Kolsky Memo 8

PROJECT 7000 November 7, 1958

FILE MEMORANDUM

SUBJECT: Some Studies Using an Improved Sigma Timing Simulation Program

I. Introduction

The SIGMA timing simulator has been used to study a number of machine design effects during the past year.* During the various studies the simulator code has been modified and added to several times. However, an attempt has never been made to duplicate the machine logic and timing exactly. There are two reasons why this was not done: (1) Including all the logical details would be a job equivalent to designing the computer control circuits themselves-- (A job made harder because the controls have never been completely specified at any one time.) (2) It has been highly desirable to have extra logical generality and flexibility in the Simulator to permit it to study cases not possible in the real machine.

The simulation philosophy has been to ignore the fact that certain parts of the machine are really much more complicated than their simulation as long as the average time for the functions are about right. The improvements in the Simulator discussed herein are of two types: one group is a general "tidying up" of details of a minor nature, and the other is an attempt to include some of the effects of the "stripped" Look-ahead and the "interlocked" Index Memory.

After examining the results of these runs, we realize that some of the approximations made earlier were not as negligible as had been hoped--particularly those which effect instruction fetches. We also realize that a more accurate version of the I-Box including more of the internal register interlocks must be coded in order to study these instruction fetch and preparation delays in detail. The present studies must then be considered as a progress report, not as the last word.

*Reference: Project 7000 File Memos: August 29, June 30, May 29, May 19, April 18, March 12 and February 6, 1958.

II. Test Problems Used

The test problems tried were the Mesh Problem and the Monte Carlo Branching Problem used in many of the earlier studies. The Mesh Problem is mostly arithmetic speed limited while the Monte Carlo is instruction-fetch limited. The emphasis has been upon comparing runs made with the Harvest-Sigma Simulator (June 1958) and the present Revised Simulator (October 1958).

Runs were made using the two programs as originally written using index memory for the storage of intermediate results. They were also made with intermediate results stored in regular main memory. A few runs were made varying index memory speed in the former case.

The arithmetic speeds used are those listed in the August 29th memo. The "average" times for arithmetic operations in each case are: Harvest = 2.40 usec, Sigma = 1.43 usec, Standard = 0.64 usec, Fast = 0.2 usec. All cases were run with the 0.8 usec maximum I-Box repetition rate and 0.2 usec bus slots.

III. Changes Incorporated in the Revised Simulator:

The following modifications were incorporated in the June 1958 Simulator Code to make the Revised Simulator:

(1) The separate return bus from fast memory was deleted, so that instructions and data both use the same return path.

(2) The individual Look-ahead levels were stripped of their address comparison registers and forwarding mechanisms. In place of this general system, there is a single address-compare register which contains the data address of either the last instruction loaded, or of the last unexecuted store-type instruction.

(3) The Look-ahead is interlocked so that it can contain only one store-type instruction at a time.

(4) The index memory logic is tied more closely to the indexing arithmetic unit by the inclusion of indexing "pseudo-stores". Under this system the new value of the index memory word is stored immediately and the old value is retained in Look-ahead as a "pseudo-store" so that it can be replaced in the event of an interrupt or a wrong-way branch.

(5) The logic which permitted the comparison of index addresses against look-ahead data addresses was removed. Instruction fetch addresses are compared against the single Look-ahead address register for stores only.

(6) Two minor changes concerning the operation of the arithmetic unit during stores, and other logical "tidying up" changes were made.

IV. Results of Study Using Main Memory for Intermediate Results

The over-all speeds for the various combinations studied are listed in table I for the Mesh Problem and table II for the Monte Carlo Branching Problem. The speed data are shown graphically in graphs I and II.

It is interesting to note that storing intermediate results of the computations in main memory still results in an appreciable loss of performance for all the combinations studied, although the Revised Simulator indicates a smaller loss than the old one.

The Monte Carlo, which is strongly instruction-fetch limited, shows a bigger percentage reduction in the Revised Simulator than does the arithmetic-limited Mesh Problem. This alerts us to look for changes which cause delays in instruction fetching.

V. Results of Study Intercomparing the Old and Revised Simulators

To examine the reasons for the Revised Simulator being so much slower than the old one, we made detailed timing charts of the Monte Carlo and Mesh Problems for both simulators and traced the causes of each difference which appeared.

Table III lists a summary of the main factors which caused the differences and the percentage increase in total running time each caused. Some of the factors represent real changes in machine organization others are changes in Simulator logic which were never possible in the machine design--at least they are no longer possible under the present rules of economy,speed, and checking. A brief description of each of the factors follows.

(1) The transmit or swap instruction delay is caused by the single address register in look-ahead. After the acceptance of a store instruction look-ahead is prevented from loading another instruction until the store has been completed. The transmit instruction contains four load-store combinations and the swap instruction contains two loads followed by two successive stores. Recent discussions indicate that even the Revised Simulator may be too optimistic compared to the actual machine.

(2) The index memory tie delay is caused by the inability of the machine to break into the sequential index memory references of the same instruction. An example of this is the count and branch instruction where the index value is brought out, counted down, and stored back.

In the old Simulator, index memory was treated as being logically identical to main or fast memory. That is I-box stores to index memory were made from Look-ahead just as arithmetic unit stores were. The important difference between the two systems turns out not to be the time for the storing (which is about the same) but the fact that the next instruction could slip in and get its index value before the store was started. This meant that incoming instructions could often "steal a march" of 0.8 usec. or more in preparation so that their data fetches could start that much sooner and consequently the arithmetic unit had less chance of standing idle later.

(3) The forwarding of index quantities from look-ahead to the I-Box was possible in the old Simulator but not in the new one. This is really part of the same general scheme mentioned in (2) above where the old Simulator treated index memory logically the same way as other memory i.e., the old memory value was unchanged until the new value was stored from Look-ahead. In the meantime any reference to that index address would receive its data forwarded from look-ahead.

This effect also ties in with the lack of address-compare registers on each look-ahead level. Even if the data paths from look-ahead to X-register existed, there is presently no way of testing for the address comparison.

Since it was able to compare addresses and forward data from the look-ahead to the I-Box, the old Simulator had fewer index memory conflicts and instruction-fetch delays.

(4) The store delay is caused by the reduction of the four address compare registers in look-ahead to one. This is the same delay as the transmit instruction delay but is listed separately to distinguish the distributed effect from the lumped effect of the single instruction.

(5) The bus conflicts result principally from the index memory tie, not from the elimination of dual return busses. The removal of the latter caused no delays.

(6) The "Make-Up" results from the asynchronous operation of the computer which allows, in some cases, the slower machine to make a more speedy recovery from a wrong-way branch than a faster machine which has already initiated anticipated memory requests which were not in fact necessary.

VI. Results of Study Varying Index Memory Speed

A few runs were made with both simulators using SIGMA arithmetic speeds while varying the Index Core Memory cycle time. Table IV shows the results of these runs which are also plotted on graph III.

The effect of the I-Box Index Memory tie-in is very clear. The Revised Simulator, which has the tie-in, is 3 times more sensitive to Index core cycle time than the old Simulator which does not have it. Again it is the instruction-fetch-limited Monte Carlo problem which suffers the most.

VII. Conclusions

(1) Using main memory for storing all intermediate calculational results causes about a 5% speed loss over using index core memory for this purpose.

(2) The Revised Simulator shows that performance is a considerably more sensitive function of index core memory cycle time than was previously thought. We should seriously reconsider the use of transistor registers.

(3) The Revised Simulator indicates that the level of internal performance of SIGMA is about 52 times 704 for the Mesh Problem rather than the 58 times quoted earlier.

(4) The largest factor in the speed reduction are delays in instruction fetching and processing from index memory conflicts caused by the direct tie in between the I-Box and index core memory.

Unfortunately, there seems to be no real way in which the machine can duplicate the old Simulator's logic of performing index stores from look-ahead and forwarding index values from look-ahead in the times assumed, even if the data paths were present. The best way to counteract these delays is to use a higher speed index memory while leaving the logic as it is.

(5) The loss of the individual address-compare registers for each Look-ahead level and the "single-store" restriction caused the second largest delay listed under (1), (3) and (4) in table III. Those of (3), the forwarding of index values, are perhaps ruled out as being impossible in the times assumed in the old Simulator. Those of (1) and (4) are certainly real delays which would be removed by having address compares on each level.

(6) It is interesting to note that the delays in instruction fetching and processing uncovered by the new Simulator do not change the maximum preparation rate of 0.8 usec per instruction. The delays are in the form of interlocking the starting of preparation or fetching of the instructions not the duration of their preparation.

Bill C. Madden

W. C. Madden
Junior Engineer
Project 7000

Harwood H. Kolsky

H. G. Kolsky
Product Planning Representative
Project 7000

WCM/HGK:jcv
Attachment
cc: 7000 Product Planning
Mr. D. W. Pendery
Mr. R. E. Merwin
Mr. J. E. Pomerene
Mr. E. D. Foss
Mr. W. Wolensky
Mr. F. E. Johnston

7000 Engineering Planning
Mr. S. W. Dunwell
Mr. H. K. Wild
Mr. J. J. Kenney, Jr.
Mr. E. Bloch
Mr. C. R. Holleran
Dr. J. Cocke

Bloch
Jordan

TABLE I

Computer performance for the Mesh Problem as a function of Arithmetic Unit Speeds and the use of Index Memory for storage of intermediate results. Evaluated by both the old (June 1958) Simulator and the Revised Simulator (October 1958).

Simulator Used and Arith. Unit Speed	COMPUTER SPEED (Times 704) Intermediate Results in Index Mem.	Intermediate Results in Main Mem.	% Change due to use of Main Mem. Instead of Index Mem.
A. Old Simulator			
(1) HARVEST (2.40 us)	39.3	37.4	-4.9%
(2) SIGMA (1.43 us)	58.1	54.0	-7.0%
(3) STANDARD (0.64 us)	79.0	71.9	-8.9%
(4) FAST (0.20 us)	79.6	74.0	-3.0%
			Average = -6.0%
B. Revised Simulator			
(1) HARVEST	36.6	35.4	- 3.2%
(2) SIGMA	52.4	50.0	- 4.6%
(3) STANDARD	71.8	64.3	-10.4%
(4) FAST	74.0	66.3	-10.4%
			Average = -5.9%
% Change due to use of Revised Simulator instead of old one.			
(1) HARVEST	-6.9%	- 5.3%	
(2) SIGMA	-9.8%	- 7.4%	
(3) STANDARD	-9.0%	-10.5%	
(4) FAST	-7.1%	-14.2%	
Average =	-8.2%	- 9.3%	

TABLE II

Computer Performance for the Monte Carlo Branching Problem as a function of Arithmetic Unit Speed and the use of Index Memory for storage of intermediate results. Evaluated by both the old (June 1958) Simulator and the Revised Simulator (October 1958).

Simulator Used and Arith. Unit Speed	COMPUTER SPEED (Times 704) Intermediate Results in Index Mem.	Intermediate Results in Main Mem.	% Change due to use of Main Mem. Instead of Index Mem.
A. Old Simulator			
(1) HARVEST (2.40 us)	36.3	34.6	-4.7%
(2) SIGMA (1.43 us)	39.3	37.3	-5.0%
(3) STANDARD (0.64 us)	43.0	39.7	-7.6%
(4) FAST (0.20 us)	43.5	40.7	-6.5%
		Average =	-6.0%
B. Revised Simulator			
(1) HARVEST	31.4	30.4	-3.1%
(2) SIGMA	33.6	32.5	-3.4%
(3) STANDARD	35.6	33.9	-4.6%
(4) FAST	36.2	34.6	-4.5%
		Average =	-3.9%
% Change due to use of Revised Simulator instead of old one.			
(1) HARVEST	-13.6%	-12.1%	
(2) SIGMA	-14.5%	-12.9%	
(3) STANDARD	-17.4%	-14.6%	
(4) FAST	-16.8%	-15.0%	
Average =	-15.6%	-13.7%	

TABLE III

Percentage Increase in total simulated Running Time of Problems for SIGMA because of changes incorporated in the October 1958 Revised Simulator Program.

Factor Causing Time Delay (See Section V for discussion.)	Percent of Total Delay caused by Factor Monte Carlo Problem	Mesh Problem
(1) Transmit or Swap Instructions	11.4%	17.1%
(2) Index Memory to I-Box Tie	47.6%	46.3%
(3) Forwarding of Index quantities from look-ahead	36.2%	23.2%
(4) Look-ahead Store Delay	6.7%	11.0%
(5) Bus Conflicts	2.9%	3.6%
(6) Time "make-up"	-4.8%	-1.2%

TABLE IV

Computer Performance as a function of Index Core Memory cycle time.

Sigma arithmetic speeds are assumed (average 1.43 usec per operation). Read-out time of the core memory is kept the same, only the total cycle is varied.

A. Monte Carlo Branching Problem

Index Memory Cycle Time	COMPUTER SPEED (times 704) June 1958 Simulator	October 1958 Revised Simulator
0.4 usec.	40.7	37.5
0.6 usec.	40.1	35.9
0.8 usec.	39.3	33.6

B. Mesh Problem

Index Memory Cycle Time	COMPUTER SPEED (times 704) June 1958 Simulator	October 1958 Revised Simulator
0.4 usec.	59.4	55.9
0.6 usec.	58.9	54.5
0.8 usec.	58.1	52.4

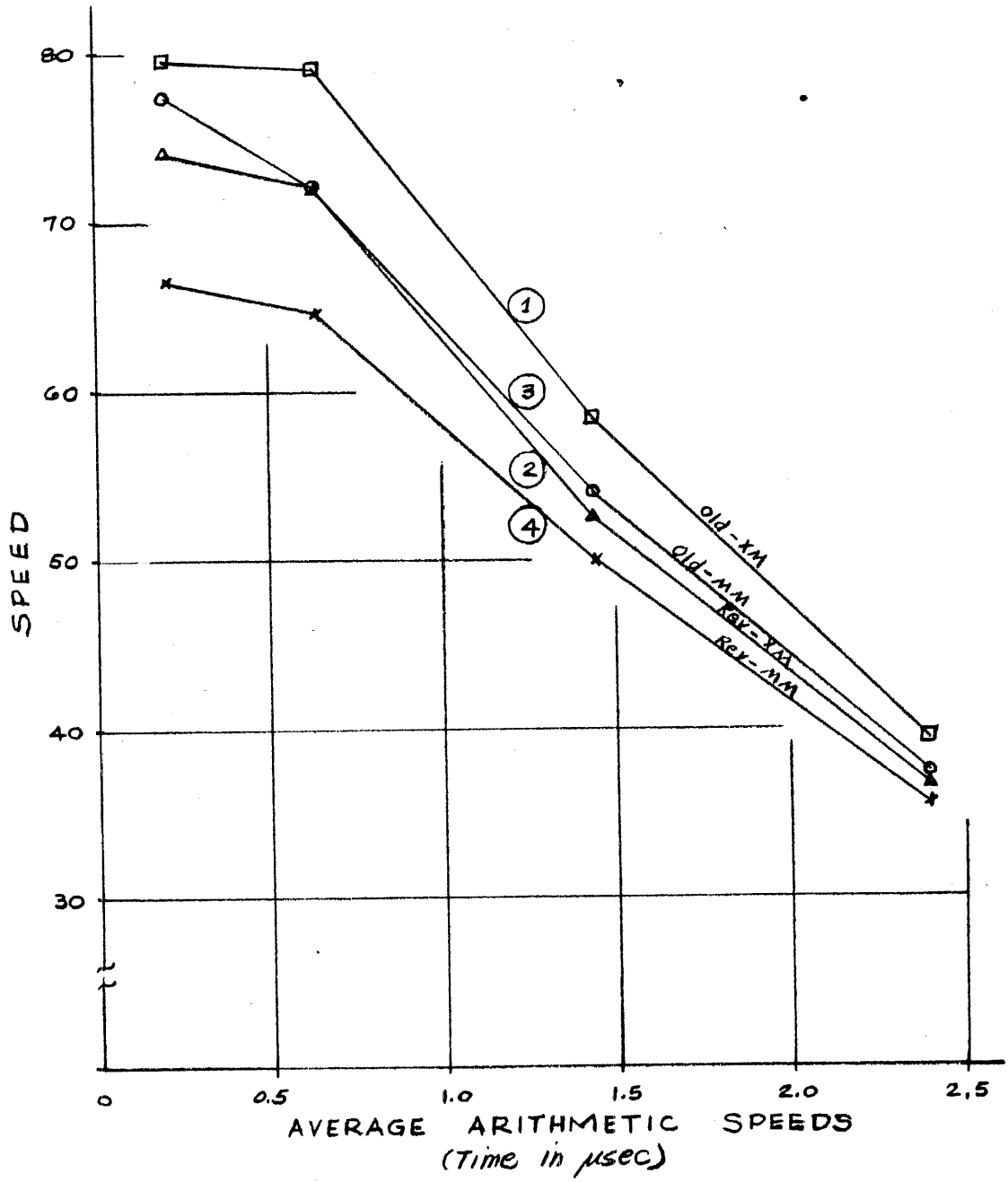
Computer Performance vs. Arithmetic Unit Speed for old and revised Simulators.

10-28-58

GRAPH I.

MESH CALCULATION

Intermediate Results in Index Memory	(1) Old	Simulator
	(2) Revised	"
Intermediate Results in Main Memory	(3) Old	"
	(4) Revised	"



BCM

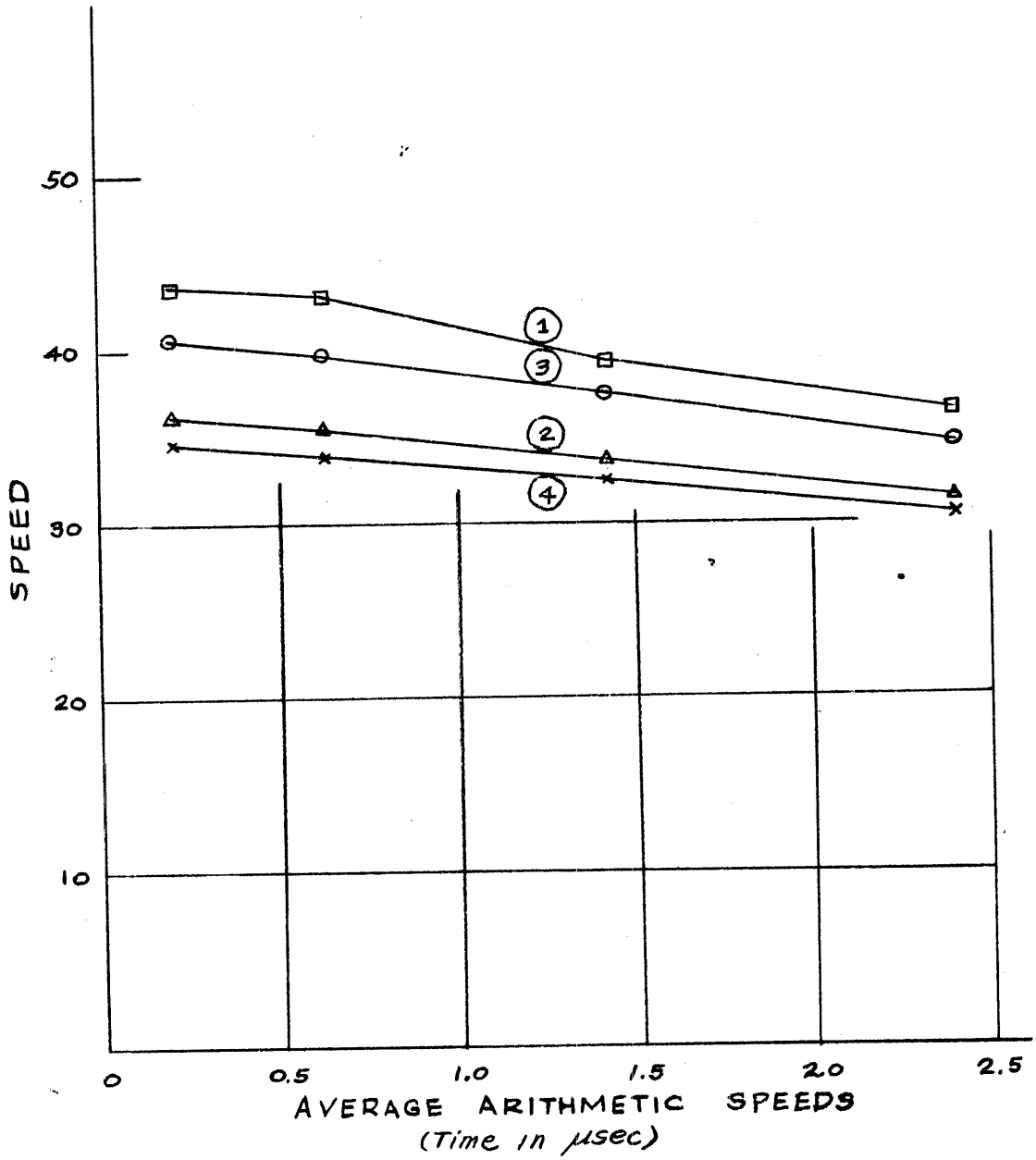
10-28-58

Computer Performance vs. Arithmetic Unit Speed for old and revised Simulators

GRAPH II.

MONTE CARLO

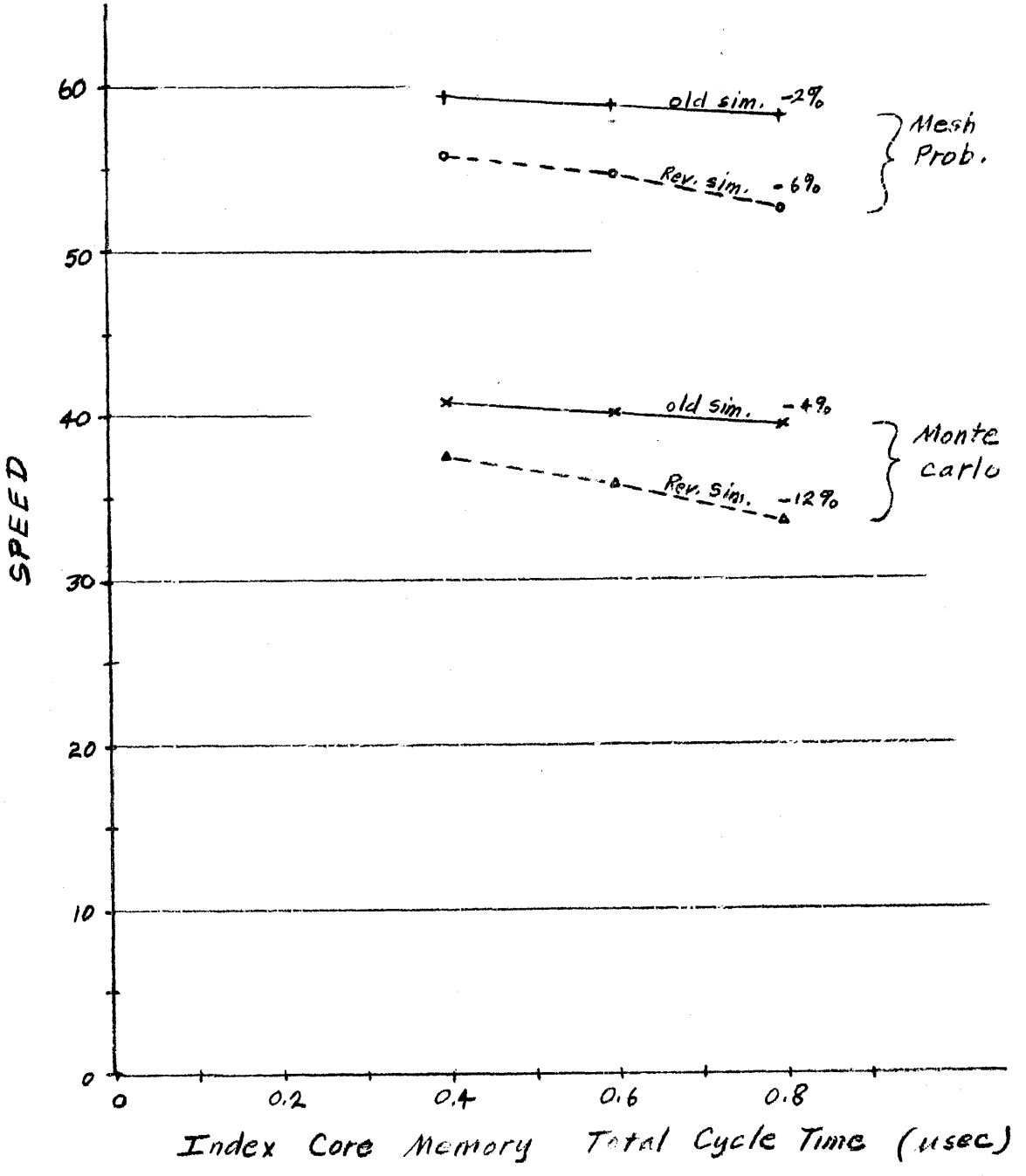
- Intermediate Results in Index Memory. ① Old simulator
- ② "
- Intermediate Results in Main Memory ③ "
- ④ "



BCM

GRAPH III

Computer Speed vs. Index Core Memory Speed



HGR