The Rockwell 900 series programmable calculators...

...Machines for people.
People for productivity.
Productivity for profit.

Advanced Operation Instruction

Rockwell

ADVANCED OPERATION INSTRUCTION MANUAL

INTRODUCTION

This manual is intended to be used as a technical supplement for the 900 Series Basic Operation Instruction Manual. The examples provided illustrate the more complex keyboard operations and demonstrate additional applications.

An explanation of those keys associated with programming and their use in elementary programming along with a completely documented program is included.

This book has been written to help an experienced programmer to write programs on a Rockwell 900 Series calculator.

If you have limited experience with programmable calculators, you can learn to program by using this book in conjunction with the Rockwell 900 Series Programming Guide. Ask your salesman how you can obtain a copy.

You may write some programs that you would like to share with us. If so, we would be happy to hear from you. Please mail any such programs to:

Rockwell International
950 DeGuigne Drive
Sunnyvale, California 94086

or

Sumlock Anita/Rockwell International
Anita House, Rockingham Road
Uxbridge, Middlesex, England
UB8 2XL

Form 511

TABLE OF CONTENTS

		PAGE NO.
I.	Advanced Calculations	1
	Constant Dividend	1
	Percentage Distribution	2
	Proration	3
	Percentage Increase and Decrease	4
	Accumulation of Results, First Factors and Second Factors With an Item Count	5
	Using [+M/IN] and [+M/OUT] to Store Constants	6
	Adding to and Subtracting From Memory by Using [+M/IN] and [+M/OUT]	7
	Square Roots	8
	Roots and Powers	10
	Chain Calculations Using x^y	11
	Using x^y to Compute High Powers	12
	Natural Logarithms and Antilogarithms	14
II.	Explanation of Programming Key Functions	17
	Programming Keys	17
	Jumps	20
	Subroutines	24
	The Indirect Memory System	26
	Summary	28
III.	The Six Operating Modes	29
	Manual Mode	29
	Program Mode	30
	Edit Mode	31
	List Mode	32
	Run Mode	33
	Trace Mode	33

TABLE OF CONTENTS (cont.)

	PAGE. NO.
IV. Sample Program	34
Introduction	34
Payroll Program Part 1	36
Program Description and Formulas	36
Operating Instructions	36
Sample Printout	38
Flowchart	41
Program Construction	42
Program Coding	43
Payroll Program Part 2	46
Program Description and Formulas	46
Operating Instructions	46
Sample Printout	48
Flowchart	49
Program Construction	52
Program Coding	53
V. Error Conditions	56

I. ADVANCED CALCULATIONS

This section provides advanced keyboard applications of the 900 Series Programmable Calculator. For basic examples refer to the Basic Operation Instruction Manual provided with your calculator.

CONSTANT DIVIDEND

SETTINGS: Decimal @2, RO, Calc Mode

PROBLEM	KEYBOARD ENTRY		PRINT OUT
		*	0.00 * 1
1368 ÷ 123 = 11.12	1368	M+	1368. M1
		÷	1368. ÷
	123	=	123. =
			11.12
1368 ÷ 456 = 3.00		◇	1368. ◇ 1
		÷	1368. ÷
	456	=	456. =
			3.00
1368 ÷ 789 = 173		*	1368.00 * 1
		÷	1368.00 ÷
	789	=	789. =
			1.73

The digit 1, shown to the right of the printout, indicates that memory 1 is being used. When the unit is turned on, memory 1 is automatically selected until another memory is selected.

PERCENTAGE DISTRIBUTION
(With a Constant Divisor)

SETTINGS: Decimal 02, RO, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINT OUT
	T	0.00 T
123 = 8.99%	123 ÷	123. ÷
	+	123.00 +
	456 +	456.00 +
	789 +	789.00 +
	%	1368. %T
		8.99
456 = 33.33%	456 %	456. %
		33.33
789 = 57.68%	789 %	789. %
1368 100.00%		57.68
	AX	100.00 F 1
	AX	1368. F 2
	N	3. N 2

When entering the first number, you must depress [÷] [+] in that order, not [+] [÷], to avoid totaling the first number from the adding machine before adding in the next two numbers.

Depressing [%] after the last [+] will total the adding machine, print the total and make it a constant divisor. It also completes the problem begun when [÷] was depressed after the first entry was made.

PRORATION
(With a Constant Multiplicand)

SETTINGS: Decimal @2, RO, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINT OUT
	975 x	975. x
975 x 5% = 48.75	5 %	5. % 48.75
975 x 22% = 214.50	22 %	22. % 214.50
975 x 33% = 321.75	33 %	33. % 321.75
975 x 40% = 390.00 100% 975.00	40 %	40. % 390.00
	AX	975.00 F 1
	AX	100. F 2
	N	4. N 2

When the percent key is used with a constant, all entries and results are automatically accumulated. Depressing [AX] once will print the sum of the results. Depressing [AX] twice in succession will print the sum of the percentages.

Also, the accumulation counter will automatically keep count of the number of entries made with the constant. Depressing [N] will print the count.

These accumulations are automatically cleared when a new problem is begun.

PERCENTAGE INCREASE AND DECREASE

SETTINGS: Decimal @2, RO, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINT OUT
	CLEAR	0.C
	T	0.00 T
1. Increase	1500 +	1500.00 +
Jan: 1250	1250 ÷	1250. ÷
Feb: 1500	-	1250.00 -
1500 - 1250 = 20%	EXCHANGE	250. E×T
1250	%	1250. %
		20.00
2. Decrease	1250 +	1250.00 +
Jan: 1500	1500 ÷	1500. ÷
Feb: 1250	-	1500.00 -
1250 - 1500 = -16.67%	EXCHANGE	250. E×T
1500	%	1500. %
		16.67

Depressing the [EXCHANGE] key immediately after depressing [+] or [-] will total and print the contents of the adding machine then exchange the total with the previously entered number. The problem a ÷ b will become b ÷ a.

ACCUMULATION OF RESULTS, FIRST FACTORS AND SECOND FACTORS WITH AN ITEM COUNT

SETTINGS: Decimal @2, RO, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINT OUT
	CLEAR	0.C
	*	0.00 * 1
		2. ×
2 x 3 = 6	2 X	
	3 =+	3. =
		6.00 M1
		4. ×
4 x 5 = 20	4 X	
	5 =+	5. =
		20.00 M1
		6. ×
6 x 7 = 42	6 X	
	7 =+	7. =
		42.00 M1
TOTALS: 12 15 68 Sum of results	*	68.00 * 1
Sum of 1st factors	AX	12. F 1
Sum of 2nd factors	AX	15. F 2
No. of accumulations	N	3. N 2

Each time [=+] is depressed, four accumulations occur simultaneously:

To see the sum of the first factors, depress [AX] once. To see the sum of the second factors, depress [AX] twice in succession. You can see the count of the accumulations by depressing [N] at any time as long as [+], [-], [S], [T] or [%] haven't been depressed since the last time [M+], [M-], [=+], [=-], [◊] or [*] was depressed. Just remember that you see the count of additions after derpessing an adding key and you see the count of accumulations after depressing a memory key. Other keys do not change the counter being addressed.

The last three accumulations occur in registers which are separate from the add register and the memories.

USING [→M/IN] AND [←M/OUT] TO STORE CONSTANTS

SETTINGS: Decimal @2, RO, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINT OUT
	[SELECT M] 4	SE 4
25 x 20 = 500	20 [→M/IN] 02	20. -M 2
25 x 30 = 750	30 [→M/IN] 03	30. -M 3
	25 x	25. ×
	[←M/OUT] 02	20. -M 2
	=+	20. =
		500.00 M4
	[←M/OUT] 03	30. -M 3
	=+	30. =
		750.00 M4
40 x 20 = 800	40 x	40. ×
40 x 30 = 1200	[←M/OUT] 02	20. -M 2
	=+	20. =
		800.00 M4
	[←M/OUT] 03	30. -M 3
	=+	30. =
		1200.00 M4
15 x 20 = 300	15 x	15. ×
15 x 30 = 450	[←M/OUT] 02	20. -M 2
TOTAL: 4000	=+	20. =
		300.00 M4
	[←M/OUT] 03	30. -M 3
	=+	30. =
		450.00 M4
		4000.00 * 4
	*	

To store a number in a memory and eliminate the value previously stored, enter the number and depress [→M/IN], then depress the two digits corresponding to the memory you have chosen. For memories 1 - 9, depress 01, 02...09. For memories above 9, depress 10, 11, 12, ...

ADDING TO AND SUBTRACTING FROM MEMORY BY USING [→M/IN] and [←M/OUT]

SETTINGS: Decimal @2, RO, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINT OUT
Add the following column in memory two by using [→M/IN]	CLEAR ALL	C A
+1.23	1.23 [→M/IN] + 02	1.23 - M + 2
+4.56	4.56 [→M/IN] + 02	4.56 - M + 2
-0.12	0.12 [→M/IN] - 02	0.12 - M - 2
+7.89	7.89 [→M/IN] + 02	7.89 - M + 2
13.56	[←M/OUT] 02	13.56 - M 2
Complete the following problems and add the answers to memory 7 by using [→M/IN]		
2 x 33 = 66	2 x	2. x
	33 =	33. =
		66.00
	[→M/IN] + 07	66.00 - M + 7
8 x 20 = 160	8 x	8. x
	20 =	20. =
		160.00
	[→M/IN] + 07	160.00 - M + 7
	[←M/OUT] 07	226.00 - M 7

Adding to memory by using the [→M/IN] key takes a few more key depressions than adding by using [M+] and [M-]. However, [→M/IN] allows you to access any memory you like while not changing the memory being addressed by [M+], [M-], [=+], [=-], [◊] and [*].

Using [→M/IN] and [←M/OUT] is the only way to access memories above memory 9.

SQUARE ROOTS

SETTINGS: Decimal @4, RO, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINT OUT
1. $\sqrt{25} = 5$	25 $\sqrt{}$	25. r 5.
2. $\sqrt{3 \times 4 \times 5} =$ 7.7459666924148	3 x 4 x 5 =	3. x 4.C x 5. = 60.0000
	$\sqrt{}$	60.0000 r 7.7459666924148
3. $2 \times \sqrt{3} \times 4 = 13.8564$	2 x 3 $\sqrt{}$	2. x 3. r 1.7320508075689
	x 4 =	1.7320508075689C x 4. = 13.8564
4. $\sqrt{2 \times 3} \times 4 = 9.7980$	2 x 3 =	2. x 3. = 6.0000
	$\sqrt{}$	6.0000 r 2.4494897427832
	x 4 =	2.4494897427832 x 4. = 9.7980
5. $2 \times 3 \times \sqrt{3} \times 4 = 41.5692$	2 x 3 x $\sqrt{}$	2. x 3.C x 3. r 1.7320508075689
	x 4 =	1.7320508075689C x 4. = 41.5692

The square root of a number will always print with a floating decimal, regardless of the decimal setting. This guarantees that you will get the greatest possible accuracy in your work.

The calculator will normally take the square root of an entry. If no entry has been made, it will take the square root of the last number printed on the tape.

The square root function can be used in chain calculations whenever desired.

ROOTS AND POWERS

SETTINGS: Decimal @2, FL, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINTOUT
	CLEAR ALL	C A
1. $2^8 = 256$	2 x^y	2. x^y
	8 =	8. =
		256.
2. $(1.13975)^{-20} = 0.0730815897$	1.13975 x^y	1.13975 x^y
	20 CHANGE SIGN =	20. =
		0.07308158917
3. $(5.31)^{3.25} = 227.2778924$	5.31 x^y	5.31 x^y
	3.25 =	3.25 =
		227.2778924
4. $64^{1/3} = 4$	64 x^y	64. x^y
	3 $\frac{1}{x}$	3. $\frac{1}{x}$
		0.33333333333
	=	0.33333333333 =
		4.

The calculator will compute roots and powers for any real numbers x and y, x>0. Use [CHANGE SIGN] for negative exponents and use [1/x] for roots.

Notice that [X^Y] has been designed so that it will only affect the number in the keyboard register (the entry or last answer). [X^Y] can be used in chain calculations, but it will not complete a chain. This provides an easy to understand system. Please see the next problem for a complete explanation of this system.

CHAIN CALCULATIONS USING x^y

SETTINGS: Decimal @4, FL, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINT OUT
1. $2 \times (3)^4 = 162$	2 x_y 3 x^y 4 =	2. × 3. x^y 4. = 162.
2. $2 \times (3)^4 \times 5 = 810$	2 x_y 3 x^y 4 × 5 =	2. × 3. x^y 4.C × 5. = 810.
3. $(2 \times 3)^4 = 1296$	2 × 3 =	2. × 3. = 6.
	x^y 4 =	6. x^y 4. = 1296.
4. $5 \times (2x3)^4 \times 6 = 38880$	2 × 3 =	2. × 3. = 6.
	x^y 4 × 5 × 6 =	6. x^y 4. × 5.C × 6. = 38880.

$[x^y]$ will operate on the number in the keyboard register (the most recent entry or result). $[x^y]$ will not complete a chain calculation. The function depressed immediately after $[x^y]$ will compute both the power function and any pre-existing chain. This is a very versatile system, as can be seen by studying the above examples.

In problem #4 above, the order of entry was switched so that the power function was performed before the multiplications. This was done to prevent the calculator from multiplying $5 \times 2 \times (3)^4 \times 6 =$.

USING x^y TO COMPUTE HIGH POWERS

SETTINGS: Decimal @4, FL, Calc Mode

PROBLEM	KEYBOARD ENTRY	PRINT OUT
123456789^2 = 1.524 ... X 10^{16}	123456789 x^y	123456789. x^y
	2 =	2. =
		16.183029954337 E
	CLEAR	
	+	16.183029954337 +
	16 -	16.000000000000 -
	10 x^y	10. x^y
	T	0.183029954337 T
	=	0.183029954337 =
		1.524157875
6668901^{713} = 3.553 ... X 10^{4865}	6668901 x^y	6668901. x^y
	713 =	713. =
		4865.550694761 3 E
	CLEAR	
	+	4865.550694761 3 +
	4865 -	4865.000000000 -
	10 x^y	10. x^y
	T	0.550694761 3 T
	=	0.550694761 3 =
		3.553814552

The calculator has an excellent system for calculating very large numbers with the x^y function. When [x^y] is used to calculate a result larger than 14 whole numbers, the machine will overflow and print 'E'. However, it will also print the common (base 10) log of the answer. This feature gives your calculator an effective capacity of $10^{10^{14}}$, or ($10^{100000000000000}$).

If you wish to use the log in further calculations, simply depress [CLEAR] once to unlock the keyboard. The log is now available for use. Just depress the proper operating key.

It is easy to convert the logrithmic result of your original problem to scientific notation, as is shown in the above examples. The procedure is as follows: After the log prints, depress CLEAR once to unlock the keyboard. Next, touch [+]. The whole number portion of the log is the exponent of 10 of the scientific notation. Enter this into the keyboard and depress [-]. Now enter the number 10 and depress [x^y] [T] [=]. Combine the final number that prints with the exponent of 10 found earlier to obtain the result of the original problem in scientific notation.

NATURAL LOGARITHMS AND ANTILOGARITHMS

SETTINGS: Decimal #2, FL, Calc Mode

PROBLEM	KEYBOARD ENTRY		PRINTOUT
Problem 1:			
Find the natural log of 25, 1069.328 and $(4.37)^{1/3.2}$	25	ln x	25.LN 3.218875825
	1069.328	ln x	1069.328LN 6.974785693
	4.37	ln x	4.37LN 1.474763009
		x	1.474763009 x
	3.2	1/x	3.2 1/x 0.3125
		=	0.3125 = 0.460863440312 5
Problem 2:			
Find the natural antilog of 3.789 and -0.00159	3.789	e^x	3.789 e^x 44.212166
	.00159	CHANGE SIGN e^x	0.00159 e^x 0.9984112634

continued on next page

SETTINGS: Decimal @2, FL, Calc Mode

PROBLEM	KEYBOARD ENTRY		PRINT OUT
Problem 3:			
Solve the following fraction using natural logs		CLEAR	0.C
		*	0. * 4
$\frac{(87654)^{19.5}\ (993366)}{(123654)^{20}}$ =	87654	ln x	87654.LN 11.38115253
		x	11.38115253 x
	19.5	=+	19.5= 221.93247335 M4
	993366	ln x	993366.LN 13.80885446
		M+	13.80885446 M4
	123654	ln x	123654.LN 11.72524262
		x	11.72524262 x
	20	=-	20.= 234.50485200 M4
3.443458675		*	1.236476395 * 4
		e^x	1.236476395 e^x 3.443458675

The last problem involves numbers that are far too large to be calculated with a 14-digit capacity calculator. By using natural logs, the problem was easy to solve. The calculator will compute e^x for any real number x. It will compute ln x for any real number x, x>0.

continued on next page

[ln X] and [e^X] will operate on the number in the keyboard register (the last entry or result). They will not complete a chain calculation, but they can be used as desired in a chain calculation.

If you try to raise e to a power larger than 32.23619 the calculator will overflow and the keyboard will lock up. The number that prints will be the common (base 10) log of the answer.

You can easily convert this log into the actual answer to your problem, expressed in scientific notation.

II. EXPLANATION OF PROGRAMMING KEY FUNCTIONS

PROGRAMMING KEYS

The following is an explanation of keys and functions used primarily or exclusively for programming.

[STOP] is a program instruction used to stop a program so that the operator can enter a new variable or perform manual calculations. When [RUN] is touched, the calculator will read and execute each program step until it reaches a [STOP] instruction. It will then change from Run Mode to Calculator Mode, and will not read any further instructions until [RUN] is touched again.

[PRINT] is used in programs to print entries and results. [PRINT] will print the contents of the keyboard register. (The number in the keyboard register is generally the last printed number.)

If a number has just been entered into the keyboard either manually or as a programmed constant and no operating key has been touched, a lower case 'e' will print on the tape to show that an entry has been made.

In all other cases, [PRINT] will print the contents of the keyboard register with no symbol next to the number.

Each depression of [SPACE] will advance the paper tape one line. Nothing will print. This is very useful when formatting the output of a program.

[INDIR] is a great step saver. Please see the explanation of the Indirect Memory System (page 26) for further information.

The decimal set key is programmable. If, for example, you program the instruction [DEC SET] 5, the calculator will operate as if you had manually set the decimal to five and conform to the setting of the RO/TR/FL switch. The calculator will remain at a decimal setting of five until you either program a new setting or manually change it.

1. This instruction takes two key depressions and two program steps.

2. It is possible to program a floating decimal. In Run Mode, if a [DEC SET] [.] sequence has been programmed, it will cause the calculator to ignore the RO/TR/FL switch and to operate with a full floating decimal. A [DEC SET] [Digit] will reset this special floating condition and the calculator will obey the RO/TR/FL switch.

   The [DEC SET] [.] condition is also reset whenever the calculator encounters a [STOP] instruction. This allows you to perform manual calculations according to the setting of the RO/TR/FL switch. To put your program back into float after the [STOP] instruction, simply re-program [DEC SET] [.].

[LABEL] followed by any two digits will establish an entry point in program storage at which a jump or subroutine can be executed. Labels may be used with jumps in the same manner as direct addresses.

A label is merely a marker in a program and it will have no affect on any calculations under any circumstances.

[LABEL] must be used with [GO SUB] as subroutines must begin with a label. [GO SUB] cannot jump to a step address.

JUMPS

The Rockwell 900 Series can jump to specific step addresses or to labels. Programmers may use either or both types of commands at their discretion. The format for both types of jumps is similar.

To jump to a specific step, the sequence is [JUMP] followed by the three digit address of the step. For example, to jump to step 369, the sequence is [JUMP] 369. Jumping to an absolute address is extremely fast internally; the program pointer is simply changed and the program begins to run from the new step.

The calculator can also jump to a label. A label can be any two digit number from 00 to 99 so it is practically impossible to run out of labels. A jump instruction, such as [JUMP] [LABEL] 04 will send the program to label 04. You can put a label anywhere you choose in the program, either before or after the jump that goes to that label. You can have many different jumps all going to the same label, but obviously you cannot use the same label in more than one place in a program.

When writing a program, you may use either absolute addressing or labels. They each have their benefits. If you have no preference, we recommend that you use labels when writing and debugging programs.

The following points can help you decide which system to use:

1. Labels are much more convenient to use than absolute step addresses. If you use labels, you will not have to number each step when writing a program. Even more important, if you use labels, debugging a program becomes much easier. You will not have to change the jump instructions every time you insert or delete steps.

2. Jumping to labels uses more steps than jumping to step addresses. For example, [JUMP] digit digit digit is four steps, and [JUMP] [LABEL] digit digit plus [LABEL] digit digit is seven steps. Please note that this difference is often insignificant. There is usually no reason to shorten most programs unless you run out of steps.

3. Switching from labels to step addresses can significantly speed up lengthly loops and subroutines. You may find it convenient to use absolute addressing when jumping to or within a loop, and to use labels for those jumps where the difference in speed will not be noticed.

The calculator can make five separate types of jumps, four conditional and one unconditional. The four conditional jumps all test the keyboard register. Depending on your requirements, the calculator can test the keyboard for the following conditions: positive, negative, zero and "no entry". The five jumps operate as follows:

1. Jump unconditionally. When the calculator encounters an unconditional jump, it will jump under all circumstances. An unconditional jump requires four steps. The code sequence [JUMP] digit, digit, digit, will go to a specific step address, while [JUMP] [LABEL] digit, digit will jump to the program step immediately following the specified label.

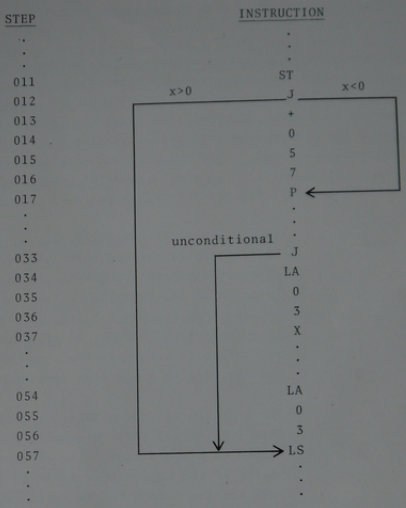
2. Jump If Positive. When the calculator encounters [JUMP] [+] in a program, it will jump if the number in the keyboard register is positive. If the number in the keyboard register is negative or zero, the jump will be ignored. For instance [JUMP] [+] 3 6 9 will jump to step 369 if the number in the keyboard register is positive. [JUMP] [+] [LABEL] 4 4 4 will jump to the first step following label 44 if the number in the keyboard register is positive.

3. Jump If Negative. When the calculator encounters [JUMP] [-], it will jump if the number in the keyboard register is negative. If the number in the keyboard is positive or zero, the negative jump will be ignored. For instance [JUMP] [-] 4 2 8 will jump to step 428 if the number in the keyboard register is negative. [JUMP] [-] [LABEL] 2 2 will jump to the first step following label 22 under the same condition.

4. Jump If Zero. When the calculator encounters [JUMP] [=] in a program, it will jump if the number in the keyboard is zero. If the number in the keyboard is greater than or less than zero, the jump will be ignored. For instance [JUMP] [=] 2 3 8 will jump to step 238 if the number in the keyboard register is zero. [JUMP] [=] [LABEL] 1 1 will jump to the first step following label 11 if the number in the keyboard is zero.

5. Jump If No Entry. When the calculator encounters a [JUMP] [AX] in a program, it will jump if no entry has been made. If an entry was made, positive, negative or zero, the calculator will ignore the jump command and proceed to the next step. [JUMP] [AX] 4 3 6 will jump to step 436. [JUMP] [AX] [LABEL] 5 5 will jump to the first step following label 55. Please note that zero is a perfectly valid entry; so entry of 0 will cause the calculator to ignore a Jump If No Entry instruction.

The following page graphically illustrates a conditional jump to an absolute address and an unconditional jump to a label.


STEP
INSTRUCTION
011
012
013
014
015
016
017
033
034
035
036
037
054
055
056
057
ST
J
+
0
5
7
P
J
LA
0
3
X
LA
0
3
LS
x>0
x<0
unconditional

SUBROUTINES

A subroutine is a miniature program that performs a set of instructions and is used by another program. Subroutines are accessed with [GO SUB]. [GO SUB] operates like [JUMP] with one exception, a return address is saved. [GO SUB] also takes the program from step A to step B but the program will calculate from step B until a [RETURN] instruction is read. [RETURN] will automatically send the program back to step (A + 1). This is a powerful programming technique. [GO SUB] and [RETURN] are defined as follows:

[GO SUB] directs the calculator to a subroutine (a sequence of program steps which may be used many times in different programs, yet requires only one location in the memory). After branching to a subroutine, the program will proceed normally until it encounters a [RETURN] instruction. It will then automatically return to the step immediately following the program steps that accessed the subroutine.

[GO SUB] may be used either unconditionally or with [+], [-] [=] and [AX] to execute a conditional branch to a subroutine. The keyboard register is tested in the same manner as it is for a conditional jump and the branch to the subroutine is executed (or not executed) based on the results.

[RETURN] is an instruction required in conjunction with the [GO SUB] key. All subroutines called by the [GO SUB] key must terminate with a [RETURN] instruction. This will return the program to the step following the program steps that accessed the subroutine.

Subroutines can be 'nested' five levels deep. 'Nesting' is defined as the ability of one subroutine to automatically branch to a second subroutine, which in turn can branch to a third, to a fourth and then to a fifth subroutine. From the fifth subroutine, the program will automatically go back to the fourth, the third, the second, then the first subroutine, and then back to the main program. Return addresses for these 'nested' subroutines are executed on a last-in-first-out basis with the return always being made to the return address most recently stored. As soon as any return is made, that return address is erased from the memory with the previous address now becoming the last one.

[GO SUB] 4 4, when executed manually, will cause the machine to search for label 44 and to start calculating from that point. The first [RETURN] that is encountered will result in a return to Manual Mode and step 000. When [GO SUB] is used manually, [RETURN] will <u>always</u> take the program to step 000, regardless of the step that was being addressed when [GO SUB] was touched.

THE INDIRECT MEMORY SYSTEM

The indirect memory system is a fast, easy way to recall (or store) numbers from (or into) as many memories as is desired. The system is very easy to learn and can save many program steps. Basically, it uses a special memory called the 'Pointer Register' to tell a program which memory to address.

In a loop, the indirect memory system can be used to store and recall numbers from many different memories by repeating the same program instructions. This greatly shortens and simplifies many programs. It occasionally makes it possible to write short routines to solve problems that would be absolutely impossible without the indirect memory system.

The Pointer Register, which is also called memory 00, is a two digit memory which retains any number between zero and ninety-nine (0,1,2,3,...98,99). Memory 00 holds positive whole numbers only so that it will hold 29 but not 2.9 or -29. To clear it, store zero in it (touch 0 [→M/IN] 00).

The Pointer Register does not store data. It stores the number of another memory that holds data. This other memory is the one that will be addressed when the indirect key ([INDIR]), is touched. Memory 00 is used to point at successive memories during an internal loop.

For example, if 12 is in memory 00, touching 125 [→M/IN] [INDIR] will store 125 in memory 12. If 39 is in memory 00, touching 45.63 [→M/IN] [+] [INDIR] will add 45.63 to memory 39. If 3 is in memory 00, touching 5.25 [→M/IN] [-] [INDIR] will subtract 5.25 from memory 03. If 8 is in memory 00, touching [→M/OUT]

[INDIR] will recall the contents of memory 08. If 17 is in memory 00, touching [→M/IN] [EXCHANGE] [INDIR] will bring the contents of memory 17 to the keyboard and store the number that was in the keyboard into memory 17.

To store a number in memory 00, use [→M/IN] 00 in the standard fashion. To add one to memory 00, touch [SELECT M] [+]. To subtract one from memory 00, touch [SELECT M] [-]. The instructions [SELECT M] [+] and [SELECT M] [-] will have no affect on the adding machine, the number in the keyboard, or any other part of the calculator except for memory 00. The following examples will show how memory 00 operates.

KEYBOARD ENTRY	PRINT OUT	EXPLANATION
[CLEAR ALL]	CA	Clear all registers
25 [→M/IN] 00	25.→M 0	Store 25 in memory 00
500 [→M/IN] [INDIR]	500.→M	Store 500 in memory 25
100 [→M/IN] [+] [INDIR]	100.→M+	Add 100 to memory 25
[→M/OUT] [INDIR]	600.→M	Recall memory 25
[SELECT M] [+]	600. SE+	Add 1 to memory 00*
120 [→M/IN] [INDIR]	120.→M	Store 120 in memory 26
[SELECT M] [+]	*120. SE+	Add 1 to memory 00
3.75 [→M/IN] [-] [INDIR]	3.75→M-	Subtract 3.75 from memory 27
[SELECT M] [-]	*3.75 SE-	Subtract 1 from memory 00
10 [→M/IN] [-] [INDIR]	10.→M-	Subtract 10 from memory 26
[→M/OUT] 25	600.→M	Recall memory 25
[→M/OUT] 26	110.→M	Recall memory 26
[→M/OUT] 27	**3.75→M	Recall memory 27
[→M/OUT] 00	26.→M 0	Recall memory 00

* Ignore the number that prints when [SELECT M] [+] or [SELECT M] [-] is used.
** This number will print in red.

SUMMARY

The calculator has several keys, such as [→M/IN] and [←M/OUT], that do not cause any action by themselves but cause action when used with other keys. These keys are [DEC SET], [SELECT M], [→M/IN], [←M/OUT], [GO SUB], [JUMP] and [LABEL]. The condition established by one of these keys may be replaced by any other non-digit key. For example, if you wanted to touch [←M/OUT] but accidently touched [→M/IN], just touch [←M/OUT]. The [→M/IN] condition will be replaced with [←M/OUT]. If you wanted to touch [GO SUB] but have already touched [JUMP], just touch [GO SUB] and the jump will be replaced. As long as you don't complete an action, you can replace any key depression.

There is a feature connected with [JUMP] and [GO SUB] that makes it easy to manually select one of several routines being held in program memory. You can program part of a jump, and complete it manually. For example, assume that there are five separate programs in the program memory. You would like to write a routine that will allow you to access any of these programs by simply entering the number of the program (1,2,3,4, or 5) and touching [RUN]. To do this, put [LABEL] 01 before the first program, [LABEL] 02 before the second program, and so on. At step 000, insert the instructions [JUMP] [LABEL] [0] [STOP]. After you touch [MANUAL] [RUN], enter the number of the program that you wish to access and then touch [RUN] again. This will complete the jump instruction and go to the correct program. This feature can be used with [GO SUB] or any of the other keys mentioned above. It is an excellent step saving device which allows great flexibility when writing and running programs.

III. THE SIX OPERATING MODES

The following is a detailed explanation of the six operating modes. The various modes allow you to enter, run, debug and correct programs and to use the 900 Series as a general purpose calculator. The six modes are called Manual, Run, Program, Edit, List and Trace.

The six modes are mutually exclusive, and cause the calculator to function in different manners. Depressing [MANUAL] will always reset any other mode. Some modes may reset themselves automatically. The mode keys and their functions are as follows:

This key selects Manual Mode, cancels any other mode. Manual is the normal operating mode of the calculator. This mode is used for manual calculations and for running programs. When turned on, the calculator will automatically select Manual Mode. Stay in Manual Mode unless you are entering a program by hand or debugging a program that doesn't work properly.

Touching [MANUAL] resets the program pointer to step 000. Touching [MANUAL] will also clear all the basic registers (the adding register, the counters, first and second factors and the keyboard). [MANUAL] will not affect the memories.

In general, the only reason for touching [MANUAL] is to address step 000 and to reset another operating mode.

Use [PROGRAM] when you want to enter a program from the keyboard. Touching [PROG] establishes the Program Mode, illuminates the Program indicator light, and cancels any other mode. In this mode, each key depression 'writes' the code for that key into the program memory at the current address, or step number, and causes the step number and the symbol for the key to print. The program pointer is then advanced to the next step. Every key is programmable with the following exceptions: [RUN], [PROGRAM], [EDIT], [LIST], [MANUAL] and [CLEAR].

To enter a program, touch [MANUAL] (to address step 000) then touch [PROGRAM]. The step number 000 will print and you can now enter your program instructions. When finished, touch [MANUAL]. Occasionally, you may wish to enter a part of a program starting from an address other than step 000. In this case, while in Manual Mode, touch [JUMP], enter the step address (always the three digits from which you wish to start, then touch [PROGRAM]. You can now enter your instructions as you normally would. Touch [MANUAL] after the last instruction.

If you are addressing a particular step while in Program Mode and there is already an instruction at that step, touching any programmable key will remove the old instruction and replace it with the new instruction.

At times, when entering a program from the keyboard, you will touch the wrong key. For example, you might be at step 086 and accidentally touch [+] instead of [-]. When this happens, you can correct the mistake immediately. To correct the error, touch [EDIT], touch [PROG] to backspace the program one step, then enter the correct step. You can backspace as many steps as you like by touching [EDIT] [PROG] repeatedly.

If you omit steps inadvertently and want to insert them, use Edit Mode.

A special 'clear program' condition exists right after depressing [PROG]. In this condition, if the first key depressed after [PROG] is [CLEAR], then the program will be cleared from the current address to the end to the program memory. A second depression will clear the entire program memory. If [CLEAR] is depressed in any other curcumstance while in Program Mode, it will have no effect and will be ignored.

Touching this key illuminates the Edit indicator, cancels any other mode and establishes the Edit Mode. Edit Mode is used to insert or delete instructions from a program. This is very helpful when correcting or changing a program that is already in the calculator.

In the Edit Mode, each depression of [CLEAR]

deletes the instruction at the current address and then 'closes' the remaining steps. The step addresses are automatically adjusted. The depression of any programmable key will insert the instruction into the existing program between the current address and the following one, expanding the program to allow for the key insertion. The step addresses are automatically adjusted. You may insert or delete as many steps as you wish. To insert or delete at a step other than step 000, manually jump to the desired step before touching [EDIT].

This key establishes the List Mode, illuminates the List indicator, and cancels any other mode. Touching [MANUAL] [LIST] will print out a complete program starting from step 000. The left side of the tape will show the step number and the right side will show the symbol for the instruction at that step.

When a blank step is encountered, the calculator will automatically stop printing, return to step 000 and go into Manual Mode. If a complete printout is not required, touch [MANUAL] to stop further listing.

If you wish to start listing from a particular step, touch [MANUAL] [JUMP], enter the three digit step address, then touch [LIST].

When there's a program stored in the memory, touching [RUN] will cause the calculator to begin processing the instructions. While instructions are being performed automatically, the calculator is in the Run Mode. It will stay in the Run Mode until it encounters a [STOP] instruction. [STOP] will return the calculator to Manual Mode, and no further program instructions will be processed until [RUN] is touched again.

While the calculator is in Run Mode, touching [MANUAL] will return it to Manual Mode and reset the program step counter to step 000. If you wish to go to a different step address, simply touch [JUMP], enter the desired address, then touch [RUN].

Trace Mode

Trace Mode is used to debug programs that do not operate properly. In Trace Mode, the calculator prints every programmed calculation as if it were being performed manually, with an easy to follow audit trail. This can help you locate the incorrect area of your program. To use Trace Mode, turn the printer switch to the OFF position, then run the program as you would normally.

IV. SAMPLE PROGRAMS

INTRODUCTION

The following section contains two completely documented sample programs. The programs form a payroll application that uses many features available in the calculator. These programs will show how the various functions can be used and how to organize, write and document a complex problem.

The first program is used to create a data card for each employee (data cards are magnetic cards that store information instead of program steps). The second program uses the data cards to compute the payroll each week. The payroll system operates as follows:

> When a new employee is hired, a data card is prepared specifically for that person. The card stores all necessary information such as salary (or hourly), number of exemptions, marital status, federal tax constants and any miscellaneous information such as union dues, payroll savings plan, etc. The card also holds totals for gross pay, federal income tax deductions, social security deductions, etc. These year-to-date figures are updated weekly during the payroll calculation.
>
> Program two uses the data cards originally created by program one to compute each employee's gross pay, deductions and net pay. After the net pay is calculated, the card is updated to store the new year-to-date totals. If there is a change in an employee's status (i.e. single to married, two exemptions to three), part one can be used again to revise the magnetic card to reflect the new information.

These sample programs have been completely documented. Following the program description and forumlas (when applicable) the operating

instructions explain all entries and results. Next, sample printouts show the results of program execution and verify the program. Then a flowchart shows program logic and organization followed by a program construction sheet. This is a standard Rockwell form which explains how the registers and the labels are used. Finally, there is another Rockwell form used for writing and documenting the program code. The final instructions are typed on this form with brief notes where necessary for coding clarification.

The two programming forms mentioned are very helpful when writing programs and may be purchased from Rockwell. The program construction form explains how registers, labels and subroutines have been used. At the top of the form is a register checklist and a label checklist that show at a glance what has been used (numbers covered in grey) and what is available for use (numbers that are not covered). These checklists help prevent duplication of registers or labels. Next, there is a description of the data contained in each register (when significant) and an explanation of the major routines. Finally, there is an area reserved for special instructions. Program steps are written on the left side of the column. Notes explaining the purpose of the program steps are written on the right side of the column. If necessary, this makes future program modification easier.

To thoroughly analyze the payroll program, the following procedure is suggested. First, enter the program code and practice operating the program. Recreate the sample tapes, then make up some new examples. Execute the program and check the validity of the results. By running the program several times, studying the flowchart, the coding forms and construction sheet for each program, a thorough understanding of the way to write and document programs for the Rockwell 900 Series will be developed.

For further assistance, the Rockwell 900 Series Programming Guide expands the theory of programming as it applies to the Rockwell 900 Series and is available through your Rockwell sales representative.

TITLE: PAYROLL - PART 1

PROGRAM DESCRIPTION AND FORMULAS:

This part of the payroll program is divided into 2 sections:

Section 1 allows you to create a new employee card or revise an existing employee card. Section 2 allows you to list out the information on an employee card.

One side of a magnetic card is required for each employee. The card contains the employee's number (red for single, black for married), hourly rate, number of exemptions, one regular deduction (such as insurance), year-to-date accumulations and federal tax constants.

OPERATING INSTRUCTIONS:

Section 1: To create a new employee card or revise an old one.

1. Depress [MANUAL]. Be sure that the LOAD/RECORD switch is set to LOAD. Insert side A of the Payroll Part 1 card into the read/write unit. Immediately insert side B of the same card.

   a. If a new employee card is being made, depress [CLEAR ALL] and proceed to step #2.

   b. If an employee card is to be revised, depress [→M/IN]. Be sure that the LOAD/RECORD switch is set to LOAD. Insert the desired employee card into the card reader.

2. Depress [MANUAL], [RUN].

3. An identifier and a current data item will print.

4. Enter new data, depress [RUN]. If data is not to be changed, depress [RUN] with NO ENTRY (see notes).

5. Repeat steps 3 and 4 for all 10 items. See sample tape for order of items.

a. Enter a negative employee number for single employees.
b. Enter year-to-date deductions as negative values by depressing [CHANGE SIGN] before depressing [RUN].
c. If employee is salaried, enter salary instead of hourly wage.
d. If you change item 10, go to step 6. If you do not change item 10, touch [JUMP] 046 [RUN], wait for two zeros to print, then go to step 7.

6. After the 10th item has been printed, the correct tax constants for that employee will be stored in the memories. When finished, two zeros will print.

7. Depress [+M/OUT]. Put the LOAD/RECORD switch in the RECORD position and insert the employee card. The information is now stored on the card. Return switch to the LOAD position.

8. To create another new card, return to step 1a. To revise another card, return to step 1b.

Section 2: To list the information on an employee card.

With appropriate employee card loaded in memory, depress [JUMP] [LABEL] 20, [RUN]. The information contained on the card will print.

SAMPLE PRINTOUT: Part 1:

Section 1

CREATING A NEW EMPLOYEE CARD

[MANUAL] [RUN]

```
Identifier                1.
Current Employee #        0.
New Employee #          300.e

Identifier                2.
Current Pay Rate          0.
New Pay Rate            5.50e

Identifier                3.
Current # of Exempt.      0.
New # of Exempt.          5.e

Identifier                4.
Current Ins. Rate         0.
New Insurance Rate      7.12e

Identifier                5.
Current Y-T-D Gross       0.
New Y-T-D Gross      1034.00e

Identifier                6.
Current Y-T-D Net         0.
New Y-T-D Net         810.28e

Identifier                7.
Current Y-T-D FWT         0.
New Y-T-D FWT         124.76e

Identifier                8.
Current Y-T-D FICA        0.
New Y-T-D FICA         60.48e

Identifier                9.
Current Y-T-D Ins.        0.
New Y-T-D Ins.         28.48

Identifier               10.
Cur. YTD Misc. Ded.       0.
New YTD Misc. Ded.    10.00e

Indication of             0.
 Program Completion       0.
```

REVISING AN EMPLOYEE CARD
(CHANGING THE NUMBER OF
EXEMPTIONS

[MANUAL] [RUN]

Identifier	1.
Current Employee #	300.
Identifier	2.
Current Pay Rate	5.50
Identifier	3.
Current # of Exemp.	5.
New # of Exemptions	4.e
Identifier	4.
Current Ins. Rate	7.12
Indicator of Program Completion	0. 0.

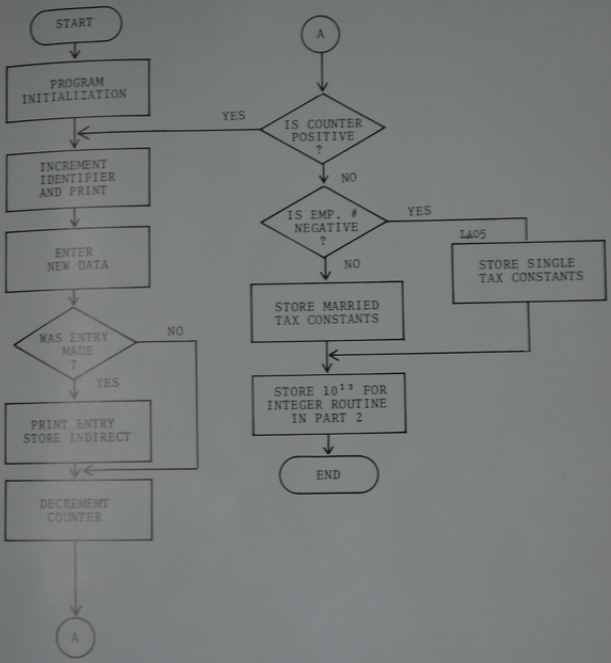
Section 2

PRINTING INFORMATION THAT IS
STORED ON AN EMPLOYEE CARD

[JUMP] [LABEL] 20 [RUN]

Identifier	1.
Employee #	300.
Identifier	2.
Salary Rate	5.50
Identifier	3.
# of Exemptions	4.
Identifier	4.
Insurance	7.12
Identifier	5.
Y-T-D Gross	1034.00
Identifier	6.
Y-T-D Net	810.28
Identifier	7.
Y-T-D FWT	124.76
Identifier	8.
Y-T-D FICA	60.48
Identifier	9.
Y-T-D Insurance	28.48
Identifier	10.
Y-T-D Miscellaneous Deductions	10.00

PAYROLL FLOWCHART: Part 1



Rockwell

900 Series

PROGRAM CONSTRUCTION

PROGRAM NUMBER:

PAGE OF

REGISTER USAGE CHECKLIST: ~~00~~ A S F₁ F₂ N₁ N₂ ~~01~~ ~~02~~ ~~03~~ ~~04~~ ~~05~~ ~~06~~ ~~07~~ ~~08~~ ~~09~~ 10 11 12 13 14 15 16 17 18 19 ~~20~~ ~~21~~ ~~22~~ ~~23~~ ~~24~~ ~~25~~ ~~26~~ ~~27~~ ~~28~~ ~~29~~ 30 31 32 33 34 35 36 37 38 39 40 41

LABEL USAGE CHECKLIST: 01 02 03 04 ~~05~~ 06 07 08 09 10 ~~11~~ 12 13 14 15 16 17 18 19 ~~20~~ ~~21~~ ~~22~~ 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 00

EXPLANATION OF REGISTER USAGE

00	Indirect	10		26	Y-T-D FWT
A		11		27	Y-T-D FICA
S		12		28	Y-T-D Insurance
F_1		13		29	Y-T-D Misc. Ded.
F_2		14		30	
N_1		15		31	
N_2		16		32	
01	↑	17		33	
02		18		34	
03		19		35	
04	Federal Tax Constants	20	Employee #	36	
05		21	Hourly rate	37	
06		22	# of Exemptions	38	
07	↓	23	Insurance	39	
08	Total Deductions	24	Y-T-D Gross Pay	40	
09	1×10^{13}	25	Y-T-D Net Pay	41	

EXPLANATION OF LABELS & SUBROUTINES

20	Print Out Routine		

SPECIAL INSTRUCTIONS:

Program Coding Form

CALCULATOR MODEL NO. 920-2 PROGRAM TITLE Payroll - Part 1

PROGRAMMER John Doe PAGE OF

DATE 10/74

000	SPACE	
01	1	
02	0	Set up counter in register 39
03	→M	
04	3	
05	9	
06	1	
07	9	Initialize reg. 00 with starting loc. of data
08	→M	
09	0	
10	0	
11	DEC SET	Set decimal at 2 places
12	2	
13	T	Clear Adder
14	SPACE	
15	1	
16	+	Add 1 to identifier
17	S	
18	PRINT	Print identifier
19	SE	Increment pointer register
20	+	
21	←M	Recall data
22	IND	
23	PRINT	Print data
24	STOP	Enter new data
25	JUMP	
26	AX	Jump if no data was entered
27	0	
28	3	
29	3	
30	PRINT	
31	→M	Otherwise store and print data
32	IND	
33	1	
34	→M	Subtract 1 from counter
35	-	
36	3	
37	9	
38	←M	Recall counter
39	3	
40	9	
41	JUMP	Jump back if not finished
42	+	
43	0	
44	1	
45	4	
46	←M	Recall emp. number
47	2	
48	0	
49	JUMP	Jump to store single tax
50	-	constants
51	LABEL	
52	0	
53	5	
54	1	Begin to store married
55	1	constants
56	.	
57	1	Store 11.14 in reg. 01
58	4	
59	→M	
60	0	
61	1	
62	2	
63	8	
64	.	
65	0	Store 28.02 in reg. 02
66	2	
67	→M	
68	0	
69	2	
70	1	
71	2	
72	8	
73	.	
74	0	Store 128.04 in reg. 03
75	4	
76	→M	
77	0	
78	3	
79	4	
80	0	
81	.	
82	0	Store 40.04 in reg. 04
83	4	
84	→M	
85	0	
86	4	
87	1	
88	1	
89	7	
90	.	
91	0	Store 117.04 in reg. 05
92	4	
93	→M	
94	0	
95	5	
96	8	
97	5	
98	.	
99	0	

FORM 859 PRINTED IN U.S.A.

SELECT MEMORY 1 2 3 4 5 6 7 8 9

Use a paper clip to indicate the memory being addressed. Move as necessary.

Program Coding Form

CALCULATOR MODEL NO. 920-2

PROGRAM TITLE Payroll - Part 1

PROGRAMMER

PAGE OF

DATE

100	4	Store 85.04 in reg. 06
01	→M	
02	0	
03	6	
04	7	
05	7	
06	.	
07	0	Store 77.04 in reg. 07
08	4	
09	→M	
10	0	
11	7	
12	JUMP	
13	LABEL	Jump to end
14	1	
15	1	
16	LABEL	
17	0	Start storing single constants
18	5	
19	1	
20	1	
21	.	Store 11.14 in reg. 01
22	1	
23	4	
24	→M	

25	0	
26	1	
27	2	
28	4	
29	.	
30	0	Store 24.04 in reg. 02
31	4	
32	→M	
33	0	
34	2	
35	3	
36	8	
37	.	
38	0	Store 38.03 in reg. 03
39	3	
40	→M	
41	0	
42	3	
43	1	
44	2	
45	9	
46	.	
47	0	Store 129.02 in reg. 04
48	2	
49	→M	

50	0	
51	4	
52	2	
53	9	
54	.	
55	0	Store 29.04 in reg. 05
56	4	
57	→M	
58	0	
59	5	
60	3	
61	8	
62	.	
63	0	Store 38.04 in reg. 06
64	4	
65	→M	
66	0	
67	6	
68	6	
69	4	
70	.	
71	0	Store 64.04 in reg. 07
72	4	
73	→M	
74	0	

75	7	
76	LABEL	
77	1	
78	1	
79	1	
80	0	
81	X^y	
82	1	Generate & store 1X10 in reg. 09
83	3	
84	=	
85	→M	
86	0	
87	9	
88	0	
89	SPACE	
90	PRINT	Print 2 zeros
91	PRINT	
92	STOP	End
93	STOP	
94	LABEL	
95	2	Begin list out card routine
96	0	
97	SPACE	
98	SPACE	
99	SPACE	

SELECT MEMORY 1 2 3 4 5 6 7 8 9

Use a paper clip to indicate the memory being addressed. Move as necessary.

Program Coding Form

CALCULATOR MODEL NO. 920-2

PROGRAM TITLE Payroll - Part 1

PROGRAMMER

DATE

PAGE OF

200	T	Clear adder
01	1	
02	0	Set up counter in reg. 41
03	→M	
04	4	
05	1	
06	2	
07	0	Store starting location of data in reg.00
08	→M	
09	0	
10	0	
11	1	Initialize identifier counter
12	+	
13	LABEL	
14	2	
15	2	
16	S	Subtotal identifier counter
17	SPACE	
18	PRINT	Print identifier
19	→M	
20	IND	Recall and print data
21	PRINT	
22	1	
23	→M	Subtract 1 from counter
24	-	
25	4	
26	1	
27	M	Recall counter
28	4	
29	1	
30	JUMP	Jump when finished
31	=	
32	LABEL	
33	2	
34	1	
35	SE	Increment pointer
36	+	register
37	1	Add 1 to identifier counter
38	+	
39	JUMP	Jump back to print next data item
40	LABEL	
41	2	
42	2	
43	LABEL	
44	2	Beginning of the end
45	1	
46	SPACE	
47	SPACE	
48	SPACE	
49	STOP	
50		
51		
52		
53		
54		
55		
56		
57		
58		
59		
60		
61		
62		
63		
64		
65		
66		
67		
68		
69		
70		
71		
72		
73		
74		
75		
76		
77		
78		
79		
80		
81		
82		
83		
84		
85		
86		
87		
88		
89		
90		
91		
92		
93		
94		
95		
96		
97		
98		
99		

SELECT MEMORY 1 2 3 4 5 6 7 8 9

Use a paper clip to indicate the memory being addressed. Move as necessary.

FORM 409 PRINTED IN U.S.A.

Rockwell

900 Series

PROGRAMMER: John Doe | NUMBER OF MAGNETIC CARDS: 1 | PROGRAM NUMBER:

MODEL NUMBER: 920-2 | CATEGORY: BUSINESS | DATE: 10/74 | PAGE | OF

TITLE: PAYROLL - PART 2

PROBLEM DESCRIPTION AND FORMULAS:

Part 2 of the payroll program performs the actual payroll calculations using the information contained on the employee cards.

The regular pay is calculated by multiplying the employee's hourly rate by the number of regular hours for that week; the overtime, by multiplying the hourly rate by the number of overtime hours and 1.5.

The method used for calculating the federal withholding tax is based on the Federal Percentage Method using the weekly tax table, as described in the Payroll Management Guide, 1974 Federal Graduated Withholding Tax Tables, published by Commercial Clearing House, Inc.

The FICA is calculated by multiplying the gross of 5.85% with cutoff point at $13,200.

OPERATING INSTRUCTIONS:

1. Depress [MANUAL]. Be sure that the LOAD/RECORD switch is set to LOAD. Insert side A of the Payroll Part 2 card into the read/write unit. Immediately insert side B of the same card.
2. Depress [→M/IN]. Be sure that the LOAD/RECORD switch is set to LOAD. Insert the desired employee card.
3. Depress [MANUAL], [RUN].
4. The employee number and hourly rate or salary will print. (If salaried employee, go to step #9.)
5. Enter number of regular hours, depress [RUN].
6. Regular pay will print.
7. Enter number of overtime hours, depress [RUN]. (If no overtime, just touch [RUN].)

8. Overtime pay will print.
9. Enter miscellaneous pay, depress [RUN]. (If no miscellaneous pay, just touch [RUN].
10. Gross pay will print.
11. The following will print:
    a. Number of exemptions
    b. Federal withholding tax
    c. FICA
    d. Insurance
12. Enter the amount of miscellaneous deductions, if any, and touch [RUN]. [If no miscellaneous deductions, just touch [RUN]).
13. The following will print:
    a. Net pay
    b. Year-to-date gross
    c. Year-to-date federal withholding tax
    d. Year-to-date FICA
    e. Year-to-date insurance
    f. Year-to-date miscellaneous deductions
    g. Year-to-date net pay
14. Depress [+M/OUT]. Put LOAD/RECORD switch in the RECORD position. Insert employee's card into the card reader. After it has been returned, remove it from the card reader. The card has now been updated.
15. Program returns to step #2 for next employee. You do not have to repeat step #1.

NOTE:

Mistakes will occur occasionally while data is being entered. If so, simply start again from step 2.

SAMPLE PRINTOUT:

Part 2

HOURLY EMPLOYEE

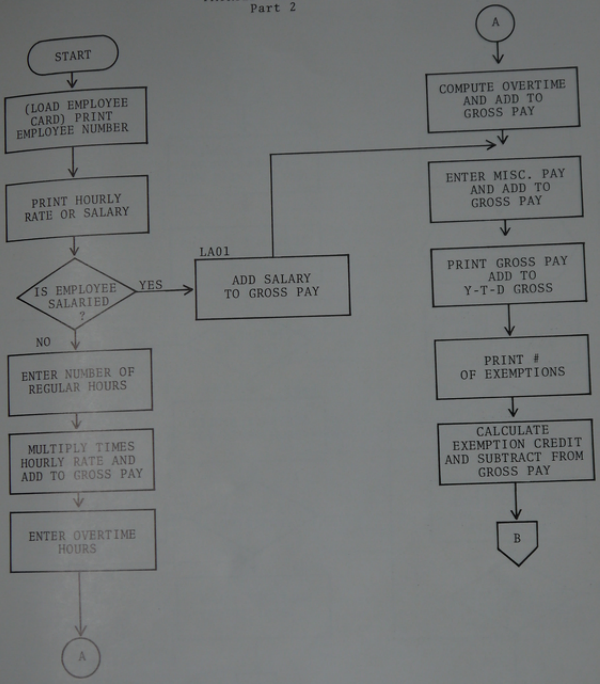
	[MANUAL]	[RUN]
Employee #	300.	
Hourly Rate	5.50	
Regular Hours	40.e	
Regular Pay	220.00	
Overtime Hours	4.e	
Overtime Pay	33.00	
Misc. Pay	5.50e	
Gross Pay	258.50	
No. of Exemptions	4.	
Fed. Withholding	31.19	
FICA	15.12	
Insurance	7.12	
Misc. Deductions	2.50e	
Net	202.57	
Y-T-D Gross	1292.50	
Y-T-D FWT	155.95	
Y-T-D FICA	75.60	
Y-T-D Insurance	35.60	
Y-T-D Misc.	12.50	
Y-T-D Net	1012.85	

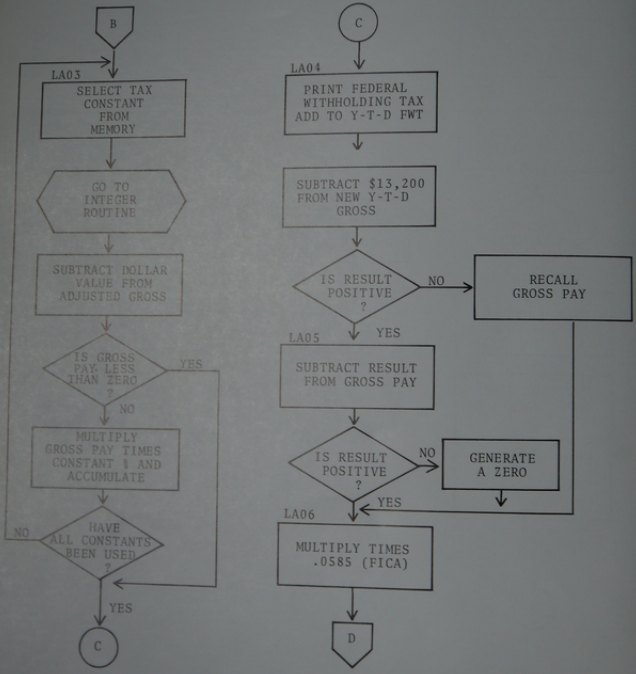
SALARIED EMPLOYEE

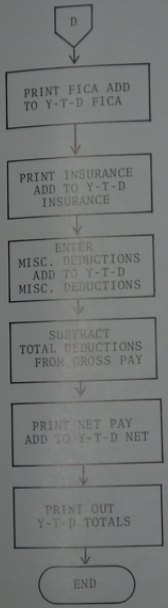
	[MANUAL]	[RUN]
Employee #	400.	
Salary	253.00	
Misc. Pay	0.e	
Gross Pay	253.00	
No. of Exemptions	4.	
Fed. Withholding	30.09	
FICA	14.80	
Insurance	7.12	
Misc. Deductions	2.50e	
Net	198.49	
Y-T-D Gross	253.00	
Y-T-D FWT	30.09	
Y-T-D FICA	14.80	
Y-T-D Insurance	7.12	
Y-T-D Misc.	2.50	
Y-T-D Net	198.49	

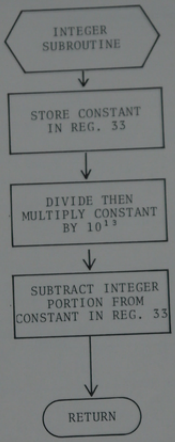
For each employee's payroll, verify that all year-to-date totals have been accumulated correctly.

PAYROLL FLOWCHART
Part 2




B
LA03
SELECT TAX CONSTANT FROM MEMORY
GO TO INTEGER ROUTINE
SUBTRACT DOLLAR VALUE FROM ADJUSTED GROSS
IS GROSS PAY LESS THAN ZERO ?
YES
NO
MULTIPLY GROSS PAY TIMES CONSTANT % AND ACCUMULATE
HAVE ALL CONSTANTS BEEN USED ?
NO
YES
C
C
LA04
PRINT FEDERAL WITHHOLDING TAX ADD TO Y-T-D FWT
SUBTRACT $13,200 FROM NEW Y-T-D GROSS
IS RESULT POSITIVE ?
NO
RECALL GROSS PAY
YES
LA05
SUBTRACT RESULT FROM GROSS PAY
IS RESULT POSITIVE ?
NO
GENERATE A ZERO
YES
LA06
MULTIPLY TIMES .0585 (FICA)
D





NOTE: Upon return from this subroutine the integer portion of the number that was in the K register will be in K and the decimal portion will be in register 33.

PROGRAM NUMBER:

PROGRAM CONSTRUCTION

PAGE OF

REGISTER USAGE CHECKLIST: 00 A S F F_1 N N_1 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41

LABEL USAGE CHECKLIST: 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 00

EXPLANATION OF REGISTER USAGE

00	Indirect	10		26	Y-T-D FWT
A		11		27	Y-T-D FICA
S		12		28	Y-T-D SDI
F_1		13		29	Y-T-D insurance
F_2		14		30	Y-T-D misc. ded.
N_1		15		31	Gross pay
N_2		16		32	Working reg. gross pay
01		17		33	Working reg. for integer routine
02		18		34	
03		19		35	
04	Federal tax constants	20	Employee #	36	
05		21	Hourly rate	37	
06		22	# of exemp.	38	
07		23	Insurance	39	
08	Total deductions	24	Y-T-D gross pay	40	
09	1×10^{13}	25	Y-T-D net pay	41	

EXPLANATION OF LABELS & SUBROUTINES

35	Integer routine*		

SPECIAL INSTRUCTIONS:

* The tax constants are stored as decimal numbers with the dollar values as the whole number and the percentage constants as the decimal portion. The integer routine is used to separate the dollar value from the percentage. Upon returning from this routine the dollar value is in the K register and the percentage is in register 33.

Program Coding Form

CALCULATOR MODEL NO. 920-2

PROGRAM TITLE Payroll - Part 2

PROGRAMMER

DATE

PAGE OF

000	SPACE	
01	SPACE	
02	T	Clear adder
03	←M	Recall and print
04	2	employee number
05	0	
06	PRINT	
07	SPACE	
08	←M	Recall and print hourly
09	2	rate or salary
10	1	
11	PRINT	
12	SPACE	
13	+	
14	7	Subtract 75 from the
15	5	value printed to determine if
16	-	it is sal-
17	T	aried or hourly emp.
18	JUMP	Jump if salaried to
19	+	store salary in gross
20	LABEL	pay reg.
21	0	
22	1	
23	STOP	Enter no. of reg. hours
24	PRINT	Print entry
25	X	
26	←M	Multiply reg. hours
27	2	times hourly rate
28	1	
29	=	
30	PRINT	Print reg. pay
31	→M	Store in gross pay register
32	3	
33	1	
34	SPACE	
35	0	
36	STOP	Enter no. of overtime hrs
37	PRINT	Prt. entry
38	X	
39	1	Multiply overtime
40	.	hours times 1.5 times
41	5	hourly rate
42	X	
43	←M	
44	2	
45	1	
46	=	
47	PRINT	Print over-time pay
48	→M	
49	+	
50	3	Add to gross pay
51	1	
52	LABEL	
53	0	
54	2	
55	SPACE	
56	0	
57	STOP	Enter misc. pay
58	PRINT	Print entry
59	→M	
60	+	Add to gross pay
61	3	
62	1	
63	←M	Recall and print gross pay
64	3	
65	1	
66	PRINT	
67	→M	Add gross pay to
68	+	year-to-date gross
69	2	pay
70	4	
71	→M	Store gross pay in
72	3	working register
73	2	
74	SPACE	
75	SE	Select & clear memory 8
76	8	
77	*	
78	0	
79	←M	Recall & print no.
80	2	of exemptions
81	2	
82	PRINT	
83	SPACE	
84	X	
85	1	
86	4	Number of exemptions
87	.	times 14.40 (allowance)
88	4	
89	0	
90	=	
91	→M	
92	-	Subtract exemption
93	3	allowance from gross
94	2	pay
95	1	
96	→M	Store 1 in register 00
97	0	
98	0	
99	LABEL	

SELECT MEMORY 1 2 3 4 5 6 7 8 9

Use a paper clip to indicate the memory being addressed. Move as necessary.

Program Coding Form

CALCULATOR MODEL NO. 920-2

PROGRAM TITLE Payroll - Part 2

PROGRAMMER

PAGE OF

DATE

00	0	
01	3	
02	←M	Recall tax constant
03	IND	
04	S	Go to integer routine
05	3	
06	5	
07	→M	
08	-	Subtract dollar
09	3	amount from
10	2	adjusted gross pay
11	←M	
12	3	
13	2	
14	JUMP	Jump if adjusted
15	-	gross pay is less than
16	LABEL	zero
17	0	
18	4	
19	X	Multiply tax % times
20	→M	adj. gross and add to
21	3	accumulated tax
22	3	
23	=-	
24	SE	Increment pointer reg.
25	+	to point at next tax
26	8	constant
27	+	Subtract
28	←M	from pointer reg. to det.
29	0	if all of the tax
30	0	consts. have been used.
31	-	
32	T	If all consts. have
33	JUMP	not been used, go
34	+	back to calculate
35	LABEL	with next constant
36	0	
37	3	
38	LABEL	
39	0	
40	4	
41	◊	Recall tax
42	→M	
43	+	Add to Y-T-D fed.
44	2	withholding tax
45	6	
46	PRINT	Print tax
47	←M	Recall Y-T-D gross
48	2	
49	4	
50	+	
51	1	
52	3	Subtract FICA
53	2	cut-off from Y-T-D
54	0	gross
55	0	
56	-	
57	T	
58	JUMP	Jump if Y-T-D gross
59	+	exceeds FICA cut-off
60	LABEL	
61	0	
62	5	
63	←M	Recall gross pay
64	3	
65	1	
66	LABEL	
67	0	
68	6	
69	X	
70	.	Multiply times FICA
71	0	rate
72	5	
73	8	
74	5	
75	=	
76	CHANGE SIGN	Change signs
77	PRINT	Print FICA
78	→M	
79	+	Add to Y-T-D FICA
80	2	
81	7	
82	M	Add to total
83	+	deductions
84	←M	
85	2	Recall & print
86	3	insurance deduction
87	CHANGE SIGN	
88	PRINT	
89	→M	Add to total
90	2	deductions
91	8	
92	0	
93	STOP	Enter misc. deductions
94	CHANGE SIGN	Change sign
95	PRINT	Print entry
96	SPACE	
97	SPACE	
98	M	Add to total deductions
99	+	

SELECT MEMORY 1 2 3 4 5 6 7 8 9

Use a paper clip to indicate the memory being addressed. Move as necessary.

Program Coding Form

CALCULATOR MODEL NO. 920-2

PROGRAM TITLE Payroll - Part 2

PROGRAMMER

PAGE OF

DATE

200	→M	
01	+	Add to Y-T-D misc. deductions
02	2	
03	9	
04	◊	Recall tot. deductions
05	+	
06	+M	A neg. tot. ded. added
07	3	to the gross pay gives
08	1	the net pay
09	+	
10	T	
11	PRINT	Print net pay
12	→M	
13	+	Add to Y-T-D net pay
14	2	
15	5	
16	SPACE	
17	SPACE	
18	+M	
19	2	Recall & print
20	4	Y-T-D gross pay
21	PRINT	
22	+M	Recall and print
23	2	federal withholding
24	6	tax
25	PRINT	
26	+M	
27	2	Recall & print
28	7	Y-T-D FICA
29	PRINT	
30	+M	Recall & print Y-T-D
31	2	insurance
32	8	
33	PRINT	
34	+M	Recall & print Y-T-D
35	2	misc. deductions
36	9	
37	PRINT	
38	+M	
39	2	Recall and print
40	5	Y-T-D net pay
41	PRINT	
42	SPACE	
43	SPACE	
44	STOP	End of program
45	STOP	
46	LABEL	Start of integer
47	3	routine
48	5	
49	→M	
50	3	Store no. in working
51	3	register
52	÷	
53	+M	Dividing & then mult.
54	0	by $1X10^{13}$ will leave
55	9	the integer only in K
56	X	
57	=	
58	→M	Subtract the integer
59	-	part from comp. no.
60	3	Reg. 33 now holds
61	3	decimal part
62	←S	Return
63	LABEL	
64	0	
65	5	
66	-	Subtract amount over
67	+M	FICA cutoff from gross
68	3	pay to determine
69	1	portion to be taxed
70	+	
71	T	
72	JUMP	
73	+	
74	LABEL	Jump back to main prog.
75	0	
76	6	
77	0	
78	JUMP	Generate zero &
79	LABEL	jump back if no
80	0	portion is to be
81	6	taxed
82	LABEL	
83	0	
84	1	
85	←M	
86	2	
87	1	Recall salary &
88	→M	store in gross pay
89	3	register
90	1	
91	JUMP	
92	LABEL	Jump back to main
93	0	program at misc. pay
94	2	entry
95		
96		
97		
98		
99		

SELECT MEMORY 1 2 3 4 5 6 7 8 9

Use a paper clip to indicate the memory being addressed. Move as necessary.

V. ERROR CONDITIONS

Under normal operating conditions the calculator will rarely, if ever, go into an error condition (i.e. prints 'E', keyboard locked). If this occurs, it is an indication that an improper calculation has been attempted. If you are not sure what is wrong, then consult the following list for an explanation.

1. The following calculations are considered improper. Attempting them will cause an error.

   a. Division by zero
   b. Square root of a negative number
   c. [x^y] if x is negative or zero
   d. [Ln x] if x is negative or zero
   e. [$\frac{1}{x}$] if x = 0

2. The following conditions are beyond the calculator's capacity and will cause an overflow condition. The symbol 'E' will print and the keyboard will lock.

   a. Overflow will occur during multiplication or division when a result is greater than or equal to 10^{14}. When this happens, the answer that prints will be correct to 14 digits if multiplied by 10^{14}.
   b. Overflow will occur if you try to add two numbers whose sum is greater than 10^{14}. When this happens, the last number will not be added so that the previous (correct) balance will be undisturbed. For example, if you touch 60,000,000,000,000 [+] 50,000,000,000,000 [+], the calculator will overflow. If you touch [CLEAR] [T], 60,000,000,000,000 will print. [M+], [M-], [→M/IN], [+] and [←M/OUT]

will cause an overflow if the above conditions are attempted.

c. Overflow will occur if you try to manually enter a number larger than 10^{14}.

d. Overflow will occur if $[x^y]$ produces an answer that is larger than or equal to 10^{14}. The common log of the answer will print.

e. Overflow will occur if e^x produces an answer larger than 10^{14}. (This will occur when x is greater than 32.236191-301.) The common log of the correct answer will print.

3. The following program instructions are illegal and will cause the calculator to lock up.

a. A conditional or unconditional jump to a label will cause an error if the label does not exist in the program memory.

b. A conditional or unconditional branch [GO SUB] to a subroutine will cause an error if there is no corresponding subroutine.

c. Trying to access a memory, step or label that doesn't exist will cause an error condition. For example, if you have a 920-2 (41 memories and 448 program steps), touching [→M/IN] 50 or [→M/OUT] 75 will cause an error. Touching [GO SUB] 57 or [JUMP] [LABEL] 39 [RUN] will cause an error if these labels have not been used in the program.

d. A memory cannot be selected indirectly if the number in memory 00 is larger than the number of memories in the calculator. For example, if your calculator has 9 memories and 37 is in memory 00, [→M/IN] [INDIR] will cause an error condition.

4. If the calculator reads a [RETURN] instruction without first reading a [GO SUB] instruction, it will go into an error condition.

5. Loading a data card without first touching [→M/IN] may cause an error and will destroy a program already in the machine.

6. Entering a defective magnetic card with the switch set to LOAD or TEST will cause an error condition.

4. If the calculator reads a [RETURN] instruction without first reading a [GO SUB] instruction, it will go into an error condition.

5. Loading a data card without first touching [→M/IN] may cause an error and will destroy a program already in the machine.

6. Entering a defective magnetic card with the switch set to LOAD or TEST will cause an error condition.

Rockwell