JPL PUBLICATION 77-39

Report on Phase II of the Microprocessor Seminar Held at Caltech, April 1977

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California 91103

JPL PUBLICATION 77-39

Report on Phase II of the Microprocessor Seminar Held at Caltech, April 1977

Edited By
W. Richard Scott

August 15, 1977

National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California 91103

Prepared Under Contract No NAS 7-100
National Aeronautics and Space Administration

PREFACE

The Information Systems Division of the Jet Propulsion Laboratory was responsible for arranging the Microprocessor Seminar reported herein and for preparing the Proceedings.

These Proceedings report on Phase II of the Microprocessor Seminar, which was held at the California Institute of Technology, Pasadena, California. Phase I of the seminar was held in October 1976; Phase II, in April 1977.

The two-phase seminar was organized by the Jet Propulsion Laboratory to bring together participants from JPL, NASA Centers, and other organizations concerned with the use of microprocessors and other large-scale-integrated (LSI) components in high-reliability applications. Seminar Chairman was Robert E. Covey; Deputy Chairman was W. Richard Scott.

The Proceedings of Phase I of the seminar were published by the Jet Propulsion Laboratory as JPL Publication 77-6, Report on Phase I of the Microprocessor Seminar Held at Caltech, October 1976.

ABSTRACT

This report documents Phase II of a Microprocessor Seminar held at the California Institute of Technology. Workshop sessions and papers were devoted to various aspects of microprocessor and large-scale-integrated (LSI) circuit technology. Presentations were made by LSI manufacturers on advanced LSI developments for high-reliability military and NASA applications. Microprocessor testing techniques were discussed, and test data were presented. High-reliability procurement specifications were also discussed.

SEMINAR PROGRAM

WEDNESDAY, APRIL 20

8:00 a.m. Registration

8:35 Chairman's Introduction - Robert Covey, JPL

8:40 Welcoming Remarks - Fred Felberg, JPL

8:45 Report on Microprocessor Seminar, Phase I - Paul Lecoq, JPL

9:00 Objectives of Microprocessor Seminar, Phase II -
W. Richard Scott, JPL

PRESENTATIONS ON ADVANCED LSI DEVELOPMENTS

Chairmen: Mike Ebersole, JPL
Ralph Martinez, NOSC

9:15 CMOS LSI Arrays for Systems Applications -
William Clapp, RCA/Advanced Technology Laboratories

10:30 Coffee Break

11:00 Current Status of Radiation Hardness of CMOS/SOS -
Robert Conklin, AFWL, and Alan Stanley, JPL

11:30 Radiation Hardened CMOS/SOS Developments -
Daryl Butcher, Rockwell International

12:30 Lunch

1:30 Macromodular Microcomputer Family Using CMOS/SOS 2900-Series
Hybrid LSI - Frank Langley and Steve Kaplan, Raytheon/Missile
Systems Division

2:30 Implications of High-Rel Specifications on the Intel 8080
Microprocessor - Hank Malloy, Intel Corporation

3:15 2901 Bit Slice Microprocessor Family - John Springer, Advanced
Micro Devices, Inc.

4:00 The Texas Instruments I^2L SBP9900 Microprocessor -
Ben Sloan, Texas Instruments, Inc.

5:30-7:00 Social Hour - Athenaeum

THURSDAY, APRIL 21

MICROPROCESSOR TESTING

Chairman: W. Richard Scott, JPL

9:00 a. m. Microprocessor Qualification Criteria - Eugene R. Hnatek, Monolithic Memories, Inc.

10:10 Coffee Break

10:30 Overall Review of Testing Techniques for Microprocessors - Rick McCaskill, Macrodata Corporation

11:30 An Approach to Qualification Testing Microprocessors - Lenward Holness, Hughes Aircraft Co.

12:30 Lunch

1:30 Hi-Rel Procurement Specifications for Microprocessor and LSI Memory Circuits - John Shea, Integrated Circuit Engineering Corporation

3:00 Low Cost, Functional Approach to Microprocessor Testing - V. V. Nickel, Questron Corporation

3:40 Testing Microprocessors: Stepping up to the Task - Douglas Smith, Tektronix, Inc.

FRIDAY, APRIL 22

POTPOURRI

Chairman: W. Richard Scott, JPL

9:00 a. m. General Processing Unit - Robert Fosdick, Tracor, Inc.

9:40 Radiation Effects on Microprocessors - Paul Measel, The Boeing Company

10:30 Adjournment

CONTENTS

SECTION 1

INTRODUCTIION

CHAIRMAN'S INTRODUCTION

Robert E. Covey
Jet Propulsion Laboratory

This is the second of these microprocessor and LSI conferences. The first one, held about six months ago here at Caltech, was more of a workshop, in that we divided into smaller groups and addressed specific problems. We tried to reach a consensus on what we thought we needed in microprocessors and how to go about obtaining them, testing them, etc. We have also labeled this meeting as a workshop, although the intention is to be together in a single group for the entire two and one-half days.

This is a rather formidable room to attempt to have an informal workshop, but I would like to foster an air of informality, if possible. We don't wish to have presentations with no feedback from the audience. We would like to get comments and discussion and controversy.

What we plan to cover first this morning is a quick report on what we did at the last meeting. This will be a very brief version because most of you have the written report of the last meeting. Then we will describe what we hope to accomplish at this meeting. Then the session begins in earnest, with presentations from manufacturers. The big difference between this workshop and the last one is that at the last one there were primarily users talking about what they needed and would like to have. This time we have invited the manufacturers, who are the people who are going to have to provide those things (if it is possible) to tell us what they are doing in the area of high reliability integrated circuits and microprocessors. Hopefully, we may act as a catalyst to get something going.

One rather formal element of this meeting is the customary welcome from the Laboratory's top management. Last time we had Fred Felberg give us a few words and I have asked him to do it again. Fred is the Assistant Laboratory Director for Technical Divisions. That means that mail which comes to JPL addressed to the Chief Engineer or the Technical Director, or similar titles, goes to Fred. The Lab is organized in a matrix organization, with project offices and the working scientists and engineers in technical divisions. Fred is thus the leader of the bulk of the scientific and engineering people at JPL. We are happy to see you all here, and I hope we'll have our beautiful California weather for the next few days.

WELCOMING REMARKS

Fred Felberg
Jet Propulsion Laboratory

I want to extend a welcome to all of you to this second workshop. As testimony both to the dynamic character of the technology with which you are working and also the significance of and difficulty of the challenges that are involved, yesterday I received in the mail a letter announcing that the Institute for Defense Analysis (IDA) is planning on holding a workshop in May which appears to have an agenda very similar to this one. I suspect that there are efforts being made all over the country to try to address some of the same challenges. I think this is testimony to the importance of what you are trying to do.

I was particularly impressed to see that, in fact, there are going to be fourteen representatives from manufacturers of integrated circuits here. I think that should provide the basis for a very interesting program.

Again, I want to welcome you on behalf of Caltech and JPL and give you all kinds of encouragement. With that, I'll let you get on with the real business of the workshop. Thanks for coming!

REPORT ON THE MICROPROCESSOR SEMINAR, PHASE I

Paul Lecoq
Jet Propulsion Laboratory

These workshops were set up as relatively informal forums for the discussion of LSI as it is used in high reliability applications. Formal papers were not solicited to assure that discussions would be timely and flexible. The goals of Phase I of the Microprocessor Seminar were as follows:

- To establish communications and to encourage cooperation in qualifying and using high rel LSI.
- To work toward common LSI requirements in high rel applications.
- To work toward common approaches to qualification of LSI.
- To present a coordinated market for high rel LSI to interested manufacturers.

Only government users and contractors were invited to allow users to "get their ducks in order" before asking for response from manufacturers. Discussions were frank and far-ranging. Attendees got an opportunity to meet their counterparts in other agencies of the government, arguing out differences of opinion and seeking agreement. The first goal, communications, was met very well. If fully common specifications and requirements were not agreed upon, at least the reasons for the differences that exist and the problems were understood. Several cooperative efforts at common development resulted from the workshop.

The workshop started with individuals in the Air Force, Army, Navy, JPL, Caltech, and NASA presenting the status of high rel LSI from their perspectives. The military market was assessed. Several innovations to the normal design and qualification process were suggested. The workshop then split up into three working groups, each with its own problem to discuss. On the third day the leaders of the working groups presented their conclusions for further general discussion.

The following outlines summarize the presentations.

<u>Market Assessment</u> - R. Martinez, NOSC

- A discussion of the Navy standardization program. LSI can be standardized both as parts and as systems.
- Surveys define a general set of microprocessor requirements. Users want everything in the standard that is available in any microprocessor.
- Three-step standardization program

    An interim 8-bit processor

2. 16-bit longer term standard
3. Bit-slice architecture is a long term goal

<u>LSI in the Air Force</u> - J. DeCaire, AFAL

- The real technology driver is money. As money decreases, there will be more cooperation.

- Standardization is a way of doing business, not just a way of selecting parts. Standardization must be an integral part of the development process to save money.

- Programming is an iceberg. Only 10% is visible. Much more work is needed in standardizing software.

- Qualification potential and radiation hardening are major factors in device selection.

- The Air Force RCA GPU bit-slice processor is a good candidate for standardization.

<u>LSI Standardization</u> - D. Haratz, Army ELC

- Standardization is not always desirable. In trying to please everyone, sometimes no one is pleased.

- Different environments lead to different requirements.

- Standardization at the component level is a good idea. Module level standardization may not be usable in some environments.

<u>Non-standard LSI Approach</u> - C. Mead, California Institute of Technology

- Custom LSI is not a bad word. It works. It can be made reliable. It can be made at a reasonable cost.

- Caltech is developing custom chips as class projects.

- Caltech development has shown that design on silicon is feasible for military use.

  -design the device

  -do the artwork

  -contract the fabrication out to an approved line

  -do your own qualification

- Structure the design so it can be understood and tested. If you don't, the device will surely be impossible to test completely.

- Plan ahead on needs. If future needs are anticipated, development can be guided. If all effort is concentrated on present needs as it seems things are done now, we will have to be satisfied by what others develop. Therefore future needs may not be met.

Diagnostics and Test Chips for Predicting Reliability - J. Maserjian, JPL.

- There is very little visibility for the user in the manufacturing process.
- LSI devices cannot possibly be tested completely. Therefore confidence in the fabrication process is an important part of qualification.
- Test chips provide a window into pertinent process parameters.
- Chemical as well as electric diagnostics further reveal important process parameters.

Non-user Perspective - Sam Davis, Electronic Engineering Times

- Manufacturers are also concerned about high rel specifications.
- There are frequently difficulties in meeting specifications. Manufacturers feel they are arbitrarily tighter than necessary to assure reliability.
- A great deal of work is still required to develop MIL-M-38510 in specifying reasonably achievable specifications.
- More user-manufacturer communications are required to develop reasonable specifications.

WORKING GROUPS

Group A - E. Urban, NOSC

Commonality of Requirements and Potential for Standardization

- A large amount of commonality exists between users, particularly at the parts level.
- There is a large class of requirements on which general agreement can be reached. However, no single processor family can answer all needs. A byte-slice processor family such as the GPU or a CMOS/SOS 2900 is a good candidate for general standardization.
- Standard specifications cannot completely satisfy all users; however, specifications could be written to eliminate many differences.

Group B - R. Conklin, AFWL

LSI Qualification Mechanisms

- MIL-M-38510 is a good mechanism for qualifying microprocessor families but the devices are really systems, not components, and must be specified as such.

- Understanding the process is a significant part of the qualification of LSI.

- The software and other system-oriented parameters must be characterized.

Group C - W. R. Scott, JPL

Testing Microprocessors and Other LSI

- How can a system-on-a-chip be tested?
  With great difficulty and much care. The problems of microprocessor testability were discussed and recommendations were made for thorough characterization and qualification testing.

- Which potential users are involved in LSI testing?
  Various testers were catalogued and discussed. It was also recommended that a government test equipment users group be formed. Compatible test program software could be exchanged between agencies and Centers.

- Can testability be designed into LSI before manufacture?
  Yes, much more could be done than is currently being done.

- How can tests be specified for LSI?
  User characterization of the device plus manufacturer's test sequence must be used as a minimum. The user's applications, to the extent they are known at the time of LSI testing, should also be included.

OBJECTIVES OF THE MICROPROCESSOR SEMINAR, PHASE II

W. Richard Scott
Jet Propulsion Laboratory

The best way to describe the objectives of this seminar is to start with what they are not. This workshop is <u>not</u> a trade show in which microprocessor marketeers fill us with future product release dates and descriptions of those devices planned but not yet out of the lab. That's good and useful information but it is not the subject of this workshop. Neither is it a pitch by test equipment manufacturers on which brand of automated test equipment is the best for qualifying and screening microprocessors. Nor is it a final coordination of a MIL-M-38510 slash sheet.

What it is, rather, is a survey of the problems associated with high reliability applications of a new, unproven, rapidly growing generation of large scale integrated circuits--problems like optimal selection and standardization; detailed evaluation and qualification; controlled procurement screening and acceptance testing; and a host of other similar problems.

What makes this workshop different is that it is an attempt to bring all the different parties together in an atmosphere of friendship and helpfulness to air all sides of all of the issues. We hope to strike some note of acceptable compromise: where the LSI manufacturer is not required to perform the impossible; where the test agency is selectively and systematically identifying and removing workmanship defects and deficient designs without introducing failures; and where the hardware designer is getting the over-all best device for his requirements, a device that he will correctly apply and which will perform according to specification. Put simply, the workshop is meant to air all sides of the issues so that no party gets unreasonably saddled with impossible requirements.

To this end, the conference is divided into four sessions. During the first session, manufacturers and users will report on recent developments in LSI intended for high reliability applications. Next we will hear from a number of people who have been developing new techniques in qualification testing of microprocessors for high reliability applications. Following the testing session there will be a discussion on hi-rel procurement specs--the application of MIL-M-38510 and MIL-STD-883 as well as alternative approaches. In the final session there will be presentations on the Air Force general processing unit and on microprocessor radiation effects testing.

Though it has been mentioned already, it should be repeated here that this workshop is not a formal presentation of technical papers, it is an informal presentation of technical ideas intended to stimulate a lot of discussion. We hope you will ask questions, challenge points of view, and offer alternative solutions to our many problems.

SECTION II
PRESENTATIONS ON ADVANCED LSI DEVELOPMENTS

Chaırmen

Michael M. Ebersole
Jet Propulsıon Laboratory

and

Ralph Martinez
Naval Ocean Systems Center

CMOS LSI ARRAYS FOR SYSTEM APPLICATIONS

W. A. Clapp
RCA Corporation
Camden, New Jersey

This paper presents a brief summary of the main points made during my talk at the JPL Workshop on microcomputers held April 20, 21 and 22. The two major sections covered during the talk were: 1) why use LSI?, and 2) what LSI is available?

Why Use LSI?

The incentive to use LSI in system applications falls into three major factors which are the predominant influences on final system cost:

1. LSI reduces the number of parts from the viewpoint of assembly, reliability, and logistics, overall costs decrease almost linearly with the number of parts. Even low cost parts require additional money to test, assemble, solder, and stack. A major electronic equipment built by RCA in the late 1960's with MSI parts cost about $8 per gate in the equipment. Another major electronic equipment built by RCA in the early 1970's utilizing LSI costs about $0.35 per gate in the equipment.

2. LSI reduces interconnects - LSI minimizes interconnects--wires, pins, cables, connectors, etc., all of which are expensive and contribute to reliability problems.

3. LSI reduces power dissipation - Power dissipated means power to be supplied, converted, and removed by convection and/or radiation. LSI provides the ultimate in reduction of power dissipation to accomplish a given function.

A major challenge in the use of LSI is provided by the way dynamic growth is provided to the potential user. Technology has been increasing in complexity approximately by a factor of two every year. One then is always tempted to wait one more year for the solution to the problem, or having made a decision, one, in the following year, is presented with a much more economical solution. This paper will highlight some of the efforts RCA has been involved with to help meet the challenge. Many of these efforts have been funded by government agencies as well as RCA funds.

What LSI is Available?

I will attempt to summarize the parts available from RCA from both its commercial lines and the special products developed internally but available for government applications. From the commercial side of RCA in CMOS on Silicon, the following categories of parts are available: standard CD4000 parts, high reliability CD4000 parts, radiation hard CD4000 parts, the 1802 microprocessor (which is multiple sourced), and 1802 support parts including RAMs, ROMs, and additional support parts announced or soon to be announced. Also available from the commercial side of RCA are two types of 1K RAMs (which are also multiple sourced) in the CMOS/SOS technology. This CMOS/SOS effort is being supported by a new production facility in West Palm Beach, Florida. Other SOS parts are currently available with additional products to be announced within one year.

Our special products have been developed on our internal quick turnaround facility at Somerville, New Jersey, the Solid State Technology Center (SSTC). This facility for the last 2-1/2 years has been running CMOS/SOS in a pilot line fashion with data collected on the process during this time from a process control insert chip (TCS-010) which is used on all SOS wafers. Both process parameters and reliability data have been collected. Reliability data to date indicates better than 200,000 hours MTBF at 125°C with 10 volt V_{DD} applied.

This line has been used to produce close to 100 CMOS/SOS custom arrays. These arrays were designed using four different approaches. RCA selects the best approach for the LSI array from a review of the boundary conditions of the program. One approach is completely automatic, and three approaches heavily utilize computer aided design techniques.

The automatic approach uses the standard cell concept. Chips are literally laid-out over night. This capability exists in CMOS metal gate, CMOS silicon gate, CMOS/SOS, and radiation hardened CMOS/SOS. A flow chart of this process is shown in Figure 1.

The three approaches to LSI using computer aided design techniques are handcrafted custom cells, Gate Universal Array (GUA), and modifying the output of the automatic placement and routing program. Each of these approaches has been used in 4-5 different technologies within RCA during the last ten years.

In addition to generating the LSI array layout, we also have extensive support programs for circuit simulation, logic simulation, fault analysis, test generation, and related capability up through hybrid packaging and printed circuit board fabrication.

Many of the existing CMOS/SOS chips are general purpose ones which have been used on several programs. A summary of some of these will indicate the types of arrays available. Figure 2 summarizes several useful chips for signal processing. The SOS ROM chip characteristics are summarized in Figure 3.

We have also constructed a custom microcomputer chip set--the ATMAC. This set has an 8-bit expandable data path and an expandable 8-bit control chip. When four LSI arrays are put together to form a 16-bit microcomputer, it is capable of 1-3 million instructions per second. A block diagram of the ATMAC is shown in Figure 4.

The General Processing Unit (GPU) is another 8-bit expandable CMOS/SOS data path chip. This chip was designed and processed under an Air Force contract. A block diagram of the GPU is shown in Figure 5.

APPENDIX A

CONFERENCE ATTENDEES

JPL ATTENDEES
(4800 Oak Grove Drive, Pasadena, CA 91103)
(Telephone Area Code 213)

Wil Bartel	354-2047
Daniel Bergens	354-5622
Merrill P. Burnett	354-6441
R. Caplette	354-4124
Lyn Coffey	354-2819
Robert E. Covey	354-5590
Stuart Dodge	354-5003
Mike Ebersole	354-4595
E. Greenberg	354-7257
Tom Griswold	354-4145
Larry Hess	354-5527
George Jenkins	354-4219
Merrill Johnston	354-3574
Ervin Klippenstein	354-4009
Gary Kunstmann	354-5341
Paul Lecoq	354-5517
James Lonborg	354-4029
Walter Mallen	354-3070
Joe Maserjian	354-3801
Emmett Miller	354-6026
Richard Piety	354-2298
Melvyn Pinck	354-3439
Dave Rennels	354-4096
Dick Scott	354-5750
Alex Shumka	354-5059
Dean Slaughter	354-2084
Kamal Soliman	354-5622
Alan G. Stanley	354-5787
Roy Summers	354-5166
Man Tam	354-6288
Art Trigonis	354-6167
Gordon Wicki	354-3533
Vance Tyree	354-4623
Gordon Wiki	354-3533
John Wilson	354-6246
Larry Wright	354-5603
E. Zundel	354-2084

NON-JPL ATTENDEES

J. R. Adams
Sandia Corp.
Albuquerque, N. M. 87115
(505)264-7327

W. F. Barrows
NASA-Ames
M/S 244-7
Moffett Field, CA
(415)965-6525

D. Butcher
Rockwell Int.
3340 Mira Loma Ave.
Anaheim, CA
(714)632-1573

Tony Cara
Intel Corp
3065 Bowers Ave.
Santa Clara, CA 95051
246-7501

Ralph Carroll
AFWL/ELS, M/S 53
Kirtland AFC, New Mexico 87117
(505)264-5608

C. Chrones
DCA
975 Benica Avenue
Sunnyvale, CA 04086

W. A. Clapp
RCA/ATL
Front & Cooper STS, Bldg. 10-4-7
Camden, New Jersy 08102
(609)963-8000 XPC 6757

R. E. Conklin
Air Force Weapons Lab
Wright Patterson AFB Ohio 45433
(513)255-4557

J. A. Davidson
Martin-Marietta Corp
P. O. Box 179
Denver, Colo. 80201
(303)979-7000

Sam Davis
Electronic Business (EBN)
22543 Ventura Blvd.
Woodland Hills, CA 91364
(213)347-3814

Larry Drayer
Tracor, Inc.
6500 Tracor Lane
Austin, Texas 78721
(512)926-2800

Andy Edwards
NASA HQ
Washington, D. C. 20546
(202)755-3524

O. Ely
Teledyne-Brown
(205)532-1608

Frank Emmens
NASA-MSFC
Huntsville, Ala
(205)457-3803

R. L. Fitzwilson
Questron Corp
101 Continental Blvd, #500
El Segundo, CA 92131
(213)640-0901

R. E. Fosdick
Tracor, Inc.
6500 Tracor Lane
Austin, Texas 78721
(512)926-2800

T. C. Goldsmith
NASA-GSFC
Greenbelt, MD 20771
(301)982-5374

Dave Griswold
RCA
6363 Sunset Blvd.
Hollywood, CA
(213)461-9171

Leon Hamiter
NASA-MSFC
Huntsville, Ala 35812
(205)453-4572

E. R. Hnatek
Monolithic Memories
1165 E. Avenue
Sunnyvale, CA 94080
(408)739-3535

Len Holness
Hughes Aircraft Corp
Bldg. 6, C137
Culver City, CA 90230
(213)648-8587

William H. Howle
NASA-MSFC
Huntsville, Ala. 35812
(205)453-4572

Steven Kaplan
Raytheon
Hartwell Road
Bedford, Mass 01730
(617)274-7100

Gerard Kiernan
Sperry/GSFC, Code 310.1
Greenbelt, Md. 20771
(301)982-6633

Michael G. Knoll
AFWL/ELP, Stop 53
Kirtland AFB, New Mexico 87117
(505)264-0316

Frank Langley
Raytheon Co.
Hartwell Rd.
Bedford, Mass 01730
(617)274-7100

Hank Malloy
Intel Corp/JC 133-13
3065 Bowers
Santa Clara, CA
(408)996-0585

James P. Martin
National Semiconductor
2900 Semiconductor Dr.
Santa Clara, CA 95051
(408)737-5880

Ralph Martinez, Code 4300
Naval Electronics Lab
271 Catalina Blvd.
San Diego, CA 92120
(714)225-2752

Tak Matsumoto
NASA-Ames, M/S 244-7
Moffett Field, CA
(415)965-6525

Rick McCaskill
Macrodata
6203 Variel Avenue
Woodland Hills, CA 91365

Paul Measel
Boeing Corporation
Seattle, Washington
(206)655-2104

Gary Mocko
USA Electronics Command
Ft. Monmouth, N. J.
(201)544-2213

Vince Nickels
Questron Corp
101 Continental Blvd., Suite 500
El Segundo, CA 90245
(213)640-0901

Les Palkuti
Naval Research Lab, Code 5216
Washington, D. C. 20375
(202)767-2252

W. E. McInnis
NASA HQ
Washington, D. C. 20546
(202)755-3040

L. X. Phan
Hughes Aircraft Corp.
(213)391-0711, X6518

R. E. Price
Hughes Aircraft Corp
P. O. Box 92919
Los Angeles, CA 90009
(213)648-8587

Jim Raymond
Mission Research
(714)454-3046

Pete Rosenberg
Questron
101 Continental Blvd. #500
El Segundo, CA 92131
(213)640-0901

J. A. Saultz
RCA/ATL
Front & Cooper Sts., Bldg. 10-4-7
Camden, N. J. 08102
(609)963-8000 XPC 6757

Joe A. Ruggieri
CTL
763 US Hwy 17-92
Fern Park, Florida 32730
(305)831-2700, X25

John D. Shea
ICE - Suite 260
N. Winchester
Santa Clara, CA
(408)248-1843

W. C. Slemmer
Sandia
Albuquerque, N. M. 87115
(505)264-4364

Ben Sloan
Texas Instruments
P. O. Box 5012
Dallas, Texas 75222
(214)238-3933

Doug Smith
Tektronix
P. O. Box 500, M/S 39-804
Beverton, Oregon

John Springer
Advance Micro Devices
901 Thompson Place
Sunnyvale, CA 94096
(409)732-2400

Otis Tabler, Jr.
FAST
6568 Beachview Dr., Suite 133
Palos Verdes, CA 90274

T. F. Tao
Naval Postgraduate School
Monterey, CA 93940
(408)646-2393

James E. Thomas
Hughes Aircraft
Bldg. 6, M/S D147
Culver City, CA
(213)391-0711, X4766

D. Vesely
Hughes Aircraft Corp
Bldg. 377, M/S D208
P. O. Box 92919
Los Angeles, CA 90009
(213)648-8587

Bill Walsh
Counter Measures Magazine
2065 Martin Ave., Suite 104
Santa Clara, CA 95050

C. Winfred
Teledyne-Brown
(215)532-1342

H. E. Yearwood
NASA-MSFC
EG-02
Huntsville, Alabama 35812
(205)453-5469

M. B. Colligan
Hughes Aircraft
Culver City, CA
(213)391-0711, X6518

I would now like to move to the radiation hardened CMOS/SOS efforts within RCA.

The insulating substrate provides a large increase in protection to transient radiation over that obtained with the CMOS on silicon. Additionally though, there are two approaches necessary to achieve the hardest part available. A certain level of hardness can be achieved through process developments alone--developing the hardened oxide to achieve total dose protection. To this must be added a circuit design approach to further compensate for the radiation induced effects to achieve the hardest parts. RCA is working on programs to achieve the hardened oxide on CMOS/SOS as well as circuit designs to further extend the level of hardness achievable. We are very optimistic about the results achievable in a reproducible process. We have already made parts hard to greater than 10^6 rads (Si) and hard to greater than 10^{11} rads (Si)/sec. Currently, we are evaluating several approaches to improve the process still further.

In conclusion, the following six points summarize the CMOS capability at RCA:

- Production lines for CMOS on Silicon running now
- Production lines for CMOS on Sapphire running now
- Three approaches for generating special products
- Two microcomputers available now

  1802
  ATMAC

- Two bit slice microprocessors available now

  GPU
  DEU

- Several efforts currently focused on radiation hardened parts

  CD4000 Line
  1802
  GPU
  ROM
  GUA
  Standard Cell Family
  Code Generator

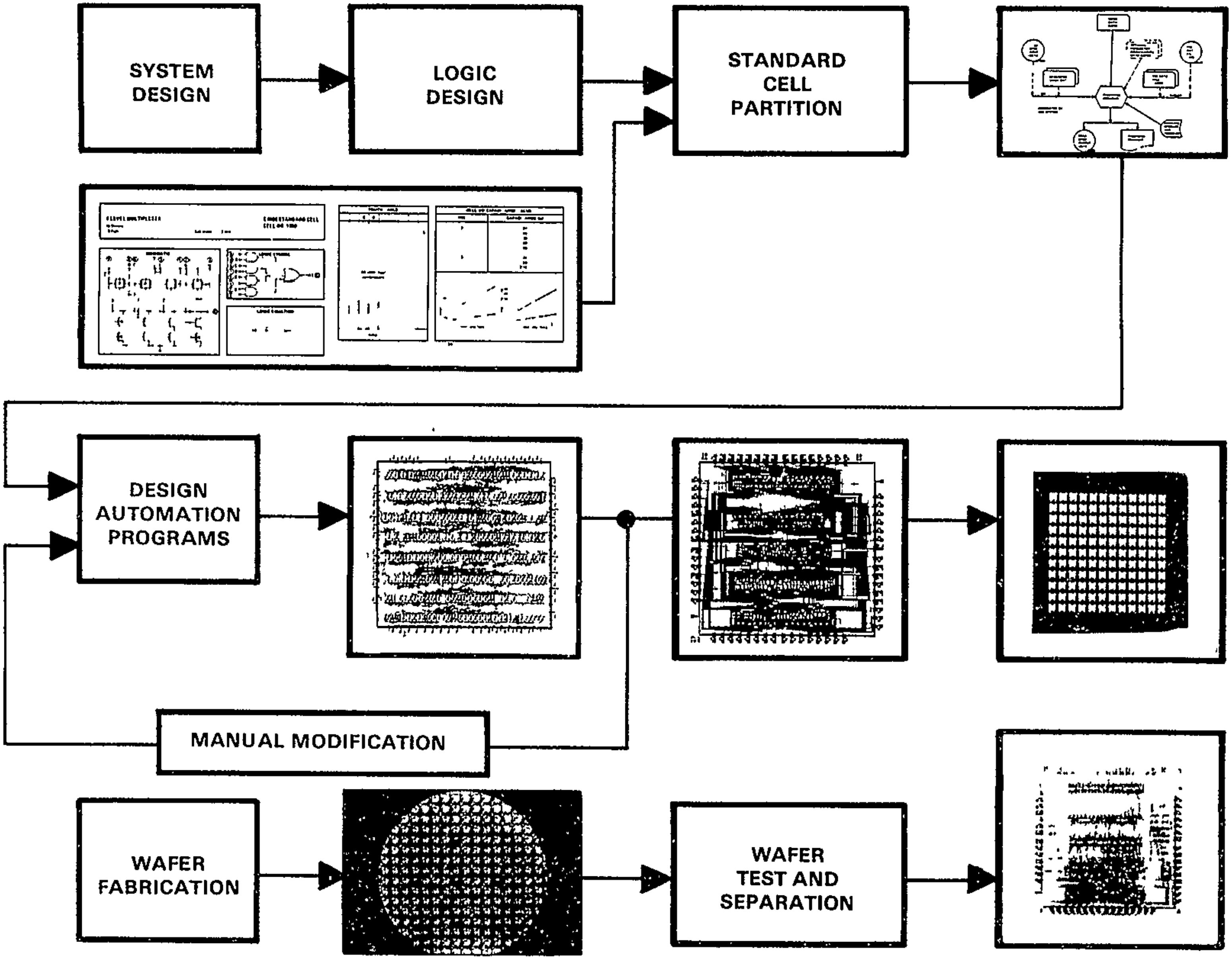


Figure 1. Computer aided design (CAD) approach

ORIGINAL PAGE IS
OF POOR QUALITY

ADDER: TCS-008, TCS-030, TCS-065

MULTIPLIER: TCS-001, TCS-002, TCS-039, TCS-057

CORRELATOR: TCS-040

FFT BUILDING BLOCKS: TCS-015, TCS-016, TCS-017

CODE GENERATORS: TCS-045

FREQUENCY SYNTHESIZER: TCS-047

D/A CONVERTER (RAD. HARD): TCS-043

A/D CONVERTER (RAD. HARD): Just Started

Figure 2. Signal processing building blocks

- CMOS/SOS Technology
- Fully Static Operation
- 1024 Bits, 256 x 4 Formats
  512 x 2
- Mask Programmable (EPI)
- 100-ns Cycle Time/50-pF Load
- Tri-State Output (TTL Compatible)
- 4-13 Volt Operation, Single Supply
- 10-uA Leakage at 10-V Typical
- Dynamic Power 100 mW at 10 V
- Chip Size: 132 mils x 144 mils
- Chip Select/Output Register

Figure 3. SOS ROM chip characteristics

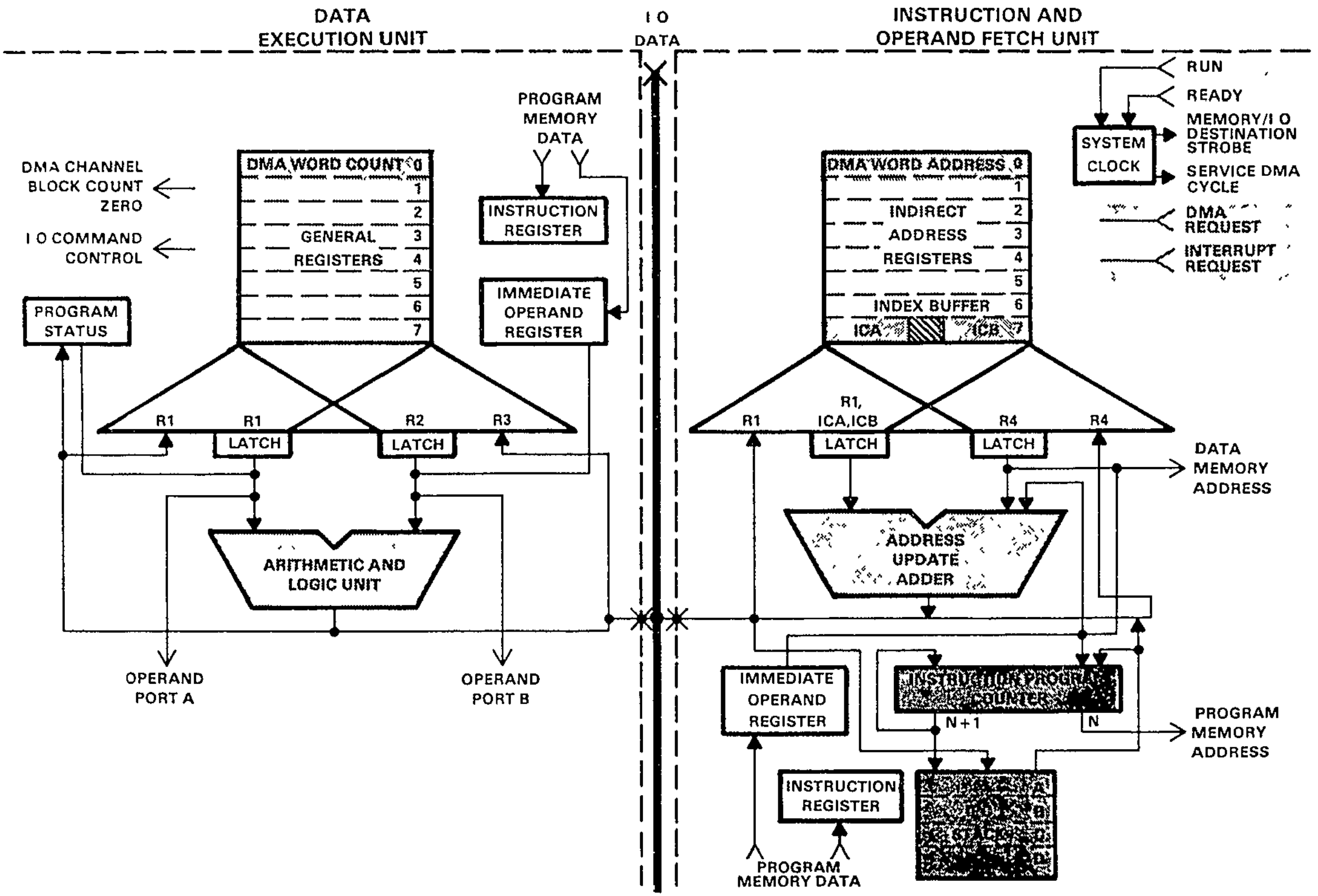


Figure 4. Block diagram of the ATMAC

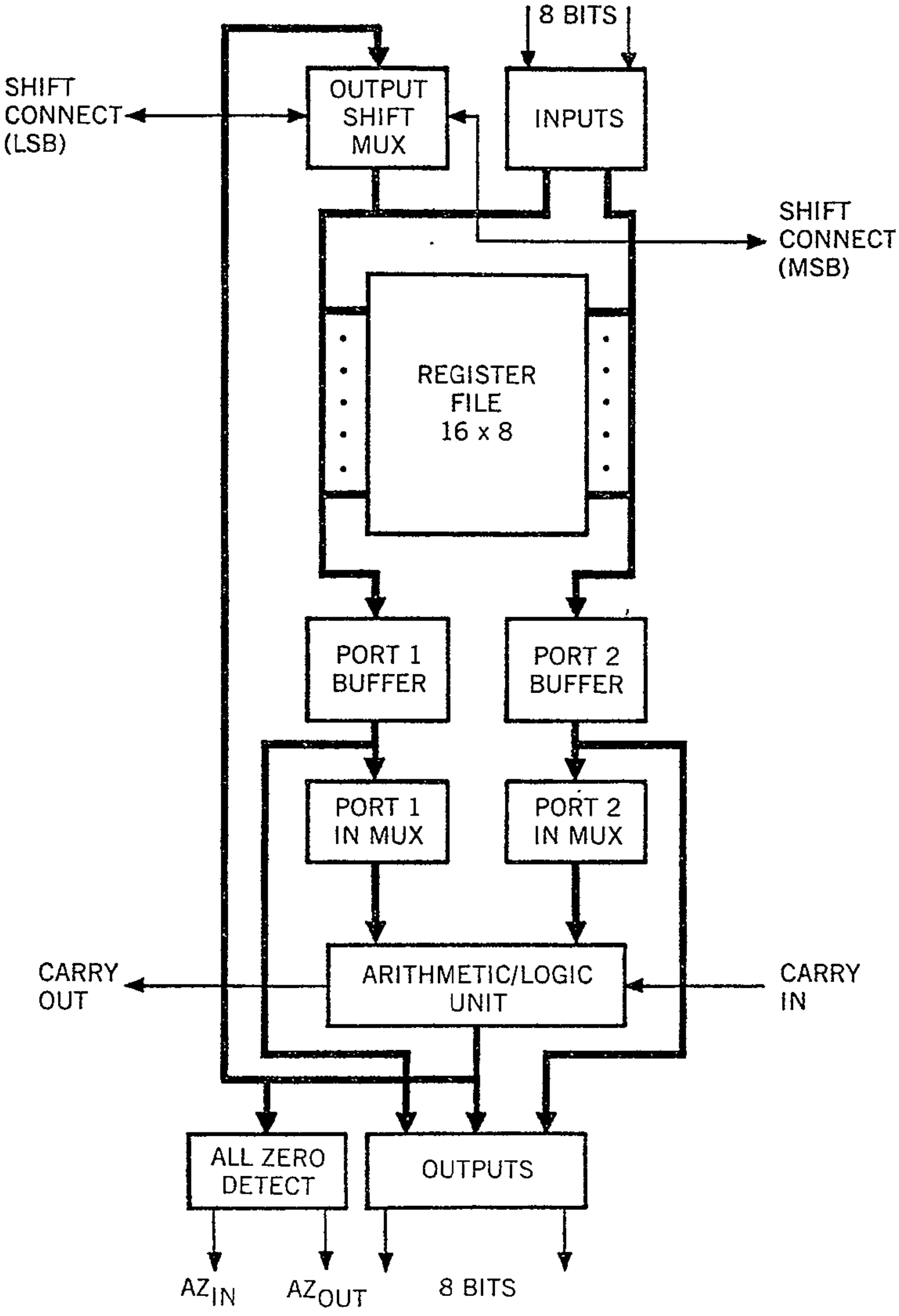


Figure 5. Block diagram of the General Processing Unit (GPU)

CURRENT STATUS OF RADIATION HARDNESS OF CMOS/SOS

Robert Conklin
Air Force Weapons Laboratory

and

Alan G. Stanley
Jet Propulsion Laboratory

[The informal taped remarks of Conklin and Stanley are summarized below.--Editor.]

ROBERT CONKLIN

My talk will emphasize philosophy rather than data. I think the need for understanding the philosophy behind some of the efforts that are being made today is basically what is causing people to have a lot of questions here. Let's look at the various things that are affected in radiation.

First, there is transient radiation performance. We've talked about silicon on sapphire approaches, which have been CMOS/SOS or other SOS approaches in the past. By using the dielectrically isolated substrate, one gets improved transient capabilities. And there is not the great amount of leakage that occurs with the bulk silicon. The GPU device, which was not designed specifically to be transient-radiation hard per se, can survive 8 x 10^9 rad/sec without upset. But although SOS provides much better transient performance, that performance can't be achieved without going to a great deal of trouble. Gold doping and a lot of other things have been tried by people like Sandia in order to achieve it.

Now, let's look at the question of total dose performance. There are two ways to achieve total dose performance. One can take an inferior process and do everything possible in the circuit design as, for example, on sapphire, tying down back channels and eliminating the transmission gates, both of which cause some problems as far as total dose is concerned. As a result, one can probably improve, say, a 5 krad part up to something approaching 50 krads. The best one can usually do with design is probably 10 to 1. It is not always possible to achieve even that, and there will often be hardly any improvement. That's one approach to hardening.

In the DNA APAR design, they have instituted those kinds of circuit tricks to improve the hardness as much as possible. There's another way of getting hardness (and this must go hand-in-hand with the circuit tricks if one is a purist and wants the absolute hardest device possible)--and that has to be a hard process. In order to get a hard process, there are many things one must do. All kinds of exotic gate insulators have been tried. Other techniques have included the use of aluminum oxide, chrome doping, straight dry oxide, and even the aluminum ion implant.

So we have tried everything. But when we get right down to brass tacks, it's been demonstrated recently that the best means of getting good performance, as far as radiation total dose is concerned, is to use what we know the most about--which is steam oxide--and to control our processing parameters by such means as (1) no high-temperature annealing (eliminating all those processing and reprocessing steps at more than 1000°) and (2) doing everything else that we can, even if it means ion-implanting the impurities in the n and p channels and things of this nature.

Even though we do all these things, we don't always succeed, because some days are bad processing days. A bit of plaster falls off the ceiling, or something like that.

There is a basic disagreement among some of us. Personally, I don't adhere to the idea that we need the world's hardest parts. We just need parts that are hard enough. To give an example (and this is no criticism of the APAR routine): if we strap the p channel devices and eliminate transmission gates, we have to put in possibly 5 or 6 devices to do the same thing that would have been done with one transmission gate. If we say "I'm not going to allow them to stack more than three high," that has an impact on the fan-out and probably on the fan-in capability of the device, so although we've achieved something we wanted, we've also given up something--the packing density.

On the other hand, if we could sit down and design the chip using transmission gates, being able to stack 5 outputs, etc., we would have a smaller device. Our approach in the GPU has been to do that. We have not tied down the back channels, we have not eliminated transmission gates (we sprinkle them through quite continuously), and we have not limited the fan-out.

Why can we do that kind of thing? Basically, we have taken that approach because we felt that an 8-bit chip was probably the optimum as far as systems applications are concerned. When we go for an 8-bit chip, we are going to have something that is pretty hard to build to begin with. And we are talking about a chip that is 201 by 210 mils.

What is the effect of this? Well, obviously, our chip won't be quite as hard as the chip where all these tricks are taken into consideration, but, on the other hand, our chip may be half the size of the other one. To get back to the basic facts of the situation, it doesn't matter what we do in trying to get a hard device, if we don't develop something in the way of a hard process. And even a hard process probably won't get us to the pot of gold at the end of the rainbow.

What we have essentially said, in some of our approaches at the Avionics Lab, is that we don't want the hardest thing in life--we want something that gives us the maximum capability we can achieve with decent yield in computer applications--and we hope we can develop a hard process that will get us up around the level of 300 krads or so. We think it is conceivable that we can do just that with a hard process.

Now I've stated my philosophy, and I'll allow equal time to anyone else. I just want to set the facts straight as to why people say they can get harder devices by doing these circuit tricks, and why some devices are hard when they don't have these circuit tricks.

ALAN G. STANLEY

I wasn't going to make any philosophical remarks; I just want to give an idea of the present state of the hardening effort. One of the more interesting things in a meeting like this is that people discuss all types of radiation hard circuits, but if you check into it, you will find that they aren't actually radiation hard at this moment, and the reason this comes about, I think, is worth discussing.

Now, first of all we are here talking about LSI, and LSI devices have certain requirements. For one thing, the way that LSI is designed requires a fairly dense pattern, and we are therefore talking about silicon gate technology. In addition, if we want both fast circuits and circuits that are not affected by transient upset, we must have CMOS/SOS. So we ought to inquire about the present state of hardening of this particular combination and not of some other combination which has already been hardened; for example, the CD4000 series are hard now to practically 10^6 rad, but they are not silicon gate CMOS/SOS, so therefore they do not have this problem. In order to build any microprocessors of any complexity, and also RAM's and ROM's, we must have this combination. It appears that up to this moment no silicon gate CMOS/SOS devices have been built and tested of a complexity greater than either n or p channel test chips or something like the CD4007. How does this come about?

First of all, where does the present hardening technology stand? It appears that if one takes a simple device one can divide its radiation characteristics into the shifts of the n channel and the p channel threshold voltage of a standard test transistor. If we examine these particular parameters, it appears that the n channel shift is under reasonably good control even up to 10^6 rads. The shift is less than 1 volt or thereabouts, and if one starts with a voltage threshold of about 2 volts or so, one certainly doesn't have a problem under radiation.

Now the p channel situation is slightly different. The shift at 10 rads is about 1 to 5 volts, and the cause of this variation is its dependence on the thickness of the oxide. So here one is trading reliability against radiation hardening: the thinner the oxide the harder it gets. Now that is an important parameter, because the shift in the p channel threshold voltage affects the speed; the bigger the shift the slower the device is going to get after radiation.

Another parameter, which is very important, is the back channel leakage current in CMOS/SOS devices. At this moment it appears the state of the art is as follows: 1 to 10 μA per mil of channel width. If we now consider a complex device which has many thousands of mils of channel width, it means that at this moment it is not possible to get any kind of LSI device that doesn't leak at least in the milliamp range , after radiation - maybe even more - and that is a serious disadvantage to the present process, because one of the main reasons why people use CMOS is to keep the power down, and now suddenly the power gets worse with radiation.

As far as I know, nobody has gotten consistently better results than what I have put here on the board. Those are the present state of the art, and I think you can argue that accepting all these criteria and, in addition to that, making design changes such as Bob has just mentioned, you could easily design devices right now that could survive to 10^6 rad except that they would be quite seriously degraded, and if we can live with degraded devices I think we are there.

There is one additional problem which I haven't mentioned, and that is the problem of manufacturability. What that means is that in order for the manufacturer to achieve the numbers I have put on the board he has to have a tightly controlled process and he is unable to do this at the moment; at least he gets a terrible yield. Of course that is just another definition of process immaturity. I think it is true to say that there is no mature process around right now that can make radiation hard LSI devices. That is really the greatest deterrent to getting devices right now, since there are relatively very few devices made on a given wafer of LSI form, and if the yield also goes down to a very low level, the whole game becomes uneconomical. So the biggest effort that needs to be made right now is to improve the manufacturability; in other words, make sure that the process isn't so difficult to produce that a very slight change in any of the processing parameters causes a lower yield. If that is achieved, and that has not yet been achieved, it is then possible to make LSI devices that will operate to 10^6 rad, provided that the design rules are taken care of, which is not that difficult to do. Unfortunately, with degraded parameters, it may be necessary for someone who needs to have his devices operating to 10^6 rad make do with these degraded parameters. In other words, the device may leak a lot and things of that nature.

There is one final statement I would like to make, and that is the philosophy of why one might want to have devices that are hard to at least 10^6 rads. This again, just like the yield situation, is a question of safeguards, because if you have a process that is very temperamental, as the present ones are, you have to continuously test it for radiation and your yields are going to go down very badly. You are much better off in having a process that is hard, let us say, to 10^6 rads, and then if your requirements are one order of magnitude lower than this you do not get additional very large losses in testing a process that is insufficiently developed and therefore shows very large variations in yield. So I think that is another concept that should be taken into account. I think it is this present set of circumstances that causes us not to be able to purchase LSI devices in silicon gate CMOS/SOS at this moment.

77-39

RADIATION HARDENED CMOS/SOS DEVELOPMENTS

Daryl Butcher
Rockwell International
Anaheim, California

(Paper Not Available)

77-39

RAYTHEON MACROMODULAR MICROCOMPUTER FAMILY

Frank Langley and Steven Kaplan
Raytheon Company
Bedford, Mass.

An overview was presented of the work performed under an Office of Naval Research contract to investigate the application of modular digital techniques to missile guidance and control. The work is now in the fourth phase.

Phase I and II

The Phase I & II studies addressed the feasibility of applying digital techniques to the missile functions of seeker signal processing estimation, guidance, seeker stabilization, autopilot/control, inertial reference, fuzing, telemetry, test and mode control for all classes of air-to-air missiles. Three generic classes of air-to-air missiles were identified during the course of the study to cover the range of applications from Sidewinder to Sparrow and to Phoenix. Digital algorithms for each function were defined, and various levels of complexity and sophistication were identified for application to each generic class. The analyses of Phase I and II showed that:

Modular, programmable, digital guidance is feasible, affords performance improvements and provides flexibility, modular expansion and system updating without major redesign.

2. A family of 14 major computer functional elements, very large-scale-integrated (VLSI) macromodules, in various configurations, using a common bus interface, will support the entire range of air-to-air missile functions. Figures 1 & 2 and Table 1.

3. Radar sensor signal processing dominates the throughput requirement and can be supported by a high-speed, general-purpose microprocessor module augmented with a 64-point, complex Fast Fourier transform module connected to the common bus.

4. Federated/distributed macrocomputer systems provide the best match of missile functions with computer capability, providing desired subsystem autonomy for modular design manufacture, assembly, test, maintenance and subsequent modification without system disruption.

5. Missile guidance and control systems readily partition into four autonomous and asynchronous functional groups for modular, federated microcomputer systems:

   a. Steering command generation (signal processing, estimation and guidance).

b. Missile stabilization and control (autopilot and inertial reference).

c. Seeker stabilization and control (tracking and stabilization).

d. Support functions (fuzing and telemetry).

6. Serial digital multiplex as defined in MIL-STD-1553A, provides an optimum interface between missile subsystems/computers and carrier aircraft avionics.

7. Unified software system using one high-order-language for system simulation and missile computer code generation together with structured design and modularity minimize software cost and risk.

Phase III

Phase III validated the performance and effectiveness of the macromodular microcomputer configurations defined in Phase II on an individual module basis; as whole microcomputers; and as federated microcomputer systems applied to specific missile types, i.e. Class I and Class II, using digital simulation techniques. Modular growth of hardware and software from Class I to Class II missile microcomputer systems was demonstrated to show performance improvement through simple macro-module replacement.

In both the missile systems configured, a common 16-bit microcomputer was found to satisfy the performance requirements for target seeker signal processing, estimation and guidance, (i.e. the steering command loop), by adding the hardware 64-point FFT module (FFT-1) to the μBus. The remaining missile functions, i.e. seeker head/platform stabilization and control, autopilot and fuzing/telemetry could each be satisfied with the 8-bit byte microprocessor module (MIL 8080-based) with or without a hardware multiplier module to meet the required throughput. Class II missile autopilots require a bipolar or CMOS-SOS version of the 8080 (Am2901-based) with hardware multiplier to meet the shorter computational delay requirement (600 μsec). Program sizes for each of these 8-bit processors do not exceed 1500 8-bit bytes. Throughputs of the 8 and 16-bit microcomputer configurations extend over the entire missile throughput range (Figure 2), using a standard μBus interface and the add-on/replacement of hardware modules. Support software depends upon the user's choice of an existing 16-bit minicomputer for emulation with bit-slice Am2901 RALUs, AN/AYK-14, PDP-11/34, and the 8080 package, as does the programming language, i.e. CMS-2, PL/M, FORTRAN IV.

Standard μBus

The common interface between μCPU, RAMs, (P)ROMS and digital I/O modules (i.e. DMAIO and PDIO) is the μBus which contains bidirectional parallel digital data (16), address (16) and read/write (1) lines (Figure 3). To satisfy the wide range of microcomputer configurations without restricting throughput, microbus interfaces were established during

Phase III based on the following rationale pertinent to the practicality of the µBus and system operational requirements:

1. µBus Standardization Concept - Determines standard input-output interfaces for connecting microcomputer family macrofunction modules whatever the phase of technology advancement, and hence the internal improvements/changes made to the modules.

2. µBus Use/Extent - Internal, (12 ins. max length), parallel digital interface between microcomputer macrofunction modules only, e.g. µCPU, memory (RAMS & P/ROMs), and I/O (DMAIO & PDIO) for a simplex microcomputer configuration, i.e. one µCPU. Federated microcomputer systems to interface between individual microcomputer I/O channels.

3. µBus Traffic - Minimized by autonomy of user modules, i.e. µCPU, µFFT, and I/O macromodules, through architectures which minimize frequent memory accesses/overhead operations.

4. General-Purpose µCPU Architecture - General-register for active and partial results, using multi-address, register-register instructions. No programmed I/O transfers to/from memory via µCPU except initializing commands to I/O channels.

5. µFFT Module Architecture - Internal register file to store FFT data points for high-speed/pipelined, butterfly arithmetic unit operation.

6. I/O Modules - Can initiate input sampling and data transfers to main memory in response to integral cyclic interval timers, i.e., for repetitive body motion sensing and stability loop processing, and can be commanded, (by programmed instructions), to initiate data transfers from memory for conversion and output to gimbal torquers/fin actuators. Radar sensor data sampling can be initiated by a programmed command and sampling rates similarly programmed.

7. µBus Control - µCPU receives memory access requests direct from I/O modules and provides access to bus upon completion of current µCPU - memory data/instruction transfer. Priority of access assigned by system timing constraints. I/O module generates end-of-block (EOB) interrupt for µCPU upon completion of data transfers to memory.

Figure 2 illustrates the inter-module interfaces with respect to standard µBus operation.

Macromodule Packaging

Each macromodule is packaged on a standard electronic module (SEM) (Figure 4) with fixed pin/function assignments on a standard 100-pin connector, to provide consistent compatibility with the µBus, and external analog (ADAC) and digital (SDIO and PDIO) system interfaces.

To achieve low-power low-cost and multi-source semiconductor circuit procurement, each macromodule is supported by standard industry

LSI/MSI circuits. The circuit family favored for the digital modules is the AMD 2900-Series in CMOS-SOS technology. Figure 5 illustrates the family tree of devices and packaging levels identified for the microcomputer family. Raytheon is licensed by AMD to produce the AM 2900 - Series in Schottky-bipolar and is currently developing a CMOS-SOS equivalent of the AM 2901 on company funds. Plans are to extend the CMOS-SOS 2900-Series equivalents to cover the entire family of devices.

Table 1a. Microcomputer Standard Electronic Module (SEM) Microprocessors

SEM	Description	VLSI Circuit Technology	Application
1. μCPU-1	Microprocessor/Central Processing Unit, 8-bit byte general-register, 2 μsec R-R add	N-MOS, CPU-on-a-chip, (MIL 8080)	Telemetry Fuzing Head Control Autopilot
2. μCPU-2	Microprocessor/Central Processing Unit, 8-bit byte, general-register, 600 nsec (8080 Emulator)	CMOS-SOS, bit-slice RALU & μPCU hybrids (2900/3000-series or equiv.)	Autopilot Head control Fuzing
3. μCPU-3	Microprocessor/Central Processing Unit, 16-bit word, fixed-point, general-register, 600 nsec R-R add	CMOS-SOS, bit-slice RALU & μPCU hybrids (2900/3000-series or equiv.)	Autopilot (adaptive)
4. μCPU-4	Microprocessor/Central Unit, 16-bit word, fixed & floating-point, general-register, 600 nsec R-R add (1.0 to 3.25 μsec flt. pt.)	CMOS-SOS, bit-slice RALU & μPCU hybrids (2900/3000-series or equiv.)	Signal processing Estimation Guidance

Table 1b. Microcomputer Standard Electronic Module (SEM) High-Speed Arithmetic and Memories

	SEM	Description	VLSI Circuit Technology	Application
5.	HMPY-1	Hardware multiplier, 200 nsec, 16 x 16-bit multiply	CMOS-SOS single hybrid	Throughput enhancement for μCPU, e.g., Class I Sig. Proc.
6.	μFFT-1	Micro Fast-Fourier Transform processor, 40-400 μsec 64-pts, 8 + J8.	CMOS-SOS or CCD RALU & μPCU hybrids (2900-series or equiv.)	Throughput enhancement for CPUs, e.g., Class II & III Sig. Proc.
7.	RAM-1	Random-access, read/write memory, medium-speed, 128-2K bytes, 500 nsec max, access time	N-MOS DIP/hybrid	Data Telemetry Fuzing
8.	P/ROM-1	Programmable (mask/electrically) read-only memory, medium-speed, 1k-16K bytes, 500 nsec max. access time	N-MOS DIP/hybrid	Programs
9.	RAM-2	Random-access, read/write memory, high-speed, 256-1K x 16-bit or 256-2K bytes, 100 nsec max. access time	CMOS-SOS DIP/hybrid	Data Sig. Proc. Estimation Guidance Head Control Autopilot Fuzing
10.	P/ROM-2	Programmable (mask/electrically) read-only memory, high-speed, 1K-4K x 16-bits or 1K-8K bytes, 100 nsec max. access time	CMOS/SOS DIP/hybrid	Programs

Table 1c. Microcomputer Standard Electronic Module (SEM) Input-Output Modules

SEM		Description	VLSI Circuit Technology	Application
11.	DMAIO	Direct memory access input-output channel, parallel word/byte transfers to/from microcomputer RAM	CMOS-SOS/bipolar single hybrid	All microprocessor applications
12.	PDIO	Parallel digital input-output channel, parallel discrete transfers to/from μCPU	CMOS-SOS/bipolar single hybrid	Telemetry
13.	ADAC	Analog to digital/digital to analog input-output channel A-D: 8/16/24 Chs., Sim. S/H, Mux, 8/10/12-bit, A-D 3/6/8 μsec max/Ch. D-A: 8 Chs. Demux., S/H, 12-bit D-A, 5 μsec max/Ch.	CMOS-SOS single hybrid	Head control Autopilot Telemetry Radar receiver
14.	SDIO	Serial digital input-output channel word & bit serial data/command transfers, imbit/sec max, MIL-STD-1553A	CMOS-SOS single hybrid	Avionics Inter microcomputer

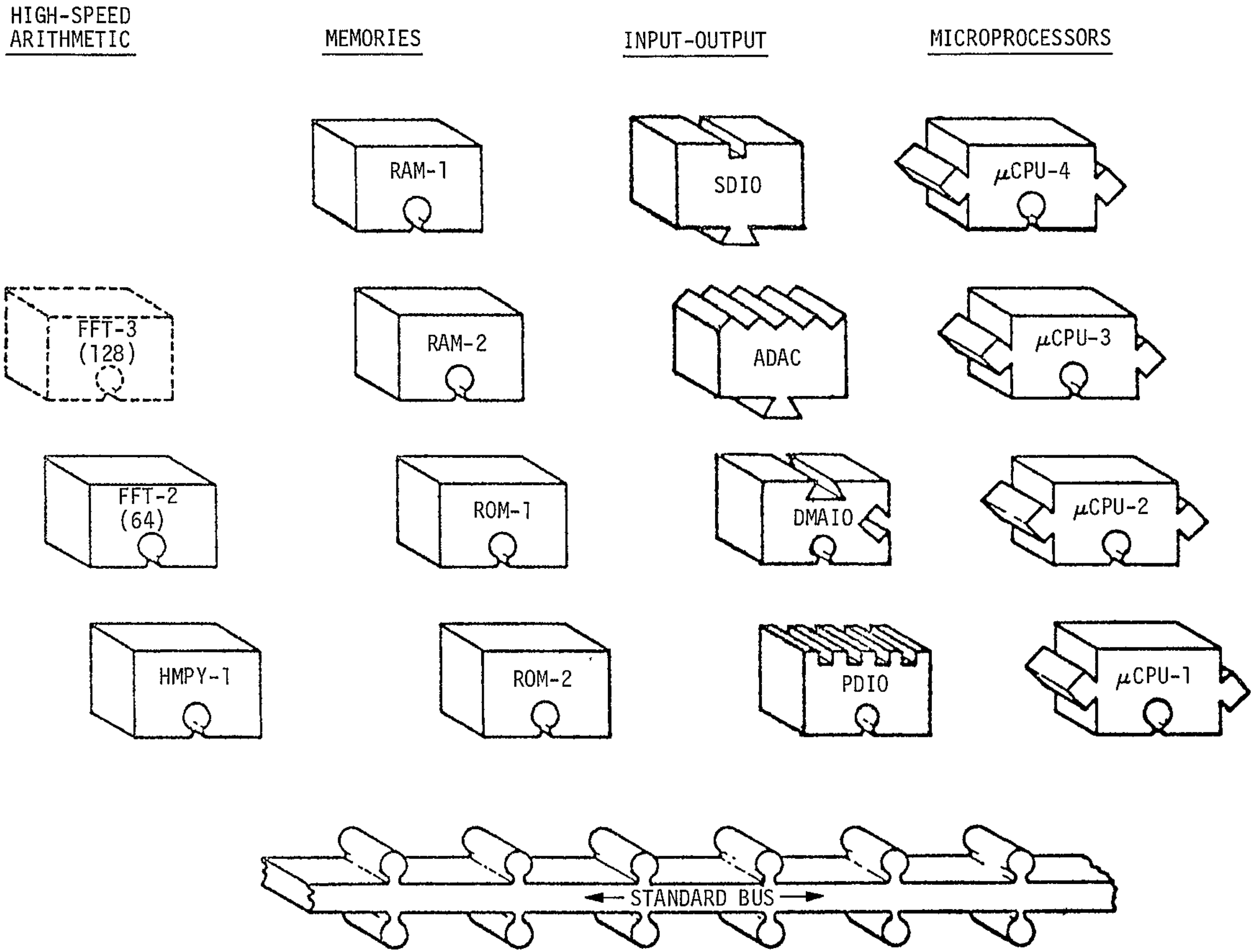


Figure 1. Microcomputer Macromodules

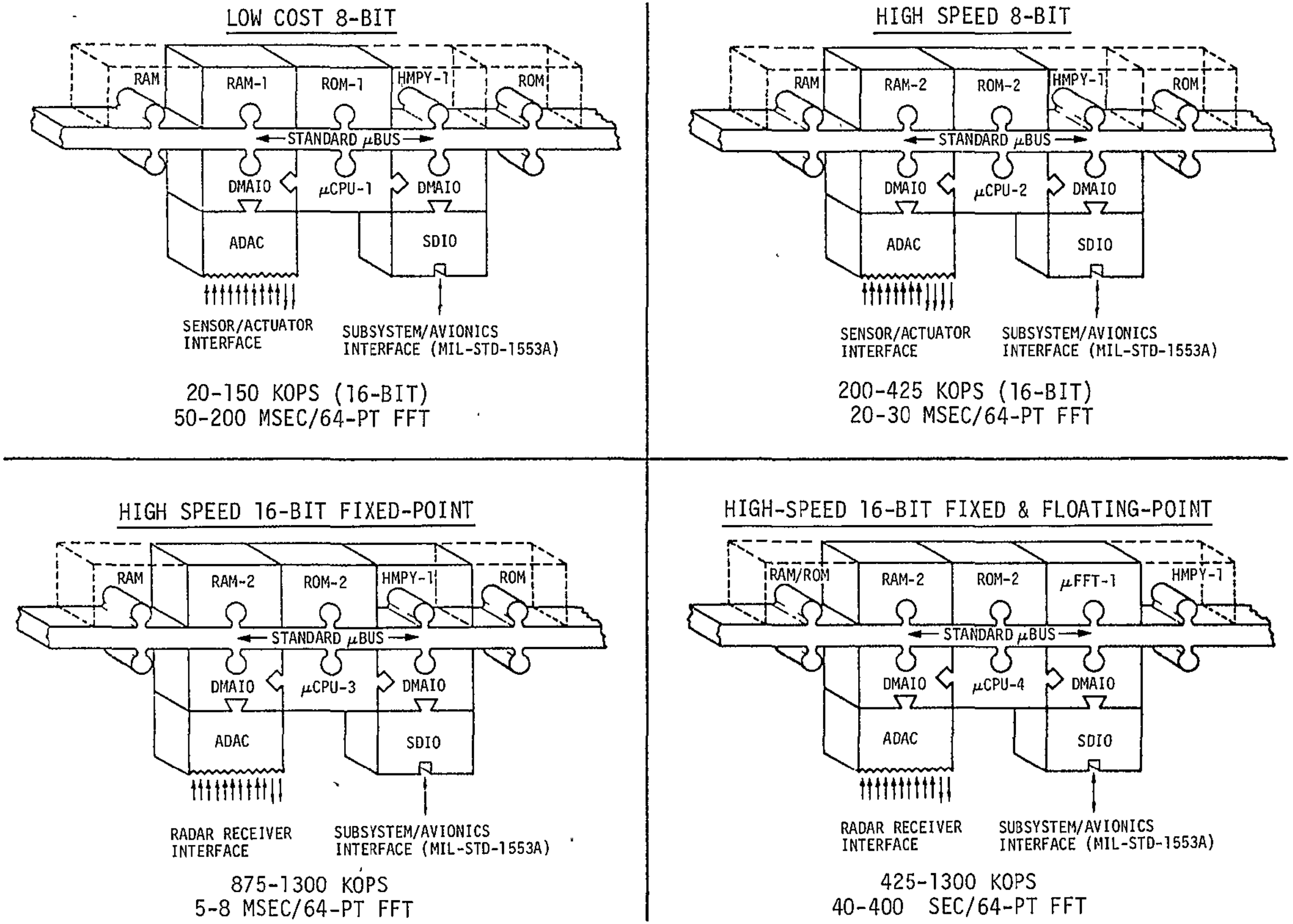


Figure 2. Macromodular Microcomputer Family

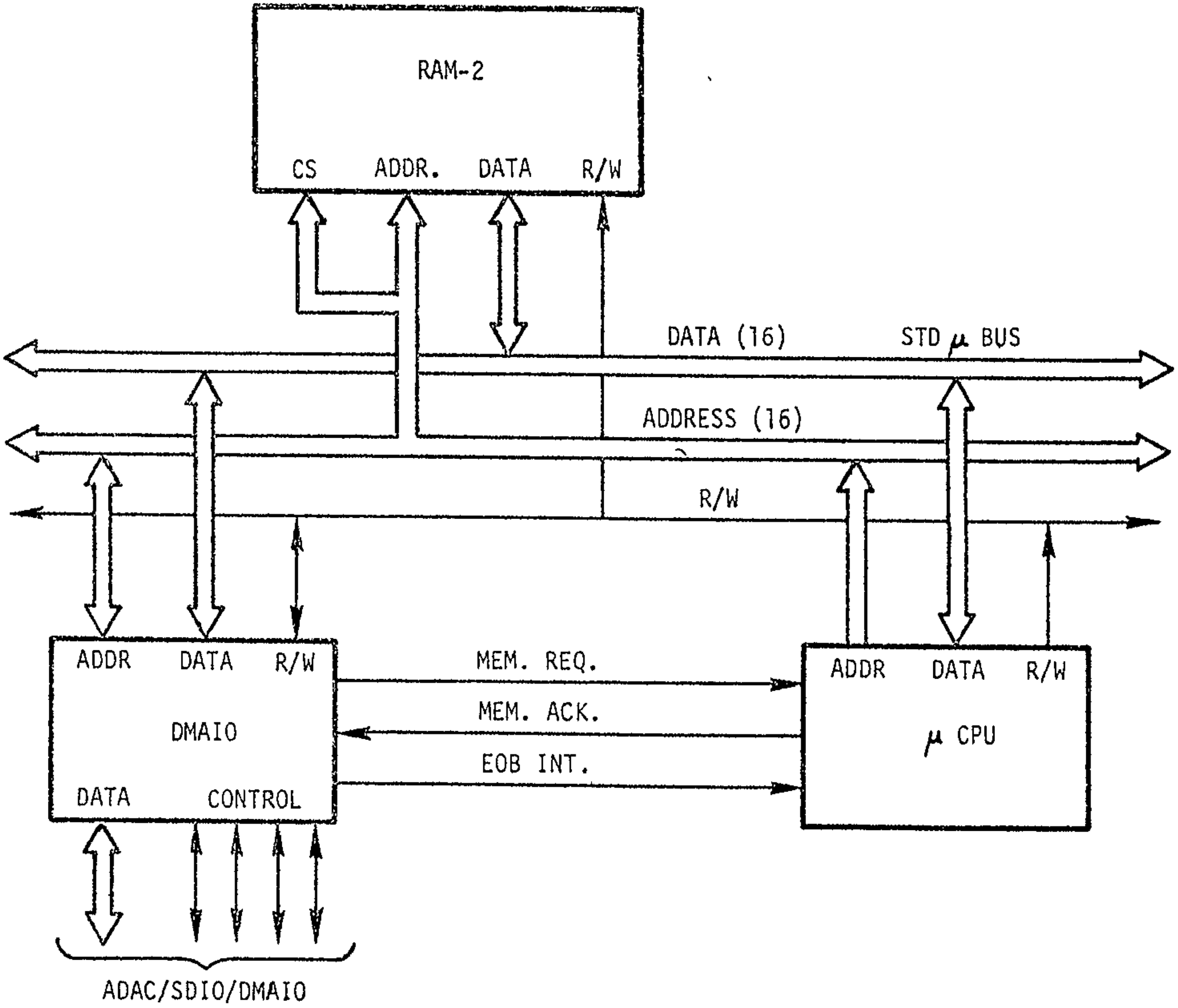


Figure 3. Standard Macromodule Interfaces

ORIGINAL PAGE IS
OF POOR QUALITY

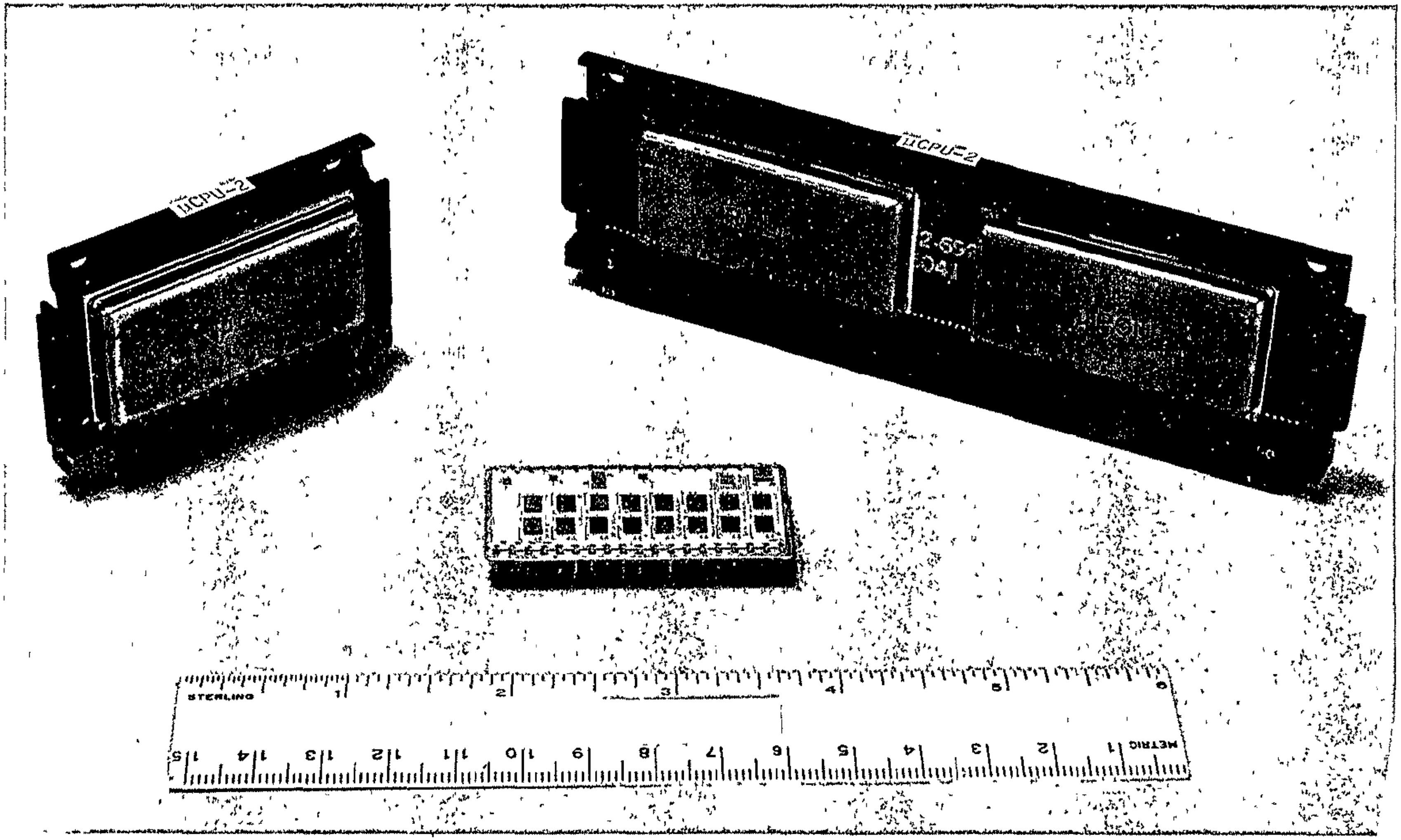


Figure 4. Standard Electronic Module

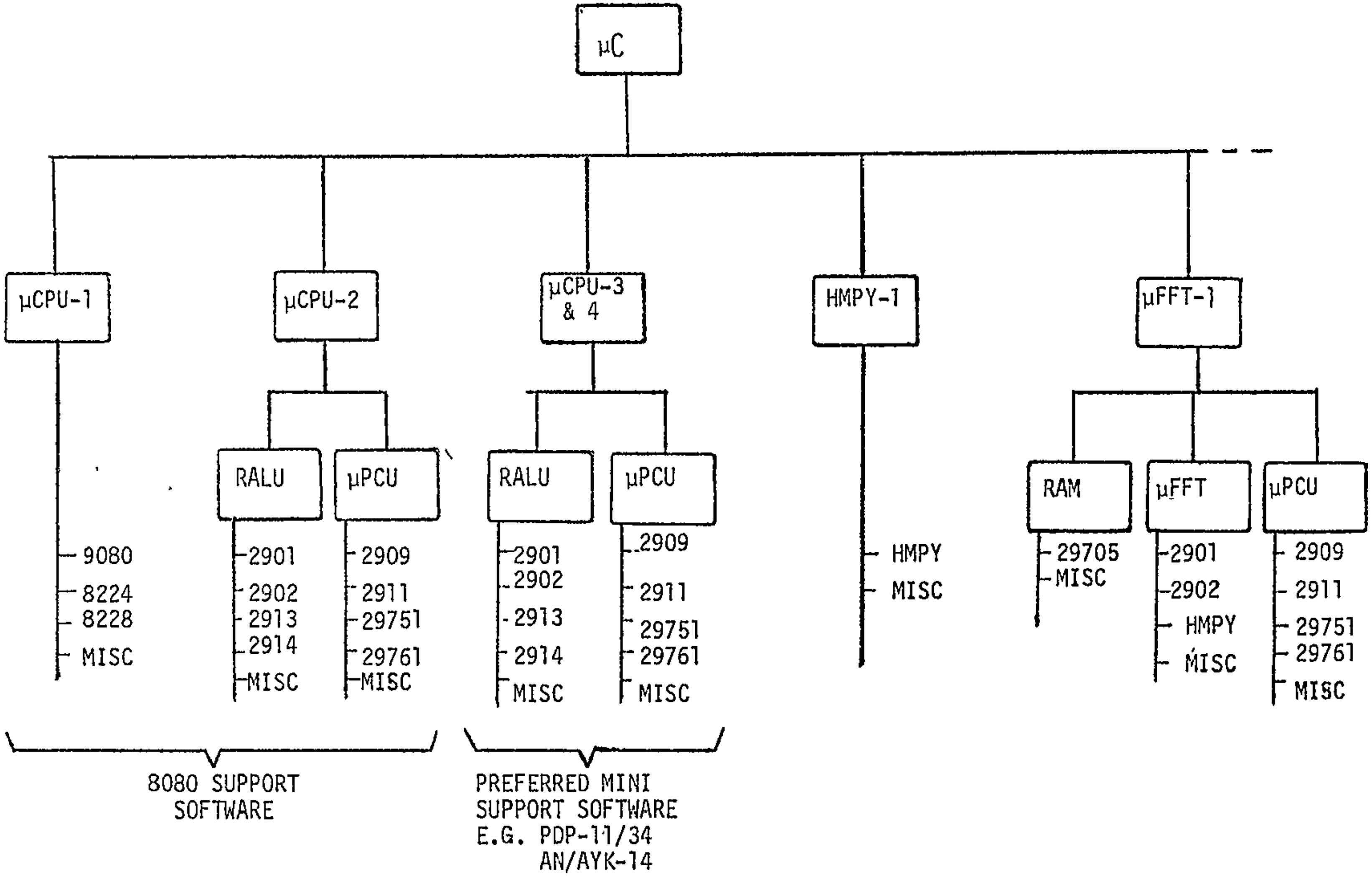


Figure 5. Macromodular Microcomputer Famıly Tree

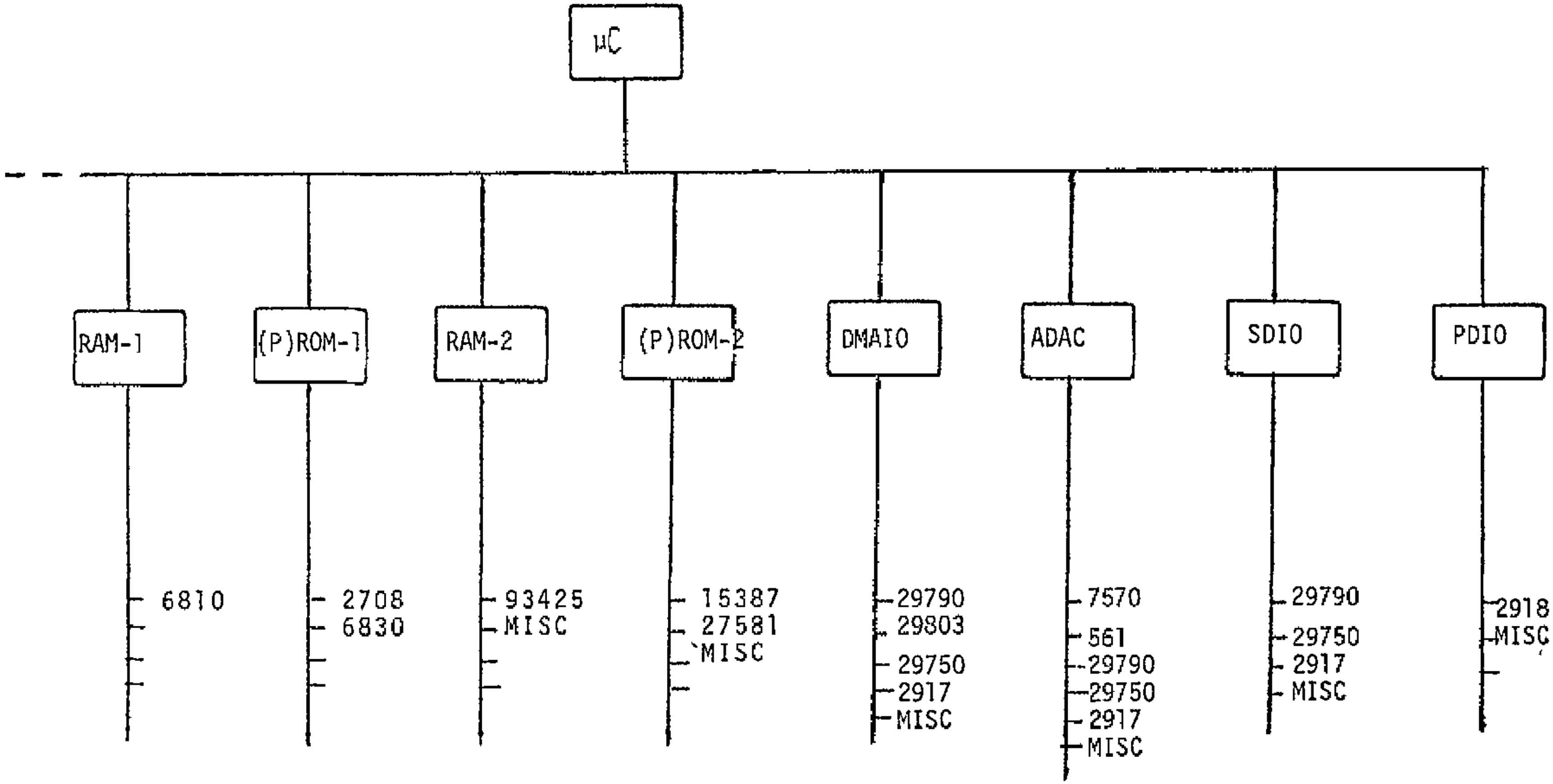


Figure 5. Macromodular Microcomputer Family Tree (continued)

IMPLICATIONS OF HI-REL SPECIFICATIONS ON THE INTEL 8080 MICROPROCESSOR

Hank Malloy
Intel Corporation
Santa Clara, California

(Paper Not Available)

77-39

2901 BIT SLICE MICROPROCESSOR FAMILY

John Springer
Advanced Micro Devices, Inc.
Sunnyvale, California

(Paper Not Available)

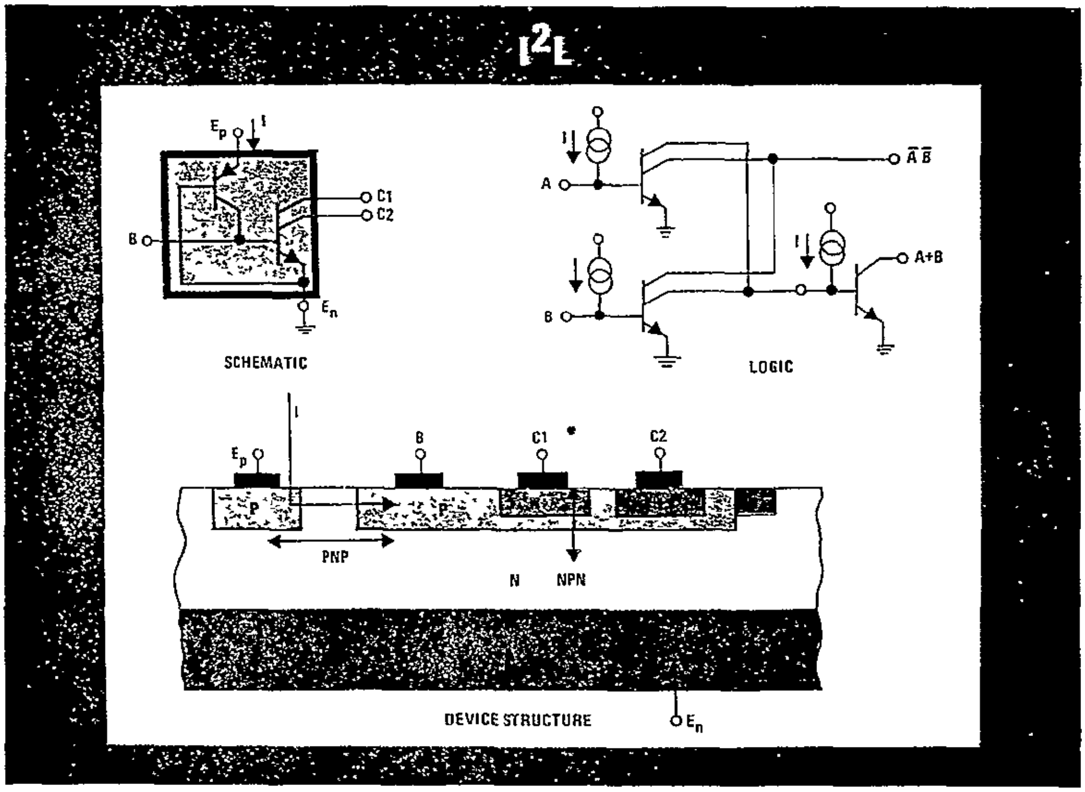
THE TEXAS INSTRUMENTS I^2L SBP9900 MICROPROCESSOR

Ben Sloan
Texas Instruments, Inc.
Dallas, Texas

(No Text Available)

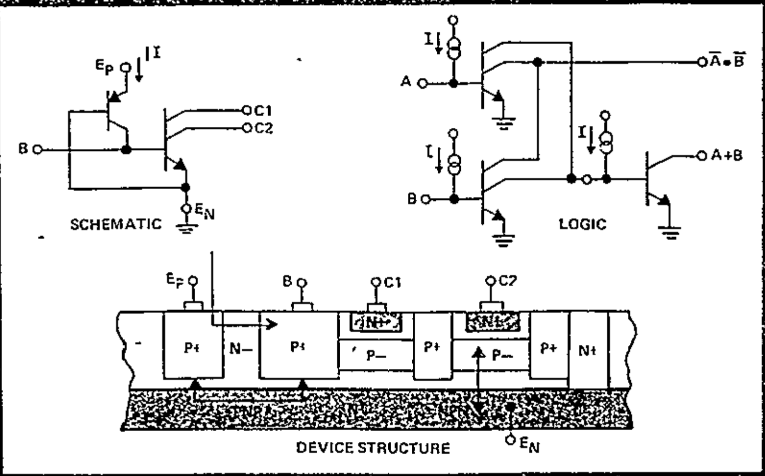
INTEGRATED
INJECTION
LOGIC

I^2L TODAY

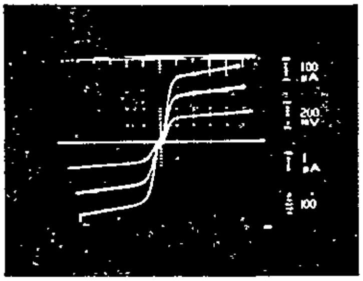


I^2L

ADVANCED I²L



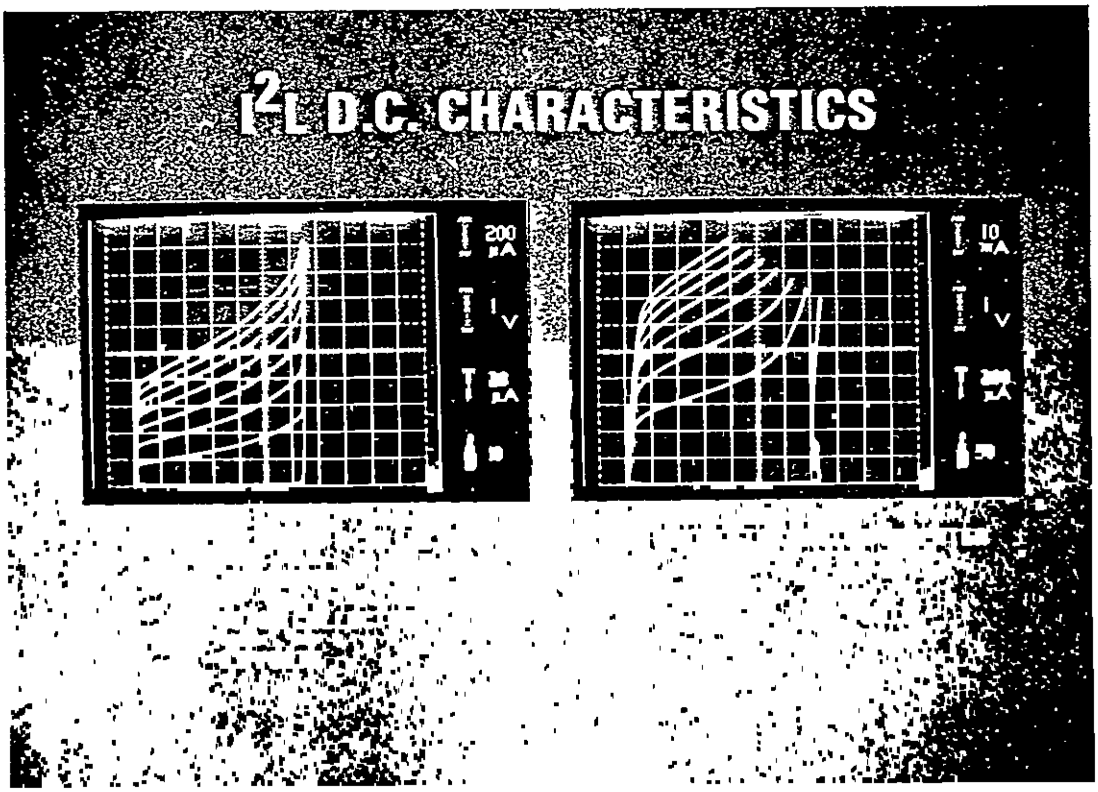
I²L D.C. CHARACTERISTICS



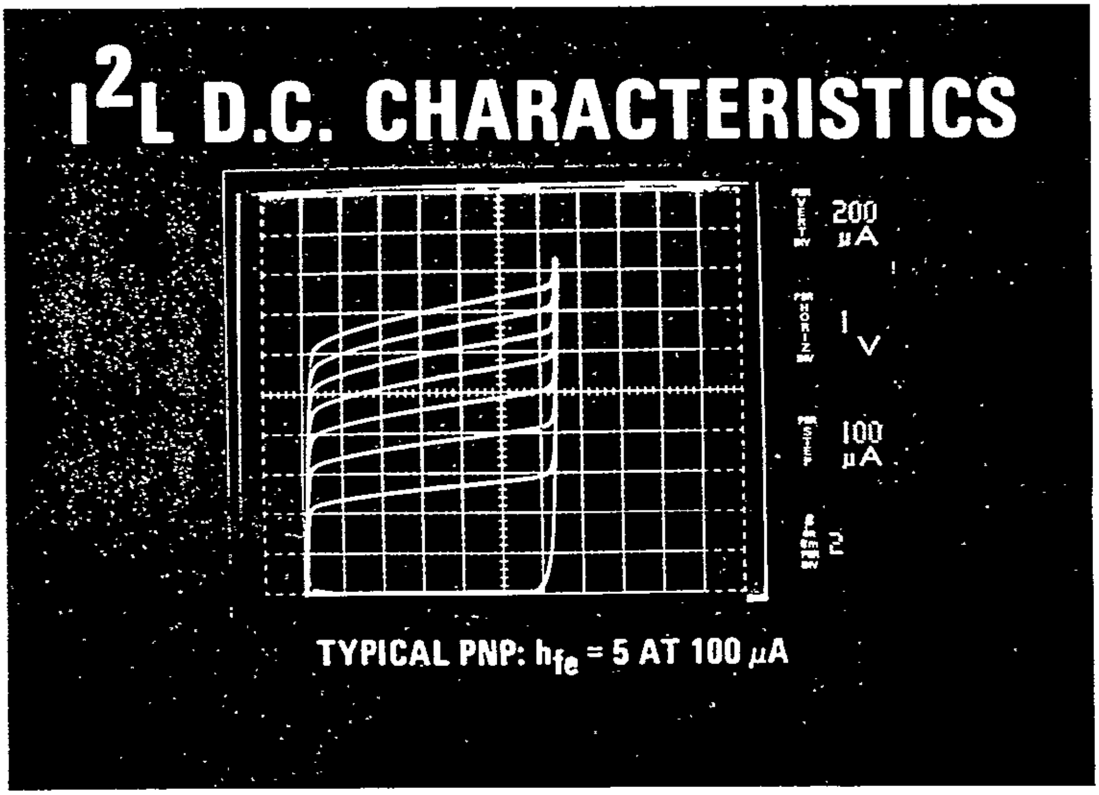
FORWARD CHARACTERISTIC

INVERSE (I²L) CHARACTERISTIC

INTRINSIC I²L NPN TRANSISTOR
β UP = β DOWN = 150

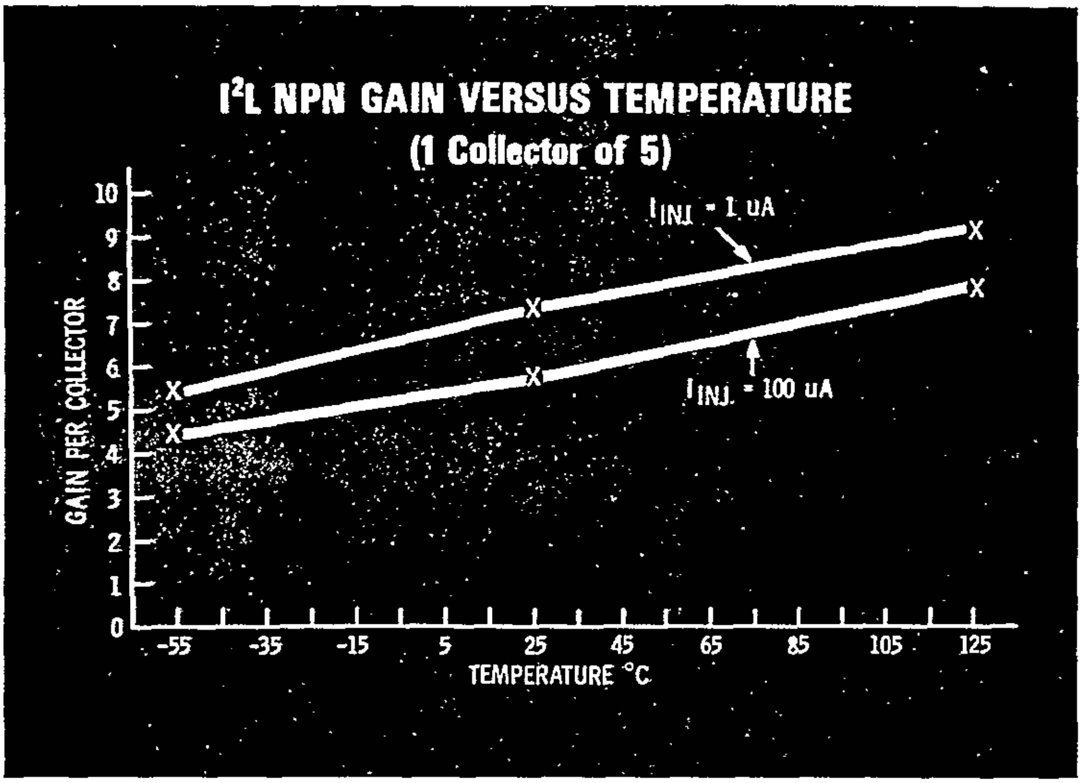


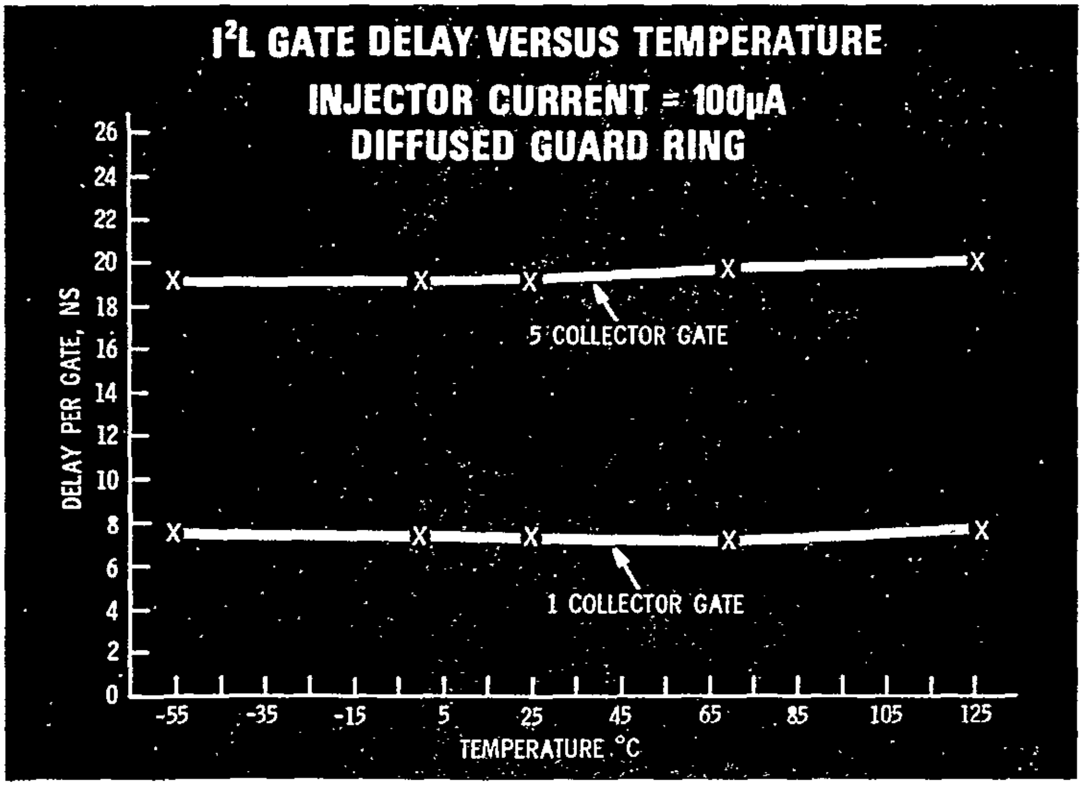
I^{2}L D.C. CHARACTERISTICS

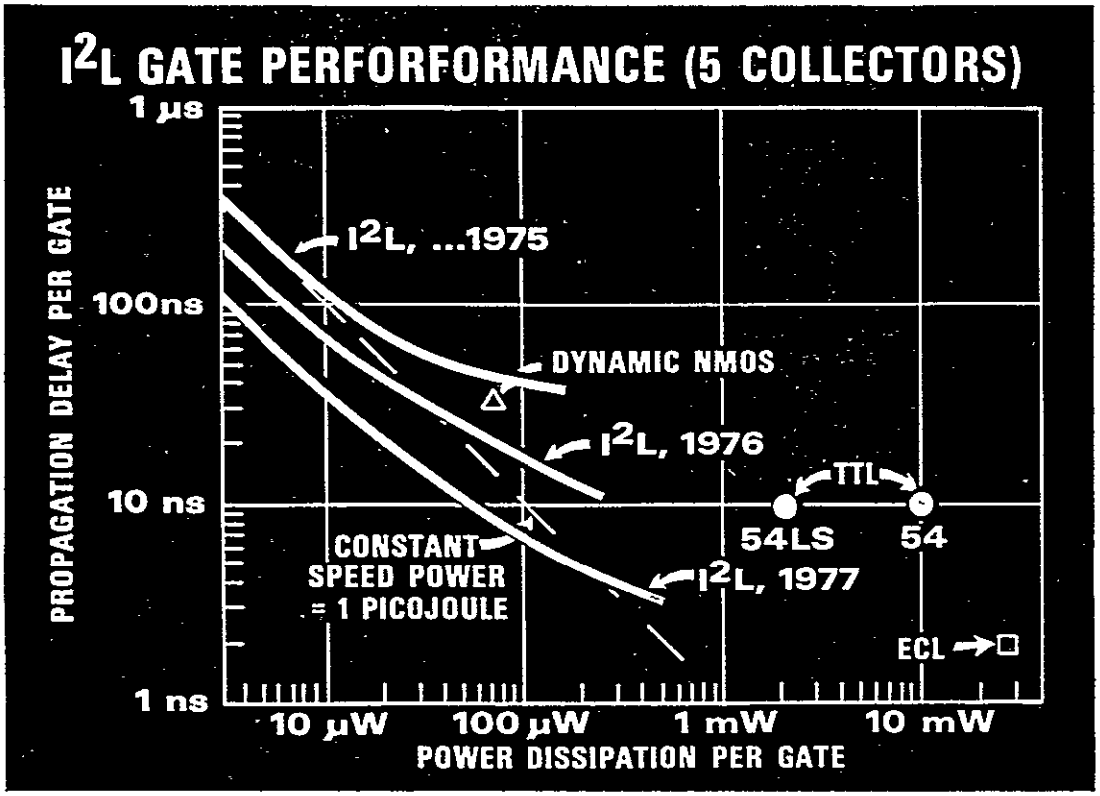


I^{2}L D.C. CHARACTERISTICS

TYPICAL PNP: h_{fe} = 5 AT 100 µA

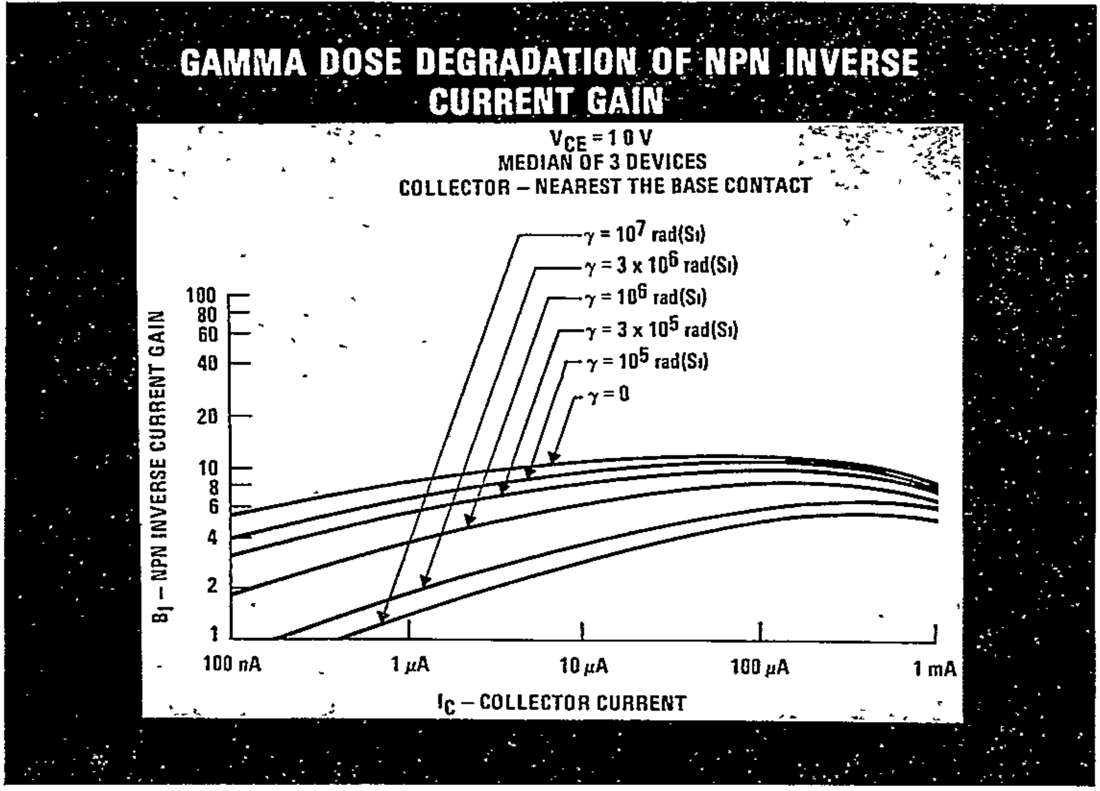

I²L NPN GAIN VERSUS TEMPERATURE
(1 Collector of 5)
GAIN PER COLLECTOR
0
1
2
3
4
5
6
7
8
9
10
I_INJ = 1 uA
I_INJ = 100 uA
-55
-35
-15
5
25
45
65
85
105
125
TEMPERATURE °C


I²L GATE DELAY VERSUS TEMPERATURE
INJECTOR CURRENT = 100µA
DIFFUSED GUARD RING
DELAY PER GATE, NS
0
2
4
6
8
10
12
14
16
18
20
22
24
26
5 COLLECTOR GATE
1 COLLECTOR GATE
-55
-35
-15
5
25
45
65
85
105
125
TEMPERATURE °C

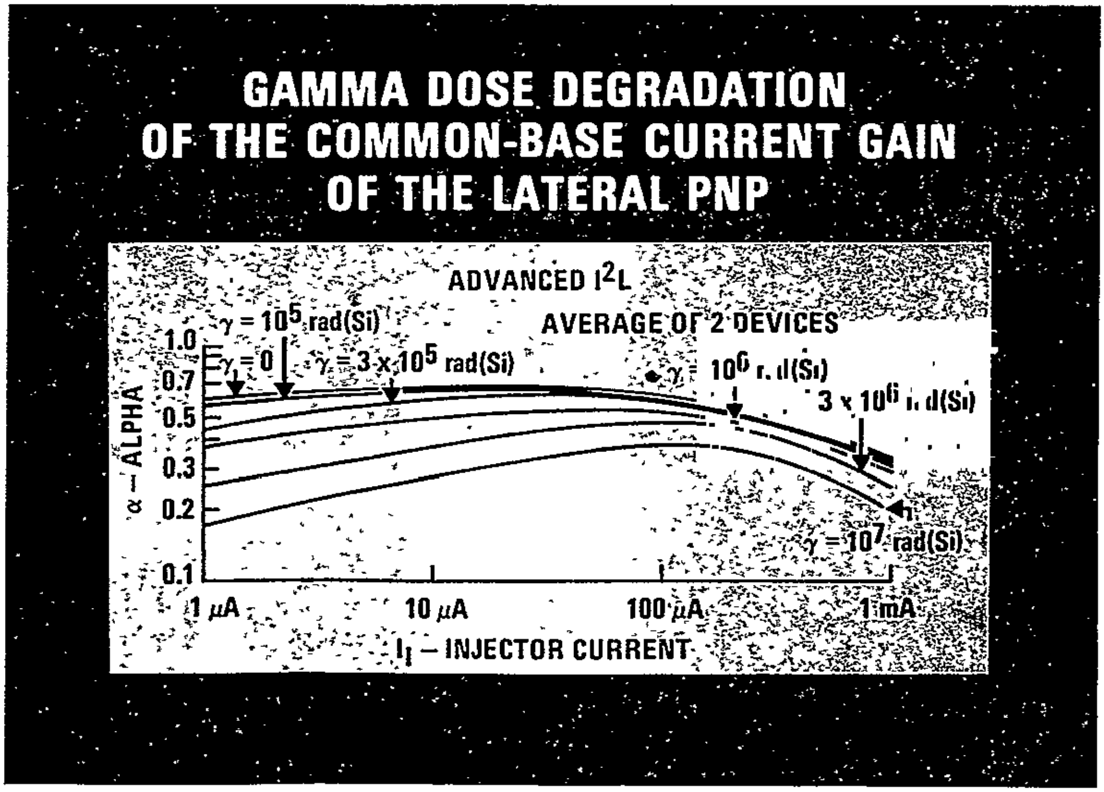


I^2L GATE PERFORFORMANCE (5 COLLECTORS)

RADIATION EFFECTS

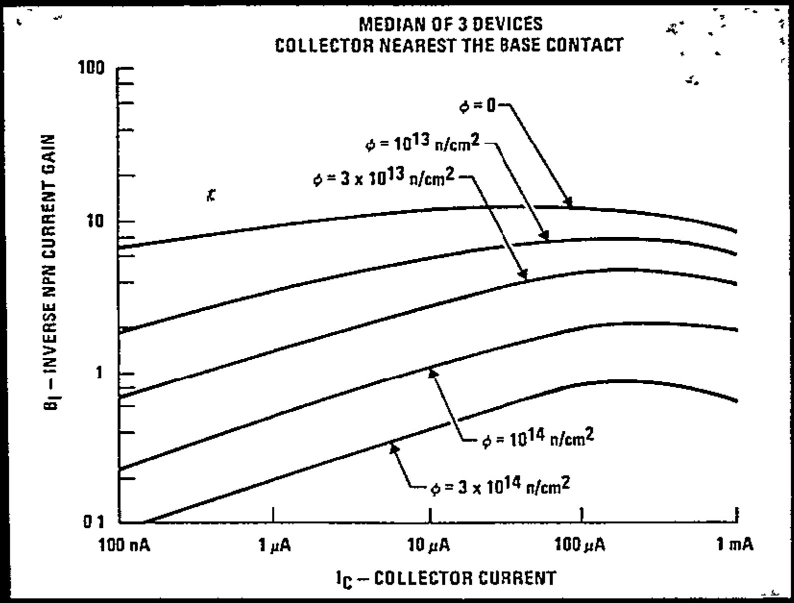


GAMMA DOSE DEGRADATION OF NPN INVERSE CURRENT GAIN

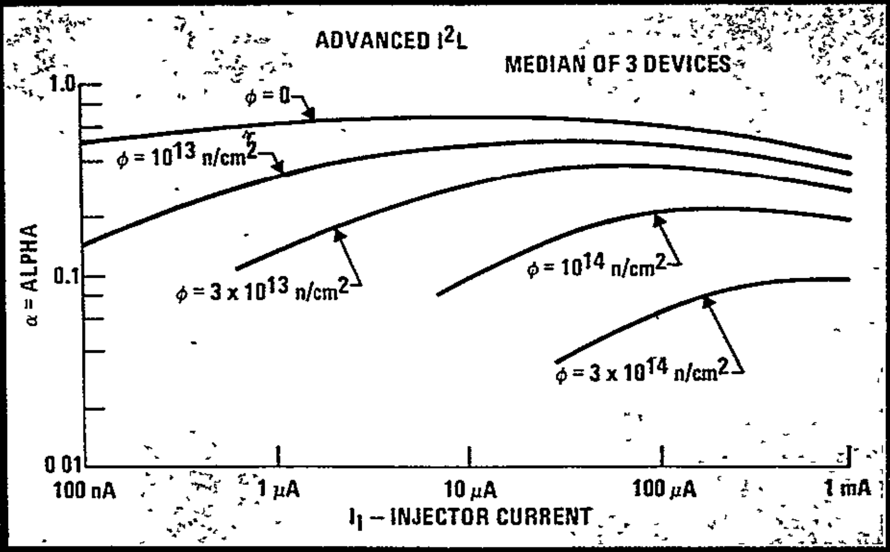


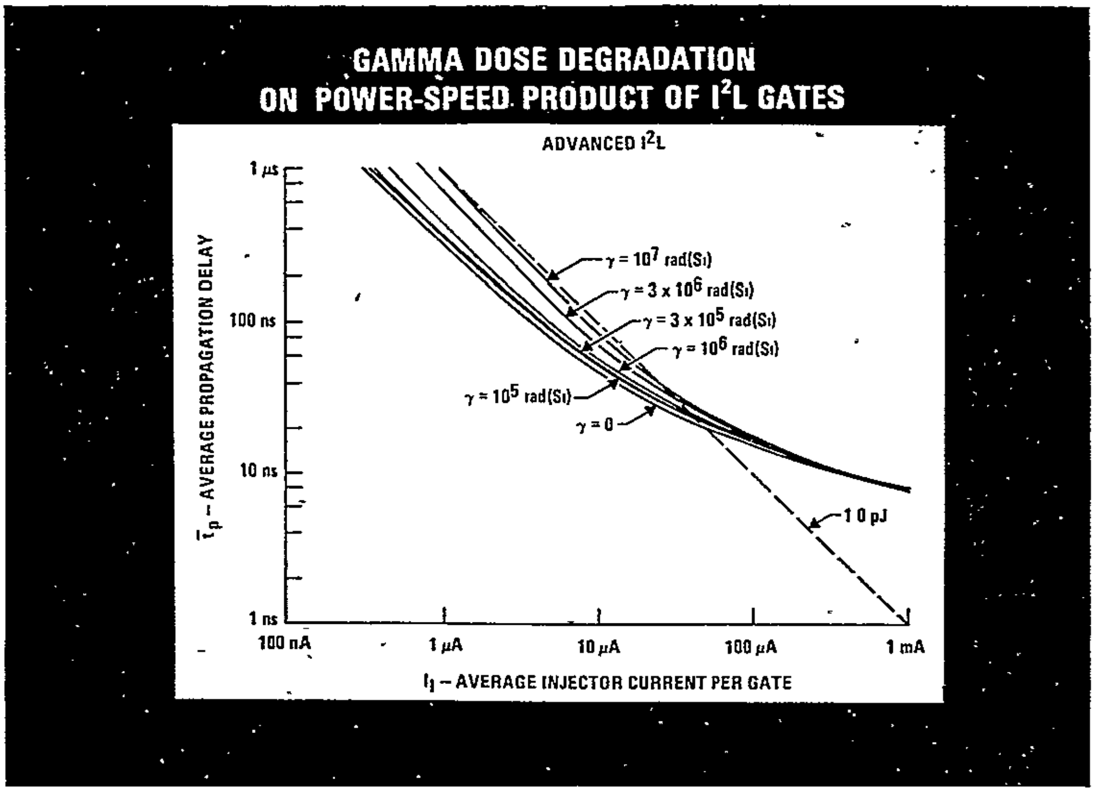
GAMMA DOSE DEGRADATION OF THE COMMON-BASE CURRENT GAIN OF THE LATERAL PNP

NEUTRON DAMAGE ON THE INVERSE NPN TRANSISTOR

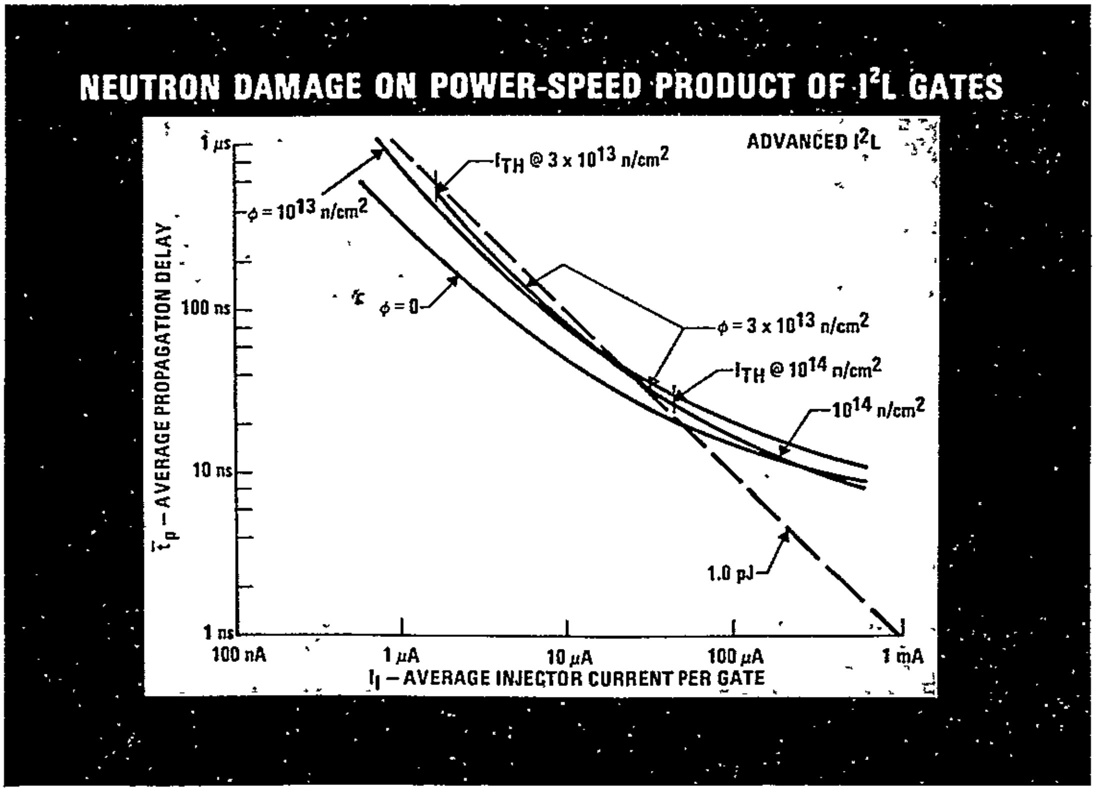


NEUTRON DAMAGE ON THE COMMON-BASE CURRENT GAIN OF THE LATERAL PNP





GAMMA DOSE DEGRADATION ON POWER-SPEED PRODUCT OF I²L GATES



NEUTRON DAMAGE ON POWER-SPEED PRODUCT OF I²L GATES

RELIABILITY

I^{2}L RELIABILITY, TESTING TO DATE

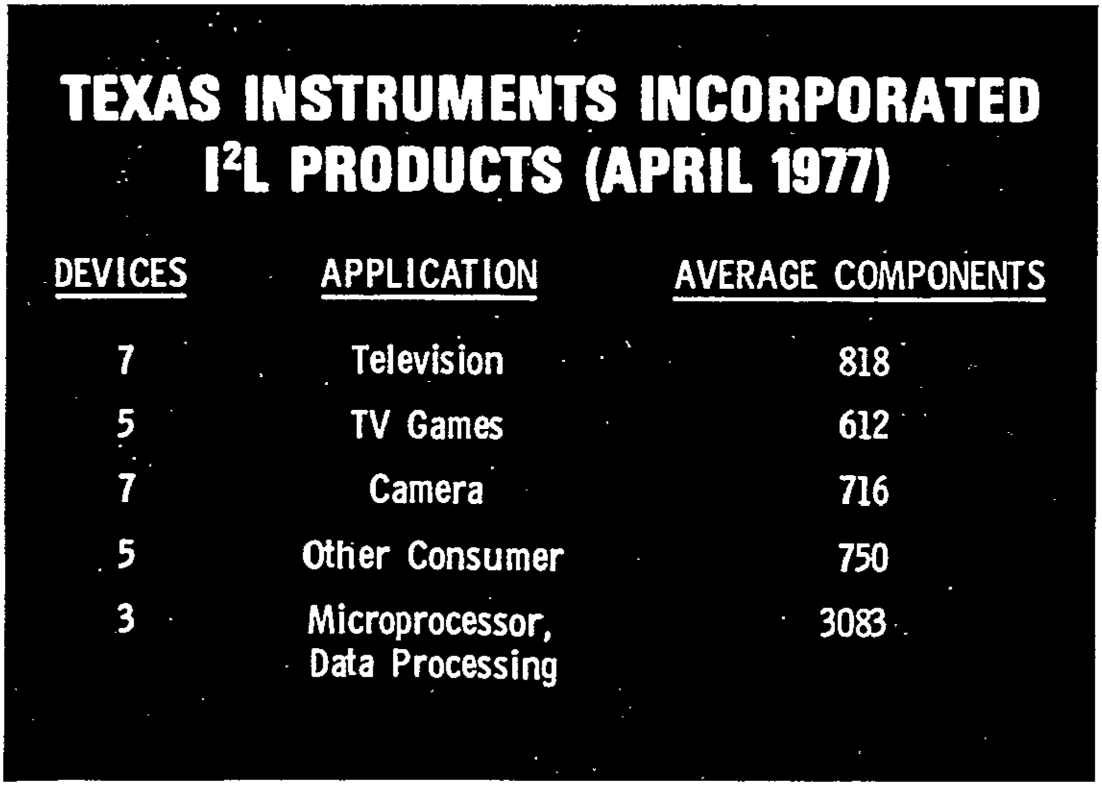
TI I^{2}L WATCH MODULE IC (750 GATE COMPLEXITY)

- EQUIVALENT TESTING 5 x 10^9 HOURS
- NUMBER OF FAILURES. 4
- FAILURE RATE. 0 00018% PER 1000 HOURS (95% CONFIDENCE)

SBP9900 I^{2}L MICROPROCESSOR (6300 GATE COMPLEXITY)

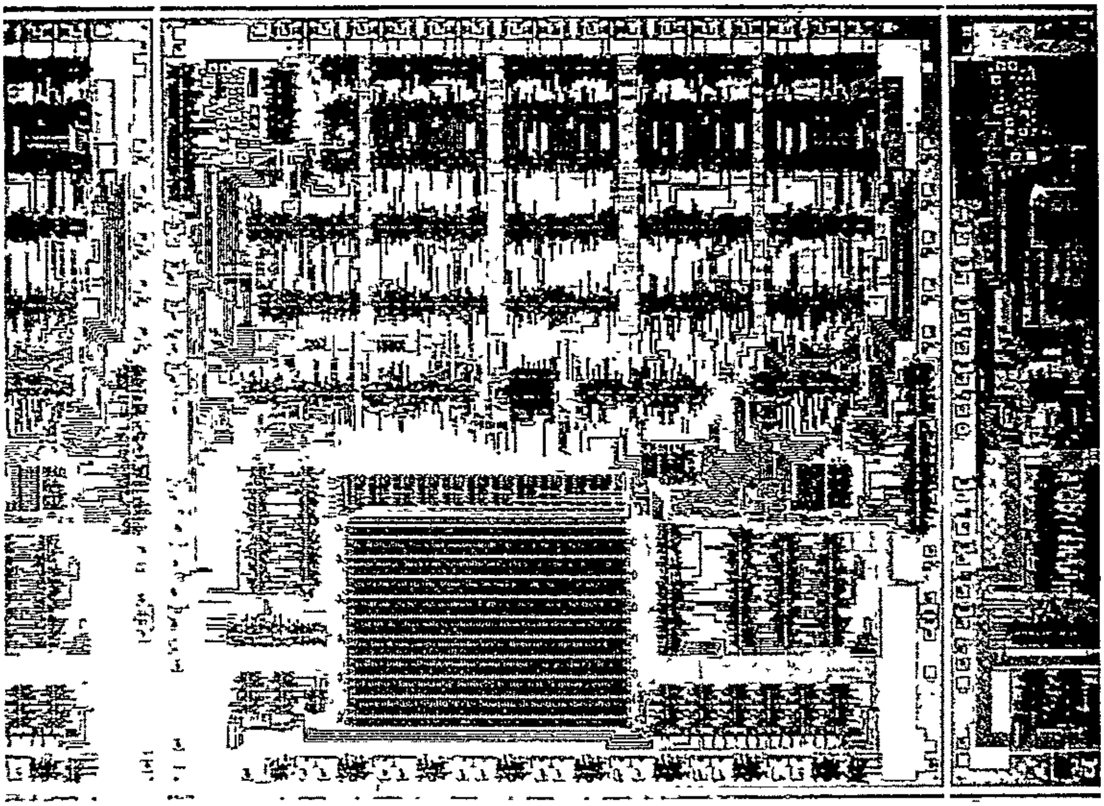
- 20 UNITS PASSED 168 HOURS, 125° C OPERATING LIFE WITH NO FAILURES
- 10 UNITS PASSED 168 HOURS, 85° C OPERATING LIFE WITH NO FAILURES
- 20 UNITS PASSED 168 HOURS, 150° C STORAGE LIFE WITH NO FAILURES
- JAN SPECIFICATION SHEET TO BE SUBMITTED JUNE-JULY, FORECAST SHIPPING JAN PRODUCT BY YEAR END

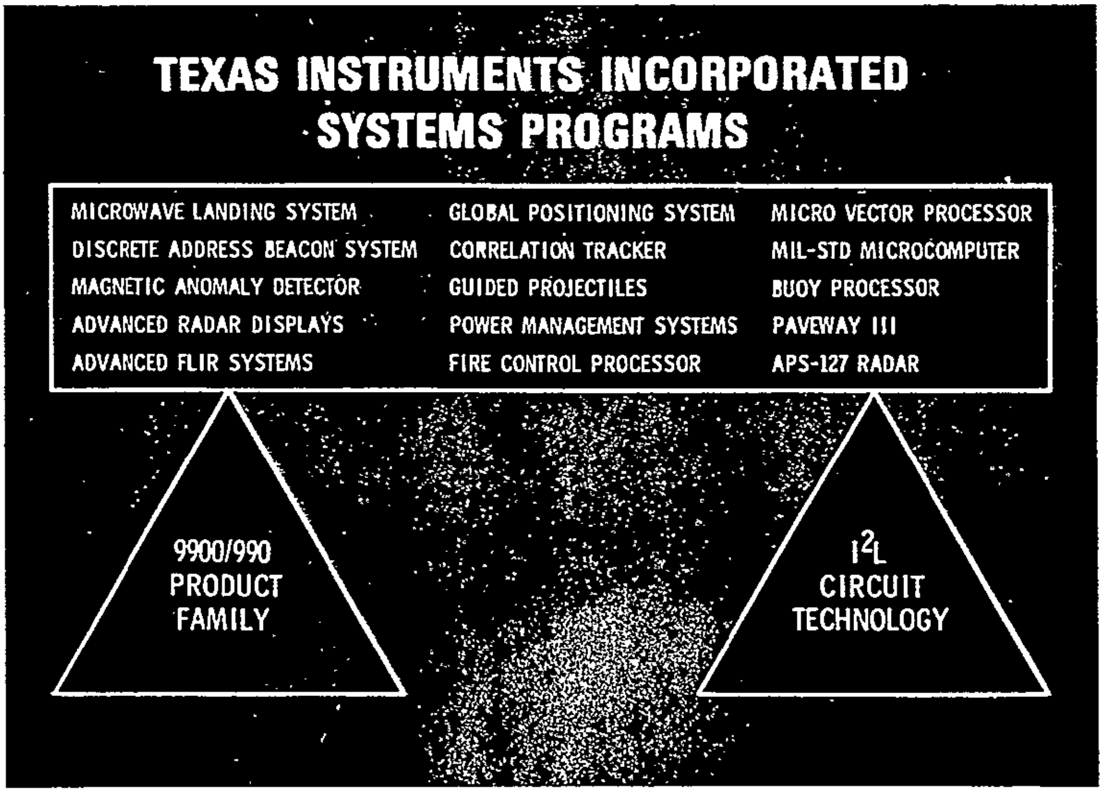
PRODUCTS APPLICATIONS



TEXAS INSTRUMENTS INCORPORATED I^2L PRODUCTS (APRIL 1977)

DEVICES	APPLICATION	AVERAGE COMPONENTS
7	Television	818
5	TV Games	612
7	Camera	716
5	Other Consumer	750
3	Microprocessor, Data Processing	3083





TEXAS INSTRUMENTS INCORPORATED
SYSTEMS PROGRAMS

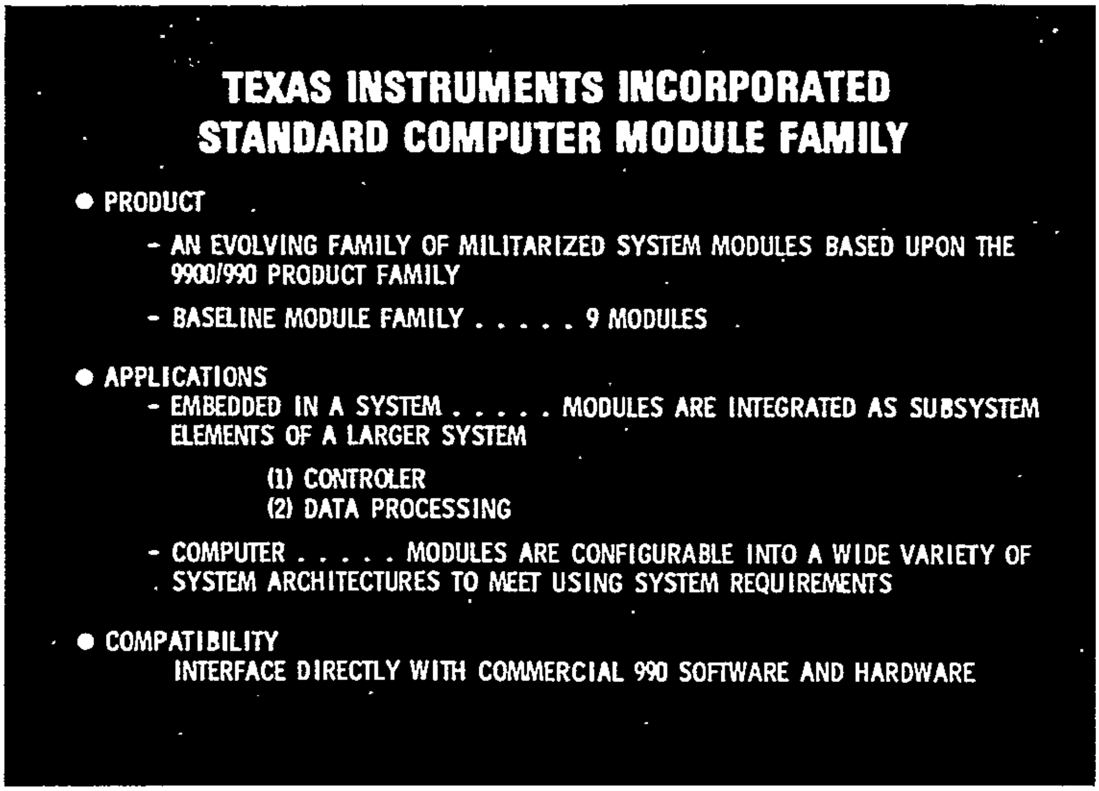
MICROWAVE LANDING SYSTEM
DISCRETE ADDRESS BEACON SYSTEM
MAGNETIC ANOMALY DETECTOR
ADVANCED RADAR DISPLAYS
ADVANCED FLIR SYSTEMS

GLOBAL POSITIONING SYSTEM
CORRELATION TRACKER
GUIDED PROJECTILES
POWER MANAGEMENT SYSTEMS
FIRE CONTROL PROCESSOR

MICRO VECTOR PROCESSOR
MIL-STD MICROCOMPUTER
BUOY PROCESSOR
PAVEWAY III
APS-127 RADAR

9900/990
PRODUCT
FAMILY

I^2L
CIRCUIT
TECHNOLOGY

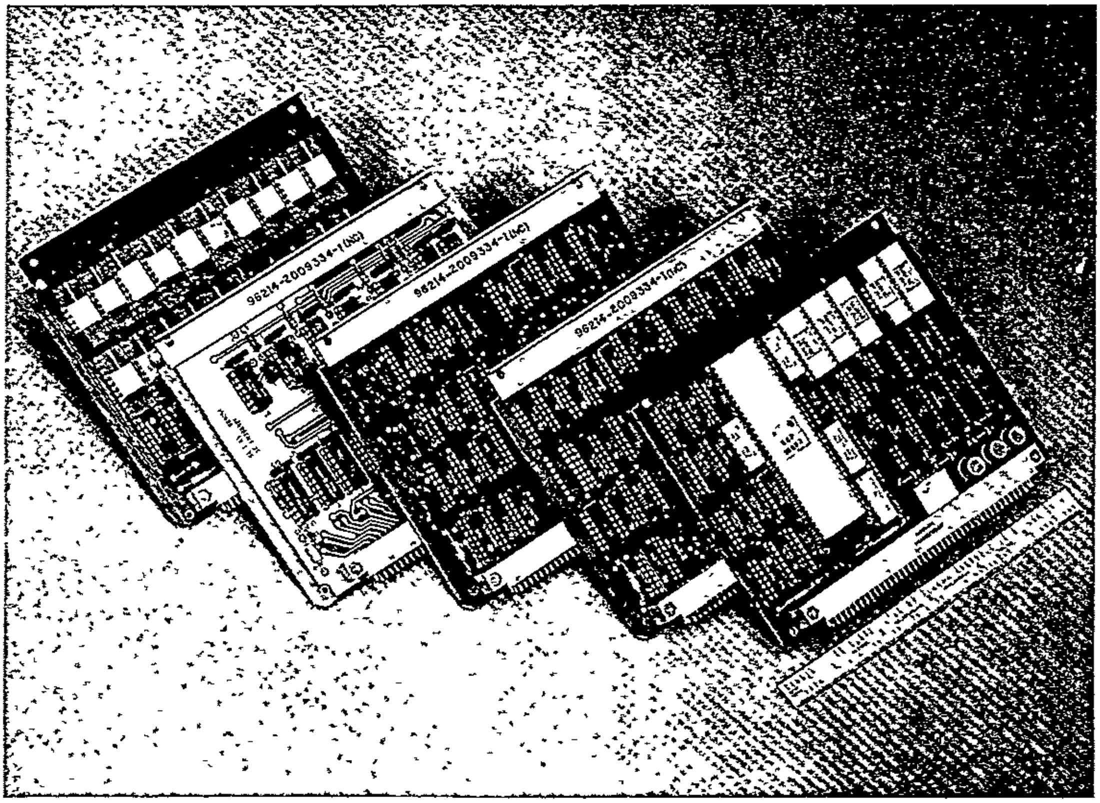


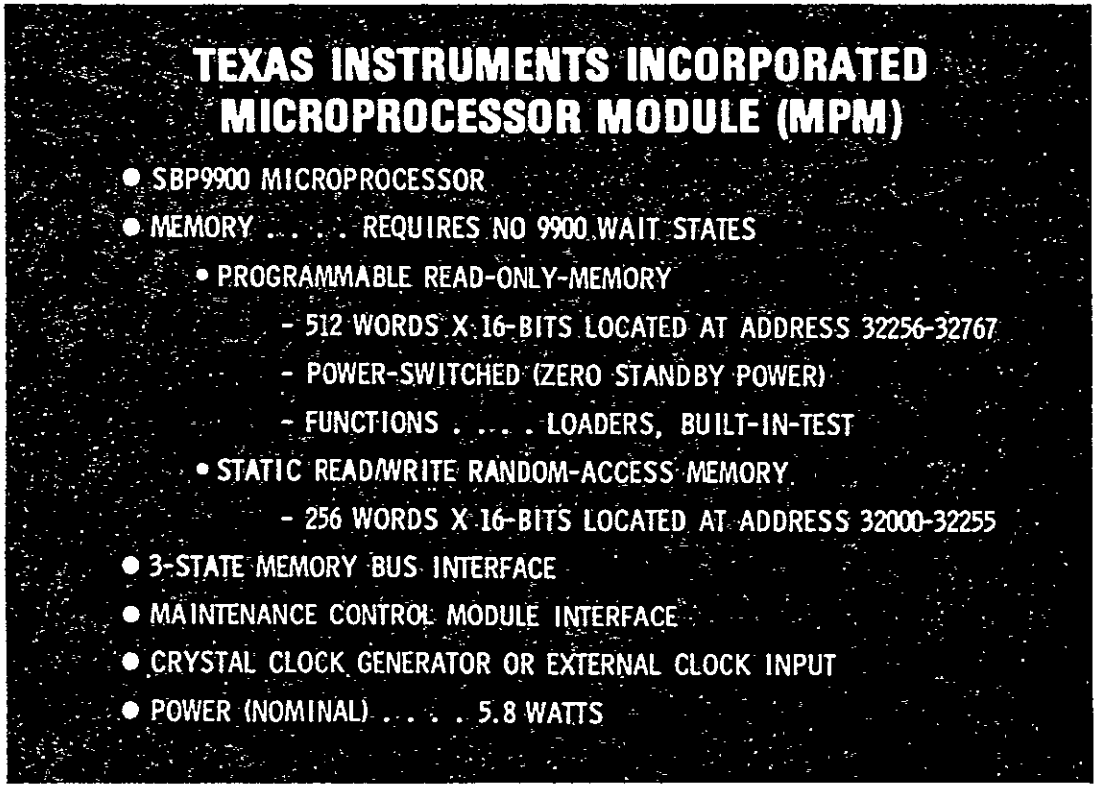
TEXAS INSTRUMENTS INCORPORATED
STANDARD COMPUTER MODULE FAMILY

- PRODUCT
  - AN EVOLVING FAMILY OF MILITARIZED SYSTEM MODULES BASED UPON THE 9900/990 PRODUCT FAMILY
  - BASELINE MODULE FAMILY . . . . . 9 MODULES
- APPLICATIONS
  - EMBEDDED IN A SYSTEM . . . . . MODULES ARE INTEGRATED AS SUBSYSTEM ELEMENTS OF A LARGER SYSTEM

    (1) CONTROLER
    (2) DATA PROCESSING
  - COMPUTER . . . . . MODULES ARE CONFIGURABLE INTO A WIDE VARIETY OF SYSTEM ARCHITECTURES TO MEET USING SYSTEM REQUIREMENTS
- COMPATIBILITY

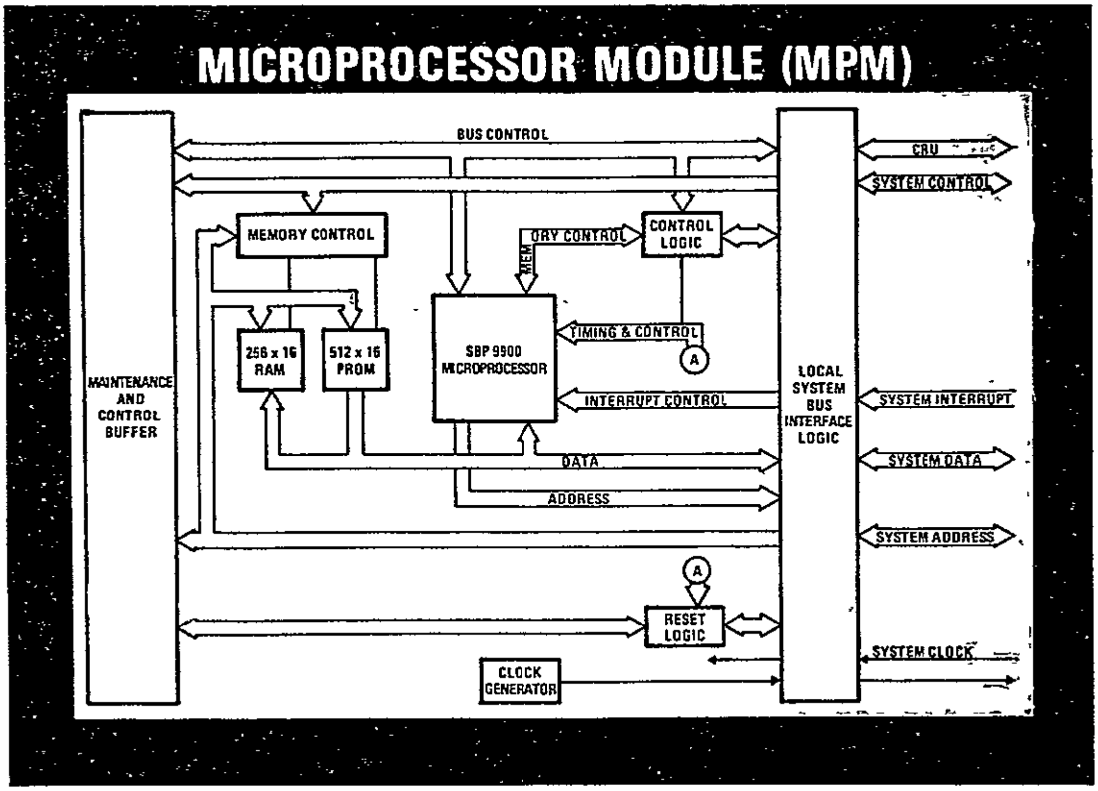
  INTERFACE DIRECTLY WITH COMMERCIAL 990 SOFTWARE AND HARDWARE





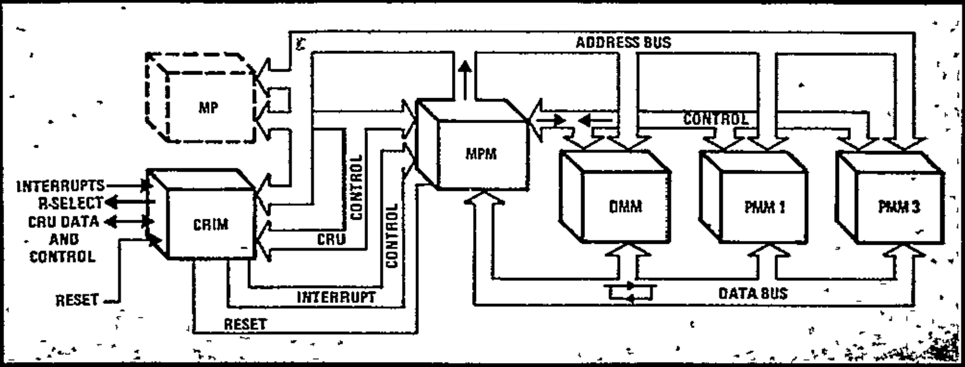
TEXAS INSTRUMENTS INCORPORATED MICROPROCESSOR MODULE (MPM)

- SBP9900 MICROPROCESSOR
- MEMORY . . . . . REQUIRES NO 9900 WAIT STATES
  - PROGRAMMABLE READ-ONLY-MEMORY
    - 512 WORDS X 16-BITS LOCATED AT ADDRESS 32256-32767
    - POWER-SWITCHED (ZERO STANDBY POWER)
    - FUNCTIONS . . . . . LOADERS, BUILT-IN-TEST
  - STATIC READ/WRITE RANDOM-ACCESS MEMORY
    - 256 WORDS X 16-BITS LOCATED AT ADDRESS 32000-32255
- 3-STATE MEMORY BUS INTERFACE
- MAINTENANCE CONTROL MODULE INTERFACE
- CRYSTAL CLOCK GENERATOR OR EXTERNAL CLOCK INPUT
- POWER (NOMINAL) . . . . . 5.8 WATTS

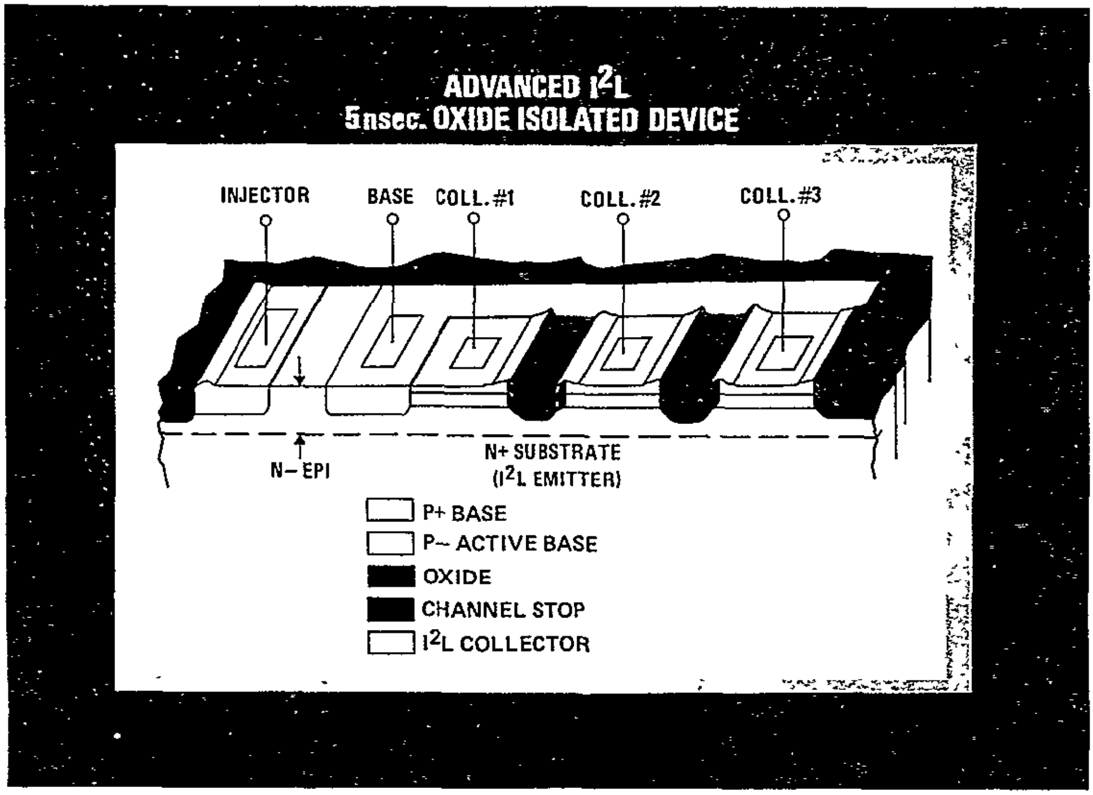


MICROPROCESSOR MODULE (MPM)

GPS MAN-PACK/VEHICULAR DATA PROCESSOR UNIT FUNCTIONAL ORGANIZATION

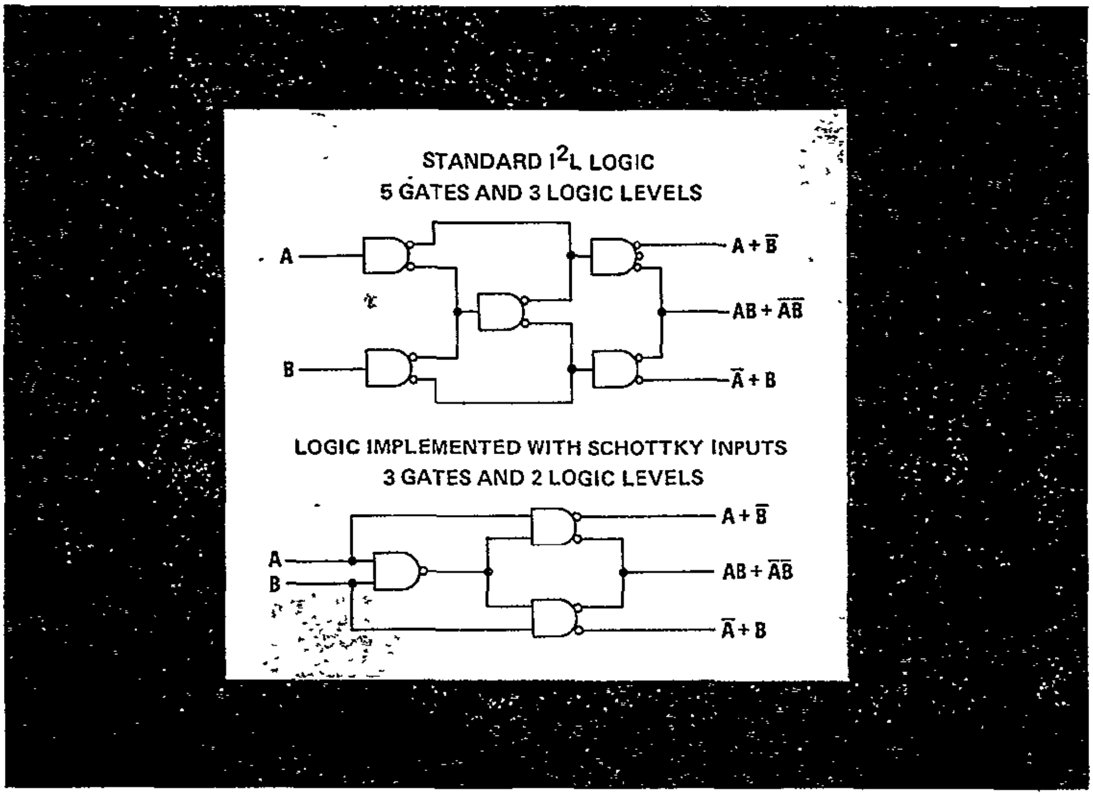


I^{2}L - THE FUTURE



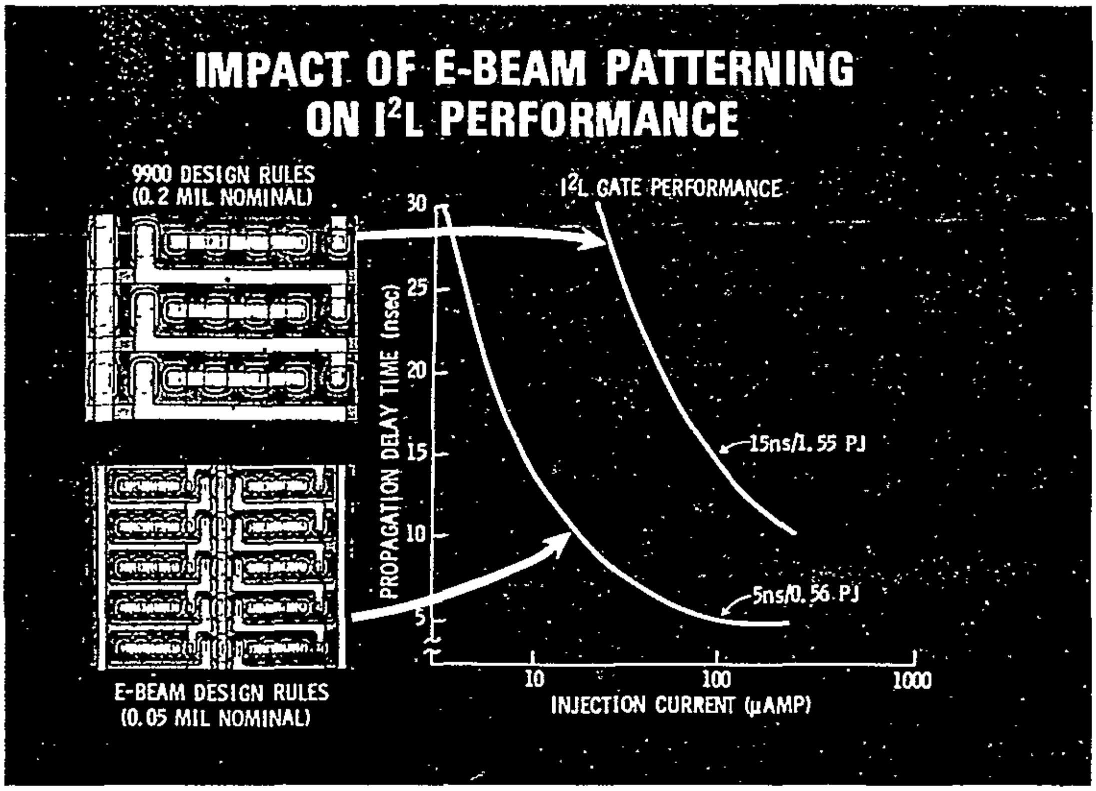
ADVANCED I²L
5nsec. OXIDE ISOLATED DEVICE

□ P+ BASE
□ P– ACTIVE BASE
■ OXIDE
■ CHANNEL STOP
□ I²L COLLECTOR

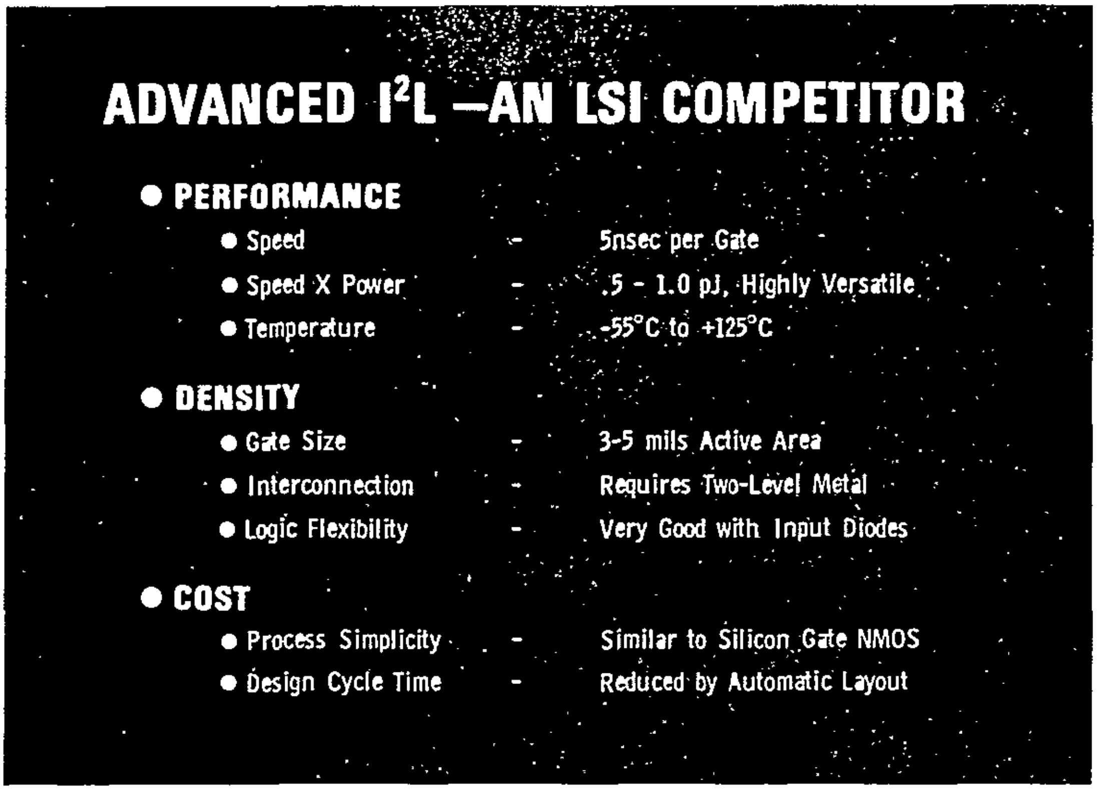


STANDARD I²L LOGIC
5 GATES AND 3 LOGIC LEVELS

LOGIC IMPLEMENTED WITH SCHOTTKY INPUTS
3 GATES AND 2 LOGIC LEVELS



IMPACT OF E-BEAM PATTERNING ON I²L PERFORMANCE



ADVANCED I²L —AN LSI COMPETITOR

- **PERFORMANCE**

● Speed	-	5nsec per Gate
● Speed X Power	-	.5 - 1.0 pJ, Highly Versatile
● Temperature	-	-55°C to +125°C

- **DENSITY**

● Gate Size	-	3-5 mils Active Area
● Interconnection	-	Requires Two-Level Metal
● Logic Flexibility	-	Very Good with Input Diodes

- **COST**

● Process Simplicity	-	Similar to Silicon Gate NMOS
● Design Cycle Time	-	Reduced by Automatic Layout

SECTION III
MICROPROCESSOR TESTING

Chairman

W. Richard Scott
Jet Propulsion Laboratory

MICROPROCESSOR QUALIFICATION CRITERIA

by

Eugene R. Hnatek

Monolithic Memories, Incorporated
Sunnyvale, California

μp QUALIFICATION CRITERIA

1. Introduction to μps - understanding the device
2. Understanding μp failure causes
   - chip (die) related
   - package related
3. μp failure modes
   - catastrophic failures
   - soft failures
4. The testing problem
   - electrical testing
   - electrical characterization testing
5. Qualification testing
   - 100% inspection - method 5004
     - precap visual inspection
     - burn in
     - electrical measurements
   - QI/QCI inspection - method 5005 considerations
     - group A
     - group B
     - group C
     - group D
6. Conclusions

INCREASED USAGE OF μPS IN MILITARY/AEROSPACE PROGRAMS HAS BROUGHT FORTH MANY QUESTIONS REGARDING THE RELIABILITY OF THESE DEVICES AND HOW BEST TO PURSUE THE QUALIFICATION PROCEDURE.

THE MYRIAD OF AVAILABLE μPS HAS PRESENTED A PERPLEXING AND OFTENTIMES CONFUSING PICTURE TO THE USER AS WELL AS A REAL CHALLENGE AS HOW TO BEST APPROACH THE QUALIFICATION OF THESE DEVICES.

POPULAR μPS FOR MILITARY/AEROSPACE USAGE

- 2901A
- 8080
- 6800
- 1802
- TMS9900

TO CONSTRUCT A VIABLE QUALIFICATION PROGRAM FOR A GIVEN μP ONE MUST UNDERSTAND THE PHYSICAL STRUCTURE OF THE DEVICE UNDER CONSIDERATION...

- topographical layout
- interrelationship between inputs and outputs
- interrelationship between on chip functional blocks
- interrelationship between instructions and on chip functions
- relationship between die topography and accessible ports
- sensitive portion of circuits due to positioning
- chip related failure modes
- package related failure modes
- peculiarities of alternate suppliers μps

UNDERSTANDING μP FAILURE CAUSES

Same <u>basic</u> physical failure causes as with SSI/MSI circuits plus others associated with smaller device geometries and high circuit density...

- prone to defects such as pin holes, metallization faults,etc.
- more bonds - greater probab-lity of bond failure
- hermeticity problems

FAILURE CAUSE ASSESSMENT COMPARISON

	SSI/MSI Bipolar	MOS LSI
Assembly/package failure	50%	45%
Chip failure	50%	28%
Misc (handling)	-	27%

ASSEMBLY AND PACKAGE RELATED FAILURE CAUSES

- open bond wires
- lifted bonds
- lifted chips
- hermeticity

CHIP RELATED FAILURE CAUSE AND RELATIVE FREQUENCY OF OCCURRENCES...

	PMOS	NMOS	BIPOLAR
● Photolithographic defects	6	6	10
● Oxide defects *	6	10	4
● Oxide/junction contaminants	6	10	2
● Metallization faults **	6	4	8
● Diffusion defects	8	10	6
● Mechanical defects in the chip	6	6	10
● Design defects	8	10	6

10 = high frequency of occurrence

* Oxide defects include...

- thin oxide
- slow charge trapping
- polarization
- surface charge phenomenon
- pin holes or cracks

** Metallization defects include...

- contamination
- electromigration
- microcracks
- contact failures

μP FAILURE MODES

1. Catastrophic failures - destructive

   - oxide rupture
   - interruption of Al lines
   - wire bond failures
   - lifted chips
   - corrosion due to contamination (such as trapped moisture in cerdip due to devitrification of glass material)

2. Soft failures - not destructive

   - out of specification conditions
   - parametric drift
   - pattern and pattern sequence sensitivity
   - interrupt
     - trigger on wrong priorities for multilevel interrupt
     - lose data
   - failure to execute instruction and/or interrupt -mnemonic sensitivity
   - loss of carry and bits during recirculation of data
   - instruction and instruction sequence sensitivity

ELECTRICAL TESTING

The key to testing a microprocessor is developing a meaningful and viable electrical test program that is effective in locating weak sisters during group A testing and at the end point electrical measurements.

THE TESTING PROBLEM

μP PECULIARITIES

Problem of testing ups is difficult because you have to contend with software as well as hardware peculiarities and the relationship between the two. Furthermore, the problem of testing different μps are compounded by variations...

- Device architecture
- Chip layout
- Data routing
- Instruction languages
- Random logic nature of the μps
- Pin configuration
- I/O capabilities
- Bit sizes
- Bus organizations
- Fabrication processes
- Interrelationship between on chip functional blocks.
- Second source suppliers circuits
    - are these circuits the same as prime suppliers?

To approach the testing of a μp, one must not only understand its design, layout and construction, but also what the vendors data sheet specification limits mean...

- How does he test the device?
- Are the parameters tested over the entire operating temperature range or only at 25°C
- Are all parameters listed tested or are some guaranteed by design? Which ones?
- etc.:.

How does one adequately test a μp to ensure that it has no shortcomings for all possible usage conditions...

- μP complexities include...
  - random logic nature
  - bus organization
  - circuitry between inputs and outputs
  - on chip interrelationship between functional blocks chip layout.
  - construction of on chip constituent components, i.e. 2 port RAM on 2901.(see example)
  - accessibility of all constituent ports.
  - interrelationship between hardware and software.
  - are the second source products identical to the primary source μp?
- Because of this, it is important to first perform a characterization program to help wring the device out - this leads to meaningful electrical test programs.

EXAMPLE:

2901A

On Chip 2 Port RAM

- Read 2 words simultaneously and asynchronously from 2 different ports.
- Can't get data out of device from both ports simultaneously (multiplexed out).
- Internally can access 2 ports to ALU. (Might want to access both simultaneously for add for example).
- Choose word you want to read from 2 ports.
- Can't run true Galpatt because 2 output ports can run in different direction (not synchronized).
- Need to develop new pattern to most effectively exercise both addresses and take account of different delays through 2 ports such that all possible address jumps are performed on both addresses simultaneously.
- It is effective to test parts with long random data patterns to simulate system usage.

MICROPROCESSOR testing is a trial and error evolutionary procedure that relies heavily on...

- Characterization testing
- Iteration of instructions/data patterns
- Customer feedback

TESTING MICROPROCESSORS MUST BE VIEWED AS TESTING A SYSTEM RATHER THAN TESTING INDIVIDUAL COMPONENTS OR COMBINATIONS OF COMPONENTS.

ELECTRICAL CHARACTERIZATION

Characterization testing is mandatory and it must be performed early in the product design cycle...
before system design is completed such that product shortcomings can be adjusted for in system design margins or a different µp chosen.

WHY ARE CHARACTERIZATIONS PERFORMED?

The complexity of today's LSI device dictates the need. The information obtained from a characterization helps the LSI user to evaluate the following key criteria...

- Vendor's process
- Device sensitivity
- Device application under system variables
- Environment, temperature, other specs
- The testing process
- Evaluate alternate sources

WHY IS CHARACTERIZATION REQUIRED?

- Vendor cannot thoroughly test part for all possible failure modes.
- User needs to know how a given part will work in his system with many variables.
- User/Vendor need to know operating limits.
- To use device in intelligent and effecient design.
- To establish viable device test programs.
- To detect process changes which may affect design safety margins.
- User needs to know how alternate sources compare with each other and with primary source.

Electrical characterization testing is the thorough and exhaustive testing of a sample of a given device type through all practical combinations of supply voltages, temperatures, timing conditions, parametric variations, instruction sequences and the like to find its response under these conditions and to find the limits within which the device remains functional- i.e. it defines the operating limits.

CHARACTERIZATION TESTING INCLUDES...

- stringent functional testing using the worst case patterns or truth tables.
- worst case instruction sequences/data patterns
- timing/parametric variations
- temperature extremes

DEVELOPING A VIABLE μP TEST PROGRAM INVOLVES THE FOLLOWING STEPS...

1. Develop Test Plan
2. Develop "worst case" Instruction Sequence
3. Generate Bit Patterns
4. Test the μP

Items 1 and 2 are the most crucial to testing a μP. Once these are developed items 3 and 4 follow automatically.

Developing a "worst case" instruction sequence (item 2) is an iterative procedure wherein various combinations of instruction sequence/data patterns are applied to the device to determine which are the most sensitive.

GENERATING AN INSTRUCTION SEQUENCE IS THE CRUX OF THE ENTIRE MATTER OF μP TESTING IN ADDITION TO HANDLING THE VARIOUS ON CHIP COMPONENTS VIA THE PACKAGE PINS.

FIVE METHODS HAVE BEEN DEVELOPED FOR GENERATING INSTRUCTION SEQUENCE AND SUBSEQUENTLY TESTING THE μP...

- Self diagnostic method
- Comparison method
- Algorithmic pattern generation method
- Modular sensorialization
- Stored response method
  - emulation
  - simulation

Each of these has its advantages and disadvantages. In general, however, a combination of some of these is used to develop a data pattern for a given μp.

UNIQUE FORM OF CHARACTERIZATION TESTING USING USERS SYSTEM CONSTRAINTS

During characterization testing, use typical data sheet or system constraints (supply voltage, instruction, time condition, operating temperatures, and the like).

The test program/parameters are held constant except one which is varied until that parameter results in a device failure (non functionality). The last valid operating point is noted and testing of another variable is begun. Process is repeated until all parameters are tested to failure for the entire sample population.

This results in voluminous amount of data being generated that must be converted to meaningful easy to interpret format so that meaningful and valid conclusions can be reached. To this end, histograms and shmoo plots are extensively used to make it easy to grasp the precise effects of temperature, supply voltage, or timing conditions or any of the parts key parameters.

Characterization testing provides a meaningful 100% electrical test program exercising only those worst case instructions, voltages, temperature and timing conditions to which the device is sensitive, allowing one to reduce the number of test vectors from many thousand to several hundred.

QUALIFICATION TESTING

- 100% inspection per method 5004 of Mil-Std-883.
  - LSI devices not covered by Rev A.
- Sample testing per method 5005 of Mil-Std-883.

100% ENVIRONMENTAL TEST CONDITION CHANGES FOR LSI DEVICES

Notice 2 Mil-Std-883

Method 5004-3

Precap visual	Change inspection criteria to be compatible with LSI technology
Temperature cycle	Increase number of cycles from 10 to 50
Constant acceleration	Change from condition E (30KGs) to condition D (20KGs) if package meets certain physical criteria. Y1 axis is most important.
Burn in	Increase duration from 160 hrs. to 240 hrs.

PRE-CAP VISUAL INSPECTION

- Much more difficult than SSI/MSI circuits
  - reaching photolithography limits through high density per chip
  - use of scaling techniques
  - assessing what to look for

- 2010.2 notice 2 rev A included the following technologies
  - double layer metal
  - polysilicon technology
  - MOS in general

- 2010.2 also loosens up visual criteria (contacts, contact coverage,etc.)

- 2010.2 does not include the following LSI technologies
  - isoplanar
  - double polysilicon
  - VMOS
  - DMOS

BURN-IN

- Burn in is the most effective screen in weeding out weak devices.
- Burn in conditions developed for a particular device must be based on an understanding of that devices' construction, operation and topography.
- One must partition the device and determine how many junctions/gates are reverse biased and how many are actually stressed under forward and reverse bias conditions to determine which type of burn in is most effective.
- Combinations of temperatures, voltage stress and time are used to accelerate removal of infant mortalities.
    - High voltage cell stress tests at elevated temperatures are more effective in uncovering oxide defects for certain MOS devices than is dynamic burn in.
    - For complex bipolar digital devices dynamic burn in is more effective and efficient than static burn in.
- In addition to the conditions of temperature, voltage, current and time which apply to the burn in of less complex devices, one must determine what instructions/data pattern with which to load the μp and circulate through it during the burn in.
- This is very crucial to obtain an effective means of burning in a μp.

ELECTRICAL MEASUREMENT (INTERIM AND END POINTS)

Guidelines

A. Gross defects

1. Generate vector list which shows weakest (worst case) transition by device design.

2. Generate pattern from vector list

3. Test part

B. Random defects

1. Use simple tests that check every addressable location but not all combinations in both states (1's and 0's).

C. Full functional and AC tests are required over the entire temperature range. There is a lot of circuitry between the inputs/outputs. Thus, functional and AC testing is mandatory.

D. Soft errors can be a result of insufficient testing for data pattern, temperature, voltage margins, timing conditions and instruction sequences, thus the need for electrical testing using comprehensive pattern at power supply and temperature extremes.

QI/QCI INSPECTION

METHOD 5005

μP RECOMMENDATIONS FOR ASSESSING DIE INTEGRITY

1. Group A tests - Very important

   Subgroup 9 (switching tests) should be included in subgroups 4,5, and 6 tests (dynamic tests).

2. Group B tests - Not important on μp per se

3. Group C tests - die related failures - very important

   Subgroup 1 operating life test

   - Use dynamic op life for all μps (condition D) especially Bipolar
   - Additionally some MOS μps require reverse bias op life (condition C) to isolate drift rejects, because dynamic BI may not promote drift as much as R.B. op life.

   Subgroup 2 - very important

   Temperature cycling, constant acceleration, seal.

4. Group D tests - package related

   Subgroup 1 - Physical dimensions - not important

   Subgroup 2 - Not very important

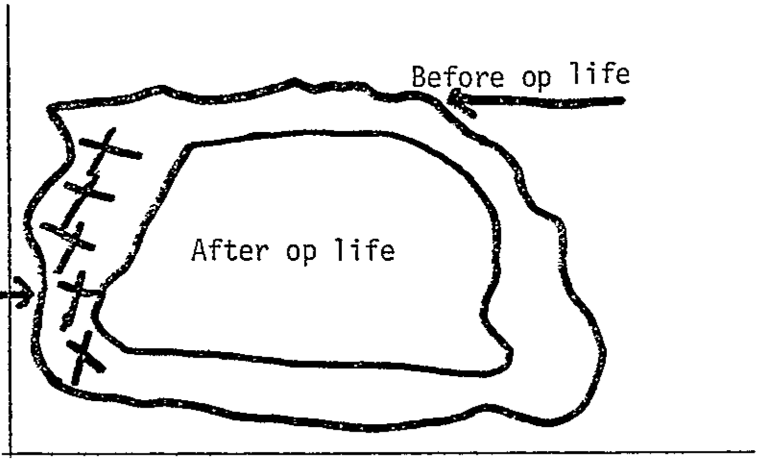
   Subgroup 3 - Very important

   Moisture resistance testing is very important in identifying instabilities.

   Subgroup 4 and 5 - identify gross rejects

END POINT ELECTRICAL MEASUREMENTS

- DC/Functional as a minimum over power supply and temperature extremes. Look for functional shift after op life.
- For delta criteria use die parameters - those showing leakage.
- Display drifts via shmoo plots.



Weakness in circuit may not be identified by dc or one point functional tests.

METHOD 5005 μP QUALIFICATION EFFECTIVITY RATING

Operating life	10
End Point electrical measurements	10
Subgroup 2, group C	4
Subgroup 3, group D	5

(10 = highest)

Mil-Std-883 and Mil-M-38510 must be reviewed and drastically revised to address LSI devices.

Additionally, a means of specifying the instruction set/data patterns to be used in testing a given device type (slash sheet) must be specified such that the µp can be understood and tested on any LSI test system.

CONCLUSION

- Introduction of µP has presented users with a serious test/qualification challenge for these complex devices.
- Component engineer must discard traditional component testing, evaluation and qualification concepts and restructure their thinking to treat the UP as a system.
- In order to approach the qualification problem one must understand in detail the device design, topography and construction to effectuate a meaningful and viable test plan.
- In the qualification process, operating life tests, and end point electrical measurements are extremely important to assess die integrity.
- Reliability test data taken to date, indicates that the inherent reliability of µP's is no worse than for SSI/MSI devices, memories, or other LSI devices. More data needs to be taken.
- Mil-Std-883, Mil-M-38510 must be drastically revised to address µps (hardware and software).

REFERENCES

1. Microprocessor Device Reliability, By Eugene R. Hnatek, Southeastcon April 4-6, 1977, Williamsburg, Virginia.

2. Introduction to Microprocessor Testing Problems and Test Methods, by Eugene R. Hnatek, 1976 Nepcon West (March) East (June).

3. Users Test Not Data Sheet Assure IC Performance by Eugene R. Hratek, Electronics, November 27, 1975.

4. Automatic Testing of Integrated Circuits, A short course, ISCM, Chicago, Illinois.

5. IEEE 1975/1976 Memory/LSI Test Symposium Proceedings.

EVALUATING MICROPROCESSORS WITH MODULE SENSORIALIZATION APPROACH

Richard McCaskill
Macrodata Corporation
Woodland Hills, California

The problems of testing microprocessors has been elevated past the conventional methods of testing integrated circuits. Just the fact that the microprocessor is not a simple collection of gates in a random format or a well ordered structure like that of a large scale memory, does not lend itself to the conventional means of testing. What is meant by the conventional means of testing is the commonly used DC test checking for input and output voltages and currents. This DC testing cannot prove tnat the microprocessor is operational, because there are from four to six or more levels of logic between the input and output pins. Also the conventional way to test random logic by applying a string of input patterns in a burst will only check for steady-state faults stuck at logic 1 or stuck at logic 0.

There presently are many ways that both manufacturers and users are performing testing of microprocessors today. These include such methods as self-test, comparison testing, stored pattern testing, and algorithmic-aided pattern testing.

First Step in Testing

The first item to be considered when testing a microprocessor is to understand the operation and architecture structure of the microprocessor. The operation of the microprocessor is controlled by the execution of an instruction set unique to each microprocessor. There are a great variety of microprocessors on the market today, ranging from 2 and 4 bit slices to 4, 8, and 16 bit complete microprocessor units. But of all the product types, 4-bit slices like the 2901 and 8-bit units like the 8080 have gained the widest acceptance. This makes the 2901 and the 8080 microprocessors good examples to use in describing the testing techniques.

In general, a microprocessor has two internal buses: an 8-bit bidirectional data bus and a 16-bit unidirectional address bus (Fig. 1). The data bus carries both the instruction codes and data. Instructions are decoded and executed in connection with the appropriate controls, with data going to both the arithmetic logic unit and accumulator to be manipulated by special arithmetic or logic operations. The address bus links the main memory, where both instruction codes and data are housed. Stack pointers, program counters, and register files also supply information to the address. Finally, there is an instruction decoder which interprets each instruction and controls all operation of' the microprocessor.

Module Sensorialization

Since a microprocessor is a complex sequential logic structure and not simply a few gates or an LSI memory, a true and meaningful test requires the understanding of the hardware architecture and software functionality rather than only the simple logic of the elemental structures.

The hardware architecture is the internal organization which consists of an ordered set of modules, such as the register stack, accumulator, arithmetic logic unit (ALU), etc. Software functionality is a set of ordered microinstructions which can be used to monitor the operation of these modules.

Upon developing complete knowledge of the microprocessor slice through both areas, one can develop an ordered set of test sequences in the microprocessor slice instruction set for testing each module one by one until a complete test has been developed.

In general, a microprocessor slice has two buses: an address bus and a data bus. The address bus performs two functions: addressing the external memory and/or addressing the internal scratch pad memory. The data bus also performs two functions: supplying input data to the processor and outputting processed data. The data bus links the internal functions, like the scratch pad memory, registers, ALU, etc., together. (See Figure 2.)

Modular Breakup

The next step in microprocessor slice testing is to partition the device into modules. Some modules may possibly overlap. The selection of each module will be accessible from its input/output bus by the execution of its microinstructions. In other words, data should be able to be applied to the slice input and propagated to the output directly or indirectly by the use of the microprocessor instructions set. The test shall then be generated for each module of the slice so that a worst-case test pattern will be run on that module. The sensitivity to this pattern is determined when a pattern of galloping 1's and 0's is applied to the device. This occurs when one of the modules happens to be a random access memory (RAM).

From the standpoint of software functionality, a set of MPU instructions should be executed when testing the first module. Proceeding toward the second module, another set of new microprocessor instructions will be executed. (Some of these instructions may have been executed previously.) This process will then continue until all of the instructions within the instruction set are used while testing each module.

Two-fold diagnostic information is provided by this technique. First, from a hardware point of view, if a failure occurs, the faulty module is pinpointed. Inherent in this type of modular procedure is the fact that convenient breakpoints exist in a module by module basis. Secondly, in conjunction with each module, a set of microinstructions is executed. Therefore, if any fault occurs, that specific instruction(s) can be isolated and identified.

Architecture and Test Flow

The architecture of the 2901 lends itself to the modular sensorialization approach because of its own hardware and microinstruction architecture. (Figure 2 illustrates the block diagram of the 2901.) In examining this diagram, one will notice that the device can be divided into the following modules: RAM, Q register, arithmetic logic unit (ALU), ALU source decode multiplexer, RAM and Q register right/left shift logic.

Once one has acquired the information on the module breakdown, a test flow can then be generated. The first thought should be that since the 2901 has an ALU section, the first areas to be tested should be those areas which supply data to the ALU. The most logical of these is in the RAM module and then the Q register module. Once these modules have been thoroughly tested, they can be used as reliable data sources for the ALU module test.

A typical test flow for the 2901 would start with the RAM memory, followed by the Q register, ALU source decode multiplexer, ALU, and finally the RAM and Q register right/left shift logic. (See Figure 3.) During this test flow, all microinstructions for the 2901 will be used.

Test Technique

Formulating a test plan will differ between the manufacturer and user. The reason for this is that the manufacturer has access to the logic diagrams of the device, which the user in most cases cannot obtain, and their quantities are in larger amounts than the user. Therefore, more elaborate tests can be developed which optimize test performance and test time. The user has an advantage over the manufacturer because his test, at its simplest form, can be tailored to his specific needs, but the manufacturer's test has to guarantee all operations of the microprocessor. Not receiving schematics, logic diagrams or other circuit information, the user must, therefore, rely on either vendor supplied test programs or he must perform extensive characterization to generate worst-case test patterns. This characterization is needed to guarantee full operation of the microprocessor for the variety of applications in which the device is used.

The Optimum Test

At first glance at the block diagram (Fig. 4) of the 8080 MPU, the complexity of the device is not readily indicated. This is because there are only eight data input lines. However, in addition to accepting data from the input bus, the MPU can accept data from internal registers and accumulators. If the MPU could only perform one instruction, a test could be developed without much difficulty, but the MPU is capable of executing many instructions in sequence. Because of this, the number of combinations of instructions and data patterns that the MPU can perform would be extremely long.

A commonly used formula for calculating the total test time to exhaustively test an MPU has been used. The formula is $C=2^{mn}$, where C is the number of combinations of instructions and data patterns, m is the number of data bits in each word, and n is the number of instructions the MPU is capable of executing.

For example, an eight-bit MPU that has only ten instructions would require 2^{80} test cycles for an exhaustive test of all possible combinations. Assuming a test cycle of 1 µsec, the MPU would take approximately 38 years to exhaustively check all combinations of instructions and data patterns.

The 8080 MPU can perform approximately 76 different instructions. Using the above formula, there would be a total of 2^{608} possible combinations that could be performed. Obviously, this is an astonishing number to exhaustively test the 8080 MPU.

Test Techniques

Once realizing that the optimum test cannot be created, one looks for other means to test the MPU. The first approach to be considered is what is called self-test. Self-test is the simplest and cheapest means of determining if an MPU is working. Self-test or in-circuit test is the technique in which the device is placed into the circuit where it will be used and the circuit is tested for correct operation. This is used by some users who feel the cost of incoming inspection cannot be justified. Therefore, they will typically test the device using several different system operations. The advantage of this way of testing is that the actual operation of the device is tested in its circuit and it eliminates the requirement for a separate costly test system. The disadvantage of this technique is that any of the in-circuit condition changes, like voltage fluctuations, temperature, timing, and instruction changes, may not be detected until the unit is in the field. The rework cost of finding and removing a faulty device must be considered before a person would select this method of testing. Typical cost for finding and replacing a gate is as follows:

$3	- board level
$30	- system level
$300	- in the field

Since an MPU is more complex than a gate, the above cost would be multiplied by the complexity factor of the MPU.

The second method of testing would be that of comparison testing. Comparison testing is the method in which a known good device is compared to the device under test. The hardware required for this type of test is very simple (Fig. 5). All that is required is a pseudo number generator connected to all inputs and all outputs from the known good device compared to the device under test. If exact comparison does not occur the device under test is bad. The advantage of this method is that the test system is inexpensive to develop and, with a little more hardware, added voltage and timing conditions can be created. Also, if the device is operated for a few minutes, most paths through the device will be checked. Like any test method it has its disadvantages also. The biggest disadvantage is that this method requires a known good device, which is a problem in itself. Some MPU's have illegal instructions, therefore, no guarantee can be made of the data coming out of the device. Also, critical timing into the device may not be able to be maintained if pseudo numbers are applied to the input of the MPU. Last of all, if the device fails, no failure information can be obtained to determine what caused the failure.

The next method of testing is the stored pattern method. The stored pattern method utilizes a known good pattern stored in some form of data memory and input and compared to the device under test (Fig. 6). There are two means of generating patterns using this method. The first method is to input a test pattern into a known good device and record all input stimuli and output data. The input patterns would be created from some known application. The second method of generating the stored pattern would be to develop a software or hardware simulator for the device to be tested. A known instruction sequence would then be stored and used to compare with the device under test. The advantage of this technique is that the user's instruction sequence can be completely tested or that sensitive data paths can be checked with relative ease. Due to the fact that the tester required to perform this type of test usually incorporates variable voltage and timing circuits, these parameters can also be checked. The main disadvantage of using a known good device for generating the test pattern is a "known good device." What test is available to determine what is a known good device? The disadvantage of the simulator approach is that a software or hardware simulator is required. Since the schematic and logic diagrams for each MPU are not readily available from the vendor, it is difficult for a user to develop the simulator. Even if these could be obtained, it would take a knowledgeable programmer three to six months, at least, to develop the software. Other disadvantages to this method are:

- Lack of diagnostics. Virtually no information is generated to indicate which instructions or parts of the device caused a failure. Analysis of faults requires a separate test routine or a sophisticated program to interpret the results of the stored-program tests.

- Large, expensive memory. High-speed random-access memories or shift registers become quite expensive when any great amount of memory is needed. In testing the program counter for the 8080, for example, 262,000 distinct patterns are required. A memory test on the register array of an 8080 takes approximately 50,000 patterns. The cost of memory can quickly become a major part of the total cost of the test system.

- Long transfer time. The overhead time required to transfer a long pattern from disk, core, or other mass memory to high-speed RAM can make a large dent in the throughput rate of the test system. If transferring a 1,024-bit pattern from disk to RAM takes 50 milliseconds--a typical figure--transferring the test pattern for the program counter takes 13.1 seconds of overhead time (262 x 50 x 10^{-3} seconds) in addition to the test-execution time.

- Inflexible program. The stored program cannot easily be modified while tests are in progress. This rigidity makes it difficult to perform special or unusual tests on a single unit. A substantial amount of off-line software support is therefore needed if such tests are to be accomplished.

Algorithmically-aided pattern generation is the technique of developing algorithms for each working module in the MPU. The technique is based around the method of modular sensorialization in which the MPU is divided into working modules which can be tested separately utilizing the instruction set of the MPU. The test is developed by selecting modules that will be required by other modules in the development of their test first, then testing further modules until all are tested. Once these modules have been determined, algorithms are written which will simulate the module's operation at real time and compare the tester results to the MPU results.

Algorithmic pattern generation eliminates the stored pattern problems. In the algorithmic method, a sequence of defined patterns is formed by a high-speed pattern generator under microprogram control. The user can change the program easily, even while tests are in progress, to generate a variety of distinct patterns. This technique, which eliminates the cost of memory for pattern storage and the delay time in transferring patterns from mass memory, is extremely efficient and flexible in generating patterns for logic modules such as binary counters, random-access and read-only memories, and shift registers, as well as microprocessors.

When used in conjunction with module sensorialization, algorithmic pattern generation permits faults to be diagnosed so that the particular module or instruction that caused a failure can be isolated. The disadvantages of this method is that a sophisticated tester is required. The programmer needs to be knowledgeable on both operation of the MPU and the test system to develop the program.

Test Setup

The advantages of algorithmic pattern generation can be well illustrated by tests on the 'Q' register and the Right-Left shift operation on the RAM. In both cases it will be shown that not only will all operation on these areas be tested but also all number combinations will be checked with no stored input or output patterns. The conventional method of testing would require over 8000 patterns to be storea for the same test.

All that is required to test for all number combinations is a short algorithm. A simple test is written in which data is input into the data input port through the ALU and into the 'Q' register. Then the 'Q' register is selected and its contents tested.

The test setup for this test is shown in Figure 7. All that is required is to have a register to keep track of the input and output data, a register to keep track of the test cycles, and three bits to control microinstruction and the clock to the MPU. As shown in Figure 7 a four bit register called the 'T' register is used to supply the data input and for comparing to the output data. The 'A' register is used to keep track of the test cycles performed. The 'R' bit selects the ALU data source. The 'W' bit selects the ALU data destination. The 'C' bit is used to clock the device. Last of all, a microcontroller multiprocessor is used to control all the registers, select bits, and testing for an error on the compare data.

Test Description

As illustrated in the flow chart (Figure 8) a simple test is run on the register but the loading of all possible number combinations and their compliments. For this test, data is input through the data input port, then through the ALU and clocked into the 'Q' register. For this part of the test the ALU source is selected to octal code 7 (R=D, S=0) and the ALU function is selected to octal 3 (R or S) throughout this test. Keeping the ALU function in one selection (R or S) allows for easy fault isolation due to the fact that only one path is ever used through the ALU. Following the load of the 'Q' register operation it is selected and tested for the correct value. This is done by selecting the 'Q' register position through the ALU selector (octal code 2) and the ALU destination to octal code '0' for the 'Q' register to the 'Y' output. The test vector is then complemented and the above operation repeated. Upon completion of this operation the test vector is decremented and complemented and the same test is performed again until all combinations are tested (0,15,1,14---14,1,15,0).

Complex Test Pattern Generation

Testing the Right/Left shift operation of the RAM, a more complex test pattern is required. The purpose of this test is two-fold:
1) To test completely the Right/Left operation of the RAM and
2) To test the RAM data output latches. To perform this test, two locations in the RAM will be used. The first location will be the location in which the shifted data will be stored. The second location is used for storing the background pattern that is output to the latch after the shifted data has been clocked into the data latch.

Additional hardware is required to perform this test over the previous test. (See Figure 9.) First of all, two data values are needed to be stored: the shifted data and the background data. The shifted data will be stored in the previous 'T' register and the background data will be stored into the background register or 'B' register.

Two other features will be added to the 'T' register. Those are additions of a circular shift of the most significant bit to the least significant bit and a bidirectional 16 bit transfer path between the 'T' and 'B' registers. Last of all, two index registers will be used to keep track of each shift operation ('J' registers) and each incrementation of the background value. All of these operations are then controlled by the microcontroller multiprocessor.

Testing The RAM Shift Operation

The algorithm for testing the Right/Left shift operation of the RAM is not really as complex as it appears in the flow chart (Figure 10). Two locations are used in the RAM; one stores the test shift data and the other the background data. The test performed loads these two locations and then shifts the pattern $A1E5_{16}$ through the RAM.

After each shift, the shifted data is checked and the background is then addressed and the output is again tested to see that the latch did not change state. The number $A1E5_{16}$ was selected because all number combinations will be shifted through the RAM using this pattern. After all sixteen bits of the pattern have been shifted through the RAM, the background word is incremented and the test repeated. The same test is then repeated on all RAM addresses making the test pattern address the complement of the background pattern.

In conclusion, the algorithmic pattern generation technique can be applied to develop the required tests outlined in Figure 3. Presently, there are on the market, commercial test systems, such as the Macrodata MD-501 LSI test system, that are specifically designed for algorithmic pattern generation as well as pattern storage testing. These are not necessarily required to apply this technique. Special purpose testers can be developed in-house for this specific application.

The modular sensorialization technique, as applied to testing microprocessor slices, can reduce the difficulties encountered and the cost of testing microprocessor slices.

Bibliography

1. 'AMD - Am2900 Bipolar Microprocessor Family', manual, AMD Corporation, June 1976.

2. 'Two New Approaches Simplify Testing of Microprocessors', Albert C. L. Chiang and Richard McCaskill, Electronics, January 22, 1976.

3. 'The Microprocessor Test Quandary', Scott Holyfield, Electronic Packaging and Production, February, 1976.

4. 'Microprocessor Testing -- Method or Madness', D. H. Smith, 1976 Semiconductor Test Symposium, October, 1976.

5. 'Test Approaches for Four Bit Microprocessor Slices', Richard McCaskill, Macrodata Corporation, 1976 Semiconductor Test Symposium, October, 1976.

6. Intel - 8080 microprocessor manual, Intel Corporation, 1976.

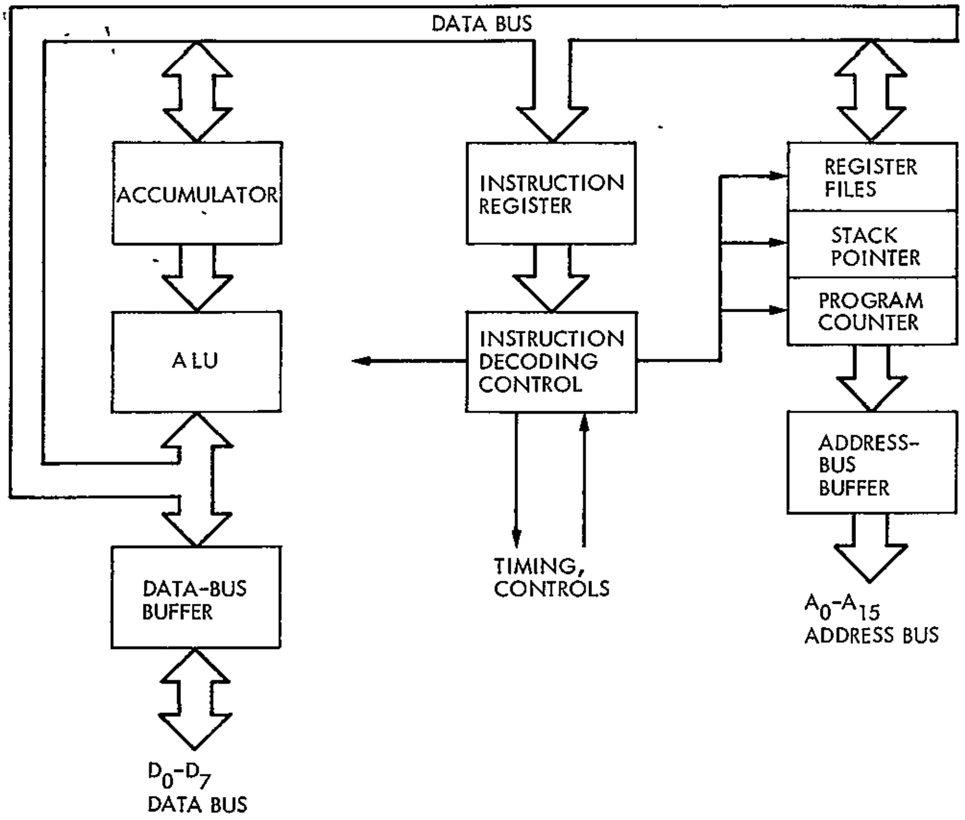


Figure 1. Basic MPU Block Diagram

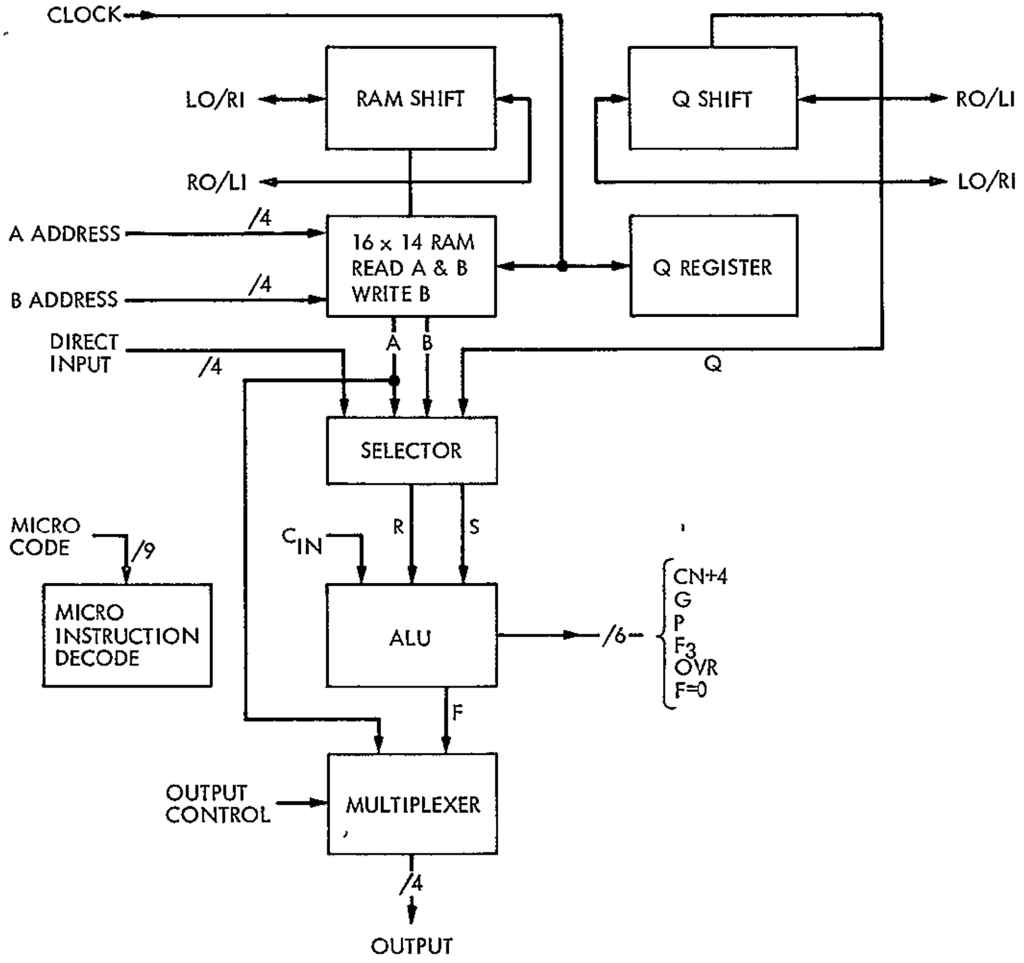


Figure 2. Mıcroprocessor Slıce Block Diagram

Test-Flow Chart	Functional-Test Description	Test Pattern
RAM Test	A galloping "1" and "0" pattern is applied to the RAM in three combinations. 1. The RAM addressed by the "A" address and tested through the "Y" output port directly. 2. The RAM addressed by the "A" address and tested through the ALU. ALU is held at a fixed instruction. 3. The RAM addressed by the "B" address and tested through the ALU. ALU is held at a fixed instruction.	Approx. 3000
"Q" Register	A number 15 is loaded into the register and then read. Next, a number "0" is loaded and read. This is followed by a 14, 1, 13, 2, etc. until a "0" then a 15 is loaded.	Approx. 100
ALU Source Decode	The ALU Source Decodes are tested to see if all decodes are possible. The test is performed by loading values into the RAM and "Q" register and selecting all decodes while testing for any interaction between bits or selections.	Approx. 50

Figure 3. Test Flow

Test-Flow Chart	Functional-Test Description	Test Pattern
ALU	A series of numbers are loaded into the RAM and "Q" register. These numbers are then used as inputs to the ALU. At the same time, all outputs and flags from the ALU are monitored, while incrementing operations the ALU can perform.	Approx. 1000
RAM and "Q" Register Right/Left Shift Mux.	All numbers from 0 to 15 are shifted through the RAM and "Q" register. While the RAM section is being tested, all locations are tested. After each shift, all possible number combinations are outputted to the output latch without clocking the latch, to see if there is any latch sensitivity.	Approx. 8200

Figure 3. Test Flow (Continuation 1)

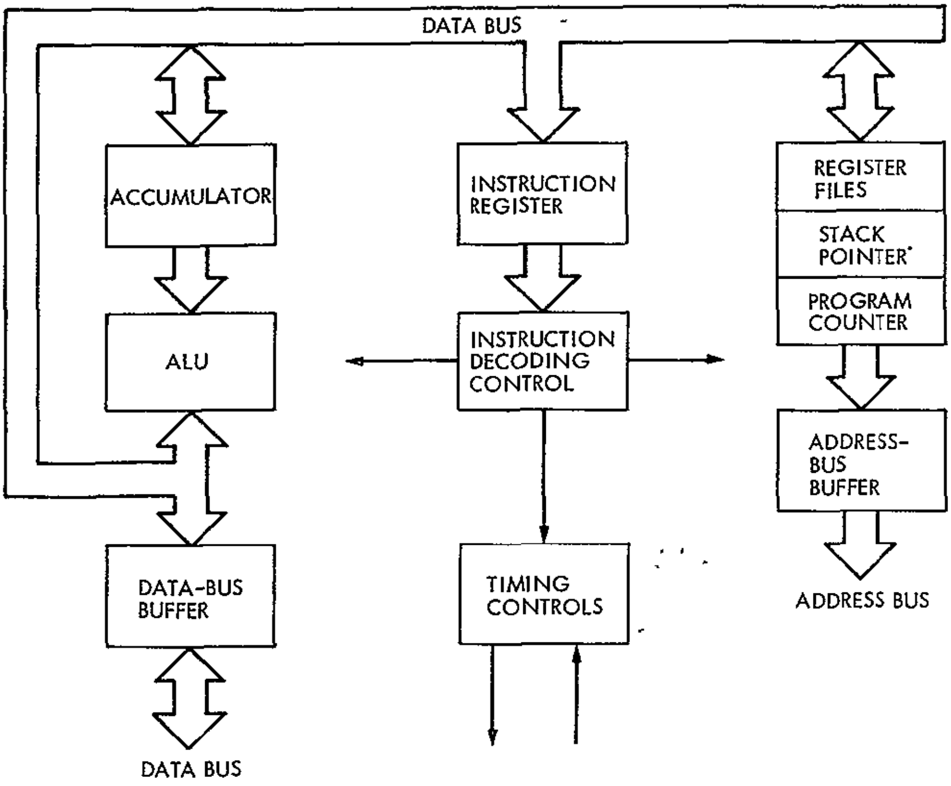


Figure 4. 8080 Block Diagram

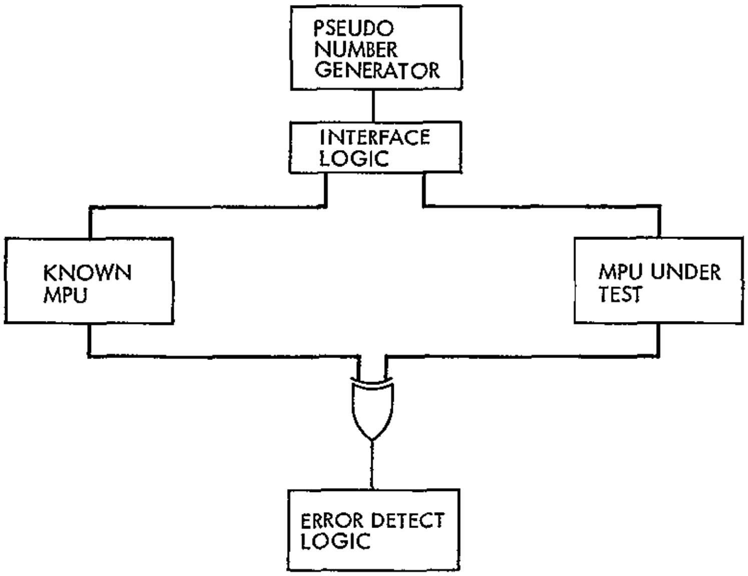


Figure 5. Comparison Testing

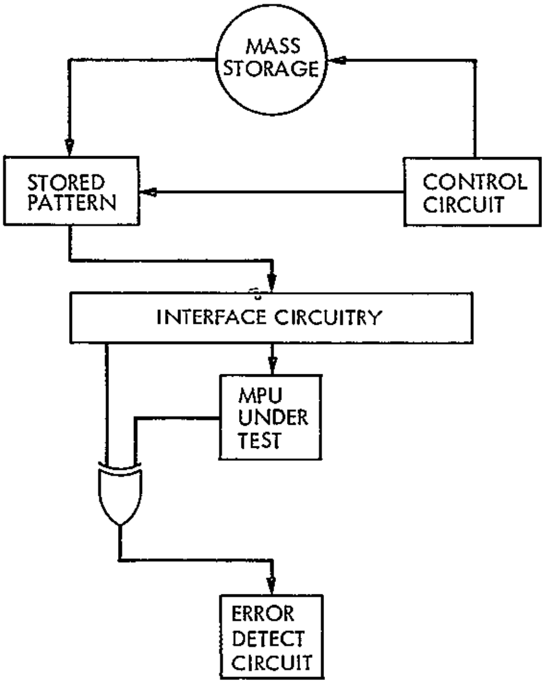


Figure 6. Stored Pattern Testing

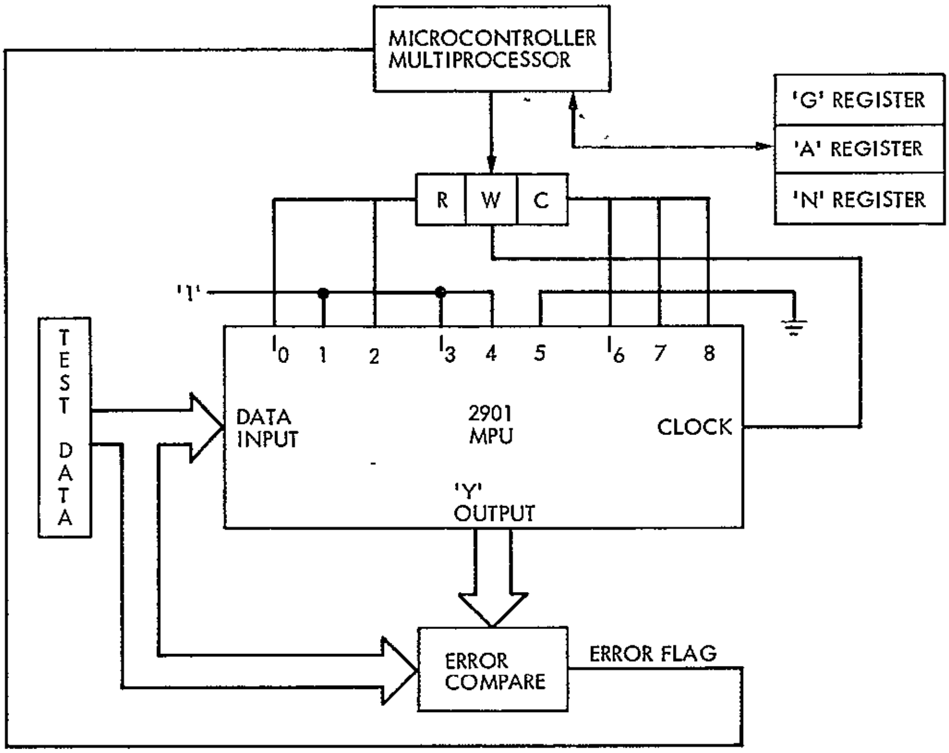


Figure 7. Test Setup for 'Q' Register Test

C-2

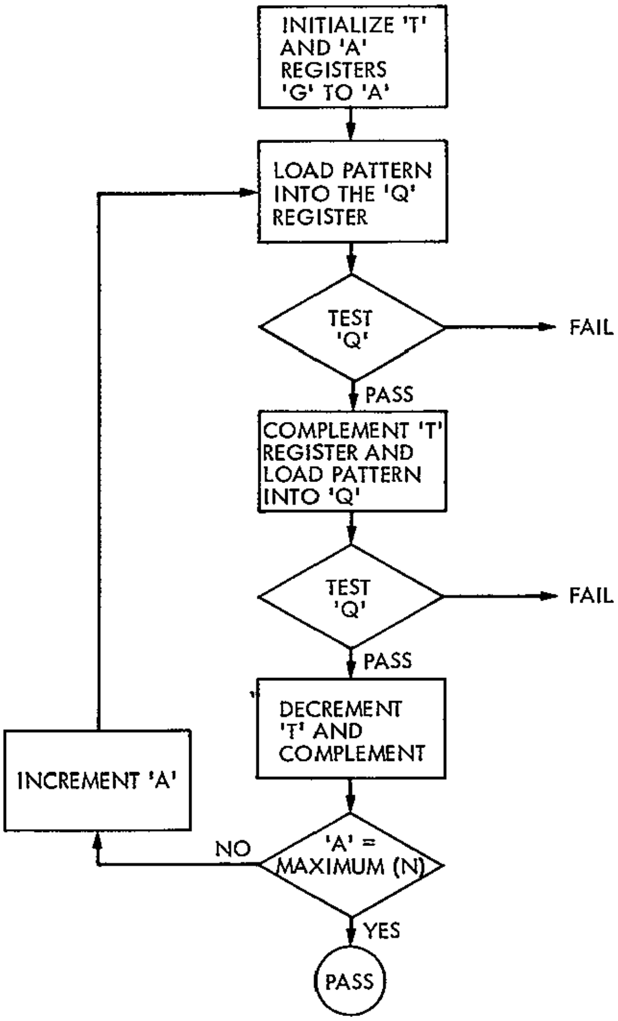

INITIALIZE 'T' AND 'A' REGISTERS 'G' TO 'A'
LOAD PATTERN INTO THE 'Q' REGISTER
TEST 'Q'
FAIL
PASS
COMPLEMENT 'T' REGISTER AND LOAD PATTERN INTO 'Q'
TEST 'Q'
FAIL
PASS
DECREMENT 'T' AND COMPLEMENT
INCREMENT 'A'
NO
'A' = MAXIMUM (N)
YES
PASS

Figure 8. 'Q' Register Test Flow Chart

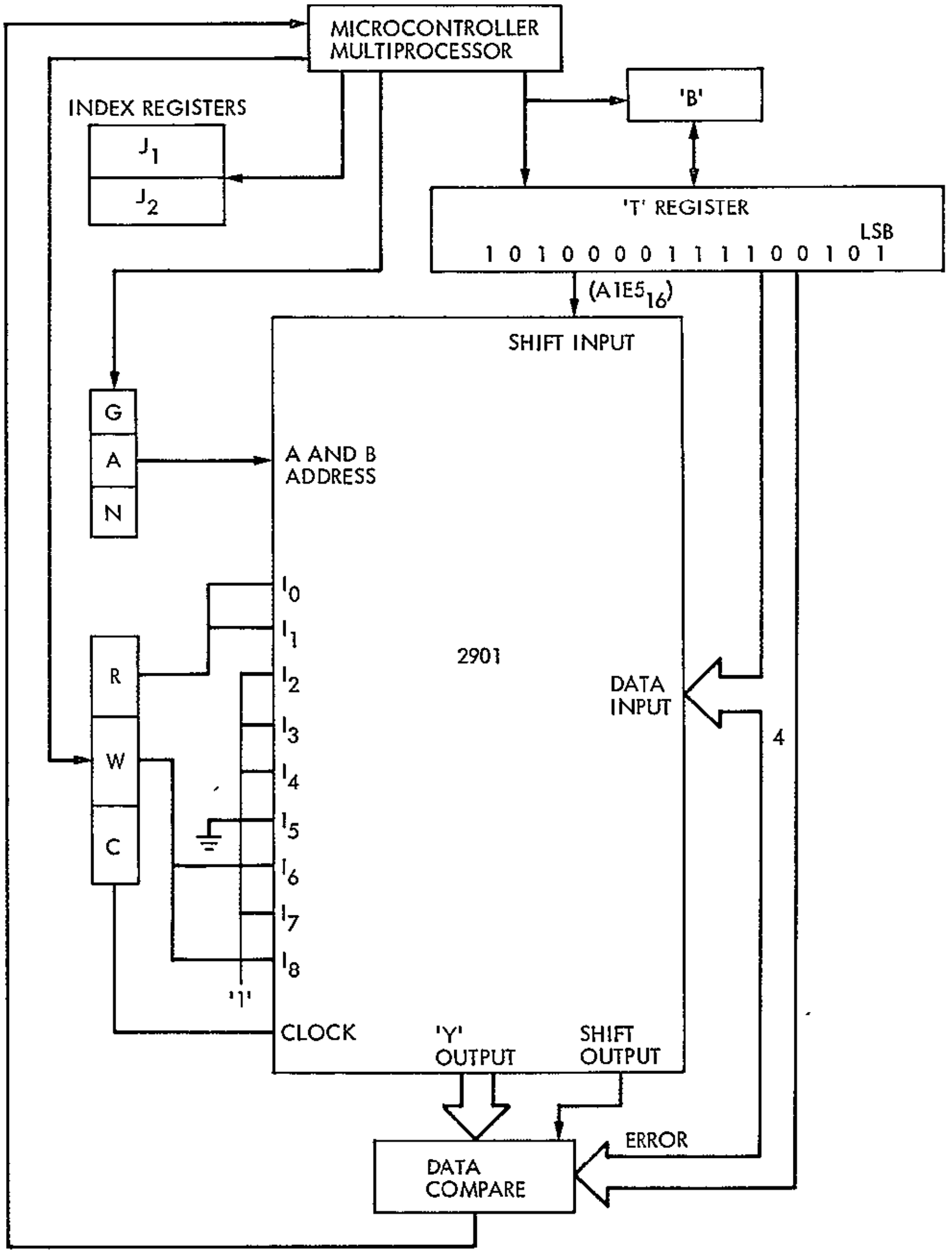
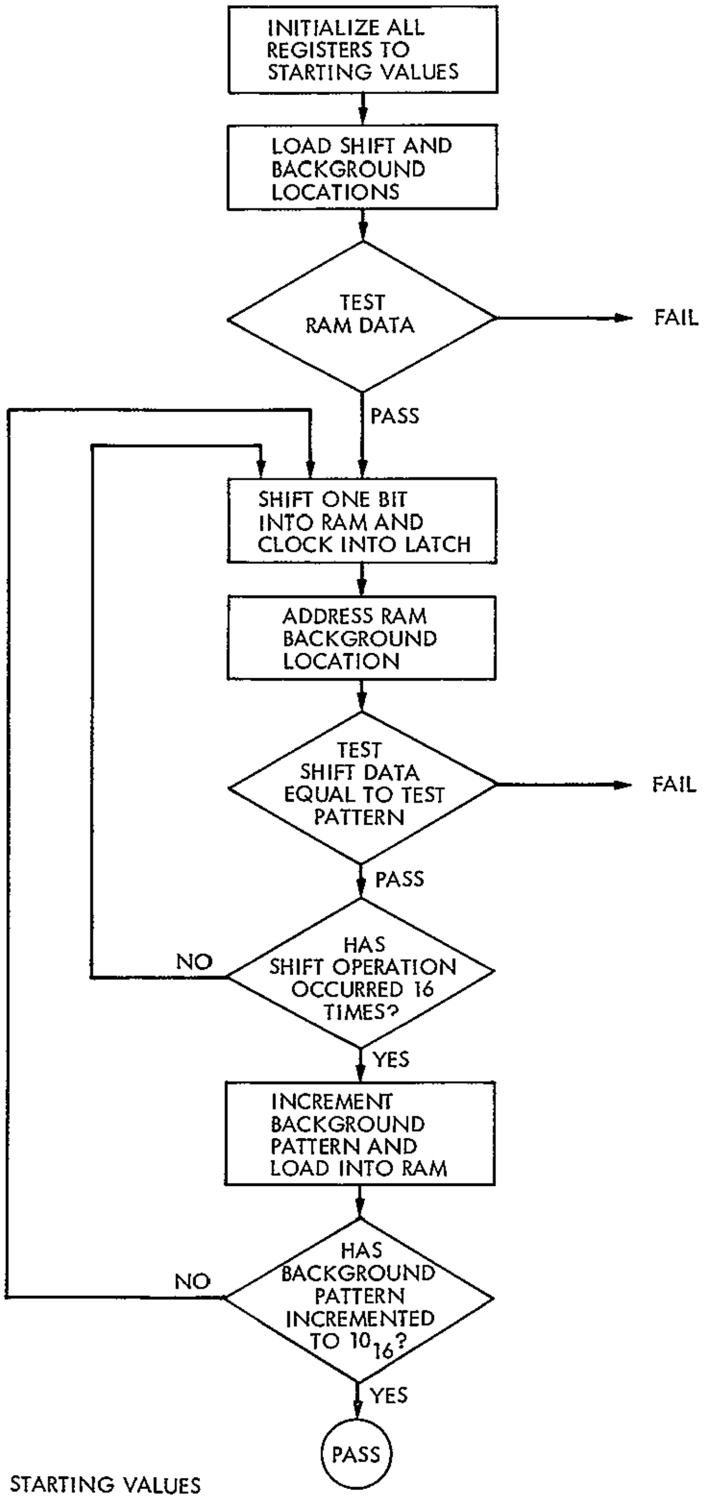


Figure 9. Test Setup for the Right/Left Shift Test


INITIALIZE ALL REGISTERS TO STARTING VALUES
LOAD SHIFT AND BACKGROUND LOCATIONS
TEST RAM DATA
FAIL
PASS
SHIFT ONE BIT INTO RAM AND CLOCK INTO LATCH
ADDRESS RAM BACKGROUND LOCATION
TEST SHIFT DATA EQUAL TO TEST PATTERN
FAIL
PASS
NO
HAS SHIFT OPERATION OCCURRED 16 TIMES?
YES
INCREMENT BACKGROUND PATTERN AND LOAD INTO RAM
NO
HAS BACKGROUND PATTERN INCREMENTED TO 10_{16}?
YES
PASS

STARTING VALUES
SHIFT PATTERN = $A1E5_{16}$
BACKGROUND PATTERN = 0
RAM SHIFT DATA LOCATION = 0
RAM BACKGROUND LOCATION = 15

Figure 10. Right/Left Shift Flow Chart

AN APPROACH TO QUALIFICATION TESTING MICROPROCESSORS

Lenward Holness
Hughes Aircraft Corporation
Culver City, California

A. INTRODUCTION

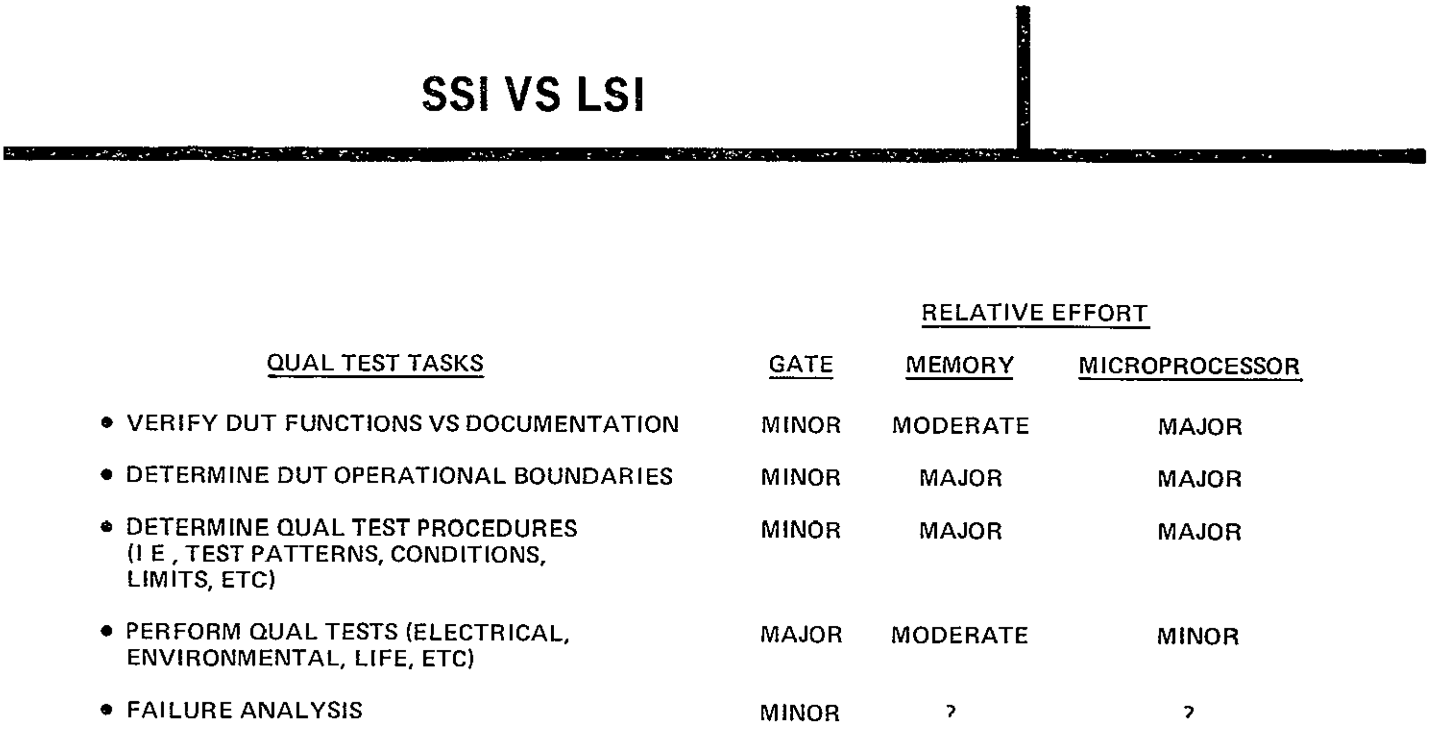
The following article is a summary of the presentation on "AN APPROACH TO QUAL TESTING MICROPROCESSORS" given by L. Holness at the Microprocessor Workshop II, California Institute of Technology, on April 21, 1977.

B. PRESENTATION

Qualification tests of integrated circuits are usually designed to answer two basic questions:

1. Will the circuit meet its electrical performance requirements?
2. Will the circuit continue to work over its required life and environmental requirements?

While the QUAL test procedures to meet these objectives are essentially the same for SSI, MSI and LSI, the relative effort required for each step can differ considerably as shown in Slide 1. The first step on this slide includes the effort necessary to generate test patterns to be used during functional, DC parametric, and AC parametric tests. The next two steps include the electrical characterization phase. These three steps can easily represent between 50 percent to 70 percent of the effort required to perform qualification tests on microprocessors. Item 5 from this list, that is failure analysis, should not be attempted on a device of this complexity without the addition of another phase, called device physical characterization. The effort required for physical characterization could easily equal the effort required for electrical characterizations. Failures on LSI devices are frequently caused by marginal performance characteristics rather than catastrophic defects. Thus, intimate design and construction knowledge is necessary if meaningful failure analysis results are to be expected.



SSI VS LSI

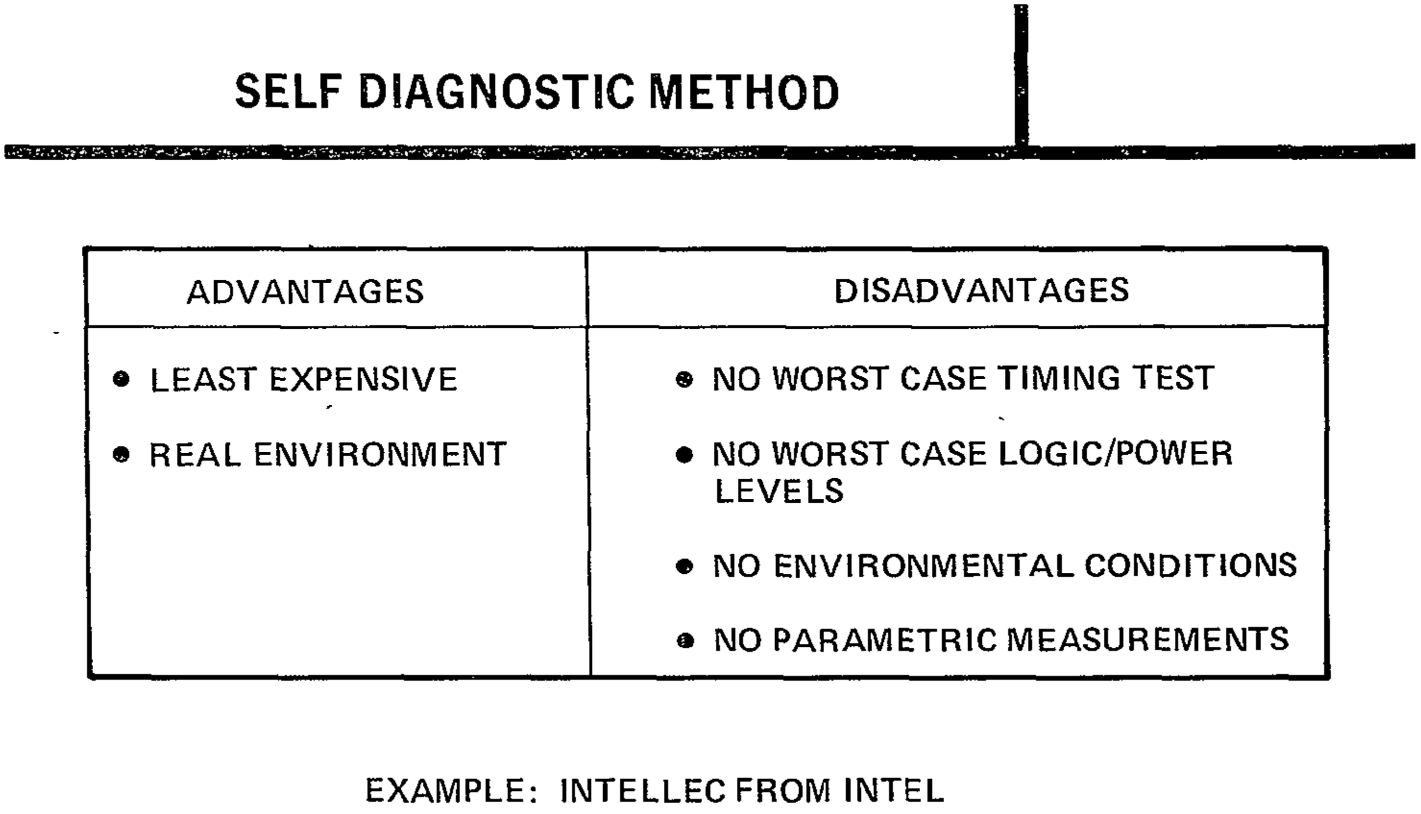
QUAL TEST TASKS	RELATIVE EFFORT GATE	 MEMORY	 MICROPROCESSOR
• VERIFY DUT FUNCTIONS VS DOCUMENTATION	MINOR	MODERATE	MAJOR
• DETERMINE DUT OPERATIONAL BOUNDARIES	MINOR	MAJOR	MAJOR
• DETERMINE QUAL TEST PROCEDURES (I E , TEST PATTERNS, CONDITIONS, LIMITS, ETC)	MINOR	MAJOR	MAJOR
• PERFORM QUAL TESTS (ELECTRICAL, ENVIRONMENTAL, LIFE, ETC)	MAJOR	MODERATE	MINOR
• FAILURE ANALYSIS	MINOR	?	?

Slide 1

The first step in evaluating the electrical characteristics of a microprocessor is to select a test method (i. e. , a method of generating and applying large complex test patterns). Most of the commonly discussed microprocessor test methods can be quickly summarized under one of the following five categories: Self-Diagnostic, Comparison, Algorithmic Pattern Generation, Learned Stored Response, and Predicted Stored Response. A brief description with advantages and disadvantages of each method follows.

Self-Diagnostic Method

The self-diagnostic test usually consists of running an application-oriented program on the microprocessor in its natural environment. The environment may be achieved by either plugging the device under test (DUT) into the system or into a bench-top chip analyzer such as the "Intellec" from Intel or the "EXORciser" from Motorola. Although the chip analyzers are primarily system development tools, they can provide a "dead or alive" functional test at the instruction level. A summary of the advantages and disadvantages of the method is shown in Slide 2.



SELF DIAGNOSTIC METHOD

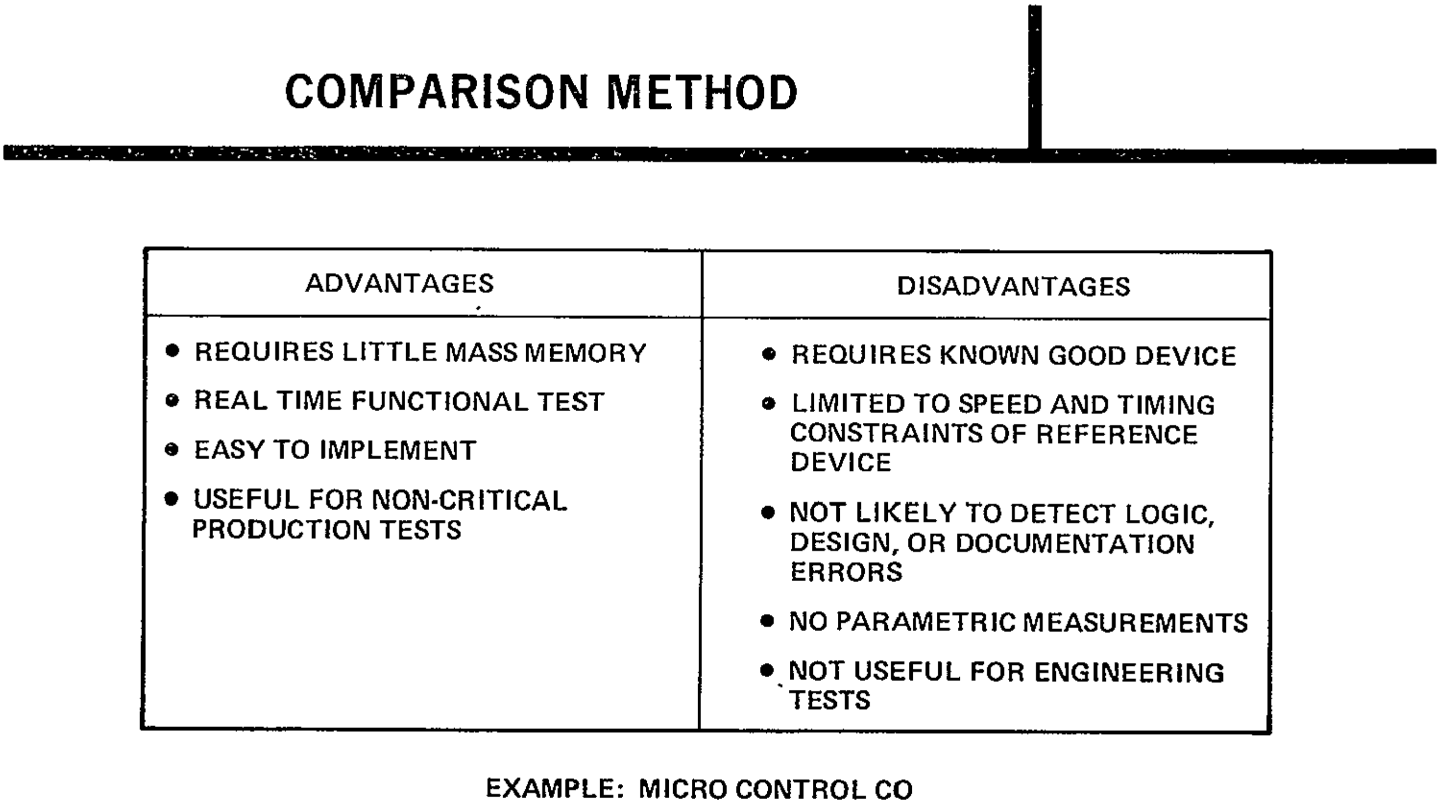
ADVANTAGES	DISADVANTAGES
• LEAST EXPENSIVE • REAL ENVIRONMENT	• NO WORST CASE TIMING TEST • NO WORST CASE LOGIC/POWER LEVELS • NO ENVIRONMENTAL CONDITIONS • NO PARAMETRIC MEASUREMENTS

EXAMPLE: INTELLEC FROM INTEL

Slide 2

Comparison Method

The comparison test consists of building a test box that provides parallel input data to a "known good device" and the DUT. The output signals of both devices are then compared. If the two output signals match, the DUT is passed: but if the two output signals differ for any reason, the DUT is considered a failure. See Slide 3.



COMPARISON METHOD

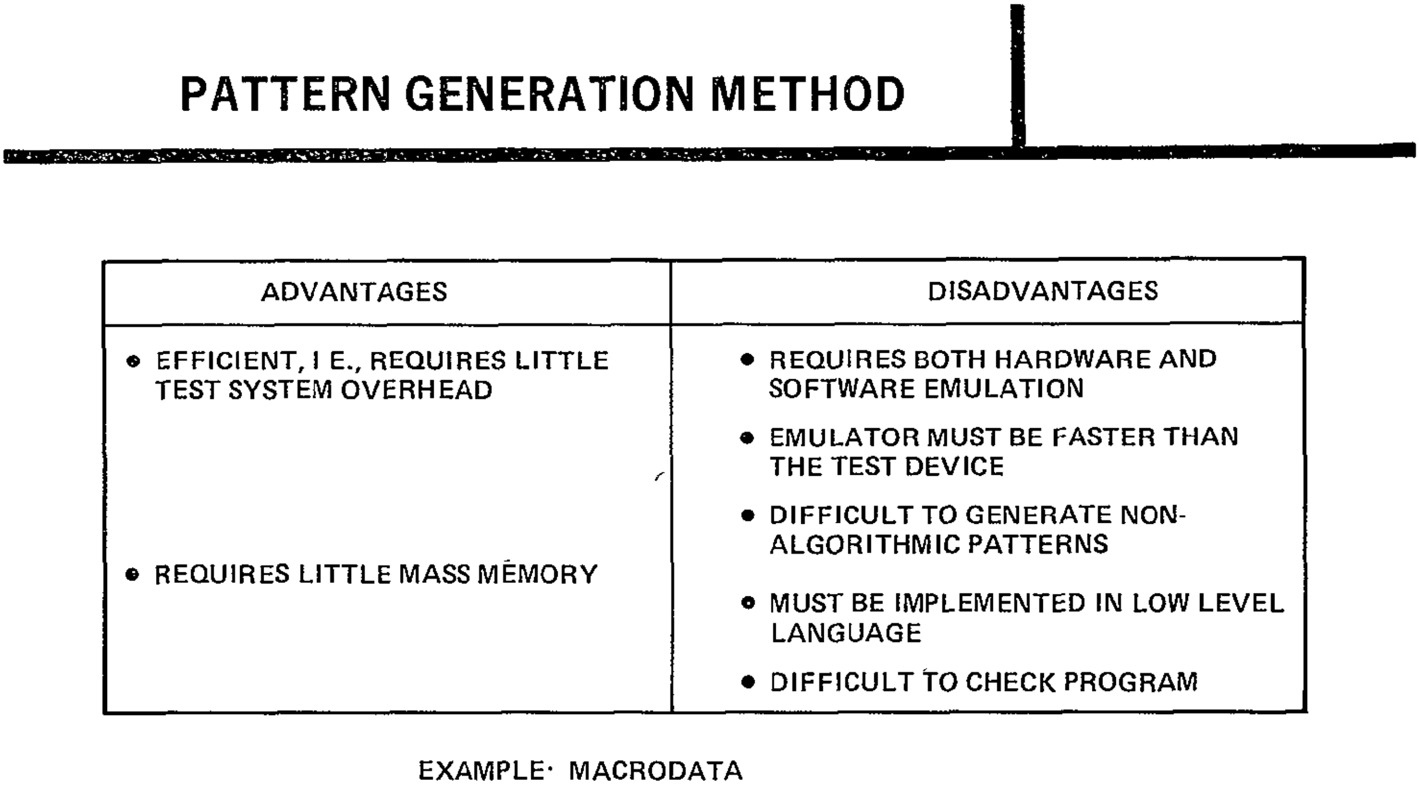
ADVANTAGES	DISADVANTAGES
• REQUIRES LITTLE MASS MEMORY	• REQUIRES KNOWN GOOD DEVICE
• REAL TIME FUNCTIONAL TEST	• LIMITED TO SPEED AND TIMING CONSTRAINTS OF REFERENCE DEVICE
• EASY TO IMPLEMENT	• NOT LIKELY TO DETECT LOGIC, DESIGN, OR DOCUMENTATION ERRORS
• USEFUL FOR NON-CRITICAL PRODUCTION TESTS	• NO PARAMETRIC MEASUREMENTS
	• NOT USEFUL FOR ENGINEERING TESTS

EXAMPLE: MICRO CONTROL CO

Slide 3

Algorithmic Pattern Generation Method

The algorithmic pattern generation technique consists of simulating either the complete microprocessor or a small section of the microprocessor in a hardware pattern computer. The input and output patterns are then generated during the functional test and compared with those from the DUT. The method may be exceptionally complex, as in the case of full hardware simulation, or relatively simple, as in the case of partial or sectional simulation. See Slide 4.



PATTERN GENERATION METHOD

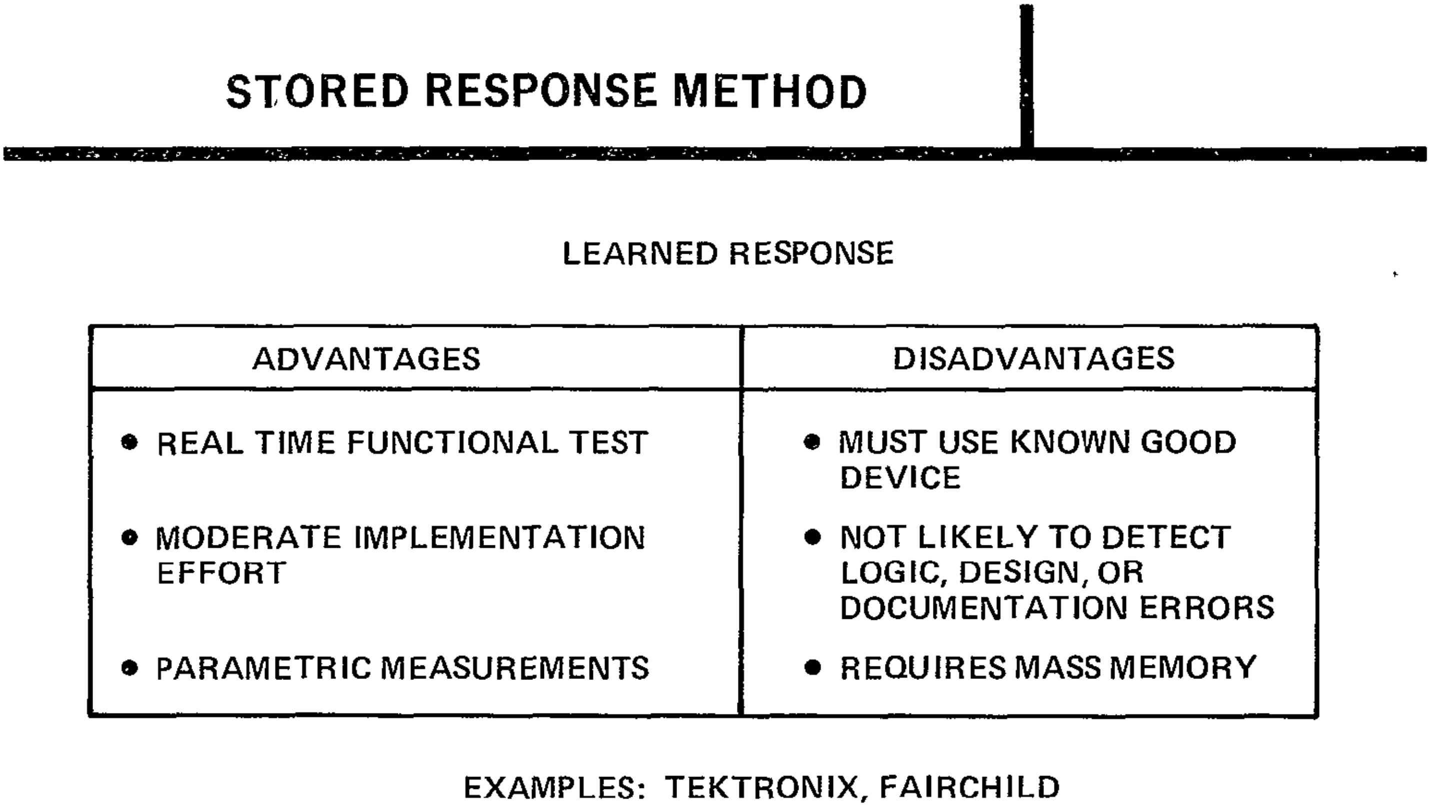
ADVANTAGES	DISADVANTAGES
• EFFICIENT, I E., REQUIRES LITTLE TEST SYSTEM OVERHEAD • REQUIRES LITTLE MASS MEMORY	• REQUIRES BOTH HARDWARE AND SOFTWARE EMULATION • EMULATOR MUST BE FASTER THAN THE TEST DEVICE • DIFFICULT TO GENERATE NON-ALGORITHMIC PATTERNS • MUST BE IMPLEMENTED IN LOW LEVEL LANGUAGE • DIFFICULT TO CHECK PROGRAM

EXAMPLE· MACRODATA

Slide 4

Stored Response Methods

The stored response techniques can be implemented with either a predicted pattern or learned pattern. With either technique, the DUT is tested with a complex pattern (i. e. , at the cycle level) stored on bulk memory. To use the learned response technique, the test engineer codes the input instructions and uses a "known good device" to learn the output responses. Both input and output responses are saved on bulk memory to be used later on the test device. See Slide 5.



STORED RESPONSE METHOD

LEARNED RESPONSE

ADVANTAGES	DISADVANTAGES
● REAL TIME FUNCTIONAL TEST	● MUST USE KNOWN GOOD DEVICE
● MODERATE IMPLEMENTATION EFFORT	● NOT LIKELY TO DETECT LOGIC, DESIGN, OR DOCUMENTATION ERRORS
● PARAMETRIC MEASUREMENTS	● REQUIRES MASS MEMORY

EXAMPLES: TEKTRONIX, FAIRCHILD

Slide 5.

The predicted response technique does not require a "known good device" for assembling the test pattern. Methods such as Logic Simulation, Functional Simulation, or Manual techniques are used to predict the input and output responses required for assembling the test pattern. Once the test pattern is complete, the remaining procedures are the same for predicted and learned stored responses.

Logic Simulation involves building a software program which simulates the gate level schematic of the circuit. Then one of several fault isolation techniques is used to generate patterns which check each circuit node for stuck at one or stuck at zero conditions. A summary of advantages and disadvantages of this technique is shown in Slide 6.

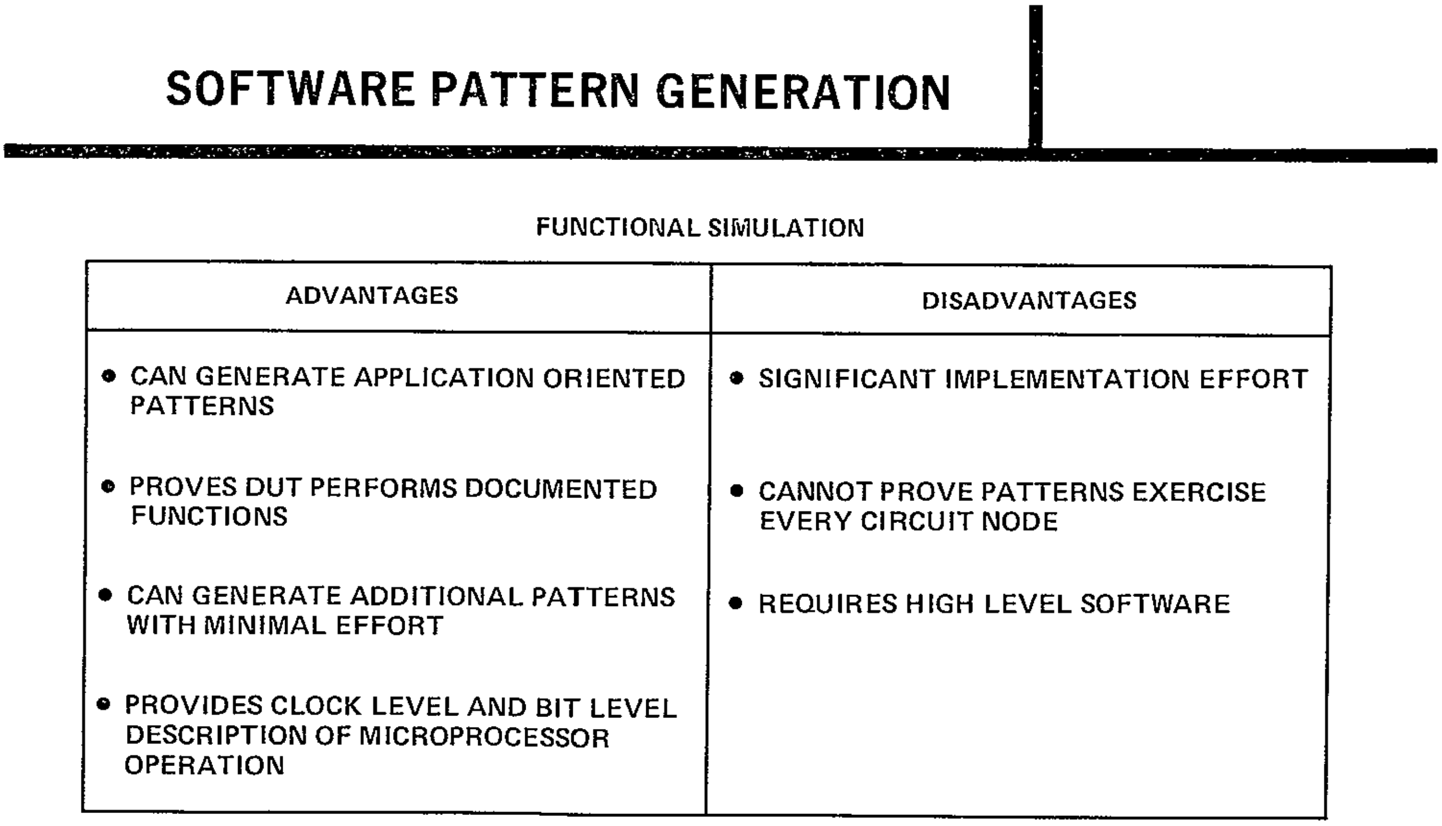
SOFTWARE PATTERN GENERATION

LOGIC SIMULATION

ADVANTAGES	DISADVANTAGES
• WILL THEORETICALLY COVER 99% OF THE DUT's CIRCUIT NODES	• DOES NOT PROVE THAT DUT PERFORMS DOCUMENTED FUNCTIONS
• WILL PROVE DUT MATCHES CIRCUIT SCHEMATIC	• REQUIRES THE CORRECT CIRCUIT SCHEMATIC
• WILL USUALLY PRODUCE SHORT EFFICIENT PATTERN	• CANNOT BE USED TO GENERATE APPLICATION ORIENTED PATTERN
	• SIGNIFICANT IMPLEMENTATION EFFORT

Slide 6

Functional Simulation involves writing a software pattern generator which simulates the DUT's functional description in order to predict output responses. The generator includes a function simulator which performs a detailed simulation of each microprocessor instruction in accordance with published instruction definition. It predicts the input code and output response, including interrupts, wait states, and hold states, on a clock cycle basis. As the pattern is being coded, the pattern generator requests source register, destination register, and data content. Upon receiving a request for status, the pattern generator displays the operational status of each device register, program counter, stack pointer, condition bits, and other relevant information. A summary of the advantages and disadvantages of the technique is shown in Slide 7.



SOFTWARE PATTERN GENERATION

FUNCTIONAL SIMULATION

ADVANTAGES	DISADVANTAGES
• CAN GENERATE APPLICATION ORIENTED PATTERNS	• SIGNIFICANT IMPLEMENTATION EFFORT
• PROVES DUT PERFORMS DOCUMENTED FUNCTIONS	• CANNOT PROVE PATTERNS EXERCISE EVERY CIRCUIT NODE
• CAN GENERATE ADDITIONAL PATTERNS WITH MINIMAL EFFORT	• REQUIRES HIGH LEVEL SOFTWARE
• PROVIDES CLOCK LEVEL AND BIT LEVEL DESCRIPTION OF MICROPROCESSOR OPERATION	

Slide 7

The second step or phase in QUAL testing microprocessors is the electrical characterization of the device. Characterization includes the performance of functional and parametric tests while changing timing, voltage loads, test patterns, temperature, and other variables which may affect device performance. These tests are used to define device operational boundaries and limits. They are also used to define and prove the effectiveness of developed test patterns, worst case test conditions, and test procedures for parametric measurements. Since it is usually not practical to include all combinations of instructions and data, it may be necessary to supplement both pattern and test conditions with those that are application oriented.

Perhaps the most valuable benefit from the characterization phase is the opportunity to identify device and test procedure problems in an informal setting prior to the start of QUAL test measurements. Microprocessors are complex devices which require a complex test system and complex test programs (test procedures) in order to perform meaningful tests. It is unlikely that these three elements may be successfully combined into a reliable qualification test program without encountering at least a few significant problems. It only takes two or more parameters operating at or near operational limits of either the device or the test system to produce unreliable test results (i. e. , results which are often not repeatable). The prompt resolution of these problems cannot be accomplished without thorough knowledge of the test device, test system and applicable test program. Thus, device characterization has become an integral part of qualification tests whether planned or unplanned.

The third phase of QUAL testing microprocessors includes the traditional electrical performance measurements and a variety of mechanical and environmental tests. While this phase represents the major effort during the qualification of SSI type devices, it represents a relatively minor effort during the qualification testing of microprocessors.

A Summary of microprocessor test considerations is shown in Slide 8.

SUMMARY

- QUAL TESTING IS A THREE PHASE EFFORT
- SELECT TEST METHOD SUITABLE FOR MULTIPLE APPLICATIONS
- TEST PATTERNS MUST BE DOCUMENTED AND COMMENTED TO BE USEFUL BY OTHERS
- SUPPLEMENT BASIC TEST PATTERN WITH APPLICATION ORIENTED PATTERNS
- SUPPLEMENT BASIC TEST CONDITIONS WITH APPLICATION ORIENTED CONDITIONS
- START MICROPROCESSOR QUALIFICATION PRIOR TO FINAL SYSTEM DESIGN

Slide 8

HI-REL PROCUREMENT SPECIFICATIONS FOR MICROPROCESSOR AND LSI MEMORY CIRCUITS

John D. Shea
Senior Staff Engineering
Integrated Circuit Engineering Corporation
Santa Clara, California

With the complexity of MPU and LSI devices that will be utilized in military and aerospace applications both for satellite as well as manned space flight it will become increasingly important that both prime and subcontractors with systems responsibility will have all the necessary tools to effectively design a meaningful procurement specification. Where MIL-M-38510 slash sheets exist this would be the ideal method of procurement either to level B or level A depending upon the criticality of the application and overall mission performance. In most cases relevant to MPU's and complex LSI memory devices, slash sheets are just starting to come off the drawing board at RADC and in many cases haven't even been preliminarily laid out. In these cases it requires skillful specsmanship on the part of the procuring activity, whether it be the prime program office of a DOD agency or a prime program procurement function at a prime or subcontractor.

As soon as the OEM merchant market started to execute a growth trend in most situations many of the semiconductor manufacturers tended to take the emphasis that had been placed earlier on military aerospace business and place those resources and capacities at the disposal of the OEM merchant market place. It is evident that in generating Hi-Rel procurement specifications we must view the following items:

1.0 Review of Hi-Rel Requirements:

1.1 Package Types

1.2 Screening Levels

1.3 Control Line vs 883 vs MIL-M-38510 to customer spec requirements

1.4 In-house screening vs vendor vs outside test labs

2.0 Understanding Vendor Hi-Rel Processing Flow (Problem and Games):

2.1 Spec Review

2.2 Response to RFQ or RFP

2.3 Order Entry

2.4 Customer Service

2.5 Hi-Rel Lot Formation

2.6 Hi-Rel Production Control

2.7 Hi-Rel Manufacturing (U.S. assembly vs offshore)

2.8 Hi-Rel QA/RA

2.9 Hi-Rel Test and Screening

2.10 Hi-Rel Failure Analysis

3.0 Vendor Selection and Cost Effective Procurement:

3.1 Identifying Available Vendors

3.2 Selecting the Right Vendor

3.2.1 Use of Checklist

3.2.2 Use of Survey Team

3.3 Negotiating the Cost Savings Contract

3.4 Second Sourcing

4.0 Post Order Follow-Up:

4.1 Contract Review

4.2 Detail Interface Between Customer and Vendor

It requires a team effort both in the areas of procurement, quality and reliability assurance, engineering and manufacturing to effectively generate meaningful Hi-Rel procurement specifications. It is also key that when the source surveillance and qualification take place that this team be made up again of reliability and quality assurance, procurement manufacturing and engineering personnel knowledgeable of semiconductor devices and the applications in their perspective systems.

LOW COST, FUNCTIONAL APPROACH TO MICROPROCESSOR TESTING

by V V. Nickel and P. A Rosenberg

Questron Corporation
El Segundo, California

1. Summary

This presentation outlines the methodology being utilized to establish the functional integrity of the General Processor Unit (GPU), fabricated by RCA for the Air Force. The GPU is a new 8-bit processor bit slice with 16 dual access registers. The concept and utilization of low-cost functional testing, as outlined in this presentation, is not limited to processor bit slice LSI types, but is applicable to any new, functionally complex device that is to be verified. The key element in low-cost functional testing is the automatic generation of test vectors and functional testing of an LSI device, "off-line" from an expensive full-blown LSI tester utilizing a low-cost test stand which has been designed specifically to facilitate automatic test vector generation and functional testing. Vectors required for AC parametric testing and eventual production line testing are also generated on this low-cost test stand, then transferred to the expensive LSI tester or production line tester to do the actual parametric characterization. This off-loads the expensive tester of all test vector generation functions, leaving only actual AC parametric testing. With the extremely high gate to pin ratios of LSI devices, it is functional test generation that consumes most of a tester's resources, while the cost of AC parametric characterization of LSI devices is becoming proportionally less. Unfortunately, in general, LSI testers on the market do not provide adequate tools for functional test generation; therefore, generating and running functional tests and generating AC tests on a low-cost facility, which is designed specifically to facilitate automatic test vector generation, results in more efficient utilization of the expensive LSI tester and, thus, a more economical verification program.

The total number of possible test states for the GPU is astronomical (2^{180}), therefore, the tests are being designed according to a theoretical fault model to limit the total number of test vectors required to achieve an acceptable level of fault coverage. A "functionally" modified nearest neighbor model is being utilized to guide the development of tests for the GPU. It is envisioned that approximately 1,000,000 test vectors will be generated using this model. The structure and organization of the tests is also important in approaching an acceptable level of fault coverage for the GPU. Not only are GPU tests structured to verify individual sub-functions on the device, but are also organized in a hierarchial structure in which previously tested sub-functions are utilized to test other sub-functions. The design of this hierarchial structure, such that all sub-function interfaces are adequately tested, is crucial. Relying solely on the commonly espoused heuristic argument that good fault coverage is achieved by testing individual sub-functions can have disastrous results if sub-function interfaces are not considered. A hierarchial test structure is also required if fault isolation and diagnosis are to be attempted

Because of the large number of test vectors envisioned to be required to functionally verify the GPU, a sophisticated support software capability was developed to automatically generate test vectors. This support software consists of a GPU Microprogram Assembler, GPU Simulator and GPU Exerciser (Xrciser)—all written in PLM (a dialect of PL/1) and resident on an in-house Intel MDS-800 microcomputer system. The GPU Microprogram Assembler and GPU Simulator work in conjunction to automatically generate test vectors. The GPU Xrciser is a powerful CRT console based interactive support package which runs the tests and provides features which facilitate fault isolation and diagnosis. Figure 1.1 illustrates the structure of the MDS-800 microcomputer system which serves as a low-cost, functional test stand. The GPU is connected directly to a parallel input/output port and is exercised under direct program control. The system includes a line printer, dual floppy disk and a CRT console. There are 65K bytes of memory in the Intel 8080 based MDS-800 mainframe.

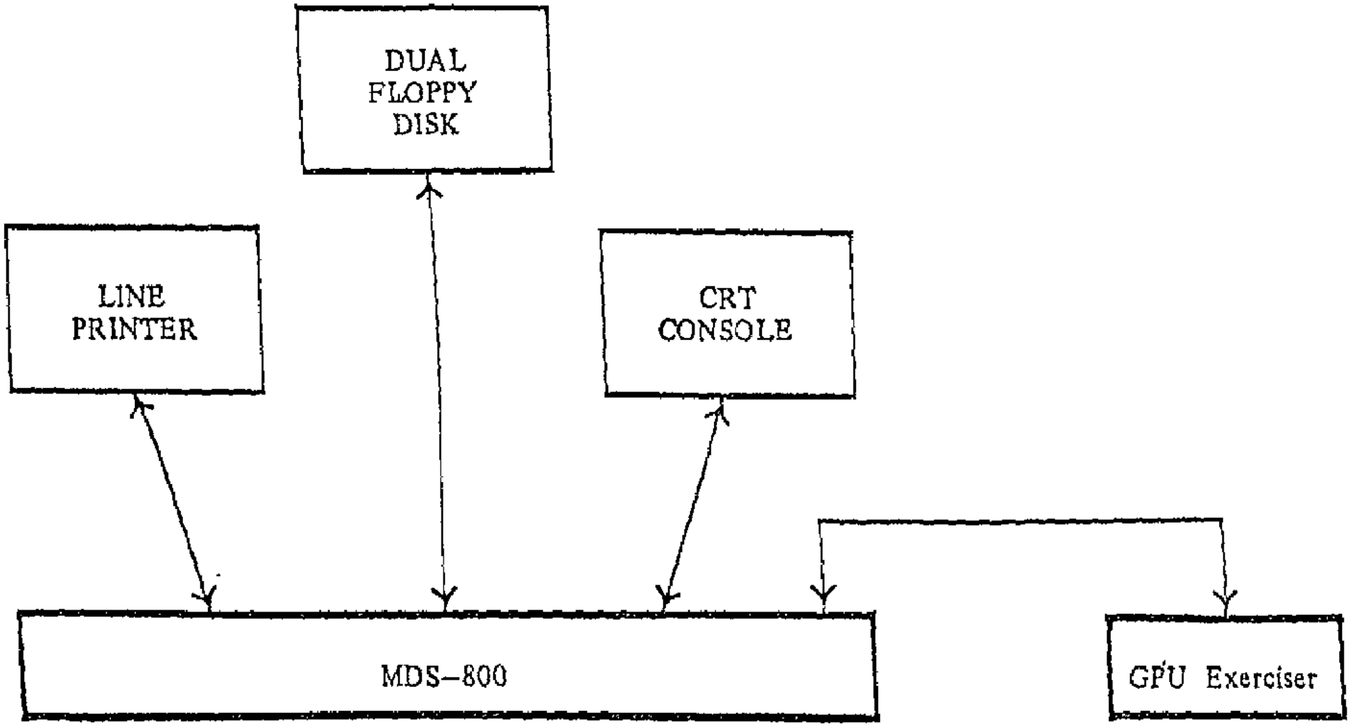


Figure 1.1 MDS–800 Microcomputer Development System

2. GPU Summary of Characteristic

To place this presentation in perspective a brief description of the GPU is provided in this section. Table 2.1 summarizes the pertinent characteristics of the GPU. It is a 48-pin device consisting of 2943 transistors of various sizes, in a 43.2 x 10^3mil^2 area; this is an average of 14.7mil^2 per transistor. The GPU is fabricated using the silicon-gate CMOS/SOS process employing only one epitaxial deposition. A single ion implant is used to dope the silicon islands. The channel oxide is formed by the conventional wet-HCL thermally grown SiO_2 method.

As illustrated in Figure 2.1 the GPU is an 8-bit processor bit slice consisting of: 16 word by 8 bit dual access register file, two port buffers (P1B, P2B), extensive data type selection multiplexing, arithmetic/logic circuit (ALC) and powerful shifting capability. It has separate 8-bit data input and data output paths. Data can be input directly to the register file, P1B or P2B. Data can be output from P1B, P2B or from the ALC. Output drivers are designed to drive 30pf with a rise and face time of 30ns at 10 volts. The data paths are designed to facilitate the implementation of complex algorithms. For example, two bit multiple data paths consisting of a right shift of one into the ALC and right shift of one or two into the register file are implemented on the chip. Although the device is not yet characterized, preliminary tests indicate it is fast – full cycle (accessing of two operands from the file, operating on them through the ALC and storing the result back to the register file) operations of up to 10MHz have been observed.

Table 2.1 GPU SUMMARY OF CHARACTERISTICS

- 8-bit processor bit slice.
- 16 dual access general purpose registers.
- Single clock operation – ability to access two registers, operate on them and store away result in register file, all in one clock cycle.
- 8-bit parallel arithmetic logic circuit (ALC) with carry look ahead, all-zero detection and overflow indication.
- Data path and control provided for multiple step algorithms such as two-bit multiply, division and floating point.
- Expandable – can concatenate any number of GPU's for larger word size machines.
- Microprogram versatility – can independently control selection of sources, ALC operation and destination of data.
- CMOS/SOS – chip size 201 x 215.
- Static operation.
- Single DC power source 4 Volts to 15 Volts

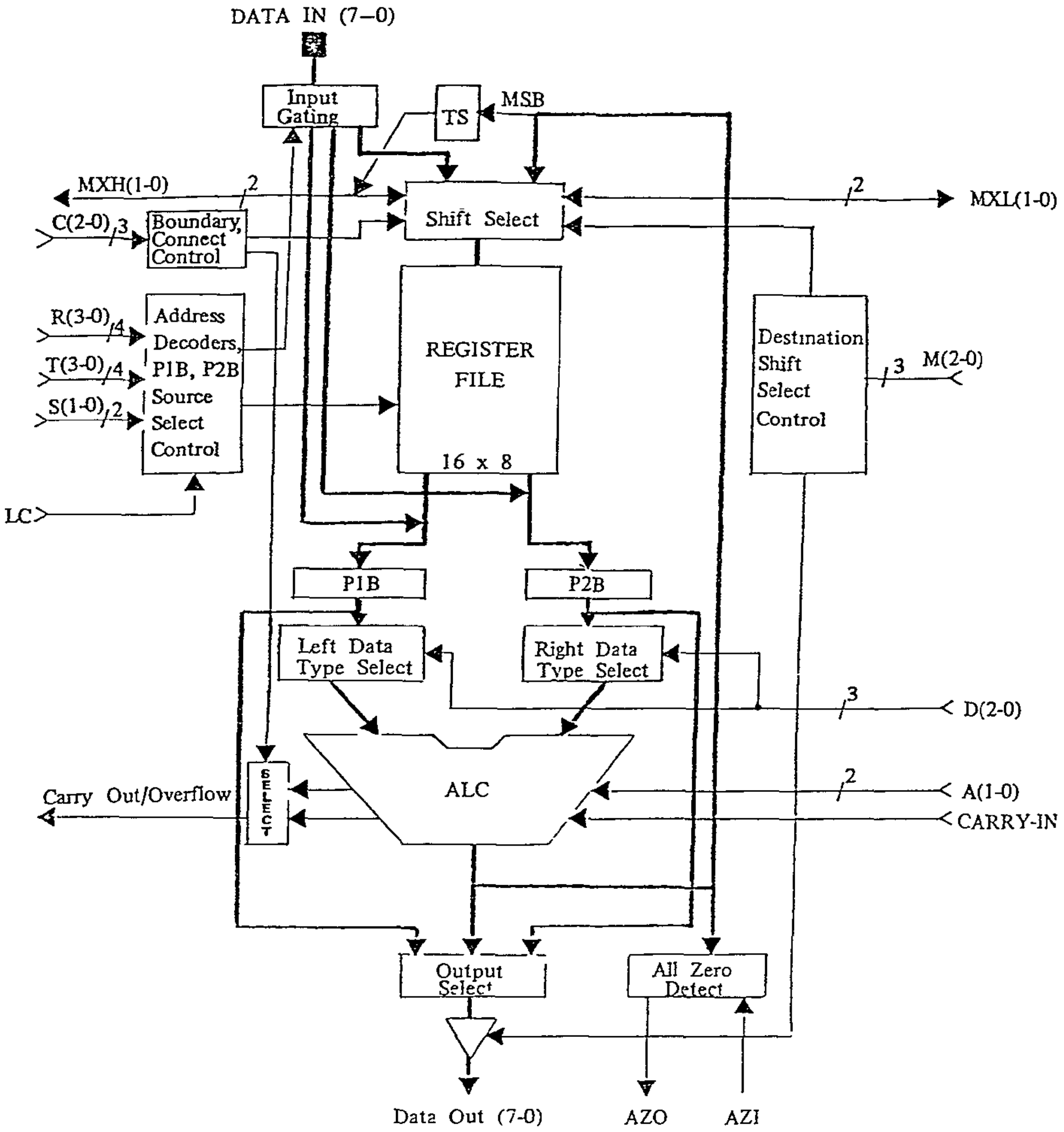


Figure 2.1 GPU ARCHITECTURE

3. Software System

Overview

The GPU will be tested by applying approximately 1,000,000 test vectors. Each test vector consists of a stimulus and an expected response. The stimulus is a set of 38 control and data bits. The responses consist of 14 output data and status bits. The sheer number of test vectors required, to attain an acceptable level of fault coverage, precludes manual generation of binary data. Even symbolic specification on, perhaps, a one line - one vector basis is not practical since it involves the writing of 1,000,000 lines of symbolic code. An even more overwhelming problem would be the manual computation of the expected responses of the GPU.

For these reasons it is evident that a sophisticated test vector generation system is required. Figure 3.1 shows an overview of an automatic test vector generation software system that has been developed at Questron.

The automatic test vector generation software system consists of three software elements. The first element of the system is the GPU Microprogram Assembler. The GPU Microprogram Assembler accepts symbolic microcode written in a register transfer language with many higher level constructs. These higher level constructs allow generation of large numbers of test vectors from very short and simple specifications. The output of the GPU Microprogram Assembler is binary microcode. The binary microcode represents the stimuli that will be applied to the GPU.

The second element of the software system is the GPU Simulator. The GPU Simulator contains a software model of the GPU data paths and storage elements. The model computes the next state and output configuration of the GPU based on its current state and the stimulus applied. As a result the stimuli specified in the binary microcode are augmented with the corresponding expected responses, thus completing the test vectors.

The third element of the software system is the GPU Exerciser software. This program facilitates interactive control of the actual execution of the test and makes the idiosyncratic details of the GPU Exerciser hardware transparent to the test engineer. The GPU Exerciser software package includes many features that facilitate fault isolation.

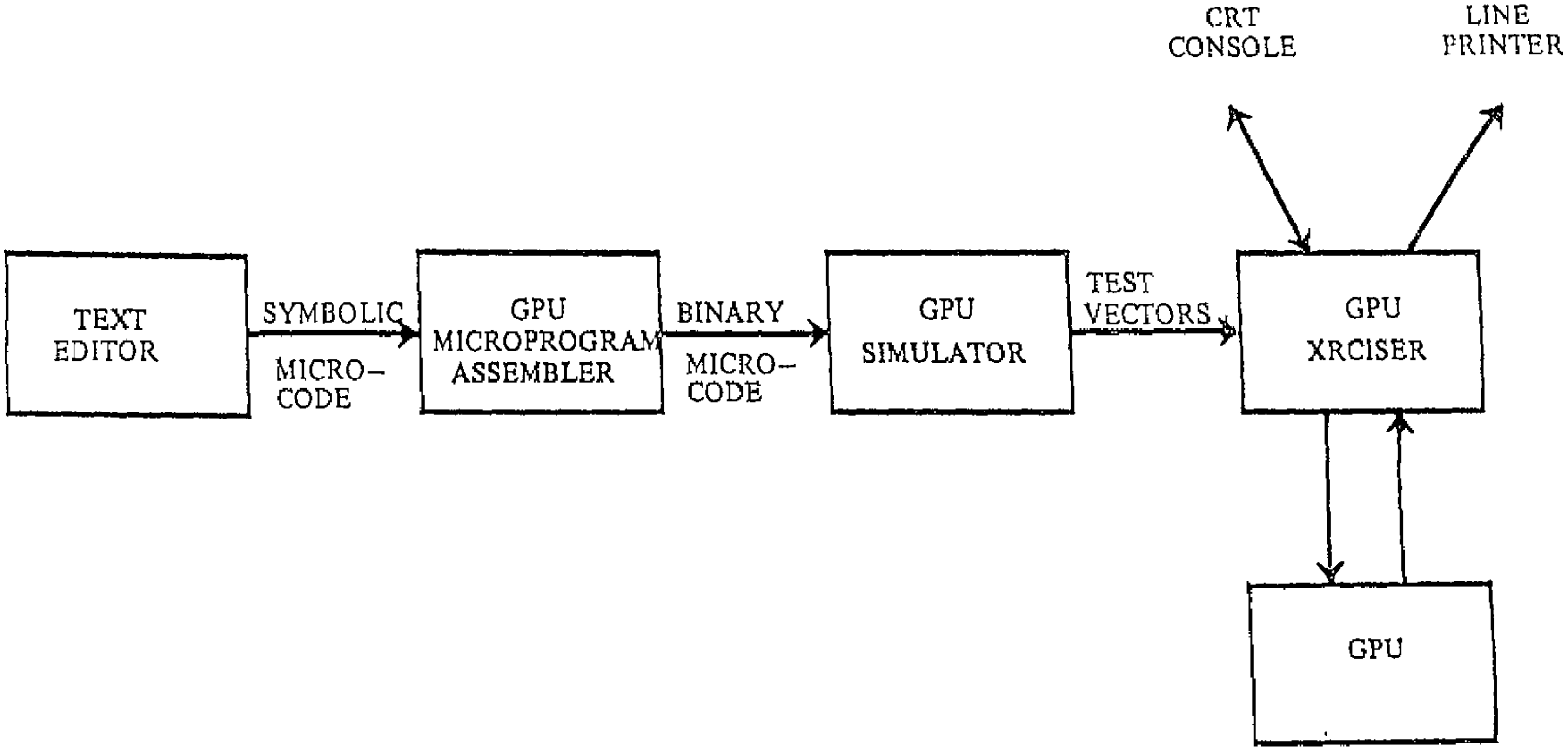


FIGURE 3.1 SYSTEM OVERVIEW

The GPU Microprogram Assembler

The symbolic language accepted by the GPU Microprogram Assembler is basically a register transfer language. Each microword (i.e. set of stimuli) is specified as a series of register transfers, called phrases. Table 3.1 enumerates the phrases accepted by the GPU Microprogram Assembler.

The process of translating these phrases into binary microcode is complicated by two factors: in many cases there is more than one control configuration that will cause the execution of a given phrase, and some phrases are incompatible with others when specified in the same microword. Thus it could happen that two phrases in a microword are apparently incompatible. However if one (or both) were translated differently they would be compatible. The GPU Microprogram Assembler attempts to resolve such cases and will flag an incompatibility only if all possible alternate translations have also resulted in incompatibility in the given context.

In those cases where more than one translation is valid the test designer has the option to influence the assembler's choice by specifying some control fields explicity. This can be regarded as a lower level feature of the language.

The most significant features of the GPU Microprogram Assembler language are the higher level constructs. The most important of these are the iterative repetitions. Table 3.2 shows their syntax.

The effect of an iterative repetition specification is that the code included between REPEAT and ENDREPEAT will be translated repeatedly substituting values as specified for the loop variable (R, T or DI). Repeats can be nested.

Other higher level constructs allow initialization of all storage elements of the GPU and display of the contents of the register file.

Figure 3.2 illustrates the power of these higher level constructs. Here 7 lines of code generate 4370 (1112_{16}) microwords. (This example is not extreme because a third level of nesting could have been used.)

TABLE 3.1 GPU MICROPROGRAM ASSEMBLER LANGUAGE

Phrases	Default
1) TS = ALC (7),	TS = TS
2) REG(n) = ALC, REG(n) = SHIFTER,	REG(n) = ALC

3) P1B = REG(n),
P1B = REG(n+1),
P1B = DI,
	P1B = REG(0)

4) P2B = P2B,
P2B = REG(n),
P2B = DI,
	P2B = REG(0)

5) DI = xx,	DI = 00

6) DO = OFF,
DO = P1B,
DO = P2B,
DO = ALC,
	D0 = OFF

7) ALC = *left* + *right*
ALC = *left* + *right* + 1
ALC = *left* – P2B
ALC = *left* OR *right*
ALC = *left* AND *right*
	ALC = left + right

left is one of: 0 P1B MXH1IN\\P1B(7-1) ·NOT·P1B	left = P1B
right is one of: 0 P2B ·NOT·P2B	right = 0

TABLE 3.1 (cont.)

Phrases	Default
8) COUT = OFL	COUT = COUT
9) AZIN = x	AZIN = 1

10) SHIFTER = DI,
SHIFTER = ALC(6-0)\\0,
SHIFTER = ALC(6-0)\\1,
SHIFTER = ALC(6-0)\\MXLOIN,
SHIFTER = 0\\ALC(7-1),
SHIFTER = 1\\ALC(7-1),
SHIFTER = ALC(7)\\ALC(7-1),
SHIFTER = MXHOIN\\ALC(7-1),
SHIFTER = 00\\ALC(7-2),
SHIFTER = 11\\ALC(7-2),
SHIFTER = ALC(7)\\ALC(7)\\ALC(7-2),
SHIFTER = MXHOIN\\MXH1IN\\ALC(7-2),
	SHIFTER = DI

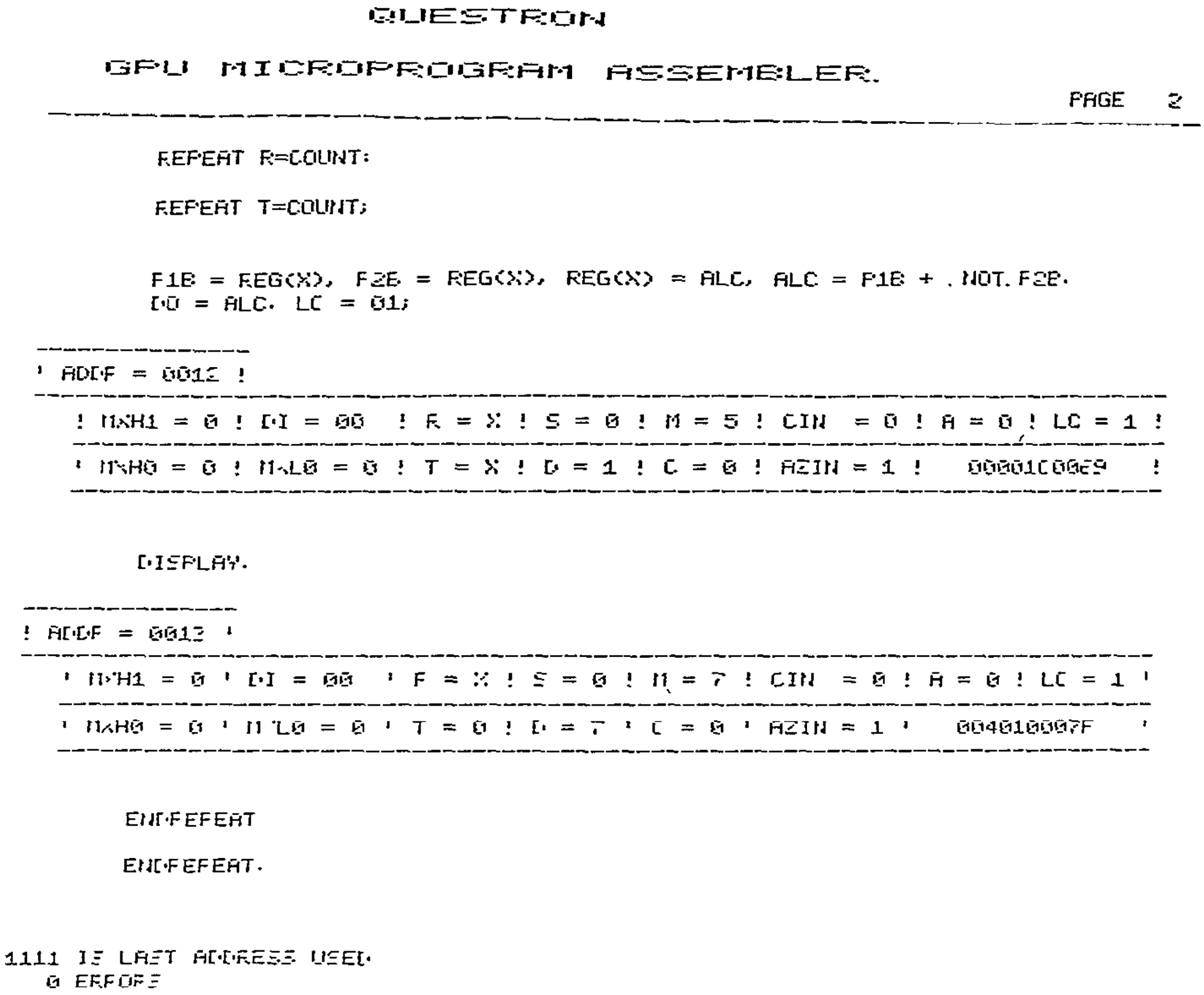
11) MXL0IN = x,	MXL0IN = 0
12) MXH0IN = x,	MXH0IN = 0
13) MXH1IN = x,	MXH1IN = 0
14) MXL0OUT = ALC(0),	input
15) MXL1OUT = ALC(1), MXL1OUT = P1B(0),	OFF
16) MXH0OUT = ALC(7),	input

17) MXH1OUT = ALC(6),
MXH1OUT = TS,
MXH1OUT = P2B(7),
	input

18) LC = 00,
LC = 01,
LC = 10,
	LC = 00

; instead of , ends the microword.

[END] ends the program.



```
                         QUESTRON

      GPU  MICROPROGRAM  ASSEMBLER.
                                                                PAGE   2
 ----------------------------------------------------------------------

        REPEAT R=COUNT:

        REPEAT T=COUNT;

        F1B = REG(X), F2B = REG(X), REG(X) = ALC, ALC = F1B + .NOT.F2B.
        DO = ALC. LC = 01;

 ---------------
 ! ADDR = 0012 !
 ------------------------------------------------------------------------
   ! MXH1 = 0 ! DI = 00  ! R = X ! S = 0 ! M = 5 ! CIN  = 0 ! A = 0 ! LC = 1 !
   ------------------------------------------------------------------------
   ! MXH0 = 0 ! MXL0 = 0 ! T = X ! D = 1 ! C = 0 ! AZIN = 1 !   00001C00E9   !
   ------------------------------------------------------------------------


       DISPLAY.

 ---------------
 ! ADDR = 0013 !
 ------------------------------------------------------------------------
   ! MXH1 = 0 ! DI = 00  ! F = X ! S = 0 ! M = 7 ! CIN  = 0 ! A = 0 ! LC = 1 !
   ------------------------------------------------------------------------
   ! MXH0 = 0 ! MXL0 = 0 ! T = 0 ! D = 7 ! C = 0 ! AZIN = 1 !   004010007F   !
   ------------------------------------------------------------------------


       ENDREPEAT

       ENDREPEAT.


1111 IS LAST ADDRESS USED.
   0 ERRORS
```

FIGURE 3.2 EXAMPLE OF HIGHER LEVEL CONSTRUCT USAGE

Table 3.2 Iterative Repetitions

$$\text{REPEAT}\begin{Bmatrix}\text{DI}\\ \text{R}\\ \text{T}\end{Bmatrix} = \begin{Bmatrix}\text{WALK 0}\\ \text{WALK 1}\\ \text{COUNT}\end{Bmatrix};$$

. . . . any number of microinstructions

. . . .

ENDREPEAT;

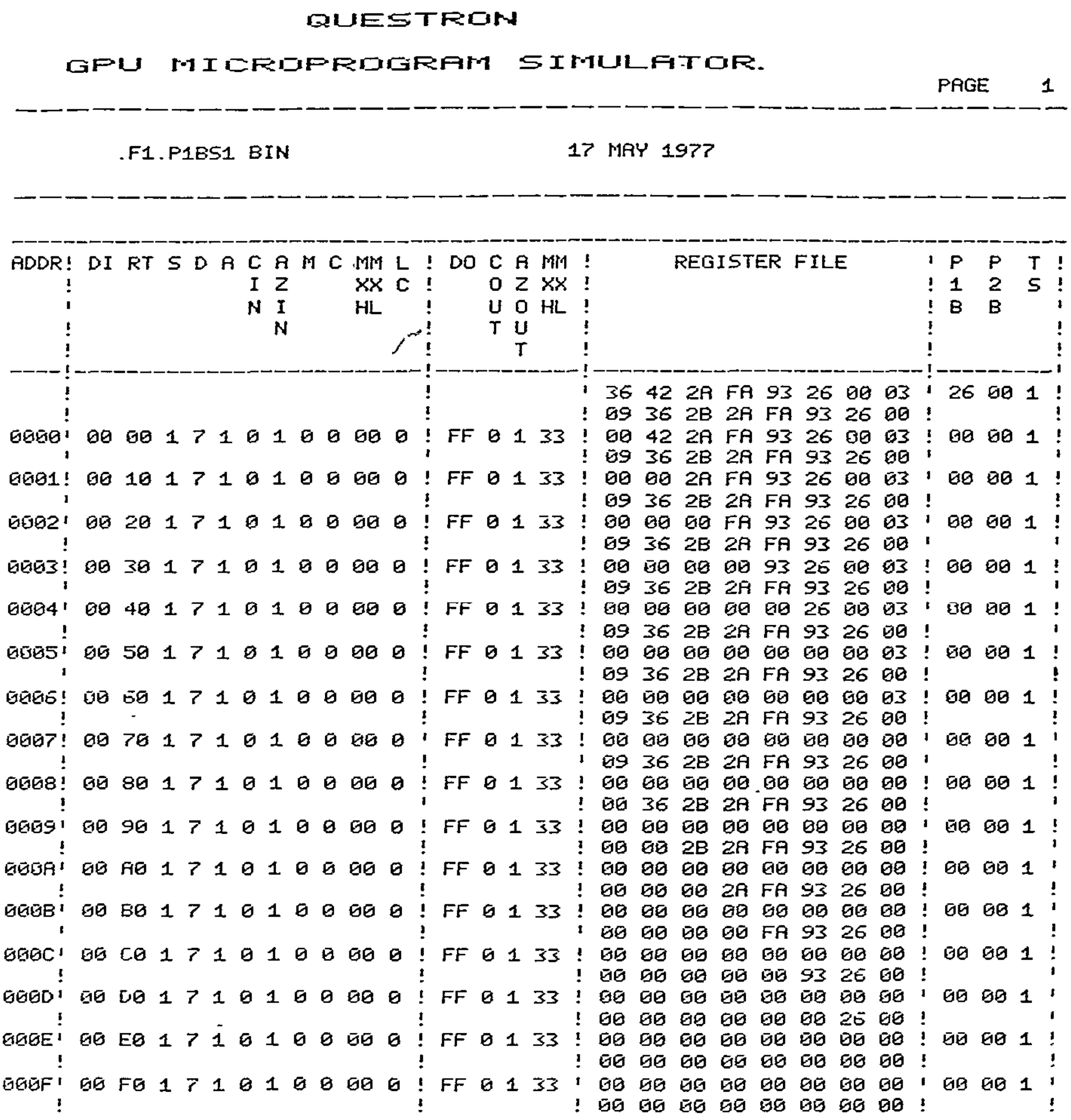
Within repetitions registers may be referenced as REG(X). Varying values of R or T (as determined by context) will be substituted. DI = XX indicates that the value is to be determined by the REPEAT instruction.

GPU Simulator

The GPU Simulator operates on the binary microcode produced by the GPU Microprogram Assembler. It generates the next state and the expected outputs of the GPU based on the current state and the current input vector (stimulus). The expected outputs are used to augment the input vector thus creating complete test vectors.

The most important part of the simulation program is the model of the GPU. This model is implemented at the element device level, as opposed to the gate or transistor level. For example, a multiplexer is simulated with a select case construct. The model produces two-valued output (i.e. 0 or 1 and not 4.7V or don't care.) No time delays are simulated.

Figure 3.3 shows a sample of the printed output from the GPU Simulator. The leftmost column contains addresses which can be used to key to GPU Microprogram Assembler of GPU Exerciser listings. The next two columns represent the contents of the test vector, stimuli and expected responses, respectively. The last two columns represent the assumed contents of all GPU storage elements. This latter information is not part of the test vector, but knowing what the simulator assumes the contents of storage elements to be is an important testing aid.



```
                    QUESTRON
    GPU MICROPROGRAM SIMULATOR.
                                                                        PAGE    1
--------------------------------------------------------------------------------

          .F1.P1BS1 BIN                      17 MAY 1977

--------------------------------------------------------------------------------

--------------------------------------------------------------------------------
ADDR! DI RT S D A C A M C MM L ! DO C A MM !     REGISTER FILE         ! P  P  T !
    !                 I Z     XX C !    O Z XX !                         ! 1  2  S !
    !                 N I     HL   !    U O HL !                         ! B  B    !
    !                   N          !    T U    !                         !         !
    !                              !      T    !                         !         !
----!------------------------------!-----------!-------------------------!---------!
    !                              !           ! 36 42 2A FA 93 26 00 03 ! 26 00 1 !
    !                              !           ! 09 36 2B 2A FA 93 26 00 !         !
0000! 00 00 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 42 2A FA 93 26 00 03 ! 00 00 1 !
    !                              !           ! 09 36 2B 2A FA 93 26 00 !         !
0001! 00 10 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 2A FA 93 26 00 03 ! 00 00 1 !
    !                              !           ! 09 36 2B 2A FA 93 26 00 !         !
0002! 00 20 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 FA 93 26 00 03 ! 00 00 1 !
    !                              !           ! 09 36 2B 2A FA 93 26 00 !         !
0003! 00 30 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 93 26 00 03 ! 00 00 1 !
    !                              !           ! 09 36 2B 2A FA 93 26 00 !         !
0004! 00 40 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 26 00 03 ! 00 00 1 !
    !                              !           ! 09 36 2B 2A FA 93 26 00 !         !
0005! 00 50 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 03 ! 00 00 1 !
    !                              !           ! 09 36 2B 2A FA 93 26 00 !         !
0006! 00 60 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 03 ! 00 00 1 !
    !                              !           ! 09 36 2B 2A FA 93 26 00 !         !
0007! 00 70 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 00 ! 00 00 1 !
    !                              !           ! 09 36 2B 2A FA 93 26 00 !         !
0008! 00 80 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 00 ! 00 00 1 !
    !                              !           ! 00 36 2B 2A FA 93 26 00 !         !
0009! 00 90 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 00 ! 00 00 1 !
    !                              !           ! 00 00 2B 2A FA 93 26 00 !         !
000A! 00 A0 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 00 ! 00 00 1 !
    !                              !           ! 00 00 00 2A FA 93 26 00 !         !
000B! 00 B0 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 00 ! 00 00 1 !
    !                              !           ! 00 00 00 00 FA 93 26 00 !         !
000C! 00 C0 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 00 ! 00 00 1 !
    !                              !           ! 00 00 00 00 00 93 26 00 !         !
000D! 00 D0 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 00 ! 00 00 1 !
    !                              !           ! 00 00 00 00 00 00 26 00 !         !
000E! 00 E0 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 00 ! 00 00 1 !
    !                              !           ! 00 00 00 00 00 00 00 00 !         !
000F! 00 F0 1 7 1 0 1 0 0 00 0 ! FF 0 1 33 ! 00 00 00 00 00 00 00 00 ! 00 00 1 !
    !                              !           ! 00 00 00 00 00 00 00 00 !         !
```

FIGURE 3.3 SAMPLE OF GPU SIMULATOR LISTING

GPU Exerciser Software

In its simplest use the GPU exerciser allows the test engineer to run a file of test vectors on the GPU and to receive an on-line message on the CRT whereever the actual responses don't correspond to the expected responses. The idiosyncracies of the I/O interface to the GPU are completely transparent.

Table 3.3 summarizes the more advanced features of the GPU Exerciser Software. Output can be directed to CRT, line printer or both. Continuous tracing can be turned on or off, execution can take place in single steps, from beginning to end of file or repeatedly from beginning to end of file. Execution can be interrupted by pressing the INT7 control switch and can thereafter be continued. It can be specified that the test shall be halted after a mismatch or that it shall not be halted. Four breakpoints can be specified. Contents of all storage elements can be displayed non-destructively and storage elements can be initialized from the console at run time. Figure 3.4 is a sample of lineprinter listing from the GPU Exerciser. Here a test ran successfully, tracing listing directed to the lineprinter being specified.

TABLE 3.3 XRCISE SUBCOMMANDS

- OUTPUT CONTROL

  OUTPUT = CRT – ALL OUTPUT TO CRT

  OUTPUT = LP – TRACE OUTPUT TO LP ONLY, SUBCOMMANDS TO CRT AND LP

  OUTPUT = BOTH – ALL OUTPUT TO CRT AND LP

- TRACE CONTROL

  TRACE – DISPLAY STATUS VECTOR FOR EACH MICROWORD EXECUTED

  NOTRACE – DISPLAY STATUS VECTOR ONLY
  – ON MISMATCH
  – ON BREAKPOINT MATCH
  – AFTER *INT7.*

- EXECUTION CONTROL

  STEP OR *null* – EXECUTE ONE MICROINSTRUCTION

  RUN – BEGIN AT CURRENT MICROWORD, STOP AT END OF FILE

  LOOP – BEGIN AT CURRENT MICROWORD, AT END–OF–FILE START FROM BEGINNING–OF–FILE.

  LOOP = n – SAME AS LOOP, STOPS AFTER n–th END–OF–FILE (n = 1 to 5 DECIMAL DIGITS, n < 64K)

  C – CONTINUE AFTER *INT7* BREAKPOINT OR MISMATCH

- STOP CONTROL

  ERRSTOP – STOP ON MISMATCH

  NOERRSTOP – DO NOT STOP ON MISMATCH

  BP0 = XXXX
  BP1 = XXXX
  BP2 = XXXX
  BP3 = XXXX
  } SET ONE OF FOUR BREAKPOINTS (XXXX = FOUR HEX DIGITS)

  NOBP – ALL BREAKPOINTS DEACTIVATED

  INT7 – STOP AFTER CURRENT MICROWORD

- DISPLAY CONTROL AND INITIALIZATION

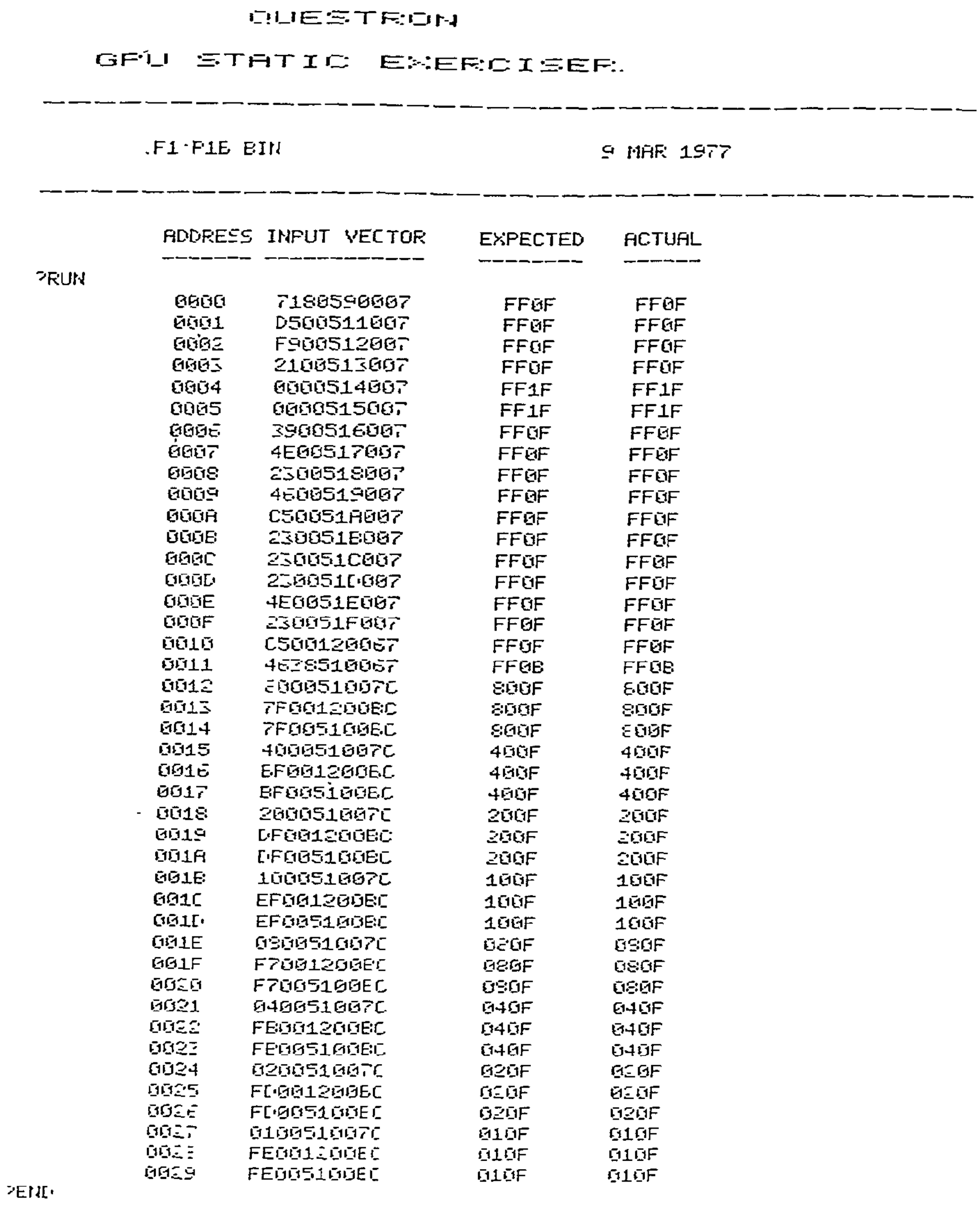
  P1B = XX
  P2B = XX
  REG(n) = XX
  } INITIALIZATIONS

  DISPLAY – DISPLAY ALL REGISTERS

- OTHER SUBCOMMANDS

  REOPEN – NEXT MICROINSTRUCTION TO BE EXECUTED IS AT BEGINNING–OF–FILE

  END – TERMINATE EXERCISE, RETURN TO ISIS–II



```
                 QUESTRON
     GPU  STATIC  EXERCISER.
-----------------------------------------------------------
      .F1·P1E BIN                         9 MAR 1977
-----------------------------------------------------------

         ADDRESS  INPUT VECTOR     EXPECTED    ACTUAL
         -------  ------------     --------    ------
?RUN
          0000     7180590007        FF0F        FF0F
          0001     D500511007        FF0F        FF0F
          0002     F900512007        FF0F        FF0F
          0003     2100513007        FF0F        FF0F
          0004     0000514007        FF1F        FF1F
          0005     0000515007        FF1F        FF1F
          0006     3900516007        FF0F        FF0F
          0007     4E00517007        FF0F        FF0F
          0008     2300518007        FF0F        FF0F
          0009     4600519007        FF0F        FF0F
          000A     C50051A007        FF0F        FF0F
          000B     230051B007        FF0F        FF0F
          000C     230051C007        FF0F        FF0F
          000D     230051D007        FF0F        FF0F
          000E     4E0051E007        FF0F        FF0F
          000F     230051F007        FF0F        FF0F
          0010     C500120067        FF0F        FF0F
          0011     4628510067        FF0B        FF0B
          0012     800051007C        800F        600F
          0013     7F001200BC        800F        800F
          0014     7F005100BC        800F        800F
          0015     400051007C        400F        400F
          0016     BF001200BC        400F        400F
          0017     BF005100BC        400F        400F
        - 0018     200051007C        200F        200F
          0019     DF001200BC        200F        200F
          001A     DF005100BC        200F        200F
          001B     100051007C        100F        100F
          001C     EF001200BC        100F        100F
          001D     EF005100BC        100F        100F
          001E     090051007C        020F        090F
          001F     F7001200EC        080F        080F
          0020     F7005100EC        090F        080F
          0021     040051007C        040F        040F
          0022     FB001200BC        040F        040F
          0023     FB005100BC        040F        040F
          0024     020051007C        020F        020F
          0025     FD001200BC        020F        020F
          0026     FD005100EC        020F        020F
          0027     010051007C        010F        010F
          0028     FE001200EC        010F        010F
          0029     FE005100EC        010F        010F
?END
```

FIGURE 3.4 SAMPLE OF GPU EXERCISER LISTING

4. Testing Approach

Overview

Design and process verification and validation is the purpose of the current phase of GPU testing. As such, the tests are designed to detect problems that may have occurred during each step of the design cycle that begins with a functional specification and ends with an LSI device in hand. Therefore, the tests detect system design errors, logic design errors, electrical design errors; mask layout errors, processing problems, handling and packaging problems. Because of a tight schedule (testing must be completed by June so that a planned manufacturing methods technology program can be initiated), GPU testing is being conducted in two phases; a static exerciser phase and a dynamic exerciser phase. In the static phase test vectors are applied to the GPU, one at a time, under direct program control-therefore the tests are run at the speed of the main frame. In the dynamic exerciser phase, blocks of test vectors are buffered in the dynamic exerciser and executed at full speed – over 10 MHz. Static exerciser testing could be initiated much sooner than dynamic testing (because of its relative simplicity) and it is capable of detecting functional errors (errors in interpretation of the functional specification) which require the longest lead time to ameliorate. Electrical design errors, mask layout errors, processing problems, packaging and handling problems that manifest themselves as simple "stuck ats" or pattern dependent "stuck ats" are also detectable during static testing. Static tests are also designed to check the clock and control decode race conditions in the storage elements. The Dynamic exerciser testing phase verifies GPU operation at speed. In essence, it will rerun tests developed for static testing and allow throughput characterization of the device.

Degree of Test Complexity

One of the first major decisions made on this program was to determine to what level the GPU should be tested and what the resulting approximate number of test vectors would be. An estimate of the number of test vectors was required in order to guide us in specifying the capabilities of the test system. Because the GPU is a new part implemented with an immature technology, that has not been functionally, logically or electrically verified, tests are not only designed to establish the top level functional (TLF) integrity of the device and that there are no node "stuck ats" (NSATS) within the device, but also that there is no interaction between "nearest neighbors" Application of the nearest neighbor fault model takes on two aspects within the verification tests of the GPU; nearest electrical neighbor (NEN) and nearest physical neighbor (NPN). Application of nearest neighbor fault model testing, of course, requires detailed part documentation – logic diagrams, electrical diagrams and masks.

What is meant by nearest electrical neighbor can best be cited by example. Consider a data bus with many drivers that can be individually enabled. While testing one driver, the source data inputs to all the other drivers are placed in the opposite state as that applied to the driver under test. Such a test can determine if there is excessive leakage in transistor nodes. In GPU verification tests, nearest electrical neighbor testing often places the GPU in electrically illegal states, such as, two drivers on simultaneously on a bus. This is done to verify the drive capability of specially sized transistors. Electrical neighbor tests for the GPU also check for race conditions into a common storage element.

Nearest physical neighbor testing verifies that there is no spatial interaction between elements. In its simplest application it entails, for example, utilizing a checkerboard pattern in the register file. However, strict adherence to the model requires detailed analysis of the mask, cataloging all parallel signal runs and cross points. Because of the nature of the layout of the GPU, this is an overwhelming task Therefore, to limit the amount of analysis required, a "functionally" modified nearest physical neighbor fault model is applied. What is meant by this is that by the inherent nature of the device, relative physical location and major parallel signal runs and crossings can be identified by doing a functional analysis. It is possible to take this approach with the GPU because the 8-bit data paths of the device are laid out in parallel, top to bottom, and the major sub-functions along the data path (P1B, P2B, Data Type Select, ALC, Output Shift Select) are more or less, allocated within specific unique areas along the data paths. For example, the three data type select lines (D), refer to Figure 2.1, are easily identifiable as an extremely long parallel signal run with many crossings For critical control signals, such as the load clock (LC), the masks are analysed to determine parallel signal runs and crossings.

Fortunately, in many cases, the tests developed to test nearest electrical neighbor interaction can also test for nearest physical neighbor interaction on the GPU because of the near regular layout and its functional nature. Figure 4.1 presents the results of the initial analysis performed to estimate the number of test vectors required to validate the GPU. As shown in the figure, the difference between top level functional testing node stuck at testing for the GPU is insignificant (both require approximately 10^3 test vectors) However, tests designed to include nearest electrical neighbor interaction bring the total way up to over 10^5 test vectors Including functionally modified nearest physical neighbor testing of the GPU is estimated to add another order of magnitude to the total number of test vectors Therefore, it is expected that approximately 10^6 test vectors will be generated to verify the GPU.

For interest, the analysis was extended to consider what it would take to test a memory of the same complexity (2943 transistors) with the same cell structure and dimensions as the register file on the GPU; a 200-bit dual access memory would be the result. The same significant jump in the number of test vectors occurs when nearest electrical neighbor fault model is used. Note however, that there is no significant increase when testing is extended to include nearest physical neighbor. This is because, with judicious choice of data and test patterns, the same test sequences can be utilized to verify both, therefore the total is almost the same.

The application of the nearest electrical neighbor and functionally modified nearest physical neighbor fault models is necessary to limit the total number of test vectors while maintaining an acceptable level of fault coverage. If these models are adequate, or for that matter, even necessary to attain an acceptable level of fault coverage is not known, only process maturity and research will tell. We are relying on past experience in the application of these models.

The need for a fault model is graphically illustrated by Figure 4 2. If vectors were brute force randomly generated it would take 2^{180} test vectors to assure 100% fault coverage. Accepting 50% coverage only cuts the number required in half to 2^{179} and so forth. Therefore, as illustrated by the bottom graph, intelligent design of tests (based on fault models) results in better fault coverage for the same number of vectors. It should be pointed out that no claims are made based on the slope of the curve in Figure 4.2 – it is there only to make a point.

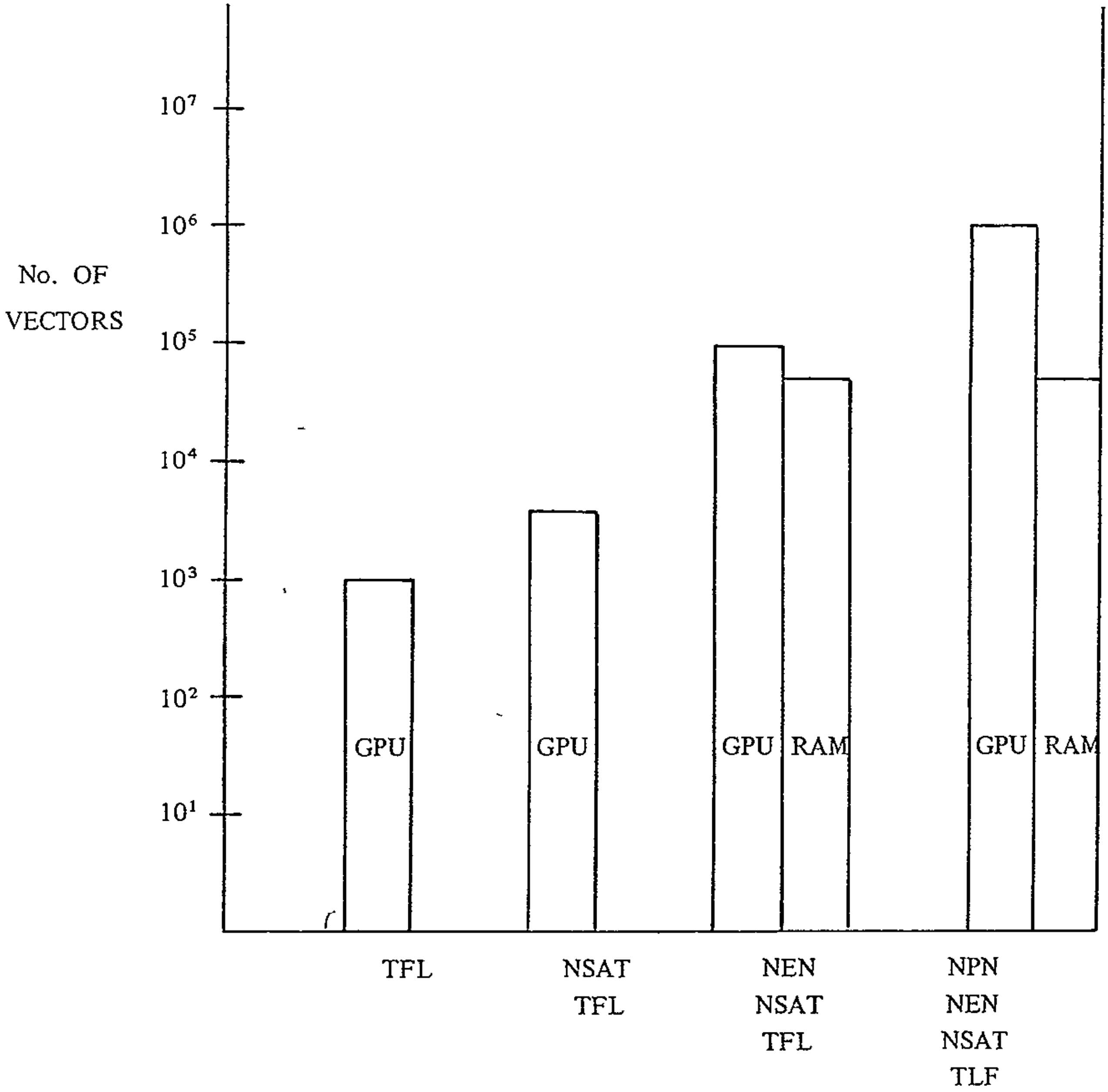


Figure 4.1 DEGREE OF TESTING

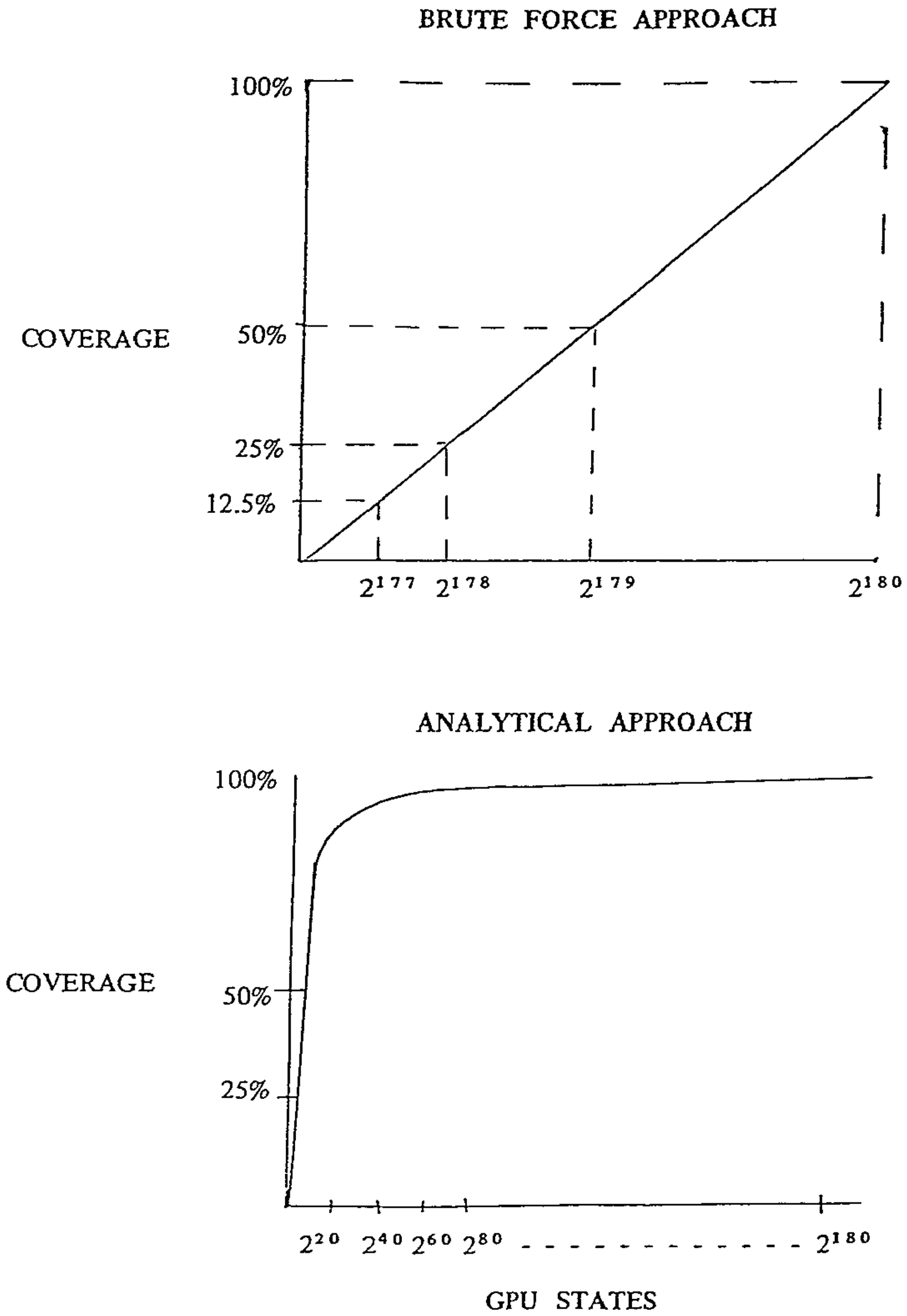


Figure 4.2 CONCEPTUALIZATION OF TEST VECTOR DESIGN APPROACH

Test Structure

To make the testing of the GPU manageable, tests are designed on a sub function basis. That is, the GPU has been decomposed into eleven separate sub functional entities (refer to Table 4.1) and separate test modules written, assembled and simulated for each sub function. These modules, called testware, are combined to form the complete test. They are generated utilizing the software system described in Section 3

Each testware module is designed to establish the functional integrity of the sub function, assure there are no node "stuck ats" within the sub function, determine the integrity of the electrical interfaces with other sub functions and test the sub function based on the nearest neighbor fault models described previously.

The structure of the complete test (i.e. the organization of the individual testware modules) is critical in approaching an acceptable level of fault coverage. Not only are GPU testware modules designed to verify individual sub functions on the device, but they are organized, as a whole, in a hierarchial structure in which previously tested sub functions are utilized to test other sub functions. The design of this hierarchial structure, such that all sub function interfaces are adequately tested is crucial in overall test success.

Fault isolation and diagnosis requirements also dictate a hierarchial test structure. With this type of structure, faults can be easily isolated within a sub function or its interface with another particular sub function. Figure 4.3 illustrates the concept of the GPU hierarchial test structure. The sub functions electrically closest to the interface pins are tested first, then more and more of the device is used to test itself with each new sub function test. For example, the RAM testware module utilizes P1B and P2B in performing the tests and the P1B, P2B Source tests utilize the RAM, P1B and P2B.

A detailed diagram of the test structure is presented in Figure 4.4. It illustrates the organization of the total test program. This organization also defines a module development schedule

Status

Currently, static testing is approximately 65% complete. Referring to Figure 4.4, tests are being developed at the 4th level of the test structure (Left and Right Data Type Selects). The Software System has proved to be extremely useful and error free. As the tests increased in complexity some problems were found in the simulator but were easily corrected. Dynamic testing will be initiated in the near future.

TABLE 4.1 TESTWARE MODULES

1. P1B
2. P2B
3. RAM
4. P1B SOURCE
5. P2B SOURCE
6. ALC → RAM
7. LEFT DATA TYPE
8. RIGHT DATA TYPE
9. ALC FUNCTION
10. SHIFT SELECT
11. OUTPUT DISABLE

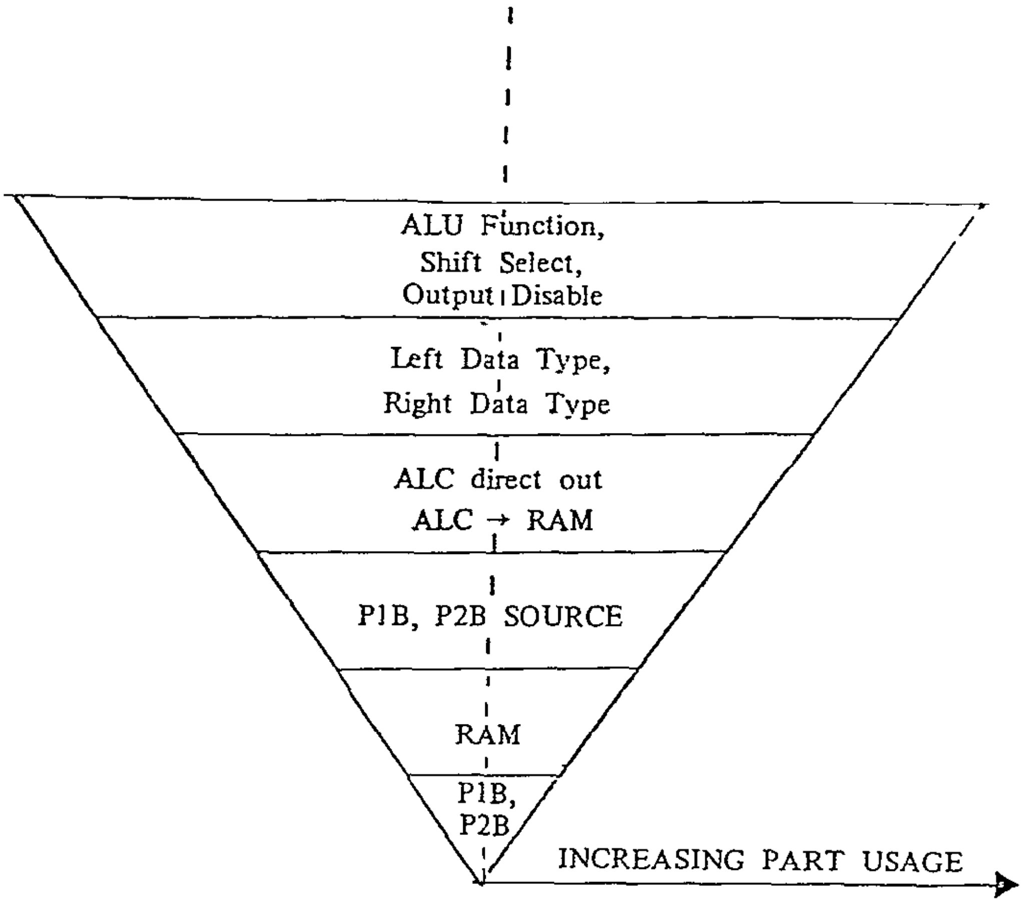


FIGURE 4.3 TESTWARE ORGANIZATION

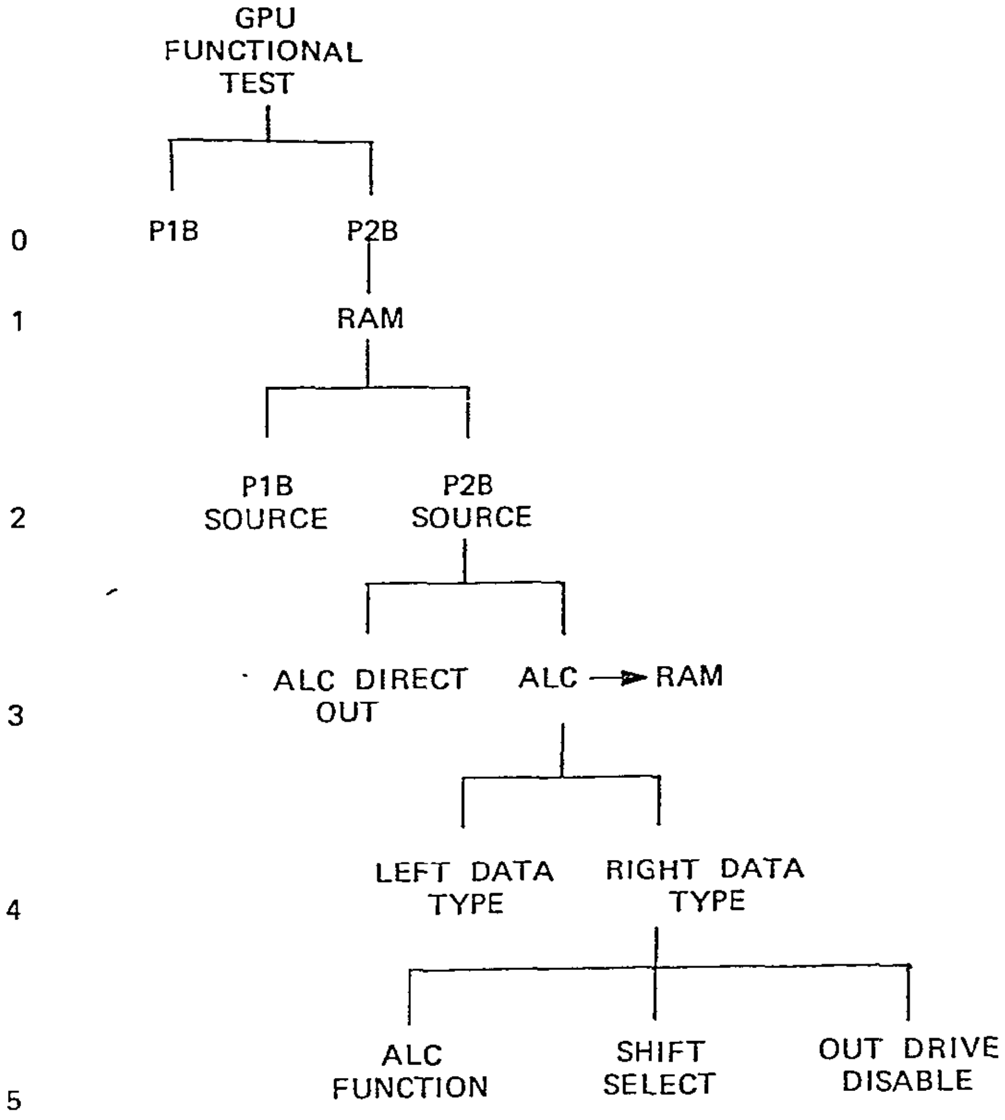


FIGURE 4.4 GPU TESTING STRATEGY

NEW TESTER MEMORY ARCHITECTURE ADVANCES STATE-OF-THE-ART IN MICROPROCESSOR TESTING

BY: Douglas H. Smith,
Applications Engineer
Tektronix, Inc.

Introduction

This article describes how the new memory architecture, now incorporated in Tektronix S-3260 family of LSI testers, has made significant contributions to the art of microprocessor testing.

Before describing what the new architecture is and how it can be used, let's examine how the microprocessor is used and then establish some criteria by which to judge some of the techniques used today. Figure 1 shows where microprocessors stand in the general hierarchy of the microcomputer.

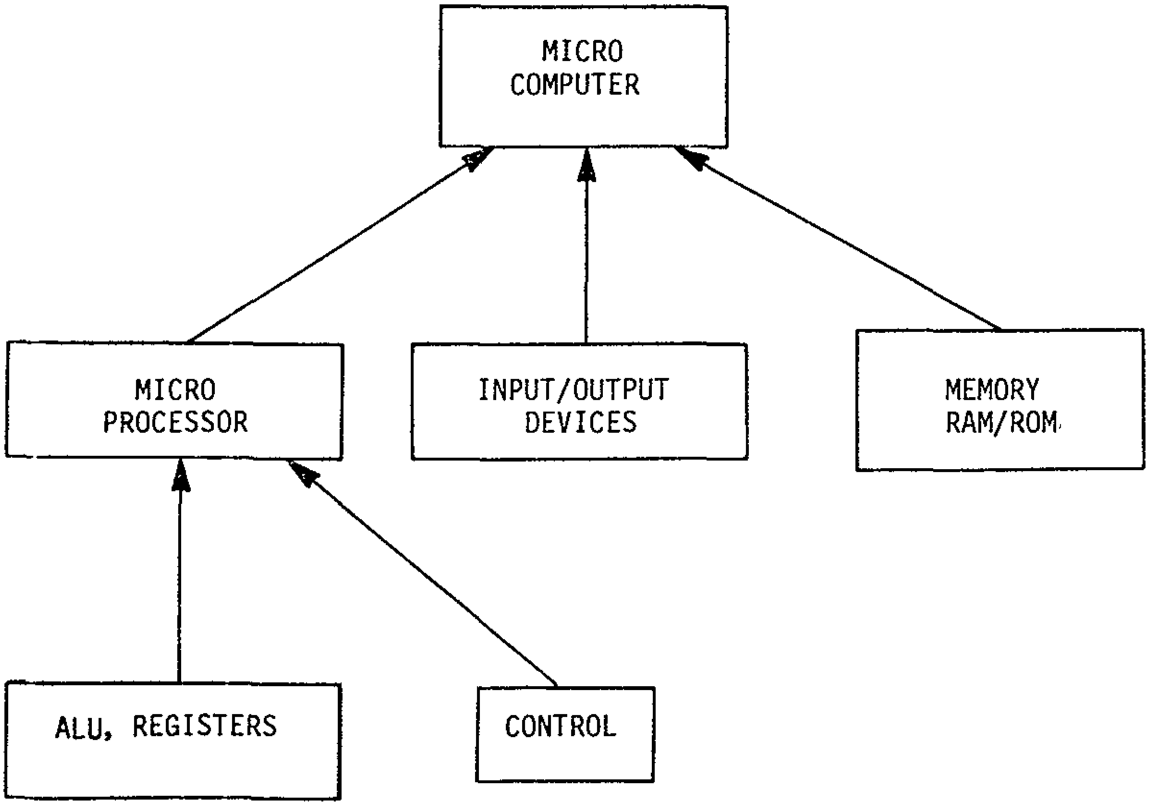


FIG. 1, DEFINITION OF HIERARCHY

In its real-world environment, the microprocessor has its own READ/WRITE memory and peripherals. It is at the microcomputer level that we are able to run system diagnostics because the system contains a CPU, some R/W memory, I/O devices and a source of interrupts. In the LSI tester world, we have only the CPU and a stimulus/response system (drivers, receivers, and test station high speed memory) with which to test the device. The simplest stored-response test is apparently, to duplicate on a tester, the system diagnostic used to check out the finished microcomputer.

Let's examine that approach and see why it is probably the most inefficient method. The check out diagnostic routines are used to check the whole system, including R/W memory, ROM, and I/O handling, and usually require a long time to execute. The task of the tester, however, is just to test the MPU, and the system dignostic is a gross overkill. Furthermore, in the system diagnostic, the CPU typically uses R/W scratchpad areas in a RAM to control the diagnostic algorithm. In the tester environment, these are represented as data in local memory which typically cannot be modified at the test rate. Therefore, data has to be continually reloaded from tester bulk memory (usually disc), an extremely time consuming process.

Since the diagnostic is impractical to run, we must approach the problem from another direction and define a test philosophy aimed directly at the microprocessor's structure. A thorough examination of the test requirement provides us with our criteria by which to select a tester.

Microprocessor testing consists of three basic elements:

1) Verify the existence and uniqueness of each of the internal elements.

2) Verify the interactions that can occur between the elements in the course of running a real program.

3) Check the ability of the MPU to handle random interrupts.

It is by these three criteria that the effectiveness of a given tester architecture should be judged.

S-3260 Advanced Memory Architecture

Behind each driver and receiver in the S-3260 are two memories, one a 1024 deep ECL RAM, the other a 1032 bit shift register (the significance of using two memories in satisfying the three basic test requirements for microprocessors will become more obvious as we proceed to actual test examples).

Both memories can be controlled at up to 20MHz by an advanced pattern controller developed by Tektronix. The controller can be used to control clock rate tester functions such as:

1) Branch on DUT condition (Pass/Fail branch)

2) Pattern Loops

3) Pattern subroutines

4) Change pattern source (RAM/Shift Register)

5) Store DUT output data

The Intel 8Ø8Ø microprocessor is used exclusively in this article as an example. For those readers unfamiliar with its mnemonics, here is a glossary.

FIG. 2, GLOSSARY

8Ø8Ø INSTRUCTIONS USED

XRA r	Exclusive or register with an accumulator
MOV r_1, r_2	$(r_1) \leftarrow (r_2)$
RAL	$(A_{N+1}) \leftarrow (A_N)$; $(CY) \leftarrow (A_7)$ $(A_0) \leftarrow (CY)$
JMP	$(PC) \leftarrow$ (BYTE3,BYTE2)
RAR	$(A_N) \leftarrow (A_{N+1})$; $(CY) \leftarrow (A_0)$ $(A_7) \leftarrow (CY)$
STA	((BYTE3)(BYTE2)) $\leftarrow$ (A)
LXISP	(SP) $\leftarrow$ ((BYTE3)(BYTE2))
PUSH (rp)	(SP-1) $\leftarrow$ (rh);(SP-2) $\leftarrow$ (r1) (SP) $\leftarrow$ (SP - 2)
STC	(CY) $\leftarrow$ (1)

General Device Description

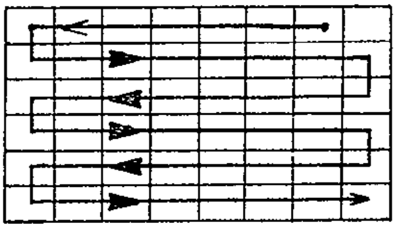
The 8Ø8Ø is a complete 8-bit parallel CPU for use in general purpose digital computer systems and is packaged in a 40-pin dual inline package. It contains six 8-bit data registers, an 8-bit accumulator, four 8-bit temporary registers, four flag bits, and an 8-bit parallel binary arithmetic unit.

Separate 16-line address and 8-line bidirectional data busses are used to allow direct interface to memory and other I/O ports.

Putting the 8Ø8Ø Through its Paces

Let's consider a test for verifying both the existence and uniqueness of the 8Ø8Ø registers. We can consider registers A through L as an 8 x 7 memory, and test for both existence and uniqueness of each register cell by walking a ONE through an all ZERO field, and checking the contents of every cell each time. The scheme is illustrated in Figure 3A.

FIG. 3A, WALKING 1 REGISTER TEST



The basic instructions used to accomplish the functions shown in Figure 3A are:

RAL - Rotate accumulator left

RAR - Rotate accumulator right

MOV r_1, r_2 - Transfer r_2 to r_1

Additional instructions used are:

STA - Store accumulator direct

PUSH B - Store B & C on Stack

PUSH D - Store D & E on Stack

PUSH H - Store H & L on Stack

The last four instructions cause the microprocessor to WRITE registers A through L on the stack, thus providing data for the tester to compare against. Several other instructions are necessary for the part to "stay in sync" with the pattern processor in the tester. The basic 8Ø8Ø test algorithm is described in detail in Figure 3B. Steps 1 through 8 initialize all registers to zero. Step 10 initialized the algorithm. There then follow six loops (Loop 1 through Loop 6), each of which is iterated 8 times. Loop 1 rotates the accumulator left 8 times, transfers it to the B register and then uses the subroutine CHECK to verify the contents of all registers via PUSH commands. Loop 2 rotates the accumulator right 8 times and transfers it to register C; also uses the CHECK subroutine to verify the contents of all the registers. This procedure is continued until the original carry bit from step 9 is in position (L, Ø) and the test is complete.

Please Note:

Every time the subroutine CHECK is called, the register contents which appear on the data bus in the PUSH commands are different, requiring the controller to use an alternate data source. This would not be possible if the tester only had one memory.

FIG. 3B, EXISTENCE AND UNIQUENESS OF 8Ø8Ø REGISTERS

Step Number	8080 Code	$(PC)_8$ Value	Tester Locations	Tester Controller
1	(Power on Reset)	N/A	0-5	MASK OUTPUT
2	XRA A	0	6-9	
3	MOV B,A	1	10-14	
4	MOV C,A	2	15-19	
5	MOV D,A	3	20-24	
6	MOV E,A	4	25-29	
7	MOV H,A	5	30-34	
8	MOV H,A	6	35-39	
9	STC	7	40-43	
10	LOOP 1: ┌▶ RAL	10	44-47	
11	8 MOV B,A	11	48-52	CALL CHECK SUBROUTINE
12	└▶ JMP ◁10▷	12	53-62	LOOP TO LOOP 1 8X
13	RAL	10	63-66	
14	JMP	11	67-72	
15	LOOP 2: ┌▶ RAR	10	73-76	
16	8 MOV C,A	11	77-81	CALL CHECK SUBROUTINE
17	└▶ JMP ◁10▷	12	82-91	LOOP TO LOOP 2 8X
18	RAR	10	92-95	
19	JMP 10	11	96-105	

Step Number	8080 Code	$(PC)_8$ Value	Tester Locations	Tester Controller
20	LOOP 3: →RAL	10	106-109	
21	8 MOV D,A	11	110-114	CALL CHECK SUBROUTINE
22	→JMP ⟨10⟩	12	115-124	LOOP TO LOOP 3 8X
23	RAL	10	125-128	
24	JMP 10	11	129-138	
25	LOOP 4: →RAR	10	139-142	
26	8 MOV E,A	11	143-147	CALL CHECK SUBROUTINE
27	→JMP ⟨10⟩	12	148-157	LOOP TO LOOP 4 8X
28	RAR	10	158-161	
29	JMP 10	11	162-171	
30	LOOP 5: →RAL	10	172-175	
31	8 MOV H,A	11	176-180	CALL CHECK SUBROUTINE
32	→MJP ⟨10⟩	12	181-190	LOOP TO LOOP 5 8X
33	RAL	10	191-194	
34	JMP 10	11	195-204	
35	LOOP 6: →RAR	10	205-208	
36	8 MOV L,A	11	209-213	CALL CHECK SUBROUTINE
37	→JMP ⟨10⟩	12	214-223	LOOP TO LOOP 6 8X, HALT
38	CHECK: LXISP⟨1234⟩	12	224-233	
39	STA ⟨4321⟩	15	234-236	
40	PUSH B	20	247-257	USE AND INCREMENT
41	PUSH D	21	258-268	ALTERNATE DATA
42	PUSH H	22	269-279	(SEE FIGS 3A & 3B)
43	MVI B⟨0⟩	23	280-286	
44	MOV C,B	25	287-291	
45	MOV D,B	26	292-296	
46	MOV E,B	27	297-301	
47	MOV H,B	30	302-306	
48	MOV L,B	31	307-311	
49	JMP 12	32	312-321	RETURN FROM SUBROUTINE

In Shift register memory are 48 successive patterns such as the one in Figure 3C. They are successively incremented after being used.

00010000	A	CHECK pattern used
00000000	B	During 5th iteration of
00000000	C	3rd loop. The test has
00010000	D	48 of these patterns
00000000	E	prestored in the Shift
00000000	H	register.
00000000	L	

FIGURE 3C

The above example shows how the use of two separate memories greatly enhances testing for existence and uniqueness of functional elements. Similar algorithms can also be developed for other elements of the MPU to check for interactions between functional elements.

The Spinning Interrupt

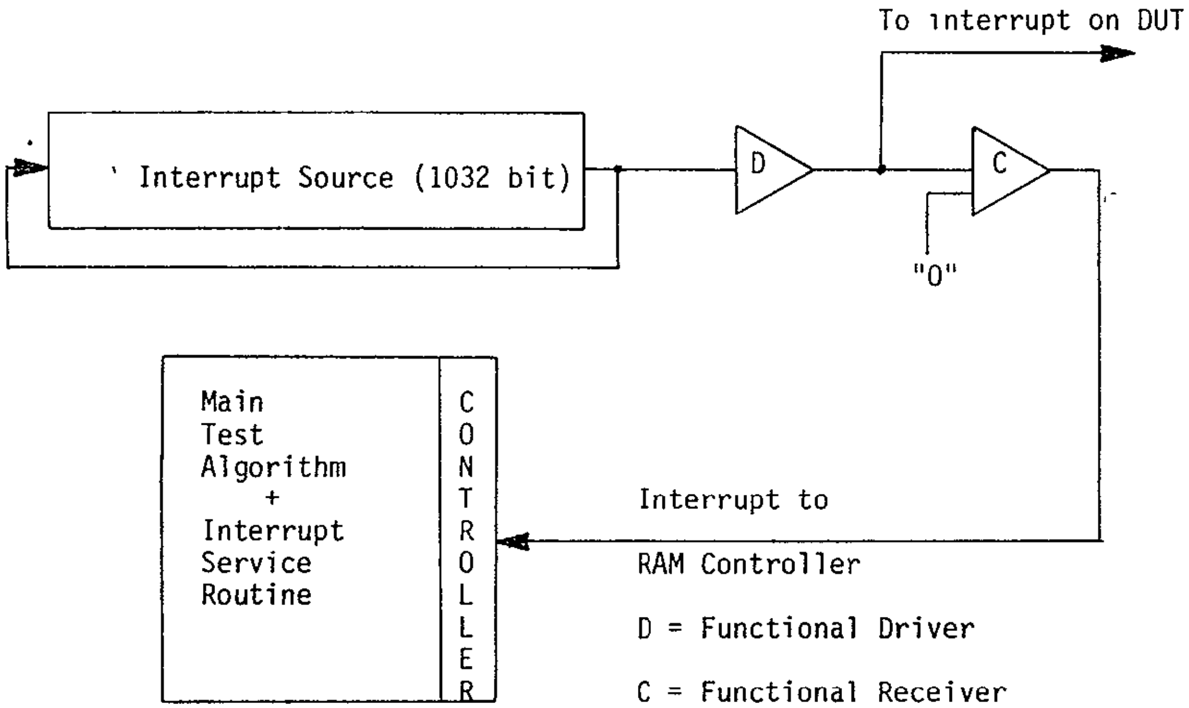
In the real computer, interrupts can arrive from any part of the system at any point in a program. That is to say, they are essentially asynchronous in nature. It is therefore essential that any valid test of the interrupt structure take this into account. It is not sufficient to just include a few interrupts in the main test algorithm, since this would not test the processor's ability to handle interrupts at any point in any instruction sequence.

An effective interrupt testing scheme is shown in Figure 4. In this case, the interrupt source is the 1032 bit shift register operating in a recirculating mode. Any one in the bit stream causes two things to occur:

A) The microprocessor receives an interrupt.

B) A signal is generated which causes the main test algorithm to branch to one of several interrupt service routines.

FIG. 4, INTERRUPT DRIVEN TEST MODE



Since the interrupt source is modulo 1032, and the main algorithm is some other modulo, the interrupts arrive at random time points in the test program - a situation paralleling the real world.

This particular test points out something basic -- that the tester should parallel (i.e., have similar capabilities) to the DUT. In this test, the interrupt source provides an interrupt in a random manner to both the DUT and the RAM controller. The interrupt service routines in the RAM are written in such a way as to ensure the device functions correctly in response to the interrupting signal. This level of testing can only be accomplished on a Tektronix S-3260.

What Went Wrong?

So far, we have only talked about using the tester memories to provide stimulus for the DUT and expected output data. There is one other very important area of device testing failure diagnosis. In previous test techniques, discrepancies between DUT output data and expected data have been used to stop the tester and indicate failure. This gives us minimal information on what the DUT was doing when the error occurred. By branching on error in the RAM controller, it is possible to compare DUT output with ZERO while storing the resultant error pattern in the 1032 bit register. If we execute instructions such as PUSH R during the error diagnostic, we can obtain a "snapshot" of the internal DUT conditions when failure occurred at functional test rates -- information hitherto unavailable on any LSI tester!

Conclusion

When judging a microprocessor tester, the criteria boil down into three important areas.

1) Is the tester memory architecture sufficiently advanced to "emulate" the device under test?

2) How well can asynchronous interrupts be handled?

3) What failure diagnosis can be performed?

The S-3260, with its new memory architecture is unparalleled in the world when it comes to providing solutions to basic microprocessor testing problems.

SECTION IV
POTPOURRI

Chairman

W. Richard Scott
Jet Propulsion Laboratory

77-39

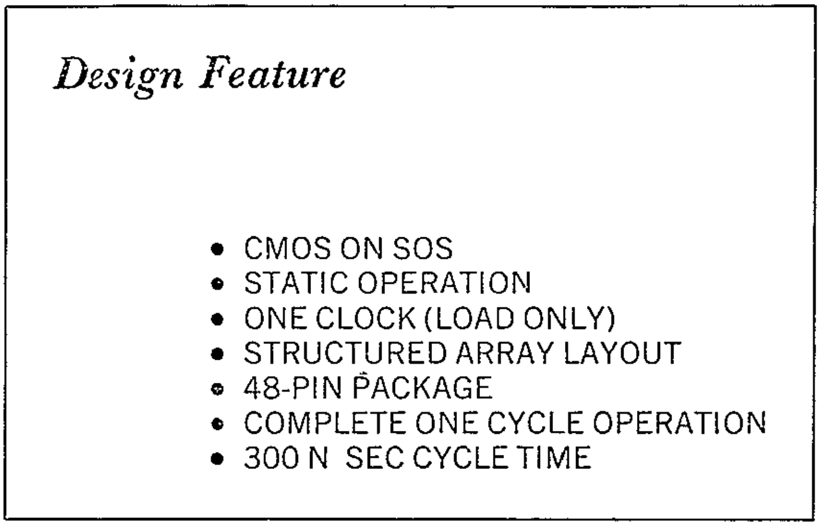
GENERAL PROCESSING UNIT

Robert Fosdick
Tracor, Inc.
Dallas, Texas

(No Text Available)

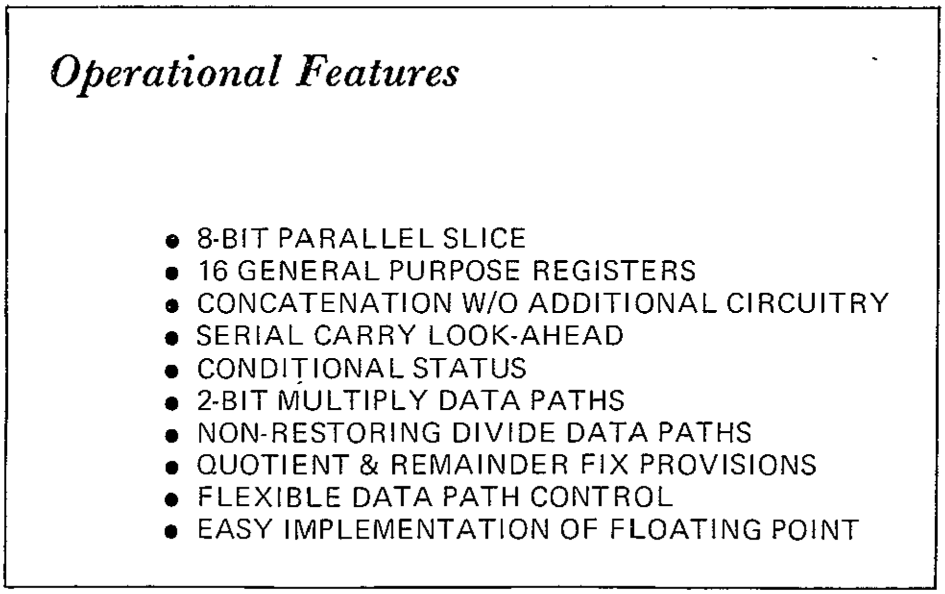
Program Intent

PROVIDE A BENCHMARK OF THE COMPLEXITY ATTAINABLE WITH COMPLEMENTARY MOS TECHNOLOGY AND THE SILICON-ON-SAPPHIRE (SOS) PROCESS. THE END CIRCUIT IS TO BE COMPATIBLE FOR USE AS A BUILDING BLOCK IN THE IMPLEMENTATION OF A HIGH PERFORMANCE COMPUTER



Design Feature

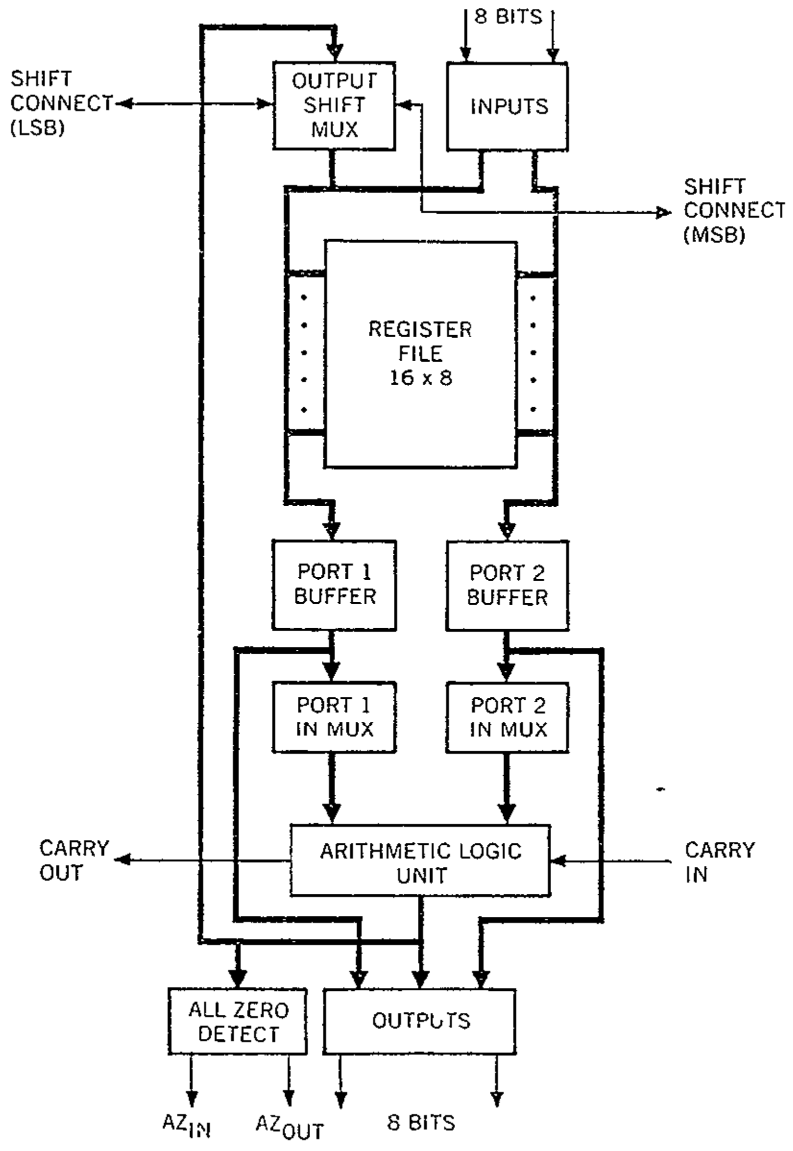
- CMOS ON SOS
- STATIC OPERATION
- ONE CLOCK (LOAD ONLY)
- STRUCTURED ARRAY LAYOUT
- 48-PIN PACKAGE
- COMPLETE ONE CYCLE OPERATION
- 300 N SEC CYCLE TIME

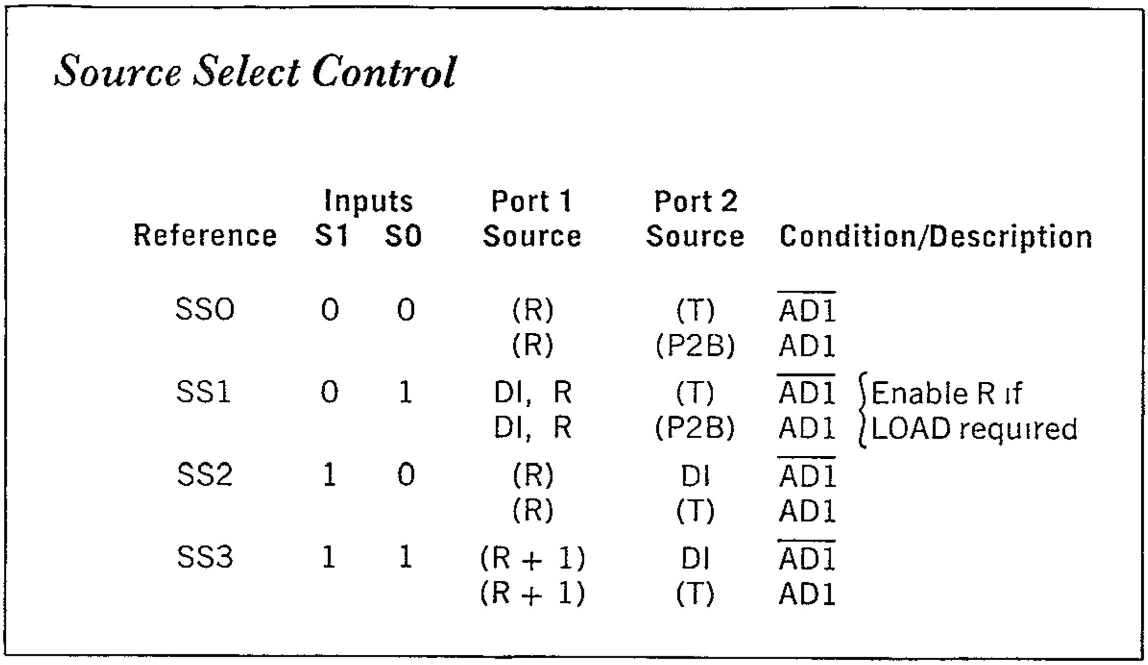


Operational Features

- 8-BIT PARALLEL SLICE
- 16 GENERAL PURPOSE REGISTERS
- CONCATENATION W/O ADDITIONAL CIRCUITRY
- SERIAL CARRY LOOK-AHEAD
- CONDITIONAL STATUS
- 2-BIT MULTIPLY DATA PATHS
- NON-RESTORING DIVIDE DATA PATHS
- QUOTIENT & REMAINDER FIX PROVISIONS
- FLEXIBLE DATA PATH CONTROL
- EASY IMPLEMENTATION OF FLOATING POINT

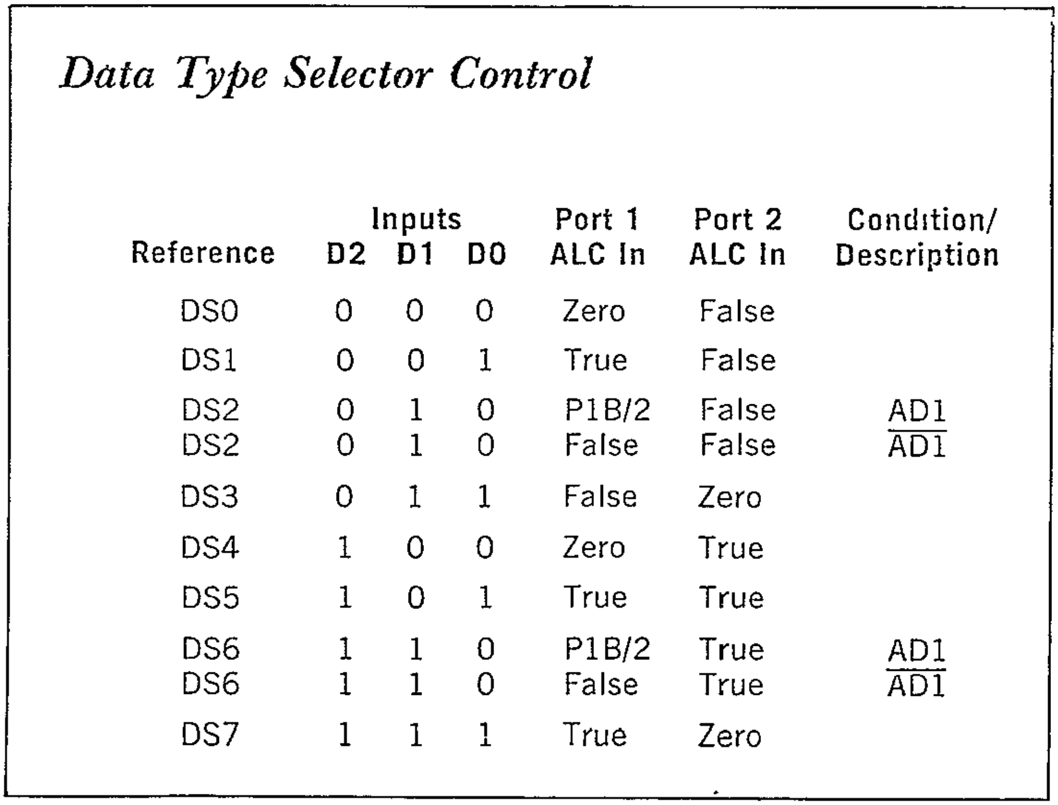
GPU Organization





Source Select Control

Reference	Inputs S1	S0	Port 1 Source	Port 2 Source	Condition/Description	
SS0	0	0	(R)	(T)	$\overline{AD1}$	
			(R)	(P2B)	AD1	
SS1	0	1	DI, R	(T)	$\overline{AD1}$	Enable R if LOAD required
			DI, R	(P2B)	AD1	
SS2	1	0	(R)	DI	$\overline{AD1}$	
			(R)	(T)	AD1	
SS3	1	1	(R + 1)	DI	$\overline{AD1}$	
			(R + 1)	(T)	AD1	



Data Type Selector Control

Reference	Inputs D2	D1	D0	Port 1 ALC In	Port 2 ALC In	Condition/ Description
DS0	0	0	0	Zero	False	
DS1	0	0	1	True	False	
DS2	0	1	0	P1B/2	False	AD1
DS2	0	1	0	False	False	$\overline{AD1}$
DS3	0	1	1	False	Zero	
DS4	1	0	0	Zero	True	
DS5	1	0	1	True	True	
DS6	1	1	0	P1B/2	True	AD1
DS6	1	1	0	False	True	$\overline{AD1}$
DS7	1	1	1	True	Zero	

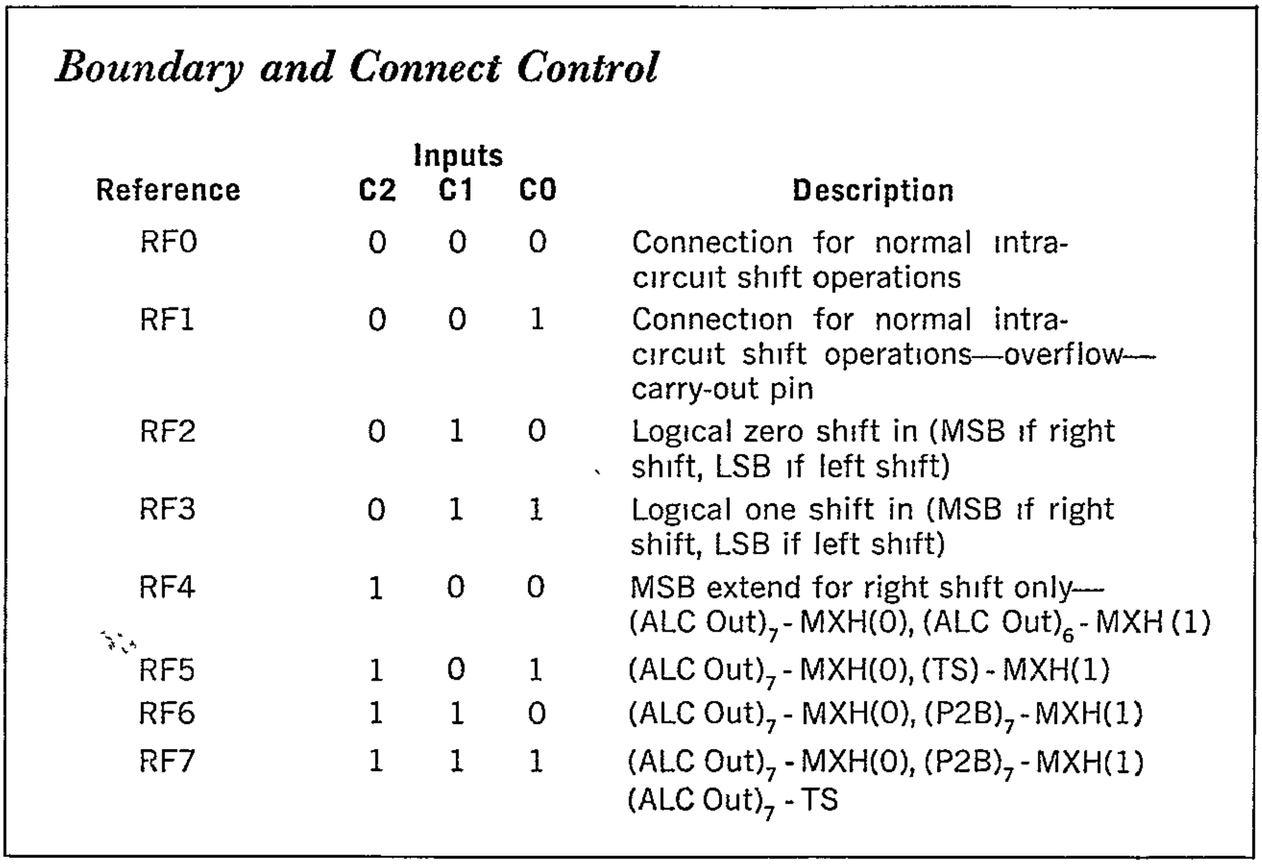
ALC Control

	Inputs			
Reference	A1	A0	Function	Comments
AD0	0	0	ADD	1st Port 2 Source, External carry-in
AD1	0	1	ADD	2nd Port 2 Source, External carry-in
AD2	1	0	AND	1st Port 2 Source, Logical "1" carry-in*
AD3	1	1	OR	1st Port 2 Source, Logical "0" carry-in*

*Group level propagate of external carry-in

Destination Select Control

	Inputs			
Reference	M2	M1	M0	Description
A00	0	0	0	Direct Store Input to Register File
A01	0	0	1	ALC result left shift one into Port 1
A02	0	1	0	ALC result right shift one into Port 1
A03	0	1	1	ALC result right shift two into Port 1
A04	1	0	0	ALC result No shift into Port 1
A05	1	0	1	ALC result No shift into Port 1 and to circuit output
A06	1	1	0	P2B to circuit output, ALC result No shift into Port 1 if Load Clock
A07	1	1	1	P1B to circuit output, ALC result No shift into Port 1 if Load Clock



Boundary and Connect Control

		Inputs		
Reference	C2	C1	C0	Description
RF0	0	0	0	Connection for normal intra-circuit shift operations
RF1	0	0	1	Connection for normal intra-circuit shift operations—overflow—carry-out pin
RF2	0	1	0	Logical zero shift in (MSB if right shift, LSB if left shift)
RF3	0	1	1	Logical one shift in (MSB if right shift, LSB if left shift)
RF4	1	0	0	MSB extend for right shift only— (ALC Out)$_7$ - MXH(0), (ALC Out)$_6$ - MXH (1)
RF5	1	0	1	(ALC Out)$_7$ - MXH(0), (TS) - MXH(1)
RF6	1	1	0	(ALC Out)$_7$ - MXH(0), (P2B)$_7$ - MXH(1)
RF7	1	1	1	(ALC Out)$_7$ - MXH(0), (P2B)$_7$ - MXH(1) (ALC Out)$_7$ - TS

RADIATION EFFECTS ON MICROPROCESSORS

Paul Measel
The Boeing Company

A. SUMMARY

The applicability of microprocessors to military systems with a radiation survival requirement was investigated on a Boeing Company IR&D program. A hard microcontroller module was designed and built. Radiation tests were performed on a variety of LSI part types and technologies to select appropriate types and validate the module hardness. The radiation response data is summarized below. Initial radiation response data on the RCA CMOS/SOS processor are also included.

B. PRESENTATION

Microprocessor based circuits provide the well-known potential for increased flexibility, and reduced cost, power, weight and volume over hardwired logic. Consequently, evaluating the applicability of microprocessors and associated LSI devices to certain systems requiring radiation survival was undertaken under a Boeing Company IR&D program. Some of the results have been reported in Reference 1 and are summarized below.

On this IR&D program, a hard microcontroller module was designed and built to perform a monitor and control function of missile operational ground equipment. The preparation of the microcontroller module required hardware and software design, selection of LSI and other piece part types, development of piece part and module electrical and radiation test techniques, and the performance of radiation tests on the LSI piece parts and the completed module.

Figure 1 illustrates a typical missile ground electronics system including the monitor and control function. Such equipment has used bipolar discrete and SSI devices. Figure 2 shows the module design concept, which is a microprocessor based controller, consisting of a 16-bit central processor unit, a read only memory (4096 word), a random access (scratch pad) memory (256 word), a clock-timer, interrupt (including circumvention) logic, and input and output registers. Bipolar technology was selected for use in the module because this technology was expected to provide adequate hardness in commercially available parts types. The processor device used in the module was the MMI 6701 T^{2}L Schottky bipolar 4-bit slice. A later redesign was performed for the AM 2901. The module was constructed on two 8" x 10" circuit cards. The parts count in the module was one-third the count of equivalent hardwired logic.

A simple self-test algorithm was used in the module for performance testing during irradiation. For the operational demonstration of the module a TI 960A minicomputer was used to provide the required complex inputs to the module and verify the module outputs. Dose rate tests were performed on the module and proper circumvention and recovery obtained up to and including the maximum test level of 10^8 rad(Si)/sec. The module is expected to be hard to radiation levels in excess of 7 x 10^5 rad(Si) and 6 x 10^{13} n/cm^2, by parts test data.

Electrical exerciser circuits were developed for in-situ electrical testing of microprocessors and memory devices during irradiation. Radiation test data was acquired on several LSI device types, including T^{2}L, I^{2}L and MOS technologies, during the development of the module. These data are summarized in Table 1. Sample size is given in parenthesis, i.e., (1). Where the "greater than" symbol (>) is used, the stated environment is the highest level reached, with the test terminated without observing functional failure. Appropriate parametric data was also obtained for these parts. Some of these part types are discussed in more detail below.

The MMI 6701D Shottky bipolar T^{2}L microprocessor was functionally exercised by storing a word (0101) on the data input lines in the Q register then adding the contents of the accumulator to the same word on the data input and outputting the result (1010) on the data output lines. The output states were alternating "1" and "0" levels.

During dose rate tests all four data outputs, clock pulses, surge current, supply current I_{CC}, and photodiode current were monitored. Transient upset was observed for a Linac pulse width of 70 nsec at 2 x 10^8 rad(Si)/sec for one device and 4 x 10^8 rad(Si)/sec for two other devices. No latch-up was observed up to 5.6 x 10^8 rad(Si)/sec. For a 4 µsec Linac pulse width, transient upset was observed at 5.1, 23, 23 and 41 x 10^6 rad(Si)/sec for four devices. One device showed latch-up above 3.9 x 10^8 rad(Si)/sec as indicated by an I_{CC} shift from ~200 mA up to ~320 mA upon Linac firing. I_{CC} was restored by interrupting power. This effect was repeatable.

The total dose irradiation was performed by applying Linac pulses at dose rates of 5.5 x 10^7 rad(Si)/sec which is slightly above transient failure threshold. The only electrical parameter monitored on the MMI 6701D which exhibited measurable degradation (~5%) after an exposure of 7 x 10^5 rads(Si) was the sink current (I_{OL}). Other electrical parameters monitored, which did not change with total dose, included instruction to data-out delay, DC power supply current, output source current (I_{OH}), and output voltage levels. The device remained functional at this dose level.

The same parameters were measured for neutron exposure as for total dose. After 1.5 x 10^{14} n/cm^2 fluence the microprocessor remained functional and the changes in the various parameters were small. The relative changes were I_{CC}: -6%, I_{OL}: -22%, I_{OH}: -20%. Delay times at the various outputs: 1st: +5%, 2nd: +22%, 3rd: +13%, 4th: +23%.

The AMD 2901 4-bit S/T^{2}L microprocessor was exercised with a gated train of alternate pass and add instructions.

For the dose rate tests all four outputs, clock pulse, surge current, supply current and photo diode current were monitored. Transient upset was observed for a Linac pulse width of ~30 nsec at 1.0 x 10^8, 1.4 x 10^8 and

1.1 x 10^8 rad(Si)/sec for three devices. For a 4 μsec Linac pulse width, transient upset was observed at 1 1 x 10^7, 1.2 x 10^7 and 8.8 x 10^6 rad(Si)/sec. No latchup was observed up to the maximum rates delivered (5.2 x 10^9, 1 1 x 10^{10} and 8.8 x 10^9 rad(Si)/sec respectively). Following the latchup tests the device that had received 1.1 x 10^{10} rad(Si)/sec was inoperative for undetermined reasons.

The total dose irradiation was performed by applying wide Linac pulses at dose rates of $\sim 10^9$ rad(Si)/sec. After 2 x 10^7 rad(Si) total dose the output high voltages had increased ~10% and the supply currents had decreased 10 to 15%.

The Intel 8080B n-channel MOS microprocessor was functionally exercised using a simple add and compare program. Upon initiation by an external trigger signal, the program recycled until a program error occurred or until an external reset signal was applied. A HALT instruction was placed in all memory locations not utilized in the test routine so that a transient upset resulting in address alteration would halt the routine.

For an electron pulse width of 5 μsec, the dose rate upset threshold was found to be 1.8 x 10^5 rads(Si)/sec. For latchup tests the Linac pulse was narrowed to 200 nanoseconds to reduce the total dose accumulation per pulse at the higher dose rates to permit finding the latchup threshold prior to total dose failure of the circuit. No latchup was observed at any of the test levels up to 1.8 x 10^9 rads(Si)/sec where functional failure from total dose occurred. The total dose resulting in failure of the 8080 was determined to be 1,500 rads(Si).

Three additional commercial devices and three MIL spec devices were subjected to wide (4 μs) pulse and narrow (100 ns) pulse tests. The wide (4 μs) pulse upset thresholds were (1.2, 1.24 and 1.4) x 10^5 rad(Si)/sec for the commercial devices and (1.2, 1.6 and 1.75) x 10^5 rad(Si)/sec for the militarized devices. Narrow pulse thresholds were (4.6, 2.7 and 3.6) x 10^6 rad(Si)/sec for the commercial devices and (2.9, 3.6 and 5.2) x 10^6 rad(Si)/sec for the Mil spec devices. Neutron irradiation was not performed.

The Texas Instruments X0400, I^2L technology, prototype of the commercial SBP0400 4-bit parallel binary processor was functionally exercised using procedures similar to those described for the 6701D device.

The transient upset threshold for the X0400 occurred at a dose rate of 2.1 x 10^7 rads(Si)/sec (4 μsec pulse). Resumption of the proper switching sequence occurred within 2 μsec of the end of the radiation pulse. Recovery time at the highest dose rate, 3.7 x 10^8 rads(Si)/sec, was approximately 3 μsec. No measurable response on the outputs was observed at dose rates below 2.1 x 10^7 rads(Si)/sec. Because of the current sourcing for I^2L, no surge currents were seen. No latch-up was observed up to 3.7 x 10^8 rads(Si)/sec.

During total dose tests functional failure of the X0400 occurred at 1.4 x 10^6 rads. At this dose no degradation was noted in output levels, however, improper logic states were observed on the outputs. No significant annealing was observed after 60 minutes at room temperature following the total dose tests (after exposure to a total dose of 2.1 x 10^6 rads).

Following the dose exposures the test device was annealed for 48 hours at 200°C. After this anneal cycle the device had recovered functionally, however, the electrical parameters monitored (clock to data out delay and sink current capability) had only recovered to levels corresponding to 3.4 x 10^5 rads.

The dependence of clock to data out delay, minimum clock period, and sink current (I_{OL}) on injection current was measured as a function of neutron exposure. The most sensitive parameter to neutron fluence was the sink current (~30% degradation at 5 x 10^{11} n/cm^2). The device became non-functional for injector currents below 5.6 mA after 1 x 10^{13} n/cm^2. After 4 x 10^{13} n/cm^2 the microprocessor was no longer functional at any injector current between 3.2 mA and 150 mA.

Since the bipolar technology has relatively high power requirements, and the Schottky construction is potentially vulnerable to latchup, alternative technologies are being investigated. Preliminary radiation tests have been completed on the AFAL/AFML/ABRIES General Processor Unit (GPU) built by RCA. It is a CMOS/SOS 8-bit microprocessor slice, which has been discussed in detail in earlier papers in this Workshop. A simple test fixture (Figure 3) was prepared, which provides the waveforms of Figure 4 so that the GPU will perform a register to data-in addition. The waveform of Figure 4 was synchronized with the LINAC pulse, with the capability of placing the LINAC pulse at any desired position relative to the operational cycle of the GPU. Four GPU devices were irradiated for narrow and wide Linac pulses, and total dose. The LINAC pulse was placed just after the 1-0 transition in the pin 4-6 output (Figure 4). The average dose rate upset levels were 8.8 x 10^9 rad(Si)/sec for a 40 nsec pulse, and 2.4 x 10^9 rad(Si)/sec for a 1 μsec pulse.

The GPU devices tested were from a sample of the first lot produced, and employed an unhardened gate oxide. The average total dose level for 5% output degradation (pin 46) was 4.9 x 10^3 rad(Si) and for no operation was 1.3 x 10^4 rad(Si). It is expected that the hardness of this oxide can be improved. Also, a hard oxide is planned with a total dose hardness of 10^6 rad(Si). No neutron irradiations were performed.

In conclusion, selected commercial LSI devices have hardness adequate to meet moderate survivability requirements. The CMOS/SOS technology shows significant survivability potential in LSI, particularly if the total dose hardness goals are met.

Reference 1. Measel, P. R. and Sivo, L. L., "Development of a Hard Micro-controller", IEEE Transactions on Nuclear Science, Vol. NS-23, No. 6, December 1976.

Table 1. Summary of Piece Part Functional Failure Levels

Devices	Technology	Dose rate [rad (Si)/sec]	Total dose [rad (Si)]	Neutrons [n/cm^2]
MMI 6701 μP	S/T^{2}L	Wide pulse (4) (0 51 — 4 1) x 10^7 narrow pulse (3). (2 — 4) x 10^8	(1) > 7 x 10^5	(1) >1.5 x 10^{14}
AM 2901 μP	S/T^{2}L	Wide pulse (3) (0 88 — 1.2) x 10^7 narrow pulse (3) (1 0 — 1.4) x 10^8	(3) > 2 x 10^7	Not taken
Intel 8080 μP commercial	NMOS	Wide pulse (4) (1 2 — 1 8) x 10^5 narrow pulse (3) (2 7 — 4 6) x 10^6	(1) 1.5 x 10^3	Not taken
Intel 8080 μP Mil spec	NMOS	Wide pulse (3) (1.2 — 1 75) x 10^5 narrow pulse (2 9 — 5 2) x 10^6	Not taken	Not taken
Intel 3002 μP		Narrow pulse (6) (1 25 — 4 8) x 10^8	Not taken	Not taken
TI X0400 μP	I^{2}L	Wide pulse (1). 2 1 x 10^7	(1) 1.4 x 10^6	(1) 1 x 10^{13}
IM 5533A RAM	Gold T^{2}L bipolar	Medium pulse (5). 1.7 x 10^8 Wide pulse (3) (1 7 — 2 5) x 10^8	(2) >3 x 10^7	(5) 1 3 x 10^{14}
IM 6508 RAM	CMOS	Not taken	Active, C0 60(3). (2.7 — 3 3) x 10^3 passive C0 60(1) 79 x 10^3	Not taken
Harris 6508-8 RAM	CMOS	Not taken	Active, C0 60(4) (8.6 — 10.2) x 10^3 (1) passive, C0 60(1) 61 x 10^3	Not taken
MWS 5001D RAM	CMOS/SOS	Wide pulse (2). (0 9 — 1 4) x 10^9 narrow pulse (3) >3 x 10^9	Active, C0 60(3). 1 x 10^4 passive, C0 60(1). 2 5 x 10^4	Not taken
MMI 6340D 4K PROM	S/T^{2}L	Wide pulse (1) 9.2 x 10^6	(1) >1 x 10^6	(1) 1 5 x 10^{14}

Table 1. Summary of Piece Part Functional Failure Levels (cont'd)

Devices	Technology	Dose rate [rad (Si)/sec]	Total dose [rad (Si)]	Neutrons [n/cm^2]
Intel 3624 4K PROM	S/T^2L	Wide pulse (1). 2.3×10^7	(1) $> 1 \times 10^7$	Not taken
Q-Tech QT6T8 Oscillator (4 MHz)	Hybrid bipolar	Wide pulse (1). $(3-4) \times 10^6$	(1) $> 1 \times 10^6$	(1) between $(5-1) \times 10^{14}$
McCoy MC702A1 Oscillator (4-MHz)	Hybrid bipolar	Wide pulse (1): $8\ 5 \times 10^7$	(1) $> 1.1 \times 10^7$	(1) $> 6 \times 10^{13}$
Boeing 32 Cell inverter	I^2L	Not taken	Not taken	1×10^{13}
XC 5850 8 x 8 Bit multiplier	EFL Bipolar	Wide pulse (1). 1.2×10^7 narrow pulse (1). $3\ 9 \times 10^8$	(1) $> 2 \times 10^6$	Not taken

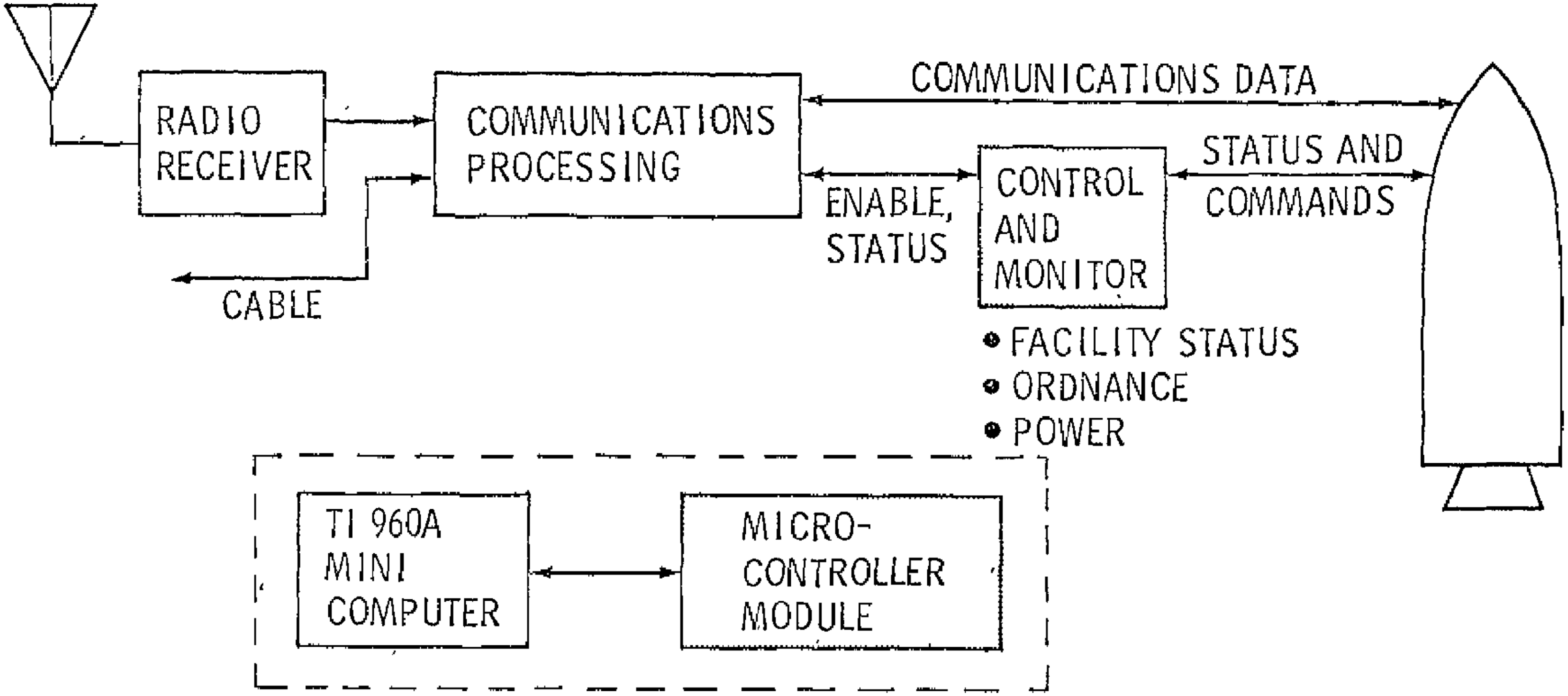


Figure 1 Missile Ground Electronics

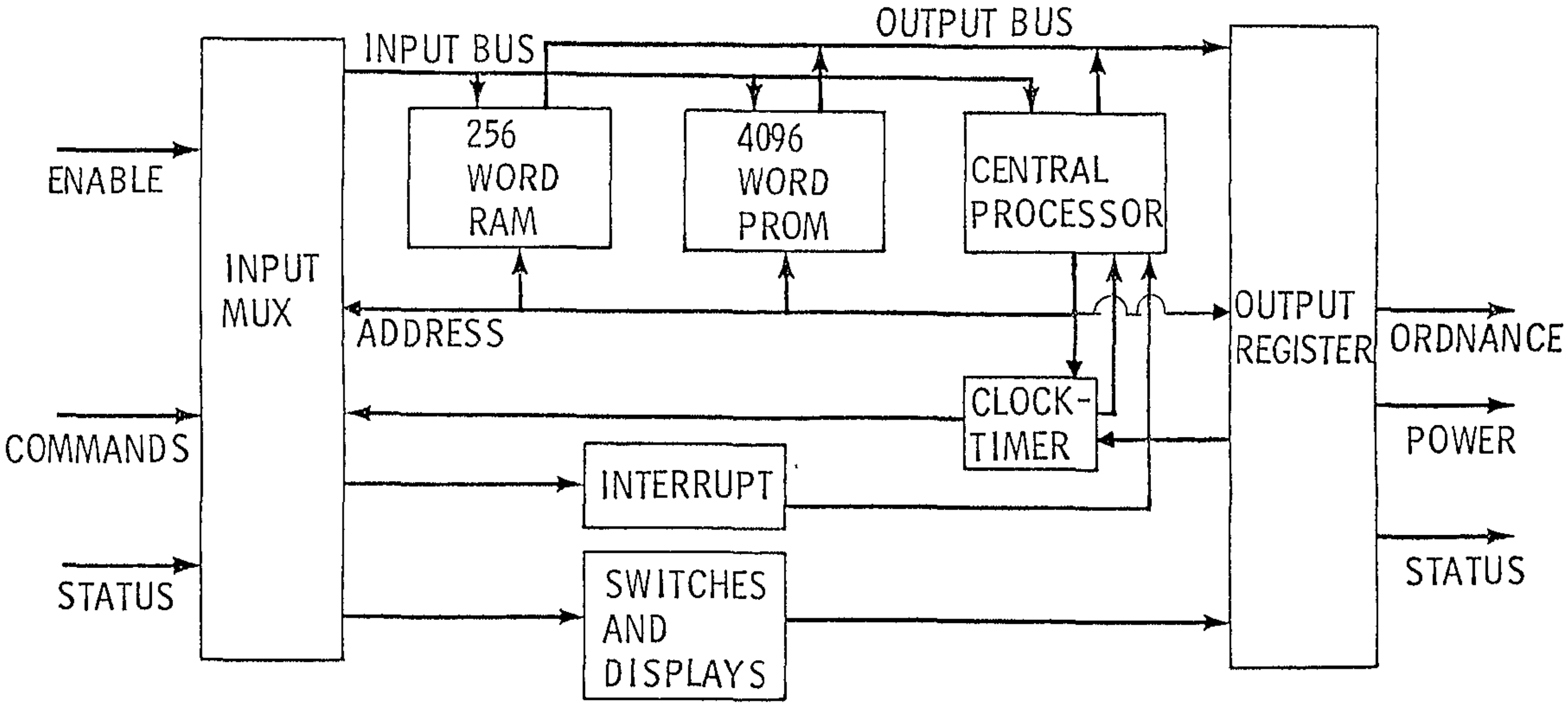


Figure 2 Module Design Concept

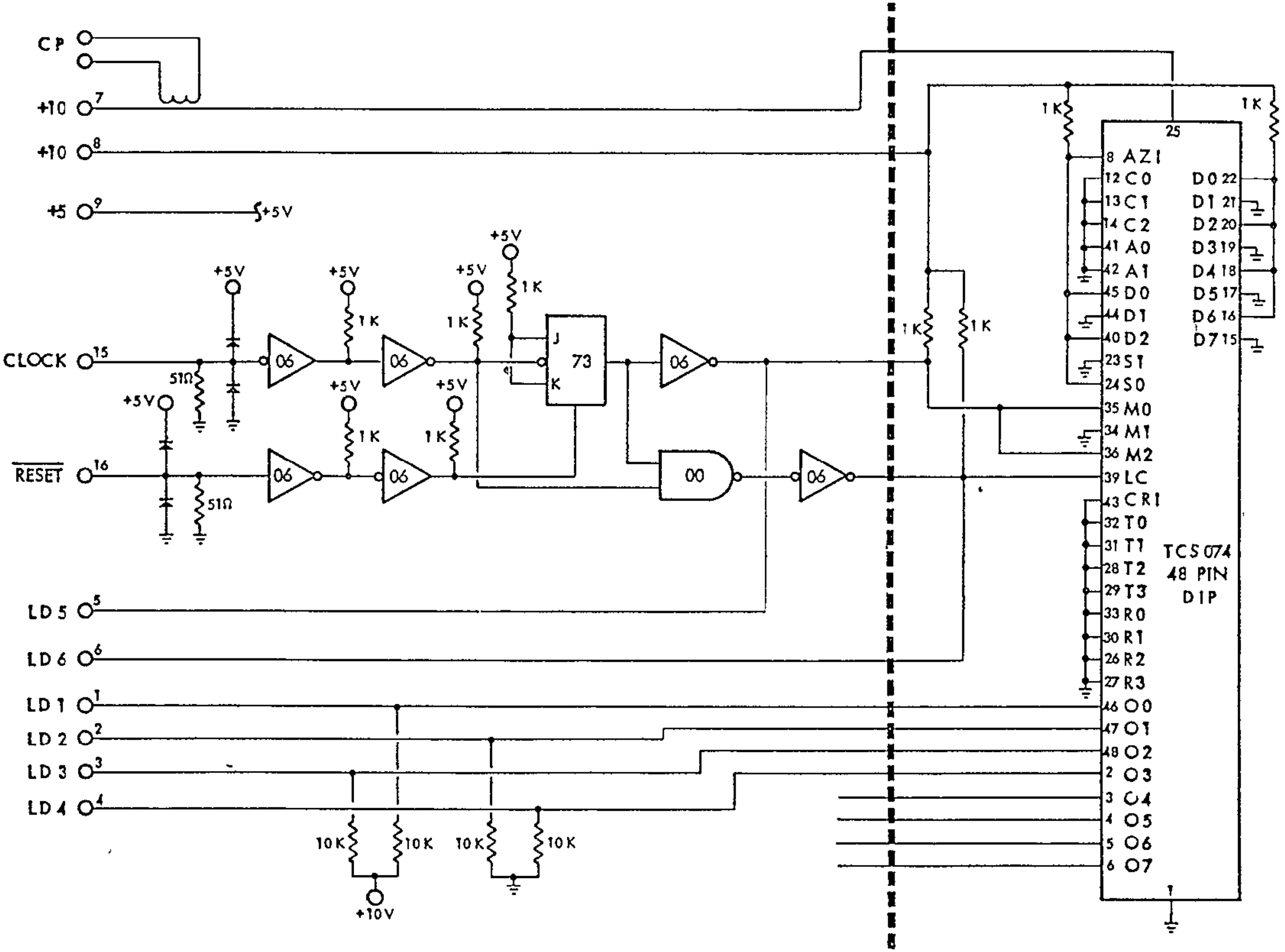


Figure 3. TCS 074 Test Fixture

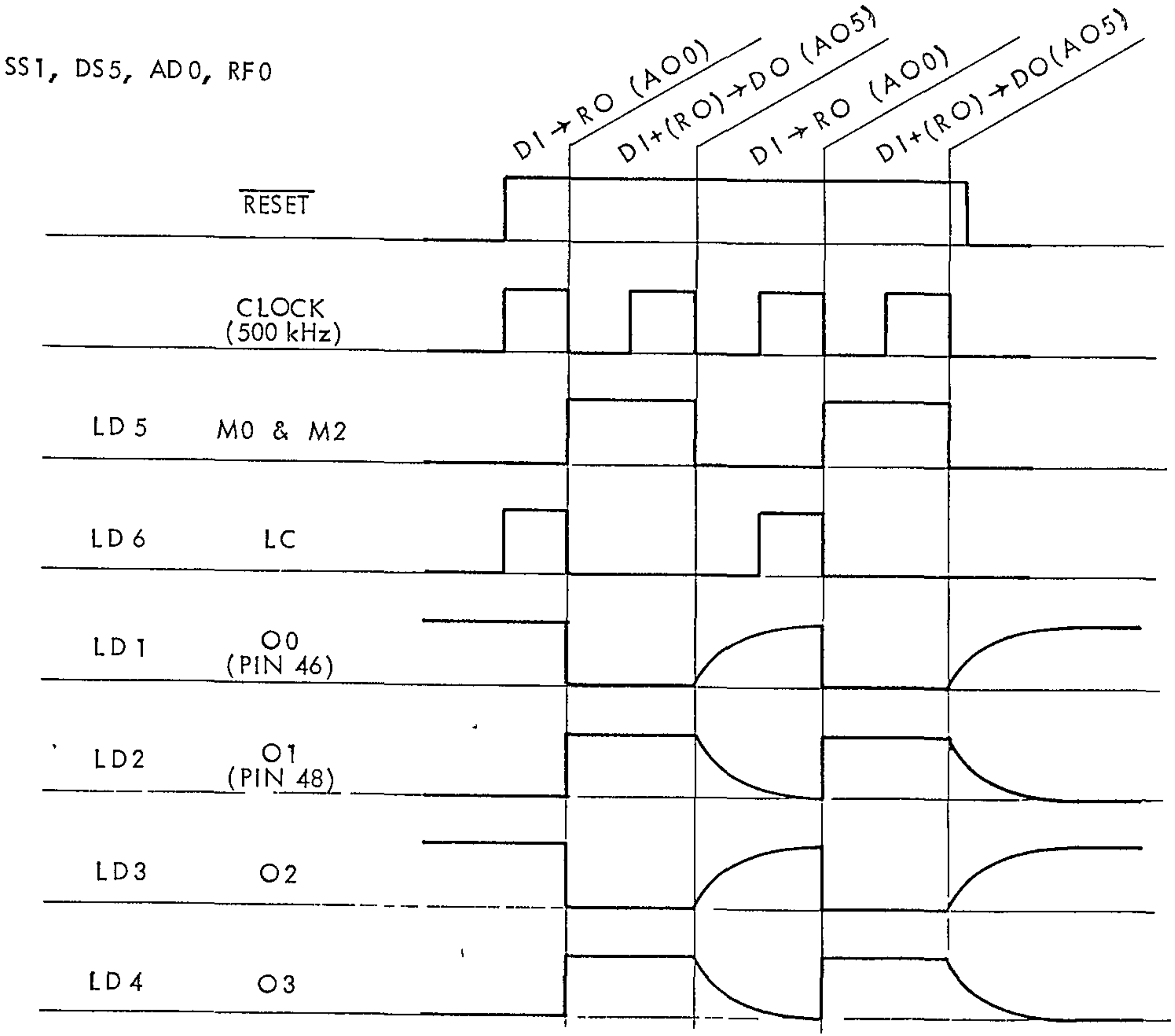


Figure 4. TCS 074 Radiation Test Timing