Historical Document

H. G. Kolsky

The Dual Coding System was a very successful program which allowed one to use Floating Point arithmetic on the IBM 701. It served as a forerunner to the IBM 704 which had built-in floating point.

THE DUAL CODING SYSTEM

FOR THE IBM 701 CALCULATOR

1955

Los Alamos

Alt: { 1028 - 1039
Dual Prop } 1041 - 2047

THE DUAL CODING SYSTEM*

7 card load 0000 - 0350
0000 - $(0346)_{10}$

INTRODUCTION:

Dual II uses { $(3026)_{10}$ - $4095)_{10}$
$(5722)_8$ - $(7777)_8$

This report assumes that the reader has some familiarity with the IBM 701 Calculator (previously called the Defense Calculator), or with similar high-speed computing equipment such as the MANIAC. If such prior knowledge is not possessed by the reader, or if it is desired to review specific details about the 701 Calculator, it is suggested that reference be made to the IBM manual, "Principles of Operation, Type 701 and Auxiliary Equipment". Copies of this manual are available through Group T-1.

The present form of the Dual System has been checked out on the Pilot Model of the IBM 701 Calculator which is equipped with 1024 (instead of 2048) full words of fast storage. Though the system is at present in usable form for 2048 words of electrostatic memory, it will be relocated to a more optimum location for the Los Alamos 701, causing some minor changes in the addresses given on succeeding pages.

(*) Several members of Group T-1 contributed time and ideas to the development of the Dual Coding System. Principally involved were Bengt Carlson, Stewart Schlesinger, Dura Sweeney, Janet Bendt, Ivan Cherry, and Lloyd Hubbard, the latter of the IBM Company. Questions regarding the use and application of the Dual Coding System should be referred to Stewart Schlesinger, Telephone: 2-2128.

PURPOSE:

The Dual System has been devised for the IBM 701 Calculator with a two-fold purpose. Primarily, it is designed to provide a flexible and fast mechanism for carrying out large scale computations requiring the use of floating-point operations. Its second purpose is to provide a floating-point system that can be used conveniently in conjunction with fixed-point calculations. The main features of this coding system are: (1) the great ease with which the programmer can alternate between the use of standard machine commands, and orders which are interpreted as floating-point commands, and (2) the strong similarity of structure between the floating-point orders and standard machine commands, since the Dual System is entirely "one-address" in nature.

There has been no attempt, in this system, to alter the form of the machine orders from their original one-address structure; instead, every effort has been made to parallel the scheme of commands which has been built into the machine. For this reason the Dual System is precisely as easy to program for, as if the 701 itself had been constructed as a combined floating-point and fixed-point computer, since the Dual System effectively transforms the 701 into such a machine. The Dual System is therefore recommended to those people who plan to become familiar with the system of orders built into the machine, and who will require floating-decimal computations for either an entire problem or for selective parts of one.

STRUCTURE:

As was indicated previously the Dual System uses a one-address system of commands, with the entire order situated in one half-word of the machine's storage. In floating-point orders the first (or sign) bit is used as a breakpoint* indication for all commands except transfer orders, where it will designate whether or not the next order (if the transfer is executed) is to be a floating-point or fixed-point command. The second through sixth bits indicate the operation to be performed, and the remaining twelve bits designate the address of the number which is to enter into the calculation. The interpretation of all but the sign bit is exactly the same as that used by the machine's internal circuitry.

All floating point orders in the Dual System which refer to storage locations, refer to full words, independent of the sign of the order. In the full word location referred to by such an order is contained a complete floating-decimal number, with the exponent (i.e., power of ten) in the first seven bits, exclusive of the sign, and the remaining twenty-eight bits containing the significant bits of the number (with the binary point assumed to be before the first of those bit positions). The sign of the entire word is the sign associated with the significant bits. The seven bit exponent is assumed positive with a range from one to ninety-nine, and is interpreted as the true exponent plus 50. The sole

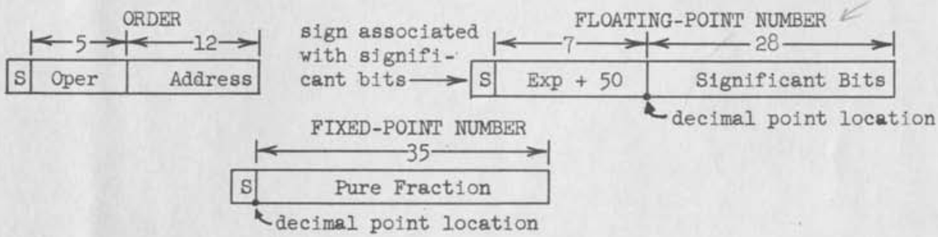
---

(*) The term breakpoint is here used to designate a controlled display of the result of a calculation, where the control is contained both in the order itself and on the operator's console of the 701.

exception to the above description is the Dual System representation of zero. This number will carry both an exponent and a significant part of zero, and neither part can be zero without the other also being zero.

Diagrams of the structure of orders and numbers are found immediately below:

8.4 dec



Both the Accumulator and the MQ registers are simulated by a single location in the Dual System, referred to as the AMQ*. It is to this register that the RADD, RSUB, and LOAD MQ orders bring their numbers, and from this location the number is taken for the STORE and STORE MQ orders. This AMQ register contains the result of the calculation after all arithmetic and transcendental operations, and both multiplicands and dividends are stored here prior to multiplication or division operations.

Keeping in mind that this single register acts as both the Accumulator and MQ, it is possible to interpret all orders (except the STORE A order) of the Dual System as if they were standard orders of the 701,

---

(*) For a detailed explanation of the structure of the AMQ, refer to the Appendix.

except that they are performed in floating point. The STORE A order, within the Dual System, is used to indicate a given transcendental function, which is computed with contents of the AMQ as the argument, with the answer then being placed in the AMQ. The particular transcendental function is determined by specifying the beginning address of the sub-routine (which computes that function) as the address part of the STORE A order. This technique allows for removal and repositioning of most sub-routines as individual problems might demand, and the adjoining of additional function sub-routines when they prove necessary.

USING THE SYSTEM:

The Dual System itself occupies 930 half-word storages at the end of the first bank of Electrostatic Memory. The remaining 3166 half-words (1583 full words) as well as the four tapes and four drum units are completely available to the programmer.

As has already been mentioned, the coding techniques for the Dual System are exactly the same as for the 701 itself. This similarity has reduced the time of interpretation and execution of floating-decimal arithmetic commands, such as ADD, SUBAB, MPY, and DIV to about two milliseconds. Loading orders, such as RSUB and LOAD MQ, and the STORE order take about one millisecond. The transfer orders, both conditional and unconditional also take in the neighborhood of one millisecond.

In order to activate the interpretation of a succeeding block of commands as floating-decimal orders it is necessary to give a calling sequence of two orders after which all orders will be treated as floating-decimal commands until a positive transfer order is encountered. This transfer, if executed, will carry the program back into fixed point and out of the control of the interpretation scheme. The calling sequence, if the interpretation is to begin at order number 1356, is as follows:

To Transfer to ~~Floating~~

1354	RADD	$(1354)_{10}$
1355	TR	$(1800)_{10} = (3410)_8$
1356		
1357	Interpreted as floating-decimal commands	
ETC		

The function of the positive transfer order in the Dual System is explained above, but to reiterate, whenever such a transfer is executed (i.e., if it is a conditional transfer the condition is satisfied), then the next order executed is contained in the storage whose address is the address part of the transfer order, and this order as well as those succeeding are executed independent of any interpretation scheme except those built into the machine. The negative transfer orders, on the other hand, merely instruct the interpretation loop of the Dual System to extract its next order from the location indicated by the address part of the transfer order. Hence, this next order and those succeeding will be interpreted as floating-decimal instructions.

In almost all problems using the Dual System it is necessary to use the positive transfer orders frequently, since all address modification is done outside of the block of interpreted orders. Making full use of the similarity in order structure between the Dual System and the built-in coding system, all address arithmetic and storing of the modified orders is done in fixed point, as if the modified orders were to be executed independent of any interpretation scheme. This technique allows for the modification of addresses at standard machine speed and allows for precise arithmetic on indices which is otherwise a rather difficult feat in floating-point calculations (in a binary machine where decimal input and output are desired).

Since certain calculations with a stored program computing machine require the majority of the commands to be devoted to logical orders or index and address arithmetic, it is necessary to reassess the comparison of speed between a fixed-point calculation and a floating-point calculation using the Dual System. A direct comparison of computing speeds between the floating-point and fixed-point orders would indicate that the former would take about fifteen times as long; however, if in particular the problem of matrix multiplication is considered, so many of its commands are performed outside the interpretation scheme that the ratio of computing time between the floating-point and fixed-point calculations is reduced to about three.

To code a problem in the Dual System it is possible to use all of the simplifying techniques already designed for 701 coding. All coding may be done in decimal (or octal if the programmer so prefers) and the exact same form is followed as if one was coding directly for the 701 Calculator. Regional Coding (which has been described previously for the 701 in a paper written by IBM and distributed to all interested parties) is directly applicable to the Dual System. With this coding technique it is possible to code a problem for the Dual System in many separate parts and later assemble them together into a single integrated unit. This assembly can be so performed that any gaps in the sequence of orders may be removed so that no storage is wasted, and if a correction is necessary that will require a number of orders to be inserted in the middle of a block of commands, the necessary reassignment of order locations for the commands below the correction can be performed automatically. The Dual System can take full advantage of all the standard input programs for the 701 for all of its orders (both fixed-and floating-point).

Like the orders in the Dual System, the input data in the form of floating-decimal numbers can be entered into the machine by any standard input program for data in the form of full words. In loading such a decimal number the first two positions are reserved for the exponent (written in the form of the true exponent plus 50), and the next eight places represent the significant digits expressed as a pure decimal fraction. The sign of the number is the sign of the significant digits. The exponent and significant digits are thought of as being separated by a decimal point. Some illustrations of this notation are found immediately below

Actual Number	Dual System Representation
-.56789123	-50.56789123
59.78	52.59780000
-1,673,330.5	-57.16733305

In using the standard input programs the decimal point is actually thought of as being after the first two digits of the Dual System Representation of the floating-point number and the input program is instructed to position the binary point after the seventh bit of the converted number which is precisely the dividing point between the exponent and significant bits of the Dual System Representation of a floating-point number expressed in binary. By this simple technique, any input program designed by IBM for variable decimal point input (as all have been to date for the 701) can handle Dual System data (in both fixed-and floating-point).

Since the auxiliary storage devices of the 701 (Magnetic Tapes and Magnetic Drums) are constructed to transfer either orders or data in arbitrary sized blocks assuming the machine to contain orders and data in standard form (in half and full word units), sections of a problem coded with the Dual System can be stored and called forth from these auxiliary storage devices in precisely the same manner as if they were coded independent of any interpretational scheme. This can be accomplished by actually coding a specific routine applicable to a given problem or by using a standard tape or drum input-output program. For further details on the auxiliary storage devices the IBM manual, "Principles of Operation, Type 701" should be consulted. When using the Dual System one would (by means of a positive transfer order) become free of the interpretational scheme and then proceed, as with a standard coded problem, to activate the input-output program for the drum or tape units.

Though it is possible to print any numbers represented in the Dual System by standard 701 printing programs, a special printing program has been devised which will facilitate the printing process for problems utilizing the Dual System. The print routine will print either floating-point or fixed-point decimal numbers with a maximum of six per line, the printer operating at full speed (150 lines per minute) if more than one line has been directed to print. The floating-point numbers are printed in the same word-form as was used for input, i.e., -1,673,330.5 is represented as -57.16733305. The fixed-point numbers print as pure ten-digit fractions (e.g., .0005978630), though a simple scaling process may be performed prior to printing, in order to space the true decimal-point location arbitrarily in these output numbers. A calling sequence of four orders is used to activate the printing program. For example, if the six floating-point numbers stored in full word locations 1000 through 1010 are to be printed, three per line, the following sequence of orders (outside of the interpretational scheme) are used, with the calling sequence beginning, say, at location 500: 6 is max no.

$(2703)_8$ Printing calling seq.

500 + RADD	500	α	Radd α
501 - TR	$(1475)_{10}$	$\alpha+1$	Tr (− for floating, + for fixed)
502 + (3)	1000	$\alpha+2$	(no. wds. per line), (starting addr)
503 + STOP	1012	$\alpha+3$	(2 + addr of last word)
504	next order after printing completed		

The first two commands are standard, with the address of the first order being its own location and the negative sign of the second order indicating floating-point printing. A positive sign on the second order would indicate fixed-point printing. The operation part of the third order indicates the number of words per line and its address part is the location of the first word to be printed. The address of the fourth order is two plus the address of the last word to be printed.

In order to facilitate checking problems, a breakpoint has been introduced on all floating-point commands <u>excluding</u> transfer commands. If switch #1 is in normal position the printer will print the following decimal information <u>after</u> executing a positive floating-point order (which is not a transfer order):

Location of Order   Operation and Address   Contents of AMQ

If switch #1 is placed in its down position the calculator will proceed at full speed. A simple modification of the Dual System (accomplished by the addition of an extra card when loading) will make the printer produce the breakpoint information for <u>every</u> floating-point order (except transfers) to enable the programmer to produce a tracing of his code.

In the process of executing a problem it is possible to produce numbers which are no longer within an acceptable range for the floating-point system. If such violations of range exist, the machine will automatically stop at order location number 1814; however, in the case of an exponent which is too negative for the existing range, i.e. the true exponent ≤ -50, the floating-point number zero will be substituted for that number and the calculation will continue without interruption. The device of stopping at this definite location is used to detect other untenable conditions such as an attempt to extract the square root of a negative number, or to compute the logarithm of zero. A table of such violations will be given later in this write-up.

For the programmer who plans to use the Dual System in conjunction with fixed-point computations, commands have been provided to convert full-word fixed-point numbers to Dual-System-type floating-point numbers and to perform the reverse transformation. A special predetermined exponent (situated in the first seven bits of a half-word) is placed in a standard location. The floating-to-fixed conversion will shift the significant part of the floating number to the right d places, where d is the difference between the predetermined exponent and the exponent of the number being shifted (where the floating-point number is the contents of the AMQ and the resulting fixed point number appears in the AMQ register). The resulting fixed-point number is stored in proper form only by the floating-point STORE order*. The fixed-to-floating conversion assumes that the fixed point number has been scaled to be less than one by means of some power of ten which is stored in the same standard location (and in the same form) as is used in the floating-to-fixed conversion. This power of ten, which must be stored in the first seven bits of this standard location after 50 has been added to it, is taken as the exponent of the floating-point

---

(*) See Appendix for details on how the floating-point STORE and RADD orders are actually executed.

number to which the fixed-point number is to be transformed, and the new floating-point number is put in standard form by shifting off leading zeros. The final result appears in the AMQ. The fixed-to-floating conversion assumes that the fixed-point number has been placed in AMQ by the programmer using the floating-point RADD order*.

A summary of all Dual System commands are to be found in the next section in tabular form. Only those orders which are actually performed in the floating-point system are included in the first table.

(*) See Appendix for details on how the floating-point STORE and RADD orders are actually executed.

EXPLANATION OF FLOATING-POINT ORDERS

Decimal Code	Abbreviation	Name of Operation	Explanation
± 00	± STOP	Stop and Transfer	Stop the calculator and select as the next order the contents of the storage whose address is the address part of the STOP order. This next order and those succeeding are executed in fixed point if the sign of the STOP order is positive, and in floating-point if the sign is negative.
± 01	± TR	Transfer	Select as the next order the contents of the storage whose address is the address part of the TR order. This next order and those succeeding are executed in fixed point if the sign of the TR order is positive, and in floating point if the sign is negative.
± 03	± TR+	Transfer on Plus	If the contents of the AMQ carries a positive sign, select as the next order the contents of the storage whose address is the address part of the TR+ order, otherwise proceed with the next floating-point order. In the case the AMQ is positive, the next order and those succeeding are executed in fixed point if the sign of the TR+ order is positive, and in floating point if the sign is negative.

Decimal Code	Abbreviation	Name of Operation	Explanation
± 04	± TR 0	Transfer on Zero	If the contents of the AMQ is zero select as the next order the contents of the storage whose address is the address part of the TR 0 order, otherwise proceed with the next floating-point order. In the case the AMQ is zero the next order and those succeeding are executed in fixed point if the sign of the TR 0 order is positive, and in floating point if the sign is negative.
- 05	-SUB	Subtract	The contents of the full word storage whose address is the address part of the SUB order is subtracted in floating-point from the contents of the AMQ, the result remaining in the AMQ.
- 06	-RSUB	Reset and Subtract	The floating-point number in the full word storage whose address is the address part of the RSUB order is placed negatively in the AMQ.
- 07	-SUBAB	Subtract Absolute Value	The absolute value of the contents of the full word storage whose address is the address part of the SUBAB order is subtracted in floating point from the contents of the AMQ, the result remaining in the AMQ.
- 08	-No Op	No Operation	No operation is performed and the sequence of floating-point orders proceeds uninterrupted.
- 09	-ADD	Add	The contents of the full word storage whose address is the address part of the ADD order is added in floating-point to the contents of the AMQ, the result remaining in AMQ.

Decimal Code	Abbreviation	Name of Operation	Explanation
- 10	-RADD	Reset and Add	The floating-point number in the full word storage whose address is the address part of the RADD order is placed in the AMQ.
- 11	-ADDAB	Add Absolute Value	The absolute value of the contents of the full word storage whose address is the address part of the ADDAB order is added in floating point to the contents of the AMQ, the result remaining in the AMQ.
-12	-STORE	Store	The floating-point number in the AMQ is rounded and stored in the full word storage whose address is the address part of the STORE order.
-13	-STORE A	Store Address	The function routine which begins at the location whose address is the address part of the STORE A order is activated with the contents of the AMQ as argument and the answer being stored in the AMQ.
-14	-STORE MQ	Store Contents of MQ	Performs the same function as the STORE order.
-15	-LOAD MQ	Load MQ Register	Performs the same function as the RADD order.
-16	-MPY	Multiply	The contents of the full word storage whose address is the address part of the MPY order is multiplied in floating point by the contents of the AMQ, the product remaining in the AMQ.

Decimal Code	Abbreviation	Name of Operation	Explanation
-18	-DIV	Divide	The contents of the AMQ is divided in floating point by the contents of the full word storage whose address is the address part of the DIV order, the quotient remaining in the AMQ.

NOTE: If any order 5 through 18 carries a positive sign this is a breakpoint signal. The operation will proceed as if it had a negative sign; however, if switch #1 is in standard position, the order location, the operation and address part of the order, and the contents of the AMQ will be printed out. If switch #1 is placed in the down position there will be no effect from the positive sign.

SPECIAL CALLING SEQUENCES

(all commands are assumed outside of the blocks of interpreted orders except where otherwise noted).

To Enter Floating-Point Interpretation

i	+	RADD	i
i+1	+	TR	1800*
i+2		first order to be interpreted as floating-point command	

TO Print (in decimal) the Floating-Point Numbers Starting at Full Word Storage b and Ending with Full Word Storage d, with n Words Per Line of Printing. ($n \leq 6$)

j	+	RADD	j
j+1	-	TR	1475*
j+2	+	(n)	b
j+3	+	STOP	d+2
j+4		next order after printing completed (executed in fixed point)	

(*) These addresses will be altered when the Dual System is relocated.

To Print (in decimal) the Fixed-Point Numbers Starting at Full Word Storage b and Ending with Full Word Storage d, with n Words Per Line of Printing ($n \leq 6$)

k	+	RADD	k
k+1	+	TR	1475*
k+2	+	(n)	b
k+3	+	STOP	d+2
k+4		next order after printing completed (executed in fixed point)	

(*) These addresses will be altered when the Dual System is relocated.

COMPARISON OF FLOATING AND FIXED POINT POINT COMMANDS IN THE DUAL SYSTEM

Decimal Code	Abbreviation	Function in Fixed Point	Function in Floating Point
00	STOP	Stop and Transfer	with (+) sign: Stop and transfer outside of interpretational scheme (STOP OUT).
			with (-) sign: Stop and transfer within interpretational scheme (STOP IN).
01	TR	Transfer	with (+) sign: Transfer outside of interpretational scheme (TR OUT).
			with (-) sign: Transfer within interpretational scheme (TR IN).
02	TR OV	Transfer on Overflow	No Counterpart
03	TR+	Transfer on Plus	with (+) sign: Transfer on Plus outside of interpretational scheme (TR+ OUT).
			with (-) sign: Transfer on Plus within interpretational scheme (TR+ IN).
04	TR 0	Transfer on Zero	with (+) sign: Transfer on Zero outside of interpretational scheme (TR 0 OUT).
			with (-) sign: Transfer on Zero within interpretational scheme (TR 0 IN).

Decimal Code	Abbreviation	Function in Fixed Point	Function in Floating Point
05	SUB	Subtract	Subtract in Floating
06	RSUB	Reset and Subtract	Reset and Subtract in Floating
07	SUBAB	Subtract Absolute Value	Subtract Absolute Value in Floating.
08	NO OP	No Operation	No Operation
09	ADD	Add	Add in Floating
10	RADD	Reset and Add	Reset and Add in Floating
11	ADDAB	Add Absolute Value	Add Absolute Value in Floating
12	STORE	Store Acc	Store contents of AMQ
13	STORE A	Store Address from Acc	Activate specific sub-routines computation (see p. 18)
14	STORE MQ	Store Contents of MQ Register	Store contents of AMQ
15	LOAD MQ	Load MQ Register	Reset and Add in Floating
16	MPY	Multiply	Multiply in Floating
17	MPY R	Multiply and Round	No Counterpart
18	DIV	Divide	Divide in Floating
19	ROUND	Round	No Counterpart
20	L LEFT	Long Left Shift	" "
21	L RIGHT	Long Right Shift	" "
22	A LEFT	Accumulator Left Shift	" "
23	A RIGHT	Accumulator Right Shift	" "

Decimal Code	Abbreviation	Function in Fixed Point	Function in Floating Point
24	READ	Prepare to Read	" "
25	READ B	Prepare to Read Backward	" "
26	WRITE	Prepare to Write	" "
27	WRITE EF	Write End of File	" "
28	REWIND	Rewind Tape	" "
29	SET DR	Set Drum Address	No Counterpart
30	SENSE	Sense and Skip or Control	" "
31	COPY	Copy and Skip	" "

TABLE OF VIOLATIONS CAUSING A CALCULATOR STOP

Stop At Order Location	
1769	Exponent overflow: (exponent + 50) > 99
1769	Square root extraction attempted for negative number
1769	Logarithm computation attempted for zero
1769	Trigonometric function argument exceeds the maximum absolute value range: 10^9
1769	Logarithm argument is beyond the maximum absolute value range: 10^{43} or the minimum absolute value range: 10^{-43}
1897	Attempted division by zero*

(*) Will also turn on Division Check Light on Console of the 701.

TABLE OF "STORE A" ADDRESSES*

A STORE A order in the Dual System will activate the following functions if the appropriate address parts are given:

STORE A	1240	cos a
STORE A	1245	sin a
STORE A	1048	$\sqrt{a}$
STORE A	1154	arctan a
STORE A	1386	e^a
STORE A	1080	$\log_e a$
STORE A	2013	-a
STORE A	2007	$\|a\|$
STORE A	2010	$-\|a\|$

where a is the floating-point number in the AMQ.

STORE A	2025	floating-to-fixed conversion (where the floating-point number is that in the AMQ and the resulting fixed-point number is in a simulated floating-point form so that the Dual System STORE order will store it properly).
STORE A	2016	fixed-to-floating conversion (where the fixed-point number has been placed in the AMQ by the Dual System RADD order and resulting floating-point number is placed in the AMQ.

(*) These addresses will be altered when the Dual System is relocated.

APPENDIX. Precise Construction of the AMQ Register and Related Operations in the Dual Coding System

The concept of the "AMQ Register" is a useful one in programming a problem for the Dual System. It may be thought of as a single 36 bit register (including sign), like the Acc and MQ which have been built into the arithmetic unit of the 701 Calculator. However, this is not actually the case.

As was pointed out in the body of this write-up, the AMQ serves as a repository for results of floating-point operations as well as a source of one of the numbers which enter into such calculations, and therefore can be thought of as the center of all operations in floating point.

Since it is necessary prior to the performance of floating-poing orders, to actually separate the exponent and significant parts of the numbers, the quantity preserved in the AMQ is retained in this separated form; hence the AMQ must, of necessity, occupy more than one full word of the Electrostatic Storage, of which it is actually a part.

When some quantity is in the "AMQ Register", its first seven bits (normally an exponent) is stored in the first seven bits of a specific half-word of Electrostatic Storage, designated L_X. The sign of this half-word (which is never used in relation to these seven bits) is, in general, not related to the number which is presently in the AMQ. The remaining bits of this quantity in the AMQ (usually the significant bits of a floating-point number) are stored in a specific full word of Electrostatic Storage, designated L_A, with the first of these remaining bit positions occupying the first bit position of L_A, i.e., if the quantity is a floating-point number, its significant part occupies the first 28 positions. The sign of this full word is the sign of the quantity presently in the AMQ.

When a floating-point RADD order is executed, the first seven bits (the exponent) are placed in L_X, and the remaining 28 bits (the significant part) are placed in the first 28 positions of L_A. The sign of the (floating-point) number then becomes the sign of L_A. After the completion of any other arithmetic order, the results are stored similarly; however, it might be possible that the significant part of the answer exceeds 28 bits, in which case they are all preserved up to the full 35 bit capacity of L_A. The floating-point STORE order reassembles the floating-point number and stores it in the designated location. Prior to the assembly it rounds and then truncates the significant part to 28 bits.

In the conversion of numbers from floating-to-fixed, the resulting fixed-point number (which occupies a full 35 bits plus sign) is separated as if it were a floating-point number, with its first seven bits being placed in L_X and the remaining 28 being placed in the first 28 positions of L_A. This separation is performed to enable the programmer to store this fixed-point number with the floating-point STORE order. The conversion of numbers from fixed-to-floating assumes that the fixed-point number has been called forth by the floating-point RADD order, in order that the programmer can conveniently initiate this conversion process. Before the conversion is begun, the fixed-point number will be assembled back into one full word.

H. Kolsky
T-5

Additions to Dual Manual (LA 1573)

1. Notes on printing:

   If a floating point print is called for, the numbers to be printed must have exponents less than 100.

   In the example shown in LA 1573, the number of words to be printed is a multiple of the number per line. If, however, these same numbers were to be printed four per line, the last two words of the second line would be those contained in the two locations immediately succeeding those specified. These two numbers must lie within the floating decimal scheme.

2. Notes on improper orders:

   Orders $(19\text{-}31)_{10}$, interpreted as Dual orders, produce stops at locations $(3138\text{-}3151)_{10}$, $(6102\text{-}6117)_{8}$, $(71F0171\text{-}71F0184)_{R}$. After correcting such an order, one may proceed with the corrected order by manually transferring to $(7452)_{8}$.

3. Additional information.

Name	Contents	Octal Address
AMQ	Exponent	7757
	Fraction	7762 - 7763
Instruction Counter	Address of current instruction	7452
Instruction Register	Current instruction	7474
Trace test number	R Add - - - -	7433
Tracing Instruction Counter (Fixed point)	Address of current instruction	0117
Tracing Instruction Register (Fixed point)	Current Instruction	0174

4. Notes on Functions:

Sin A

Range	Approximation
$10^6 > \|A\| > 10^{-4}$	$\frac{\|A\|}{A}\left[c_1 y + c_3 y^3 + c_5 y^5 + c_7 y^7 + c_9 y^9\right]$
$10^{-4} \geq \|A\|$	A
$\|A\| \geq 10^6$	Out of Range.

The reduced argument y is obtained from the equation $A = (n + y)\frac{\pi}{2}$ where n is an integer and y a positive pure fraction. The c_i's are the coefficients in the Rand Approximation for $\sin \frac{\pi}{2} y$,

$$c_1 = 1.57079631847$$
$$c_3 = -.64596371106$$
$$c_5 = .07968967928$$
$$c_7 = -.00467376557$$
$$c_9 = .00015148419$$

Cos A

$10^6 > A > 10^{-6}$	$\sin(\frac{\pi}{2} - A)$
$10^{-6} \geq A$	1
$A \geq 10^6$	Out of Range.

e^A

$50 \ln 10 > \|A\| > 10^{-11}$	$10^{I+1} \cdot 4 \left[a_0 + a_1 r + a_2 r^2 + a_3 r^3 + a_4 r^4 + a_5 r^5 + a_6 r^6 + a_7 r^7\right]^2$
$10^{-11} \geq \|A\|$	1
$\|A\| \geq 50 \ln 10$	Exponent overflow or underflow

where the a_i's $= 1/2\sqrt{10}\ (4/\ln 10)^i c_i$, c_i being the Rand coefficients for 10^x.

$$c_0 = 1.00000000000$$
$$c_1 = 1.15129277603$$
$$c_2 = .66273088429$$

$$c_3 = .25439357484$$
$$c_4 = .07295173666$$
$$c_5 = .01742111988$$
$$c_6 = .00255491796$$
$$c_7 = .00093264267$$

I is the integral part of $A/\ell n10$ and r is defined by by $A/\ell n10 = I + 4r/\ell n10$ so that r is positive.

Sinh A

$\lvert A\rvert > 10^{-4}$	$\dfrac{e^A - e^{-A}}{2}$
$10^{-4} \geq \lvert A\rvert$	A

Cosh A

$\lvert A\rvert > 10^{-4}$	$\dfrac{e^A + e^{-A}}{2}$
$10^{-4} \geq \lvert A\rvert$	1

$\ell n|A|$

$|A| = x \cdot 10^{a-50} \neq 0$

$$(a - 50)\ell n10 - (K + 1/2)\ell n2 + 4z(1/2 + z^2/6 - \frac{z^4 c_1}{5z^2 - 10c_1})$$

where $c_1 = .69946289$,

$$z = \frac{2^K x - \sqrt{2}/2}{2^K x + \sqrt{2}/2},$$

and K is such that $1 > 2^K x \geq 1/2$.

Arth A

$1 > \lvert A\rvert > 10^{-4}$	$\dfrac{\lvert A\rvert}{2A}\ell n\dfrac{1 + \lvert A\rvert}{1 - \lvert A\rvert}$
$10^{-4} \geq \lvert A\rvert$	A
$\lvert A\rvert \geq 1$	Out of Range.

Art A

$1 > |A| \geq 2 - \sqrt{3}$ $\qquad \frac{|x|}{x}\pi/6 + 2z_1\left(1/2 - \frac{z_1^2}{6} + \frac{z_1^4 c_1}{5z_1^2 + 10c_1}\right)$

$2 + \sqrt{3} \geq |A| > 1$ $\qquad \frac{|x|}{x}\pi/3 + 2z_2\left(1/2 - \frac{z_2^2}{6} + \frac{z_2^4 c_1}{5z_2^2 + 10c_1}\right)$

$2 - \sqrt{3} > |A| > 10^{-4}$ $\qquad 2A\left(1/2 - \frac{A^2}{6} + \frac{A^4 c_1}{5A^2 + 10c_1}\right)$

$10^{10} > |A| > 2 + \sqrt{3}$ $\qquad \frac{|A|}{A}\pi/2 - \frac{2}{A}\left(1/2 - \frac{(\frac{1}{A})^2}{6} + \frac{(\frac{1}{A})^4 c_1}{5(\frac{1}{A})^2 + 10c_1}\right)$

$|A| < 10^{-3}$ $\qquad A$

$|A| > 10^{10}$ $\qquad \frac{|A|}{A}\pi/2$

$$z_1 = \frac{x - |x| / x\sqrt{3}}{1 + |x| / \sqrt{3}}$$

$$z_2 = \frac{|x| / x\sqrt{3} - 1/x}{1 + 1/|x|\sqrt{3}}$$

$$c_1 = .7030487$$

H. Kolsky
T-5

DUAL TRACING MODIFICATION #2: Traces and prints specific addresses while problem is in operation.

DESCRIPTION: This program is contained on 3 binary cards, two of which modify the Dual Tracing program as contained on the Dual CH cards 01-06, and therefore must be loaded after the Dual Tracing program. The third card contains an additional 28 orders. This program will trace instructions which have addresses between and inclusive of the two addresses specified, with only one pair of addresses given at one time.

INPUT: Put the first address for which the search is to be made in location 2F27 and the second in 2F28. With sense switch 2 down, the program will trace and print those floating point instructions containing as addresses the addresses specified. With sense switch 3 down, this program will trace and print those fixed point instructions containing as addresses the addresses specified and occurring after the execution of a positive transfer floating point operation. With switches 2 and 3 up, the machine will operate at full speed. The tracing output is the same as that of Dual Two.

There are two origins, 2F and 3F. The 3F region contains two orders which modify Dual. 3F54 and 3F138 must be located where Dual 73F2 and 73F86 are located respectively. To be used with Dual as presently located, 3F54 and 3F138 must have absolute locations 54 and 138 respectively. 2F0-2F28 is located as shown below, but it may be located any place that does not interfere with Dual or Dual Tracing.

Laslik

2F0 - $(222)_{10}$ - $(336)_8$

2F28 - $(250)_{10}$ - $(372)_8$

First add 2F27 - $(249)_{10}$ - $(371)_8$

Last add 2F28 - $(250)_{10}$ - $(372)_8$

CODED: W.W. Wood, checked, W.W. Wood

H. Kolsky
T-5

Dual Punch (4 cds)

Description: This program converts and punches dual numbers as CPC spread-read-in cards. The numbers are converted and punched as proper CPC floating decimal numbers in fields 1 through 7.

There are 137 orders which may be located in conjunction with dual, over dual check, or over dual proper. It is entered by a fixed point calling sequence as follows:

α	R Add	α
$\alpha + 1$	tr	Fo
$\alpha + 2$	n	FWA
$\alpha + 3$	stop	LWA-2
$\alpha + 4$	---------------	

Where $o < n \leq 7$ is the number of words to be punched per card, $\alpha + 4$ is the next order to be performed.

Storage:

	Eo	$(0000 - 0060)_{10}$	$(0000 - 0074)_8$
A region (over dual check)	Fo	$(0062 - 0199)_{10}$	$(0076 - 00307)_8$
B region (with dual)	Fo	$(2888 - 3025)_{10}$	$(5510 - 5721)_8$
C region (over dual proper)	Fo	$(3584 - 3721)_{10}$	$(7000 - 7211)_8$

Additional dual stops: Orders $(19 - 31)_{10}$ interpreted as dual orders produce stops at $(71F0171 - 71F0184)_R$, $(3138 - 3151)_{10}$, $(6102 - 6117)_8$.

Correction: Dual Check occupies $(0052 - 0221)_{10}$.

H. Kolsky
T-5

MODIFICATION TO DUAL II TRACING (W. W. WOOD)

DESCRIPTION: For our own convenience in GMX-DO, we have programmed the following modification to Dual II Tracing, which may be of use to others. The tracing test feature is replaced in such a way as to allow discrete blocks of the program to be traced (i.e. a tracing is printed), the length and number of blocks being at the option of the coder. The information appearing in the tracing is the same as in the regular dual tracing, and the control by means of switches 2 and 3 is unaltered. A similar restriction on effecting fixed point tracing holds, namely, fixed point instructions will be traced if switch 3 is down, but only after one floating point positive transfer order has been executed with switch 3 down.

LOADING: This modification is contained on three binary cards, (labeled ck. modif) two of which modify the tracing program as punched on the standard cards CK01 - 06 of the dual deck, and therefore must be loaded after the latter. The third card contains an addition of 35 orders.

INPUT: The input is the specification of the blocks of coding to be traced, in the form of an ascending sequence of addresses (entered as + R ADD orders) grouped in pairs starting at 1F35.

The instrucions to be traced are those located between the locations specified by each pair of addresses, excluding the end-points. The instructions not traced are those in locations from 0000 to the first address of the first pair, inclusive, and those from the last address of each pair to the first address of the next pair. This series of locations must be terminated, in either pair position, by a location which is certain not to be exceeded by the locations of the code being traced, e.g. 7777_8. At least one, two, or three addresses must be specified for proper functioning with switches 2 and 3 down.

TIMING: With a small number of block specifications, the use of this tracing program does not seem to materially increase the program time for floating point orders. For fixed point, however, the traversals of the tracing interpretation loop considerably increase the time, even when no printing is done. Of course, with switch 3 up, fixed point calculations proceed at full machine speed.

LOCATIONS: There are two origins, OF and 1F. OF refers to Dual II Tracing, and in this program consists of seven instructions which modify Dual II. For Dual II as presently located, OF0000 = 0000. The 1F program consists of 35 instructions with the above described pairs of locations following; it can be located as desired. The program is available in binary with OF0000 = 0000, 1F0000 = $(0222)_{10}$.

CODED AND CHECKED: W. W. W.

Dual

5. RELOCATION

The entire Dual Coding System is presently in Regional Coding and therefore may be placed at will by the coder. It is necessary to include the three fundamental sections of the system: The Arithmetic Program (71F), the Transfer Program (72F), and the Print Program (75F). The balance of the programs may be introduced then, whenever a problem warrants. In Table VI of Section II can be found a complete list of these programs along with their entrances and other pertinent information.

6. PROGRAM CHECK-OUT

6.1 Break Point. In order to facilitate checking problems, a break-point has been introduced on all floating-point commands excluding transfer commands. If switch #1 is in normal position the printer will print the following decimal information after executing a positive floating-point order (which is not a transfer order):

Location of Order   Operation and Address   Contents of AMQ

If switch #1 is placed in its down position the calculator will proceed at full speed.

6.2 Tracing Program

6.2.1 Description and General Use. If it is desired to further inspect the course of a calculation, it is possible to trace the individual orders of the program. This is made possible by the introduction of a special Tracing Program which occupies 158 half-words of memory, (presently in locations 52 - 209). When this program has been loaded, Switch #2 controls the tracing of floating-point orders, and Switch #3 the tracing of fixed-point orders. When either (or both) of these switches are in the down position the corresponding type of command will be traced, i.e., the following information will be printed.

For Floating-Point Commands (with Switch #2 Down):

Location of Order	Operation and Address	Contents of AMQ	Contents of Addressed Storage as Fl-Pt Number

For Fixed-Point Commands (with Switch #3 Down):

±	Location of Order	Operation and Address	Acc Intepreted as a Command	Ov Bits	Acc Interpreted as a ten Digit Fraction	MQ Interpreted as a ten Digit Fraction	Addressed Storage Interpreted as a Ten Digit Fraction

Minus Sign Indicates Overflow Indicator Is On

Aside from the control of Switches 2 and 3, the programmer has a further control over what is to be traced. If an address is stored in 3867 (72F 45), the Trace Test Number Location, (which must always retain its RADD operation part), only those commands (both fixed-and floating-point) which reside in locations whose addresses exceed that address stored in the Trace Test Number Location, will be traced. If no number has been stored as the Trace Test Number, all orders will be traced when Switches 2 and 3 are set appropriately.

6.2.2 Starting, Breaking and Restarting the Tracing Sequence. In order to initiate the tracing procedure for either fixed- or floating-point commands, it is necessary only to execute one floating-point command. When this is accomplished, all subsequent orders may be traced dependent on the control of the Sense Switches and the Trace Test Number. However, due to timing considerations, all programs containing a Read, Write, or Read Backwards commands will be executed at full machine speeds (without being traced) after the Read, Write, or Read Backward command is reached, and the tracing will then be continued with the next floating-point command. Since the

Tracing Program utilizes the same basic Print Program as is used for data printing, the same control panel for the printer suffices for both tracing and data printing; hence these two procedures may occur intermixed in the course of checking a code.

6.2.3 Non-print of Executed Transfers. An idiosyncracy of the Tracing Program does not allow it (except at the expense of increasing its size) to print the associated information for an unconditional transfer or for a conditional transfer which is executed. However, due to the change in the sequence of order selection, this should not be a hardship as long as one has available a list of all commands in the program.

TABLE III

TABLE OF CONDITIONS CAUSING A CALCULATOR STOP

Stopping Location: Absolute	Stopping Location: Regional	Condition
$(3956)_{10}$ - $(7564)_8$	71F 0080	Exponent Overflow (1)
$(3951)_{10}$ - $(7557)_8$	71F 0075	Exponent Underflow (1)
$(3520)_{10}$ - $(6700)_8$	74F 0012	Exponent too large for floating-to-fixed conversion (4)
$(3826)_{10}$ - $(7362)_8$	72F 0004	Programmed Stop (4)
$(4028)_{10}$ - $(7674)_8$	71F 0152	Programmed Stop (4)
$(0068)_{10}$ - $(0104)_8$	73F 0016	Programmed Stop when tracing (4)
$(0103)_{10}$ - $(0147)_8$	73F 0051	Programmed Stop when tracing (4)
$(3990)_{10}$ - $(7626)_8$	71F 0114	Attempted division by zero (2) (4)
$(3475)_{10}$ - $(6623)_8$	81F 0003	Square root of negative (3)
$(3469)_{10}$ - $(6615)_8$	86F 0105	Trigometric function argument exceeds the maximum absolute value range: 10^9 (1)
$(3119)_{10}$ - $(6051)_8$	84F 0045	Argument of hyperbolic arctangent equals or exceeds one. (1)

(1) Pressing START button prints out break point information for the order which produced the untenable condition and calculation recommences.

(2) Divide Check light is turned on, also.

(3) Program continues with absolute value of argument.

(4) Pressing START button simply causes the calculation to recommence.

TABLE IV

TABLE OF "STORE A" ADDRESSES

A STORE A order in the Dual System will activate the following functions if the appropriate address parts are given (where a is the floating-point number in the AMQ):

Absolute Address	Regional Address	Function
3472	81F 0000	$\sqrt{a}$
3364	86F 0000	cos a
3369	86F 0005	sin a
3268	85F 0000	e^a
3204	82F 0000	$\log \lvert a \rvert$
3124	83F 0000	arctan a
3074	84F 0000	arctanh a
3026	87F 0000	cosh a
3037	87F 0011	sinh a
4016	71F 0140	$\lvert a \rvert$
4019	71F 0143	$- \lvert a \rvert$
4022	71F 0146	- a
3516	74F 0008	floating-to-fixed conversion (where the floating-point number is that in the AMQ and the resulting fixed-point number is in a simulated floating-point form so that the Dual System STORE order will store it properly).
3508	74F 0000	fixed-to-floating conversion (where the fixed-point number has been placed in the AMQ by the Dual System RADD order and resulting floating-point number is placed in the AMQ.

TABLE V

SPECIAL CALLING SEQUENCES

(all commands are assumed outside of the blocks of interpreted orders except where otherwise noted).

To Enter Floating-Point Interpretation

i	+	RADD	i	
i+1	+	TR	3876*	(71F 0000)
i+2		first order to be interpreted as floating-point command		

TO Print (in decimal) the Floating-Point Numbers Starting at Full Word Storage b and Ending with Full Word Storage d, with n Words Per Line of Printing. ($n \leq 6$)

j	+	RADD	j	
j+1	-	TR	3536*	(75F 0000)
j+2	+	(n)	b	
j+3	+	STOF	d+2	
j+4		next order after printing completed (executed in fixed point)		

(*) These addresses may be altered if the Dual System is relocated.

To Print (in decimal) the Fixed-Point Numbers Starting at Full Word Storage b and Ending with Full Word Storage d, with n Words Per Line of Printing ($n \leq 6$)

k	+	RADD	k	
k+1	+	TR	3536*	(75F 0000)
k+2	+	(n)	b	
k+3	+	STOP	d+2	
k+4	next order after printing completed (executed in fixed point)			

(*) These addresses may be altered if the Dual System is relocated.

TABLE VI

REGIONAL INFORMATION

PROGRAM	INDEX AND EXTENT	ENTRANCES	STOPS
Trigonometric	86F 0000 - 86F 0107	cos - 86F 0000 sin - 86F 0005	86F 0105 - Range violation*
Exponential	85F 0000 - 85F 0094	exp - 85F 0000	None
Logarithm	82F 0000 - 82F 0063	log - 82F 0000	None
Arctangent	83F 0000 - 83F 0079	art - 83F 0000	None
Arctanh	84F 0000 - 84F 0048	arth - 84F 0000	84F 0045 - Argument exceeds one
Hyperbolic	87F 0000 - 87F 0047	cosh - 87F 0000 sinh - 87F 0011	None
Square Root	81F 0000 - 81F 0035	sq. r. - 81F 0000	81F 0003 - Negative argument
Fixed and Floating	74F 0000 - 74F 0027	fixed-to-floating -74F 0000 floating-to-fixed -74F 0008	74F 0012 - Exponent exceeds reference exponent
Arithmetic	71F 0000 - 71F 0219	floating-point -71F 0000	71F 0075 - Exponent underflow 71F 0080 - Exponent overflow 71F 0114 - Division by zero 71F 0152 - Programmed stop
Print	75F 0000 - 75F 0285	fixed or floating print -75F 0000	None
Break Point & Tr	72F 0000 - 72F 0053	no direct entrance	72F 0004 - Programmed stop
Tracing	73F 0000 - 73F 0157	no direct entrance	73F 0051 - Programmed stop 73F 0016 - Programmed stop

*Due to limitations imposed by a finite number of significant figures, it is necessary to restrict the range of the arguments of trigonometric functions.