COVER SHEET FOR TECHNICAL MEMORANDUM

TITLE— Bus Interference in a Single Bus Multi-processor Environment

MM72 — 1353 — 16

CASE CHARGED— 39394

DATE— September 20, 1972

FILING CASES— 39394

AUTHOR | LOC. | EXT.
Lycklama, H. | M.H. | 6170

FILING SUBJECTS— Computers
Telephone Switching
Multi-processing
Simulation

ABSTRACT

Introduction

In the past, several multi-processor configurations have been proposed to handle the demands of a telephone switching system. A different solution has been proposed by H. S. McDonald to perform the call-processing functions of a large (~100000 line) DWC (Digital Wire Centre). The particular architecture proposed offers the following attractive features:

(1) high processing throughput capability
(2) modular growth
(3) reliability by means of redundancy

This study will attempt to determine the processing capabilities of the multi-processor and single-bus configuration proposed. Given a certain instruction mix, the factors which affect the throughput are:

NO. OF PAGES— 38

NO. OF REFERENCES— 3

NO. OF TABLES—

NO. OF FIGURES— 20

SEE REVERSE SIDE FOR DISTRIBUTION LIST

(1) size of local processor memory
(2) cycle time of local processor
(3) cycle time of bus
(4) number of processors on the bus

In the case where there are a small number of processors on the bus, throughput is limited strictly by the cycle time of the local processors; whereas in the case of many processors on the bus throughput is limited by bus interference and hence bus cycle time. Throughput of the multi-processor configuration will be determined for various combinations of the parameters listed above. A comparison with No. 1 ESS will give us an estimate of the actual throughput in terms of calls per hour.

DISTRIBUTION
(REFER GEI 13.9-3)

COMPLETE MEMORANDUM TO

CORRESPONDENCE FILES -HO

OFFICIAL FILE COPY
(FORM E-7770) - PLUS
ONE WHITE COPY FOR
EACH ADDITIONAL
FILING CASE
REFERENCED

DATE FILE COPY
(FORM E-1328)

13 REFERENCE COPIES

PATENT DIVISION
IF MEMORANDUM HAS
PATENT SIGNIFICANCE

10 EXD
1352
1353
1356
135 DPH
13 DIR
COHALD
COHART
COHASI

AMORY,R W
ANDERSON,M M
ANDERSON,ROBERT V
ANDERSON,THOMAS F
ANDERSON,WILLIAM A
ARTHURS,EDWARD
BAHLER,L G
+BAKER,W O
BAUGH,C R
BECKETT,J T
BERGLAND,G DAVID
BERRYMAN,R D
BEYER,JEAN-DAVID
BILINSKI,D J
BIREN,MRS IRMA B
BOLSKY,M I
BOYCE,W M
+BOYD,G D
BRAINARD,RALPH C
BREECE,HARRY T III
BROOKS,RICHARD D
BROWNE,T E
+BUCHSBAUM,S J
CAMLET,J V JR
CEMASHKO,FRED
CHANG,HERBERT Y
CHOW,MING-CHWAN
CHRISTENSEN,C
COHEN,HARVEY
+CONDON,J H
CONNOLLY,C V
COOK,ROBERT W
CORASICK,MISS M J
CRUME,LARRY L
+CUTLER,C CHAPIN
DE LUGISH,BRUCE G
DOLAN,MRS MARIE T
DRAGER,JOHN
ESTVANDER,ROBERT A
FETTE,CHARLES J
FEUSTER,I REED
FISCHER,W C
FOUGHT,B T
FRASER,A G
+FREENY,S L
GARCIA,R F
GAY,FRANCIS A
GERTZ,JEFFREY L
GEYLING,F T
GILBERT,MRS HINDA S
GILLETTE,DEAN
GIORDANO,PHILIP P
GITHENS,JOHN A
GORDON,BRIAN G
GRAHAM,R L
GRAVEMAN,R F
GREENLAW,RICHARD L

COMPLETE MEMORANDUM TO

+GUENTHER,R J
HAGELBARGER,D W
HAGGERTY,JOSEPH P
HAHN,J R JR
HAIGHT,R C
HALL,W G
HANNAY,N B
HARTWELL,WALTER T
HAUSE,A D
HEIMBIGNER,G W
HONIG,W L
IPPOLITI,O D
IRVINE,M M
IVIE,EVAN L
JARVIS,JOHN F
JOEL,A E JR
KAISER,J F
KAMINSKI,WILLIAM
KANE,JOHN M
KERNIGHAN,BRIAN W
KOMPFNER,R
KUBIK,P S
LA MARCHE,ROBERT E
LEHRMAN,WILLIAM
+LIMB,J O
LIU,T K
LYCKLAMA,HEINZ
MANCUSI,M D
MARINO,P J
MC EOWEN,JAMES R
MCDONALD,H S
MENON,P R
MICHAEL,ROBERT E
MILLER,S E
+MILLMAN,SIDNEY
MILNE,D C
MOLLENAUER,J F
MOORE,F RICHARD
MORGAN,DENNIS J
MORRIS,ANDREW A JR
NINKE,WILLIAM H
OSSANNA,J F JR
PATEL,C K N
PERDUE,R J
PETERSON,RALPH W
PINSON,ELLIOT N
PLAUGER,P J
PRIEVE,BARTON G
+PRIM,ROBERT C
REPSHER,WILLIAM G
ROBERTS,C S
ROCHKIND,M M
RODRIGUEZ,ERNESTO J
ROSENFELD,PETER E
ROWLINSON,D E
+RUX,PETER T
SALMON,MISS RUTH L
SATZ,L R
SCHEINMAN,ARNOLD H
SCHURTER,W H
SETZER,DAVID E
SEYMOUR,ROBERT F
SHIVELY,RICHARD R
SHORTER,J W
SIPES,J D
SJURSEN,C A
SLANA,M F
SNARE,R C
SPANG,THOMAS C
SPIRES,R J
SPRINGUT,MILTON
STEIGERWALT,P A
STONE,R C
STUKAS,MRS LORETTA I
SWARTZWELDER,JOHN C
TAMMARU,ENN
TEWKSBURY,S K
THOMPSON,JOHN S
THOMPSON,K
+TILLOTSON,L C
TONG,S Y
TOSTO,JOSEPH M
TOY,W
+TUKEY,JOHN W
TUTELMAN,DAVID M
VERMA,SHIV P

COMPLETE MEMORANDUM TO

VILLAROSA,A
WAGNER,BRUCE D
WALKER,MISS E A
WANG,T L
WARNER,JACK L
WARNER,TERRY L
WASSERMAN,MRS Z
WATSON,D S
WEBB,FRANCIS J
WEBSTER,MRS PATRICIA M
WELLER,DAVID R
WILD,J CHRISTIAN
WILLIAMS,R D
WOLONTIS,V MICHAEL
WOOMER,F NELSON
WRIGHT,MISS ROSEMARY A
+YOUNG,JAMES A
ZYDNEY,HERBERT M
162 NAMES

COVER SHEET ONLY TO

CORRESPONDENCE FILES -HO

5 COPIES
PLUS ONE COPY FOR
EACH ADDITIONAL
FILING SUBJECT

127
135
511
522
523
531
533
542
CO
COHA

AAGESEN,JOHN
AARDEN,J E
AARON,D JAMES
ABRAHAM,STUART A
ABRAMS,MRS ELAINE G
ABSHER,JOHN L
ACKERMAN,L J
ACQUAVIVA,V J
ADRIAN,J MARK
AHERN,PETER L
AHO,A V
AHRENS,RAINER B
AITCHESON,E J
ALCALAY,DAVID
ALEXANDER,E J
ALHO,C R
ALLEN,JAMES R
ALLERS,JOHN E
ALMQUIST,R P
AMBEKAR,SUDHIR M
AMES,H S
AMEY,H J
AMRON,I
AMSTER,S J
ANDERSON,D A
ANDERSON,L G
ANGNER,RONALD J
ANTOLICK,D R
ARDON,M T
ARIDAS,E J
ARMAN,MRS S L
ARMBRUST,W D
ARMSTRONG,ANGUIN
ARMSTRONG,DOUGLAS B
ARNDT,DENNIS L
ARNOLD,T F
ASTROM,RICHARD L
ATAL,B S
ATHAY,R D
AVERILL,R M JR
AVEYARD,ROBERT L
AXELSON,ALLEN L
AXON,S L

COVER SHEET ONLY TO

BACKMAN,G A
BAILEY,G G
BALDINGER,W R
BALDWIN,G L
BALDWIN,GARY L
BALDWIN,PATRICK K
BANKS,D D
BARBERIO,J L
BARBER,PAMELA M
BARGE,R F
BARNA,G J
BARNES,MRS A
BARNETT,M P
BARNEY,DUANE R
BARTHOLOMEW,DANIEL R
BARTHOLOMEW,W J
BARTLETT,MRS MAUREEN
BARTLETT,WADE S
BARTON,M E
BATKA,MRS RHEA E
BATTAGLIA,SAMUEL A
BATTISTA,RALPH N
BAUER,MISS H A
BEHMER,LAWRENCE A
BELEK,EMIL
BENGRAFF,W C
BENGTSON,A H
BENSON,G R
BERING,D E
BERNASEK,B H
BERNSTEIN,LAWRENCE
BERRANG,J E
BERRY,W A
BETHEL,LESLIE D
BEUSCHER,H J
BIAZZO,MARTIN R
BILOWOS,RICHARD M
BIRCHALL,R H
BISKOSKI,G B
BLAIR,BILLY W
BLINN,JAMES C
BLOOM,S
BLUMENTHAL,R
BLUM,MRS C
BLY,JOSEPH A
BOCHAR,J
BODEN,F J
BODNAR,J J
BOHACEK,PETER K
BOLAND,T G
BONACHEA,R N
BONNET,R E
BONSER,R H
BORCHERING,J W
BOROWSKI,MRS JANE
BOSWORTH,R H
BOWEN,EDWARD G
BOWERS,T E
BOYCE,PAUL M
BRADY,THOMAS E
BRAHM,D J
BRANDIN,M E
BRANDT,R J
BRAND,JAMES F
BRAND,JOE E
BRAUN,EDWIN J
BRAUN,KENNETH A
BREEN,ROBERT S
BREUNISSEN,J W
BRILEY,B E
BRILLHART,F ROBERT
BRINSFIELD,WILLIAM A
BRISSON,R J
BROWN,ARTHUR L
BROWN,COLIN W
BROWN,DONALD W
BROWN,EARL F
BROWN,G W
BROWN,GORDON W
BROWN,L C
BROWN,P G
BROWN,PAUL F JR
BROWN,W STANLEY
BRUCKMANN,R W
BRUMMEL,MISS M
BUCKNER,MRS PAMELA R

COVER SHEET ONLY

BUCK,I D
BUDRIKIS,Z L
BULFER,ANDREW F
BURKARD,JAMES W
BURT,H LEROY
BUSINGER,P A
BUST,V H
BUTLER,DAVID E
BUTLER,THOMAS T
BUTZIEN,PAUL E
BUYANSKY,DONALD V
BYLANDER,J R
BYRNE,C J
BZOWY,D E
CADWELL,K W
CALHOUN,E T D
CAMPBELL,STEPHEN
CANDY,JAMES C
CANNON,JOHN M
CARBAUGH,DAVID H
CARESTIA,PAUL D
CAREY,J H
CARMICKEL,G N
CARNEY,D L
CARON,L
CARRAN,J H
CARROLL,J J
CARROLL,RONALD L
CASEY,J A
CASEY,JOSEPH P
CASEY,RODGER L
CATTOIR,ROBERT J
CAVINESS,JOHN D
CERMAK,IVAN A
CHANG,S-J
CHANG,TAO-YUAN
CHAPPELL,S G
CHEN,PAUL E
CHERRY,MS L L
CHESSLER,MISS FRA
CHESTON,FRANK C I
CHIN,GEN M
CHMURA,JAMES A
CHOU,R
CHRISTENSEN,D J
CHRISTENSON,D A
CHURCH,C R
CIESIELCZYK,EDWAR
CIESLAK,MRS PAMEL
CIESLAK,THOMAS J
CLARK,GLENN T
CLARK,MISS EDNA L
CLAYTON,D P
CLEMENT,G F
COISMAN,D J
COLDREN,LARRY A
COLLIER,ROBERT J
COLLUM,D J
COLTON,JOHN R
COLYER,WALTER R
COMELLA,WILLIAM K
COMPTON,HARRY B
CONFALONE,D E
CONNOR,DENIS J
COOK,RICHARD F
COONCE,HOMER E
COPP,DAVID H
CORBIN,JOSEPH E
CORNHILL,D T
COSTELLO,J W
COSTELLO,MRS LAUR
COURTNEY-PRATT,J
COURTNEY,GALEN R
COZINE,JOHN J
CRANE,B A
CROSS,MARY-JANE
CROWE,P P
CROXALL,LANCE M
CRUDUP,MISS G D
CUTLER,V H JR
DAHLBOM,CARL A
DAHLING,W B
DANNS,A M
DAVIDSON,CHARLES
DAVIEAU,LEON A
DAVIS,JAMES A

+ NAMED BY AUTHOR

> CITED AS REFERENCE SOURCE

Bell Laboratories

subject: BUS INTERFERENCE IN A SINGLE BUS MULTI-PROCESSOR ENVIRONMENT

date: Sept. 20, 1972

from: H. Lycklama

MM-72-1353-16

MEMORANDUM FOR FILE

Introduction

In the past, several multi-processor configurations have been proposed to handle the demands of a telephone switching system. A different solution has been proposed by H. S. McDonald (1) to perform the call-processing functions of a large (~100000 line) DWC (Digital Wire Centre). The particular architecture proposed offers the following attractive features:

(1) high processing throughput capability

(2) modular growth

(3) reliability by means of redundancy.

This study will attempt to determine the processing capabilities of the multi-processor and single-bus configuration proposed. Given a certain instruction mix, the factors which affect the throughput are:

(1) size of local processor memory

(2) cycle time of local processor

(3) cycle time of bus

(4) number of processors on the bus

In the case where there are a small number of processors on the

bus, throughput is limited strictly by the cycle time of the local processors; whereas in the case of many processors on the bus throughput is limited by bus interference and hence bus cycle time. Throughput of the multi-processor configuration will be determined for various combinations of the parameters listed above. A comparison with NO. 1 ESS will give us an estimate of the actual throughput in terms of calls per hour.

System Configuration

The Common Control for a DWC must satisfy the following requirements:

(1) an open ended call capacity

(2) hardware and software must be as reliable as possible

(3) must have enough real-time capability to manage manual changes and perform testing and maintenance functions as well as call processing.

(4) must be easily expandable in terms of hardware and added software functions.

The system proposed will have two separate common control systems, one for call processing and one for administration and testing. As the two common control systems are identical, both could assume call processing functions during critical periods due to partial outage of the call processing system. This redundancy limits the maximum outage time. A block diagram of one of the common control systems is depicted in Figure 1. Very fast (~150 nsec. cycle time) memory and many parallel processors of two types, mid-level and low level are used to achieve throughput. Mid-level processors are on one of two duplicated

buses and can access all memory modules. Two low-level processors are incorporated into each system memory module and work exclusively on information within that module. These low-level processors perform the real-time functions such as scanning the lines for on-hook and off-hook conditions and handling simultaneous service tasks. These processors access memory in alternate memory cycles from the main memory bus and thus do not interfere with the mid-level processors.

Reliability is achieved through redundancy consisting of dual common control systems, error correction and multiple system elements. The following assumptions are made for the study to be presented in this memorandum. The memory modules will consist of nominally 150 nsec. cycle time solid state memory, 16K words each. The bus will be synchronous with a nominal cycle time of 300 nsec. The mid-level processors will be small, slow processors with a typical cycle time of 5 to 10 usec. per instruction and a word size of 24 bits. Each processor will have a local memory consisting of at least 512 words. Up to 40 identical processors may be connected to the bus. A bus controller will scan request lines from the processors in a "round robin" fashion and grant access to the bus.

As the low-level and mid-level processors access memory in different memory cycles, the low-level processors do not contribute to memory interference and will not be considered in this study. Also the mid-level processors on the duplicated bus do not introduce memory interference. Thus this study will be concerned only with the throughput of the call-processing capability

of a single bus system which has attached to it M memory modules and N mid-level processors. The throughput of a duplicated system will be essentially double that of a single unduplicated system.

Instruction Mix

The processor under study is the CSX processor designed and built by D. C. Milne (2). The processor as it now exists is microprogrammed and has a basic primitive cycle time of 300 nsec. Its local memory consists of 256 words currently but is expandable. It is possible to characterize each processor instruction as consisting of a number of primitives and a number of memory accesses. The access may be either to local memory (hence requiring no bus interaction) or to the system memory consisting of a number of modules. Also the execution of each instruction consists of an instruction fetch and the fetch of one or more operands, each of which may or may not require a bus access. The instructions executed by the processor can be categorized into 6 different types as depicted in Figure 2. The first type of instruction is simply the fetch of the instruction and subsequent execution without another memory reference. The second type involves the fetch of an operand as well as of the instruction. The third type is the RAW (read-alter-write) instruction which involves the reading and subsequent writing of the operand in a second memory reference. The MOV instruction is unique to this processor in that it involves the fetch of the instruction followed by the moving of up to 64 words from 64 consecutive locations in memory to another 64 consecutive locations in memory.

The other two types of instructions are similar to types 2 and 3 with one more memory fetch required due to one level of indirection.

Of the 6 types of instruction discussed, only the first 4 will be considered in this study as the last two occur so infrequently in a general register machine. By counting the occurrences of each type of instruction in some code which has been written for this processor it was found that the frequency of each was as follows:

Type - inst.	Freq.
1 operate	0.515
2 memory ref.	0.380
3 RAW inst.	0.100
4 MOV inst.	0.005

It has been found that code is executed in very nearly the same proportion as it appears in linear code on the basis of instruction types. This approximation has been taken in this study and hence the instruction mix which appears above has been taken throughout.

The probabilities of the executed instructions being in local memory (PILC) and of the requested operands being in local memory (POLC) are a measure of the size of local memory. The larger the local memory of each processor is, the greater the probability will be of finding both the instruction and the operand in local memory and hence the less the bus interference will be. In general, software could be designed such that the probability of finding an instruction in local memory would be

enhanced; but the probability of finding operands in local memory for a large data base may be quite small. As it is difficult to estimate the parameters PILC and POLC as a function of local memory size and more so yet to estimate the ratio of these two parameters, PILC has been taken to be equal to POLC throughout this study. The actual value taken for POLC or PILC is then some weighted average of the true values of PILC and POLC.

Analytical Results

It is possible given the length of each type of instruction as well as the time during which it may request a bus transaction and the probabilities of that interaction, to come up with an analytical expression for the limits of the throughput of the multi-processor system. Each instruction type is comprised of a fixed number of primitive cycles plus a bus cycle for each reference to system memory, as shown in Figure 2. For example, type 1 instructions execute eight (8) primitives before fetching the instruction. If the instruction is in local memory, one primitive is taken to fetch it; however, if a reference to system memory is required, at least one bus cycle is necessary depending on the number of processors making bus requests at the time. Seventeen (17) primitives are taken to execute the instruction itself. Thus the basic instruction time is given by:

INST(1)=IOP2

However taking into account that the probability (PILC) that the instruction is local is not equal to one (1.0), the average time to execute instruction type 1 is given by:

INAV(1)=IOP2+(BTIM-IPRM)*(1-PILC) (1)

assuming no wait for the bus access is necessary. Here BTIM is the bus cycle time and IPRM is the processor primitive cycle time. Similarly the average execution times for the other three instruction types are given by:

INAV(2)=IDR2+IDR4+(BTIM-IPRM)*(2-PILC-POLC) (2)

INAV(3)=IRW2+IRW4+(BTIM-IPRM)*(3-PILC-2*POLC) (3)

INAV(4)=IMV2+128*IMV4+IMV5+(BTIM-IPRM)*(65-PILC) (4)

The execution time for the MOV instruction assumes that the instruction itself may or may not be in local memory, but that either the source or destination address is definitely in local memory. One can generalize these results as follows:

INAV(i)=INST(i)+(BTIM-IPRM)*((1-PILC)+NOPF(i)*(1-POLC)) (5)

where NOPF(i) is the number of operand fetches for instruction type i.

Each instruction type has a certain probability PTP(i) of being executed. Therefore the average execution time for one instruction for any processor is simply:

IAVE= PTP(i)*INAV(i) (6)

It then follows that the maximum instruction rate for a given processor is given by:

MXRT=1/IAVE (7)

instructions per second. For an N processor configuration the maximum instruction rate is:

MXRT=N/IAVE (8)

However this is only a theoretical limit and is not practically attainable due to bus interference.

The theoretical limit to the instruction rate due to bus

interference can be calculated for any given set of parameters. This limit is a function of the number of bus requests which can be handled per second and is not a function of the number of processors on the bus. The average number of bus requests per instruction is given by:

BRAT= PTP(i)*((1-PILC)+NOPF(i)*(1-POLC)) (9)

Hence given that a bus cycle occurs in time BTIM the absolute maximum number of instructions which can be executed per second is given by:

INMX=1/(BTIM*BRAT) (10)

assuming no bus interference. However bus interference does occur and reduces the number of instructions which can actually be executed per second. No analytical expression can be obtained for the loss of throughput due to bus interference because of the complexity of the factors involved. Therefore the exact nature of the bus interference was simulated to obtain a quantitative measure of loss in throughput.

Simulation Results

A discrete event simulation model was constructed to obtain a quantitative measure of the loss in throughput due to bus interference of the proposed multi-processor, single bus system configuration. The model was programmed in the FSNAP language (3) and run on the DDP-516 machine in Dept. 1352 under 516-TSS. A complete listing of the program appears in Appendix A.

The results of all simulation runs can be interpreted in terms of the concepts portrayed by Figure 3. Throughput is measured in terms of MIPS (million instructions per second). The

horizontal line shown on the typical set of results in Figure 3 represents the maximum throughput possible given by equation (10) for a given set of values of BTIM, POLC, PILC and PTP(i). The ramp function shown is described by equation (8) and is of course a linear function of the number of mid-level processors on the bus. Equation (10) is very dependent on the bus cycle time whereas equation (8) is highly dependent on the mid-level processor cycle time. The point at which the two functions intersect is given by:

NC=INMX*IAVE (11)

and is the point at which the maximum theoretical throughput could be achieved if there were no bus interference at all. The simulation results typically indicated a loss in throughput at this point.

In all simulation runs whose results are discussed below the instruction mix taken was:

PTP(1)=0.515

PTP(2)=0.380

PTP(3)=0.100

PTP(4)=0.005

Throughput was found to be fairly insensitive to small variations in the instruction mix. The parameters which had a distinct affect on throughput and were varied over suitable ranges were the following:

BTIM - bus cycle time

IPRM - processor primitive cycle time

PILC - probability of instruction fetch from local memory

POLC - probability of operand fetch from local memory

N - number of mid-level processors on the bus.

The ranges of the various parameters were as follows:

BTIM - 200 nsec. to 400 nsec.

IPRM - 200 nsec. to 400 nsec.

PILC=POLC - 0.0 to 0.95

N - 1 to 40

In each individual simulation run the total real time simulated was of the order of 5 milli-seconds.

The first processor configuration simulated included processors with 300 nsec. primitive cycle time. and 300 nsec. bus cycle time. The probability of an instruction or an operand being in local memory was taken to be 0.25. The configuration was simulated for 1, 2, 4, 8, 12, 16, 20, 24, 32 and 40 processors. A typical simulation run output is included in Appendix B. Each configuration was simulated at least three times to achieve statistically significant results. The results which are shown in Figure 4 are actually the average values of many runs. From the given set of parameters one can calculate:

INMX = 2,214,840 instructions per second

MXRT = 122,680 instructions per second per processor.

From the results in Figure 4, one sees that the loss in throughput at the point defined by equation (11) i.e. NC=18 processors, is:

loss = ((2214840-1920000)/2214840)*100 = 13.31%

Only by adding more processors to the configuration is it possible to achieve near the maximum theoretical throughput. However, the slope of the curve decreases rapidly beyond this point and it may become uneconomical to add more processors after a certain

number. The effective cost of a processor actually increases as one adds more processors as the throughput per processor decreases. In this case the maximum throughput is approached with a configuration of 24 processors and it would certainly not pay to add more processors to achieve more throughput. However it may be advantageous to add more processors to achieve redundancy in case of the failure of one or more processors.

The same configuration was then simulated again but this time with the probability of an instruction or an operand being in local memory of 0.0. Here INMX=1,754,386 instructions per second and MXRT=122,680 instructions per second per processor. The loss in throughput defined by equation (11) at NC=14 processors is 13.6%. These results are depicted in detail in Figure 4. The other curves in Figure 4 are for the values of PILC=POLC=0,0.50,0.75 and 0.95.

In order to determine the effect on throughput of the primitive cycle time of the processors, the simulation runs were run for values of the primitive cycle time of 200 and 400 nsec. The results are portrayed in Figures 5 and 6 respectively. The horizontal lines determining the absolute maximum instruction rate of the configurations remain the same as the corresponding ones in Figure 4 as determined by equation (10). However the instruction rate per processor as determined by equation (7) increases the slope of the ramp function as a function of the primitive cycle time. The slope of the ramp function is slightly different for each value of PILC and POLC but is barely discernible on the scales at which Figures 5 and 6 are drawn.

In order to determine the effect on throughput of the bus cycle time, the processor primitive cycle was held constant at 300 nsec. while the bus cycle time was varied from 200 nsec. to 400 nsec. The results of these simulation runs are shown in Figures 7 and 8 respectively. Here the slope of the ramp functions as determined by equation (7) remains relatively constant. However the maximum possible instruction rate as determined by equation (10) varies inversely as the bus cycle time. Hence the wide variations in the maximum instruction rate as a function of bus cycle time for corresponding configurations in Figures 4, 7 and 8 respectively.

Figures 4 to 8 inclusive portray the main results of all the simulation runs carried out in this study. They represent a total of more than 1000 hours of actual run time on the 516-TSS computer system. These figures show the basic dependence of throughput on the number of processors in the configuration as well as the dependence on bus cycle time and processor primitive cycle time. They show that any one configuration has a definite maximum throughput capability which cannot be exceeded regardless of how many processors are put on the bus. However the dependence of throughput on other parameters can best be shown by other graphs of the results.

Figure 9 shows the dependence of throughput on PILC and POLC. Three curves are drawn, one for each bus cycle time of 200, 300 and 400 nsec. respectively. Halving the bus cycle time doubles the maximum instruction rate. At low probabilities of the instructions and the operands being in local memory, the

curves slope upward very slowly; however, as PILC and POLC approach the value of one (1.0), the maximum throughput approaches the theoretically unobtainable value of infinity, limited only by the number of processors on the bus. This demonstrates that at low values of PILC and POLC throughput is enhanced much more by decreasing the bus cycle time than by increasing the size of the local memory.

Another important parameter in this study is the number of processors required in each configuration to achieve the theoretically maximum throughput at the cut-off point as defined by equation (11). Figure 10 shows the dependence of this parameter on PILC and POLC for each of the five different configurations simulated in this study. The cut-off point represents the approximate number of processors which could be put on the bus for each configuration to make reasonably efficient use of all of the processors with small loss in throughput. The higher the cut-off point is, the larger the number of processors which must be utilized on the bus to obtain the maximum throughput.

The instruction rate per processor is determined by equation (7). The rate is very configuration dependent but only slightly dependent on the parameters PILC and POLC as depicted in Figure 11. Here throughput is measured in terms of KIPS (thousand instructions per sec.). In fact the instruction rate per processor is not very dependent on the bus cycle time either as shown by the three curves for IPRM=300 nsec. and BTIM=200, 300 and 400 nsec. respectively. Of course the instruction rate per processor is very dependent on processor primitive cycle time.

The results summarized in Figure 4 can be shown from a different perspective to demonstrate the dependence of throughput on the parameters PILC and POLC for various numbers of processors on the bus for IPRM=300 nsec. and BTIM=300 nsec (see Figure 12). For 8 and 16 processors on the bus the throughput does not increase very rapidly as a function of PILC and POLC as there is very little bus interference generated by references to system memory. However, for more than 16 processors on the bus the throughput at PILC=POLC=0.0 is restricted severely by interference on the bus. Throughput increases slowly as a function of PILC and POLC at low values of these independent variables. The slopes of these throughput curves increase in the mid-range of PILC and POLC and then gradually approach zero again towards PILC=POLC=1.0, at which maximum throughput is obtained.

The dependence of throughput on the various parameters can also be discerned by looking at the throughput for a specific number of processors on the bus (e.g. 24) for the 5 various configurations studied as a function of the parameters PILC and POLC (see Figure 13). It is interesting to note that the three curves for BTIM=300 nsec. and IPRM=200 ,300 and 400 nsec. respectively converge near PILC=POLC=0.0 as the processor primitive cycle time has little effect on throughput here for a fixed number of processors. On the other hand the three curves for IPRM=300 nsec. and BTIM=200, 300 and 400 nsec. respectively converge as PILC and POLC approach 1.0 since the parameter BTIM has little effect on throughput when very few accesses are made to system memory.

Loss in throughput is due to two related factors, loss when

there is interference on the system bus and loss when all possible cycles on the bus are fully utilized. A close look at the results portrayed in Figures 4 through to 8 shows that there is an increasing loss in throughput as the number of processors approaches the cutoff point. Up to this point loss is entirely due to interference on the bus. Beyond this point loss is due partly to interference on the bus but increasingly due to maximum possible utilization of all possible cycles on the system bus. The maximum loss in throughput due solely to bus interference which occurs at the cut-off point is shown in Figure 14 for the 5 various configurations simulated as a function of the parameters PILC and POLC. Note that on the average the percentage loss at this point increases as the ratio of BTIM to IPRM increases. Some of this loss is due to the fact that an instruction which accesses system memory takes longer than one which does not, even though that particular instruction does not generate bus interference.

Conclusions

An attempt will now be made to interpret the simulation results in economic terms and in relation to NO. 1 ESS capabilities. The total cost of a system configuration is given by:

$$COST=CSTM+CSTP*N \qquad (12)$$

where the cost function CSTM is the cost of the basic system configuration including high speed system memory and the system bus to which all of the local processors are interfaced. The parameter CSTM is very dependent on the bus cycle time since the cost increases quite rapidly with decrease in memory cycle time and

bus cycle time. Of course this parameter assumes a fixed size system memory. The parameter CSTP is the cost of one local processor with a primitive cycle time of IPRM and a given fixed size of local memory. The total system cost is made up of two parts. It is maintained here that the cost factor CSTM will make up the largest part of the total for a reasonable size of local memory per processor. Then the total addition to the cost of the system of adding a few processors should be quite minimal since the cost function increases linearly as the number of processors on the bus. Even with a large number of processors on the bus the total cost of all the processors should be much less than the cost of the system represented by the fast system memory and the fast system bus. Since each processor in itself is relatively slow with a small amount of local memory, the cost per processor should be low.

As a reasonable estimate, it is expected that with 256K fast system memory, a total of 40 processors with 4K local memory each could be added to the system before the cost of the processors would approach the cost of the system memory and the system bus. However, one must bear in mind in making these calculations that the effective cost of a processor may be somewhat higher than its actual cost when one considers the real throughput of a processor in terms of the maximum obtainable throughput as given by equation (7). The effective cost of a processor is lowest when all processor primitive cycles are used, whereas the effective cost of the system memory and system bus is lowest when all bus cycles are fully utilized. Hence a system configuration operating near the cut-off point should prove to give the most throughput per

dollar. However the optimum operating point can only be obtained if one knows the effective cost of the processors relative to the effective cost of the system memory and system bus.

One can define the effective cost of a processor in terms of the cost of executing an instruction:

CSTI(P)=CSTP*N/IRAT (13)

where IRAT is the instruction execution rate as shown in Figures 4 to 8 for the particular configuration under study. The cost of executing an instruction at the maximum instruction rate per processor, as defined by equation (7), is given by:

CSTX(P)=CSTP*N/MXRT (14)

Then the effective cost of a processor is given by the ratio of equation 13 to 14 times the cost of one processor:

ECSTP=CSTP*MXRT/IRAT (15)

Figure 15 depicts the effective cost of a local processor as a function of the number of processors on the system bus for a processor primitive cycle of 300 nsec. and PILC=POLC=0.25 for the bus cycle times of 200, 300 and 400 nsec. respectively. Note that the bus cycle time has quite a marked effect on the effective cost of a processor for a relatively large number of processors where throughput is bus limited. As a first approximation, one can make the assumption that the cost of a processor is inversely proportional to the primitive cycle time. To speed up a processor one can increase the complexity of the control logic so that one primitive cycle executes a larger part of an instruction than before, thus requiring less primitives to do the instruction. Less ROM (read only memory) is required, but the increased complexity will increase the cost of the processor. As far as

simulation is concerned, the effect of less primitives is equivalent to an equal number of shorter primitives. Assuming a linear dependence of cost per processor on the inverse of processor primitive cycle time, Figure 16 gives a good indication of how the 5 various configurations compare in effective cost of a processor as a function of the number of processors on the bus.

Since the relationship between PILC and POLC and the size of local memory is not clear, it is difficult to assign a cost to a processor as a function of PILC and POLC. A detailed study of some actual code could be used to determine this relationship accurately. Taking the particular configuration of IPRM=BTIM=300 nsec., the relative effective cost of a processor will follow the curves shown in Figure 17. Here the assumption is made that the cost of a processor to obtain 25% reference to local memory would be a factor of 2 greater than the cost of one which has no local memory. To obtain 95% reference to local memory would increase the cost of a basic processor by a factor of 5. These factors are only "ballpark" figures, but the shape of each curve is independent of these assumptions. The actual factors which should be used are very dependent on the technology used to build the processor and its local memory as well as on the design of the software. In a similar manner curves can be obtained analogous to Figure 17 for the other configurations which were studied here. However, the curves are only valid as a guide to the effective cost of a processor since absolute costs are not known.

In a similar manner the effective cost of the system memory and the system bus can be defined in terms of executing an

instruction as:

CSTI(M)=CSTM/IRAT (16)

Here the cost of executing an instruction approaches a minimum when the instruction rate approaches the maximum obtainable, as defined by equation (10):

CSTX(M)=CSTM/INMX (17)

Then the effective cost of the memory and bus system is given by the ratio of equation 16 to equation 17 times the actual cost of the system memory and system bus:

ECSTM=INMX*CSTM/IRAT (18)

For the total system, including system memory, system bus and local processors, the cost of executing as instruction is given by:

CSTI=(CSTM+CSTP*N)/IRAT (19)

Making certain assumptions about the ratio of the cost of a processor to the total cost of the system memory and system bus, one can plot the curves shown on Figure 18. This set of curves is for the particular configuration with IPRM=BTIM=300 nsec. and PILC=POLC=0.25. The lower the cost of a processor, the better the cost effectiveness appears to be. In all three curves, the most economic configuration seems to be in the rather broad range of 16 to 32 processors. For the case of PILC=POLC=0.0, the results portrayed in Figure 19 indicate that the most economical operating point is more sharply defined than that for PILC=POLC=0.25. It is likely in this case that the cost of a processor is a small fraction of the total system memory, since a processor here has essentially no local memory. Therefore the lower curve is probably nearer reality. However for

PILC=POLC=0.95, the most economical operating point seems to be near N=40, regardless of the cost of a processor, as shown in Figure 20. The cost of a processor is relatively high in this configuration, so that the the upper curves will more closely describe the actual cost function. Basically these results as portrayed in Figures 18 to 20 indicate that regardless of the size of local memory, the cost of a processor is still much less than the total cost of the total system and therefore there is a large range of number of processors over which the various configurations are economical.

The capacity of a NO. 1 ESS central control office is of the order of 66000 calls per hour (ref.). This is with a processor with 5.5 usec. basic cycle time per instruction, i.e. 0.182 MIPS instruction rate. The processing of a typical intra-office call requires 5078 cycles. Given that the word size of the ESS machine is 37 bits and that that of the proposed processor is 24 bits, a call may require 37*5078/24 instructions. However many tasks required to handle a call which are done by software in NO. 1 ESS, are performed by the low-level processors in the current proposal and therefore the actual number of instructions required may actually be much less than 8000. Using the conservative estimate that a call requires as many instructions as in NO. 1 ESS, the instruction rate required to obtain 500000 call attempts per hour would be 0.182*500000/66000 = 1.40 MIPS. Most of the configurations in Figures 4 to 8 satisfy this criterion quite easily. Thus the throughput requirement is satisfied by the proposed configuration. However the response of the system and the reliability of the over-all system are questions which cannot be

answered by these simulation results.

The above study of the proposed multi-processor, single-bus system has shown that the throughput capacity is adequate to handle a 100000 line office. An accurate measure of throughput can only be obtained by defining the call-processing tasks in terms of instructions per task. Throughput is very configuration dependent. The nature of this dependence on size of local memory, on cycle time of a local processor, on cycle time of the system bus and on number of processors on the bus has been shown in detail in the study above. The factors affecting the economics of each configuration have been pointed out. Given these results it is possible to choose the configuration which will give adequate throughput for a given size office and prove to be most economical.

Subsequent to the work which is described in this memo, improvements have been made to the design of the local processor. Variable speed primitives have been implemented in the control logic. This has the effect of decreasing the average instruction time. The effect can be simulated by decreasing the average primitive cycle time. Some instructions also take less primitives to execute than with the previous local processor. This effect can again be produced by decreasing the average primitive cycle time. Despite these changes, interaction with the system bus remains essentially the same and the results obtained here remain valid in a qualitative manner. The improvements to the local processor have the effect that a smaller number of processors are now required to obtain the same throughput (generating

the same amount of bus interference) as with the larger number of processors used to obtain the results in this memo.

MH-1353-HL-JER

H. LYCKLAMA

H. Lycklama

Att.
References
Appendices A and B
Figures 1-20

References

1. Memorandum by H. S. McDonald on a Large Digital Wire Center (July 1971).

2. Private communication with D. C. Milne.

3. FSNAP User'S GUIDE BY H. Lycklama (Dept. 1352 Document September 1971).

List of Figures

16. Relative effective cost per processor as a function of the number of processors on the bus for PILC=POLC=0.25 for the various configurations assuming negligible increase in cost per processor as a function of primitive cycle time.
17. Relative effective cost per processor as a function of the number of processors on the bus for IPRM=BTIM=300 nsec. and various values of PILC and POLC assuming an increase in cost of a processor as a function of PILC and POLC as given.
18. Relative cost per instruction as a function of the number of processors for the system configuration with IPRM=BTIM=300 nsec. and PILC=POLC=0.25.
19. Relative cost per instruction as a function of the number of processors for the system configuration with IPRM=BTIM=300 nsec. and PILC=POLC=0.0.
20. Relative cost per instruction as a function of the number of processors for the system configuration with IPRM=BTIM=300 nsec. and PILC=POLC=0.95.

APPENDIX A - Program Listing

```
IOFLG=0
MPCL=8000
DIM       ITIM(40),ITYP(40),ISTG(40),ICNT(40,6)
DIM NTIM(40),NCNT(40),INST(6),PCNT(6)
DIM IBRQ(40),INAV(6)
READ  NTX,RAN,MXTM,NTYP,IPRM,BTIM
READ PTP1,PTP2,PTP3,PTP4,PILC,POLC
IOP1=8*IPRM
IOP2=26*IPRM
IOP3=17*IPRM
IDR1=8*IPRM
IDR2=23*IPRM
IDR3=14*IPRM
IDR4=4*IPRM
IDR5=3*IPRM
IRW1=8*IPRM
IRW2=23*IPRM
IRW3=14*IPRM
IRW4=5*IPRM
IRW5=3*IPRM
IMV1=8*IPRM
IMV2=12*IPRM
IMV3=3*IPRM
IMV4=1*IPRM
IMV5=4*IPRM
IP=IPRM*50
IT=BTIM*50
```

```
INST(1)=IOP2
INST(2)=IDR2+IDR4
INST(3)=IRW2+IRW4
INST(4)=IMV2+128*IMV4+IMV5
INAV(1)=INST(1)+(BTIM-IPRM)*(1-PILC)
INAV(2)=INST(2)+(BTIM-IPRM)*(2-PILC-POLC)
INAV(3)=INST(3)+(BTIM-IPRM)*(3-PILC-2*POLC)
INAV(4)=INST(4)+(BTIM-IPRM)*(65-PILC)
BRAT=PTP1*(1-PILC)+PTP2*(2-PILC-POLC)+PTP3*(3-PILC-2*POLC)+PTP4*(
INMX=MXTM/(BTIM*BRAT)
MXRT=MXTM/(PTP1*INAV(1)+PTP2*INAV(2)+PTP3*INAV(3)+PTP4*INAV(4))
WRITE !"INTERFERENCE SIMULATION STATISTICS FOR N PROCESSORS ON CS
WRITE ! %2,NTYP " TYPES OF INSTRUCTIONS"
WRITE !%4.3 "PROB(1)="PTP1" PROB(2)="PTP2" PROB(3)="PTP3" PROB(4)
WRITE ! "PRIMITIVE CYCLE TIME ="%3,IP " NSEC"
WRITE ! "BUS CYCLE TIME =" %4,IT " NSEC"
WRITE ! "PROB. OF INSTRUCTION FETCH FROM LOCAL MEMORY =" %6.2,PILC
WRITE ! "PROB. OF OPERAND FETCH FROM LOCAL MEMORY =" %6.2,polc
WRITE ! "AVERAGE MAXIMUM NUMBER OF INSTRUCTIONS PER PROCESSOR ="
WRITE ! "MAXIMUM INSTRUCTION THROUGHPUT =" %8.2 INMX
WRITE " (" %8 INMX*1E9/(50*MXTM) " /SEC. )"
WRITE ! "CUT-OFF POINT OCCURS AT" %6.2 INMX/MXRT "  PROCESSORS"
PTP2=PTP1+PTP2
PTP3=PTP2+PTP3
FOR ITX=1,NTX
READ NPRC
WRITE!!! %2,NPRC " PROCESSORS *** RANDOM NUMBER GENERATOR ="%9.6,
TIME=0
```

```
      FOR I=1,NPRC
      FOR J=1,NTYP
      ICNT(I,J)=0
      PCNT(J)=0
      NEXT J
      ISTG(I)=1
      ITIM(I)=0
      NTIM(I)=0
      NCNT(I)=0
      IBRQ(I)=0
      NEXT I
      FOR I=1,NPRC
      GOSUB 15
      NEXT I
      J=NPRC
      GOTO 180
140   ITP=ITYP(J)
      ICNT(J,ITP)=ICNT(J,ITP)+1
      I=J
      GOSUB 15
180   IX=J
      MTIM=10000000
      FOR K=1,NPRC
      IX=IX+1
      IF(IX>NPRC)IX=1
      IF(ITIM(IX)<TIME)ITIM(IX)=TIME
      IF(ITIM(IX)=>MTIM)GOTO 200
      MTIM=ITIM(IX)
```

```
      J=IX
200   NEXT K
      IX=INT((MTIM-TIME+BTIM)/BTIM)*BTIM
      IF(IX<>MTIM-TIME+BTIM)IX=IX+BTIM
      TIME=TIME+IX
      IBRQ(I)=IBRQ(I)+1
      IF(TIME>MXTM)GOTO 2000
      ITP=ITYP(J)
      IF(ITP-2)300,400,500
300   ITIM(J)=TIME   +IOP3
      GOTO 140
400   IF(ISTG(J)=1)GOTO 410
      ITIM(J)=TIME   +IDR5
      GOTO 140
410   ITIM(J)=TIME   +IDR3
      RAN=RND(RAN)
      IF(RAN>POLC)GOTO 432
      ITIM(J)=ITIM(J)+IDR4
      GOTO 140
432   ISTG(J)=2
      GOTO 180
500   IF(ITP=4)GOTO 600
      IF(ISTG(J)-2)510,520,530
510   ITIM(J)=TIME   +IRW3
      RAN=RND(RAN)
      IF(RAN>POLC)GOTO 532
      ITIM(J)=ITIM(J)+IRW4
      GOTO 140
```

```
532   ISTG(J)=2
      GOTO 180
520   ITIM(J)=TIME
      ISTG(J)=3
      GOTO 180
530   ITIM(J)=TIME+IRW5
      GOTO 140
600   IF(ISTG(J)>1)GOTO 610
      ITIM(J)=TIME+IMV3
      ISTG(J)=2
      GOTO 180
610   ITIM(J)=TIME+IMV4
      ISTG(J)=ISTG(J)+1
      IF(ISTG(J)=<65)GOTO 180
      ITIM(J)=ITIM(J)+IMV5
      GOTO 140
2000  TPRC=0
      TITM=0
      TNTM=0
      NBRQ=0
      TCNT=0
      WRITE ! " PROC #"
      FOR J=1,NTYP
      WRITE " INST"%1,J
      NEXT J
      WRITE "  TOTAL   TIME  EFF. TIME  % LOSS"
      FOR I=1,NPRC
      FOR J=1,NTYP
```

```
      NTIM(I)=NTIM(I)+ICNT(I,J)*INST(J)
      NCNT(I)=NCNT(I)+ICNT(I,J)
      NEXT J
      NTIM(I)=NTIM(I)+(BTIM-IPRM)*IBRQ(I)
      ITP=ITYP(I)
      IF(ITP-2)2051,2052,2053
2051  NTIM(I)=NTIM(I)+IOP1
      GOTO 2060
2052  IF(ISTG(I)=1)NTIM(I)=NTIM(I)+IDR1
      IF(ISTG(I)=2)NTIM(I)=NTIM(I)+IDR2
      GOTO 2060
2053  IF(ITP=4)GOTO 2054
      IF(ISTG(I)=1)NTIM(I)=NTIM(I)+IRW1
      IF(ISTG(I)=2)NTIM(I)=NTIM(I)+IRW2
      IF(ISTG(I)=3)NTIM(I)=NTIM(I)+IRW2+BTIM
      GOTO 2060
2054  IF(ISTG(I)=1)NTIM(I)=NTIM(I)+IMV1
      IF(ISTG(I)=>2)NTIM(I)=NTIM(I)+IMV2+2*IMV4*(ISTG(I)-2)
2060  PRC=100*(ITIM(I)-NTIM(I))/ITIM(I)
      TPRC=TPRC+PRC
      IF(IOFLG<>0) WRITE ! %5,I,%6
      FOR J=1,NTYP
      IF(IOFLG<>0)WRITE ICNT(I,J)
      PCNT(J)=PCNT(J)+ICNT(I,J)
      NEXT J
      IF(IOFLG<>0)WRITE  %6,NCNT(I),%7,ITIM(I),%9,NTIM(I),%7.3,PRC
      TCNT=TCNT+NCNT(I)
      TITM=TITM+ITIM(I)
```

```
    TNTM=TNTM+NTIM(I)
    NBRQ=NBRQ+IBRQ(I)
    NEXT I
    WRITE ! "  AVE." %5.1
    FOR J=1,NTYP
    WRITE PCNT(J)/NPRC
    NEXT J
    WRITE TCNT/NPRC,%7,TITM/NPRC,%9,TNTM/NPRC,%7.3,TPRC/NPRC
    WRITE ! "TOT. BUS REQ. =" %6 NBRQ
    WRITE "  TOT. INST. =" TCNT
    WRITE "  MAX. INST. =" NPRC*MXRT
    WRITE ! "THROUGHPUT FACTOR =" %8.2  100-TPRC/NPRC
    MIPS=TCNT/((TITM/NPRC)*50/1E9)
    WRITE "  INST. RATE =" %8 MIPS " /SEC. "
    WRITE ! "I. E." MIPS*3600/MPCL "  CALLS PER HOUR"
    WRITE "  (" %5 MPCL " INSTRUCTIONS PER CALL)"
    NEXT ITX
    STOP
15  RAN=RND(RAN)
    IF(RAN>PTP2)GOTO 40
    IF(RAN>PTP1)GOTO 30
    RAN=RND(RAN)
    IF(RAN>PILC)GOTO 25
    ICNT(I,1)=ICNT(I,1)+1
    ITIM(I)=ITIM(I)+IOP2
    GOTO 15
25  ITIM(I)=ITIM(I)+IOP1
    ITYP(I)=1
```

```
      ISTG(I)=1
      RETURN
30    RAN=RND(RAN)
      IF(RAN>PILC)GOTO 35
      ITIM(I)=ITIM(I)+IDR2
      RAN=RND(RAN)
      IF(RAN>POLC)GOTO 32
      ITIM(I)=ITIM(I)+IDR4
      ICNT(I,2)=ICNT(I,2)+1
      GOTO 15
32    ITYP(I)=2
      ISTG(I)=2
      RETURN
35    ITIM(I)=ITIM(I)+IDR1
      ITYP(I)=2
      ISTG(I)=1
      RETURN
40    IF(RAN>PTP3)GOTO 50
      RAN=RND(RAN)
      IF(RAN>PILC)GOTO 45
      ITIM(I)=ITIM(I)+IRW2
      RAN=RND(RAN)
      IF(RAN>POLC)GOTO 42
      ITIM(I)=ITIM(I)+IRW4
      ICNT(I,3)=ICNT(I,3)+1
      GOTO 15
42    ITYP(I)=3
      ISTG(I)=2
```

```
      RETURN
45    ITIM(I)=ITIM(I)+IRW1
      ITYP(I)=3
      ISTG(I)=1
      RETURN
50    RAN=RND(RAN)
      ITYP(I)=4
      IF(RAN>PILC)GOTO 55
      ITIM(I)=ITIM(I)+IMV2
      ISTG(I)=2
      RETURN
55    ITIM(I)=ITIM(I)+IMV1
      ISTG(I)=1
99    RETURN
```

Appendix B - Some Simulation Results

INTERFERENCE SIMULATION STATISTICS FOR N PROCESSORS ON CSX BUS

4 TYPES OF INSTRUCTIONS

PROB(1)= 0.515 PROB(2)= 0.380 PROB(3)= 0.100 PROB(4)= 0.005

PRIMITIVE CYCLE TIME = 300 NSEC

BUS CYCLE TIME = 300 NSEC

PROB. OF INSTRUCTION FETCH FROM LOCAL MEMORY = 0.25

PROB. OF OPERAND FETCH FROM LOCAL MEMORY = 0.25

AVERAGE MAXIMUM NUMBER OF INSTRUCTIONS PER PROCESSOR = 613.4

MAXIMUM INSTRUCTION THROUGHPUT = 11074.20 (2214840 /SEC.)

CUT-OFF POINT OCCURS AT 18.05 PROCESSORS

1 PROCESSORS *** RANDOM NUMBER GENERATOR = 0.985611

PROC #	INST 1	INST 2	INST 3	INST 4	TOTAL	TIME	EFF. TIME	% LOSS
AVE.	316.0	225.0	75.0	2.0	618.0	100122	100122	0.000

TOT. BUS REQ. = 859 TOT. INST. = 618 MAX. INST. = 613

THROUGHPUT FACTOR = 100.00 INST. RATE = 123449 /SEC.

I. E. 55552 CALLS PER HOUR (8000 INSTRUCTIONS PER CALL)

2 PROCESSORS *** RANDOM NUMBER GENERATOR = 0.468831

PROC #	INST 1	INST 2	INST 3	INST 4	TOTAL	TIME	EFF. TIME	% LOSS
AVE.	311.0	228.5	68.0	3.5	611.0	100176	100029	0.147

TOT. BUS REQ. = 1933 TOT. INST. = 1222 MAX. INST. = 1227

THROUGHPUT FACTOR = 99.85 INST. RATE = 243971 /SEC.

I. E. 109787 CALLS PER HOUR (8000 INSTRUCTIONS PER CALL)

```
 4 PROCESSORS *** RANDOM NUMBER GENERATOR =   0.657360
 PROC # INST 1 INST 2 INST 3 INST 4  TOTAL   TIME  EFF. TIME  % LOSS
  AVE.  321.0  219.8   59.8    4.5  605.0  100200     99650    0.549
TOT. BUS REQ. =   3982  TOT. INST. =   2420  MAX. INST. =   2454
THROUGHPUT FACTOR =     99.45  INST. RATE =   483034 /SEC.
I. E.   217365  CALLS PER HOUR  (  8000 INSTRUCTIONS PER CALL)

 8 PROCESSORS *** RANDOM NUMBER GENERATOR =   0.346034
 PROC # INST 1 INST 2 INST 3 INST 4  TOTAL   TIME  EFF. TIME  % LOSS
  AVE.  320.6  224.5   59.0    2.6  606.8  100030     98637    1.392
TOT. BUS REQ. =   7033  TOT. INST. =   4854  MAX. INST. =   4907
THROUGHPUT FACTOR =     98.61  INST. RATE =   970512 /SEC.
I. E.   436730  CALLS PER HOUR  (  8000 INSTRUCTIONS PER CALL)

12 PROCESSORS *** RANDOM NUMBER GENERATOR =   0.424363
 PROC # INST 1 INST 2 INST 3 INST 4  TOTAL   TIME  EFF. TIME  % LOSS
  AVE.  307.5  221.5   61.7    3.0  593.7  100069     96923    3.144
TOT. BUS REQ. =  10711  TOT. INST. =   7124  MAX. INST. =   7361
THROUGHPUT FACTOR =     96.86  INST. RATE =  1423818 /SEC.
I. E.   640718  CALLS PER HOUR  (  8000 INSTRUCTIONS PER CALL)

16 PROCESSORS *** RANDOM NUMBER GENERATOR =   0.876103
 PROC # INST 1 INST 2 INST 3 INST 4  TOTAL   TIME  EFF. TIME  % LOSS
  AVE.  298.8  221.3   54.1    2.8  576.9  100074     93975    6.094
```

TOT. BUS REQ. = 13626 TOT. INST. = 9230 MAX. INST. = 9815
THROUGHPUT FACTOR = 93.91 INST. RATE = 1844635 /SEC.
I. E. 830086 CALLS PER HOUR (8000 INSTRUCTIONS PER CALL)

20 PROCESSORS *** RANDOM NUMBER GENERATOR = 0.529792

PROC #	INST 1	INST 2	INST 3	INST 4	TOTAL	TIME	EFF. TIME	% LOSS
AVE.	272.0	201.3	51.5	2.3	527.0	100052	85790	14.254

TOT. BUS REQ. = 15597 TOT. INST. = 10540 MAX. INST. = 12268
THROUGHPUT FACTOR = 85.75 INST. RATE = 2106909 /SEC.
I. E. 948109 CALLS PER HOUR (8000 INSTRUCTIONS PER CALL)

24 PROCESSORS *** RANDOM NUMBER GENERATOR = 0.304215

PROC #	INST 1	INST 2	INST 3	INST 4	TOTAL	TIME	EFF. TIME	% LOSS
AVE.	234.0	173.9	45.1	2.3	455.4	100063	74396	25.652

TOT. BUS REQ. = 16534 TOT. INST. = 10929 MAX. INST. = 14722
THROUGHPUT FACTOR = 74.35 INST. RATE = 2184424 /SEC.
I. E. 982990 CALLS PER HOUR (8000 INSTRUCTIONS PER CALL)

32 PROCESSORS *** RANDOM NUMBER GENERATOR = 0.275238

PROC #	INST 1	INST 2	INST 3	INST 4	TOTAL	TIME	EFF. TIME	% LOSS
AVE.	184.6	134.9	36.1	1.4	357.0	100051	58067	41.963

TOT. BUS REQ. = 16659 TOT. INST. = 11425 MAX. INST. = 19629
THROUGHPUT FACTOR = 58.04 INST. RATE = 2283837 /SEC.
I. E. 1027727 CALLS PER HOUR (8000 INSTRUCTIONS PER CALL)

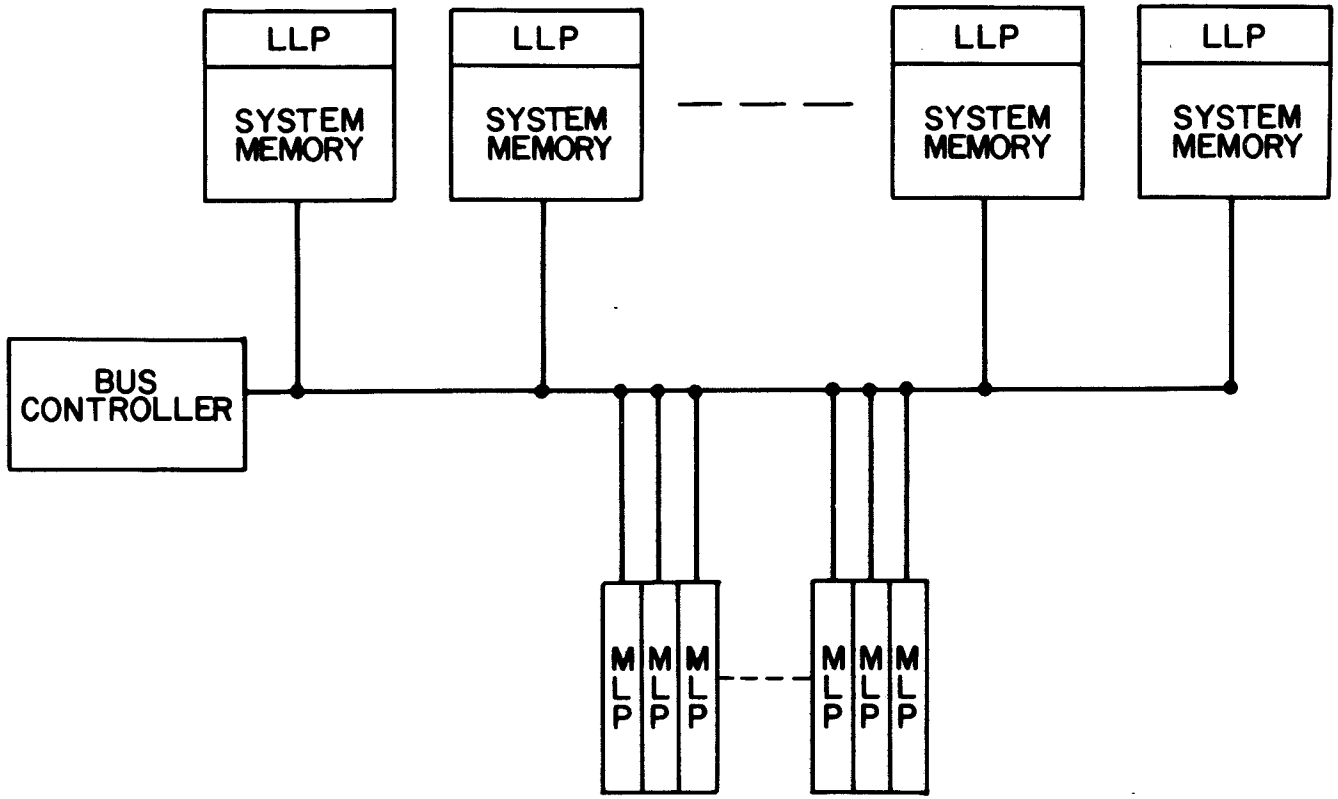
```
 40 PROCESSORS *** RANDOM NUMBER GENERATOR =   0.261342

 PROC #  INST 1  INST 2  INST 3  INST 4   TOTAL    TIME   EFF. TIME   % LOSS

  AVE.   143.7   103.4    28.1     1.3   276.5   100047      45185    54.837

TOT. BUS REQ. =  16659  TOT. INST. =  11060  MAX. INST. =  24537

THROUGHPUT FACTOR =      45.16  INST. RATE =  2210955 /SEC.
```



LLP - MULTI-PLEXED LOW-LEVEL PROCESSORS
MLP - MID-LEVEL PROCESSOR WITH LOCAL MEMORY

Figure 1

SYSTEM CONFIGURATION FOR DIGITAL WIRE CENTER

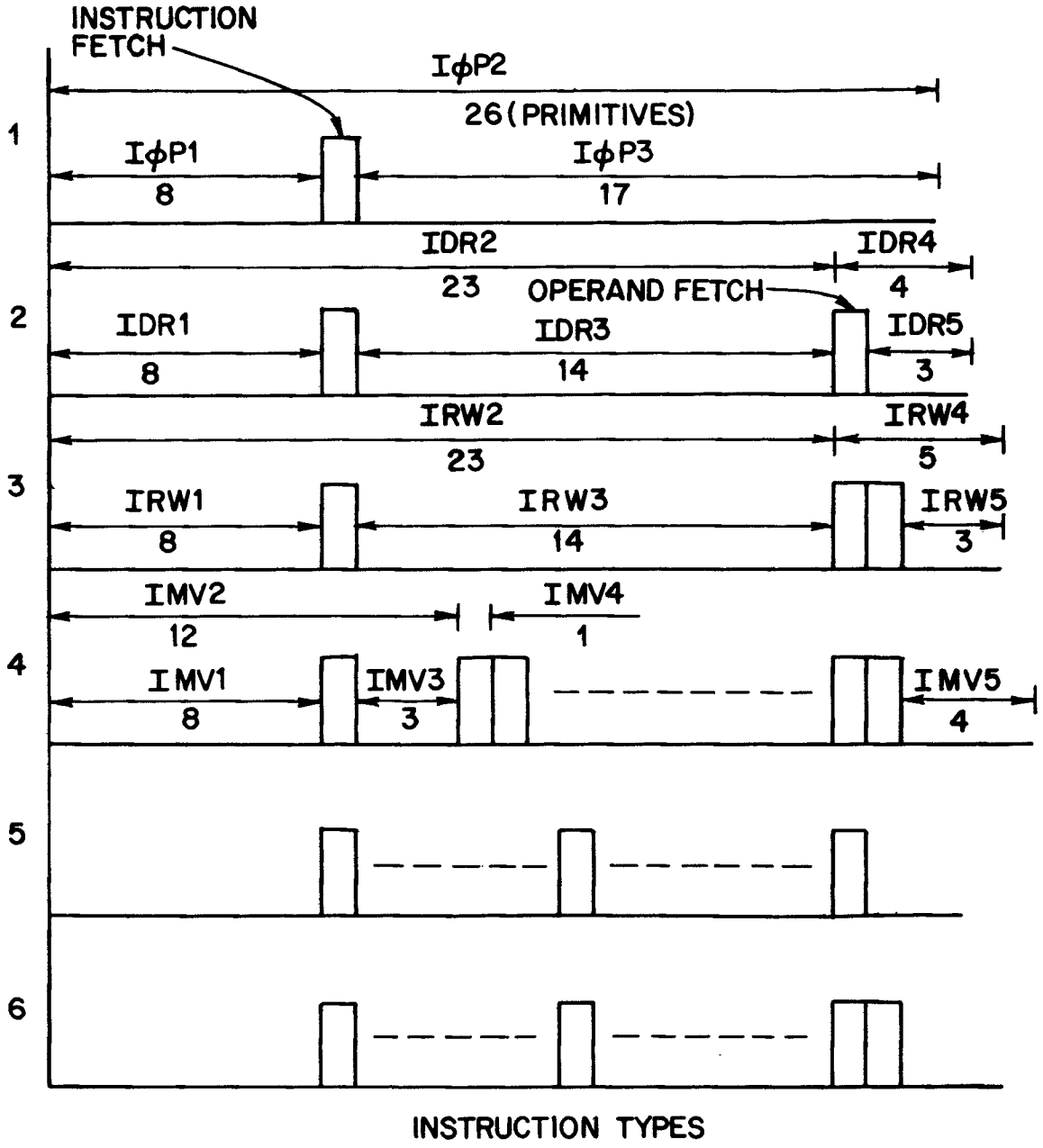

INSTRUCTION FETCH
IφP2
26(PRIMITIVES)
1
IφP1
8
IφP3
17
IDR2
23
IDR4
4
OPERAND FETCH
2
IDR1
8
IDR3
14
IDR5
3
IRW2
23
IRW4
5
3
IRW1
8
IRW3
14
IRW5
3
IMV2
12
IMV4
1
4
IMV1
8
IMV3
3
IMV5
4
5
6
INSTRUCTION TYPES

Figure 2

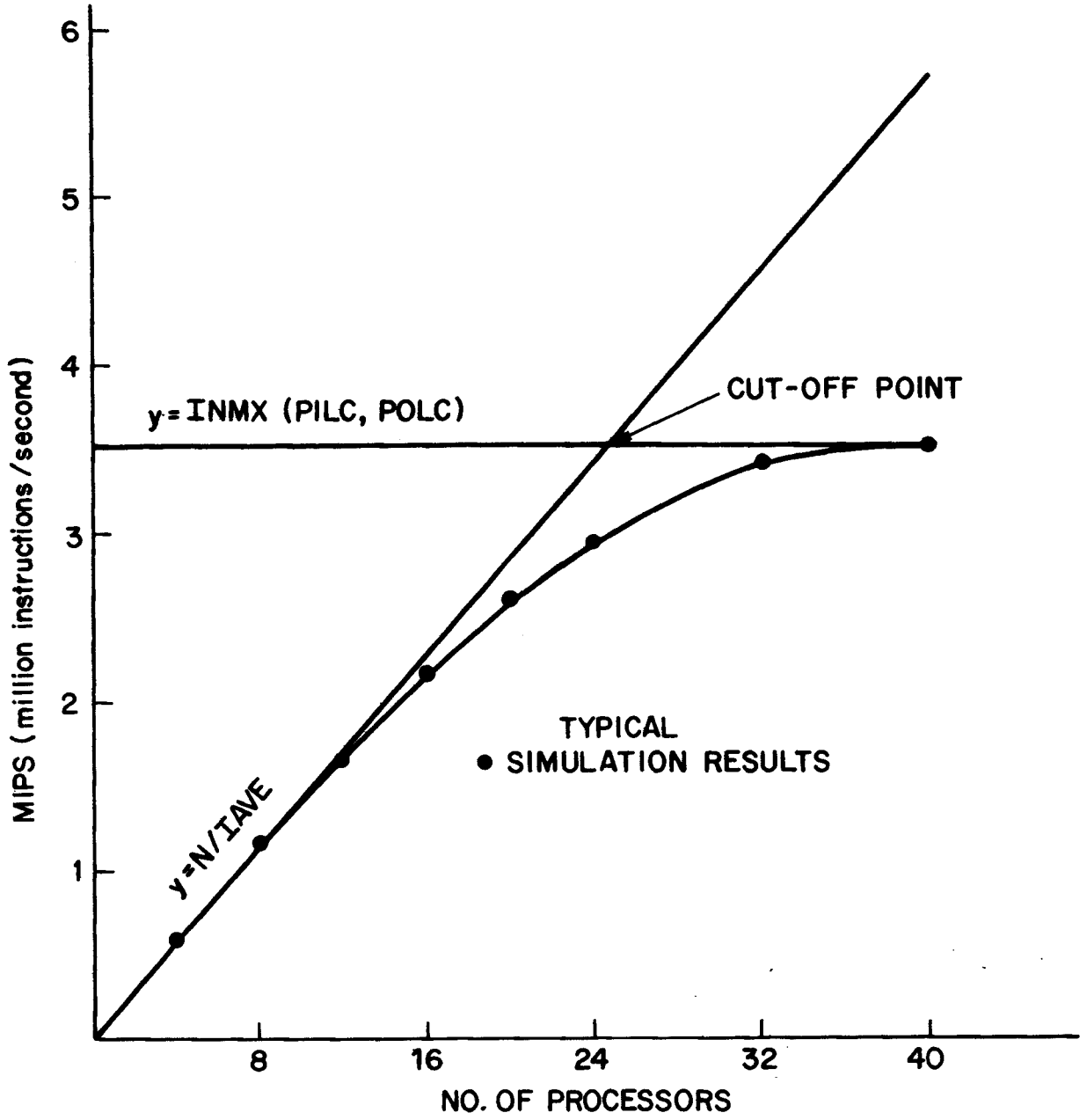

6
5
4
3
2
1
MIPS (million instructions/second)
y=INMX (PILC, POLC)
CUT-OFF POINT
y=N/IAVE
TYPICAL
● SIMULATION RESULTS
8
16
24
32
40
NO. OF PROCESSORS

Figure 3

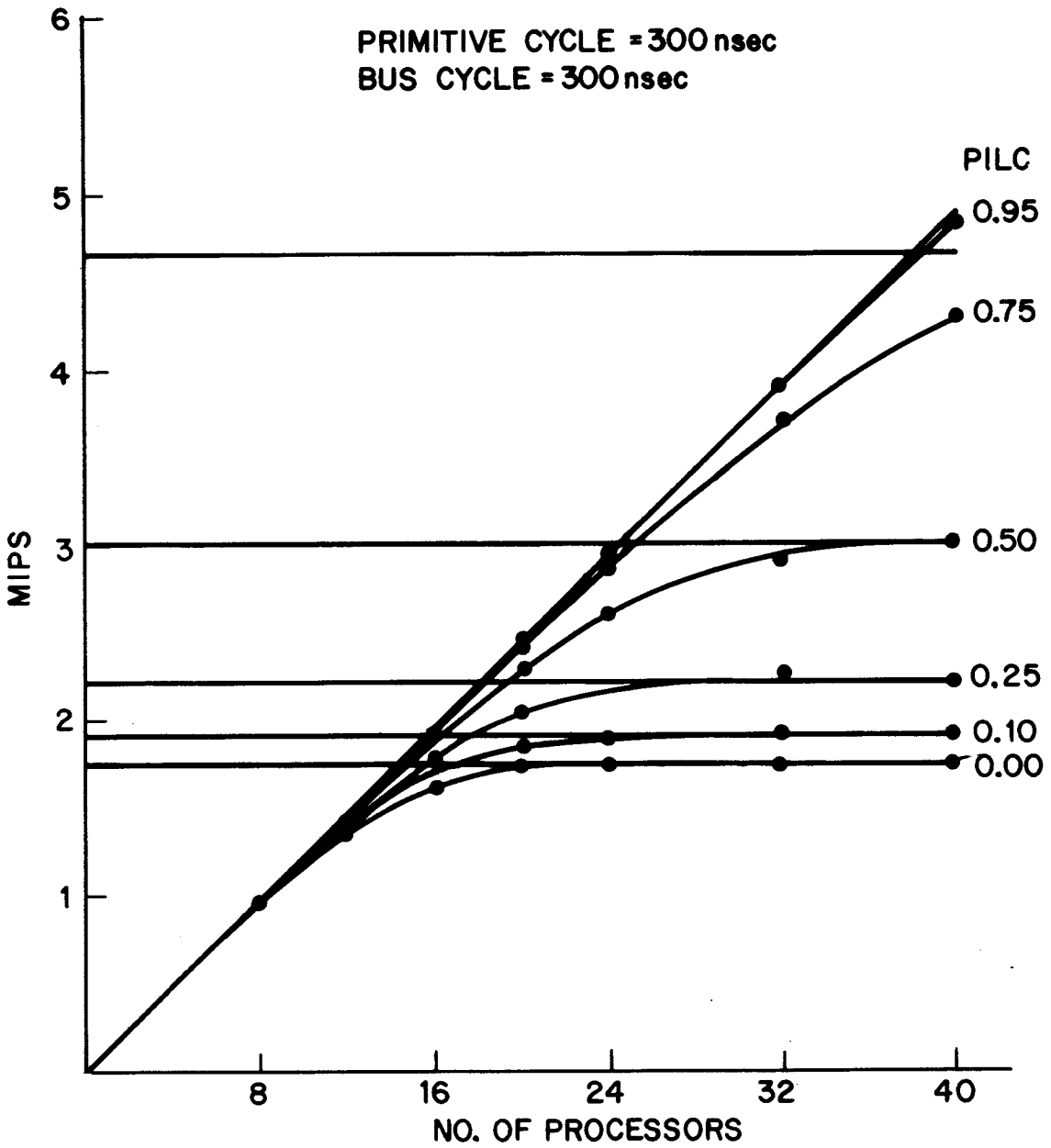

PRIMITIVE CYCLE = 300 nsec
BUS CYCLE = 300 nsec
PILC
0.95
0.75
0.50
0.25
0.10
0.00
MIPS
6
5
4
3
2
1
8
16
24
32
40
NO. OF PROCESSORS

Figure 4

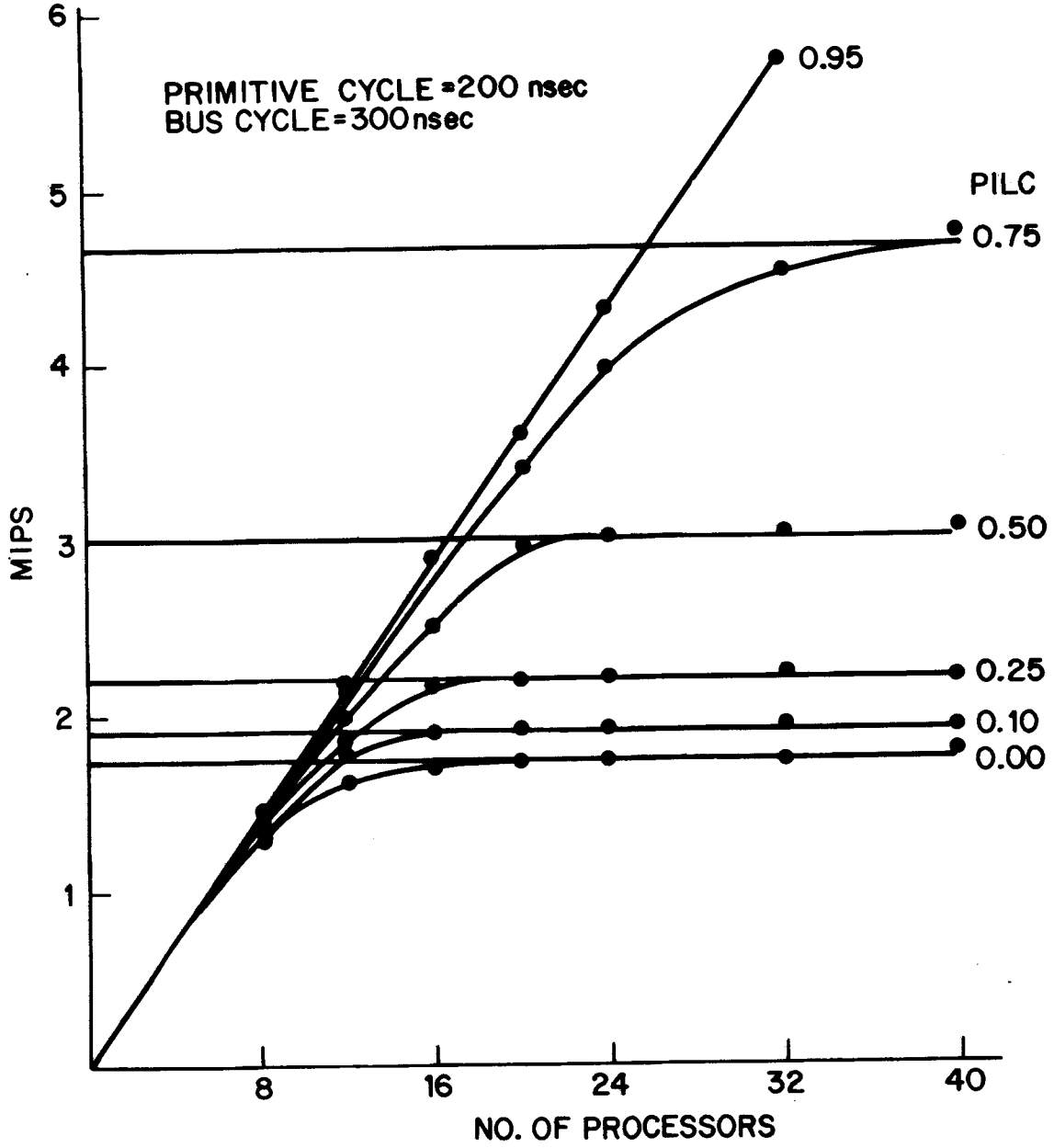

PRIMITIVE CYCLE=200 nsec
BUS CYCLE=300nsec
PILC
0.95
0.75
0.50
0.25
0.10
0.00
MIPS
NO. OF PROCESSORS
1
2
3
4
5
6
8
16
24
32
40

Figure 5

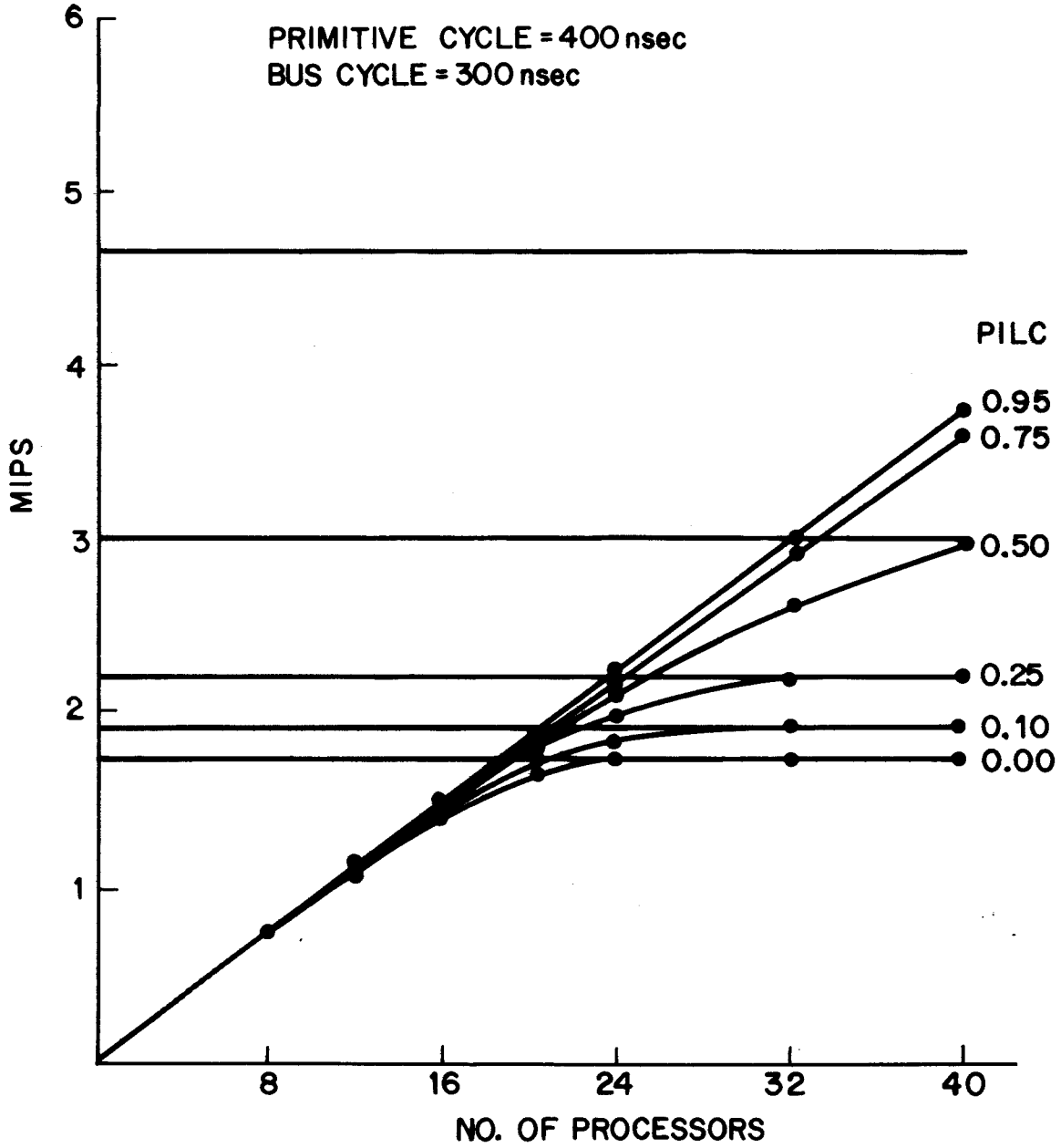

PRIMITIVE CYCLE = 400 nsec
BUS CYCLE = 300 nsec
MIPS
NO. OF PROCESSORS
PILC
0.95
0.75
0.50
0.25
0.10
0.00
1
2
3
4
5
6
8
16
24
32
40

Figure 6

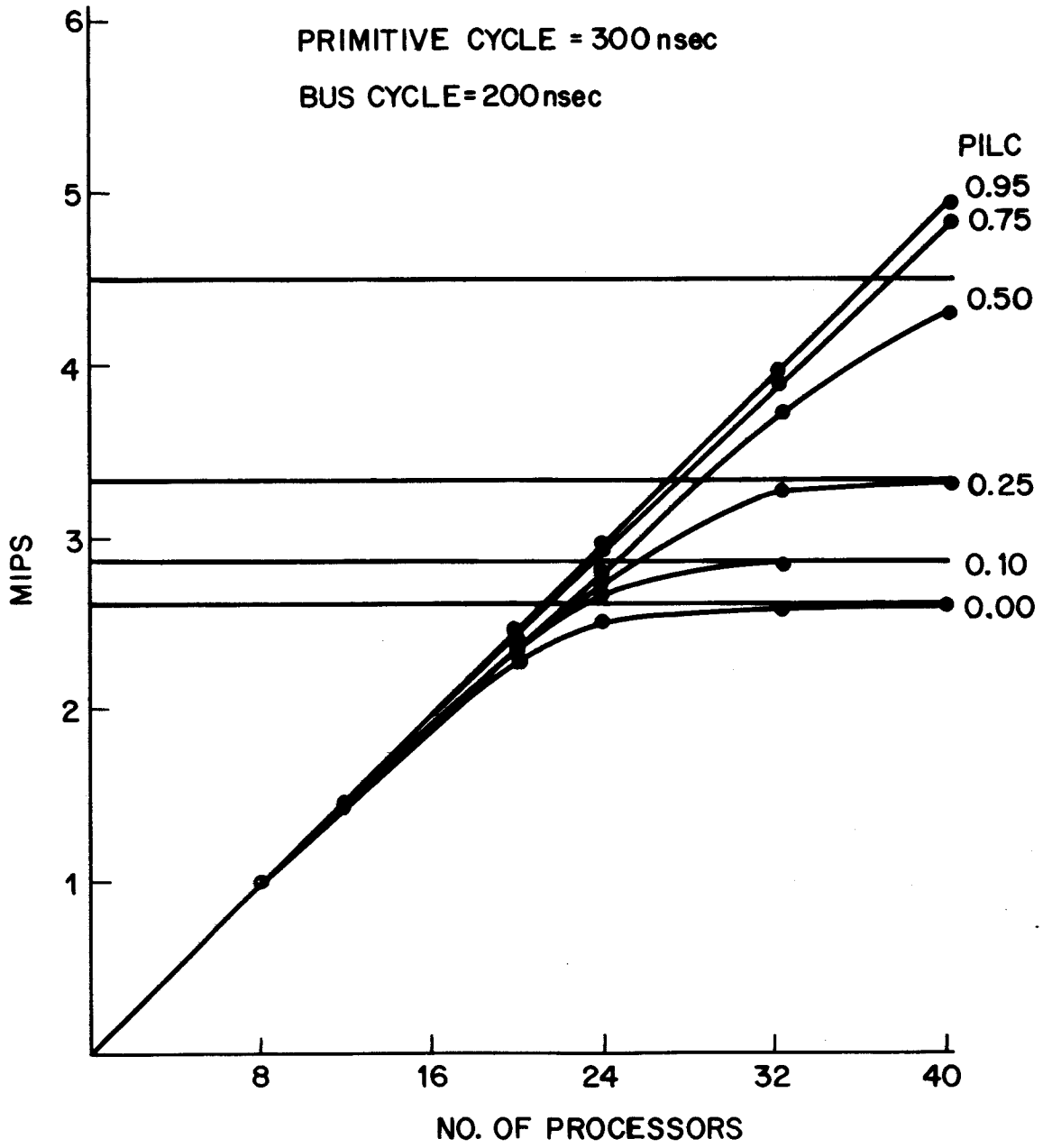

PRIMITIVE CYCLE = 300 nsec
BUS CYCLE = 200 nsec
PILC
0.95
0.75
0.50
0.25
0.10
0.00
MIPS
1
2
3
4
5
6
8
16
24
32
40
NO. OF PROCESSORS

Figure 7

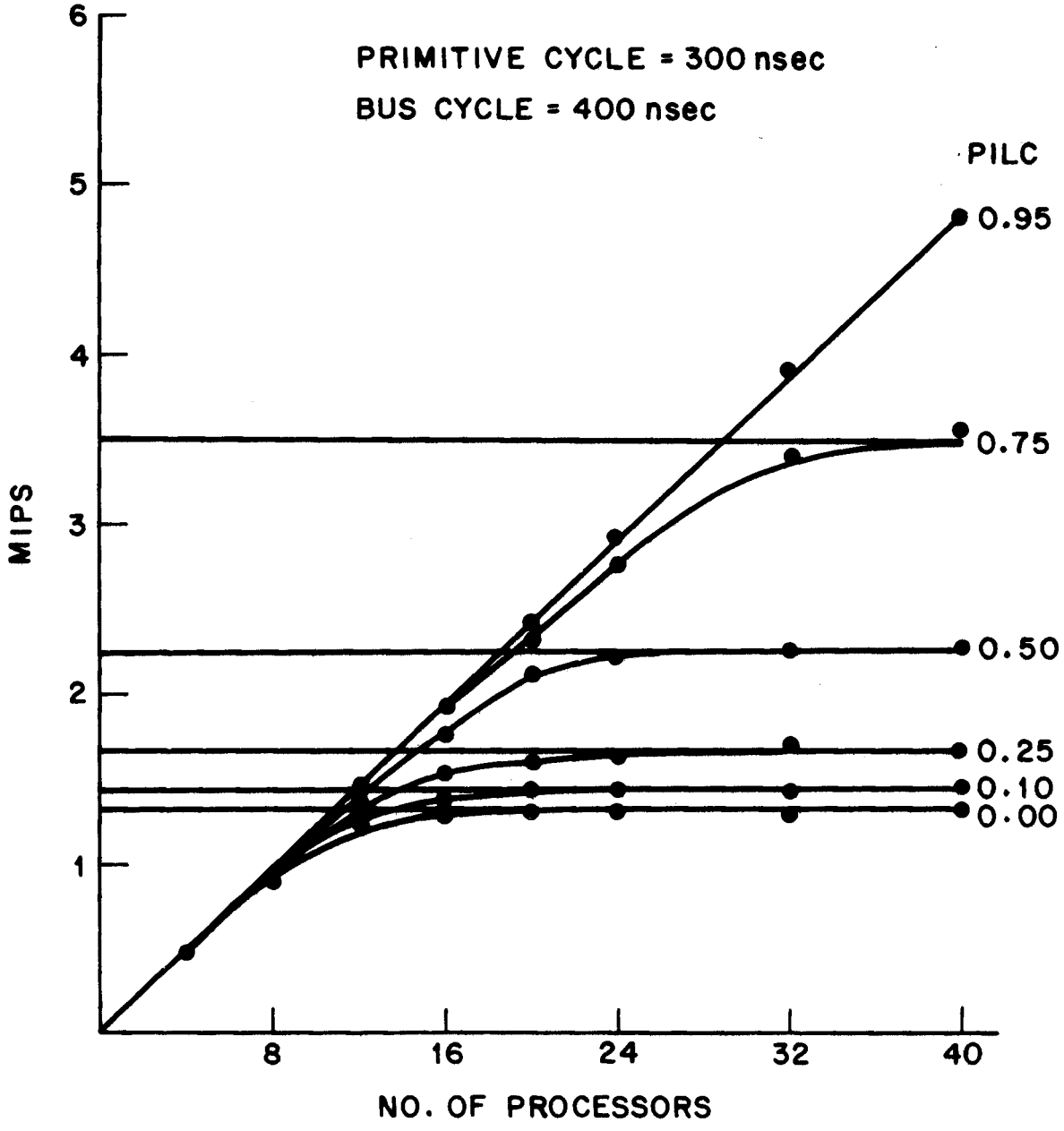

PRIMITIVE CYCLE = 300 nsec
BUS CYCLE = 400 nsec
PILC
0.95
0.75
0.50
0.25
0.10
0.00
MIPS
6
5
4
3
2
1
8
16
24
32
40
NO. OF PROCESSORS

Figure 8

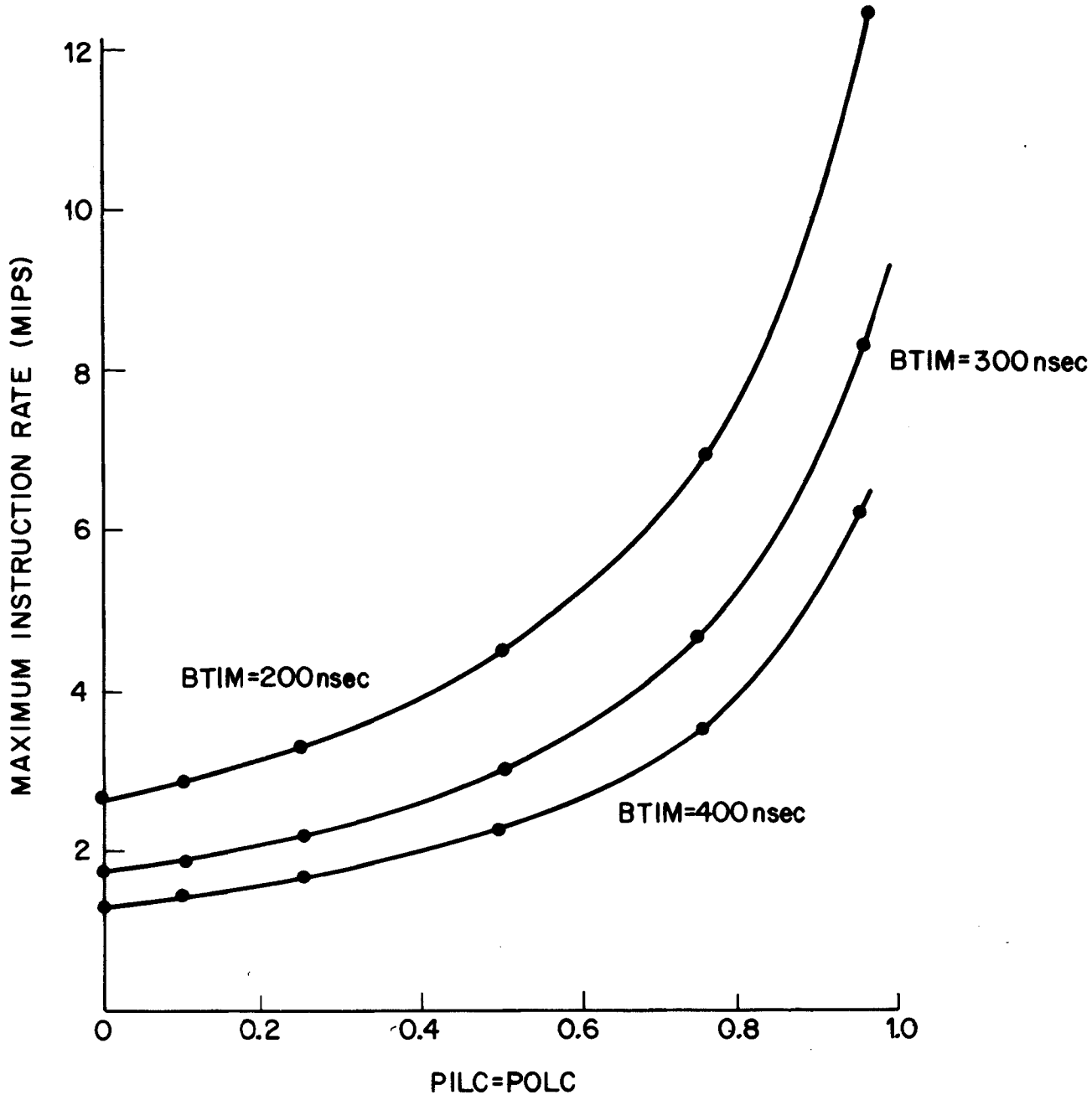

MAXIMUM INSTRUCTION RATE (MIPS)
12
10
8
6
4
2
0
0.2
0.4
0.6
0.8
1.0
PILC=POLC
BTIM=200nsec
BTIM=300nsec
BTIM=400nsec

Figure 9

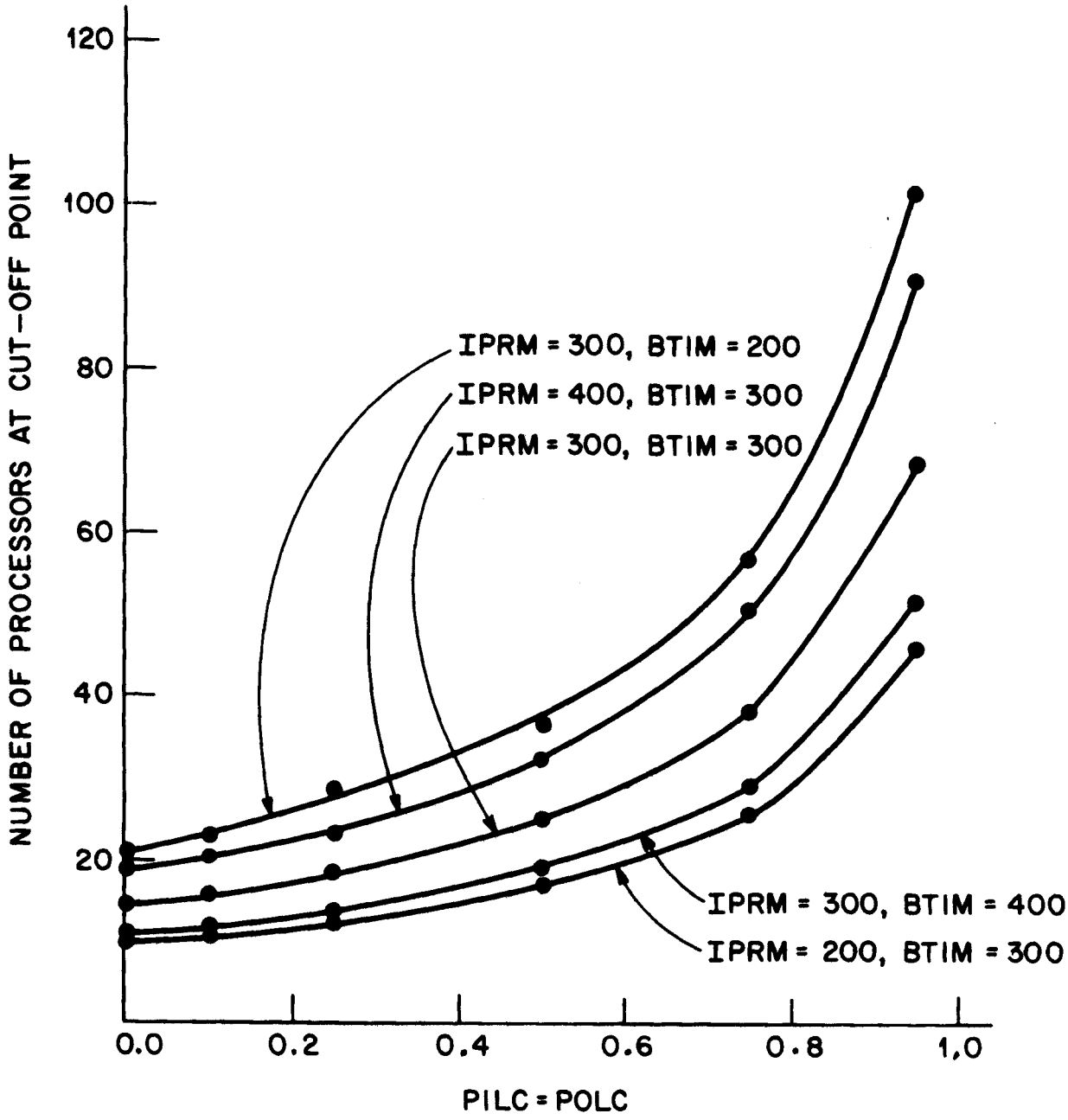

NUMBER OF PROCESSORS AT CUT-OFF POINT
120
100
80
60
40
20
0.0
0.2
0.4
0.6
0.8
1.0
PILC = POLC
IPRM = 300, BTIM = 200
IPRM = 400, BTIM = 300
IPRM = 300, BTIM = 300
IPRM = 300, BTIM = 400
IPRM = 200, BTIM = 300

Figure 10

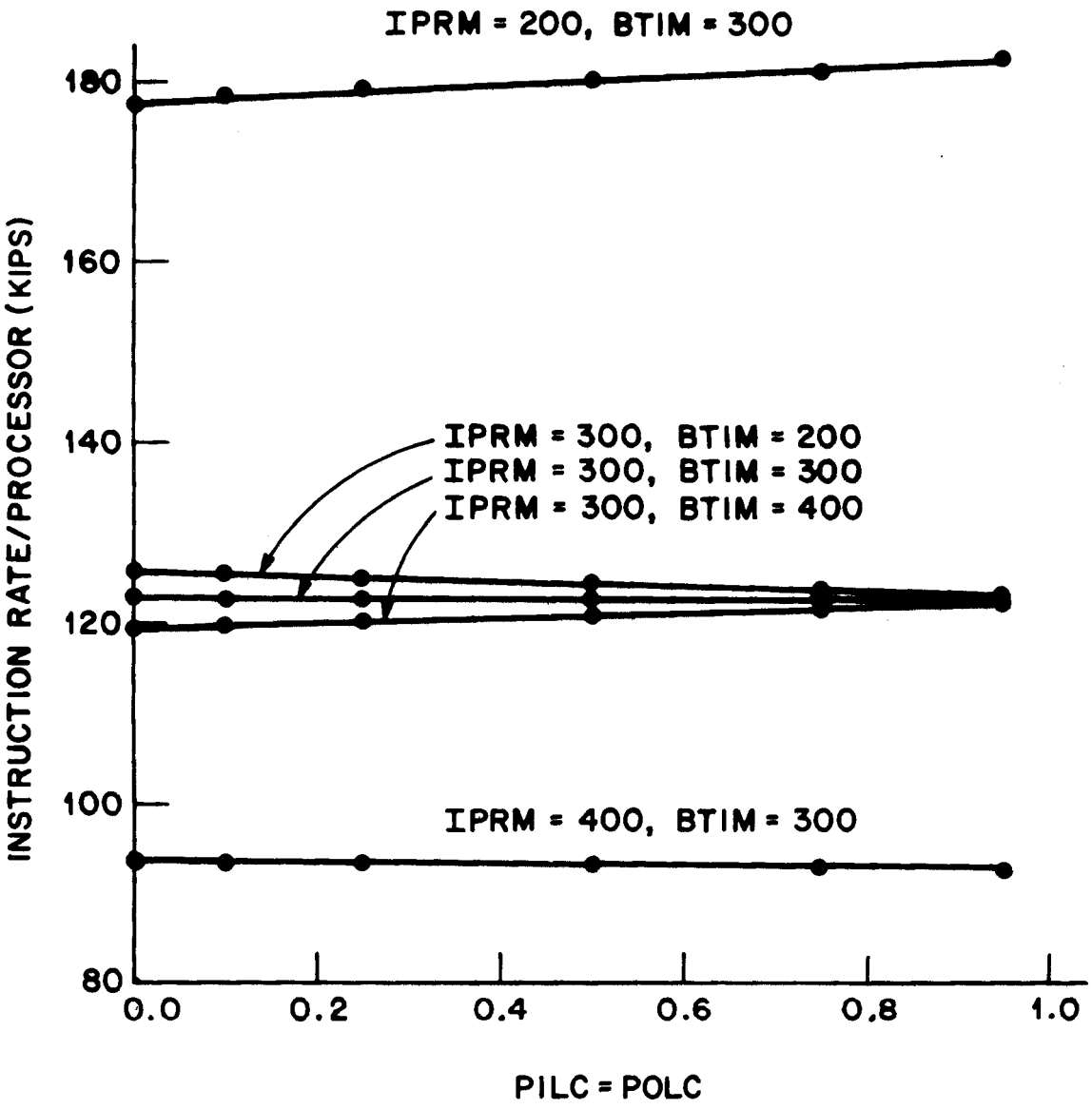

IPRM = 200, BTIM = 300
IPRM = 300, BTIM = 200
IPRM = 300, BTIM = 300
IPRM = 300, BTIM = 400
IPRM = 400, BTIM = 300
INSTRUCTION RATE/PROCESSOR (KIPS)
180
160
140
120
100
80
0.0
0.2
0.4
0.6
0.8
1.0
PILC = POLC

Figure 11

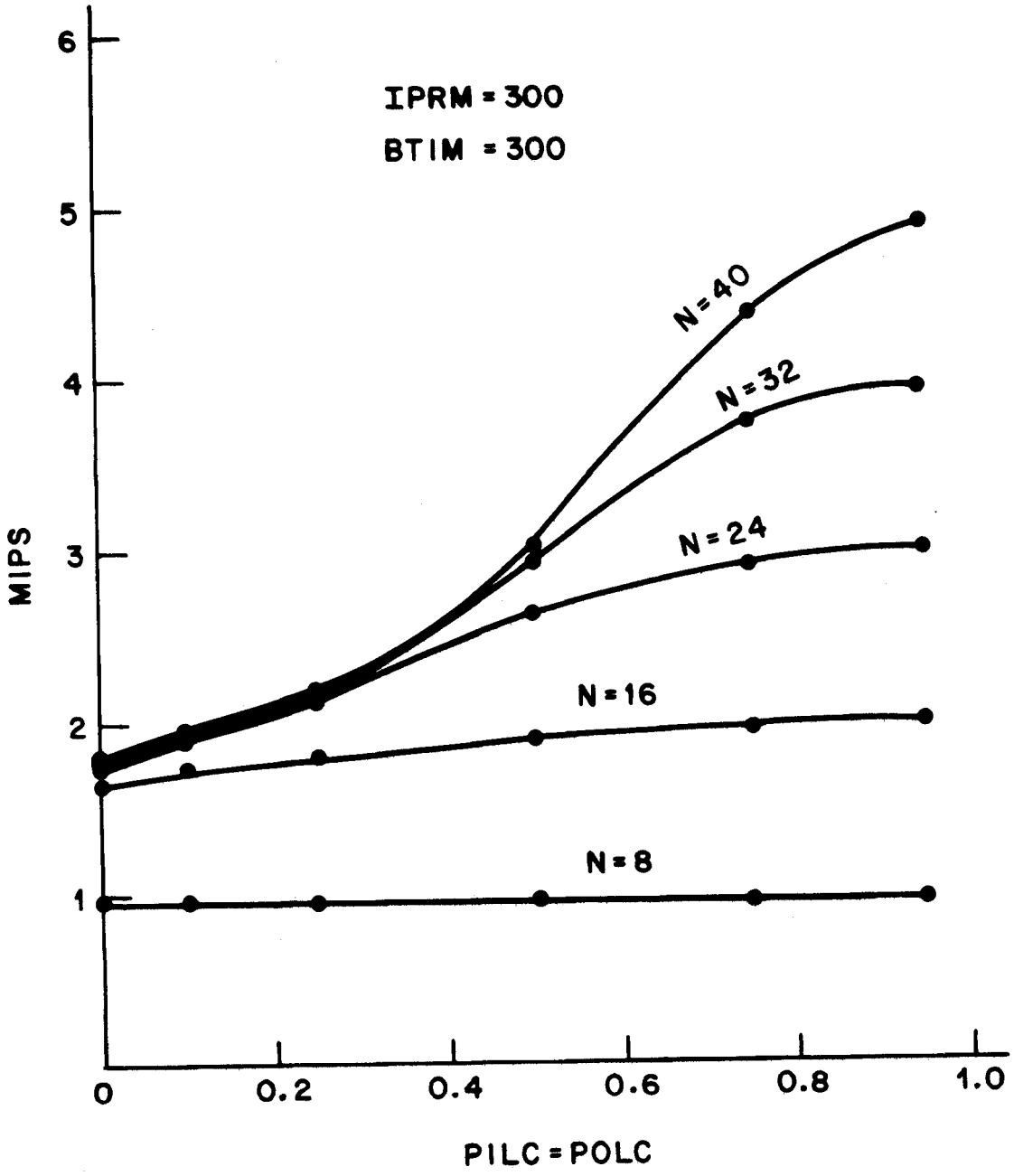

IPRM = 300
BTIM = 300
N=40
N=32
N=24
N=16
N=8
MIPS
6
5
4
3
2
1
0
0.2
0.4
0.6
0.8
1.0
PILC = POLC

Figure 12

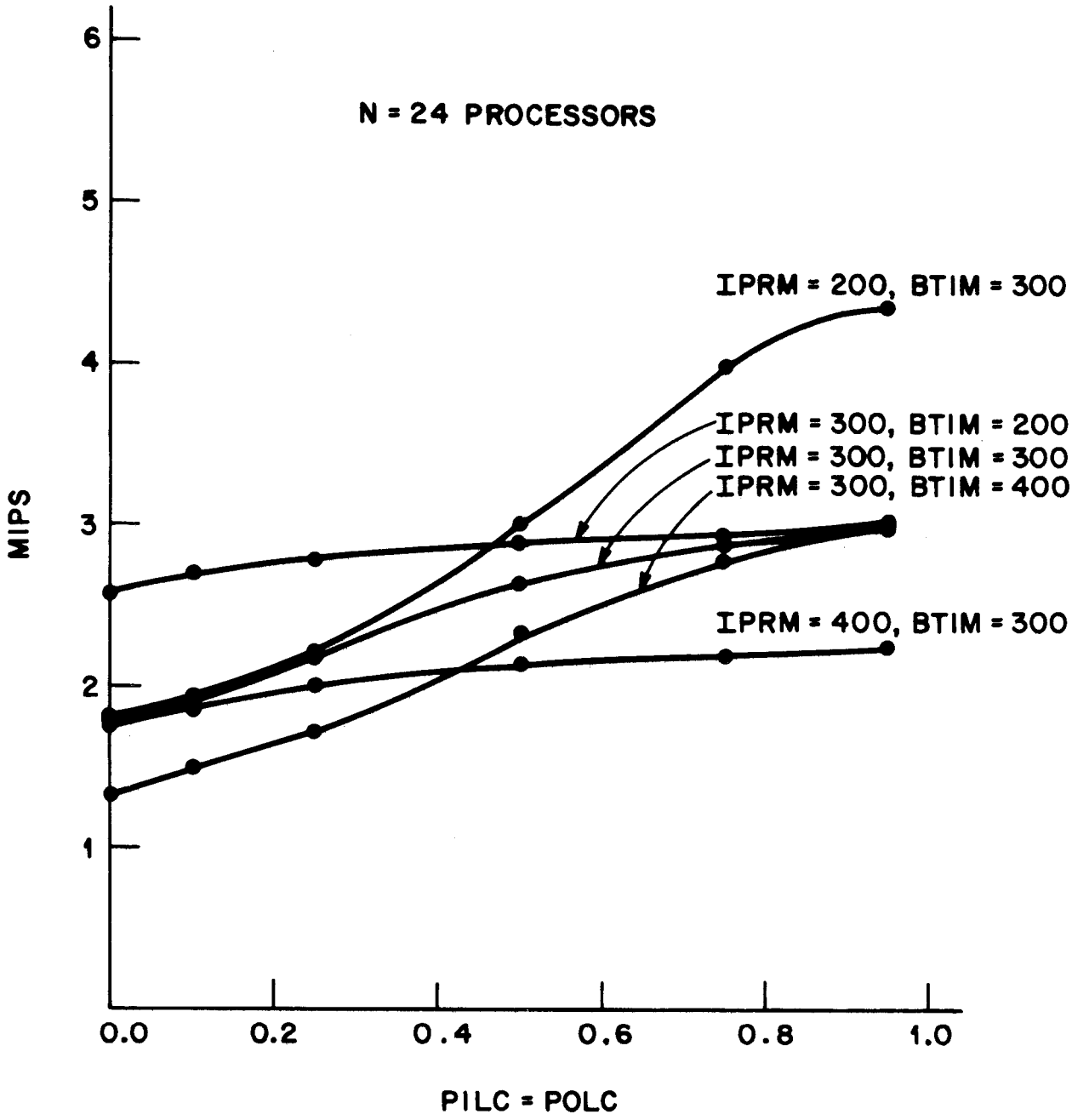

N = 24 PROCESSORS
IPRM = 200, BTIM = 300
IPRM = 300, BTIM = 200
IPRM = 300, BTIM = 300
IPRM = 300, BTIM = 400
IPRM = 400, BTIM = 300
MIPS
6
5
4
3
2
1
0.0
0.2
0.4
0.6
0.8
1.0
PILC = POLC

Figure 13

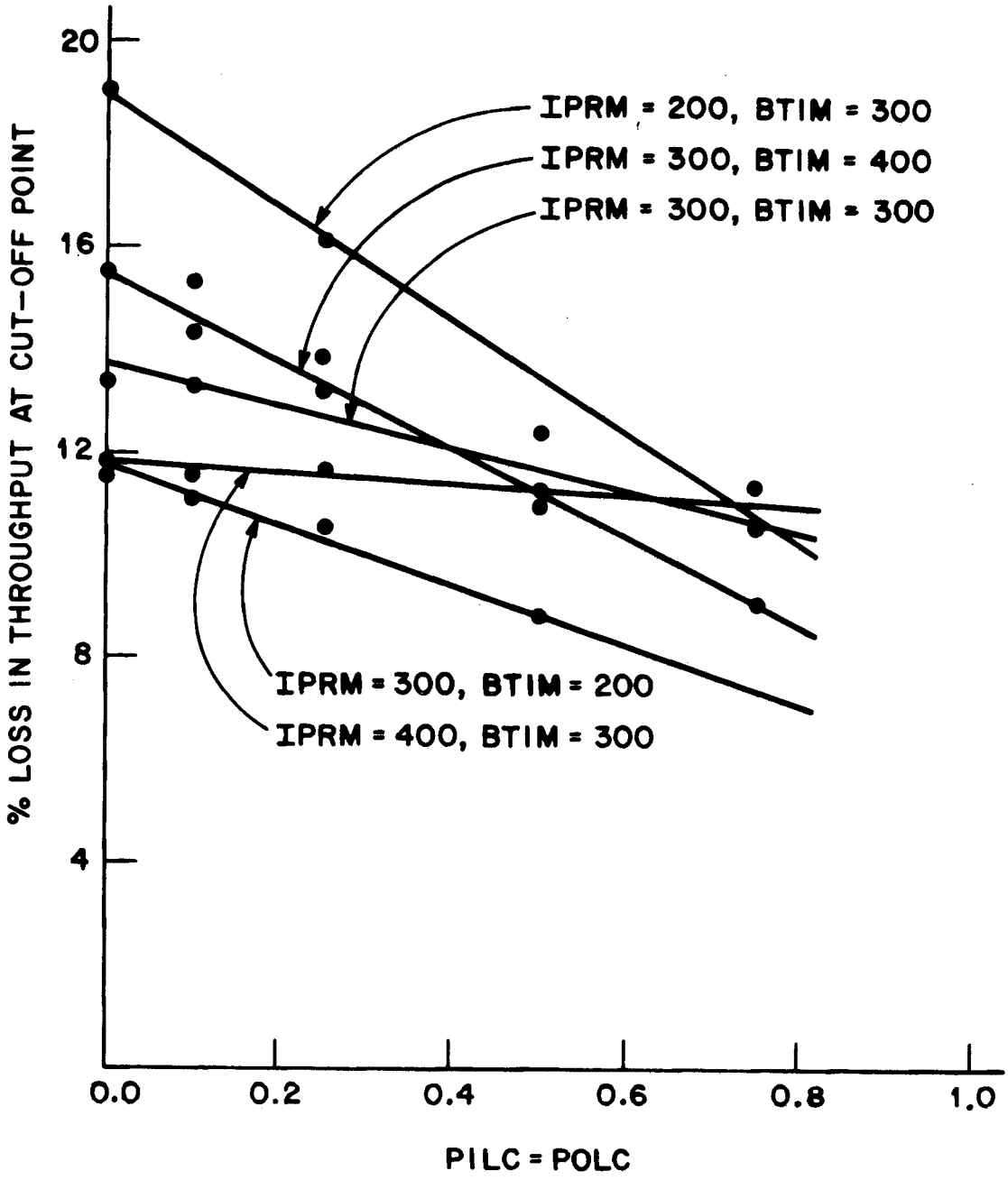

20
16
12
8
4
% LOSS IN THROUGHPUT AT CUT-OFF POINT
IPRM = 200, BTIM = 300
IPRM = 300, BTIM = 400
IPRM = 300, BTIM = 300
IPRM = 300, BTIM = 200
IPRM = 400, BTIM = 300
0.0
0.2
0.4
0.6
0.8
1.0
PILC = POLC

Figure 14

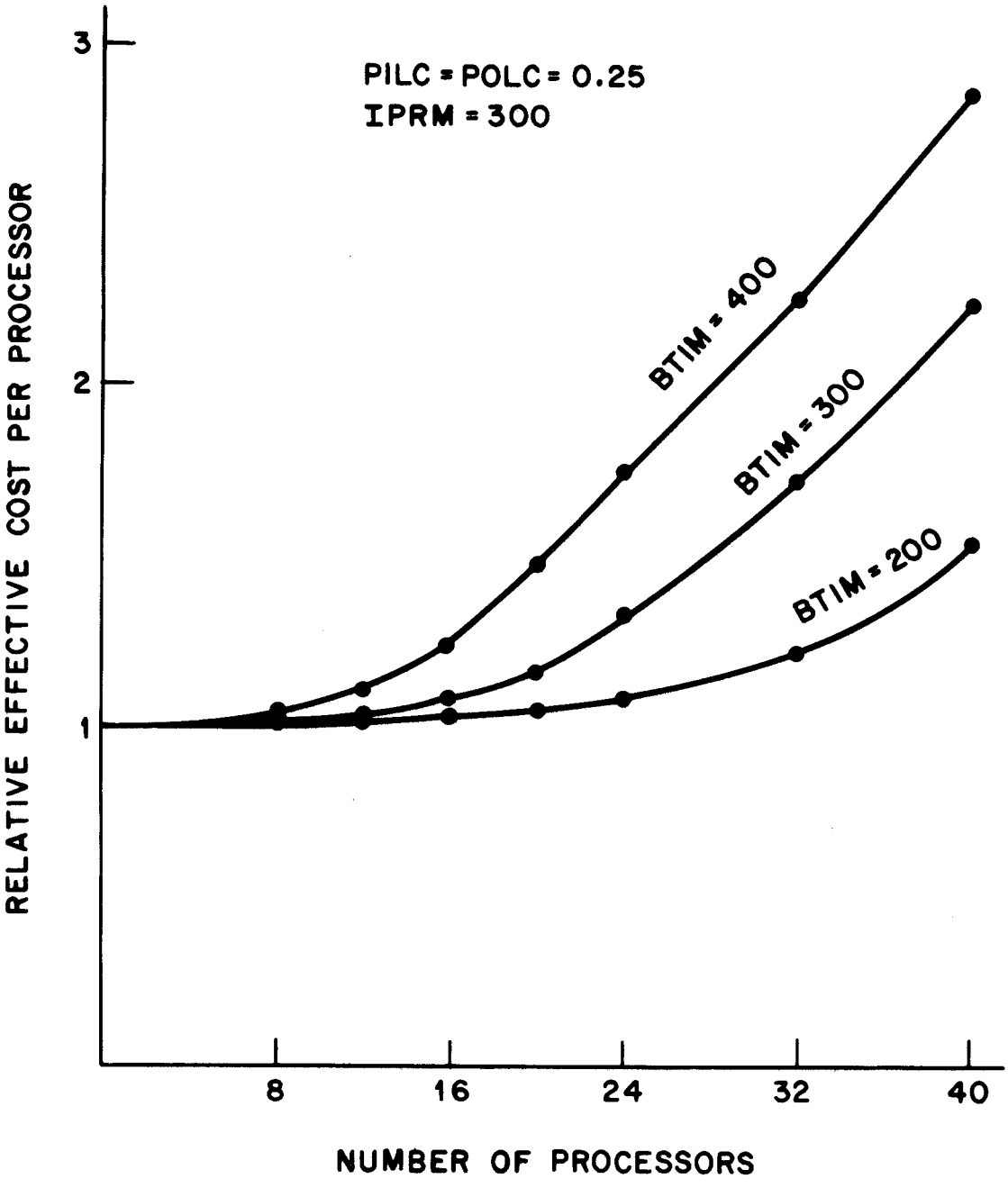

PILC = POLC = 0.25
IPRM = 300
BTIM = 400
BTIM = 300
BTIM = 200
RELATIVE EFFECTIVE COST PER PROCESSOR
NUMBER OF PROCESSORS
1
2
3
8
16
24
32
40

Figure 15

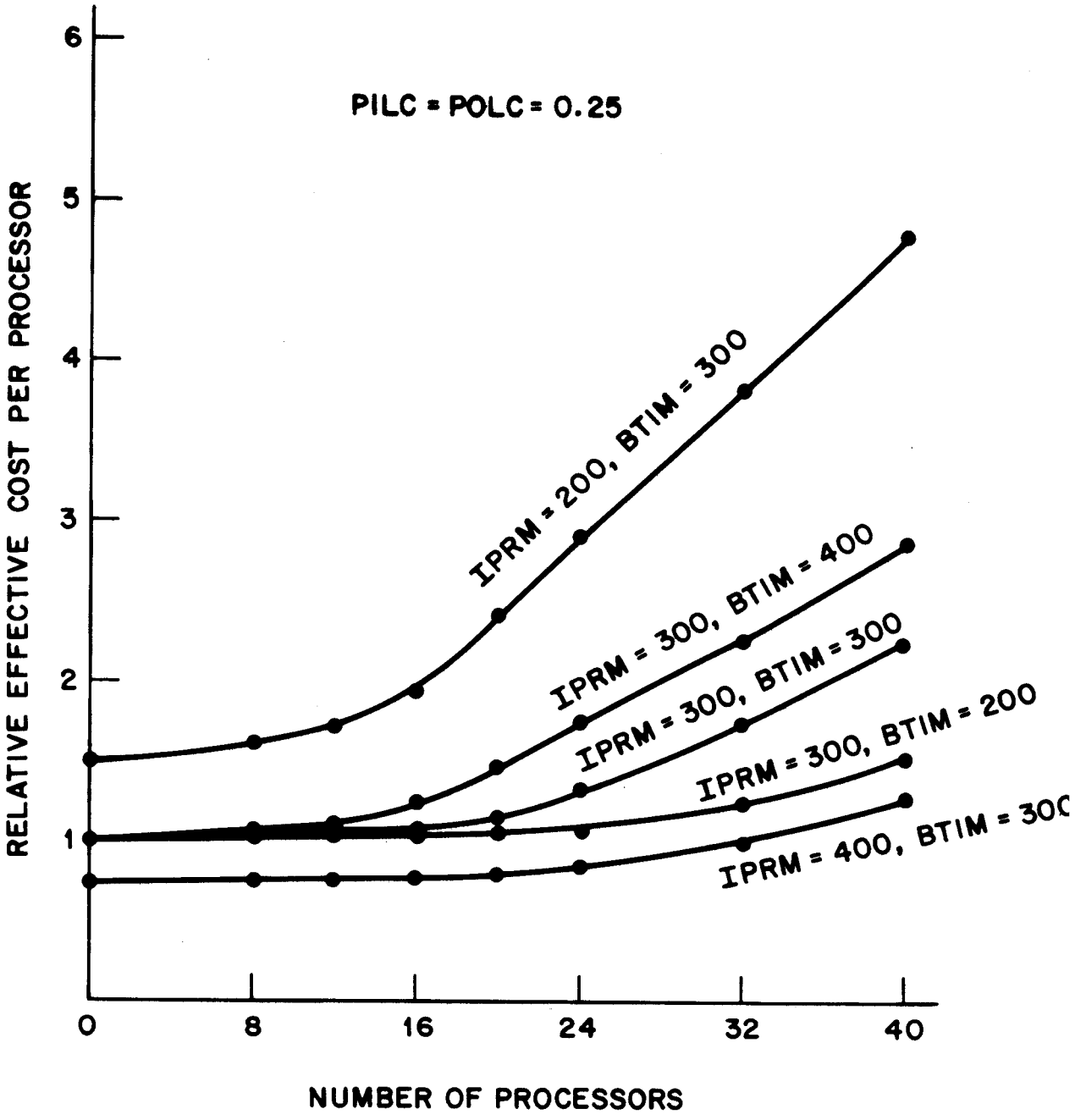

PILC = POLC = 0.25
RELATIVE EFFECTIVE COST PER PROCESSOR
NUMBER OF PROCESSORS
IPRM = 200, BTIM = 300
IPRM = 300, BTIM = 400
IPRM = 300, BTIM = 300
IPRM = 300, BTIM = 200
IPRM = 400, BTIM = 300
0
8
16
24
32
40
1
2
3
4
5
6

Figure 16

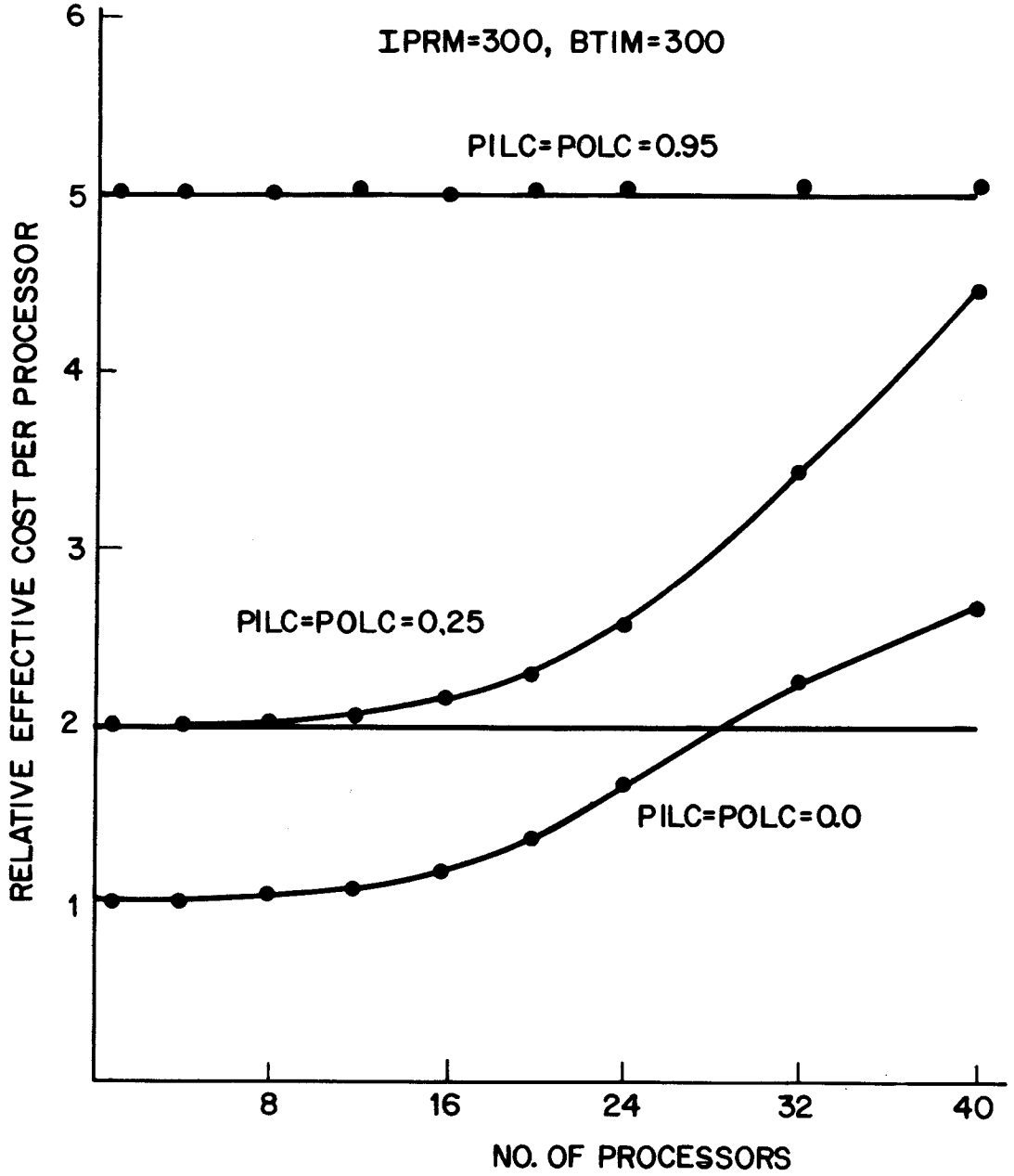

IPRM=300, BTIM=300
PILC=POLC=0.95
PILC=POLC=0.25
PILC=POLC=0.0
RELATIVE EFFECTIVE COST PER PROCESSOR
NO. OF PROCESSORS
1
2
3
4
5
6
8
16
24
32
40

Figure 17

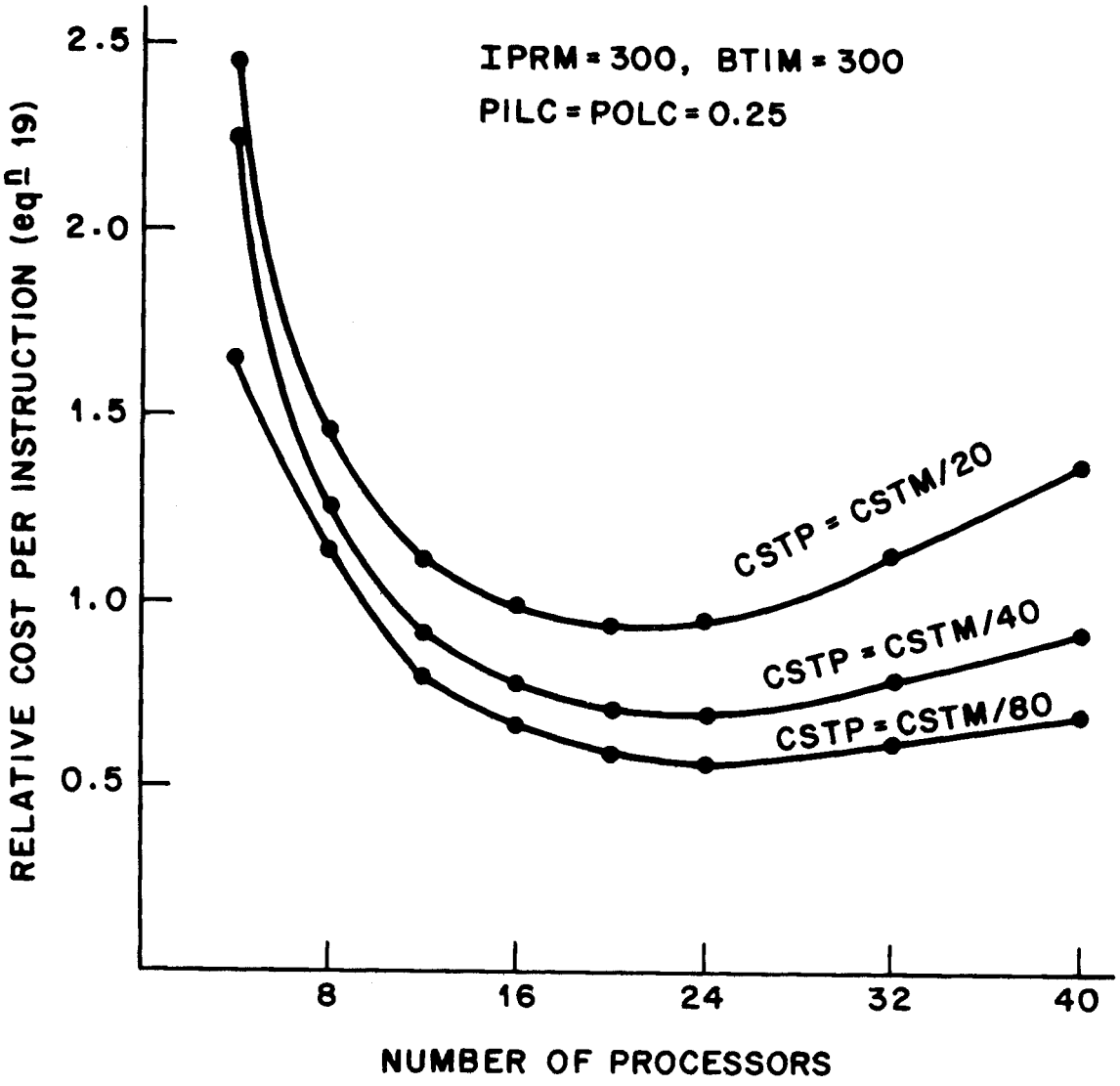

IPRM = 300, BTIM = 300
PILC = POLC = 0.25
CSTP = CSTM/20
CSTP = CSTM/40
CSTP = CSTM/80
RELATIVE COST PER INSTRUCTION (eqn 19)
2.5
2.0
1.5
1.0
0.5
8
16
24
32
40
NUMBER OF PROCESSORS

Figure 18

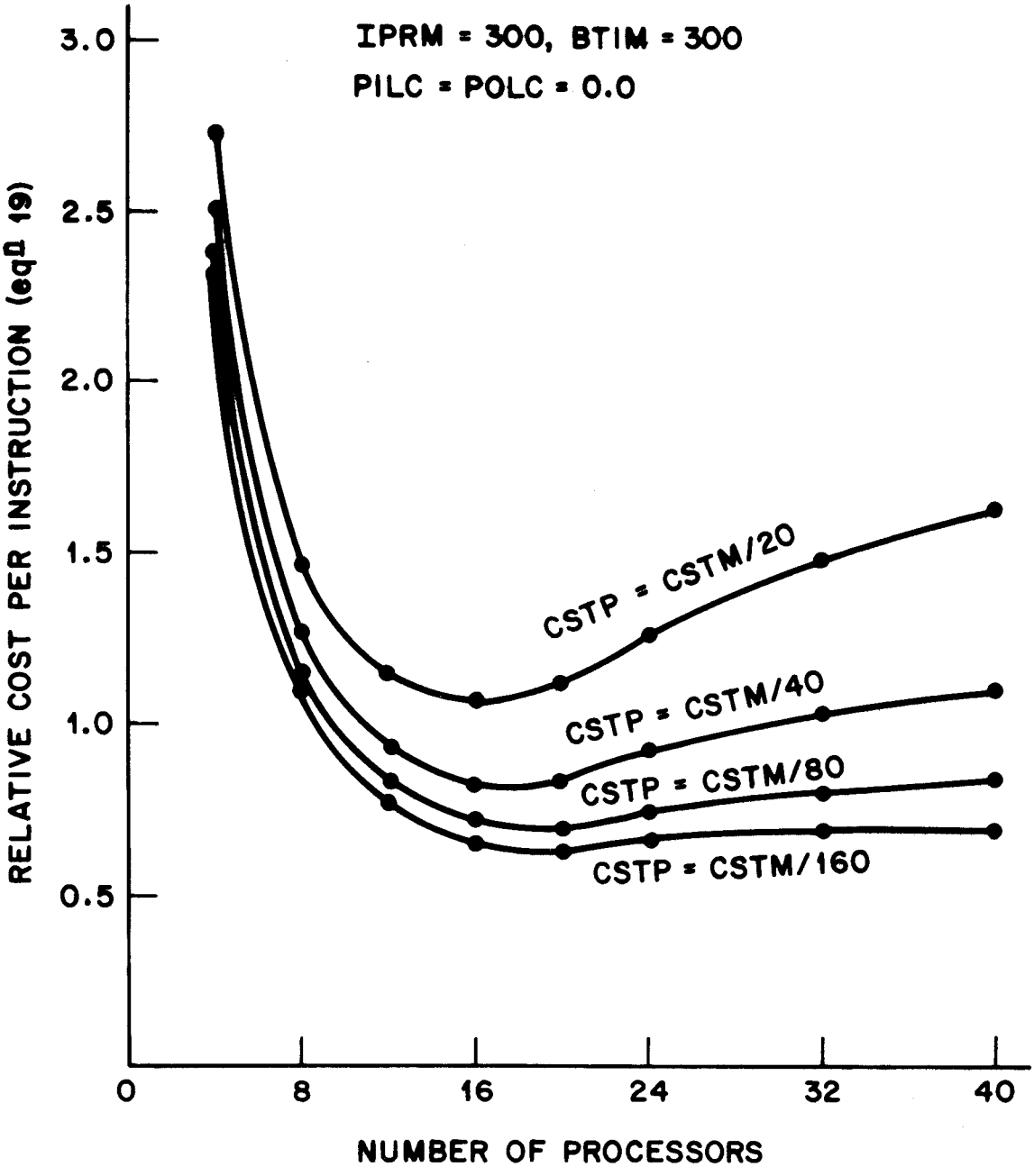

IPRM = 300, BTIM = 300
PILC = POLC = 0.0
RELATIVE COST PER INSTRUCTION (eq. 19)
3.0
2.5
2.0
1.5
1.0
0.5
0
8
16
24
32
40
NUMBER OF PROCESSORS
CSTP = CSTM/20
CSTP = CSTM/40
CSTP = CSTM/80
CSTP = CSTM/160

Figure 19

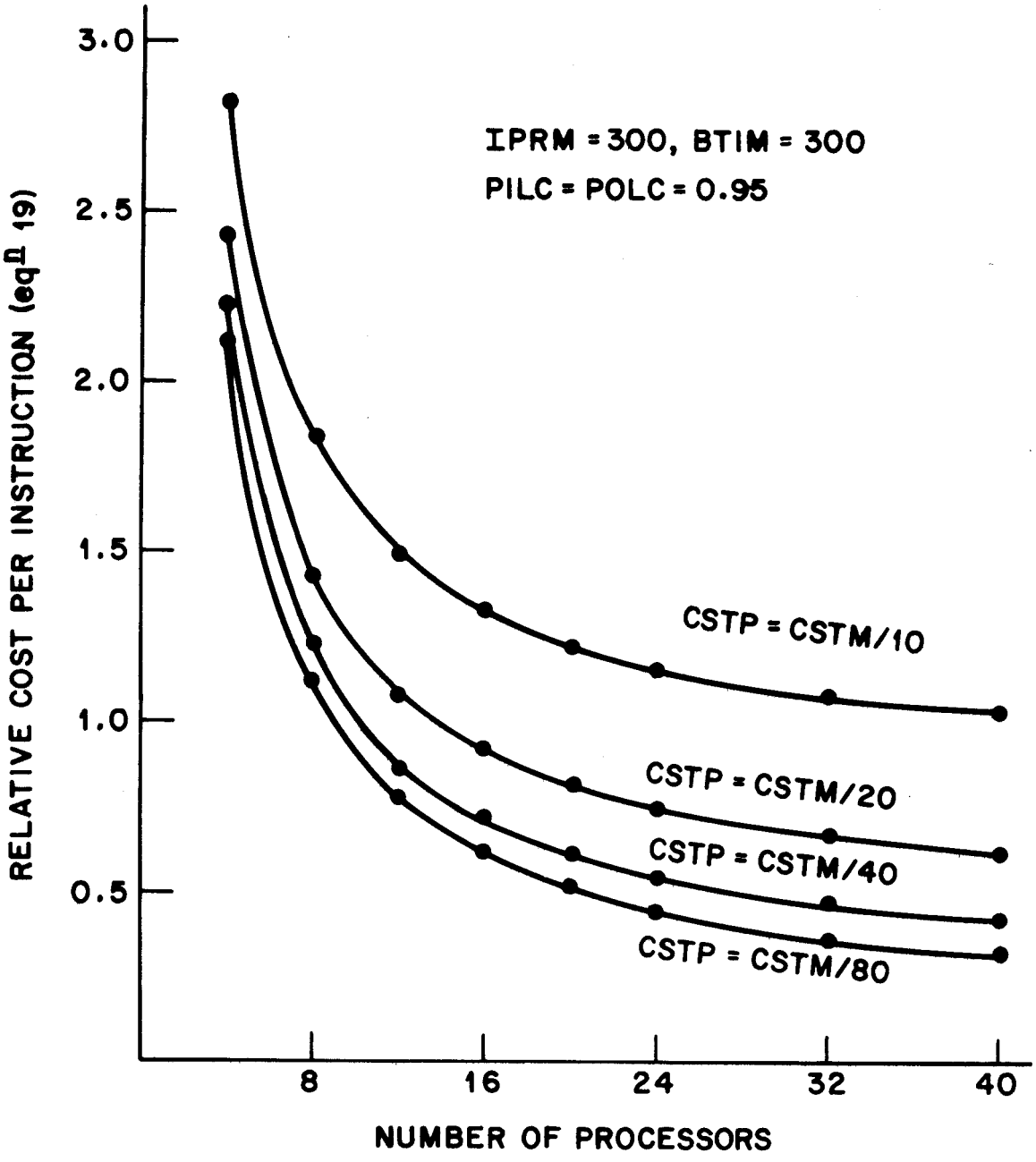

3.0
2.5
2.0
1.5
1.0
0.5
RELATIVE COST PER INSTRUCTION (eq[1] 19)
IPRM = 300, BTIM = 300
PILC = POLC = 0.95
CSTP = CSTM/10
CSTP = CSTM/20
CSTP = CSTM/40
CSTP = CSTM/80
8
16
24
32
40
NUMBER OF PROCESSORS

Figure 20