I have described my invention in connection with certain specific embodiments thereof. It is to be understood that modification may now suggest itself to those skilled in the art and that this invention is not limited to the specific embodiment disclosed, except as set forth in the appended claims.





TABLE IIA

HEX ADDRESS CHIP INSTRUCTION WORDS+

Table with 13 columns: Hex Address, Chip, and Instruction Words. Contains 256 rows of hexadecimal data for chip TMC5A2.

TABLE IIB

HEX ADDRESS CHIP INSTRUCTION WORDS+

Table with 13 columns: Hex Address, Chip, and Instruction Words. Contains 16 rows of hexadecimal data for chip TMC5A3.

TABLE IIB

HEX CHIP INSTRUCTION WORDS+

Table with 17 columns: Address, Chip, Instruction, and 15 hex words. The table lists instructions for chips TMC5A3 and TMC5A4 across addresses 0AFC to 10B4.



TABLE IIC

| HEX ADDRESS | CHIP   | INSTRUCTION                                  | WORDS+  |
|-------------|--------|--|---|
| 160F        | TMCS71 | 0105 0F29 1000 049A 040R                     | 1005 049A 041F 06D3 0704 0A0C 0611 0A1A 0A04 065A 0A68 072C |
| 1620        | TMCS71 | 180F 0A0D 0A1A 0A0D 0105 049A 049A 0106 009A | 1810 0A0C 0A97 0A1A 00A5 0095 0A6A 1A04                     |
| 1631        | TMCS71 | 0A9A 0A0C 00FA 007A 100A 0709 1A0A 0A7A 1A1A | 0219 064A 0A1A 0A04 0A01 0A6A 1009 0030                     |
| 1642        | TMCS71 | 0A0C 1A04 00A5 0A6A 0130 030A 0A0C 1A0A 0095 | 020A 0A6A 0A0C 0009 1A06 0219 1F2F 094A                     |
| 1653        | TMCS71 | 0A1A 0A04 0219 0A6A 1021 0C0A 1A25 0103 0107 | 0901 02FA 0A1A 0A75 1040 0A10 0AFA 1A3A                     |
| 1664        | TMCS71 | 0A04 0A75 1A0C 0619 0030 0020 1111 0A67 09F0 | 0A00 09AA 111A 0A0C 0095 0075 0045 0A9A                     |
| 1675        | TMCS71 | 0A0A 1005 0A8A 000A 100A 0A9A 1A07 0A6A 0A4A | 0110 1A49 0107 0A45 0A97 02F3 0111 0A1F                     |
| 1686        | TMCS71 | 0103 0AFA 0A5A 1267 0A40 0AFA 1005 0A0R 116C | 0619 1A04 0031 0C0A 0A4A 0A47 0A63 0A0D                     |
| 1697        | TMCS71 | 009A 06A9 1222 0A1A 06A0 06A4 020C 0A05 0A49 | 100C 030A 008A 1A06 0030 120A 090C 0032                     |
| 16A8        | TMCS71 | 0A1A 0619 19EA 072C 0A2A 0A4A 0A40 0164 021A | 0126 0A40 0AFA 11CA 0A80 0A0R 100A 0A01                     |
| 16B9        | TMCS71 | 00AA 1A06 0A5A 1A0A 0A4A 0A40 0AFA 1009 0A37 | 09F0 09A0 008A 020A 0A27 0A6F 0A0C 007F                     |
| 16CA        | TMCS71 | 00AA 190F 0A5A 0A2A 0A0C 010E 0A0A 02FA 02A0 | 0A5A 0A2A 0A0C 0C7E 0A0A 09FA 02F3 02A0                     |
| 16DB        | TMCS71 | 1002 070A 1A15 0619 19AA 01A8 0119 19AA 0020 | 1A06 0A85 1A2E 0130 0A0C 0099 0A40 0AFA                     |
| 16EC        | TMCS71 | 1010 0A80 0A0A 100F 0A5A 0A04 0A05 1A07 004A | 1146 1A90 0020 1A04 0085 0A0A 0A3A 1A10                     |
| 16FD        | TMCS71 | 0F2A 0306 1A4F 0A77 0AFA 0C76 0C64 0AFA 01A1 | 0113 0C7E 0A5E 097E 097E 097E 097E 097E                     |
| 170E        | TMCS71 | 0A7A 097E 027F 097F 097E 0153 0153 0A0C 0931 | 0A1A 02F3 0A45 1A04 0085 0A55 1A04 0095                     |
| 171F        | TMCS71 | 0A65 1A04 00A5 0A75 1A04 0085 02A0 097E 0931 | 0A1A 02F3 02B0 1002 0A0A 0A3A 070A 1A4F                     |
| 1730        | TMCS71 | 0A65 072A 000A 1A67 0A0C 0201 0A1A 0AFA 0A3A | 0A0A 0A3A 0A0A 0107 0C4A 0C43 0C90 0A8A                     |
| 1741        | TMCS71 | 0C80 0A0A 0000 0A4A 1002 004A 00AA 0091 020A | 009A 1A45 000A 0A1F 000A 0529 1A4F 1A87                     |
| 1752        | TMCS71 | 02A0 0231 1002 0A1A 0A0A 0A3A 017E 070A 1A11 | 0A0C 0105 0206 0C76 0A0E 010C 0107 0130                     |
| 1763        | TMCS71 | 0107 0A77 0AFA 00AA 1A61 0A5A 0A2A 0A0C 0153 | 0A0A 02F3 0A07 0A1A 0A85 1A04 0045 0A95                     |
| 1774        | TMCS71 | 1A04 00A5 0A45 1A04 00A5 0A45 1A04 0075 02A0 | 0153 0A0A 02F3 02A0 1A02 070A 1A37 0C6A                     |
| 1785        | TMCS71 | 0AFA 01A1 0111 0C6C 0A5A 0A2A 0A0C 072A 00AA | 0A77 0AFA 1A4A 0A0A 02F3 02A0 0219 0A4A                     |
| 1796        | TMCS71 | 1A4A 0A0A 0099 0A02 0A0A 0C0A 0C0A 1A83 008A | 1A1A 070C 19EA 01A9 0A6F 0230 117C 0A85                     |
| 17A7        | TMCS71 | 1953 0031 1957 0049 19A7 050A 0A1A 0A47 02F0 | 1012 0A65 101F 02A6 0131 0A0F 0111 0107                     |
| 17B8        | TMCS71 | 1C1A 0A0C 0300 0101 0A0F 0A04 0A65 1A0A 1013 | 0A9A 0A0A 1005 0A8A 0A45 1A04 0011 0A87                     |
| 17C9        | TMCS71 | 02F0 0101 0A1F 0900 0A47 0AFA 0010 1A0A 070A | 0A0C 094A 0A6A 094A 0A1A 0A55 1A04 0085                     |
| 17DA        | TMCS71 | 0A65 1A04 0095 0A45 0A3A 0A1A 1A04 0075 0A6A | 070A 1A25 0010 0051 0102 1A83 0102 030A                     |
| 17EB        | TMCS71 | 02A9 1A45 0A8A 1A6F 0099 1CC5 0030 1960 0A97 | 02F3 0111 0A1F 0106 0C76 0506 0C7E 0111                     |
| 17FC        | TMCS71 | 0A0F 0131 0021 1987                          |   |

TABLE III

TABLE III-continued

| PROGRAM CODE | FUNCTION                        | KEY(s)         |    | PROGRAM CODE | FUNCTION                                | KEY(s)         |
|--------------|---------------------------------|----------------|----|--------------|---|----------------|
| 00           | 0                               | 0              | 30 | 44           | Sum into Memory #                       | SUM            |
| 01           | 1                               | 1              |    | 45           | Y <sup>x</sup>                          | Y <sup>x</sup> |
| 02           | 2                               | 2              |    | 46           | Insert Program Code                     | 2nd, INS       |
| 03           | 3                               | 3              |    | 47           | Clear Memories                          | 2nd, CMs       |
| 04           | 4                               | 4              | 35 | 48           | Exchange Display and Memory #           | 2nd, EXC       |
| 05           | 5                               | 5              |    | 49           | Multiply Display into Memory #          | 2nd, prod      |
| 06           | 6                               | 6              |    | 50           | Absolute Value                          | 2nd,  x        |
| 07           | 7                               | 7              |    | 51           | Back Step                               | BST            |
| 08           | 8                               | 8              |    | 52           | Exponent Entry                          | EE             |
| 09           | 9                               | 9              |    | 53           | (                                       | (              |
| 10           | E <sup>1</sup>                  | 2nd, E         | 40 | 54           | )                                       | )              |
| 11           | A                               | A              |    | 55           | ÷                                       | ÷              |
| 12           | B                               | B              |    | 56           | Delete Program Code                     | 2nd, DEL       |
| 13           | C                               | C              |    | 57           | Engineering Notation                    | 2nd, ENG       |
| 14           | D                               | D              |    | 58           | Fixed Point Notation                    | 2nd, FIX       |
| 15           | E                               | E              |    | 59           | Integer                                 | 2nd, INT       |
| 16           | A <sup>1</sup>                  | 2nd, A         | 45 | 60           | Degree                                  | 2nd, DEG       |
| 17           | B <sup>1</sup>                  | 2nd, B         |    | 61           | Go To                                   | GTO            |
| 18           | C <sup>1</sup>                  | 2nd, C         |    | 62           | Indirect Program Page #                 | 2ND, PGM       |
| 19           | D <sup>1</sup>                  | 2nd, D         |    |              |   | 2ND, IND       |
| 20           | Clear                           | 2ND, CLR       |    | 63           | Exchange Indirect Memory # with display | 2ND, EXC       |
| 21           | 2nd                             | 2nd            |    | 64           | Multiply Display Into Indirect Memory # | 2ND, PROD      |
| 22           | Inverse Function                | INV            | 50 | 65           | Multiply                                | X              |
| 23           | LNX                             | LNX            |    | 66           | Pause                                   | 2nd, PAUSE     |
| 24           | Clear Entry                     | CE             |    | 67           | Go To # if x = t                        | 2nd, x = t     |
| 25           | Clear                           | CLR            |    | 68           | No Operation                            | 2nd, NOP       |
| 26           | 2nd                             | 2nd, 2nd       |    | 69           | Operation Code #                        | 2nd, OP        |
| 27           | Inverse Function                | 2nd, INV       |    | 70           | Radians                                 | 2nd, RAD       |
| 28           | log                             | 2nd, log       | 55 | 71           | Subroutine Call                         | SBR            |
| 29           | Clear Program                   | 2nd, CP        |    | 72           | Store in Indirect Memory #              | STO, 2ND, IND  |
| 30           | Tangent                         | 2nd, TAN       |    | 73           | Recall Indirect Memory #                | RCL, 2ND, IND  |
| 31           | Learn                           | LRN            |    | 74           | Add Display into Indirect Memory #      | SUM, 2ND, IND  |
| 32           | Exchange display and T register | X ⇌ T          |    | 75           | Minus                                   | -              |
| 33           | X <sup>2</sup>                  | X <sup>2</sup> |    | 76           | Label Key                               | 2nd, LBL       |
| 34           | √X                              | √X             | 60 | 77           | Go To # if x ≥ t                        | 2nd, x ≥ t     |
| 35           | 1/X                             | 1/X            |    | 78           | Insert Data Point                       | 2nd, Σ+        |
| 36           | Program Page                    | 2nd, PGM       |    | 79           | Mean                                    | 2nd, X̄        |
| 37           | Polar to Rectangular            | 2nd, P → R     |    | 80           | Grad                                    | GRD            |
| 38           | Sine                            | 2nd, Sin       |    | 81           | Reset                                   | RST            |
| 39           | Cosine                          | 2nd, COS       | 65 |              |   |                |
| 40           | Indirect Addressing             | 2nd, IND       |    |              |   |                |
| 41           | Single Step                     | SST            |    |              |   |                |
| 42           | Store in Memory #               | STO            |    |              |   |                |
| 43           | Recall from Memory #            | RCL            |    |              |   |                |

TABLE III-continued

| PROGRAM CODE | FUNCTION                                   | KEY(s)                     |
|--------------|--|----------------------------|
| 82           | Hierarchy Address                          | Not directly accessible    |
| 83           | Go to Indirect Address                     | GTO, 2ND                   |
| 84           | Operation Code Indirect #                  | IND<br>2ND, OP<br>2ND, IND |
| 85           | Plus                                       | +                          |
| 86           | Set Flag #                                 | 2nd, ST FLG                |
| 87           | If Flag # Set, Go To #                     | 2nd, IF FLG                |
| 88           | Degrees, Minutes, Seconds                  | 2nd, DMS                   |
| 89           | $\pi$                                      | 2nd, $\pi$                 |
| 90           | List Program                               | 2nd, LIST                  |
| 91           | Run/Stop                                   | R/S                        |
| 92           | Return                                     | INV, SBR                   |
| 93           | Decimal Point                              | .                          |
| 94           | Change Sign                                | +/-                        |
| 95           | Equals                                     | =                          |
| 96           | Write                                      | 2nd, Write                 |
| 97           | Decrement Register # and Go To # when zero | 2nd, DSZ                   |
| 98           | Advance Paper                              | 2nd, PAP                   |
| 99           | Print                                      | 2nd, PRT                   |

TABLE IV

| Program codes following an heirarchy address code (82) |  |
|--|--|
| FIRST DIGIT  | FUNCTION                                     |
| 0  | store  |
| 1  | recall                                       |
| 2  | conditional return (second digit is ignored) |
| 3  | sum into                                     |
| 4  | multiply into                                |
| 5  | subtract from                                |
| 6  | divide into                                  |
| 7  | "  |
| 8  | "  |
| 9  | "  |

| SECOND DIGIT | HEIRARCHY REGISTER |
|--------------|--------------------|
| 0            | no operation       |
| 1            | 1                  |
| 2            | 2                  |

TABLE IV-continued

| Program codes following an heirarchy address code (82) |                    |  |
|--|--------------------|--|
| SECOND DIGIT   | HEIRARCHY REGISTER |  |
| 3  | 3                  |  |
| 4  | 4                  |  |
| 5  | 5                  |  |
| 6  | 6                  |  |
| 7  | 7                  |  |
| 8  | 8                  |  |
| 9  | no operation       |  |

TABLE V

| Codes Following OP Code (69) |   |
|------------------------------|---|
| CODE                         | MEANING   |
| 01                           | Initialize for alphanumeric printing                                |
| 02                           | Fill far left quarter of print buffer                               |
| 03                           | Fill next to left quarter of print buffer                           |
| 04                           | Fill far right quarter of print buffer                              |
| 05                           | Print the buffer as filled with OPS 01-04                           |
| 06                           | Print display plus contents of OP 04                                |
| 07                           | Plot asterisk in column number contained in display register (0-19) |
| 08                           | List labels   |
| 09                           | Download page   |
| 10                           | Signum  |
| 11                           | Variance  |
| 12                           | Slope, intercept  |
| 13                           | Correlation   |
| 14                           | y'  |
| 15                           | x'  |
| 16                           | See current partition RAM   |
| 17                           | Repartition RAM   |
| 18                           | If not error - Set flag 7   |
| 19                           | If error - Set flag 7   |
| 20                           | Increment memory 0-9  |
| 21                           |   |
| 22                           |   |
| 23                           |   |
| 24                           | Decrement memory 0-9  |
| 25                           |   |
| 26                           |   |
| 27                           |   |

TABLE VI

```
SCMD 1,STOF=0A0F,RCLF=0A1F,REG=LO,DIGITS=USFD
ADDRESS CONSTANT=1,2,3,5,A,7,A
CONSTANT 0=230258502090000
CONSTANT 1=06931471A0599045
CONSTANT 2=0095310179A0325
CONSTANT 3=000950330A5316A
CONSTANT 4=00009995003330A
CONSTANT 5=000099995000333
CONSTANT 6=0000000999950000
CONSTANT 7=00000009999500
CONSTANT 8=07A539A163397450
CONSTANT 9=0009966A65249120
CONSTANT 10=00099966A66670
CONSTANT 11=00009999996666A7
CONSTANT 12=00009999999966A7
CONSTANT 13=157079632679501C
CONSTANT 14=14159265359000C
CONSTANT 15=472957795130A01C
CONSTANT 16=4375060353530A82
CONSTANT 17=5403435501436504
CONSTANT 18=3300437505435355
CONSTANT 19=2453535454034355
CONSTANT 20=0143A59004433265
CONSTANT 21=5320A2540343555A
CONSTANT 22=3255541AA2A59053
CONSTANT 23=051AA2653253925A
CONSTANT 24=4355044353925032
CONSTANT 25=4355014353375003
CONSTANT 26=0590533320A25003
CONSTANT 27=3332540343550203
CONSTANT 28=0343550503A59053
CONSTANT 29=750543535392325A
CONSTANT 30=554034355330043
CONSTANT 31=0A2540175034353
```

```
SCMD 2,STOF=0A0F,RCLF=0A1F,REG=HI,DIGITS=UNISFD
ADDRESS CONSTANT=1,2,3,5,A,7,A
CONSTANT 32=75020435353323050
CONSTANT 33=5550034355330143
CONSTANT 34=0353539230541A82
CONSTANT 35=551436504037506
CONSTANT 36=054353535500343
CONSTANT 37=5403435533040375
CONSTANT 38=3301437502035365
CONSTANT 39=024034540034355
CONSTANT 40=3202043308A20144
CONSTANT 41=443307A24A20404
CONSTANT 42=0100903602610105
CONSTANT 43=22320200903308A2
CONSTANT 44=903307A24A20404
CONSTANT 45=37A2030400010504
CONSTANT 46=433217A206401A82
CONSTANT 47=32A5330A2539203
CONSTANT 48=53533234543307A2
CONSTANT 49=53922540505517A2
CONSTANT 50=225017A2551A8253
CONSTANT 51=390A25392245030
CONSTANT 52=4253325407A23265
CONSTANT 53=53925017A2653A1A
CONSTANT 54=0006A5590A25353
CONSTANT 55=016559221A253A5
CONSTANT 56=6550590A2540000
CONSTANT 57=5459221A2550693
CONSTANT 58=0253539250060355
CONSTANT 59=6559221A2555904
CONSTANT 60=54000001A5540693
CONSTANT 61=18A2A55953530A82
CONSTANT 62=9345500693455922
CONSTANT 63=000000092540100
```

TABLE VIa

| LOCATIONS | ROUTINE  |
|-----------|--|
| 16-0-21-6 | m,b  |
| 21-7-23-1 | X <sup>1</sup>                                     |
| 23-2-24-2 | Y <sup>1</sup>                                     |
| 24-3-26-3 | mean   |
| 26-4-29-2 | variance   |
| 29-3-34-4 | standard deviation                                 |
| 34-5-39-7 | correlation coefficient(R)                         |
| 40-0-42-3 | Σ+   |
| 42-4-47-1 | Σ-   |
| 47-2-51-2 | R → P  |
| 51-3-53-6 | P → R  |
| 53-7-58-4 | D.MS (degrees, decimal point, minutes and seconds) |
| 58-5-63-3 | D.d (degrees, decimal point, fractional degree)    |

TABLE VII

| ALPHANUMERIC CHARACTER | ALPHANUMERIC CODE |
|------------------------|-------------------|
| A                      | 13                |
| B                      | 14                |
| C                      | 15                |
| D                      | 16                |
| E                      | 17                |
| F                      | 21                |
| G                      | 22                |
| H                      | 23                |
| I                      | 24                |
| J                      | 25                |
| K                      | 26                |
| L                      | 27                |
| M                      | 30                |
| N                      | 31                |
| O                      | 32                |
| P                      | 33                |
| Q                      | 34                |
| R                      | 35                |
| S                      | 36                |
| T                      | 37                |
| U                      | 41                |

TABLE VII-continued

| ALPHANUMERIC CHARACTER | ALPHANUMERIC CODE |
|------------------------|-------------------|
| V                      | 42                |
| W                      | 43                |
| X                      | 44                |
| Y                      | 45                |
| Z                      | 46                |
| 0                      | 01                |
| 1                      | 02                |
| 2                      | 03                |
| 3                      | 04                |
| 4                      | 05                |
| 5                      | 06                |
| 6                      | 07                |
| 7                      | 10                |
| 8                      | 11                |
| 9                      | 12                |
| blank                  | 00                |
| -                      | 20                |
| .                      | 40                |
| +                      | 47                |
| x                      | 50                |
| *                      | 51                |
| √                      | 52                |
| π                      | 53                |
| e                      | 54                |
| (                      | 55                |
| )                      | 56                |
| ,                      | 57                |
| †                      | 60                |
| %                      | 61                |
| >                      | 62                |
| <                      | 63                |
| /                      | 64                |
| =                      | 65                |
| ,                      | 66                |
| ×                      | 67                |
| x <sup>2</sup>         | 70                |
| ?                      | 71                |
| ÷                      | 72                |
| !                      | 73                |
|                        | 74                |
| Δ                      | 75                |
| ∏                      | 76                |
| Σ                      | 77                |

TABLE VIII

| BCD ADDRESS | CHIP   | PROGRAM CODES+                                     |
|-------------|--------|--|
| 0           | TMC541 | 25 00 00 54 02 43 11 41 14 15 15 A2 17 01 19 51 20 |
| 17          | TMC541 | 29 21 73 22 91 24 14 26 09 27 64 29 52 30 95 32 31 |
| 34          | TMC541 | 33 63 34 80 36 51 42 40 44 31 45 26 46 20 4A 72 47 |
| 51          | TMC541 | 6A 4A 92 76 24 00 42 09 60 58 09 76 25 29 0A 42 01 |
| 6A          | TMC541 | 00 72 01 97 01 00 15 92 76 95 71 24 05 32 03 00 37 |
| 85          | TMC541 | 7A 22 37 7A 69 12 8A 7A 69 11 22 8A 22 7A 69 14 53 |
| 102         | TMC541 | 24 75 36 15 71 8A 54 52 22 52 32 03 07 07 93 02 05 |
| 119         | TMC541 | 08 00 09 05 04 67 96 00 35 76 9A 69 00 01 03 03 06 |
| 13A         | TMC541 | 03 07 01 07 03 05 69 04 03 00 69 03 69 05 01 99 92 |
| 153         | TMC541 | 76 11 9A 99 62 00 11 99 92 76 12 98 99 62 00 12 99 |
| 170         | TMC541 | 92 76 13 9A 99 62 00 13 99 92 76 14 9A 99 62 00 14 |
| 1A7         | TMC541 | 99 92 76 15 9A 99 62 00 15 99 92 76 16 9A 99 62 00 |
| 204         | TMC541 | 16 99 92 76 17 9A 99 62 00 17 99 92 76 1A 9A 99 62 |
| 221         | TMC541 | 00 18 99 92 76 19 9A 99 62 00 19 99 92 76 10 9A 99 |
| 23A         | TMC541 | 62 00 10 99 92 76 19 85 53 24 75 01 54 65 43 07 85 |
| 255         | TMC541 | 07 95 92 76 1A 61 0A 11 01 22 44 01 43 07 22 44 02 |
| 272         | TMC541 | 92 22 97 05 00 51 75 01 44 01 43 07 44 02 73 01 65 |
| 2A9         | TMC541 | 73 02 61 00 30 92 22 97 05 00 69 75 71 00 20 73 01 |
| 306         | TMC541 | 65 73 02 61 00 52 92 76 11 42 07 99 9A 92 76 12 75 |
| 323         | TMC541 | 32 01 95 65 43 07 85 0A 95 42 01 32 92 99 72 01 92 |
| 340         | TMC541 | 32 01 44 01 32 61 00 93 76 13 43 07 42 05 85 33 85 |
| 357         | TMC541 | 07 95 42 01 43 05 72 01 01 22 44 01 97 05 01 18 01 |
| 374         | TMC541 | 42 04 42 06 43 04 85 07 95 42 03 75 43 07 75 0A 95 |
| 391         | TMC541 | 42 05 43 04 19 42 02 73 02 50 32 22 97 05 01 8A 01 |
| 40A         | TMC541 | 44 02 73 02 50 22 77 01 59 32 43 07 85 43 05 85 0A |
| 425         | TMC541 | 95 42 03 61 01 59 43 03 32 43 04 85 07 95 67 02 2A |
| 442         | TMC541 | 42 02 01 94 49 06 43 07 85 01 95 42 05 73 02 63 03 |
| 459         | TMC541 | 72 02 43 07 44 02 44 03 97 05 02 12 43 04 19 42 01 |
| 476         | TMC541 | 73 01 49 0A 29 67 03 31 43 07 42 05 43 04 19 42 02 |
| 493         | TMC541 | 85 43 05 75 43 04 95 42 03 73 03 55 73 07 95 42 01 |
| 510         | TMC541 | 72 03 43 07 44 02 44 03 33 85 0A 95 32 43 03 77 02 |
| 527         | TMC541 | 97 43 01 94 65 73 02 95 74 03 61 02 69 01 22 44 05 |
| 544         | TMC541 | 43 05 32 43 04 67 03 12 61 02 45 01 44 04 43 07 32 |

TABLE VIII

| BCD ADDRESS | CHIP   | PROGRAM CODES*                                     |
|-------------|--------|--|
| 5A1         | TMC541 | 43 04 67 03 26 61 01 35 19 42 01 73 01 49 06 43 06 |
| 57A         | TMC541 | 9A 09 9A 92 6A 6A 6A 6A 6A 6A 6A 6A 6A 6A 6A 76 14 |
| 505         | TMC541 | 42 05 92 42 03 43 07 33 85 07 95 42 01 43 05 32 43 |
| 612         | TMC541 | 07 22 77 03 54 01 44 01 73 01 22 67 03 74 43 07 44 |
| 629         | TMC541 | 01 01 44 05 43 03 72 01 99 92 61 03 55 76 15 01 42 |
| 646         | TMC541 | 00 43 07 85 33 85 07 95 42 01 43 04 42 05 75 43 07 |
| 6A3         | TMC541 | 85 07 95 42 02 00 71 00 30 85 01 44 01 73 01 95 72 |
| 6A0         | TMC541 | 01 01 44 04 43 04 32 43 07 77 04 04 01 42 04 07 85 |
| 697         | TMC541 | 53 43 07 85 01 54 33 95 42 01 75 43 07 75 43 04 42 |
| 714         | TMC541 | 05 95 42 02 00 71 00 52 85 71 00 20 73 01 95 55 73 |
| 731         | TMC541 | 02 95 72 01 01 44 04 43 04 32 43 07 77 04 52 9A 01 |
| 748         | TMC541 | 92 76 16 85 32 07 85 43 07 65 53 24 85 01 54 95 42 |
| 765         | TMC541 | 01 32 92 73 01 99 92 01 44 01 61 05 25 76 17 01 42 |
| 782         | TMC541 | 04 43 04 19 95 42 01 73 01 35 72 01 01 44 04 43 04 |
| 799         | TMC541 | 32 43 07 77 05 40 01 94 44 04 01 42 03 43 04 19 42 |
| A16         | TMC541 | 01 75 43 03 42 05 95 42 02 73 01 65 73 02 94 71 00 |
| 833         | TMC541 | 52 95 65 71 00 20 73 02 95 72 01 01 44 03 43 04 32 |
| 850         | TMC541 | 43 03 22 67 05 69 01 42 03 22 44 04 43 04 32 01 22 |
| 8A7         | TMC541 | 67 05 69 43 04 19 42 01 85 43 03 42 05 95 42 02 73 |
| 8A4         | TMC541 | 02 94 71 00 30 85 01 44 01 95 72 01 01 44 03 43 03 |
| 901         | TMC541 | 32 43 07 75 43 04 95 77 06 27 01 44 04 42 03 43 07 |
| 918         | TMC541 | 32 43 04 22 67 06 27 01 42 04 43 04 42 05 19 42 03 |
| 935         | TMC541 | 43 05 19 42 01 43 03 42 02 43 07 75 43 05 85 01 95 |
| 952         | TMC541 | 4A 05 32 73 03 94 71 00 30 95 94 72 03 43 07 44 03 |
| 9A9         | TMC541 | 32 85 01 95 42 05 32 77 06 92 32 43 04 67 0A 0A 85 |
| 9A6         | TMC541 | 01 95 42 05 43 04 19 85 01 95 42 03 43 05 19 42 02 |
| 1003        | TMC541 | 43 03 42 01 43 07 75 43 05 85 01 95 4A 05 32 73 01 |
| 1020        | TMC541 | 65 73 02 94 71 00 30 95 94 72 03 01 44 03 85 32 95 |
| 1037        | TMC541 | 42 05 32 43 07 77 07 55 01 44 04 61 06 85 9A 01 92 |
| 1054        | TMC541 | 42 03 32 43 07 22 77 08 10 85 42 05 33 85 07 95 42 |
| 1071        | TMC541 | 01 00 42 04 73 01 67 08 45 01 22 44 01 97 05 0A 32 |
| 1088        | TMC541 | 43 05 75 01 95 65 43 07 85 07 95 42 01 32 92 01 44 |
| 1105        | TMC541 | 01 44 04 73 01 99 92 43 04 32 43 07 22 67 0A 60 01 |
| 1122        | TMC541 | 85 43 03 95 1A 61 0A 60 7A 10 13 24 67 0A 95 17 43 |
| 1139        | TMC541 | 0A 92 7A 11 0A 04 4A 03 43 04 9A 99 92 76 12 75 32 |
| 1156        | TMC541 | 01 95 65 43 03 85 0A 95 42 07 32 9A 92 76 50 72 07 |
| 1173        | TMC541 | 32 01 44 07 32 99 92 61 50 76 13 85 32 43 04 75 01 |
| 1190        | TMC541 | 95 65 43 03 85 0A 95 42 07 32 9A 92 61 50 76 14 4A |
| 1207        | TMC541 | 0A 0A 05 43 0A 9A 99 92 76 15 07 42 01 85 53 43 03 |
| 1224        | TMC541 | 65 43 04 54 42 07 95 42 02 76 33 01 44 01 40 02 43 |
| 1241        | TMC541 | 05 65 73 01 85 43 06 65 73 02 95 72 01 97 07 33 01 |
| 1258        | TMC541 | 92 76 1A 12 92 76 45 73 07 99 92 01 44 07 61 45 76 |
| 1275        | TMC541 | 17 85 32 43 03 65 53 43 04 85 01 54 85 0A 95 42 07 |
| 1292        | TMC541 | 32 9A 92 76 60 72 07 32 01 44 07 32 99 92 61 60 76 |
| 1309        | TMC541 | 1A 43 03 75 32 0A 95 94 42 01 76 52 43 03 65 53 43 |
| 1326        | TMC541 | 04 42 07 85 01 54 85 0A 95 42 02 00 76 59 85 43 03 |
| 1343        | TMC541 | 44 01 01 44 02 73 01 65 73 02 97 07 59 95 4A 03 44 |
| 1360        | TMC541 | 01 4A 03 72 01 32 75 01 95 42 01 29 67 25 90 85 0A |
| 1377        | TMC541 | 95 4A 01 32 61 52 76 25 01 92 76 19 85 32 07 85 43 |
| 1394        | TMC541 | 03 65 43 04 95 42 07 32 9A 92 76 57 73 07 99 92 01 |
| 1411        | TMC541 | 44 07 61 57 76 10 43 01 0A 03 42 01 43 02 4A 04 42 |
| 1428        | TMC541 | 02 00 92 76 11 29 70 4A 02 4A 01 43 02 24 92 76 1A |
| 1445        | TMC541 | 4A 0A 4A 03 43 04 24 92 76 12 43 04 4A 02 43 02 32 |
| 1462        | TMC541 | 43 03 44 01 43 01 92 76 17 01 94 49 03 49 04 12 92 |
| 1479        | TMC541 | 76 13 53 43 01 65 43 03 75 43 02 65 43 04 54 32 53 |
| 1496        | TMC541 | 43 01 65 43 04 85 43 02 65 03 03 54 42 02 32 42 01 |
| 1513        | TMC541 | 92 76 1A 01 94 49 04 53 43 03 33 85 43 04 33 54 35 |
| 1530        | TMC541 | 49 01 49 02 13 92 76 14 10 36 05 12 29 67 10 3A 05 |
| 1547        | TMC541 | 16 13 3A 05 17 92 76 19 3A 05 16 10 3A 05 1A 10 1A |
| 1564        | TMC541 | 92 76 15 10 3A 05 12 29 67 10 3A 05 16 1A 3A 05 17 |
| 1581        | TMC541 | 92 76 10 53 53 43 02 22 23 85 35 54 55 02 54 70 92 |
| 1598        | TMC541 | 74 11 3A 04 11 92 76 12 70 43 01 32 43 02 22 37 32 |
| 1615        | TMC541 | 92 76 1A 12 23 42 01 32 42 02 32 92 76 17 70 43 01 |
| 1632        | TMC541 | 22 23 32 43 02 37 42 02 32 42 01 92 76 13 12 33 53 |
| 1649        | TMC541 | 32 65 76 52 02 54 37 42 02 32 42 01 92 76 14 12 34 |
| 1666        | TMC541 | 53 32 55 61 52 76 15 01 42 03 00 42 04 3A 04 10 36 |
| 1683        | TMC541 | 04 1A 92 76 1A 53 53 43 02 22 23 75 35 54 55 02 54 |
| 1700        | TMC541 | 92 76 1A 70 53 53 43 01 85 01 54 33 85 43 02 33 54 |
| 1717        | TMC541 | 34 92 76 10 53 53 43 01 75 01 54 33 85 43 02 33 54 |
| 1734        | TMC541 | 34 92 76 15 53 53 16 85 10 54 55 02 85 53 52 22 52 |
| 1751        | TMC541 | 33 75 01 54 34 54 23 92 76 11 3A 04 11 92 76 12 70 |
| 1768        | TMC541 | 53 43 01 3A 65 3A 05 10 54 32 53 43 01 39 65 36 05 |
| 1785        | TMC541 | 1A 54 42 02 32 42 01 92 76 13 70 53 43 01 39 65 36 |
| 1802        | TMC541 | 05 10 54 32 53 43 01 38 94 65 3A 05 1A 54 42 02 32 |
| 1819        | TMC541 | 42 01 92 76 14 43 01 42 03 43 02 42 04 13 3A 04 10 |
| 1836        | TMC541 | 12 3A 04 1A 92 76 17 53 53 16 75 10 54 55 02 54 22 |
| 1853        | TMC541 | 3A 32 15 42 02 32 42 01 92 76 1A 53 53 16 75 10 54 |
| 1870        | TMC541 | 65 02 54 22 39 32 15 94 42 02 32 42 01 92 76 19 53 |
| 1887        | TMC541 | 53 01 75 43 01 33 75 43 02 33 54 32 53 02 65 43 01 |
| 1904        | TMC541 | 54 22 37 55 02 54 32 53 53 43 01 33 85 53 43 02    |
| 1921        | TMC541 | 85 01 54 33 54 55 53 43 01 33 85 53 43 02 75 01 54 |
| 1938        | TMC541 | 33 54 54 23 55 04 54 42 02 32 42 01 92 76 11 42 04 |
| 1955        | TMC541 | 99 92 76 12 53 24 85 32 05 54 42 01 32 9A 92 76 50 |
| 1972        | TMC541 | 72 01 32 01 44 01 32 99 92 61 50 76 13 9A 99 42 03 |
| 1989        | TMC541 | 53 43 04 42 02 85 05 54 42 01 01 44 02 73 01 76 60 |
| 2006        | TMC541 | 22 97 02 70 22 97 01 70 53 24 65 43 03 85 73 01 54 |
| 2023        | TMC541 | 61 60 76 70 99 92 76 11 42 01 92 76 12 53 42 02 75 |
| 2040        | TMC541 | 32 43 01 54 42 03 93 00 01 42 0A 32 92 76 13 42 03 |

TABLE VII

| BCD ADDRESS | CHIP   | PROGRAM CODES→                                     |
|-------------|--------|--|
| 2057        | TMC501 | 92 76 14 42 08 92 76 15 53 24 43 02 32 43 01 77 50 |
| 2074        | TMC501 | 42 04 85 43 03 54 42 01 42 05 36 00 14 42 07 53 43 |
| 2091        | TMC501 | 04 42 06 36 00 16 29 67 70 65 48 07 54 77 15 76 80 |
| 2108        | TMC501 | 53 53 43 04 85 43 05 54 55 02 54 42 06 43 08 32 53 |
| 2125        | TMC501 | 43 05 75 43 04 54 22 77 70 53 43 06 36 00 14 65 43 |
| 2142        | TMC501 | 07 54 29 67 70 77 45 43 06 42 05 61 80 76 45 03 06 |
| 2159        | TMC501 | 42 04 61 80 76 70 43 06 92 76 50 00 35 92 76 15 53 |
| 2176        | TMC501 | 43 01 85 43 05 65 43 03 54 92 76 11 42 01 92 76 12 |
| 2193        | TMC501 | 42 02 92 76 13 53 50 42 05 55 02 54 22 59 29 22 67 |
| 2210        | TMC501 | 52 53 43 05 35 65 53 43 02 75 43 01 54 54 42 03 92 |
| 2227        | TMC501 | 76 52 00 35 92 76 14 15 36 00 16 42 04 76 50 01 22 |
| 2244        | TMC501 | 44 05 53 15 36 00 16 65 04 54 44 04 22 97 05 45 53 |
| 2261        | TMC501 | 15 36 00 16 65 02 54 44 04 61 50 76 45 15 36 00 16 |
| 2278        | TMC501 | 44 04 53 43 03 55 03 54 49 04 43 04 92 76 11 53 50 |
| 2295        | TMC501 | 42 05 55 02 54 42 02 22 59 29 22 67 52 43 05 99 92 |
| 2312        | TMC501 | 76 52 00 35 92 76 12 42 03 99 92 76 13 53 29 85 32 |
| 2329        | TMC501 | 06 54 42 01 32 98 92 76 50 72 01 32 01 44 01 32 99 |
| 2346        | TMC501 | 92 61 50 76 14 53 43 05 85 06 54 42 01 73 01 42 04 |
| 2363        | TMC501 | 76 45 01 22 44 01 53 73 01 65 04 54 44 04 01 22 44 |
| 2380        | TMC501 | 01 22 97 02 33 53 73 01 65 02 54 44 04 61 45 76 33 |
| 2397        | TMC501 | 73 01 44 04 53 43 03 55 03 54 49 04 43 04 98 99 92 |
| 2414        | TMC501 | 76 16 53 53 43 06 33 85 43 01 33 75 43 02 33 54 55 |
| 2431        | TMC501 | 02 55 43 06 55 43 01 54 22 39 87 00 97 42 05 86 00 |
| 2448        | TMC501 | 76 57 43 06 48 01 48 02 42 06 61 16 76 97 87 01 68 |
| 2465        | TMC501 | 42 04 86 01 61 57 76 68 42 03 87 02 96 76 10 22 86 |
| 2482        | TMC501 | 00 22 86 01 22 86 02 22 86 03 92 76 14 53 53 03 02 |
| 2499        | TMC501 | 38 65 43 01 55 43 06 54 22 52 52 22 52 22 38 52 22 |
| 2516        | TMC501 | 52 42 04 94 85 01 94 22 39 75 43 02 54 42 02 42 05 |
| 2533        | TMC501 | 86 03 76 15 53 43 06 33 85 43 01 33 75 02 65 43 06 |
| 2550        | TMC501 | 65 43 01 65 43 02 39 54 34 87 03 88 42 02 86 00 86 |
| 2567        | TMC501 | 02 61 57 76 90 42 05 43 01 61 10 76 88 42 02 61 10 |
| 2584        | TMC501 | 76 17 43 04 92 76 18 43 05 92 76 11 42 06 92 76 12 |
| 2601        | TMC501 | 42 01 92 76 13 42 02 92 76 11 42 07 92 76 12 42 04 |
| 2618        | TMC501 | 92 76 13 42 05 92 76 16 53 01 94 22 39 75 43 04 75 |
| 2635        | TMC501 | 43 05 54 42 03 61 68 76 17 53 01 94 22 39 75 43 04 |
| 2652        | TMC501 | 42 03 75 43 05 54 42 04 86 00 76 88 53 43 07 65 43 |
| 2669        | TMC501 | 04 38 55 43 03 38 54 42 01 53 43 07 65 43 05 38 55 |
| 2686        | TMC501 | 43 03 38 54 42 02 87 00 69 43 03 92 76 69 43 04 22 |
| 2703        | TMC501 | 86 00 92 76 14 43 01 92 76 15 43 02 92 76 18 53 53 |
| 2720        | TMC501 | 43 07 85 43 01 45 43 02 54 55 02 54 42 06 53 03 06 |
| 2737        | TMC501 | 65 53 43 06 75 43 07 54 65 53 43 06 75 43 01 54 65 |
| 2754        | TMC501 | 53 43 06 75 43 02 54 54 34 92 76 11 22 86 00 22 86 |
| 2771        | TMC501 | 01 29 42 01 38 77 48 23 76 48 43 01 92 76 12 22 86 |
| 2788        | TMC501 | 00 22 86 01 29 42 02 77 88 23 43 02 76 88 92 76 13 |
| 2805        | TMC501 | 42 03 92 76 14 42 04 86 00 86 01 92 76 16 87 00 57 |
| 2822        | TMC501 | 53 43 03 55 43 02 54 42 01 92 76 57 70 53 53 03 04 |
| 2839        | TMC501 | 55 02 55 43 02 54 22 38 65 02 54 42 01 92 76 17 87 |
| 2856        | TMC501 | 01 88 53 43 03 55 43 01 54 42 02 92 76 68 53 71 69 |
| 2873        | TMC501 | 35 65 43 04 54 42 02 92 76 69 70 53 53 43 01 55 02 |
| 2890        | TMC501 | 54 38 65 02 54 92 76 18 53 43 01 65 43 02 54 92 76 |
| 2907        | TMC501 | 19 71 69 65 43 02 54 92 76 15 53 43 02 33 65 01    |
| 2924        | TMC501 | 55 02 54 92 76 10 70 53 43 01 65 43 02 33 55 02 75 |
| 2941        | TMC501 | 43 02 33 55 02 65 43 01 38 54 92 76 11 22 86 01 29 |
| 2958        | TMC501 | 77 87 94 86 01 76 87 53 42 03 33 22 23 65 02 65 89 |
| 2975        | TMC501 | 54 34 35 42 01 92 76 12 93 02 03 01 06 04 01 09 49 |
| 2992        | TMC501 | 03 01 44 03 43 03 35 53 53 42 02 45 04 65 01 93 03 |
| 3009        | TMC501 | 03 00 02 07 04 04 02 09 75 43 02 45 03 65 01 03 08 |
| 3026        | TMC501 | 02 01 02 05 95 09 07 08 85 43 02 45 02 65 01 93 07 |
| 3043        | TMC501 | 08 01 04 07 07 09 03 07 75 43 02 65 93 03 05 06 05 |
| 3060        | TMC501 | 06 03 07 88 02 85 93 03 01 04 03 08 01 05 03 54 65 |
| 3077        | TMC501 | 43 02 65 43 01 54 87 01 69 92 76 69 53 94 85 01 54 |
| 3094        | TMC501 | 92 76 88 53 53 02 04 02 09 08 65 43 09 85 09 09 09 |
| 3111        | TMC501 | 09 01 54 55 01 09 09 00 01 07 42 07 54 53 53 53 22 |
| 3128        | TMC501 | 59 65 43 07 54 42 09 55 43 07 65 05 22 28 54 49 55 |
| 3145        | TMC501 | 05 22 28 54 92 76 13 71 88 53 24 65 53 43 11 75 43 |
| 3162        | TMC501 | 10 54 76 37 85 43 10 54 42 07 78 43 07 92 76 18 70 |
| 3179        | TMC501 | 71 88 42 08 71 88 53 53 24 65 02 65 89 54 39 65 53 |
| 3196        | TMC501 | 43 08 23 65 02 94 54 34 65 43 11 61 37 76 10 36 01 |
| 3213        | TMC501 | 71 25 92 76 15 42 09 92 76 11 42 10 92 76 12 42 11 |
| 3230        | TMC501 | 92 76 13 29 67 79 76 89 43 01 49 04 97 01 89 76 38 |
| 3247        | TMC501 | 43 04 87 01 39 92 76 14 29 67 79 19 97 02 14 61 38 |
| 3264        | TMC501 | 76 15 29 67 79 19 43 02 22 49 04 97 02 15 61 38 76 |
| 3281        | TMC501 | 79 01 42 04 61 38 76 11 32 22 86 01 01 42 04 42 03 |
| 3298        | TMC501 | 32 76 17 29 77 87 86 01 50 76 87 72 03 32 73 03 59 |
| 3315        | TMC501 | 67 88 86 01 76 88 72 03 92 76 39 00 35 43 04 02 76 |
| 3332        | TMC501 | 19 43 01 49 04 01 22 44 01 42 76 12 32 02 42 03 32 |
| 3349        | TMC501 | 17 32 43 01 77 30 00 35 92 76 30 43 02 92 76 87 43 |
| 3366        | TMC501 | 05 72 01 44 04 01 44 01 53 43 04 55 32 54 92 76 12 |
| 3383        | TMC501 | 42 05 87 01 89 01 44 03 43 03 32 43 02 77 87 06 42 |
| 3400        | TMC501 | 01 86 01 76 89 73 01 22 44 04 43 05 44 04 72 01 01 |
| 3417        | TMC501 | 44 01 43 01 32 53 43 02 85 05 54 77 77 06 42 01 76 |
| 3434        | TMC501 | 77 53 43 04 55 43 02 54 92 76 79 00 35 92 76 10 22 |
| 3451        | TMC501 | 86 01 06 42 01 00 42 03 42 04 92 76 11 29 67 79 22 |
| 3468        | TMC501 | 77 79 42 02 32 43 02 59 22 67 79 92 76 16 53 53 43 |
| 3485        | TMC501 | 09 45 43 01 75 42 12 01 54 55 43 08 54 92 76 17 53 |
| 3502        | TMC501 | 16 65 43 09 54 92 76 18 53 16 55 43 12 54 92 76 19 |
| 3519        | TMC501 | 18 53 24 65 43 09 54 92 76 10 29 58 02 92 76 21 53 |
| 3536        | TMC501 | 43 04 55 43 03 54 92 76 25 53 71 71 21 55 13 09 23 |
| 3553        | TMC501 | 54 42 01 92 76 11 67 25 42 01 92 76 22 53 01 00 00 |
| 3570        | TMC501 | 75 24 65 71 71 22 45 43 01 54 94 67 15 76 12 67 22 |



TABLE VIII

| BCD ADDRESS | CHIP   | PROGRAM CODES+                                     |
|-------------|--------|--|
| 35A7        | TMC541 | 74 15 42 02 53 24 55 01 00 00 85 42 0A 01 54 42 09 |
| 3604        | TMC541 | 43 02 92 76 23 53 43 04 55 43 09 45 43 01 54 42 03 |
| 3621        | TMC541 | 92 76 13 67 23 42 03 92 76 24 53 43 03 65 43 09 45 |
| 363A        | TMC541 | 43 01 54 42 04 92 76 14 67 24 42 04 92 76 11 41 03 |
| 3655        | TMC541 | 7A 76 12 41 02 92 76 13 61 04 43 76 14 61 05 01 76 |
| 3672        | TMC541 | 15 61 05 39 01 42 09 07 94 22 2A 32 53 43 04 55 43 |
| 36A9        | TMC541 | 03 54 42 10 87 03 01 36 87 04 01 36 53 24 55 43 01 |
| 370A        | TMC541 | 33 75 43 10 35 54 42 0A 04 09 36 1A 16 42 14 87 01 |
| 3723        | TMC541 | 00 80 53 24 65 43 09 54 53 24 75 43 10 54 42 13 53 |
| 3740        | TMC541 | 43 14 55 43 0A 75 43 01 65 43 12 55 43 09 55 43 0A |
| 3757        | TMC541 | 54 87 01 01 20 53 24 65 43 09 75 43 14 54 22 04 13 |
| 3774        | TMC541 | 43 13 84 0A 44 09 50 77 00 65 61 02 80 53 43 05 55 |
| 3791        | TMC541 | 43 03 85 42 14 43 01 40 14 75 43 10 54 42 0A 53 43 |
| 3A0A        | TMC541 | 01 45 33 55 02 85 43 14 55 43 01 54 87 03 01 44 53 |
| 3A25        | TMC541 | 43 11 85 33 55 02 85 43 14 54 22 49 0A 43 0A 44 09 |
| 3A82        | TMC541 | 36 1A 1A 02 0A 87 03 02 06 53 24 65 43 09 54 53 24 |
| 3A99        | TMC541 | 85 43 14 55 43 01 55 43 12 75 43 10 54 42 13 53 43 |
| 3A7A        | TMC541 | 06 55 43 0A 75 43 01 55 43 12 55 43 09 55 43 08 54 |
| 3A93        | TMC541 | 87 03 02 55 53 24 65 43 09 85 43 06 54 53 24 85 43 |
| 3910        | TMC541 | 14 55 43 12 55 43 09 54 22 49 13 43 13 44 0A 44 09 |
| 3927        | TMC541 | 50 77 01 01 53 43 0A 65 01 00 00 54 29 47 02 05 67 |
| 3944        | TMC541 | 00 25 5A 04 3A 1A 15 92 6A 43 03 42 11 42 10 43 09 |
| 3961        | TMC541 | 87 02 03 19 87 04 03 19 01 49 11 49 10 53 43 04 65 |
| 397A        | TMC541 | 43 0A 54 87 01 03 54 87 02 03 54 44 10 53 43 05 65 |
| 3995        | TMC541 | 65 43 0A 54 94 44 11 61 03 56 44 11 43 11 23 42 11 |
| 4012        | TMC541 | 43 10 23 94 44 11 43 09 23 35 49 11 43 11 47 05 81 |
| 4029        | TMC541 | 67 03 01 42 01 94 42 11 94 5A 09 6A 92 6A 87 01 04 |
| 404A        | TMC541 | 22 87 02 04 16 87 03 04 10 36 1A 19 61 04 25 3A 1A |
| 4063        | TMC541 | 1A 61 04 25 3A 1A 17 61 04 25 3A 1A 16 53 24 55 53 |
| 4080        | TMC541 | 43 04 75 43 05 55 43 12 54 54 35 67 04 46 67 03 91 |
| 4097        | TMC541 | 42 03 5A 02 6A 92 6A 87 01 04 83 87 02 04 77 87 03 |
| 4114        | TMC541 | 04 71 36 1A 19 41 04 86 3A 1A 1A 61 04 86 3A 1A 17 |
| 4131        | TMC541 | 61 04 86 36 1A 1A 53 24 65 43 03 85 43 05 55 43 12 |
| 414A        | TMC541 | 54 67 05 04 67 04 52 42 04 5A 02 6A 92 6A 36 1A 19 |
| 4165        | TMC541 | 87 04 05 21 3A 1A 1A 53 53 24 65 43 03 75 43 04 54 |
| 4182        | TMC541 | 94 65 43 12 54 67 05 42 67 05 10 42 05 58 02 6A 92 |
| 4199        | TMC541 | 76 16 8A 01 92 76 17 8A 02 92 76 1A 8A 03 92 76 19 |
| 4216        | TMC541 | 8A 04 92 76 10 22 8A 01 22 8A 02 22 8A 03 22 8A 04 |
| 4233        | TMC541 | 00 42 05 5A 09 29 92 76 7A 53 93 04 65 43 01 85 02 |
| 4250        | TMC541 | 93 03 54 59 94 85 61 77 76 79 00 35 92 76 10 53 42 |
| 4267        | TMC541 | 01 29 22 77 79 22 50 22 44 01 65 04 22 2A 58 42 03 |
| 4284        | TMC541 | 32 01 05 0A 01 77 79 03 02 32 53 53 43 01 55 01 00 |
| 4301        | TMC541 | 00 54 42 01 22 59 22 44 01 65 01 00 00 54 42 02 77 |
| 431A        | TMC541 | 79 01 03 32 43 01 77 79 53 03 06 05 65 43 03 85 43 |
| 4335        | TMC541 | 02 85 03 01 65 43 01 75 03 01 85 03 32 43 01 77 7A |
| 4352        | TMC541 | 01 22 44 03 7A 77 53 43 03 55 04 54 59 75 53 93 07 |
| 4369        | TMC541 | 05 85 53 43 03 55 01 00 00 54 59 65 93 07 05 54 59 |
| 438A        | TMC541 | 54 92 76 11 10 42 0A 00 92 76 12 10 42 05 00 92 76 |
| 4403        | TMC541 | 13 53 43 05 75 43 04 54 02 76 14 53 53 10 42 01 94 |
| 4420        | TMC541 | 55 07 54 59 65 07 85 43 01 54 92 76 13 32 01 84 04 |
| 4437        | TMC541 | 53 32 75 43 03 54 42 05 53 35 50 65 43 05 54 42 76 |
| 4454        | TMC541 | 1A 94 7A 17 32 01 22 44 02 53 02 45 43 02 85 32 54 |
| 4471        | TMC541 | 50 92 76 12 53 36 15 71 8A 65 01 00 02 03 85 01 54 |
| 448A        | TMC541 | 59 42 03 00 42 0A 92 76 14 24 43 04 92 76 1A 09 42 |
| 4505        | TMC541 | 02 05 01 02 9A 92 76 19 53 01 00 75 43 02 54 92 76 |
| 4522        | TMC541 | 11 42 09 92 76 10 87 00 45 05 61 44 7A 45 0A 7A 44 |
| 4539        | TMC541 | 42 10 73 10 92 76 13 94 76 12 53 24 85 10 54 72 10 |
| 4556        | TMC541 | 92 76 11 10 92 76 16 22 76 17 8A 00 5A 02 92 76 1A |
| 4573        | TMC541 | 42 07 92 76 19 53 35 65 43 07 54 29 3A 1A 12 92 76 |
| 4590        | TMC541 | 14 29 3A 1A 11 43 06 42 03 29 3A 1A 71 24 42 0A 92 |
| 4607        | TMC541 | 76 15 87 00 34 42 05 92 76 34 42 0A 92 76 12 53 8A |
| 4624        | TMC541 | 85 7A 15 43 01 54 53 24 85 53 24 55 50 54 24 65 05 |
| 4641        | TMC541 | 94 22 2A 54 22 8A 5A 04 92 76 13 53 24 65 61 15 7A |
| 465A        | TMC541 | 14 53 35 65 61 15 7A 11 5A 09 8A 42 01 92 76 11 53 |
| 4675        | TMC541 | 24 65 02 93 05 04 54 92 76 12 53 24 65 93 03 00 04 |
| 4692        | TMC541 | 0A 54 92 76 13 53 24 65 93 09 01 04 04 54 92 76 14 |
| 4709        | TMC541 | 53 24 65 01 93 06 00 09 03 04 04 54 92 76 15 53 24 |
| 4726        | TMC541 | 65 93 0A 0A 0A 09 07 0A 02 04 54 92 76 1A 35 11 35 |
| 4743        | TMC541 | 92 76 17 35 12 35 92 76 18 35 13 35 92 76 19 35 14 |
| 4760        | TMC541 | 35 92 76 10 35 15 35 92 76 11 53 53 24 75 03 02 54 |
| 4777        | TMC541 | 65 05 55 09 54 92 76 12 53 24 65 93 00 42 09 05 07 |
| 4794        | TMC541 | 03 05 02 09 06 54 92 76 13 53 24 65 03 93 07 0A 05 |
| 4811        | TMC541 | 04 01 01 07 0A 0A 54 92 76 14 53 24 65 02 0A 93 03 |
| 482A        | TMC541 | 04 09 05 02 03 01 03 54 92 76 15 53 24 65 93 04 05 |
| 4845        | TMC541 | 03 05 09 02 03 07 54 92 76 16 53 24 65 01 93 0A 85 |
| 4862        | TMC541 | 03 02 54 92 76 17 35 12 35 92 76 1A 35 13 35 92 76 |
| 4879        | TMC541 | 19 35 14 35 92 76 10 35 15 35 92 61 11 92 92 92 92 |
| 4896        | TMC541 | 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92    |
| 4913        | TMC541 | 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92    |
| 4930        | TMC541 | 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92    |
| 4947        | TMC541 | 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92    |
| 4964        | TMC541 | 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92    |
| 4981        | TMC541 | 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92 92    |
| 499A        | TMC541 | 92 92  |

What is claimed is:

1. In an electronic microprocessor system, having a keyboard, output means for outputting data, a data memory for storing data, an arithmetic unit for performing arithmetic operations on the data stored in said data memory, and a first nonvolatile memory for storing groups of instruction words for controlling the arithmetic operations performed by said arithmetic unit, the combination which comprises:

- (a) a second non-volatile memory for storing a plurality of sets of program codes, each program code being effective for addressing a preselected group of instruction words stored in first non-volatile memory, said second non-volatile memory being disposed in a module having a plurality of electrical contacts;
- (b) a receptacle for temporarily interconnecting the contacts on said module with said microprocessor system;
- (c) keyboard logic means for decoding inputs received at said keyboard;
- (d) means for addressing said second non-volatile memory to read out preselected sets of program codes, said means for addressing said second memory including a counter responsive to selected instructions outputted from said first memory;
- (e) means for addressing said first non-volatile memory in response to the program codes read out of said second non-volatile memory, said means for addressing said first memory including a first program counter selectively responsive to said keyboard logic means and said program codes outputted from said second memory.

2. The electronic microprocessor system according to claim 1, wherein said first and second non-volatile memory are first and second read-only-memories, respectively.

3. The electronic microprocessor system according to claim 1, further including means for loading said counter in said means for addressing said second read-only-memory with a multibit number stored in said data memory.

4. The electronic microprocessor system according to claim 3, wherein said data memory comprises a plurality of registers and a multibit register coupled to the output of said arithmetic unit.

5. The electronic microprocessor system according to claim 4, wherein said keyboard logic means includes a keyboard register, the contents of which are loaded into said first program counter in response to a particular instruction word outputted from said first read-only-memory and loaded into said counter addressing second read-only-memory in response to another instruction word outputted from said first read-only-memory.

6. The microprocessor system according to claim 5, further including means for loading said keyboard register means with the contents of said multibit register connected to the output of said arithmetic unit in response to a given instruction word outputted from said first read-only-memory, whereby said arithmetic unit, multibit register connected to the output thereof, keyboard register means, and means for loading said keyboard register means provides said means for loading said counter addressing said second read-only-memory with a multibit number stored in said data memory.

7. The electronic microprocessor system according to claim 6, further including random access memory means for storing a user defined program comprising of a selected sequence of key pushes at said keyboard, said

selected sequence of key pushes being stored as a sequence of program codes therein and means for loading said program codes stored in said random access memory means into said first program counter.

8. The electronic microprocessor system according to claim 7, wherein said electronic microprocessor system is provided in a programmable calculator.

9. The electronic microprocessor system according to claim 1, further including random access memory means for storing a user defined program comprising of a selected sequence of key pushes at said keyboard, said selected sequence of key pushes being stored as a sequence of program codes therein and means for loading said program codes stored in said random access memory means into said first program counter.

10. The electronic microprocessor system according to claim 9, further including means for loading said counter in said means for addressing said second read-only-memory with a multibit number stored in said data memory.

11. The electronic microprocessor system according to claim 10, wherein said keyboard logic means includes keyboard register means for loading said first program counter in response to a particular instruction word outputted from said first read-only-memory and for loading said counter addressing second read-only-memory in response to another instruction word outputted from said first read-only-memory.

12. The electronic microprocessor system according to claim 10, wherein said keyboard logic means includes keyboard registers, the contents of which are loaded into said first program counter in response to a particular instruction word outputted from said first read-only-memory and loaded into said counter addressing second read-only-memory in response to another instruction word outputted from said first read-only-memory.

13. The electronic microprocessor system according to claim 1, wherein said keyboard logic means includes keyboard register, the contents of which are loaded into said first program counter in response to a particular instruction word outputted from said first read-only-memory and loaded into said counter addressing second read-only-memory in response to another instruction word outputted from said first read-only-memory.

14. The electronic microprocessor system according to claim 13, further including random access memory means for storing a user defined program comprising of a selected sequence of key pushes at said keyboard, said selected sequence of key pushes being stored as a sequence of program codes therein and means for loading said program codes stored in said random access memory means into said first program counter.

15. The electronic microprocessor system according to claim 1, wherein a plurality of said second non-volatile memories are provided in a plurality of modules, each one of said plurality of second-non-volatile memories storing different sequences of program codes, at least one of said modules including the second non-volatile memory disposed therein, being receivable in said receptacle at any given time.

16. The electronic microprocessor system according to claim 15, wherein said electronic microprocessor system is provided in a programmable calculator.

17. The programmable calculator according to claim 16, wherein said calculator includes a case in which said data memory, arithmetic unit receptacle and first non-volatile memory are disposed, said case having an opening adjacent to said receptacle for temporarily receiving at least a selected one of said modules.

18. The programmable calculator according to claim 17, wherein said first and second non-volatile memory are first and second read-only-memories, respectively.

19. The programmable calculator according to claim 17, further including a magnetic card reader for reading magnetic cards providing storage for programs including a plurality of program codes, a program memory for storing program codes read by said magnetic card reader, means for reading out of said program memory said program codes and means for addressing said first non-volatile memory in response to the program codes read out of said program memory.

20. The programmable calculator according to claim 19, wherein said input means includes a keyboard and wherein said calculator may be selectively placed in a learn mode or run mode at said keyboard and wherein said calculator further includes means for storing the program codes according to the keys depressed at said keyboard in said program memory when said calculator is in said learn mode.

21. An electronic microprocessor system, having input means for receiving data and for receiving input commands, said input means including a keyboard, output means for outputting data, a data memory for storing data received and data to be outputted, an arithmetic unit for performing arithmetic operations on data stored in said data memory, and a first non-volatile memory for storing groups of instruction words for controlling the arithmetic operations performed by said arithmetic unit, the combination which comprises:

- (a) a second non-volatile memory for storing a plurality of sets of program codes, each program code being effective for addressing a preselected group of instruction words stored in said first non-volatile memory;
- (b) keyboard logic means for decoding inputs received at said keyboard;
- (c) means for addressing said second non-volatile memory to read-out preselected sets of program codes in response thereto, said means for addressing said second memory including a counter responsive to selected instructions outputted from said first memory;
- (d) means for addressing said first non-volatile memory in response to the program codes read-out of said second non-volatile memory, said means for addressing said first memory including a first program counter selectively responsive to said keyboard logic and to said program codes outputted from said second memory;
- (e) magnetic card reader means for receiving program codes stored on magnetic cards;
- (f) program memory means for storing program codes received by said magnetic card reader means;
- (g) means for addressing said first non-volatile memory in response to the program codes read-out of said program memory means; and
- (h) wherein said means for addressing said first memory is further responsive to the program codes outputted from said program memory.

22. The system according to claim 21, wherein said first and second non-volatile memories are first and second read-only-memories respectively.

23. In an electronic microprocessor system of the type having a keyboard, an arithmetic unit for performing numerical operations on data, a first memory for storing a plurality of groups of instruction words, a first address register for addressing the first memory, key-

board logic means for inserting addresses into said first address register in response to key depressions at said keyboard and instruction word decoder means for controlling the arithmetic unit in response to instruction words outputted from the first memory, the combination which comprises:

- (a) a second memory for storing a plurality of sets of program codes, each program code being effective for addressing a preselected group of instruction words stored in the first memory, the second memory being disposed in a module having a plurality of electrical contacts;
- (b) a receptacle for temporarily interconnecting the contents on said module with said microprocessor system;
- (c) a second address register for addressing the second memory;
- (d) means for inserting a preselected address into said second address register in response to the depression of a "program" key at said keyboard;
- (e) means for comparing the program code stored at said preselected address with a numerical value inputted at said keyboard after the depression of said "program" key; and
- (f) means for displaying an error condition when the results of the comparison indicate that the numerical value inputted after the depression of the "program" key is greater than the numerical value of the program code stored at the preselected address.

24. The system as defined in claim 23, wherein said first and second memories are first and second read-only-memories.

25. An integrated circuit disposed in a module, the module being temporarily receivable in a receptacle of a calculator and the calculator including an instruction word memory for controlling the operations performed by the calculator, the integrated circuit comprising:

- (a) a program memory for storing a plurality of program codes, each program code having two four-bit digits and being adapted for use by said calculator to address said instruction word memory;
- (b) an address register for addressing the program memory;
- (c) an instruction decoder responsive to bit serial instructions generated by said calculator;
- (d) first means coupled to said program memory for outputting the four most significant bits of the addressed program code in serial to said calculator in response to said instruction decoder decoding a "FETCH HIGH" instruction from said calculator; and
- (e) second means coupled to said program memory for outputting the four least significant bits of the addressed program code in serial to said calculator in response to said instruction decoder decoding a "FETCH" instruction from said calculator.

26. The system according to claim 25, further including means for incrementing the address in said address register in response to a decoded "FETCH" instruction.

27. The system according to claim 26, wherein said address register stores a four digit address and further including means for outputting to said calculator in serial the contents of a sequentially different digit position of said address register each time said instruction decoder decodes an "UNLOAD PC" instruction from said calculator.

\* \* \* \* \*