H. G. Kolsky
Memo 6

June 30, 1958

PROJECT 7000

FILE MEMORANDUM

SUBJECT: Results of the HARVEST Computer Timing Simulation Studies

CONTENTS:

I. Introduction

The engineering decision to make the HARVEST Basic Computer a direct derivative of the parallel SIGMA Computer resulted in a higher performance system as well as reducing the design problems. A minor side-effect of this decision was that the SIGMA Timing Simulator (Refs. Project 7000 File Memos dated February 6, March 12, April 18, May 19, and May 28) could be made to simulate the non-streaming operations of HARVEST with only minor modifications.

Interest in having more accurate figures to describe the overall performance of the HARVEST Computer arose during the discussions preceeding the drafting of the formal contract with BuShips for the purchase of the HARVEST System.

In a meeting held with BuShips representatives at Fort Meade, Maryland, May 21, 1958, the SIGMA timing simulator was described and a series of simple test problems and configurations were agreed upon. (Ref. Product Planning meeting report dated May 27).

A preliminary report concerning the first few runs was made in a letter by N. Blazensky dated June 4, 1958.

Most of the calculations described herein were presented during a second meeting at Fort Meade on June 16, 1958, a few more were mailed on June 19.

Because the contract negotiations were proceeding at the same time that these simulation runs were being made, the main emphasis has been on speed. Often we have had to omit cases which might have helped fill out the curves. Also interesting little side issues which always arise in such studies had to be passed over in the rush to complete the main program. Fortunately the results seem to be unambiguous enough so that it is not likely that we have overlooked any large effects.

II. Brief Description of the Simulator Code

The SIGMA Simulator as modified for HARVEST consists of approximately 2200 words of 704 code and requires an 8192 word memory and 2 tape units. The Simulator's function is to draw "timing charts" which show the activity of the various components of the computer system vs time during the running of sample programs. The actual arithmetic operations of the code being simulated are not performed; their effect is obtained by using standard times from a table for the individual operations. Details of the times assumed are given in Section IV A.

The Timing considerations and interlocks existing in the actual circuits are simulated logically in the code. Samples of programs which would run about 1000 usec on the actual computer can be simulated in about five minutes on the 704 if only Summary information is printed.

The Simulator prints two types of listings, a detailed Timing Chart for the individual units (6 usec worth per page) and a summary listing which presents information accumulated during the run. The latter contains the following items.

1. The values of the input constants (No. of Main Memories, Arithmetic Times, etc.)

2. The Total Time for the run.

3. The percentage of time each component of the system has been running.

4. The percentage of time the arithmetic unit has been waiting and whether it has been waiting on data or instructions.

5. The percentages of time there were bus conflicts or memory conflicts.

Changes in the SIGMA Simulator which were necessary before doing the HARVEST runs were: (1) Bringing the bus system interlocks up to date. (2) Revising the I/O sections to include both a high speed and a medium speed exchange each capable of both reading and writing. (3) Preparing the problem codes for all the cases run.

III Summary of Results

The main reason for doing the simple programs such as the series of 32 adds was to get a measure of the extra time or "overhead" which must be added to the arithmetic unit times because of the many possible conflicts and delays in other parts of the system. Actually the results as presented in the asynchronous design of the computer has really paid off. There are only a few tenths of a microsecond added to the effective time for the separate word cases and less than an extra memory cycle added for the cross-word boundary cases.

Similarly the results for the effect of Input-Output running during a calculation were quite gratifying. The computer was still able to sneak in its memory references thru a veritable barrage of input-output references, well above any rates presently invisioned.

The memory combination results show that the fast memory pays off only for short field lengths. The read-out time seems to be more important than the total cycle except for cross-word boundary cases. Extra memory boxes help only for short field cases.

The sample programs indicate that the HARVEST Computer is about 15 times faster than the 704 provided one is using 704 full word data (or shorter fields stored one per word). However, if one compares HARVEST vs 704 with packed short field length data, a considerably larger speed advantage is obtained. Unfortunately this factor depends so heavily on the nature of the 704 program that it is hard to state it accurately---one can easily get ratios from 20 to 200 in overall speed. To avoid argument we have stuck to HARVEST times in this report. Other braver souls can discuss the relative speeds in detail.

Compared to SIGMA, the HARVEST computer is simpler to analyze because it is simpler (fewer memories, fewer look-aheads), and also because it is more consistently arithmetic unit limited. It is only at short fields (6 or 8 bits) that the non-linear effects so common in SIGMA come into play in the HARVEST System.

IV. Detailed Description of the HARVEST Simulator Studies

A. Fixed Constants used in the Simulator

1. Machine Components

a.	Levels of look-ahead	1
b.	Number of Fast Memories	2
c.	Number of Main Memories	2
d.	High Speed Exchange	1
e.	Medium Speed Exchange	1

2. Computer Speeds

a. Indexing Time, Includes instruction decoding, index addition, and storing the modified address.

b. VFL Arithmetic Unit Time, The VFL execution times were derived from the Performance formulas given in a File Memo dated May 22, 1958 by E. Bloch

Binary Add

$(2 + \frac{n}{8}) \times .45$ usecs

where n is the number of bits in the augend or addend, whichever is larger.

n	OP Time
6	1.4 usec
8	1.4
16	1.8
32	2.7
48	3.6
64	4.5

Binary Multiply

$(15 + .65\ n) \times .45$ usecs

where n is the number of bits in the multiplier.

n	OP Time
6	8.6 usecs
8	9.5
16	11.7
32	16.2
48	20.7

Binary Divide

$(20 + \frac{n}{8} + \frac{q}{2.5}) \times .45$ usecs

where n is the number of bits in the dividend and q the number of bits in the quotient.

n and q	OP Time
6	10.8
8	11.3
16	13.1
32	16.7
48	20.7

In all cases where n or q is divided by a number, the results should be rounded to the next higher integer.

3. Memory Speeds
   a. Fast Memory

Read Out Time*	0.4 usecs
End Signal Time	0.4 usecs
Memory cycle time**	0.6 usecs

* (Read out time including bus time and checking $1.15 \pm .15$ usecs depending on when the request is started.)

** (Effective cycle time is 0.9 usec since the bus clocking permits successive references to the same memory box only in multiples of 0.3 usec and the memory box must be free at the time of the reference, not just finishing.)

b. Main Memory
Read Out Time* 0.8 usec
End Signal Time 1.7 usec
Memory Cycle Time** 2.0 usec

*(2.15 ± .15 usec for same reason as above)
**(2.1 usec for same reason as above)

c. Index Core Memory
Read Out Time 0.4 usecs
Memory Cycle Time 0.8 usecs

The index cores are assumed tied directly to the IAU, so these figures include bus times.

d. BusSpeeds
Buses to and from the memories have 0.2 usec slots available every 0.3 usec. Decode and switching time in central control unit is 0.2 usec to 0.4 usec depending on bus slots available. There are logically separate buses for reading and writing.

e. Additional Delays
There is usually a 0.1 usec delay between the completion of any function and the beginning of the next one by the unit, or in the transfer from one register to another.

B. Variables Studied in the Present Runs
1. Combinations of Memories
   a. Data in Main Memory; Instructions in Fast Memory
   b. Data in Fast Memory; Instruction in Main Memory
   c. Data and Instructions in Main Memory
   d. Data and Instructions in Fast Memory

2. Number of Memories
   a. Two Fast Memories
   b. Two Main Memories
   c. Four Main Memories

3. Exchange Rates
   a. High Speed Exchange read and write every 7.1 usecs
   b. Medium Speed Exchange read and write every 32, 16, 8, 4, 2, 1.5 usecs.

4. Initial Delay
   a. With Instruction in Fast Memory, 2.3 usecs
   b. With Instruction in Main Memory, 2.7 usecs.

5. Word Boundary Conditions
   a. All fields in separate words within boundaries
   b. All fields crossing word boundaries
   c. 6 Bit Consecutive fields

C. The HARVEST Computer programs studied under the above conditions.
1. Adds
2. Multiplies
3. Divides
4. Load, Store (consisting of Load A, Load B, Store B)
5. Load, Add, Stores (consisting of Load A, Add B, Load C, Store C)
6. Load, Add to Memory (consisting of Load, Load Add, Store)
7. Sample Program (see Run No. 23 data sheet)
8. Variable Field Length Matrix Multiply (see Run No. 25 data sheet)

D. Result of Runs
1. Overhead Time
The difference between the average time and the arithmetic unit time for the Add, Multiply and Divide operations is defined as the "overhead" time. This is a combination of all delays outside the arithmetic unit. Overhead time ranges from a low of .14 usecs per Add, Multiply and Divide to a high of 2.7 usecs per Add, 1.90 usecs per Multiply and 1.80 usecs per Divide. (See graphs 16 and 17). The Lowest overhead times appear when Data is in Fast Memory and Instruction is in Main Memory or when both are in Fast Memory and when all fields lengths are in separate words. This is also true for the case of Multiply and Divide where Data and Instruction are in Main Memory or Data in Main Memory and Instruction in Fast Memory and fields are in separate words, but the Add overhead ranges up to about 1.0 usecs.

All the crossing word boundary cases for all memory combinations fall in the range of from 1.2 to 1.95 usec except the shorter Adds, when Data and Instruction are in Main Memory, which reaches the upper bound of 2.7 usecs.

2. Memory Combinations
For the Add operations the memory combinations in order of faster operation time are:
   a. Data and Instruction in Main Memory
   b. Data in Main Memory, Instruction in Fast Memory
   c. Data and Instruction in Fast Memory
   d. Instruction in Main Memory, Data in Fast Memory
   In the case of separate words, c and d give the same curve. (See graph 1)

   For the Multiply operations all memory combinations produce the same curve in the case of separate words. (See graph 7). In the crossing word boundary case the memory combinations in order of faster operation time are:

   1. Data and Instruction in Main Memory
      Data in Main Memory, Instructions in Fast Memory
   2. Data and Instruction in Fast Memory
      Data in Fast Memory, Instructions in Main Memory

   The same results apply to the Divide operations. (See graph 8)

Howard G. Kolsky

H. G. Kolsky
Product Planning Representative
Project 7000

N. J. Blazensky, Jr.

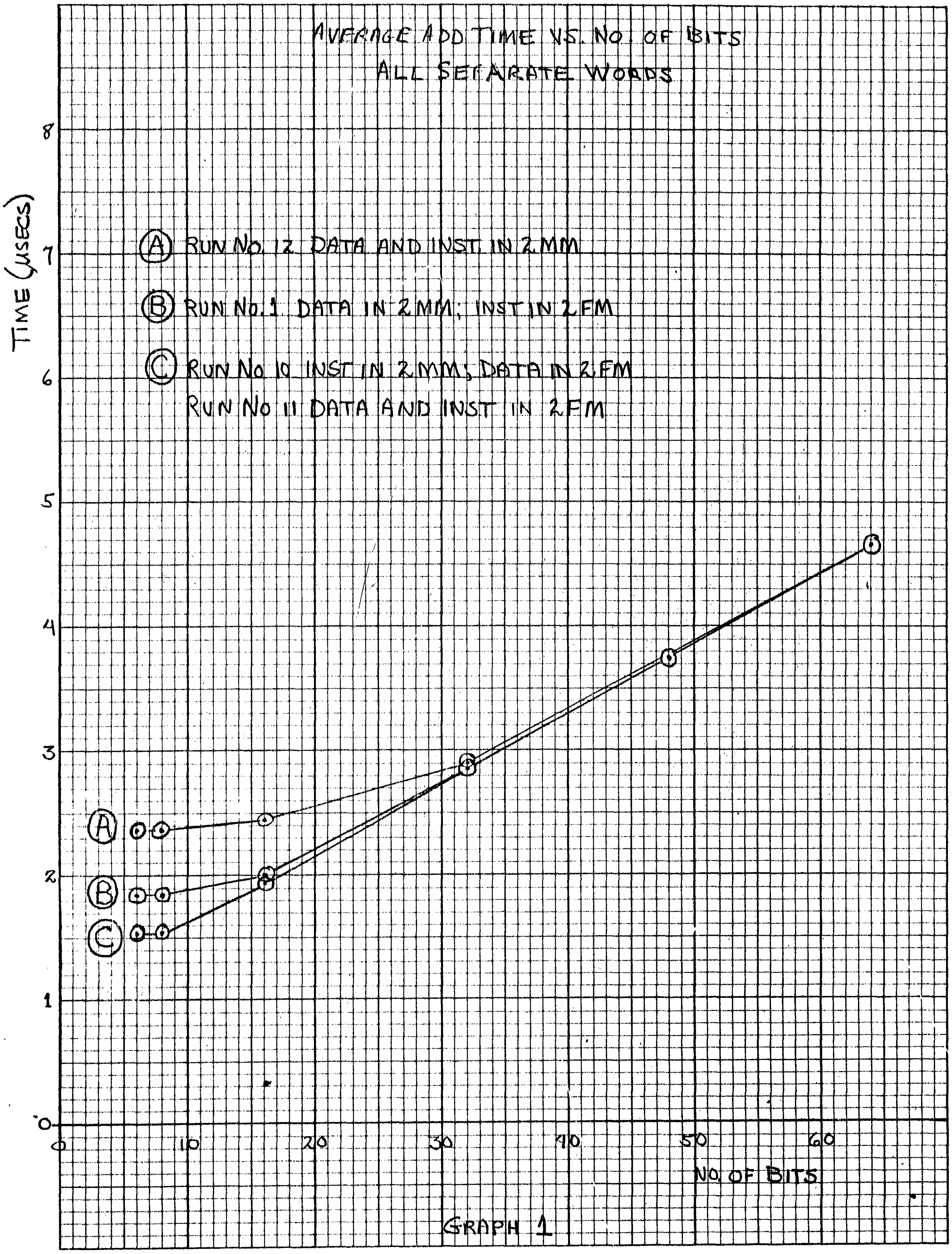
N. J. Blazensky, Jr.
Associate Engineer
Project RANCHO

HGK:NJB/jcv

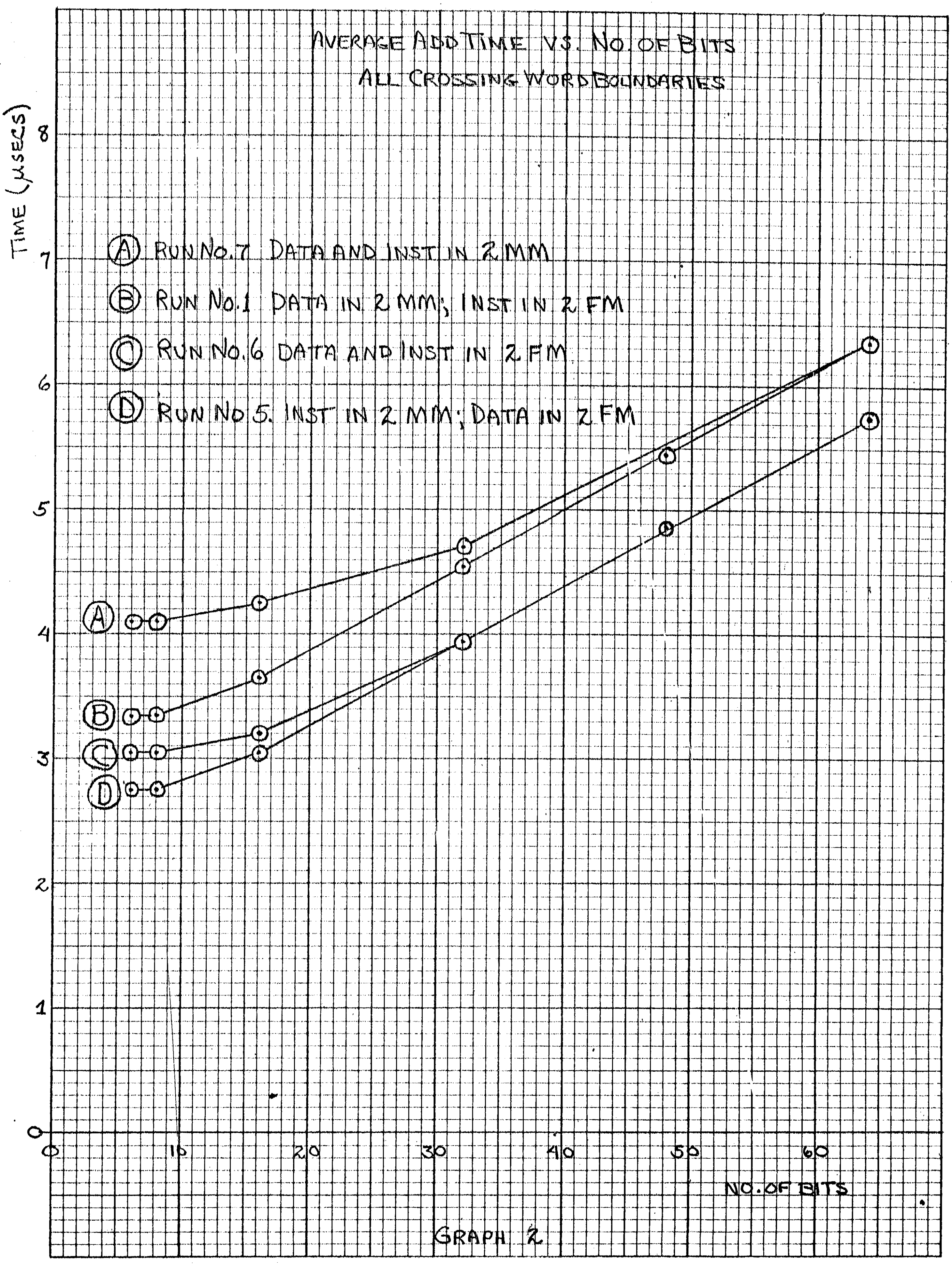
cc:	7000 Product Planners	7000 Engineering Planners
	D. W. Pendery	S. W. Dunwell
	R. E. Merwin	J. H. Pomerene
	H. A. Mussell	E. D. Foss
	H. K. Wild	E. Bloch
	L. E. Kanter	P. S. Herwitz
	J. J. Kenney, Jr.	J. J. Maher
	J. Cocke	W. J. Lawless, Jr.

V. Graphs of Results

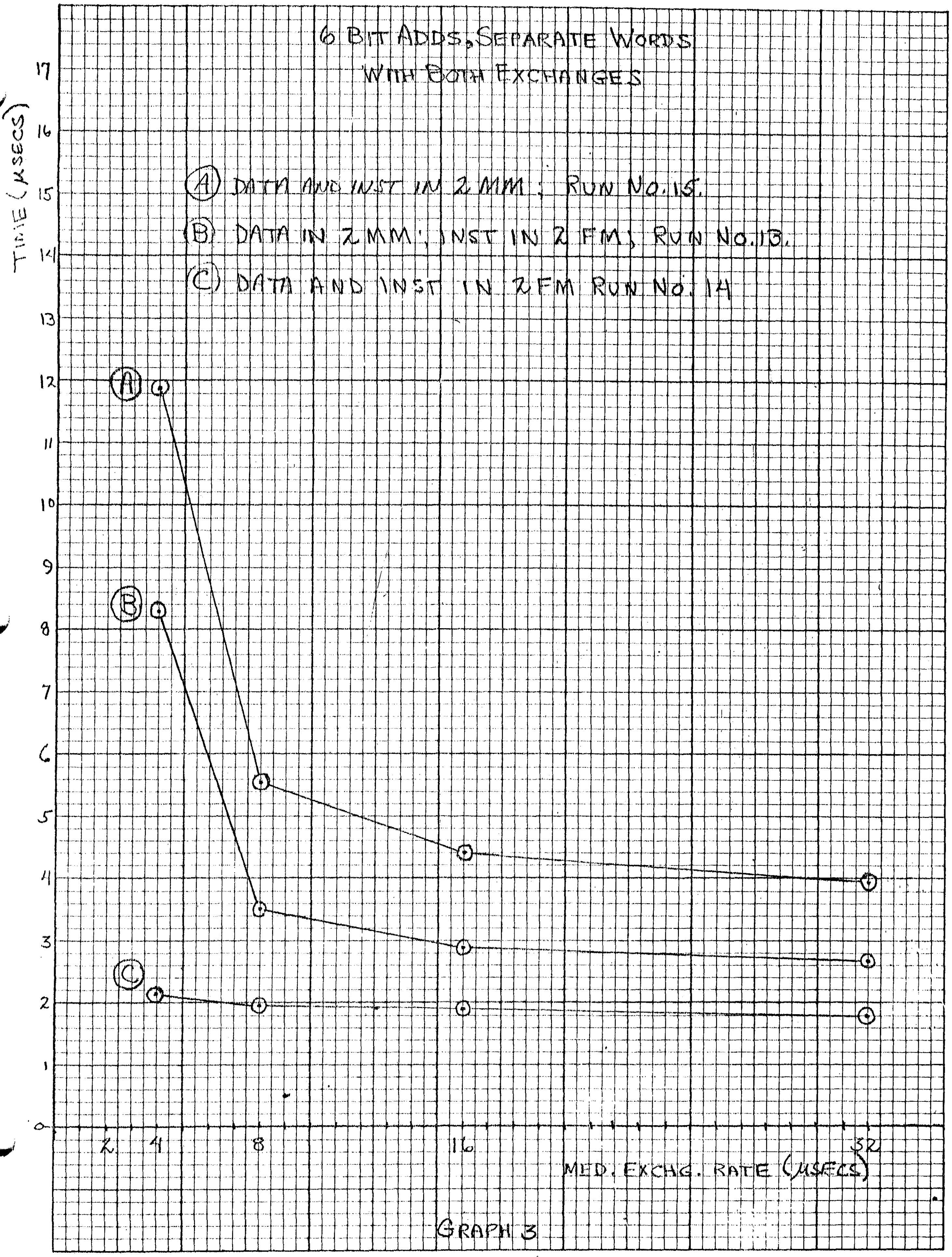
Index to Graphs


AVERAGE ADD TIME VS. NO. OF BITS
ALL SEPARATE WORDS
TIME (μSECS)
Ⓐ RUN NO. 12 DATA AND INST. IN 2 MM
Ⓑ RUN NO. 1 DATA IN 2 MM; INST IN 2 FM
Ⓒ RUN NO 10 INST IN 2 MM; DATA IN 2 FM
RUN NO 11 DATA AND INST IN 2 FM
0
1
2
3
4
5
6
7
8
0
10
20
30
40
50
60
Ⓐ
Ⓑ
Ⓒ
NO. OF BITS

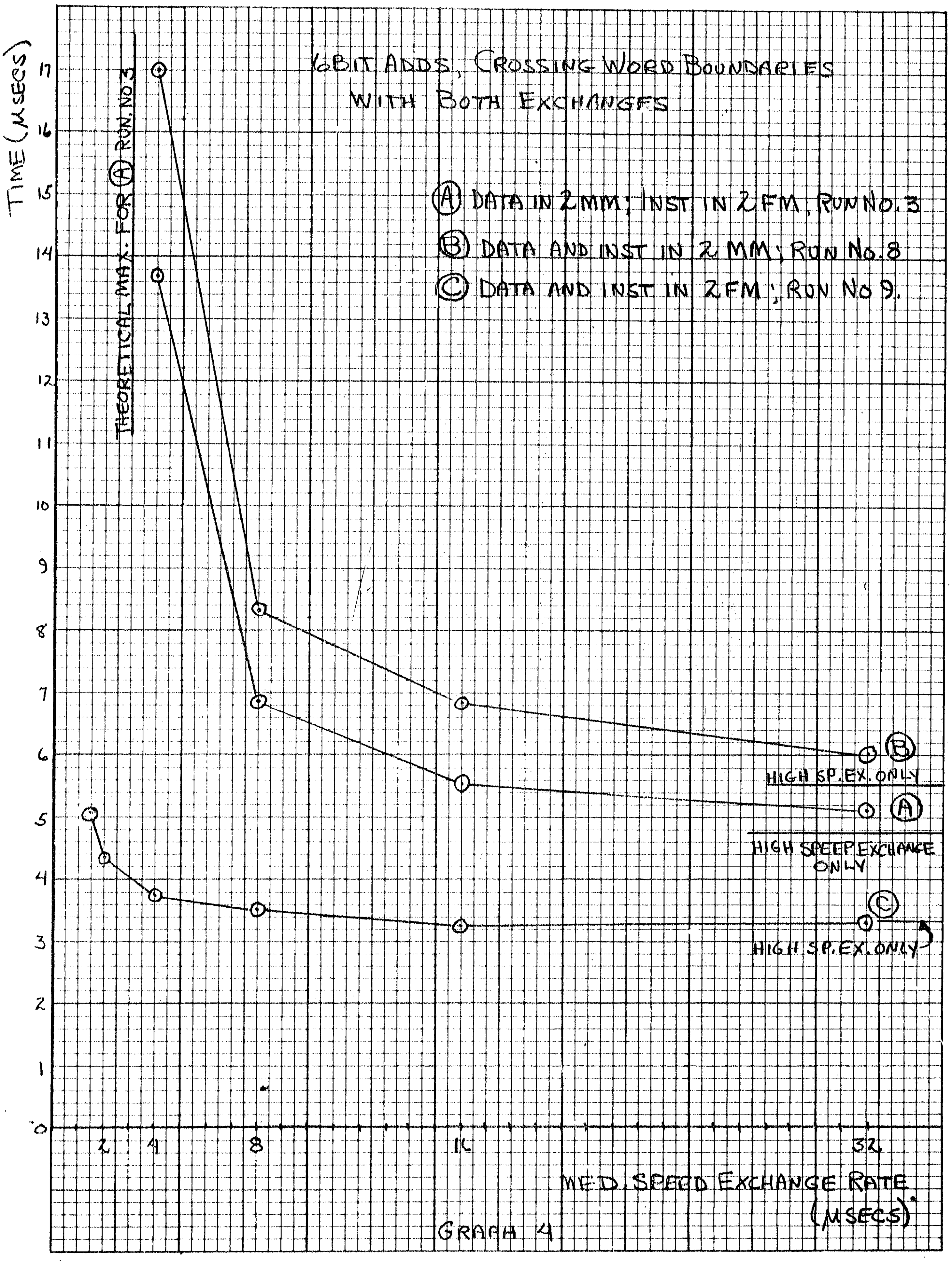
GRAPH 1


AVERAGE ADD TIME VS. NO. OF BITS
ALL CROSSING WORD BOUNDARIES
TIME (μSECS)
8
7
6
5
4
3
2
1
0
0
10
20
30
40
50
60
NO. OF BITS
Ⓐ RUN NO. 7 DATA AND INST IN 2 MM
Ⓑ RUN NO. 1 DATA IN 2 MM; INST IN 2 FM
Ⓒ RUN NO. 6 DATA AND INST IN 2 FM
Ⓓ RUN NO 5. INST IN 2 MM; DATA IN 2 FM
A
B
C
D

GRAPH 2

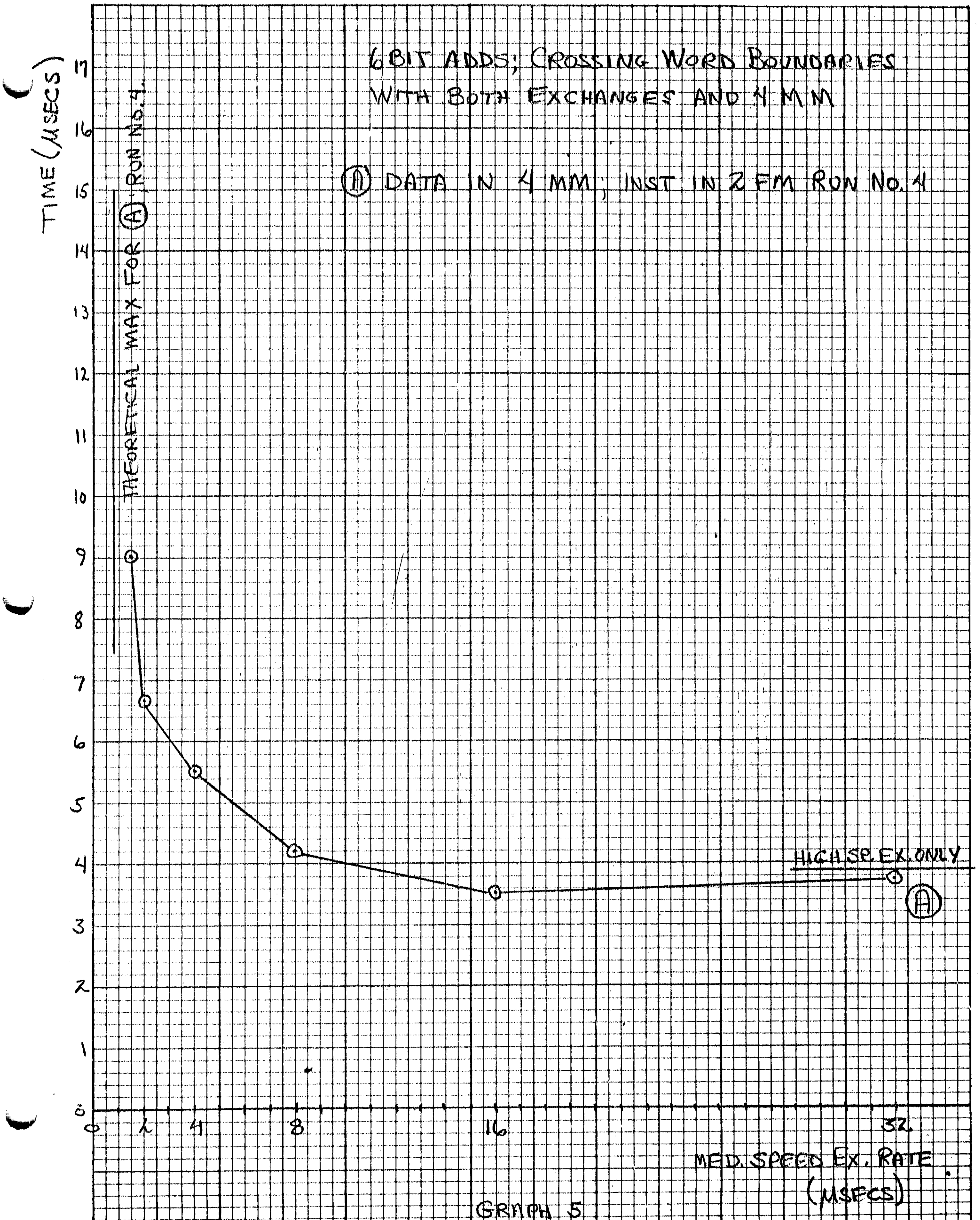

6 BIT ADDS, SEPARATE WORDS
WITH BOTH EXCHANGES
(A) DATA AND INST IN 2 MM: RUN NO. 15.
(B) DATA IN 2 MM; INST IN 2 FM; RUN NO. 13.
(C) DATA AND INST IN 2 FM RUN NO. 14
TIME (μSECS)
MED. EXCHG. RATE (μSECS)
A
B
C
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
2
4
8
16
32

GRAPH 3



6 BIT ADDS, CROSSING WORD BOUNDARIES WITH BOTH EXCHANGES
Ⓐ DATA IN 2MM; INST IN 2FM, RUN NO. 3
Ⓑ DATA AND INST IN 2 MM; RUN NO. 8
Ⓒ DATA AND INST IN 2FM; RUN NO 9.
THEORETICAL MAX. FOR Ⓐ RUN. NO. 3
TIME (MSECS)
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
2
4
8
16
32
Ⓑ
HIGH SP. EX. ONLY
Ⓐ
HIGH SPEED EXCHANGE ONLY
Ⓒ
HIGH SP. EX. ONLY
MED. SPEED EXCHANGE RATE (MSECS)

GRAPH 4


6 BIT ADDS; CROSSING WORD BOUNDARIES
WITH BOTH EXCHANGES AND 4 MM
Ⓐ DATA IN 4 MM; INST IN 2 FM RUN No. 4
THEORETICAL MAX FOR Ⓐ, RUN No. 4
HIGH SP. EX. ONLY
Ⓐ
TIME (μSECS)
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
0
2
4
8
16
32
MED. SPEED EX. RATE (μSECS)

GRAPH 5

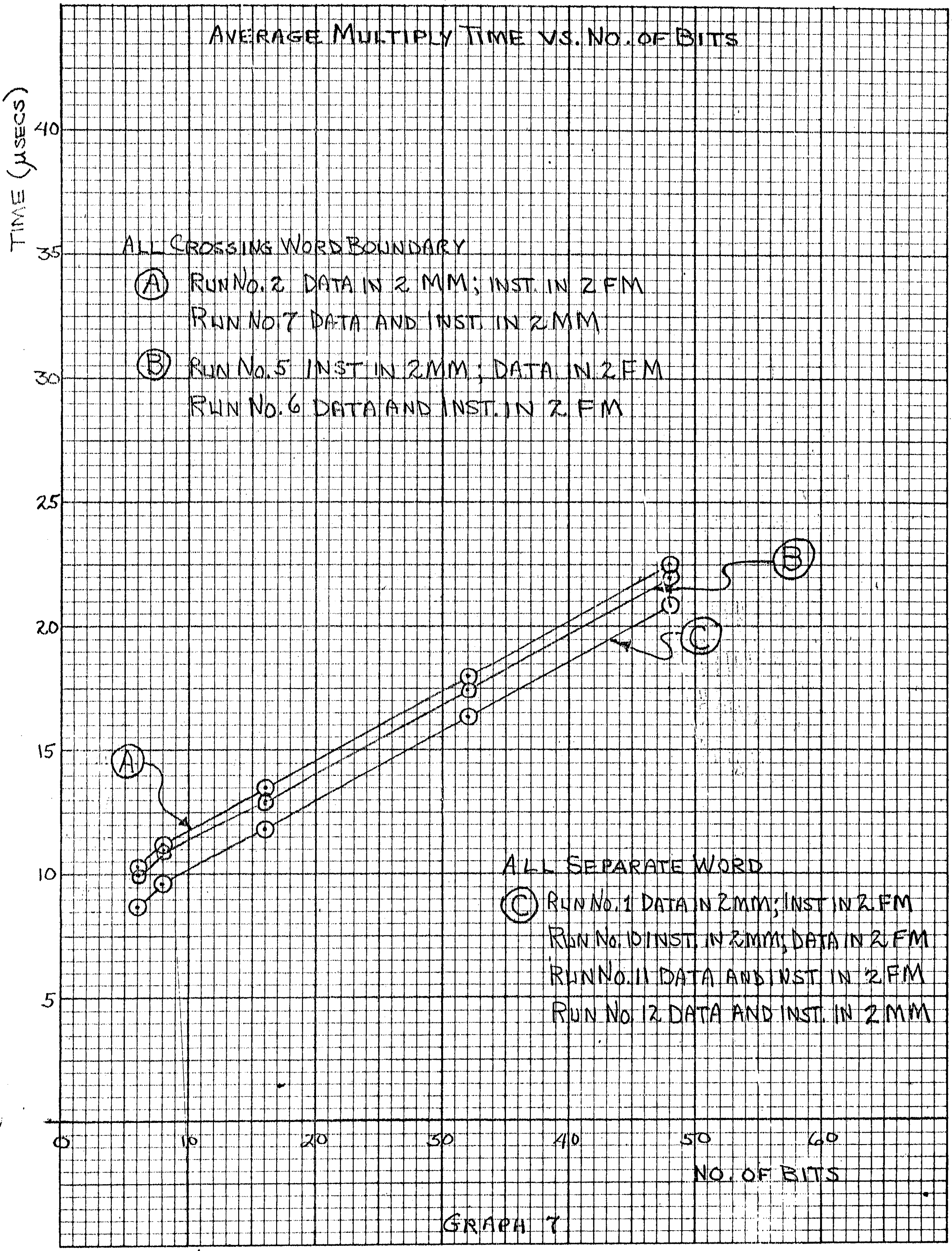
AVERAGE TIME FOR 6 BIT CONSECUTIVE ADDS
WITH EVERY 11TH BYTE CROSSING WORD BOUNDARY
RUN NO. 16

(A) DATA IN 2MM; INST IN 2 FM.
AVE. TIME = 1.655 μSECS

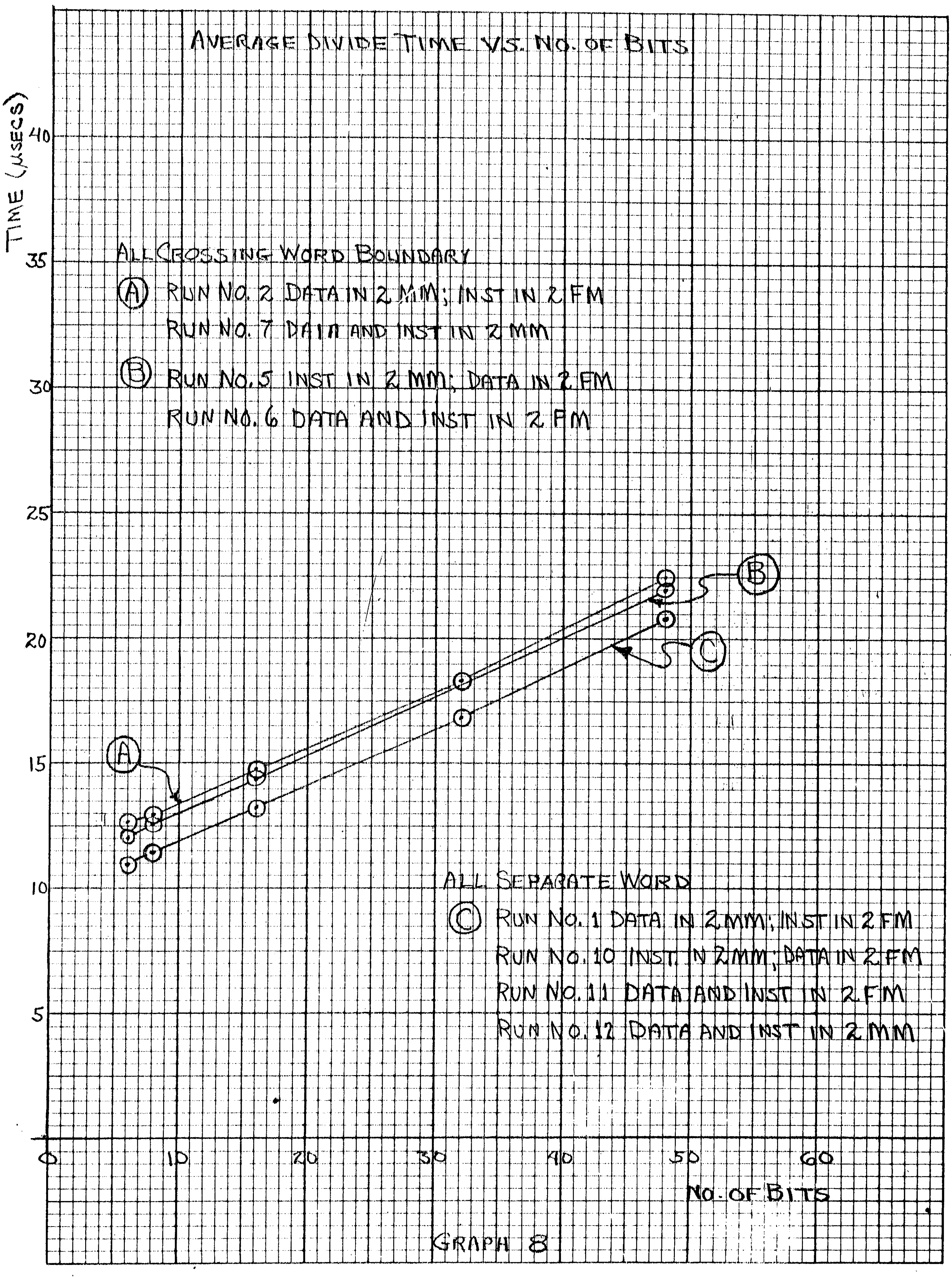
(B) DATA IN 2 FM; INST IN 2 MM
AVE. TIME = 1.650 μSECS

(C) DATA AND INST IN 2 MM
AVE. TIME = 1.652 μSECS

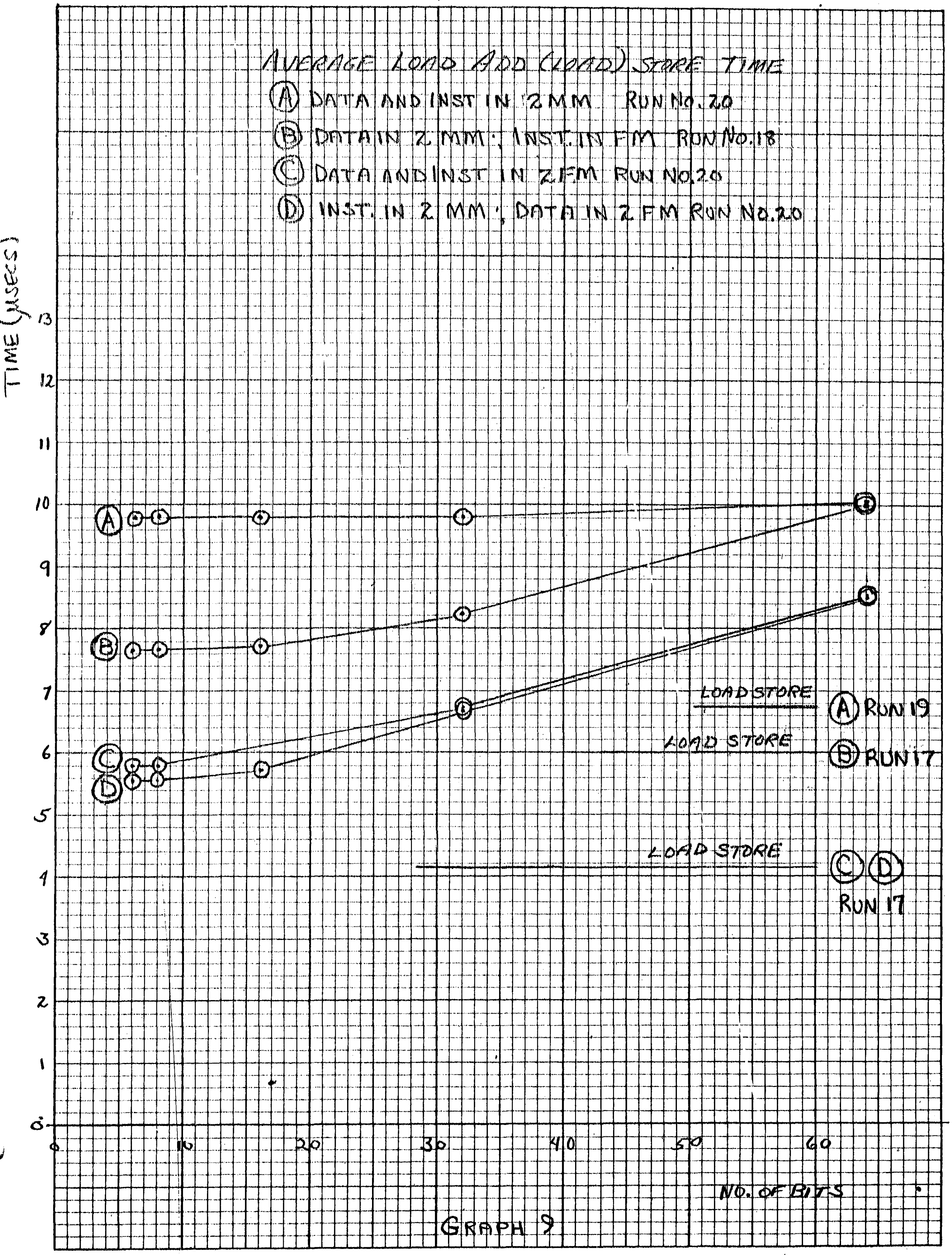
GRAPH 6


AVERAGE MULTIPLY TIME VS. NO. OF BITS
TIME (μSECS)
40
35
30
25
20
15
10
5
0
10
20
30
40
50
60
NO. OF BITS
ALL CROSSING WORD BOUNDARY
(A) RUN NO. 2 DATA IN 2 MM; INST. IN 2 FM
RUN NO 7 DATA AND INST. IN 2 MM
(B) RUN NO. 5 INST IN 2MM; DATA IN 2 FM
RUN NO. 6 DATA AND INST. IN 2 FM
A
B
C
ALL SEPARATE WORD
(C) RUN NO. 1 DATA IN 2MM; INST IN 2 FM
RUN NO. 10 INST. IN 2MM; DATA IN 2 FM
RUN NO. 11 DATA AND INST. IN 2 FM
RUN NO. 12 DATA AND INST. IN 2 MM

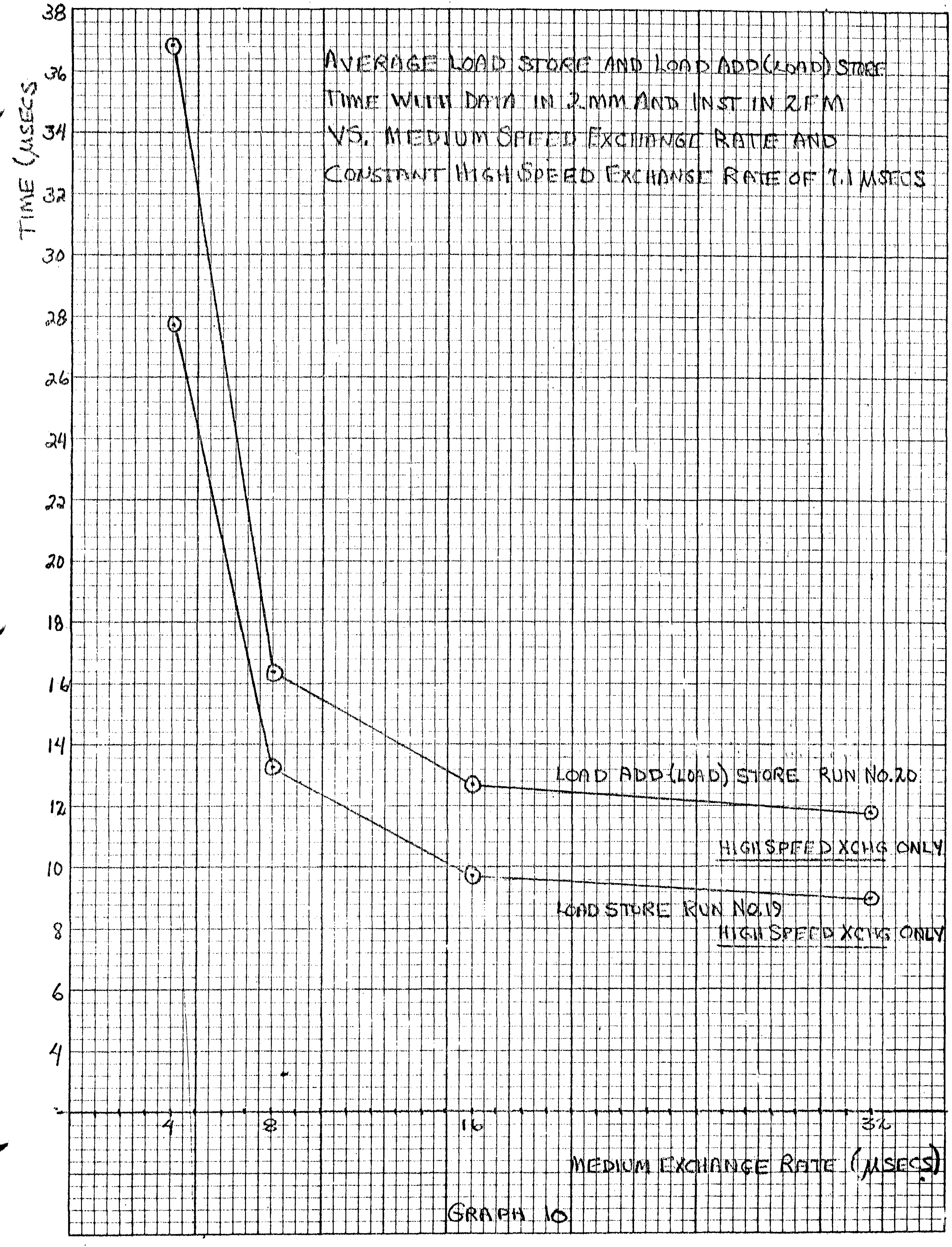
GRAPH 7


AVERAGE DIVIDE TIME VS. NO. OF BITS
TIME (μSECS)
ALL CROSSING WORD BOUNDARY
(A) RUN NO. 2 DATA IN 2 MM; INST IN 2 FM
RUN NO. 7 DATA AND INST IN 2 MM
(B) RUN NO. 5 INST IN 2 MM; DATA IN 2 FM
RUN NO. 6 DATA AND INST IN 2 FM
ALL SEPARATE WORD
(C) RUN NO. 1 DATA IN 2 MM; INST IN 2 FM
RUN NO. 10 INST. IN 2 MM; DATA IN 2 FM
RUN NO. 11 DATA AND INST IN 2 FM
RUN NO. 12 DATA AND INST IN 2 MM
A
B
C
40
35
30
25
20
15
10
5
0
10
20
30
40
50
60
NO. OF BITS

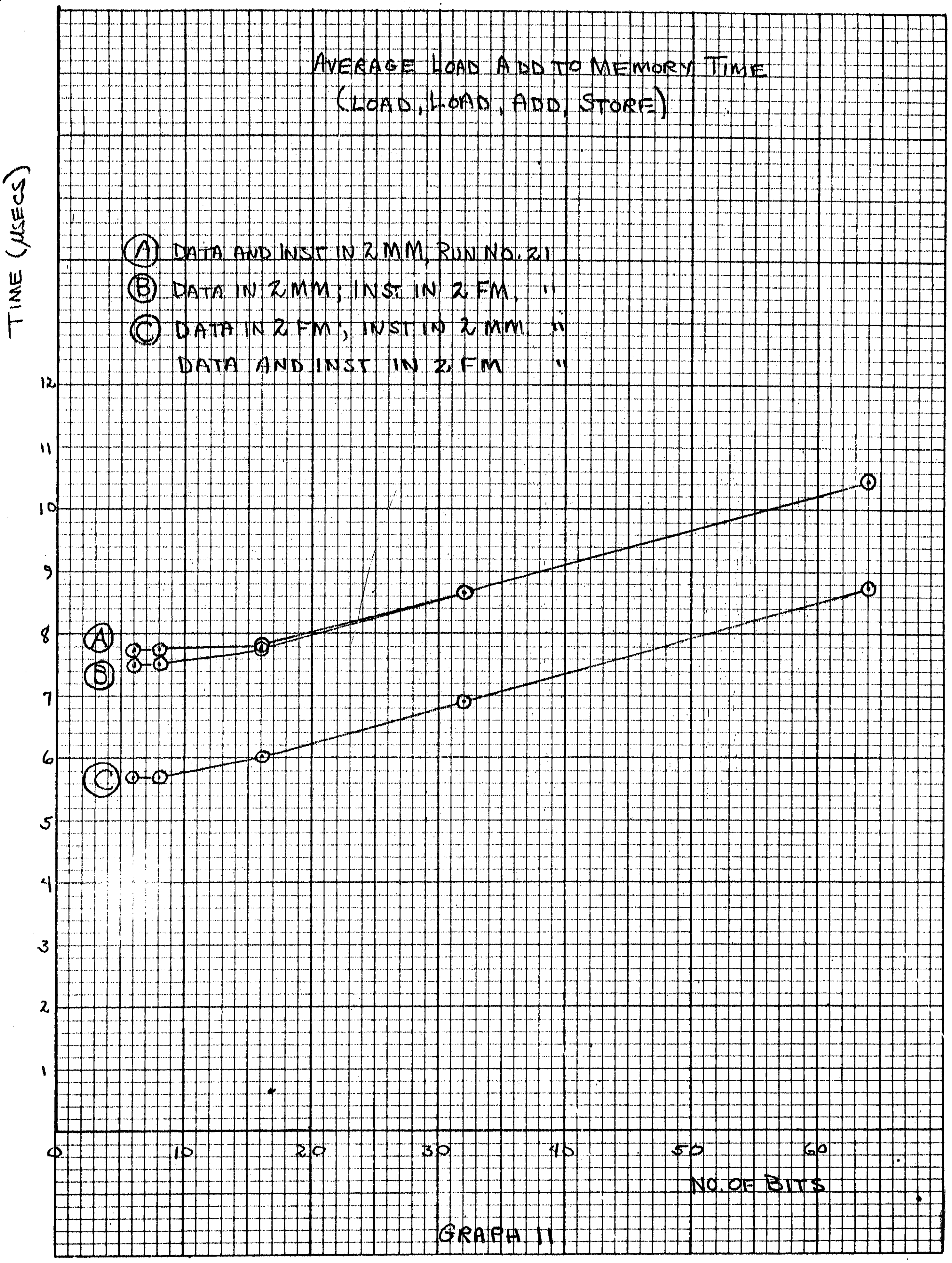
GRAPH 8


AVERAGE LOAD ADD (LOAD) STORE TIME
A DATA AND INST IN 2 MM RUN No. 20
B DATA IN 2 MM; INST. IN FM RUN No. 18
C DATA AND INST IN 2 FM RUN No. 20
D INST. IN 2 MM; DATA IN 2 FM RUN No. 20
TIME (μSECS)
13
12
11
10
9
8
7
6
5
4
3
2
1
0
A
B
C
D
LOAD STORE
A RUN 19
LOAD STORE
B RUN 17
LOAD STORE
C D
RUN 17
0
10
20
30
40
50
60
NO. OF BITS

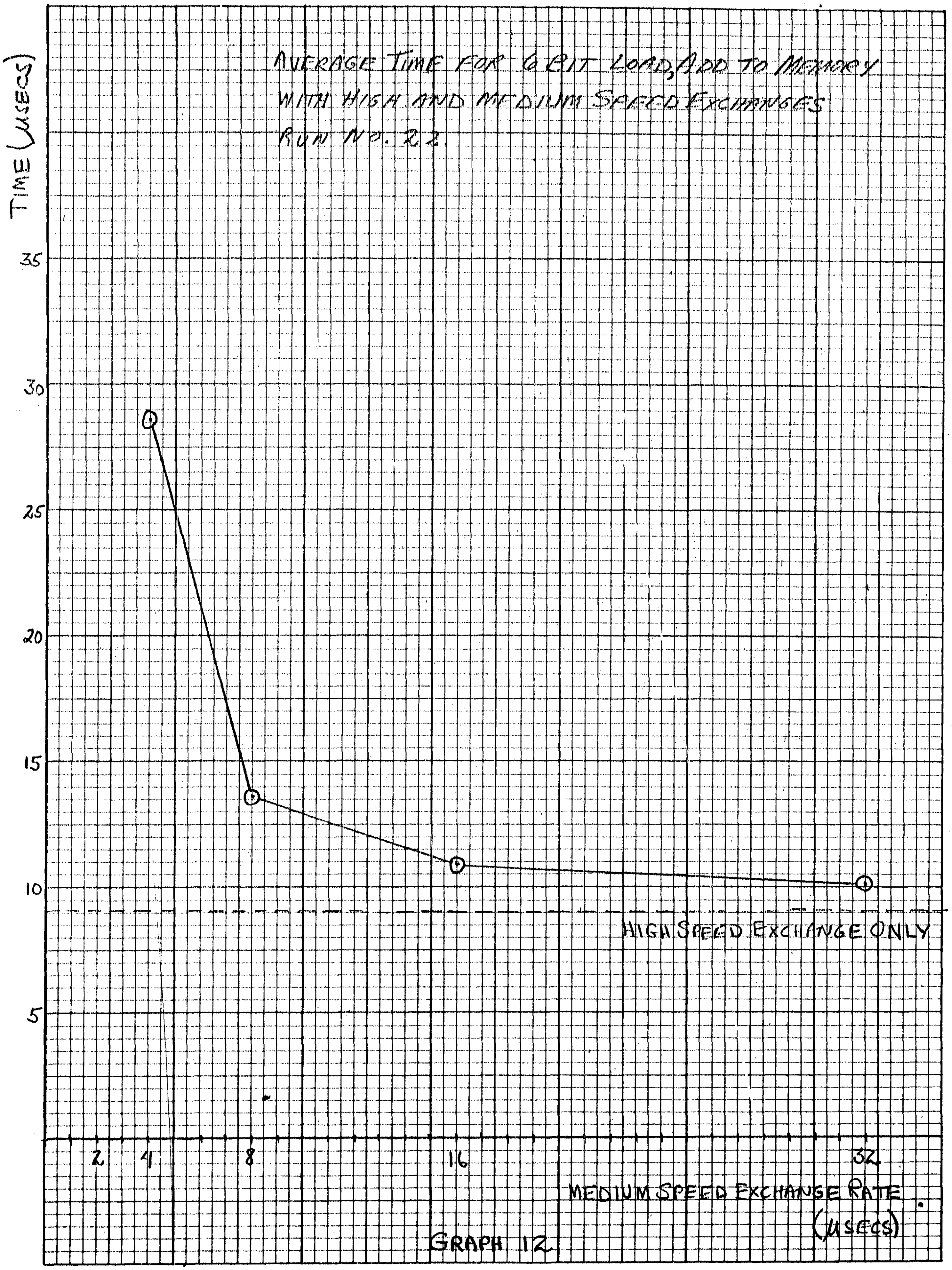
GRAPH 9


AVERAGE LOAD STORE AND LOAD ADD (LOAD) STORE
TIME WITH DATA IN 2MM AND INST IN ZFM
VS. MEDIUM SPEED EXCHANGE RATE AND
CONSTANT HIGH SPEED EXCHANGE RATE OF 7.1 MSECS
TIME (μSECS)
38
36
34
32
30
28
26
24
22
20
18
16
14
12
10
8
6
4
4
8
16
32
LOAD ADD (LOAD) STORE RUN NO.20
HIGH SPEED XCHG ONLY
LOAD STORE RUN NO.19
HIGH SPEED XCHG ONLY
MEDIUM EXCHANGE RATE (μSECS)

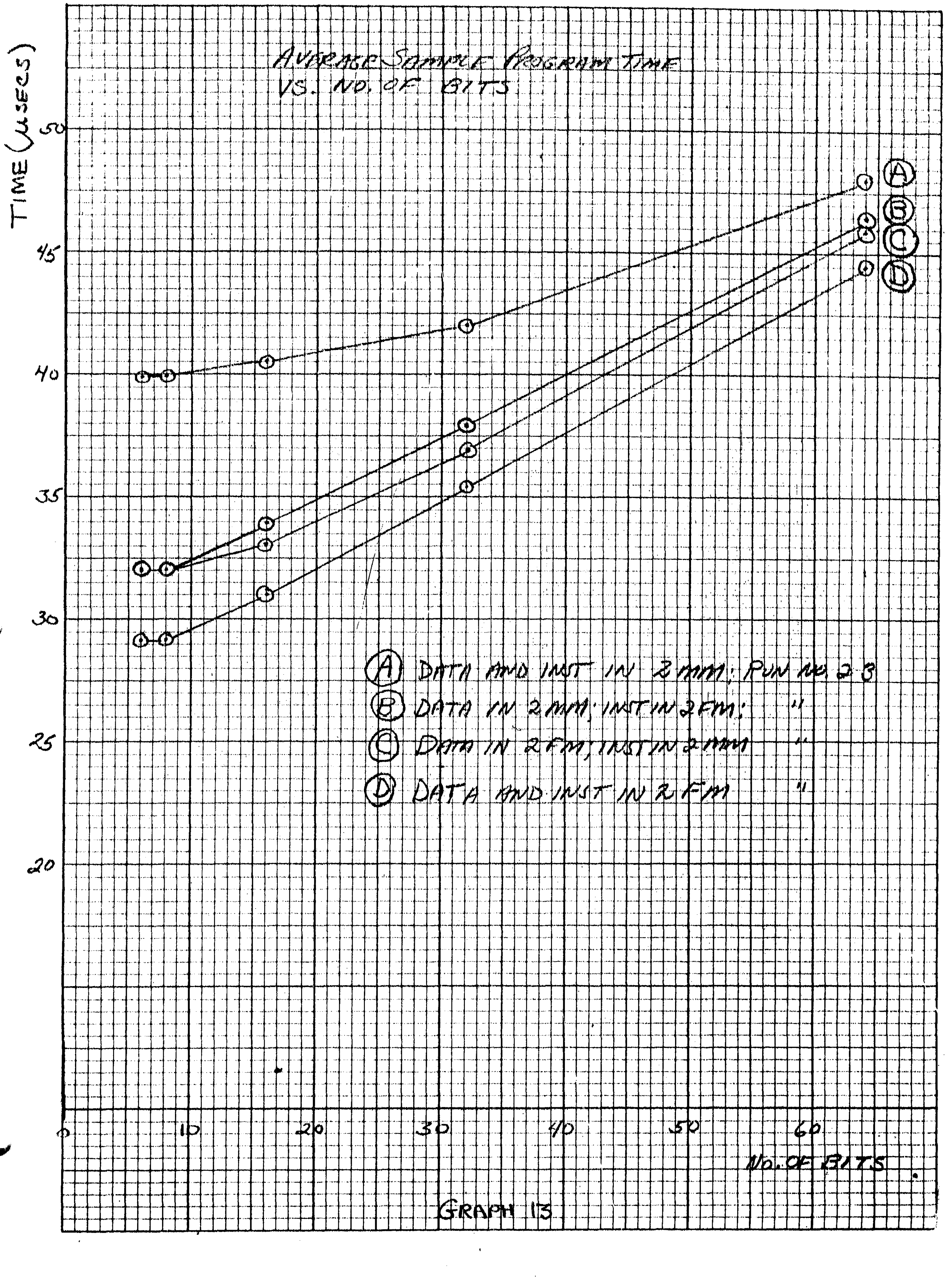
GRAPH 10


AVERAGE LOAD ADD TO MEMORY TIME
(LOAD, LOAD, ADD, STORE)
TIME (µSECS)
Ⓐ DATA AND INST IN 2MM, RUN NO. 21
Ⓑ DATA IN 2MM; INST. IN 2FM. "
Ⓒ DATA IN 2FM; INST IN 2MM "
DATA AND INST IN 2FM "
12
11
10
9
8
7
6
5
4
3
2
1
Ⓐ
Ⓑ
Ⓒ
0
10
20
30
40
50
60
NO. OF BITS

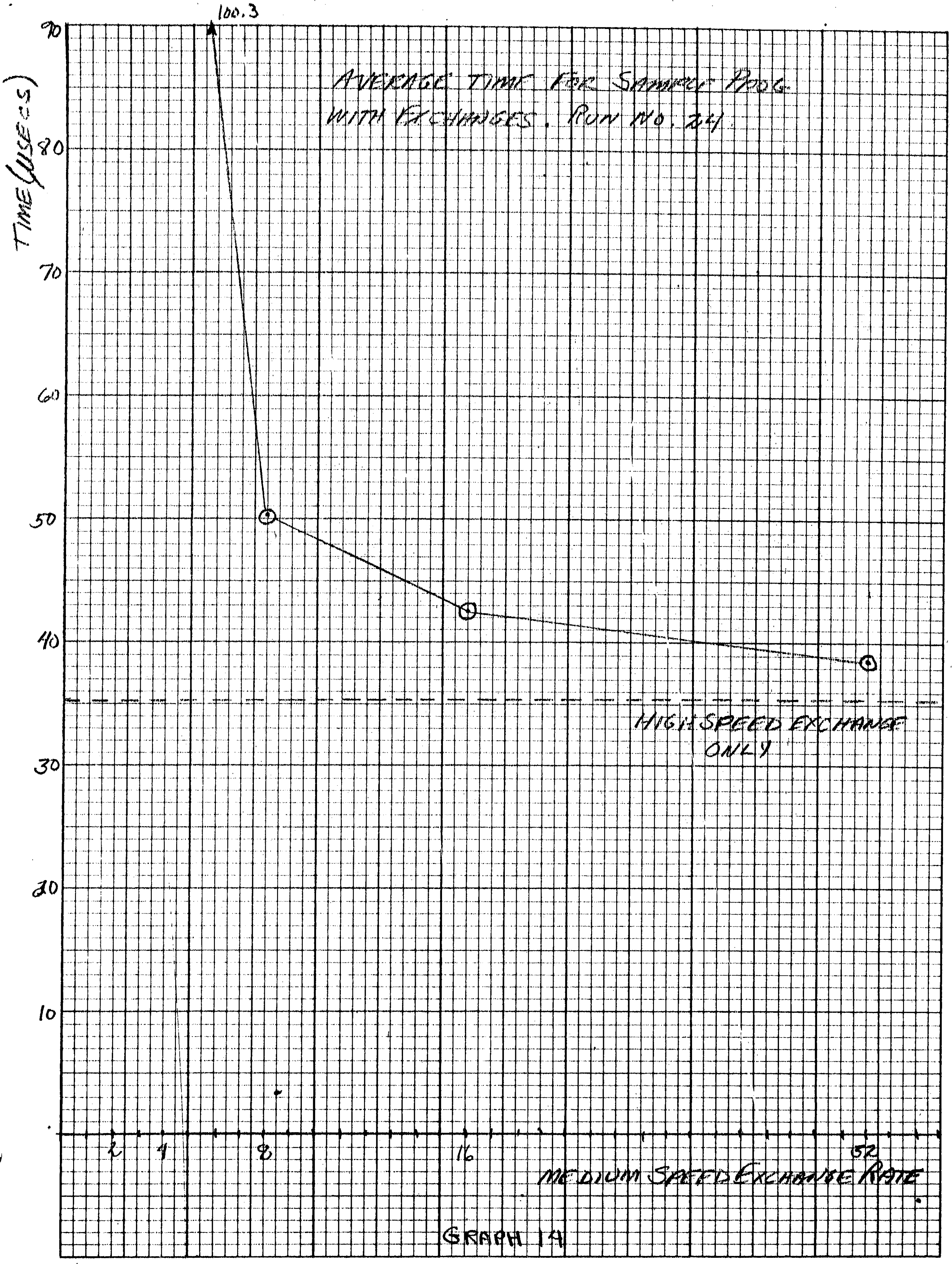
GRAPH 11


AVERAGE TIME FOR 6 BIT LOAD, ADD TO MEMORY
WITH HIGH AND MEDIUM SPEED EXCHANGES
RUN NO. 22.
TIME (μSECS)
35
30
25
20
15
10
5
HIGH SPEED EXCHANGE ONLY
2
4
8
16
32
MEDIUM SPEED EXCHANGE RATE
(μSECS)

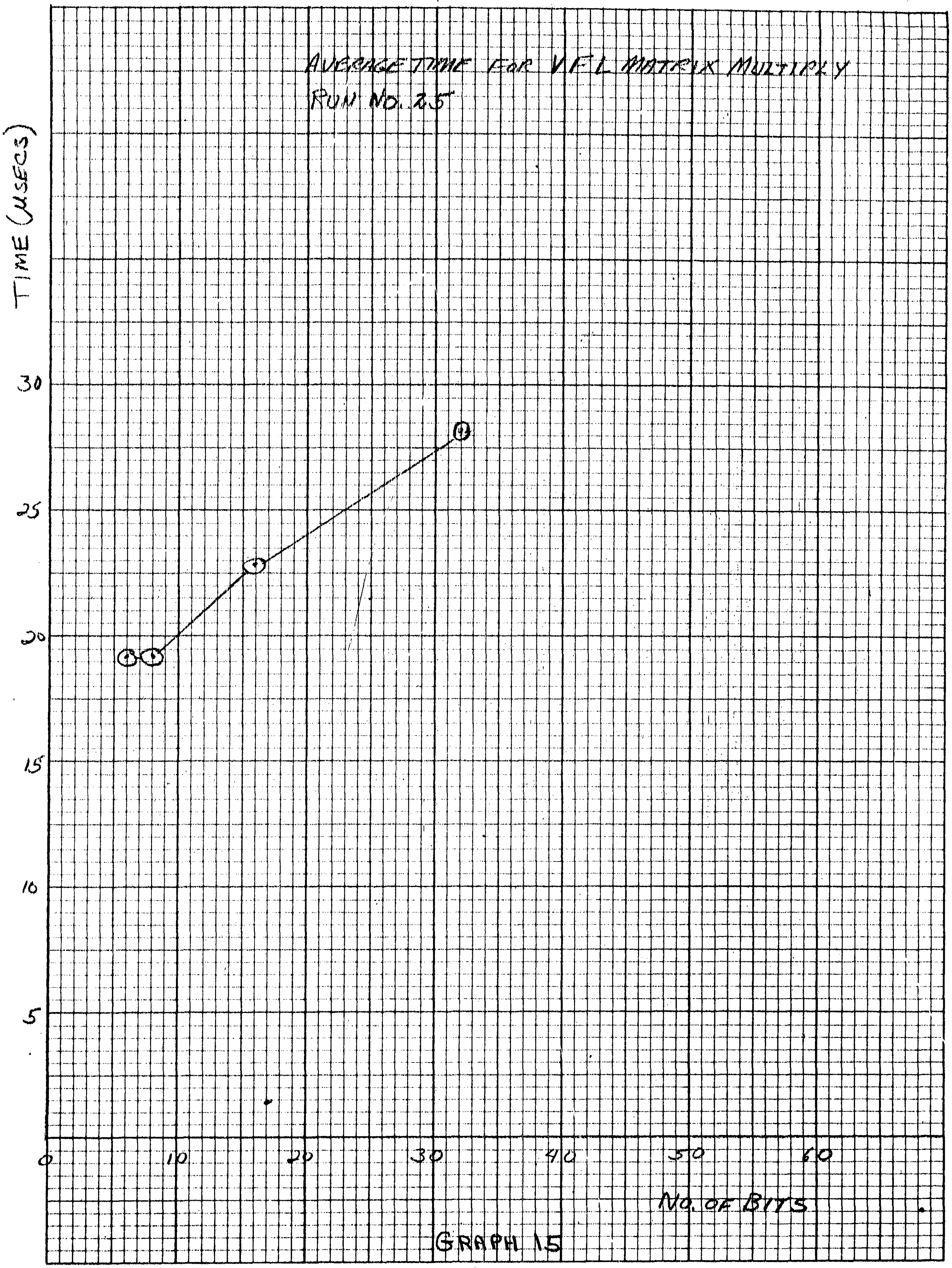
GRAPH 12


AVERAGE SAMPLE PROGRAM TIME
VS. NO. OF BITS
TIME (μSECS)
50
45
40
35
30
25
20
0
10
20
30
40
50
60
NO. OF BITS
Ⓐ
Ⓑ
Ⓒ
Ⓓ
Ⓐ DATA AND INST IN 2 MM; RUN NO. 2 3
Ⓑ DATA IN 2 MM; INST IN 2 FM; "
Ⓒ DATA IN 2 FM; INST IN 2 MM "
Ⓓ DATA AND INST IN 2 FM "

GRAPH 13

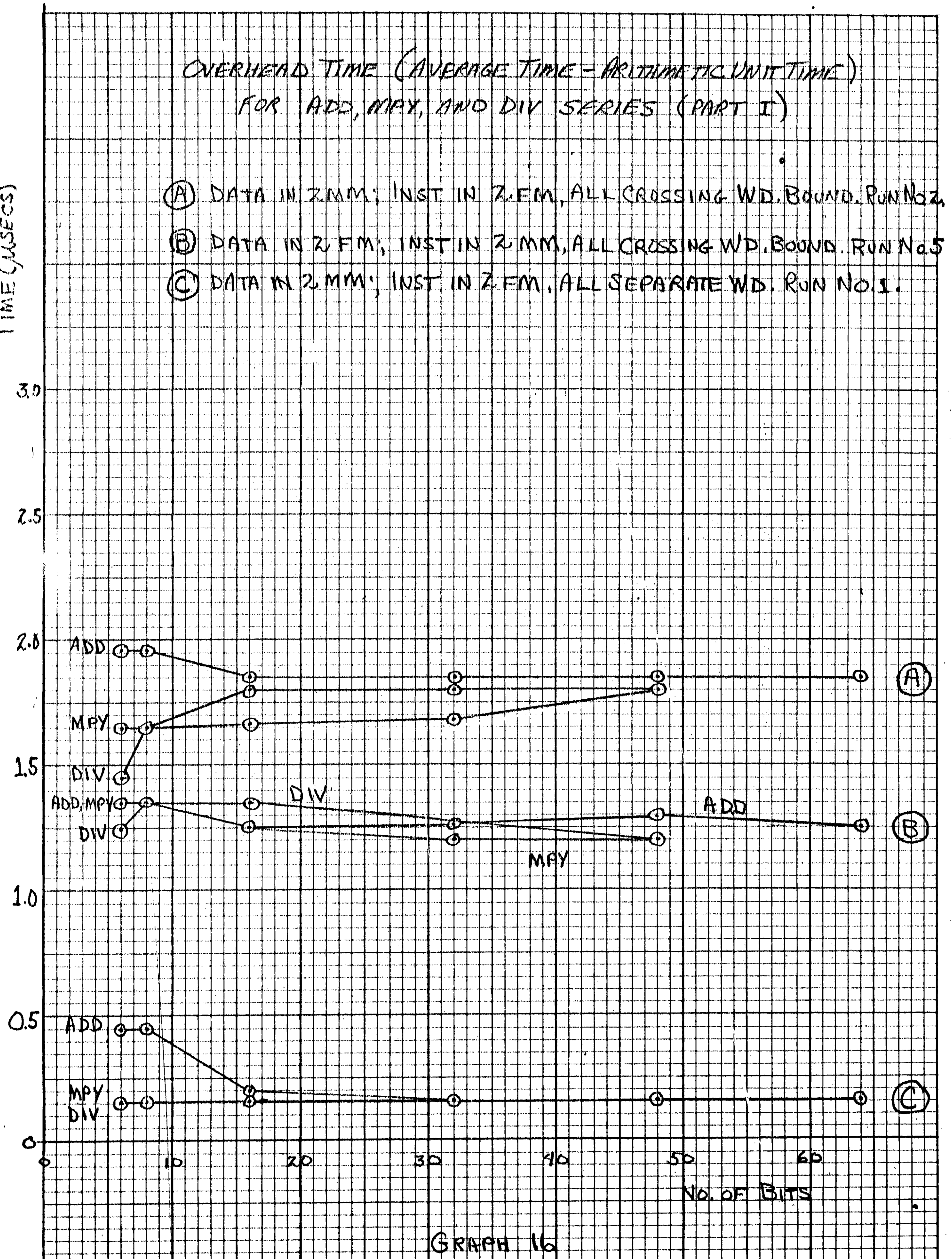

AVERAGE TIME FOR SAMPLE PROG.
WITH EXCHANGES. RUN NO. 24
TIME (μSECS)
100.3
90
80
70
60
50
40
30
20
10
HIGH SPEED EXCHANGE ONLY
2
4
8
16
32
MEDIUM SPEED EXCHANGE RATE

GRAPH 14

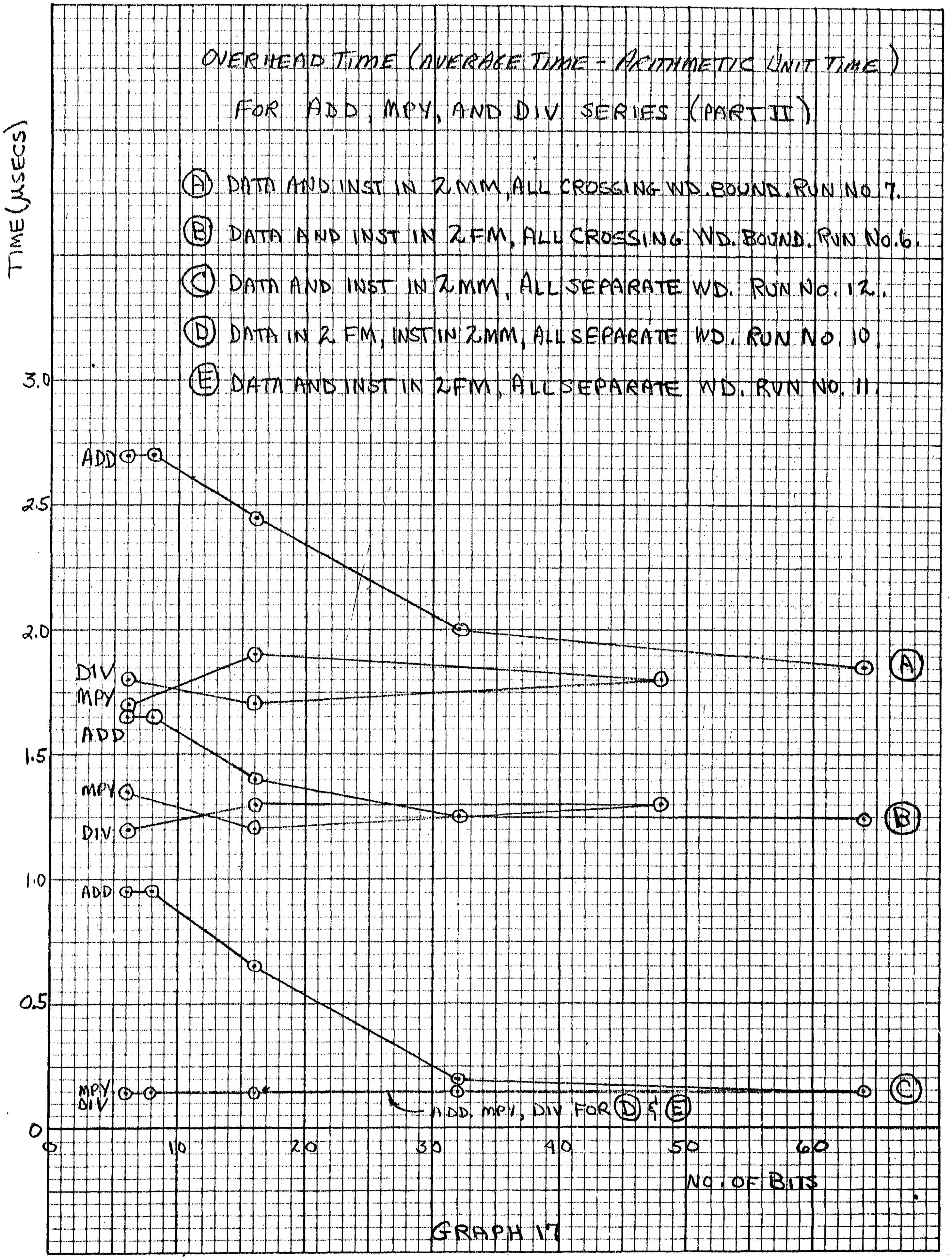

AVERAGE TIME FOR VFL MATRIX MULTIPLY
RUN NO. 25
TIME (μSECS)
30
25
20
15
10
5
0
10
20
30
40
50
60
NO. OF BITS

GRAPH 15

OVERHEAD TIME (AVERAGE TIME - ARITHMETIC UNIT TIME)
FOR ADD, MPY, AND DIV SERIES (PART I)



GRAPH 16


OVERHEAD TIME (AVERAGE TIME - ARITHMETIC UNIT TIME)
FOR ADD, MPY, AND DIV SERIES (PART II)
(A) DATA AND INST IN 2MM, ALL CROSSING WD. BOUND. RUN NO. 7.
(B) DATA AND INST IN 2FM, ALL CROSSING WD. BOUND. RUN No. 6.
(C) DATA AND INST IN 2MM, ALL SEPARATE WD. RUN No. 12.
(D) DATA IN 2 FM, INST IN 2MM, ALL SEPARATE WD. RUN No. 10
(E) DATA AND INST IN 2FM, ALL SEPARATE WD. RUN No. 11.
TIME (μSECS)
3.0
2.5
2.0
1.5
1.0
0.5
0
ADD
DIV
MPY
ADD
MPY
DIV
ADD
MPY
DIV
(A)
(B)
(C)
ADD, MPY, DIV FOR (D) & (E)
0
10
20
30
40
50
60
NO. OF BITS

GRAPH 17