MEETING OR CONTACT REPORT

Date of Report: May 1, 1959

Organization & Location: Los Alamos Scientific Laboratory Poughkeepsie, New York	Date: April 27-28, 1959 Reported By: R. S. Ballance H. G. Kolsky
Project: Stretch Mathematical Planning Group Meeting #15	Department: 749 Follow-up Date:

PERSONNEL PARTICIPATING:
(Place asterisk next to those on distribution list. Other distribution show at end of report)

LASL

B. Carlson
E. A. Voorhees
D. F. Woods
J. Worlton
R. B. Lazarus
H. G. Kolsky*
E. W. Coffin*
R. S. Ballance*
G. A. Blaauw*
W. Buchholz*
S. W. Dunwell*
W. Y. Stevens*
E. Bloch*
E. I. Jordan*
R. T. Blosk*
G. T. Paul*
J. F. Dirac*
F. E. Johnston*
G. A. Grover*

A. Progress Report and Present State of SIGMA

I. Concerning Hardware

Mr. Dunwell opened the meeting with a brief report on the status of the SIGMA machine. The project has been lacking memories, but the 2.0 microsecond memory is now working with more than 32 planes at a 2.0 microsecond total cycle time. Two more frames are being built under laboratory supervision. Cross talk in the planes was the hold-up and was caused by not enough care in assembly. The trouble can now be identified in inspection and more care can be taken in manufacturing. IBM will produce at least 20 memories this year. Power driver transistors still present a little problem in getting the yield we need, but we can always add more people. The Exchange is now very effective. We can write 256,000 words of a repetitive pattern on tape and then read it back. In a recent test the Exchange ran for 65 hours with 5 errors. Four out of five of these were due to voltage surges caused by other equipment overloading the generator. The equipment

to be delivered will have crank-up and crank-down controls which will avoid this difficulty. The other error was due to a timing problem which has now been corrected. We expect it to run for a week without errors soon. In the present model of the Exchange there are still transistor troubles due to poor lead connections. This will be corrected when the Exchange is rebuilt. Two frames have been started in the plant: a memory bus and an Exchange. We use 704 and 705 programs to draw diagrams and maintain records. This reduces the lag between laboratory and plant to a matter of weeks. Almost every frame of the laboratory model is under construction. Since most of the components and parts comes from the plant or outside, assembly can be very rapid. We have found that the double slide is too crowded and are substituting single slide frames. In each case the slide can be transferred and does not need to be rewired. The machine is at present over 50% built.

We now have a model in rack form of the index core memory. The model just checks the feasibility. The principal problems are related to noise. If this model does not work at speed (0.4 microseconds read, 0.8 microseconds write) we will build something else. We will not slow down the memory.

II. Concerning Plans

Westinghouse and GE have been quoted sales prices, although there is no actual sales program at present. Contact renogotiations will start soon. Since the machine does not precisely fit the original contract we expect IBM to approach the Atomic Energy Commission within 30 days. We originally planned to build two machines in the laboratory. We now plan to have the Los Alamos machine built in the plant.

There is no chance of getting an earlier delivery by taking the laboratory model. The date of delivery is still as stated in the original contract. We are firm that Los Alamos will do its acceptance testing in Poughkeepsie and that we will not ship the machine unless that testing is successfully completed.

B. General Items of Business

I. Schedule of Computer Availability

Dr. Kolsky presented the schedule by which the various components of the STRETCH central processing unit will become available. The schedule is reproduced in Appendix I.

The present best estimates for the dates are:

Phase I	October 1959
Phase II	December 1959
Phase III	March 1960

LASL was asked if they would consider allowing IBM Applied Programming to use their STRETCH after it is delivered to Los Alamos so that Applied Programming can complete the development of their programming systems.

Mr. Carlson replied that LASL would be happy to consider such an agreement. It should be done as an agreement separate from the AEC contract.

Mr. Voorhees asked if IBM would make 704 time available to LASL during their problem checkout time here. The answer was yes, a reasonable amount of 704 time will be made available on one of the machines in Poughkeepsie.

II. Patent Restrictions Concerning STRETCH

Dr. Lazarus is scheduled to participate in a symposium at the International Computer Conference in Paris in June 1959. A separate meeting was held with Mr. Lester, an IBM patent attorney, to discuss the patent-sensitive areas of STRETCH with Dr. Lazarus. The conclusion was that it is impossible to give a blanket approval of any topics which might arise during the symposium, but that we would have to rely on Dr. Lazarus' discretion to avoid revealing any information which might prejudice IBM's foreign patent position.

III. Expanded Character Set I/O Devices

Dr. Buchholz described a possible set of codes for use with input-output devices using more than 48 characters.

Mr. Voorhees described the LASL 3-case typewriter with half spacing up and down to a smaller group. This is the device which is presently being built by SEPD.

IV. STRAP Mnemonics

Mr. Gover discussed the recent changes in STRAP Mnemonics and gave the reasons behind them. The manual is being rewritten using them.

V. Manuals, Schools, etc.

In the AEC contract the following manuals are specified:

1.	Installation Manuals	5 copies	July 20, 1959
2.	Operation Manuals	50 copies	May 20, 1959
3.	Programming Manuals	50 copies	May 20, 1959
4.	Maintenance Manuals	10 copies	Nov. 20, 1960

At present LASL has 26 copies of the Operation Manual and approximately the same number of programming (STRAP) Manuals. The installation Manuals will be furnished by July 20--preliminary installation information has already been given.

The question was asked if LASL wished IBM to issue the additional operating and programming manuals now.

The answer was IBM should wait until the planned revisions are completed, and until LASL requests more copies.

The contract also calls for IBM to hold STRETCH classes in Poughkeepsie for 20 LASL personnel for 3 weeks. The classes are to start before May 20, 1959.

In the discussions which followed LASL indicated that they did not intend to send so large a group to Poughkeepsie. Instead they plan to teach classes in STRETCH programming at Los Alamos. They will ask IBM to send instructors to Los Alamos for this purpose later this summer. The date will be selected within a month.

C. Discussion of Questions in Dr. Lazarus' Letter

I. Questions Concerning Operation

On March 27, 1959, Dr. Lazarus wrote IBM a letter raising a number of questions concerning the operation of STRETCH in certain special cases and making a number of suggestions for new instructions to be included in the machine. One of the main items of business of the meeting was the discussion of these questions and suggestions. Most of

them were answered from a Project 7000 File Memo dated April 9, 1959. A copy of this memo is attached to this report as Appendix II.

During the discussions, other questions of a similar type were raised, they have been compiled in Appendix III.

II. Questions Concerning Hardware Changes

During the detailed discussions the following points were singled out for serious study and inclusion in the STRETCH design if possible.

1. Bits 60-63 in single precision floating point should be left alone instead of set to zero as presently specified.
2. SWAP and TRANSMIT instructions which step their addresses by -1 as well as +1. (The half word transmit order can be sacrificed if necessary.)
3. The Harvest indicators should be wired in for programmers' use.
4. The Floating Point SWAP order.
5. Some method of handling floating point zeros and exponent underflows without interrupt.
6. Load Exponent and Load Exponent Immediate.

Of the above, LASL feels most strongly about #5, exponent underflow.

Attachment 2
Brownline to: B. Carlson
P. Wilson

R. S. Ballance
Representative
Project 7000

H. G. Kolsky
Coordinator
Project 7000

APPENDIX I

SIGMA COMPUTER AVAILABILITY

Computer Functions	Phase I	Phase II	Phase III
1. Memory Bus			
2 usec Memory Addressing	X		
1/2 usec Memory Addressing			X
Basic Exchange tie-in	X		
High Speed Exchange tie-in		X	
Computer tie-in	X		
2. Instruction & Indexing Unit			
a. Instruction Execution			
Direct and Immediate Index Arithmetic	X		
Refill Operations		X	
Count and Branch Operations		X	
Load Indirect		X	
Execute and Rename Operations			X
Geometric Indexing			X
Store Address & Instruction Counter		X	
Transmission Instructions			X
Unconditional Branching	X		
Indicator Branching		X	
Index Branching	X		
Bit Branching		X	
b. Instruction Preparation			
Floating Point Instruction		X	
VFL Instructions	X		
I/O Instruction Preparation	X		
c. Miscellaneous			
Automatic Interrupt		X	
Elapsed Time Clock			X
3. Lookahead	X		
4. Arithmetic Unit			
a. VFL			
Binary Load/Add Compare	X		
Binary MPY/DIV			X
Decimal Arithmetic	X		
Convert			X
Connects	X		

Computer Functions	Phase I	Phase II	Phase III
b. Floating Point			
Load/Add/Compare	X		
Divide Type Operations	X		
MPY Type Operations			X
Square Root			X
Noisy Mode (All Op's.)			X

April 9, 1959
*Revised April 29, 1959

PROJECT 7000

FILE MEMO

SUBJECT: Letter from Dr. R. B. Lazarus of LASL
Dated March 27, 1959

Dr. Lazarus' letter lists a number of questions, remarks and requests concerning STRETCH. They may be classified into two catagories: (1) suggestions for hardware changes and new instructions, and (2) requests for information as to how the computer operates in certain special cases.

Some of the questions have been answered in telephone conversations with Dr. Lazarus, but the complete list is given here for the record.

A. Concerning Equipment:

I. Space Available

The number of operation codes not being used are, according to class:

half words

1.	Floating Point	3
2.	Miscellaneous	3
3.	Direct and Immediate Index	0
4.	Count & Br. and Indicator Br.	0 ~~X~~ (BB combination)
full words 5.	Unconditional Branch	2 ~~X~~ (enter stream)
6.	VFL Connect	2
7.	Input/Output	6
8.	Transmit	0 ~~1 15/16~~
9.	Branch	1 (enter stream)
10.	Miscellaneous	8
11.	~~Overflow~~ Indicator	32

Full length 4 tondar code (1000)

Many of the above have modifier bits belonging to the particular class of instruction so they actually represent many more cases. Care must be taken in how the above op-codes are considered "free" however, because the instruction decoding is done in several partial steps.

---

*Although revised after the April 27-29 LASL-IBM Mathematical Planning group meeting, this memo does not contain any of the decisions or changes made at the meeting.

Concerning the CONVERT instructions and Decimal arithmetic, the CONVERTS were added because they resulted in an appreciable simplification in hardware and made a powerful tool available to the programmer. Having only one CONVERT instruction which operated on the accumulator would require the programmer to write a number of extra load and store instructions to preserve the accumulator contents, etc.

Decimal VFL arithmetic enables the system to handle a whole class of problems directly without subroutines and is considered to be an important advantage in the STRETCH system, particularly in the commercial area.

II. New Orders Desired

In general, the STRETCH design effort must be considered well beyond the stage where new instruction sequences can be added. The truth tables of these items are complete and the logical design is almost finished.

Nevertheless, the suggested orders have been examined carefully by all the engineers involved and appraised for the hardware changes which would be required. The following answers are for the most part direct quotations from their comments:

1. Swap with Accumulator

Q: Floating Point order, half-word, with any interpretation of sign and normalization modifiers convenient to you. The equivalent program takes three half-word orders and an index word, or two half-word orders, a Transmit, and a memory word. (If the present SWAP addressing the accumulator takes the old sign from, and puts the new sign in, the Sign Register, then that is satisfactory.)

A: The present SWAP does not take the sign from the sign register but treats the accumulator as a memory location in the VFL sense. It can be programmed in three half-word orders without using an index: Load D from M, Store acc. to M, Load acc. from D. Although the order bears some relation to Floating Add to Memory or VFL Connect to memory, the sequences are quite different. It would be awkward to incorporate into hardware and probably slower than the above 3 half-word ops.

LFT M
ST M
L # FT

2. Store Floating Point Zero

Q: See discussion below of Negative Exponent Spill and Floating Point Zero. Assuming this problem is going to be solved somehow, it would be useful to have a half-word order for storing a floating point word with a large negative exponent and a zero mantissa.

A: Store Floating Point Zero appears to be similar to the Miscellaneous Store Zero instruction but would require non-trivial modification of existing data paths. A transmit instruction does it in a full word.

3. Set Exponent: Set Exponent Immediate

Q: These could be Floating Point or Miscellaneous. They would first set the accumulator exponent to zero and then Add Exponent. The equivalent program is Connect 0000 followed by Add Exponent, or, if it works, Add Exponent Negative, Address 8 followed by Add Exponent. This has fairly low priority but is presumably very easy for your engineers.

A: Add Exponent Negative, address 8, followed by add exponent does do this job rapidly. These orders would have been easy at one time but the accumulator is built. The spare Floating Point opcodes are not as easily decoded as "add to exponent" types.

4. Reflect Accumulator

Q: In its simplest form this changes accumulator bits according to the recipe bit n to position 127-n, for n = 0,1, ...63. In its best form it does this job for a specified field in the accumulator, leaving the rest of the accumulator unchanged. Note that the primary reason for wanting this order is to develop a right zeros count . Perhaps you can solve that problem in a direct way, such as developing a right zeros count on connect orders. Incidentally, this inversion can be performed on Maniac type machines by connecting the low order ends of two registers and shifting one register right while the other shifts left.

A: The latter solution is not possible in STRETCH since we do not have shifting registers of this type. The rotating of the accumulator can be done one bit at a time using a few VFL commands but is not fast. The Right zeros count can be found simply by connecting to get all-ones-count, subtracting one, then comparing new all-ones-count with the previous one.

5. Assorted Shift Orders

Q: There seem to be a number of jobs difficult to do with the connects alone. For simple displacements of fields shorter than 65 bits but much longer than 8 bits, connects will work but we are afraid they are too slow. Shifts of fields bigger than 64 bits are perhaps not too important. Circular shifts, especially of one field in the accumulator without distirbing the rest of the accumulator, are particularly difficult and may be rather useful. Note: by "shift" we only mean something given the same result as an old fashioned shift, and with reasonable speed.

A: There is a difference of philosophy apparent in these questions. (see remarks under 7 below). Recall that "old fashioned" shifts are not fast instructions on machines which use them. Also Circular shifts on existing machines are quite restricted in operation allowing no field length variations. STRETCH's connects are not slow in the light of what they do, the effect of a circular shift of any field in the accumulator without disturbing the rest of the accumulator can be programmed by storing the field twice in memory in adjacent locations then loading from the pair with the appropriate bit address.

6. Set Indicator to one

Q: If, as we are requesting below, we get some Programmer's Indicators (the Harvest indicators and/or the Transit register), then a fast half-word order to turn them on would be nice.

A: Since the VFL unit must always be used for this branch on bit or indicator branch is equivalent in result and speed. There are no op codes available. There are also complications concerning protection of the indicators since they are of a read-only type.

7. Extract

Q: This has been brought up before and seemed too difficult. It should be looked at again, now that you know your circuits. The order is defined as follows: where a mask word, C, has zeros, leave A alone; where the mask word has ones, take for the result the connection of A and M. This could be restricted to the connection 0011: Leave A where C is zero; use M where C is one. I believe that the logical statement of the general case is $A.\overline{C} \vee (A \oplus M).C$. The chief use of extraction is to insert into A assorted, disjoint fields from M.

A: Extracts imply the existance of a third register other than the accumulator and memory which feeds into the execution unit simultaneously. All the effects of extracting can be obtained with two or three VFL orders.

More fundamentally, extracting, masking and shifting are merely another, older technique of handling partial fields within a computer. The VFL philosophy was chosen for STRETCH over two years ago as being a more general and efficient method. Suggestions that we abandon VFL in favor of extracts or put in both systems at this late date are equivalent to saying we should start all over again on the logical design.

8. Generalized Swaps and Transmits

Q: There are two sources of slack in the Transmit order: bit 55 is not used much, and for Direct mode the value and refill fields of the index word are not used.

(1) The first generalization which comes to mind is to use the value field of the index word to step the addresses, instead of using an implied +1. Example: if we now Swap n-1 words between a and a+1, the list x1, x2, ...xn becomes x2, x3, ...x(n-1), i.e., a cyclic permutation in the other direction. There are also common cases, for certain approaches to storing arrays, for transmitting every kth word in one block to the corresponding positions in another block.

(2) The second generalization we have is to use bit 55 on Transmit direct to freeze the source address; the order then fills a block of memory with a constant value. This can be done by transmitting the the constant to a and then transmitting n-1 words from a to a+1, but we don't know whether or not this is excessively slow.

(3) The third idea is to use the Index Flag and Refill Field to do chaining; this makes sense only if combined with the use of the value field mentioned above.

A: The Transmit and Swap Immediate orders have been modified to handle full words instead of half words. A single half-word Transmit has been specified. Counting by -1 as well as +1 would have been easy to include at one time but require changes in circuitry and op codes now. The same applies to freezing the source address. The half-word transmit order would be in the way of using the op code bit needed. Using other than a count would get extremely awkward where the transmission involved the memory addresses of the index storage since the X register is used for all index words. Proper design of the transmit instruction would call for a slow down on all transmits due to the special actions that need be taken under these special conditions.

9. Connect to Sign Register

Q: The ability to move bits from memory to the flag and zone positions might be useful.

A: This instruction was in the design at an earlier date but was removed because of the hardware complexity. It can be done now in two instructions using the accumulator.

10. Locate List Element

Q: This suggestion is a rather late response to the determination made in the early days that Stretch should incorporate instructions useful for automatic coding. The desire is to generalize the Load Indirect order, which now terminates due to the flag or mark in the list element itself (i.e., op-code not that of Load Indirect). It now seems that this is less useful than an order which says Load Indirect to the nth Level. This, in turn, is less useful than an order which says Load Indirect until you

find element E. For this last, think in terms of a set of n full words arranged in memory in an disordered way(as must happen if an original ordered list is constantly being added to and subtracted from). Each such word would contain the address of the next word (as in Load Indirect). The last word might be marked, say, with an index flag bit. But each word would also have a field containing its name, and the order would specify a name and continue loading until it got an equality between the list element name and the specified name (or got to the end of the list). If such a Load Indirect, guided by a comparison or a count, seems feasible, then we have some thoughts on field assignment which I am not including here. If this does not seem feasible, how about recording somewhere the number of levels used by the Load Indirect, so that a by-product result would be the ordinal number in the list of the element loaded?

A: The functions desired here are "Harvest" type functions. They would require additional data paths not present in the machine. The idea of counting for each load indirect would require a revision of the instruction sequences for Load Indirect. The effect of "Load Indirect" to the nth level" and "Load Indirect until you find element E" can be done with relatively simple programs.

III. Miscellaneous Comments and Questions

1. Q: Does a VFL Store or Connect to Memory require a complete fetch of the old data word, or is the information merged at the memory unit? Does a two bit store to an address with bit address 63 require two full fetch bus times and two full store bus times?

A: No merging of information at the memory unit takes place. The example given requires two fetches and two stores. The main effect of this is on bus traffic. The machine normally overlaps fetches and stores with other operations.

2. Q: What is the timing for a Swap of n words between a and a + 1? In particular, when step two is reached, does Stretch have to re-fetch from a + 1 the word it just stored there? We would like this to be made reasonably fast, if possible.

A: Timing is 4.8 n microsecond. Yes, the word is refetched. Progress of Swap is as follows: $Z_L + 1 = Z_R$.

Address	a	a + 1	a + 2	a + 3	
Original	b	c	d	e	
1st	c	b	d	e	
2nd	c	d	b	e	etc.

3. Q: If an ordinary floating point number is added to a zero with large negative exponent, is the Preparatory Shift Greater than 48 indicator turned on?

A: Yes. The preparatory shift >48 indicator is based on a difference is exponents with no exceptions.

4. Q: IBM has still never replied to our request to give us an example justifying the existence of the Set and Wait instruction.

A: In practice Set and Wait is used with its own location in the address position in order to be sure the wrong interrupt doesn't send the computer off. Branch enable to itself gives the same result but the STOP lite is not turned on.

5. Q: What is the present indicator set? Is there a Noisy Mode indicator which can be masked on and off? We feel that there must be, for safety. We request very strongly that all unused indicators (in particular, the "Harvest" indicator) be wired up as programmers' indicators.

A: The noisy mode indicator (#63) is masked off. The Harvest indicators simply lack Harvest inputs. The new Chapter 3 gives the present indicator set.

6. Q: When a memory operand is at an address less than 16, is there a gain in speed? We would like things to be arranged so that one can use the miscellaneous registers for data and get fast access.

A: No. The concept that transistorized registers will yield fast access to data in a random fashion was a strong fallacy in the early planning of the STRETCH computer. More advanced planning has revealed that the orientation of the machine to receive data from core memory and fundamental logical contradictions make access from transistorized storage even slower than 2.0 usec. core storage in many cases.

7. Q: What happens if there are two op-code 8's in succession? If this gives Invalid Operation, then we are wasting full word op-codes.

A: Does give invalid operation. Necessary to avoid ambiguous cases.

8. Q: How much time is spent for the two boundary register tests? Is this slowing down the machine during otherwise fast loops? Where is this comparison made?

A: Boundary register tests do not cause a slow down due to some extra equipment in the look-ahead unit which stores the result of the test and takes appropriate action. The comparison is actually made in the memory bus unit. This time is overlapped with Memory box selection, etc.

9. Q: What is the present status of the floating point Add to Magnitude and Augment orders? Have you made the changes analagous to those made in the integer set? We feel that this should be done.

A: New Chapter ~~X~~ 5 describes present status.

10. Q: What are the reasons for not having fast one-word fixups? Can we have them if we prohibit the programmer from changing data that is used in the next few succeeding instructions?

A: Fast one-word fixups implies the existance of a special look-ahead register to hold them. This would make multiple interrupts logically impossible. Programming restrictions such as that suggested are contrary to fundamental STRETCH philosophy.

11. Q: What happens if a Rename instruction has an effective address identical to the address in the current X0 Refill field? It is essential that the store effectively precede the fetch, as it says in the manual.

A: The store does precede the fetch.

12. Q: What happens if a VFL Immediate instruction, such as Load, specifies a Field Length greater than 24 bits?

A: Fills in zeros in low order.

13. Q: What are the properties of Floating Point Zeros in connection with Exponent Underflow? If there is a mode of machine operation in which any result having negative exponent spill is replaced quickly and automatically by a number with exponent all ones and mantissa zero, then we have no further worries. If there is not such a mode, then the fixup must be via interrupt. In the days of the fast one-word fixup, it was checked that everything was all right. If fix-ups always run the look-ahead dry, then things would still be all right if, and only if, a negative exponent spill indication is not given by the product of an ordinary number with negative exponent and a zero with exponent all ones. Similarly, no indication should be given when dividing a zero with exponent all ones by an ordinary number with positive exponent.

A: A Floating Point Zero (zero fraction) is independent of exponent underflow in the circuit logic. There is no machine mode for generating the negative exponent and zero fraction. The fixup is assumed to be by interrupt. The underflow indication is set very late in the instruction sequence, too late to effect the starting of another sequence. Changing the logic would slow down all floating point instructions. Multiply and divide are not special cases as far as exponent spill is concerned. The same tests are made on all floating point instructions.

14. Q: Why are accumulator bits 60-63 cleared in some single precision floating point operations? This can be irritating in some cases, and we should like to know whether it is really necessary, please explain why it is desirable, or else change things so as to leave 60-127 of the accumulator alone.

A: These bits are used in "double" orders but not in single. The question here depends on possible complication of the checking circuitry. Any conditional saving of 60-63 is completely impossible.

15. Q: Is there any reason why bit 55 is not doing more work in the Transmit orders?

A: Has been changed to full word transmits and swaps. (See II 8 above).

16. Q: We are delighted to see a new full word internal register and two new indicators. The register could be a handy temporary storage spot, presumably fast, especially if it were equipped with its own Load from the Store to memory. It would also be tempting to make it an indicator register.

A: The N register is an address in 2 usec. core storage and load N is of the same order of speed and arrangement as load cumulative multiplicand. Due to the desire to eliminate certain minor inconsistanties both instructions are quite slow and awkward to perform. Since it is not a transistorized register, it could not serve as an indicator register very effectively. Adding new transistorized registers is impossible at this late date.

17. Q: What happens on interrupt if the Interruption Address is less than 32?

A: The Indicator A0 is turned on. If next address is also below 32, is stuck in loop.

18. Q: What happens if the interrupt table is in a protected area? Is the Instruction Fetch indicator set?

A: Gets table value, no indicators turned on.

19. Q: Is there any special protection for the boob who has an interrupt table of all binary zeros?

A: No. He is stuck in a loop.

20. Q: What happens when compound orders modify themselves? For example: at address a there is a VFL Store and Progressive Indexing; the value field of index I has a and the address field of the VFL Store has β ; is the final value field of index I 2a or a + β ?

A: a + β.

Q: Another example: in a there is a STICI to a + 1/2; in a + 1/2 there is a half-word Branch to β ; do we branch to β all right? If so, suppose β is not protected, but a + 1 is; do we get any false indicators?

A: If there is a STICI in a, then a + 1/2 contains the branch address of that instruction. If this address is β, then the machine will branch to β . If it is desired to store the instruction counter in a + 1/2 there will be no problem, since this is done by means of a store to a after the instruction counter has been placed in the contents of a.

21. Q: What happens if you put a pair of half-word orders into the interrupt table? Are they both performed, or only the first? If both, what happens if the first causes an interrupt?

A: Only the first one is executed under any circumstances.

22. Q: What happens if there is a data store to, or fetch from, a.63, say, of a field longer than one bit, and a is not protected but a + 1 is? Or a full word instruction fetched from a + 1/2?

A: Whole field is protected if any part of it falls outside the boundary. Instruction is "no-op'ed".

EIJ:HGK:jcj

E. I. Jordan H. G. Kolsky

May 4, 1959

APPENDIX III

Questions concerning STRETCH operation raised during the Joint LASL-IBM Mathematical Planning Group Meeting April 27-28, 1959

1. Q: Does Store Square Root to the accumulator run look-ahead dry? If so it should be changed to leave the root in the accumulator all the time.

   A: It does not run look-ahead dry.

2. Q: Why is the Noisy Mode used for floating point compare?

   A: Floating point compare is the same as a floating add instruction without changing the accumulator. For logical work one would probably use unnormalized arithmetic for which there is no noise inserted.

3. Q: Why is there a 1 millisecond time limit on Load Indirect?

   A: This interrupt is tied to the time clock advance. It is between 1 and 2 milliseconds. Any other timing would be difficult to do.

4. Q: If a long transmit is proceeding can the time clock interrupt properly and restart the transmitting?

   A: Yes.

5. Q: Does Branch on Bit always store the word back whether or not the bit is changed?

   A: Yes, it always stores back. Since stores are normally overlapped this should not cause a delay.

6. Q: How fast is a SWAP order in half microsecond memory?

   A: Same speed as in 2 microsecond memory after the initial start up.

7. Q: If a Transmit type instruction were to cross a protected area does it stop at the first boundary and does it resume after the second boundary?

A: It stops at the first boundary and does not resume automatically.

8. Q: If a Transmit is made from the indicator register are the indicators which are set by the TMT itself already set or are the old values still there?

A: The old values are there until the TMT is finished.

9. Q: How does a program entered by basic linkage determine the enable-disable status of the machine?

A: If the machine is disabled and one wishes to determine this without enabling, it can be done by: masking the DF indicator, attempting a data fetch from address 1, testing indicator. If the machine is enabled there is no fool-proof way of determining it without the possibility, however remote, of other interrupts coming in during the operation sequence and disabling. There is no simple hardware technique of providing such a test without logical inconsistancies.

10. Q: Are the switches on the Maintenance Console usable for momentary action by the programmer?

A: The Maintenance Console assumes that the machine is stopped when its switches are thrown. They can be thrown during operation but may give inconsistant results since they go straight into the machine with no buffering.

11. Q: What is the latest information on arithmetic speeds?

A: The current timing for arithmetic execution is:

Floating Add	0.95 usec
Floating Multiply	1.8 usec
Floating Divide	7.0 usec
Cummulative Multiply	2.75 usec
Square Root	$\sim$ 3 divide times
Floating Load	0.6 usec
Floating Store	0.5 usec

VFL Add and Connect $(2 + \frac{n}{8})$ 0.5 usec

Binary Multiply $0.5 (14 + \frac{n}{4}) + (\frac{n}{\sqrt{2}} + 2)$ 0.4 usec.

HGK:jcj H. G. Kolsky