National Library of Canada | Bibliothèque nationale du Canada

Canadian Theses Division | Division des thèses canadiennes

Ottawa, Canada K1A 0N4

0-315-00451-7.

48010

PERMISSION TO MICROFILM — AUTORISATION DE MICROFILMER

Please print or type — Écrire en lettres moulées ou dactylographier

Full Name of Author — Nom complet de l'auteur

JOHN R. BALCOM

Date of Birth — Date de naissance: Sept. 26 / 1937

Country of Birth — Lieu de naissance: Canada

Permanent Address — Résidence fixe

356 Pleasant St.
Truro N.S. B2N-3T4

Title of Thesis — Titre de la thèse

FUNDAMENTAL DIGITAL COMPUTER SKILLS FOR ELECTRONIC TECHNICIANS

University — Université

SAINT MARY'S UNIVERSITY

Degree for which thesis was presented — Grade pour lequel cette thèse fut présentée

MASTER OF ARTS

Year this degree conferred — Année d'obtention de ce grade: 1980

Name of Supervisor — Nom du directeur de thèse: Dr. B. Davis

Permission is hereby granted to the NATIONAL LIBRARY OF CANADA to microfilm this thesis and to lend or sell copies of the film.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

L'autorisation est, par la présente, accordée à la BIBLIOTHÈQUE NATIONALE DU CANADA de microfilmer cette thèse et de prêter ou de vendre des exemplaires du film.

L'auteur se réserve les autres droits de publication; ni la thèse ni de longs extraits de celle-ci ne doivent être imprimés ou autrement reproduits sans l'autorisation écrite de l'auteur.

Date: May 14/81

Signature: John R Balcom

NOTICE

The quality of this microfiche is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us a poor photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this film is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30. Please read the authorization forms which accompany this thesis.

AVIS

La qualité de cette microfiche dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de mauvaise qualité.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, examens publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de ce microfilm est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30. Veuillez prendre connaissance des formules d'autorisation qui accompagnent cette thèse.

THIS DISSERTATION HAS BEEN MICROFILMED EXACTLY AS RECEIVED

LA THÈSE A ÉTÉ MICROFILMÉE TELLE QUE NOUS L'AVONS REÇUE

FUNDAMENTAL DIGITAL COMPUTER SKILLS FOR ELECTRONIC TECHNICIANS

A Thesis

Presented to

the Faculty of Graduate Studies

Saint Mary's University

In Partial Fulfillment

of the Requirements for the Degree

Master of Arts (Education)

by

John R. Balcom

March 1980

CONTENTS

ABSTRACT

There has been a rapid increase in the use of computers by our society. This increase has been brought about by the introduction of microprocessors and large scale integrated circuits, and the associated reduction in cost per gate.

Internationally recognized computer experts are predicting that we are entering a new age, "the computer age". Many more technicians will be required to install, test and maintain computers.

The purpose of this thesis is to: 1) determine the job skills (digital/computer) of technicians employed in computer electronics, 2) propose a course of study that will train electronic technicians in fundamental digital/computer skills.

The Dacum method of occupational analysis was used to determine computer technicians skills. Representatives from several companies met for a three day workshop and developed the Dacum chart which lists the job skills for a computer technologist.

Two programs designed to teach fundamental skills in microprocessors (Hewlett-Packard and Heathkit) were reviewed to determine if they were suitable for use in Regional Vocational Schools.

The researcher has presented a course of study that will teach fundamental digital computer skills. This course is

developed around a microcomputer trainer using the 6502 chip. It is designed to prepare the student for employment in the digital/computer industry.

The rapid explosion of microprocessors in our society will require a workforce trained in digital/computer technology. This training can and should be carried out in Regional Vocational Schools.

ACKNOWLEDGEMENTS

I wish to thank Dr. B.E. Davis for his direction of this study and Dr. D.L. Burt for editorial assistance.

I wish to thank the following: Mr. Louis Gervais for his expert advice on microcomputers; Mr. John Wilson who was instrumental in developing the Dacum chart for computer technology.

Finally I would like to thank my family for their support and understanding; my wife Alice, who typed the original manuscript, my son Bruce for constructive criticism and my daughters Karen and Carla.

LIST OF TABLES

TABLE

I INTRODUCTION

The evolution of electronic technology over the past twenty years has been so rapid it has often been called a revolution, and even compared to the industrial revolution. Not only are computers in the process of changing society but they have also changed many of the techniques used in modern day research. At the heart of the electronic revolution lies the microprocessor, essentially a complete computer on a single chip no bigger than a dime. Already these devices have been found in games, consumer products, intelligent terminals, sophisticated test equipment to name just a few. In the future we will find microprocessors in our cars, telephones, washing machines or in any device where electronic control can be applied.

A microcomputer, the type used in business and industry, will consist of a microprocessor (the heart of the computer), memory and various input/output devices. These computers are constructed from Large Scale Integration (LSI) chips. These chips contain in excess of one hundred thousand (100,000) transistors on a quarter inch square slab. The speed of the computer has been increased until it will perform in excess of one

billion operations per second. All this, and it will fit in a good size suitcase.

Electronic and computer workers are in short supply. "A group of eighty companies are raising $250,000 for a direct mail advertising magazine to lure electronic engineering graduates and technicians to the Ottawa region".[1] Projections for this region alone are that the Digital and Computer industries will require 4000 to 5000 new electronic and computer workers in the next five years.

Will the youth of Nova Scotia be adequately trained to fill computer related jobs or will they only be qualified to fill rapidly disappearing jobs, using technology of the 50's and the 60's?

The purpose of this thesis is to: 1) determine the job skills (digital/computer) required by the electronics industry, 2) propose a course of study that will train technicians in fundamental digital/computer skills.

DEFINITION OF TERMS

Assembly Language: A computer language that uses mnemonic names to stand for one or more machine language instructions. The advantage of using assembly language instead of a high level language, such as Basic, is speed of execution, but a high level language is usually easier for a human being to understand.

DACUM: (Developing A Curriculum) is an approach to the development of curriculum combined with an evaluation process for occupational training programs. It was created initially in a joint effort by the Experimental Projects Branch, Canada Department of Manpower and Immigration, and General Learning Corporation of New York.[2]

Digital: Having discrete states. Most digital logic is binary, with two states, on or off.

Central Processor Unit (CPU): Computer module in charge of fetching, decoding, and executing instructions. It incorporates a control unit, an Arithmetic Logic Unit, and related facilities (registers, clock, drivers).

Computer: General purpose computing system incorporating a Central Processor Unit (CPU), memory, Input/Output facilities and power supply.

Digital/Computer: The field of electronics that employs either digital circuits or computer hardware and software or both.

Hardware: Any piece of data processing equipment is informally called hardware.

High-Level Language: Problem-oriented programming language, as distinguished from machine-oriented programming language. Examples are Basic and Fortran.

Integrated Circuit: A circuit which is fabricated on a single chip of silicon. Initial integrated circuits contained less than one hundred semiconductors devices on a single chip.

Large Scale Integration (LSI): Technology by which thousands of semiconductors devices are fabricated on a single silicon chip.

Machine Language: Set of binary codes, representing the instructions which can be directly executed by the processor.

Microcomputer: Complete system, including CPU, memory, Input/Output interfaces and power supply. The CPU is normally a Microprocessor.

Microprocessor: LSI implementation of a complete processor (Arithmetic Logic Unit / Control Unit) on a single chip.

Software: The programs that are entered in the computer.

DELIMITATIONS

The purpose of this study is to determine the kind of job skills required by a computer technologist and to develop course materials that will train electronic students in fundamental digital/computer technician skills.

This study will not determine all the skills required

by an electronics technician but will limit the field to digital/computer electronics.

Many of the skill training techniques in the proposed digital/computer technology course can be applied to a range of levels from technician to engineer. However, this course was not meant to teach engineering design skills but fundamental skill development in digital/computer techniques.

The Dacum chart, prepared during this study, showed that there are specific mechanical skills required by a computer technologist. This aspect of training, mechanical skills, has not been included in the proposed Digital/Computer Technology course. See Chapter VIII for recommendations regarding training of mechanical skills.

Related subjects (Math, Physics, Communications) are not covered by this study, however, a number of employers indicated the importance of fundamental skill development in math. For recommendations on communications skills see Chapter VIII.

LIMITATIONS

The final draft of the Dacum chart was not received until late February therefore it was impossible to receive input from industries other than those that directly participated in chart development. Possibly the chart would more accurately reflect the digital/computer industry with

input from a broader spectrum of industries.

There was insufficient time between the completion of the study and its presentation to have representatives of the digital/computer industry comment on the course content. Before implementation of the Digital/Computer Technology course input should be sought from industry regarding course objectives.

No evaluation techniques for the teacher are included in the proposed digital/Computer Technology course.

Most of the learning activities have been field tested, however, lack of hardware has meant that some activities are presented without prior testing.

II HISTORY OF COMPUTERS

The earliest recorded digital instrument was the abacus, it was first used in Egypt about 460 B.C. The abacus evolved from the use of pebbles laid in rows on the sand and used for counting purposes. The pebbles were simply held together by a string to form the first abacus. In the middle ages the abacus moved from Europe to Asia and now is very popular in Russia, China and Japan.

The first automatic digital computer was Pascal's mechanical calculating device (1641).[3] Both addition and subtraction could be performed on Pascal's instrument. Later (1694) Leibniz advanced the design to do repeated additions and subtractions. Neither Pascal or Leinbniz were engineers (they were mathematicians), their machines were not well constructed and were sometimes not reliable.

Probably the most notable contribution to modern computers was made by Charles W. Babbage. Babbage is sometimes called the grandfather of modern computers. His first invention (1822) was a difference engine, it was used to facilitate the calculation of insurance tables.

However his fame rests on the ANALYTIC ENGINE. This was considered by Babbage, in 1833, as a general purpose calculator as opposed to the specific purpose difference engine. Babbage saw his engine as composed of several smaller engines, each working together with the others, each performing his own separate chore: the "mill", which did the arithmetic: the "receiver" to take in information: the "printer" to print out information: a device to transfer information from one component to the other; and a "store" of information.[4] Unfortunately his idea was 100 years ahead of the technology, he spent the rest of his life working on the analytic engine, but without success.

In 1890 the time required to process the United States census was reduced dramatically. The census was compiled with the use of a Hollerith computing machine. Essentially it was a card sorter with data supplied to the machine by hollerith cards. Hollerith's company eventually became the International Business Machines Corporation (IBM).

Howard Aiken, a math teacher with a doctorate in physics, combined with IBM and Harvard to build the first "electric" computer. This computer, completed in 1944, was called the Mark 1 and used relays for storage. The machine was very large and data was fed in by punched tape.

The first "electronic" computer was built by Eckert and Mulchy at the University of Pennsylvania (1946). Called the ENIAC it could multiply two ten digit numbers in three one-thousandth of a second. It contained 18,000 vacuum tubes and occupied a room forty by twenty feet.

Probably the greatest contribution to computer programming was made by John von Neumann in 1946. His idea was to have both instructions and data stored in memory, this way the computer could be used to change its own program.

In 1948 Bardeen, Brattain and Shockley, working at Bell labs, developed the transistor. This event spelled doom for the vacuum tube and led to the miniaturization of electronic circuits. It is probably the most significant event in electronics in the first half of the twentieth century.

The Univac 1 (1950) was the first commercial computer, it was manufactured by Remington Rand.

The IBM 650 was the most popular computer of the 50's. The machine was widely used by the insurance, banking and accounting companies. It rented for $50 per hour versus the more powerful IBM 704 at $600 per hour. The 704 was a massive computer requiring a very large room for storage. In 1959 IBM had 90% of the computer market.

The latter part of the 50's saw the introduction of

the transistorized computer. The 650 was replaced by the transistorized 1401, much smaller and faster. The 704 was replaced by the 7040 with a reduced size and price, and increased speed.

In 1964 IBM introduced the model 7010, an upgrade from the 1401 and a new series of computers, the IBM 360 family of computers. This was the first family of compatible computers ranging from small to large. Essentially one could start with a small processor, 33,000 additions per second, and using the same software upgrade to a larger processor, 2,500,000 additions per second. With the IBM 360 came integrated circuits.

The 1970's saw the introduction of the IBM 370 family of compatible computers. They introduced large scale integration to both the arithmetic/logic units and the memory. In addition, "most System/370 models, for example have "Virtual Storage" capability that magnifies the capacity of main memory many times, and enables users to work economically with millions of characters of information."[5]

BIRTH OF THE MICROCOMPUTER

Datapoint, a manufacturer of computer terminals, wanted to have a computer terminal that contained a small computer, that is a "smart" terminal. They contacted Texas Instruments and Intel, leaders in microelectronics, to develop such a system. Intel developed a system around the

8008 microprocessor, however, Datapoint dropped the idea of a single-circuit computer terminal. Intel was left with the technology but no customer. They decided to market the 8008 microprcessor. Since 1973 they have sold more than three million.[6]

Following the 8008 came the Intel 8080, the Motorola 6800, the Ziloz Z80, and the MOS Technology 6502. These are all 8 bit microprocessors as opposed to the 16 and 32 bit central processors used by Digital Equipment Corporation and IBM.

Micro Instrumentation and Telementry Systems (MITS) sold the first microcomputer, based on the 8080. These were Kits, first advertised in Popular Electronics, January 1975, they expected to sell 800 in one year. On the Friday after Popular Electronics was published they received orders for over 400, they went on to sell several thousand.

Radio Shack with its TRS-80, based on the Z-80, was introduced in 1977 and within one year had sold in excess of 100,000.

In 1977, Commodore Business Machines introduced their microcomputer, the Commodore PET based on the 6502, and within one year sales had passed the 25,000 mark.

Several other companies (Apple, Compucolor, Heathkit) are marketing microcomputers based on the microprocessor.

Within the past two to three years sales have passed several thousand.

SUMMARY

Computers have essentially passed through four generations, Vacuum tubes, Transistors, Integrated Circuits, and Large Scale Integrated Circuits, see Table 1. Until the late seventies the history of computers has essentially been the history of IBM. However, as the seventies came to a close the sales volume of microcomputers equalled the combined sales of IBM and all other models of mini and mainframe computers.

Predictions for the future are that the field will be divided up into two distinct classes:

1. Large volume of sales of microcomputers manufactured by a number of manufacturers.

2. Very few large mainframe computers manufactured by a small number of companies dominated by IBM. [7]

The following chart shows how data processing costs and processing time have declined during the past two decades. It represents a mix of about 1700 computer operations, including payroll, discount computation, file maintenance, table lookup, and report generation. Figures show costs of the period not adjusted for inflation.

TABLE 1 FASTER, SMALLER, CHEAPER [8]

	1955	1960	1965	1976
Cost*	$14.54	$2.48	$ .47	$ .20
Processing Time	375 sec.	47 sec.	37 sec.	5 sec.
Technology	Vacuum tubes Magnetic cores Magnetic tapes	Transistors Channels Faster cores Faster tapes	Solid Logic Technology Large, fast disk file New channels Larger, faster core memory Faster tapes	Monolithic memory Monolithic logic Virtual Storage Larger, faster disk files New channels Advanced tapes Microprocessors
Programming	Stored program	Overlapped input/output- Batch processing	Operating system Faster batch processing	Virtual Storage Advanced operating systems Multiprogramming Batch/on-line processing

* Cost per 1700 operations

III HISTORY OF ELECTRONICS TRAINING

in

REGIONAL VOCATIONAL SCHOOLS

in

NOVA SCOTIA

The first program in electronics training offered in Nova Scotia was at the Yarmouth Regional Vocational School. The course was called Electrical and Radio Repair and was taught by George Williams. George is presently Vice-principal at the Nova Scotia Institute of Technology. The goal of the course was to develop skills in radio repair based on vacuum tube technology. The first class enrolled in the three year program in September 1950 and graduated in June 1953.

The second program in electronics was offered by the Halifax Regional Vocational School. This was a course in radio repair similar to the course in Yarmouth. Skills taught were confined to the vacuum tube field. The first teacher was Gerry O'Malley now the Principal of Dartmouth Regional Vocational School.

In 1954 the course content was up-dated in both schools to include television repair and the course was changed to Electronic (Radio and TV repair).

In 1962 four additional Vocational Schools were opened in Nova Scotia, each had an Electronics course. The course was similar to that taught in Halifax and Yarmouth and was called Electronic (Radio and TV). The technology was vacuum tube and the goal of the course was to teach basic electronics with Radio and Television receivers used as the training vehicle. It was felt that general skills in electronics could be taught using radio and television as a model and those skills would then be transferable to other areas of electronics. These new schools were located at Springhill, Stellarton, Sydney and Kentville.

In the mid 60's transistors were introduced to the school curriculum. At that time Halifax revised their curriculum and concentrated their studies in the semiconductor field. The remaining schools modified their curriculum to include semiconductor technology.

During the years 1968-69 seven new vocational schools were opened, each with a course in electronics. These new schools were located in Shelburne, Bridgewater, Dartmouth, Middleton, Truro, Port Hawkesbury and Windsor. All the schools offered a program in Electronics (Radio and TV) similar to that expressed above. In addition Dartmouth, Bridgewater and Kentville offered a course in Electronics (Navigation). This course was designed to teach the skills knowledge and attitudes required to secure employment in

the marine navigation field. The technology employed in both courses was both tube and transistor.

It appears that in the late 60's three closely related programs in Electronics were offered to the youth of Nova Scotia:

1. Electronics (General using radio and TV as a training vehicle.

2. Electronics (Navigation) using radio, TV and Navigation equipment as the training vehicle.

3. Electronics with special emphasis placed on pulse and switching circuits at Halifax Regional Vocational School.

In the early 70's integrated circuits were introduced into the electronics program. It is not clear how this affected training however, some schools, Halifax and Truro, became more involved in projects employing integrated circuits (IC). The navigation course at Kentville, the Electronics (Radio and TV) at Dartmouth and the Electronics course at Windsor were dropped. In Bridgewater the two Electronics courses were merged.

SUMMARY

Until the mid or early 70's electronic programs have been able to keep up with the changes in electronic technology. With the introduction of Large Scale Integrated

circuits and Microprocessors electronics training programs have fallen behind the technology. Normally this would not cause a problem because the introduction of new training materials usually lag the introduction of new technology. However, the application of microprocessors in both industry and consumer products has taken place at such a rapid pace that this typical lag must be shortened.

The 60's have seen the demise of the vacuum tube. The 70's have seen discrete transistors take the same route as the vacuum tube. Therefore, IC and LSI will and should become the training vehicle for electronics in the 80's.

After reviewing the history of electronics in Regional Vocational Schools it would appear that renewal in program materials is required. The purpose of this thesis is to present a rationale for such a renewal and to develop a course of study based on integrated circuits and large scale integrated circuits that would be applicable to Vocational Schools in Nova Scotia.

IV PROJECTIONS FOR THE COMPUTER INDUSTRY

It is very difficult to predict trends in an industry that has undergone such rapid change, especially in the last 5-10 years. However, three trends have been emerging in the latter part of the 70's, they are:

1. Dramatic increase in the sales of microcomputers.
2. Data Communications
3. Word Processing

MICROCOMPUTERS

Traditional methods of selling computers will change. Rather than have a computer salesman order you a computer, most computers (90% by 1990) will be sold out of computer stores. The world's first computer store, "The Computer Store", was opened in 1975. By the end of 1978, over 700 stores were opened in the United States alone. This rapid growth is continuing. Digital Equipment Corporation, the world's leading minicomputer manufacturer, recently opened a computer store in the "Mall of New Hampshire", Manchester, N.H. It was such a success that they have planned or are opening several more in the Boston area.

Computer stores have become successful for two reasons:

1. They give you a chance to examine a variety of different systems before you buy.

2. They eliminate sales representatives, who typically cost forty percent of what you pay for the computer system.[9]

Computer stores will almost exclusively sell microcomputers.

In addition to microcomputers being readily available to the public, especially the businessman, they can now, or will be able to in the future, perform most or all of the functions of a mini or small mainframe computer, and at a much lower cost. In fact, electronics already exists to make microcomputers as powerful as a mainframe. The development of the Intel 8086, is an example of the tremendous progress made in microcomputers.

Prediction: By the mid or late 1980's, ninety percent of all computer sales will be microcomputer systems or their equivalent. The remaining ten percent will be the largest, most powerful and most expensive mainframe computers. These mainframe computers will be dedicated to such things as:

1. Difficult computations associated with weather forecasting and geological data analysis.

2. Control of large information banks.

3. Data processing for state and federal government agencies (for example, the Social Security Administration).[10]

Sales of microcomputers will increase at a very rapid rate, and will shortly exceed 1 million per year. At a recent international computer conference held in Wolfville, Nova Scotia a Pentagon computer expert predicted that computers will become the world's largest industry in the near future, dwarfing North America's automobile industry.

DATA COMMUNICATIONS

The merger of computers and communications continues at a relentless speed, making a distinction between the two terms becomes more difficult every year. As the boundaries between data processing and communications continue to merge, it may well be that data processing firms will offer telecommunication services as well. Electronic mail appears to be a real possibility. It is expected that the main competition in data communication for International Telephone and Telegraph (ITT) and American Telephone and Telegraph (AT & T) will come from IBM and Xerox Corporation.

Canada is presently one of the world leaders in data communications. The Canadian Telidon "alpha-graphic" approach was developed by the Communications Research Centre,

a branch of the Department of Communications.

The Telidon system is made up of four major elements:

1. User terminals

2. Information supplier terminals

3. The telephone network and associated data networks such as data pac and data route

4. A computer for information storage, retrieval and switching centre.[11]

A modified version of Telidon "Vista" is being field tested by Bell Canada Ltd. and Department of Communications.

The Department of Communications and Bell Canada are co-sponsoring trials that will involve 1000 Canadian made user terminals and up to 100,000 pages of on-demand information, for display on home or office color T.V. sets. It will be one of the world's most advanced trials of Videotex, the internationally recognized term for such public, network based information systems.[12]

The Vista system, essentially identical to the Telidon, will employ a conventional Color Television, a control unit to couple the T.V. with a normal telephone and a hand held key pad. Pressing a designated key will convert the T.V. set into a Vista interactive information user terminal. The user will then have access to a Data

base of a PDP 11/60. Prime user groups of the system will be the home users and the business community. In business applications, a full alpha-numeric keyboard will be used rather than a key pad.

Projections are that this kind of system will make massive amounts of information readily available to the general public. Through such systems as "the Source", it is now possible to connect your home computer, Radio Shack, Pet, etc., to a large main frame computer in Washington, D.C. This means that your microcomputer has access to literally hundreds of data banks. The largest growth in the computer and computer related industries will take place in a data communications.

Another Canadian company, AES Data, Ltd., world leaders in word processing systems, are combining with CNCP to develop data communications for the business community.

WORD PROCESSING

Word processing is a combination of hardware and software. Usually it consists of a micro or minicomputer with dual floppy disc, a Cathode Ray Tube (CRT) and a character printer. In addition the computer contains a software package that allows the operator to type text

into the computer and manipulate the text so that it can be printed out in some pre-determined manner. The operator can append clauses, adjust page length, put data into alphabetical and numerical order, columnize, move text, insert and delete, all viewed on a CRT. To review text or processed pages, the operator simply scrolls a cursor either up or down. Files, that is text are stored on floppy disc.

Communicating word processors, data communications, allow the boss to call up a typed letter on his own CRT terminal, do his own editing and then send the electronic letter to, ultimately anywhere in the world. Infotex, for example, will be capable of transmitting a 300 word letter in seven seconds, like a postman moving near the speed of light.[13]

ABS Data of Montreal, formed in 1974 are world leaders in word processing. Projected sales for 1979 are in access of 125 million dollars. It appears that the paperless office has really caught on with business. However, only 15 percent of the market is presently being serviced, therefore, the 80's should see a tremendous growth in word processing.

The large scale marketing of computers means that thousands of new jobs will be created to install and

maintain microcomputers and peripherals. In addition the rapid explosion of word processor equipment, essentially a micro or minicomputer will mean additional maintenance jobs. The fundamental skills required for microcomputer maintenance can and should be taught at Regional Vocational Schools.

The rapid expansion of data communications will provide additional jobs. Many of these skills will be the same as those required for digital/computer electronics. However, additional skills in broadband communication will be required, these should be investigated by the Department of Education and the appropriate training provided for the youth in Nova Scotia.

V INDUSTRY REQUIREMENTS, DACUM CHART

Before developing or reviewing curriculum for training Digital/Computer technicians it was necessary to determine the kind of skills a technician employed in the industry should possess. The following tools were considered in order to find out what these skills were:

1. Inquiry forms filled out in the presence of the questioner. These forms are normally called schedules.

2. The questionnaire, which is probably the most used and abused of data gathering devices.

3. The interview where the questioner asks oral questions and notes or tape records the response.

The first two were considered inappropriate to conduct an industry survey. I discovered that industries are bombarded by questionnaires and they do not normally have the time or interest to respond.

Serious consideration was given to the interview, however, this technique is very time consuming and one of the most difficult to employ successfully. The interviewer must be skilled in asking a sequence of questions and make stimulating comments that will produce the desired results. It would appear that considerable training

is required to make this method successful.

The survey problem was discussed with Dr. D.L. Burt, Director of Instructional Services, Nova Scotia Teachers College. He suggested that I consider an occupational analysis based on the Dacum model.

Developing a Curriculum (DACUM) is a system that encompasses three main components:

1. It is an approach to occupational analysis.

2. It is an approach to planning and developing training based on the analysis.

3. It is an approach to training program operation.[14]

The result of an occupational analysis is a chart (Dacum Chart) produced by a committee of "experts" who participate in a three day workshop. The purpose of this workshop, under the guidance of a skilled co-ordinator, is to identify the skills associated with their occupation. These skilled experts are normally technicians or working foremen employed in the occupation to be analyzed. In this case they would be technicians in the occupational field of Digital/Computer Electronics.

The selection of the Dacum process to determine Digital/Computer technician skills was based on the following:

1. Several industries and educational institutions rely on the Dacum method for occupational analysis of a trade or technology. A few of these institutions are:

Nova Scotia Institute of Technology (N.S.I.T.)

Adult Vocational Training Centers in Nova Scotia

Holland College

Nova Scotia NewStart Inc.

Nova Scotia Land Survey Institute

Scott Paper Company

Ferguson Industries

National Sea Products

2. The Dacum method of occupational analysis has been used by the Electronics Technology program at Holland College since its inception. Sorenson in his article, A Summary of the Research "Entry Level Skills-Electronics", describes the application of a Dacum chart on Electronics Technology.[15] This survey illustrates the importance of the Dacum method of occupational analysis at Holland College.

3. Local expertise was available for developing a Dacum chart. The Adult Vocational Education program of the Department of Education has a resource facility, under the direction of Mike Kent, for developing Dacum Charts. This facility has developed charts for more than

twenty occupational fields, typical are: Electronic Repair; Radio Announcing and Stenography.

An application, in conjunction with the Nova Scotia Institute of Technology, was made to the Department of Education, Adult Education for approval to construct a Dacum chart for the occupational field of Digital/Computer Electronics.

The application was approved, January 28, 29, and 30th were selected as the days to construct the chart. Jim MacLennan, Department of Education, Adult Education was appointed project co-ordinator.

The following companies provided workshop participants:

1. Department of National Defence (Dockyard), participant Robert George.

2. Bedford Institute of Oceanography, participant Sidney Specce.

3. Control Data Corporation, participant Edward Billerwall.

4. Maritime Telephone and Telegraph, participant Ernest MacPherson.

5. Defence Research Eastern Atlantic, participants Vance Crowe and Howard Hart.

6. National Cash Register, participant Gordon Heffler.

Table 11 is the Dacum Chart containing the skills identified in the workshop. These are actual on the job skills and are representative of the Digital/Computer industry. This chart can be used as a blueprint in developing course materials to train Digital/Computer technicians. That is, fundamental skills taught in Vocational and technical schools should provide a solid background for actual on the job skills performed by industry.

WORK AS A MEMBER OF A TEAM

READ SCHEMATICS

REPRESENT YOUR COMPANY OR FIRM

COMPLETE TECHNICAL REPORT FORMS

WRITE TECHNICAL REPORTS

USE MULTIMETERS

USE DIGITAL VOLT METERS

USE OSCILLOSCOPES

USE LOGIC PROBES

USE LOGIC PULSERS

USE CURRENT PROBES

SOLDER

UNSOLDER

MAKE SOLDERLESS CONNECTIONS

FOLLOW SPECIAL HANDLING TECHNIQUES AROUND SPECIAL COMPONENTS

FOLLOW MANUFACTURERS' EQUIPMENT HANDLING INSTRUCTIONS

REPLACE DRIVE BELTS

REPLACE

CABLE EQUIPMENT TOGETHER

CONNECT (OR INSPECT) EQUIPMENT TO POWER SOURCES

INSTALL AND GROU

CLEAN EQUIPMENT

CLEAN OR CHANGE FILTERS

CLEAN PRINT DRUMS

CLEAN AND HANDLE MAG TAPES

CLEAN AND HANDLE DISC PACKS

RECORD MAINTENANCE

FOLLOW MA ROUTINES

MAKE UP INTERFACE CABLES

CONFIGURE OPTIONS

DE-BUG PROTOTYPES

SELECT COMPONENTS TO BE USED TO MEET PARAMETERS

DES CON

USE SYSTEM CONTROL LANGUAGES

1 OF

WRITE TECHNICAL REPORTS

CONSULT ON PHONES TO SOLVE PROBLEMS

WORK UNDER PRESSURE

GIC PULSERS

USE CURRENT PROBES

MEASURE ENVIRONMENT WITH THERMISTERS AND SWINGING PSYCHOMETERS

USE TACHOMETERS

USE ELECTRONIC FREQUENCY COUNTERS

CHECK GROUNDS WITH NOISE GENERATORS

USE BIT ERROR ANALYZERS

REPLACE DRIVE BELTS

REPLACE GEARS

REPLACE BRUSHES IN MOTORS

REPLACE BEARINGS IN MOTORS OR SHAFTS

REPAIR, REPLACE, OR ADJUST CLUTCHES AND BRAKES

IDENTIFY REPAIRABILITY OF CIRCUIT BOARDS

CONNECT (OR INSPECT) EQUIPMENT TO POWER SOURCES

INSTALL SHIELDING AND GROUNDING

DIRECT WORKERS TO MOVE EQUIPMENT

DRAW EQUIPMENT LAYOUTS

RECORD MAINTENANCE

FOLLOW MAINTENANCE ROUTINES

CHANGE BLOWER MOTORS

CLEAN OR REPLACE PINCH ROLLERS IN READERS

CHECK POWER SUPPLIES

CLEAN AND ALIGN HEADS

RUN DIAGNOSTICS ON EQUIPMENT

OPERATE EQUIPMENT TO BE DIAGNOSED

SURVEY COMPLETE SYSTEM FOR OBVIOUS PROBLEMS

USE BUILT-IN TEST FEATURES

USE DIAGNOSTIC PROGRAMS

CONFIGURE OPTIONS

OBTAIN CONFIGURING INFORMATION

INTERPRET TECHNICAL MANUALS TO OPERATE TERMINALS

SELECT DIGITAL INTERFACES

CT COMPONENTS E USED TO PARAMETERS

DESIGN AND CONSTRUCT CHASSIS

ADJUST PROTOTYPE TO FINAL PACKAGE

CONSTRUCT PROTOTYPES

DOCUMENT END PRODUCT

USE SYSTEM CONTROL LANGUAGES

LOAD AND RUN PROGRAMS

2 OF

MANIPULATE FILES

USE MACHINE CODE LANGUAGES

CHECK GROUNDS WITH NOISE GENERATORS

USE BIT ERROR RATE ANALYZERS

CHECK COMMUNICATION BAND MIXING WITH FREQUENCY GENERATORS

USE LOGIC ANALYZERS

BILITY OF BOARDS

REPLACE COMPONENTS

OBTAIN REPLACEMENT PARTS

MATCH REPLACEMENT COMPONENTS

REPAIR CIRCUIT BOARDS

CALIBRATE AND CHECK EQUIPMENT

WORKERS E EQUIPMENT

DRAW EQUIPMENT LAYOUTS

RUN UP EQUIPMENT

WRITE PRE-INSTALLATION SPECIFICATIONS FOR SUPPORT EQUIPMENT AND SERVICES

ESTABLI REQUIRE

LEAN AND ALIGN EADS

RUN DIAGNOSTICS ON EQUIPMENT

RUN MANUAL CHECKS

ZERO SYNCROS

CHECK AND ALIGN POTENTIOMETERS

SERVICE AND CALIBRATE SERVOS

CHECK AND SERVICE A. TO D. 3 OR D. TO A. 3

ESTAB PARAM INVER

LF-IN TEST

USE DIAGNOSTIC PROGRAMS

SELECT DIAGNOSTIC PROCEDURE

USE DECISION FLOW CHARTS

COMPARE ACTUAL TEST SEQUENCES TO RESULTS EXPECTED

EXTRACT TROUBLE INFORMATION FROM USERS

IDENTIFY SECTI OF SYSTEM THAT IS FAULTY

GITAL S

DO DIGITAL TESTS ON DATA

DO ANALOG TESTS ON LINES

DOCUMENT END PRODUCT

MAKE BLOCK DIAGRAMS, COMPONENT LAYOUTS, AND SCHEMATICS

IDENTIFY PARAMETERS ON EQUIPMENT TO BE DESIGNED OR MODIFIED

IDENTIFY NEED FOR MODIFICATIONS

DOCUM MODIF APPRO

USE MACHINE CODE LANGUAGES

BUILD, COMPILE, AND RUN ASSEMBLY LANGUAGE PROGRAMS

3 OF

BUILD, COMPILE, AND RUN HIGH LEVEL LANGUAGE PROGRAMS

BUILD TE FROM TAP

COMMUNICATE

USE TEST EQUIPMENT

REPAIR EQUIPMENT

- PAIR CIRCUIT ARDS
- CALIBRATE AND CHECK EQUIPMENT
- CHECK ASSEMBLY OF EQUIPMENT FOR COMPLETENESS
- CHECK COMPONENTS FOR TEMPERATURE SENSITIVITY

INSTALL EQUIPMENT

- WRITE PRE-INSTALLATION SPECIFICATIONS FOR SUPPORT EQUIPMENT AND SERVICES
- ESTABLISH POWER REQUIREMENTS
- ESTABLISH COOLING REQUIREMENTS

MAINTAIN EQUIPMENT

- CHECK AND SERVICE A. TO D. B. OR D. TO A. B
- ESTABLISH AND MAINTAIN PARTS INVENTORIES
- ESTABLISH MAINTENANCE ROUTINES

ANALYZE SYSTEM FAULTS

- EXTRACT TROUBLE INFORMATION FROM USERS
- IDENTIFY SECTION OF SYSTEM THAT IS FAULTY
- WRITE DIAGNOSTIC PROGRAMS
- USE SOFTWARE TRAPS OR TEST REPEATS
- DETECT HARDWARE FAULTS FROM SOFTWARE

INTERFACE COMMUNICATIONS EQUIPMENT

- SPECIFY MODEMS AND FACILITY REQUIREMENTS
- IDENTIFY TRANSMISSION CODES AND PROTOCOL

DESIGN, MODIFY AND CONSTRUCT INTERFACES AND EQUIPMENT

- VERIFY NEED MODIFICATIONS
- DOCUMENT PROPOSED MODIFICATIONS FOR APPROVAL
- SELECT MICRO PROCESSOR COMPONENTS
- MAKE SYSTEMS USING MICRO PROCESSORS

OPERATE COMPUTERS FOR TESTING RELIABILITY

- BUILD TEST SYSTEMS FROM TAPE FILES

4 of

COMMUNICATE

USE TEST EQUIPMENT

REPAIR EQUIPMENT

ASSEMBLY OF NT FOR ERESS

CHECK COMPONENTS FOR TEMPERATURE SENSITIVITY

INSTALL EQUIPMENT

ESTABLISH COOLING REQUIREMENTS

MAINTAIN EQUIPMENT

ESTABLISH MAINTENANCE ROUTINES

ANALYZE SYSTEM FAULTS

WRITE DIAGNOSTIC PROGRAMS

USE SOFTWARE TRAPS OR TEST REPEATS

DETECT HARDWARE FAULTS FROM SOFTWARE

INTERFACE COMMUNICATIONS EQUIPMENT

IDENTIFY TRANSMISSION CODES AND PROTOCOL

DESIGN, MODIFY AND CONSTRUCT INTERFACES AND EQUIPMENT

SELECT MICRO PROCESSOR COMPONENTS

MAKE SYSTEMS USING MICRO PROCESSORS

OPERATE COMPUTERS FOR TESTING RELIABILITY

RATING GUIDE

5	CAN PERFORM THIS SKILL SATISFACTORILY AND CAN LEAD OTHERS IN PERFORMING IT.
4	CAN PERFORM THIS SKILL SATISFACTORILY WITH INITIATIVE AND ADAPTABILITY TO SPECIAL PROBLEM SITUATIONS.
	CAN PERFORM THIS SKILL SATISFACTORILY WITH MORE THAN ACCEPTABLE SPEED AND QUALITY.
3	CAN PERFORM THIS SKILL SATISFACTORILY WITHOUT ASSISTANCE AND/OR SUPERVISION.
2	CAN PERFORM THIS SKILL SATISFACTORILY BUT REQUIRES PERIODIC ASSISTANCE AND/OR SUPERVISION.
1	CAN PERFORM SOME PARTS OF THIS SKILL SATISFACTORILY BUT REQUIRES ASSISTANCE AND/OR SUPERVISION TO PERFORM THE ENTIRE SKILL.

COMPUTER TECHNOLOGY

ADULT EDUCATION
WICHITA, KANSAS

JANUARY '80

5 OF 5

VI REVIEW OF TRAINING PROGRAMS
in
DIGITAL/COMPUTER ELECTRONICS

Three methods were used to obtain curriculums or outlines of courses that teach technical skills in digital/computer electronics.

A. The writer contacted several Community Colleges and Technical Institutes, New Brunswick Community College, Northern Alberta Institute of Technology, to name just a few, for a copy of their course outline or curriculum used to train digital/computer technicians.

B. A computer search on three data bases at Lockheed Information Systems, 3251 Hanover Street, Palo Alto, California 94304. The data bases are:

1. ERIC (Educational Resources Information Centre)
2. COMPENDEX (Engineering Index Inc.)
3. INSPEC (Institute of Electrical and Electronic Engineers).

For further information contact Mr. Douglas Vaisey, Head Information Services, St. Mary's University.

C. The writer contacted three companies, HeathKit, Hewlett-Packard, Intel Corporation, that produced

commercial training programs in microprocessors.

A. The Community Colleges that replied to A above did not offer specific training for computer technologist, however, they do supply technicians for the computer industry. The usual method is to offer options in digital circuits, microprocessors and computers along with the usual electronic options. The student could select courses that provide training in digital computer technology but programs designed to specifically train digital/computer technicians were not available. Some institutions, NSIT and New Brunswick Community College have discussed a computer technology option.

A reply to my request for program information is included in Appendix C.

B. The computer search turned up a number of possibilities but again nothing concrete. The data bases contained in the computer receive their information from articles or journals written by University teachers or professional engineers. The INSPEC data base contains articles from journals published by the Institute of Electrical and Electronic Engineers (IEEE). These articles are written for the professional electronics engineer. Unfortunately course outlines for professional engineers are not applicable to Regional Vocational Schools.

The following found in the ERIC Data Base.

ED 146012, COED Transactions, Vol IX, No. 6, June 1977.

An Introductory Course in Microprocessors and Microcomputers.

Marcovitz, Alan B., Ed.

American Society for Engineering Education, Washington, D.C., Computers in Ed. Division.

14 p. June 1977

COED Transactions, ASEE,

P.O. Box 308, West Long Branch, New Jersey.

Again the above program was written with the design engineer in mind.

COMPENDEX is an engineering data base.

C. I was able to obtain on loan, two programs designed to teach fundamental skills in microprocessors.

1. Practical Microprocessors with companion Microprocessors Lab by Hewlett-Packard.

2. Microprocessors including Microprocessor Trainer by HeathKit Continuing Education.

Two criteria were used to review the above curricula:

1. Does it meet the requirements of the Electronic Industry? The Dacum Chart for computer technology was used for comparison purposes. This chart presents a composite of skills and no course should attempt to teach

all the skills.

2. Are they suitable for use by Vocational or Technical Schools?

PRACTICAL MICROPROCESSORS by Hewlett Packard

Educational Objectives:

a. Acquire a practical knowledge of microprocessor system hardware.

b. Gain a basic understanding of the software that is used to control a microprocessor system.

c. Learn how the system uses this software to perform a wide variety of operations.

d. Use this information to learn practical trouble-shooting techniques that are applicable to any microprocessor system.[16]

A complete outline is contained in Appendix D.

Does it meet the requirements of the Digital/Computer Industry as identified by the Dacum Chart? The general areas of competence will be considered.

A. Communicate. These kind of skills are not covered by the Practical Microprocessors.

B. Use Test Equipment. Test equipment designed to test digital/computer equipment is explained in detail.

C. Repair Equipment. Mechanical skills such as replacing, repairing and adjusting are not part of this

course, however, they can be learned on the job.

D. Install Equipment. Not covered by this course.

E. Maintain Equipment. Mechanical skills are not covered. A major omission of this program is training in analog to digital and digital to analog conversion. This program provides training in running diagnostics.

F. Analyze System Faults. Provides excellent fundamental training for this area of competence.

G. Interface Communication Equipment. Again fundamental skills developed that are necessary to perform these operations.

H. Design, Modify and Construct Interfaces and Equipment. Skills in assembly language programs only.

I. Operate Computers for Testing Reliability. Requires complete system to develop these skills.

Are they suitable for use by Vocational or Technical Schools?

ADVANTAGES

The text "Practical Microprocessors", used in conjunction with the HP 5036A Microprocessor Lab provides an excellent introduction to hardware, software and troubleshooting of the microprocessor. A variety of learning experiences are provided. The development of

concepts is sequential and the skills learned in trouble-shooting are transferable to other systems. The micro lab has two design features that provide a significant contribution to understanding how a microprocessor works. The hardware step that allows you to process one bit of information at a time, and the instruction step that allows you to execute one instruction at a time.

DISADVANTAGES

The major difficulty in selecting this program for use in a Vocational or Technical School is the price, $1,000. per work station. Considering 16 work stations, this is a considerable cost. In addition the Microprocessor Lab is only a trainer and not a full blown computer, this limits application in interfacing, etc.

INDIVIDUAL LEARNING PROGRAM IN MICROPROCESSORS by HeathKit CONTINUING EDUCATION.

Educational Objectives: When you complete this program you will be able to:

a. Program a representative microprocessor

b. Interface a representative microprocessor with the "outside world".[17]

A complete outline is contained in Appendix D.

Does it meet the requirements of the Digital/Computer Industry as identified by the Dacum Chart? The general areas of competence will be considered.

A. Communication. These skills are not covered by Microprocessors.

B. Use Test Equipment. Test equipment is not mentioned in this program.

C. Repair Equipment. Mechanical skills such as replacing, repairing and adjusting are not part of this course, however, they can be taught by a companion course or learned on the job.

D. Install Equipment. Not covered.

E. Maintain Equipment. Provides training in running diagnostics only.

F. Analyze System Faults. Provides excellent fundamental training for this area of competence; however, troubleshooting skills are not included. These skills are essential to digital/computer technicians.

G. Interface Communication Equipment. This program contains an introduction to interfacing using the PIA.

H. Design, Modify and Construct Interfaces and Equipment. This course does not teach design skills.

I. Operate Computers for testing reliability.

Skills in assembly language programs only.

Are they suitable for use by Vocational or Technical Schools?

ADVANTAGES

The text "Individual Learning Program in Microprocessors" used in conjunction with the ET-3400 microcomputer trainer provides a methodical program to introduce students to microcomputers.

It concentrates on programming and interfacing to the real world. Introduction to computer architecture provides an easy transition for those new to microprocessors.

Selection of the 6800 for the microprocessor trainer is a plus because it has an easier instruction set to learn than the more popular 8080. Because the 6800 uses memory mapped I/O, interfacing is simplified especially when used with the PIA.

The cost, less than $400., is a realistic price for a computer trainer used in a Vocational or Technical school.

DISADVANTAGES

Unfortunately it does not include a section on troubleshooting, and after all, that is how we technicians make our living. The on board memory is too small to

write very extensive programs.

SUMMARY

No one course of study can teach all the skills listed on the Dacum Computer Technology chart. This chart is a composite of the skills of many technicians. Therefore a course of study should provide fundamental skills that allow one to become employed and progress in the industry.

Practical Microprocessors in conjunction with the Microprocessor Lab would develop fundamental skills in computer technology although it has limited application in the real world.

Microprocessors including the Microprocessor trainer develops skills in programming and interfacing, however, because of its hexadecimal keypad and limited memory it has little application in the real world.

I propose to develop a course of study that will teach fundamental computer skills on a computer that has application in the real world. This course will be developed around a microcomputer trainer.

SELECTION OF THE AIM 65 MICROCOMPUTER

The November 15, 1979 issue of Creative Computing contained an article comparing twenty-six one board computers (cost under $550.00) that are available for

teaching students hardware and programming fundamentals and dedicated controller applications. This was the most complete list of one board computers that the writer could find. The article included the HeathKit ET 3400 but excluded the Hewett-Packard HP 5036A because of its cost.

From the list of computers contained in the article the writer selected Rockwell's R6500 Advanced Interactive Microcomputer, the AIM 65, as a computer trainer for the proposed Digital/Computer Technology course.

The AIM 65 was selected for the following reasons:

a) The cost is reasonable, four hundred and fifty dollars American.

b) The AIM 65 contains sufficient memory (4K) for extensive programs.

c) The AIM 65 contains a full keyboard versus the keypad on most one board computers.

d) The AIM 65 comes complete with a Versatile Interface Adapter suitable for interfacing with the "real" world.

e) The AIM 65 has a 20 character alpha-numeric display versus the six digit hexadecimal display found on the ET 3400 and the HP 5036A as well as on most single board computers.

f) The AIM 65 has an on-board Advanced Interactive Monitor (8K) program that provides extensive control and program development functions.

g) ROM space is available for a plug in Basic interpreter or an Assembler. This feature is unique in one board computers, it allows the AIM 65 to be used for other than machine language programming.

h) Interfacing for a TTY and Cassette is included with the AIM 65.

i) The AIM 65 has extensive documentation. The computer is supplied with five different user manuals.

j) The AIM 65 includes an on-board printer. For educational purposes the printer is its most significant feature. The printer provides instant feedback and a permanent record for each operation the student performs.

The AIM 65 offers flexibility and expandability normally associated with only a sophisticated microcomputer development system. Its potential as an educational trainer of microprocessor systems is very extensive.

VII SUMMARY OF THE PROPOSED COURSE OF STUDY

In order to meet the requirements of industry, there is a need to introduce Large Scale Integration (LSI) and microprocessor technology into the electronics classroom. Microprocessors can provide products with improved reliability, performance, features and sophistication. With these improvements come new service, troubleshooting and repair problems.

A computer has two requirements in order to perform operations, they are:

1. Computer hardware, that is the component parts of the computer.

2. Software, that is the program that causes the computer to do its thing.

If either is defective or missing the computer will not function properly.

This means electronic technicians must learn troubleshooting skills in two areas.

1. How to troubleshoot computer software, and
2. How to troubleshoot digital logic circuits (Hardware).

Special tools are available for troubleshooting digital/computer circuits. They can be separated into two classes.

1. Inexpensive logic probes, logic pulsers, and current tracers.

2. Signature analyzers and wide band oscilloscopes used to trace programs and isolate problems.

Special problems are encountered in computers, in that they cannot normally be stopped to observe each individual operation as in logic circuits. Measurements must be taken while the processor is running and often data presented on the data bus is meaningless because of three-state outputs.

As with any circuit a technician is trying to analyze or troubleshoot, it is helpful to become familiar with the circuit. Studying the theory of operation, the block diagram, and the schematic, provides a base of knowledge from which to work.

The purpose of this course of study is to enable the student to develop technical entry level skills and knowledge in the field of Digital/Computer Electronics.

These materials have been written with the student in mind, therefore, the following features have been developed:

1. Reference is made to an extensive Glossary.

2. The Bibliography, contained in the main body of the thesis, includes journals and articles as well as texts. This will be of assistance to those students who wish to expand on their knowledge.

3. The majority of the materials are designed to be interactive; that is, the student will read a short introduction and then be asked to perform an operation. The writer has tried to keep student activity (learn by doing) at a fairly high level and leave extraneous wordage for others.

4. Several of the learning activities may seem incomplete, however, sufficient information has been provided so that the student, after some investigation, will be able to complete all the exercises.

5. Although some of the materials may be suitable for independent learning, they were not designed that way. The teacher and most of the books and articles in the Bibliography, are considered as primary resources for the student.

6. The writer has tried to separate concepts by having one concept per page, especially in the digital logic section. This has not always been possible; however, there should be a definite break when a new

concept is introduced.

7. Persons entering the course should have completed Math and English to the Grade XII level. In addition, they should have completed a course in Basic Electronics or the first year of a two year program in a Regional Vocational School. Suggested titles for a Course in Basic Electronics are:

Basic Electronics (Grob)

Applied Electronics Circuits (Weick)

Basic Mathematics for Electronics (Smith or Cooke and Adams)

Basic Electronics Workbook

Applied Electronics Workbook or projects

Fundamental skills (AC and DC) can only be learned by applying them. Projects should be developed that incorporate knowledge studied in the classroom. Listening and seeing is not enough for a skill to be learned, it must be applied in a realistic situation. We learn by doing and thinking about what we are doing.

Probably the most appropriate project for a first year student, is to design and construct a regulated power supply. After he builds it, give it to him!

TABLE OF CONTENTS FOR DIGITAL/COMPUTER TECHNOLOGY

Contents of the course are contained in Appendix A.

I Programming in Basic

The writer decided to introduce students to computers via the Basic Language. Basic (Beginners All Purpose Symbolic Instruction Code) is a high level language and is relatively easy to learn. This allows the student to gain "hands on" experience with the microcomputer in a relatively non-threatening atmosphere.

This chapter could be started at any time and the student can proceed at his own speed. It is hoped that once the student started the program he would want to complete it. There are many awkward formulas in AC theory that are quite nice to program, the student may

find it beneficial to complete this chapter while he is studying Basic Electronics. Once students are comfortable with the Basic language, it will provide a nice transition to low level assembly language.

II Digital Logic Circuits

The chapter on Logic Circuits serves two purposes.

1. It provides the necessary background for the study of computers. Most courses on Microprocessors require as a prerequisite, completion of a course in Digital Techniques.

2. It is designed as a "stand alone" introduction to Digital Logic.

Unit A, Semiconductor Review

The purpose of this unit is to provide a short review of semiconductors and to consider the transistor as a switch, that is the transistor is either fully on (conducting) or off (non-conducting). Digital integrated circuits consist of a number of transistors performing these two operations.

Unit B, Number Systems

The relationship between binary, octal, hexadecimal and the decimal system is shown and a number of exercises are provided so that the student can gain

skill in converting between the four systems. Examples are shown and exercises provided for binary addition and subtraction and a method is developed for addition of octal and hexadecimal numbers. The use of the microprocessor shift instruction is employed to illustrate binary multiplication and division. The concept of BCD is developed and several exercises are provided.

Unit C, Logic Levels

Introduces the student to the concept of logic levels, i.e., 1 and 0; Hi and Lo. Pulses and clocks are introduced and terminology associated with each is described. Characteristics of digital integrated circuits are compared as well as those of microprocessors. A brief explanation is given of the fabrication of integrated circuits.

Unit D, Introduction to the Logic Tester

The rest of the projects in this chapter and many of the projects in the Computer Architecture chapter require the use of a logic tester. This unit presents fundamental instruction so that a student could construct his own logic tester. Included are a list of materials, schematic diagram, printed circuit (PC) board layout, hardware

location etc., and information to enable the student to construct a printed circuit board. Under the guidance of a teacher, the student could construct this project without having studied digital circuits. In any event, a similar tester will be required to complete this chapter.

Unit E, Logic Gates

A number of learning activities have been developed so that the student can gain confidence in the use of the following basic gates: AND, OR, NAND, NOR, E-OR, EN-NOR, INVERTER, BUFFER, TRI-STATE LOGIC. From these basic gates, all other digital logic gates can be constructed. These learning activities have been designed to give the student the key facts and let them discover how the gates actually work. The teacher should demonstrate setting up the first few activities and monitor all results.

Unit F, Boolean Algebra

George Boole developed a number of postulates (rules) that apply to binary numbers. Many authors and circuit designers use Boolean Algebra to describe circuit operation. Therefore, for technicians, etc., an introduction to Boolean Algebra is required. A number of exercises in the book PRACTICE PROBLEMS in NUMBER SYSTEMS, LOGIC and BOOLEAN ALGEBRA (by Edward Bukstein), have been assigned

to develop skills in relating circuit function to Boolean equations.

Unit G, Sequential Logic Circuits

Sequential Logic Circuits are used in a variety of timing, sequencing and storage functions. The output of a sequential logic circuit is not only a function of its input circuit but also a result of previous operations that may have been stored in the circuit itself. There can exist an almost infinite variety of sequential logic circuits. A representative sample of the most common types are presented in these learning activities.

Unit H, Combinational Logic Circuits

Combinational Logic Circuits are digital circuits that are made up of gates and inverters. As with sequential logic circuits, a representative sample of the most common types are presented in the learning activities. After completing these activities two practical projects are assigned so that the student can apply the skills learned.

III Computer Architecture

All computers, whether micro or maxi, will require certain basic elements. The important thing is to learn how to identify these elements in any computer you

will use. Once this is done you can analyze variations on the basic theme. To fail to learn these basic elements will leave students vulnerable to those "new and improved" computers which always seem to be coming along. As with all engineering activities, there are a few truly basic ideas. Most build on existing activities. Learn the basics and the details will take care of themselves.

This chapter is presented in two parts:

A. Introduction to the Microcomputer

B. A Real Computer

Unit A, Introduction to the Microcomputer

The student is introduced to the microcomputer via a block diagram of a pseduo microprocessor (MPU). This is a stripped down version of a real MPU, its purpose is to assist in illustrating signal flow in a microcomputer. The stored program concept, developed by John von Neumann, and implemented on all computers is used to illustrate fetching and executing machine instructions.

A number of terms used with the elementary microprocessor are defined and an illustration of the arithmetic logic unit (ALU) is included.

The fetch execute method used to process instructions in all microcomputers is introduced. Each machine

state is shown as the computer sequences through a program. Only three machine instructions are introduced at this time and the immediate mode of addressing (operand is contained in the next address) is employed.

Direct or zero page addressing is introduced, that is the operand associated with the instruction is contained in the low part of memory from 00 Hex to FF Hex. The pseudo microprocessor then sequences through a short program illustrating fetch execute using zero page addressing.

Finally, a new method is introduced to illustrate the fetch execute sequence in computers. This is a shorter method using symbols to illustrate microcomputer operation.

------> data transfer
<-----> data exchange
( ) contents of a register
[ ] memory location address
([ ]) contents of a memory location address

Unit B, A Real Computer

Introduction to the AIM 65 microcomputer. In order for the student to develop skills and Knowledge about microcomputers, the following are required:

AIM 65 microcomputer with 4K memory

AIM 65 monitor program listing (Rockwell)

AIM 65 summary card (Rockwell)

AIM 65 User's guide (Rockwell)

R 6500 microcomputer system hardware manual (Rockwell)

R 6500 microcomputer system programming manual (Rockwell)

Microprocessor Systems Engineering (Matrix Publishers)

The AIM 65 is a complete general purpose microcomputer incorporating some of the latest interfacing techniques. The student is given a brief description of the AIM 65 and some possible applications are listed.

The power of the AIM 65 comes from its extensive instruction set (machine instructions) combined with an extensive set of addressing modes. A complete list of the 6502 machine instructions is included with a brief explanation of each. All thirteen addressing modes are listed and defined. This course will make application of only the first seven. The remaining six are available for those students who wish to utilize the full power of the computer.

In order to demonstrate how the AIM 65 works the student will write and execute a number of programs using machine instructions. These programs are designed to:

1. Give the student experience using the computer
2. Familiarize the student with the 6502 instruction set
3. Develop skill in converting instructions (given in mnemonic code) to opcode (machine code)
4. Gain experience in writing machine level programs
5. Develop skill using the extensive AIM 65 operating system
6. Develop skill in interfacing the AIM 65 with the real world
7. Gain an understanding of how the microcomputer works

A block diagram of the AIM 65 and the 6502 is included to help the student co-ordinate the operations he will be performing.

The learning activities have been designed so that new concepts are introduced sequentially. The student will write and execute programs and develop knowledge about the microcomputer. The majority of the activities simulate logic gates, that is the AIM 65 will be used to replace a number of the gates studied in the last chapter.

When the student completes this chapter he will be assigned two activities that interface the computer with

real world activities, they are:

1. Design a real time system that allows the AIM 65 to monitor 8 smoke detectors, 2 burgular alarms and 4 temperature regulators. The computer must be able to sound an alarm in case of fire, a different alarm in case of break in and switch off or on the furnace to control heat.

2. Write and execute a program that will control the bells at Colchester Regional Vocational School.

IV Interfacing

A computer performs essentially two functions: process data and input/output data. This chapter deals with input/output data.

Almost all microprocessors use the same busses for both memory and input/output (I/O) transfers. Two methods are used to distinguish memory data and address from I/O data and address. They are:

1. Isolated I/O used extensively by the Intel 8080 and the Zilog Z80 MPU.

2. Memory mapped I/O used by both the Motorola 6800 and the Rockwell 6502 MPU.

Nearly all activities in this chapter are designed around the 6502 and use memory mapped I/O.

Unit A, Serial Interface

Three activities utilizing three different methods illustrate how the AIM 65 can be interfaced with the real world. They are:

1. Serial Interface (TTL)
2. RS-232C
3. 20 milliamp current loop

Unit B, Parallel Interface

Because all microcomputer data and interface lines are parallel, the easiest way to interface the AIM 65 is to simply connect I/O data lines to external devices. Centronics printers, the largest selling in the world, have a standard parallel interface.

An expanded parallel interface has been accepted by the International Electrotechnical Commission (IEC). This interface bus, called the IEEE 488 bus, employs 16 lines, and allows up to 15 instruments on the same bus. The IEEE 488 is the standard interface bus for Hewlett Packard and Commodore Business Machines.

This unit has activities on both types of parallel interfaces.

Unit C, UART

The UART (Universal Asynchronous Receiver Transmitter) provides a simple hardware method for converting serial data to parallel or parallel data to serial.

Activities are provided for students to gain experience using UARTs.

Unit D, Analog-to-Digital and Digital-to-Analog

Analog to digital and digital to analog converters are used to interface the AIM 65 with the "real" world. Two exercises are provided to familiarize the student with the above.

Unit E, Software Interface

In order for data to be "sent" to an output port, a short program is required to "handle" the data and control handshake signals. This program is usually called a software interface. In this unit the student is introduced to the concept of a software interface.

Unit F, PIA and VIA

PIA (Peripheral Interface Adapter) and VIA (Versatile Interface Adapter) are large scale integrated circuits that provide a means for the MPU to communicate to the outside world. This unit introduces the PIA and VIA and describes some of their special features and includes timing diagrams.

V Troubleshooting Digital Circuits and Microcomputers

The purpose of this chapter is to introduce the student to different tools used in troubleshooting digital/computer circuits and present some general troubleshooting

techniques.

Unit A1-A7, Troubleshooting Tools

These units introduce the following tools and provide an application exercise for each:

1. Logic Probes
2. Logic Pulser
3. Current Tracer
4. Logic Comparator
5. Oscilloscope
6. Logic State Analyzer
7. Signature Analysis

Unit B, Troubleshooting Microprocessor Systems

This unit on troubleshooting techniques is presented with the permission of Hewlett-Packard.

VI Programming in Assembly Language

There are essentially three levels of languages that can be used to program a microcomputer. They are:

1. Machine Language
2. Assembly Language
3. High Level Language such as Basic or Fortran

Machine language programs are very efficient, however they are difficult to write and because they consist of only 1's and 0's they are prone to error. Assembly language programs allow the student to use the

MPU instruction set and enter programs using mnemonic code. These codes are converted to machine language by an assembler.

High level languages are easier to program but are less efficient (i.e. use of machine time) than assembly language programs.

This unit compares the instruction set of the 6800 CPU and the 6502 CPU. The 6502 is considered to be an update of the 6800. The students are given assignments in the workbook "Programming the 6800", completion of these assignments should develop skills in assembler programming in both the 6800 and the 6502.

Students who have completed all chapters of this course should feel quite comfortable working on many different microprocessor systems. Additional instruction should be provided on a full system consisting of:

1. VDU (Visual Display Unit)
2. MPU with 32 to 64 K RAM memory
3. Real World Interface
4. Dual Floppy Disc
5. Dot Matrix Printer, parallel interface
6. Character Printer, serial interface
7. Complete software package for programming in Assembler, Basic and Fortran.

This system could use either a 6800 MPU or Z80 MPU. It is important that the student have the opportunity to gain experience on a complete system.

Happy Computing!

VIII CONCLUSIONS/RECOMMENDATIONS

It is the job of the Vocational or Technical School to teach fundamental skills that will allow a graduate to gain employment and progress in the Digital/Computer field.

The Dacum chart for Computer Technology lists the skills required by a working computer technologist. The proposed course, Digital/Computer Technology, does not attempt to teach all the skills listed on the chart. Some of these skills could not be economically taught because training equipment costs would be too high. Other skills could be learned on the job after some practical experience. Finally no one technician could be expected to master all the skills listed in the chart because they are a composite of the skills of many people.

Two criteria will be used to review the proposed Digital/Computer Technology Course.

1. Does it meet the requirements of the Digital/Computer Industry as identified by the Dacum chart? These general areas of competence will be considered.

A. Communicate: In Regional Vocational Schools communication skills are taught as a separate course.

The teacher of the Communications course should stress writing technical reports, interpersonal skills and learning to work under pressure. Reading schematic diagrams is usually taught in Basic Electronics.

B. Use Test Equipment: This course assigns one chapter (Chapter V) to troubleshooting skills and the application of test equipment.

C. Equipment Repair: Soldering skills, component replacement, parts identification are skills normally taught in Basic Electronics. The mechanical skills, such as replacing gears, drive belts and motors are not covered by this course.

D. Install Equipment: Most of these skills require application on expensive equipment, therefore, they cannot be economically taught in a Vocational School. Cable routing, interconnecting of units, shielding and equipment run up are all skills that can be readily taught if the school has a small computer system. A system that could be used to teach these skills is listed at the end of this chapter.

E. Maintain Equipment: These on the job skills are not taught directly by this course. Fundamental skills are taught in running diagnostics and digital to analog and analog to digital interfacing. Course

materials to teach mechanical skills should be developed to supplement this course.

F. Analyze System Faults: The bulk of the course is devoted to developing fundamental skills in Digital/Computer Electronics so that the technician can analyze system faults.

G. Interfacing Communication Equipment: Chapter IV is devoted to developing interfacing skills.

H. Design Modify and Construct Interfaces and Equipment: Component selection, component identification, component layout, block diagrams, assembly and construction, reading schematics are all an integral part of the course and these skills will be learned by the student. Hardware and software design are not considered to be major goals of the course. Designing, modifying and constructing interfaces are considered to be important parts of this course.

I. Operate Computers for Testing Reliability: Two chapters are assigned to teaching computer programming. Chapter I is programming in Basic (high level language) and Chapter VI is Programming in Assembler Language (low level language). Both of these languages are required to operate and test computers.

2. Is it suitable for use by Vocational or Technical

Schools?

Yes, the Digital/Computer Technology course was designed to teach fundamental Digital/Computer skills to Electronic students in Regional Vocational Schools. The purpose of this program is to have the student develop sufficient skills to gain employment in the Digital/Computer industry.

The Educational goals of this course are:

1. To introduce to the student representative Transistor-Transistor-Logic (TTL) digital logic circuits with application.

2. To develop skill in programming a computer in Assembler and Basic computer languages.

3. To develop an understanding of how a computer works.

4. To develop skill in interfacing a computer with external devices.

5. To introduce tools used in troubleshooting Digital/Computer devices.

In order for this program to be successful the following equipment is required:

1. Digital Logic Trainer for each student, a typical trainer is the Hewlett Packard 5035A Logic Lab. A trainer could easily be built by the student and its

construction become an integral part of the training.

2. Rockwell AIM 65 computer. One computer should be available to each student. All the learning activities in Chapters III, IV and V are developed around the AIM 65. The cost of each computer is five hundred and forty dollars Canadian as of January 1980.

3. Logic probes and analyzers to complete the activities in Chapter V.

4. A complete microcomputer incorporating, but not limited to the following:

a. Visual Display Unit

b. CPU with 32-64 K RAM memory

c. Character or Dot Matrix serial Printer

d. Dual Floppy Disk

e. Software packages including Assembler and Basic

f. "Real World" Interface.

This computer can be used to teach many of the job skills listed in the Dacum chart.

This course should serve as a guide for teachers. The teacher may wish to substitute or ignore any of the activities.

It was to have been the Nuclear Age. It became the Computer Age. This is the title of a booklet published by IBM that traces the development of computers from 1951 to 1976.

At a recent international computer conference held at Wolfville, Nova Scotia, Commodore Grace Hopper, a Pentagon computer expert, predicted that computers will become the world's largest industry in the near future, even dwarfing North America's automobile industry.

The age of the computer. World's largest industry. Experts are making predictions about the rapid growth of computers in our society.

Someone will be required to install, test and maintain these computers. This will be the job of a trained technician with skills in the field of digital and computer electronics.

These technicians can and should be trained in Regional Vocational Schools in Nova Scotia. This writer has presented a course of study that will train electronic technicians in the fundamental Digital/Computer electronic skills. This course includes skill training similar to that in "Microprocessors", and "Practical Microprocessors". In addition the proposed course, Digital/Computer Technology, uses a "real" computer as a trainer so that skills such as interfacing and troubleshooting can be included in the program. Applications in the "real" world are an important part of this course.

RECOMMENDATIONS

During the past twenty years electronic technology has passed through four generations.

a) Vacuum tubes

b) Discrete transistors

c) Integrated Circuits

e) Large Scale Integrated Circuits (LSI) including Microprocessors

We are now in the fourth generation. Some of our Regional Vocatioal Schools may not be teaching skills in this area.

1. The writer recommends that electronics training in Regional Vocational Schools include in their curriculum comprehensive training in Digital/Computer skills using the course of study included in the Appendix of this thesis.

This program will train the youth of Nova Scotia for employment in the Digital/Computer industry.

2. Supplementary learning activities, designed to teach the necessary mechanical skills identified by the Dacum chart, should be developed by teachers using the Digital/Computer Technology course.

3. No evaluation is included in the Digital/Computer Technology course. Methods to evaluate the

activities suggested in this program should be developed by the subject teacher.

CONCLUSIONS

"Practical Microprocessors" by Hewlett-Packard contains a well designed course that would find application in a variety of training situations from technician to the engineering level. The primary disadvantage of selecting this course is the initial cost of the HP5036A Microprocessor Lab. A secondary consideration is insufficient interfacing with the "real" world.

"Microprocessors" by HeathKit Continuing Education is a reasonably priced program that teaches skills in programming and interfacing. Unfortunately, the companion computer trainer, ET 3400, contains too small a memory for commercial applications and the hexadecimal keypad is not designed with efficient programming in mind.

The writer has developed a course of study in Digital/Computer electronics using the AIM 65 as the microprocessor trainer. Because the AIM 65 is a "real" computer and not just a trainer, the knowledge gained by the student is limited only by his ingenuity and imagination. He will find myraid applications

interfacing printers, AC controllers, DC controllers and analog to digital devices to name but a few.

The key to the writer's program is the Rockwell AIM 65 Microcomputer.

REFERENCES

F O O T N O T E S

[1] Canadian Press, "Electronic and Computer Workers in Short Supply" (Halifax: Chronicle Herald Ltd., November 13, 1979), pg. 24.

[2] R. E. Adams, DACUM (Ottawa: Department of Regional Economic Expansion, 1975), pg. 23.

[3] Keith Reid-Green, "A Short History of Computing," Byte, (July 1978), pg. 86.

[4] Herman Goldstine, The Computer from Pascal to von Neumann (Princeton: Princeton University Press, 1972), pg. 21.

[5] International Business Machines, "The Computer Age," 1976), pg. 18.

[6] Adam Osborne, Running Wild The Next Industrial Revolution (Berkeley: McGraw-Hill Inc., 1979), pg. 25.

[7] Ibid., pg. 76.

[8] International Business Machines, op. cit., pg. 23.

[9] Osborne, op. cit., pg. 73.

[10] Ibid., pg. 76.

[11] "VISTA : Bell Canada's Interactive Communication System," Computing Canada, (November 15, 1979), pg. 10.

[12] Ibid.

[13] John Hewer, "Wp-the industry's $multimillion sleeper," Canadian Electronics Engineering, (December 1979), pg. 30.

[14] Robert Adams, "DACUM in Nova Scotia," Journal of Education, (Summer, 1975), pg. 15.

[15] Jack Sorenson, "A Summary of the Research" Entry Level Skills - Electronics", Canadian Vocational Journal, (February 29, 1976), pg. 32.

[16] Michael Slater and Barry Bronson, Practical Microprocessors (Santa Clara: Hewlett-Packard Company, 1979), pg. xxi.

[17] HeathKit Continuing Education (Manual), HeathKit Microprocessors, (Benton Harbour, Michigan: Heath Company 1977), pg. 1.

B I B L I O G R A P H Y

Adams, Robert. "Dacum in Nova Scotia," Journal of Education, (Summer, 1975), 14-18.

Adams, Robert. Dacum. Ottawa: Department of Regional Economic Expansion, 1975.

Ahl, David. 101 Basic Computer Games. Morristown, New Jersey: Creative Computing, 1973.

Best, John. Research in Education. Englewood Cliffs, New Jersey: Prentice Hall Inc., 1959.

Burstein, (ed.) Practice Problems in Number Systems, Logic, and Boolean Algebra. New York: Howard Sams and Co., 1975.

Camp, R.T. Smay, and C. Triska. Microprocessor Systems Engineering. Portland, Oregon: Matrix Publishers, 1979.

Chronicle-Herald, November 13, 1979. Canadian Press article concerning the shortage of skilled workers in electronic fields.

Coombs, Clyde. Printed Circuits Handbook. Toronto: McGraw-Hill Inc., 1967.

Diefenderfer, A. Basic Techiques in Electronic Instrumentation. Toronto: W.B. Saunders Company, 1972.

Dorf, Richard. Introduction to Computers and Computer Science. San Francisco: Boyd and Fraser Publishing Company, 1972.

Fairchild Industries. TTL Databook. Mountain View, California: Fairchild Camera and Instrument Corporation, 1978.

Gasperini, Richard. Digital Experiments. Los Altos, California: Movonics Company, 1976.

Gasperini, Richard. Digital Troubleshooting. Los Altos, California: Movonics Company, 1975.

Goldstine, Herman. The Computer From Pascal to Von Neumann. Princeton, New Jersey: Princeton University Press, 1972.

Guthikonda, V. Microprocessors and Microcomputer Systems. Toronto: Van Nostrand Reinhold Company, 1978.

HeathKit, Digital Techniques. Benton Harbour, Michigan: Heath Company, 1975.

Hewer, John. "Wp-the industry's $multi-million sleeper," Canadian Electronics Engineering, (December 1979).

HeathKit. Microprocessors. Benton Harbour, Michigan: Heath Company, 1977.

International Business Machines. The Computer Age. Armonk, New York: International Business Machines Corporation, 1976.

Larsen, David, and Zakas, Rodnay, The Bugbook I. Connecticut: E and L Instruments, 1974.

Larsen, David, and Rony, Peter. The Bugbook II. Connecticut: E and L Instruments, 1974.

Larsen, David, and Rony, Peter. The Bugbook IIa. Connecticut: E and L Instruments, 1975.

Larsen, David, and Rony, Peter. The Bugbook III. Connecticut: E and L Instruments, 1975.

Leach, Donald. Experiments in Digital Principles. New York: McGraw-Hill, 1976.

Lesea, Austin, and Zakas, Rodnay. Microprocessor Interfacing Techniques. Berkeley: Sybex, 1977.

Leventhal, Lance. Introduction to Microprocessors: Software, Hardware, Programming. Englewood Cliffs, New Jersey: Hayden Book Company, 1977.

Leventhal, Lance. The 6800 Microprocessor. Rochelle Park New Jersey: Hayden Book Company, 1977.

Leventhal, Lance. 6502 Assembly Language Programming. Berkeley: McGraw-Hill, 1979.

Malvino, Albert. Digital Computer Electronics. New York: McGraw-Hill, 1977.

Malvina, Albert, and Leach, Donald. Digital Principles and Applications. New York: McGraw-Hill, 1969.

Mason, Emanuel, and Bramble, William. Understand and Conducting Research. New York: McGraw-Hill, 1978.

Osborne, Adam. Running Wild-The Next Industrial Revolution. Berkeley, California: McGraw-Hill, 1979.

Osborne, Adam. 6800 Programming for Logic Design. Berkeley: Osborne and Associates, 1977.

People's Computer Company. What To Do After You Hit Return. Menlo Park, California: People's Computer Company, 1977.

Radio Shack. TRS-80 Micro Computer Technical Reference Handbook. Fort Worth, Texas: Radio Shack Inc., 1978.

Reid-Green, Keith. "A Short History of Computing," Byte, (July, 1978), 84-94.

Reid-Green, Keith. "History of Computers The IBM 650," Byte, (March, 1979), 238-240.

Reid-Green, Keith. "History of Computers The IBM 704," Byte, (January, 1979), 190-192.

Rockwell International. AIM 65 Monitor Program Listing. Anaheim, California: Rockwell International Corporation, 1979.

Rockwell International. R6500 Hardware Manual. Anaheim, California: Rockwell International Corporation, 1978.

Rockwell International. R6500 Programming Manual. Anaheim, California: Rockwell International Corporationn, 1979.

Scientific American. Microelectronics. Theme issue. New York: Scientific American Inc., September 1977.

Slater, Michael, and Bronson, Barry. Practical Microprocessors. Santa Clara, California: Hewlett-Packard Company, 1979.

Smith, Julius. Basic Mathematics with Electronics Applications. New York: The MacMillan Company, 1972.

Sorenson, Jack. "A Summary of the Research" Entry Level Skills - Electronics"," Canadian-Vocational Journal (February, 1976), 32-34.

Stow, Robert. AIM 65 User's Guide. Anaheim, California: Rockwell International Corporation, 1978.

Sybex. Microprocessor Lexicon. Berkeley, California: Sybex Inc., 1978.

"Vista: Bell Canada's Interactive Communication System," Computing Canada, (November 15, 1979,) 10-12.

Weller, W. Practical Microcomputer Programming: The M6800. Evanston, Illinois: Northern Technology Books, 1977.

Zaks, Rodnay. Programming the 6502. Berkeley: Sybex, 1978.

APPENDICES

APPENDIX A

DIGITAL/COMPUTER TECHNOLOGY

DIGITAL/COMPUTER TECHNOLOGY

The purpose of this course of study in Digital/Computer Technology is to enable the student to gain entry level skills and knowledge in the field of Digital/Computer electronics. Graduates of this program should find employment in computer electronics, industrial electronics or related fields.

The Educational objectives are:

1 To develop skill in programming a computer in Assembler and Basic.

2 To develop an understanding of how a computer works.

3 To develop skill in interfacing a computer with external devices.

4 To introduce tools used in troubleshooting Digital/Computer circuits.

The materials have been written with you, the student, in mind and the following features have been included:

1 As part of this course you will be introduced to an extensive glossary of computer related terms.

2 The bibliography contained in the main body of the thesis includes journals and articles as well as texts. This will be of assistance to those of you who wish to expand your knowledge about computers and computer related circuits.

3 The majority of the materials are designed to be interactive, that is you will read a short introduction and then be asked to perform an operation, i.e. a learning activity.

4 Several of the learning activities may seem incomplete, however, further investigation on your own should allow you to complete the exercises.

You will have your teacher and the following texts as resources.

Digital Troubleshooting by Richard E. Gasperni

Microprocessor Systems Engineering by Camp, Smay, and Triska

Introduction to Microprocessors: Software, Hardware, Programming by Lance A. Leventhal

TTL Data Book by Fairchild

Microprocessor Lexicon by Sybex

Complete set of AIM 65 reference books

Experiments in Digital Principles by Leach

Practice Problems in Number Systems, Logic, and Boolean Algebra

Programming the 6800 Microprocessor by Southern

CONTENTS

I Programming in Basic

These exercises are designed to teach you how to program a computer using the Basic language. In order to use the following exercises effectively you will require a computer (that contains a Basic interpreter) and a computer use manual.

For maximum benefit "write" all the programs included.

This part of the course can be started at any time, however it would be benefical for the student to complete the section while studying basic electronics. This would allow you to gain "hands on" experience with the microcomputer in a relatively non-threatening atmosphere.

In order to communicate with a computer we must be able to "speak" to it. Unfortunately, the computer cannot speak our language and it is very difficult for us to "speak" it's language.

The language of the computer is 1's and 0's. For example, in order for the computer to add two numbers (3+4) it would require the following information:

```
1000,0110
0000,0100
1100,0110
0000,0011
0001,1011
1011,0111
0001,0011
0001,0000
```

This is not a very good method for "programming" a computer, it is slow and provides lots of room for error. Another method for programming the computer to add two numbers is to use an assembler. The program is written in assembly language and then converted to machine language by the assembler.

The following is an example of how two numbers could be added:

```
LDAA  #$04
LDAB  #$03
ABA
STA   $0010
```

However, this is time consuming and somewhat difficult. A simpler method has been devised where we simply tell the computer to add two numbers and print the result. That is: PRINT 6+3.

This is accomplished by the use of a Basic Interpreter. The Basic Interpreter is a program inside the computer that lets us speak to the computer in an almost English-like language. That is, the Basic interpreter is similar to a language translator.

BASIC stands for Beginners All Purpose Symbolic Instruction Code.

In order to make use of Basic, we have to learn to "speak" Basic.

The following sections will introduce you to the Basic language and help you write programs.

BASIC CONCEPTS

STATEMENTS — A Basic program consists of one or more statements. "PRINT 27" is an example of a PRINT statement.

PRINT

LINE NUMBER — Each statement must be preceeded by a sequential line number. "10" is a line number.

END — The end of a Basic program must include the END statement.

The following is an example of a Basic program: (Type in this program)

READY (This is printed by the computer)

```
10 Print 27    (Return)
20 End         (Return)
```

The program is now in the computer to execute the program type:

RUN — RUN (Return)
27 should appear on the screen.

In order to see your program type in:

LIST — LIST (Return)

NEW — In order to delete an old program type NEW (R)
To see if it is deleted, type LIST (R)

You should now be able to use the following:

LINE NUMBERS	
PRINT	(statement)
END	(statement)
RUN	(command)
LIST	(command)
NEW	(command)

STRINGS

You can also print words (STRINGS) i.e. Print "My Name Is". Lets write a program that will print your name and your age.
(note quotation marks)

```
NEW                          (Return) or (R)
10 PRINT "MY NAME IS KAREN"      (R)
20 PRINT "MY AGE IS"             (R)
30 PRINT  14                     (R)
40 END
```

Now RUN and then LIST this program.

Write a program that will print your address
RUN and LIST

To expand your programming capabilities we will introduce to you three new concepts.

REMOVE A LINE

To remove a line type in the line number and press return.

OPERATORS

Operators - These are + for addition
- for subtraction
/ for division
* for multipication
↑ for exponents

Be careful the computer's order of operations will use ↑ before * and /, and * and / before + and -.

VARIABLES

This is a variable, A
This is a variable, B
This is a variable, A1

Variables are alpha numeric but the first character is always alphabetic. List 10 variables, remember the first character must be some letter between A and Z, the second character, if used, must be a numeral.

=

Example: A=3

This means the variable on the left takes on the value to the right.

Let's write a program using this new concept.

```
10 A=3        (R)
20 B=4        (R)
30 C=B+A      (R)
40 PRINT C    (R)
50 END        (R)

RUN           (R)
LIST          (R)
```

SUMMARY OF CONCEPTS LEARNED

STATEMENTS
LINE NUMBERS
PRINT
END
NEW
RUN
LIST
OPERATORS
VARIABLES
=

Write a program using variables and operators that will sum 25 and 50. Remember to include a Print statement so that you may see the result.

```
RUN   (R)
LIST  (R)
```

MORE NEW CONCEPTS

INPUT

Example: INPUT A
The input statement will ask for a value to be assigned to the variable A.

```
10 INPUT A    (R)
20 PRINT A    (R)
30 END        (R)
```

```
RUN  (R)
     ? (computer wants you to input a value)
     ? 27   (R)
     27     (computer prints out 27)
```

Try Again

```
     10 INPUT A,B,C
     20 D=A+B+C
     30 PRINT D
     40 END
```

```
RUN   (R)

     ? 10   (R)
     ?? 20  (R)  (computer keeps asking
     ??? 30 (R)   questions until it receives
                  all the inputs)
```

RELATIONAL OPERATORS

```
= equal
< less than
> greater than
<> not equal to
=> equal to or greater than
=< equal to or less than
```

IF.... THEN

IF THEN statements are used with relational operators and help the computer to make decisions.

```
IF A>B THEN PRINT "A IS LARGER"
IF A<B THEN PRINT "A IS SMALLER"
```

GOTO

GOTO statements are used to interrupt the normal flow of the program. 10 GOTO 30 means do not goto line 20 but goto line 30 skipping line 20.

Example

```
10 A=3
20 B=4
30 GOTO 50
40 PRINT A
50 PRINT B
60 END
```

This program will print only 4.
Be sure you understand the program logic.

SUMMARY OF CONCEPTS

INPUT
RELATIONAL OPERATORS
IF.,.. THEN
GOTO

Now let's use these four new concepts, Input, Relational Operators, If.... Then, and Goto to write a program.

```
10 INPUT A,B,C,D
20 IF A>B THEN 60 (means GOTO line 60)
30 IF C>D THEN 80
40 PRINT "B>A and D>C"
50 GOTO 90
60 PRINT "A>B"
70 GOTO 90
80 PRINT "C>D"
90 END
```

What would happen if you did not use the GOTO statement?

We are really moving along, now we can do arithmetic and make decisions.

What else can we do? How about writing a program that will find the average of all the marks in your class and print it out.

How do you find average? Think!

REMARK

REMARK (Rem) statement is used to insert useful information in the program. However, it is ignored during execution.

```
10 INPUT A,B,C,D
20 REM: THE NEXT STATEMENT WILL ADD THE
       , VARIABLES
30 E=A+B+C+D
```

PRINT "STRINGS"

The print statement can be used to print words (strings) as variables. The words must be enclosed in quotation marks.

```
PRINT  "WORDS",B
PRINT  "TEST", A
```

Example

```
10 INPUT A
20 PRINT "NUMBER OF CHAIRS =",A
30 END

RUN
? 24
NUMBER OF CHAIRS = 24
```

Try the same program using a ; before A. RUN Note the difference in spacing.

INPUT "STRINGS"

```
INPUT " WORDS",A
INPUT " NUMBER OF CHAIRS REQUIRED ",A
```

Example

```
10 INPUT "NUMBER OF CHAIRS REQUIRED =",A
20 PRINT "NUMBER OF CHAIRS ="; A
30 END

RUN

NUMBER OF CHAIRS REQUIRED = (you input 24)
NUMBER OF CHAIRS = 24
```

FOR.... NEXT

Sometimes it is necessary to repeat an operation over and over. This can be easily accomplished using a For Next loop.

Example

```
10 FOR I=1 TO 5
20 PRINT I
30 NEXT I
40 END
```

RUN

```
1
2
3
4
5
```

First the value I=1, then line 30 NEXT I causes the program to loop back to line 10 and I gets a new value i.e. I=2 etc.

STEP

Line 10 could have been written

FOR I=1 TO 5 STEP 1 This means that I increments by 1 each time it goes through the loop. Note if step is not used the increment is 1. In addition you may use:

FOR I=1 TO 6 STEP 2 Note I increments by 2 . ANY STEP MAY BE USED.

Write a program that uses a For Loop with a step other than 1

SUMMARY OF CONCEPTS COVERED

REMARK
PRINT "WORDS",A
INPUT "WORDS",A
FOR.... NEXT

MULTIPLE STATEMENTS

Multiple Statements may be used on one line with the use of a colon example:

```
10 A=27: B=33
```

SUBSCRIPTED VARIABLES

Examples: A(1), B(2), Z(14)

Sometimes it is convenient to use subscripted variables, especially when inputing numbers to print out as data.

DIMENSION

Subscripted variables greater than 10 must be dimensioned. (that is, space reserved in memory)

Example: DIM A(100), B(50), C(27)
O.K. Let's use the last three concepts to write a program.

FOR....NEXT
Subscripted Variables
DIMENSION

```
5 REM PROGRAM TO READ IN 25 VALUES AND
  PRINT THEM OUT
10 DIM A(25)
20 FOR I=1 TO 25
30 INPUT A(I): REM FIRST WILL BE A(1),
   THEN A(2), ETC.
40 NEXT I
50 FOR I=1 TO 25
60 PRINT A(I),
70 NEXT I
80 END
```

RUN

Not a very useful program. How about if we find the average of these values and print it out.

```
1 Z=0: REM INITIALIZE COUNTER
5 REM PROGRAM TO READ IN 25 VALUES AND
  PRINT OUT THEIR AVERAGE
10 DIM A(25): REM REQUIRED FOR SUBSCRIPTED
   VARIABLES
20 FOR I=1 TO 25
30 INPUT A(I)
40 Z=Z+A(I): REM KEEPS RUNNING TOTAL
50 NEXT I
60 X=Z/25
70 PRINT "AVERAGE OF ALL VALUES IS",X
80 END
```

Study this program very carefully as you input and run it. If you have any questions please ask.

Suppose this program was used with different populations i.e. 20, 40, 15.

Can you write a universial program that will let you find average scores of different populations.

Try a variable in place of 25.

DATA

READ

Sometimes it is convenient to include DATA in the basic program. The DATA can then be retrieved by a READ statement.

Example:

```
10 DATA 10, 20, 24, 64
20 READ A, B, C, D
30 E=A+B+C+D
40 PRINT "SUM=",E
50 END
```

RUN

Consider the following

```
10 Z=0
20 FOR I= 1 TO 5
30 READ A, B, C, D
40 Z(I)=A+B+C+D   :PRINT Z(I)
50 NEXT I
60 DATA 23, 16, 49, 73
70 DATA 68, 47, 77, 19
80 DATA 37, 29, 77, 76
90 DATA 88, 84, 44, 84
100 DATA 69, 79, 89, 99
110 END
```

Z(I) is the sum of each row.

READ and DATA statements are often used with statistics.

Write a program that uses the temperature recorded five times a month for 6 months. Find the average temperature for each month and then have the results printed out.

SUMMARY OF CONCEPTS

Subscripted Variables
DIMENSION
READ
DATA

STRING VARIABLES

A$ is a string variable
B$ is a string variable

String variables can equal words, or values, or both enclosed in quotation marks.

```
A$="YES"
B$="9 APPLES"
C$="9"
```

Example:

```
10 INPUT "WHAT IS YOUR NAME?",A$
20 PRINT "HOW ARE YOU ";A$
30 INPUT "DO YOU WANT TO CONTINUE?",B$
40 IF B$="YES" THEN 60
50 GOTO 999
60 PRINT A$;"HOW OLD ARE YOU?"
70 INPUT C$
80 INPUT "HOW OLD IS YOUR FRIEND?",D$
90 E$=C$+D$
100 PRINT "YOUR AGE AND YOUR FRIEND'S AGE
    =";E$
110 PRINT "DO YOU WISH TO CONTINUE";A$
120 INPUT B$
130 GOTO 40
999 END
```

BUILT IN FUNCTIONS

Basic contains a large number of built in functions.
Example: trig functions

```
10 A=COS (1):REM COMPUTERS USE RADIANS
20 PRINT A

10 B=SIN(.7)
20 PRINT B

10 A1=SQRT(16):REM SQUARE ROOT
20 PRINT A1
```

TAB FUNCTION

Tab lets you move the cursor to a pre-determined position before printing the character.

```
10 PRINT TAB (20);"*"
20 PRINT TAB (21);"+"
30 PRINT TAB (22);"[ + ]"
40 END
```

It should be clear that using the tab function you can draw rudimentary pictures. Try writing a program to draw a circle.

RND FUNCTION The RND function is used to generate a random number between 0 and 1.

Example:

```
10 A=RND(0)
20 PRINT A
```

RND is useful in games, Example:

```
10 A=RND (0)*5
20 B=RND (0)*5
30 IF A>B THEN 70
40 IF B>A THEN 90
50 PRINT "TIE"
60 GOTO 1000
70 PRINT "A>B"
80 GOTO 100
90 PRINT "B>A"
100 END
```

Program the computer to generate several random numbers then have them printed out.

INT FUNCTION The INT function removes the decimal portion of a number.

```
10 A=1.2345
20 B=INT (A)
30 PRINT B
40 END
```

Using the RND and INT functions write a program which has two players roll a die and prints out the winner and his roll.

If you are stuck go on to the next section.

GOSUB...RETURN When part of a program is used many times, usually that part of the program is set aside and called up when needed with a GOSUB statement.

Example:

```
10 REM DICE PROGRAM
20 REM FIRST PLAYER GETS DICE
30 GOSUB 100
40 P1 =A+B
50 REM SECOND PLAYER GETS DICE
55 GOSUB 100
60 P2=A+B
70 IF P1>P2 THEN 130
80 IF P2>P1 THEN 150
90 PRINT "TIE": GOTO 160
100 A=INT (RND(0)*6)+1
110 B=INT (RND(0)*6)+1
120 RETURN
130 PRINT "PLAYER 1 WINS"
```

```
140 GOTO 160
150 PRINT "PLAYER 2 WINS"
160 INPUT "PRESS Y AND RETURN TO
    CONTINUE",Z$
170 IF Z$="Y" THEN 30
180 PRINT "END OF GAME"
190 END
```

You are now ready to write many more, and larger programs.

DEF FN X (A,B) User defined functions are similar to Gosubs except they are on one line, and are unreferenced as to line number when used.

They accept input and return a specific value. The amount of input they can accept is determined by slots or the number of dummy arguments allowed.

```
10 REM USER DEFINED FUNCTION FOR PYTHEOGRAN
   THEOREM
20 DEF FN P(X,Y)= SQRT((X*X)+(Y*Y))
30 X=20
40 Y=30
50 Z=FN P(20,30)
60 PRINT "LONG SIDE OF RIGHT TRIANGLE =";Z
70 END
```

SUMMARY OF CONCEPTS

String Variables
Functions
TAB
RND
INT
GOSUB....RETURN
User Defined Functions DEF FN P(X,Y)

You now have enough command of the Basic vocabulary to implement the other charcteristics of Basic not already mentioned.

II DIGITAL LOGIC CIRCUITS

Logic circuits deal with discrete quantities or voltage levels. For Transistor Transistor Logic (TTL) a logic 1 is +5 volts and a logic 0 is 0 volts.

This chapter on logic circuits serves two purposes.

1. It provides a background in TTL logic circuits so that you can go on to study microprocessor circuits.

2. It is a foundation course in TTL digital logic.

The following is a suggested order of study for the activity units.

A. Semiconductor Review

B. Number Systems

C. Logic Levels

D. Introduction to the Logic Tester

E. Logic Gates

F. Boolean Algebra

G. Sequential Logic Circuits

H. Combinational Logic Circuits

A Semiconductor Review

1 A Programmed Review of Transistor Operation

2 The Bipolar Transistor as a Switch

3 Design of Transistor switching

A PROGRAMMED REVIEW OF TRANSISTOR OPERATION

Your understanding of logic circuits is dependent upon a knowledge of bipolar transistor operation. A knowledge of transistor action is a prerequisite to this program, but the following review is included to refresh your understanding of this important subject.

To use this program simply read the information in each numbered frame and answer the accompanying question by filling in the blank(s) or choosing the correct answer. Cover the frames below the one you are reading with a piece of paper so that you will not be tempted to look at the answers. As you complete each frame, slide the paper down to reveal the next frame in sequence. The correct answer to the question in the previous frame appears in parenthesis at the beginning. The lesson material then continues. For best results complete this entire section at one time rather than breaking it into several study periods.

1. A transistor is a three element semiconductor device used in electronic equipment for controlling a large current with a smaller current. Transistors are used primarily as amplifiers with gain but are also used as switches in digital logic circuits.

   Transistors are made of semiconductor materials such as silicon and germanium. These are materials whose resistance is somewhere between that of conductors and insulators.

   The resistance of silicon is
   a. greater than
   b. less than
   c. the same as

   the resistance of a good conductor such as copper.

2. (a. greater than) Since the conductivity is between that of good conductors and insulators, the resistance of a semiconductor like silicon is greater than that of copper but considerably less than the resistance of an insulator such as glass or ceramic.

There are two types of semiconductor material, P-type and N-type. A semiconductor such as silicon is combined with other materials to form these two different types. For example, certain impurities are added to pure silicon to form P-type and other inpurities are added to form the N-type. The resulting N-type material is one which has an excess of free electrons. In other words, the majority current carriers are electrons. In P-type material, the majority current carriers are holes. A hole is the absence of an electron in the atomic structure of P-type silicon material, and it acts like a positive charge. P-type semiconductor material has an excess of holes to support current flow.

Current flow is by _____ in N-type material and by _____ in P-type material.

3. (electrons, holes) Transistors and other semiconductor devices such as diodes and integrated circuits are made by combining P-type and N-type materials. For example, a diode is formed by joining P-type and N-type sections as shown in Figure 2-1A. The P-type section is designated as the anode, and the N-type section is designated as the cathode.

The PN junction thus formed has unilaterial characteristics. That is, current will flow through it in only one direction. It blocks current (electron) flow in the opposite direction. Figure 2-1B shows the symbol used to represent a PN junction diode in schematic diagrams.

A junction diode is sensitive to the _____ of current flow.

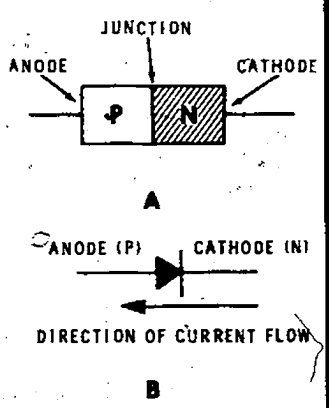


Figure 2-1
A PN junction diode
(A) and its schematic symbol (B)

4. (direction) If we apply a dc voltage to the junction diode, current may or may not flow through it depending upon the polarity of the voltage. This applied voltage is called bias. Figure 2-2 illustrates one way in which a junction diode can be biased. The series resistor R limits the current to a safe level.

In Figure 2-2, electrons flow out of the negative terminal of the battery into the N-type material. If the battery voltage is high enough to overcome an inherent potential barrier associated with the junction, the electrons will cross the junction and fill the holes. As the holes in the P-type material are filled, new holes are formed as electrons are pulled from the P-type material by the positive terminal of the battery. The result is a continuous current flow through the device. This arrangement is known as forward bias.

To bias a junction diode into conduction, the P-type element is connected to the _____ terminal of the battery and the N-type element is connected to the _____ terminal of the battery.

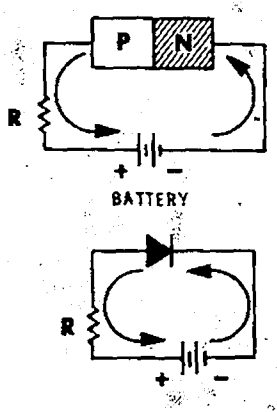


Figure 2-2
Forward biasing a PN junction diode so that it conducts

5. (positive, negative) To forward bias in a PN junction diode, the positive (P) battery terminal is connected to the P-type element and the negative (N) terminal of the battery is attached to the N-type element. The result is a continuous flow of current through the device that is effectively limited by the external circuit resistance. A voltage drop of approximately .7 volts occurs across a silicon diode. This drop is essentially constant regardless of the current value. The drop across a conducting germanium diode is about .3 volt.

How much current flows in the circuit of Figure 2-3?

I= _____ ma

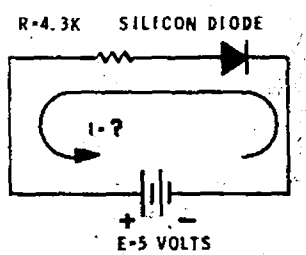


Figure 2-3
A forward biased diode.

6. (1 milliampere) In this circuit the diode is forward biased because the polarity of the applied voltage is correct. Therefore, current does flow. This current is limited by the resistance, but of course, is also a function of the battery voltage and the diode voltage drop. In this circuit, the diode drop is about .7 volts because the device is silicon. This means that the voltage drop across the resistor is (5 − .7) = 4.3 volts. The current (I) is then found by Ohm's law.

$$I = \frac{E}{R} = \frac{4.3}{4.3K} = \frac{4.3}{4300} = .001 \text{ amp} = 1 \text{ ma}$$

Current flows in a PN junction diode when it is _____

7. (forward biased) A forward biased diode conducts and acts as a very low resistance, permitting current to flow through it freely. If the polarity of the applied voltage is reversed as shown in Figure 2-4, the diode is said to be reverse biased.

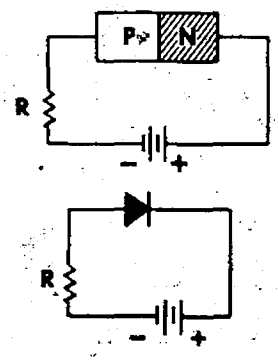


Figure 2-4 A reversed biased PN junction diode

With this arrangement, the electrons from the negative terminal of the battery fill the holes in the P-type material. The excess electrons in the N-type material are drawn away by the positive terminal of the battery. The effect is to draw the current carriers away from the junction so that no current flows. The diode acts as an effective open circuit. In a practical diode some leakage current does flow across the junction. But in a good silicon device this current is very low, in the microampere or nanoampere range, and for most applications can be considered to be negligible or zero.

To reverse bias a diode so that no current flows through it, the cathode (N) must be _____ with respect to the anode (P).

8. (positive) If the cathode is positive with respect to the anode, the diode is reverse biased and no current flows. To achieve this, the positive terminal of the battery is connected to the N-type cathode, and the negative terminal is connected to the P-type anode.

If the anode is made positive with respect to the cathode then current will flow. True or False? _____

9. (True) With the anode (P) positive with respect to the cathode (N), the diode is forward biased so current does flow. As you can see, the diode is polarity sensitive and that current does indeed flow through the device in only one direction, from cathode to anode.

Transistors are simply an extension of the junction diode concept. Transistors are formed by combining the P- and N-type material to form two junctions. This is done with three semiconductor elements. Figure 2-5 shows the two types of transistors.

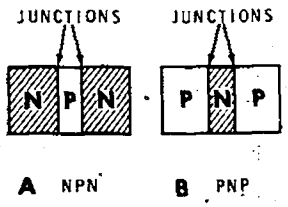


Figure 2-5
Types of junction transistors

The device in Figure 2-5A is an NPN transistor and the device in Figure 2-5B is a PNP transistor. Note the two arrangements of alternate P and N type materials.

The symbols used to represent these two types of transistors are shown in Figure 2-6 below.

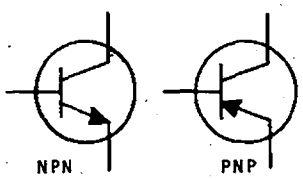


Figure 2-6
Transistor Symbols

The direction of the arrow is the distinguishing feature.

A transistor has (how many?) _____ PN junctions.

10. (two) Both types of transistors have two PN junctions. Each junction behaves exactly like the PN junction diode discussed earlier.

The three elements of each transistor are given specific names as indicated in Figure 2-7.

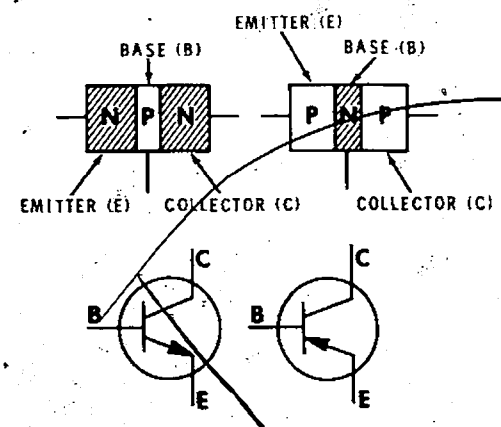


Figure 2-7
Identifying the three elements of a transistor

In a transistor, current flows through the device from the emitter through the base to the collector (holes in a PNP transistor and electrons in an NPN transistor). The presence or magnitude of this emitter-collector current is dependent upon the existence or magnitude of the base current.

The control element in a transistor is the _____

11. (base) Yes, the base is the control element. It effectively determines the magnitude (or presence) of any emitter-collector current.

In order for a transistor to function properly, the emitter-base (E-B) and base-collector (B-C) junctions must be properly biased. Proper bias to cause a transistor to conduct occurs when the E-B junction is forward biased and the B-C junction is reverse biased.

Is the PNP transistor shown in Figure 2-8 properly biased for conduction? Yes or No? ____

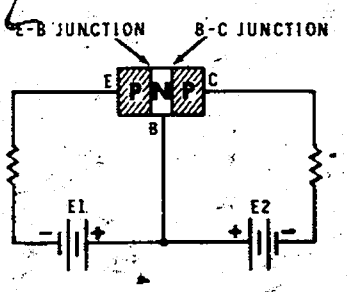


Figure 2-8

12. (No) The B-C junction is OK since it is reverse biased (+ to N and − to P), but the E-B junction is reverse, not forward, biased. The polarity of battery E_1 must be reversed.

Using the circuit configuration shown in the previous frame, draw the proper biasing for an NPN transistor.

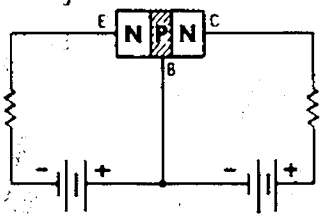


Figure 2-9
A properly biased NPN transistor

13. (Refer to Figure 2-9). The E-B junction is forward biased (+ to P and − to N) and the B-C junction is reverse biased (+ to N and − to P).

In a practical transistor circuit you can measure the voltages at each transistor element and noting the magnitudes and polarities, you can determine if the transistor is conducting or cut off.

Using the knowledge you've obtained to this point, determine the condition of the transistor in Figure 2-10.

This a. PNP transistor c. is conducting.
b. NPN d. is not conducting.

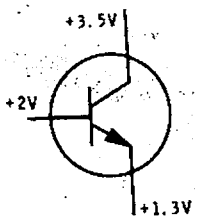


Figure 2-10

14. (b. NPN, c. is) The transistor is conducting. The base is more positive than the emitter so the E-B junction is forward biased. Note the difference of potential across the conducting E-B junction is .7 volts, the forward voltage drop of a silicon diode. Most modern transistors (diodes and integrated circuits) are silicon devices.

The base is less positive or more negative than the collector by (3.5 −2) = 1.5 volts so this junction is reverse biased. Therefore the transistor is conducting.

Is the PNP transistor in Figure 2-11 conducting or non-conducting? _____

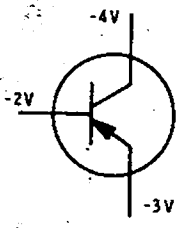


Figure 2-11

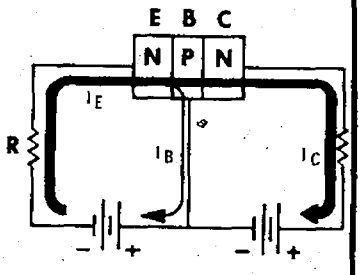


Figure 2-12 Current flow in a properly biased NPN transistor

15. (non-conducting) Both the E-B and B-C junctions are reverse biased so current does not flow from emitter to collector.

The actual path for current (electron) flow in a properly biased NPN transistor is shown in Figure 2-12.

A large current (I_E) flows into and through the emitter, through the base to the collector. Note that a small amount of emitter current divides off and flows out of the base. This is the E-B junction forward bias current or the base current I_B. Its magnitude is usually considerably less than that of the emitter current. The remaining current (I_C) flows out of the collector.

Considering the current relationship in Figure 2-12, how do you think the current flowing out of the collector compares to the current entering the emitter? The collector current is

a. equal to
b. less than
c. greater than

the emitter current.

16. (b. less than) The collector current (I_C) in reality is very nearly equal to the emitter current (I_E) but is less than the emitter current by an amount equal to the base current (I_B). The exact relationship is as expressed below.

$$I_C = I_E - I_B$$

You would expect current to flow in the E-B circuit because this junction is forward biased. But you would not normally expect current to flow in the collector because the B-C junction is reverse biased. The electrons flowing in the emitter enter the base. Here some of the electrons combine with holes in the P-type base and create the current flow out of the base. However, most of the electrons pass on through the base and into the collector. The reason for this is that the base is extremely thin and has only a minimum of available carriers to support current flow. The electrons passing through the base are then attracted by the positive charge on the collector. The collector current is

a. much higher than
b. much lower than
c. about the same as

the emitter current.

17. (c. about the same as) Most of the electrons in the emitter pass through the thin base into the collector and become collector current. A few electrons do combine with holes to produce a small base current.

The current flow in a properly biased PNP transistor is as shown in Figure 2-13. It is similar but not exactly like that in an NPN transistor.

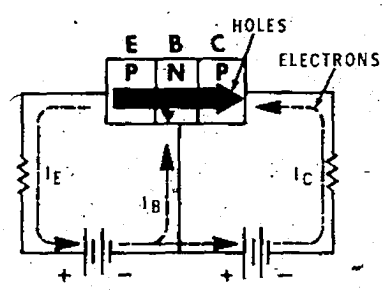


Figure 2-13
Electron and hole flow in a properly biased PNP transistor

The current carriers in a PNP transistor are holes rather than electrons. Internally the holes flow from positive to negative. External to the transistor the current is electron flow as indicated by the dashed lines. The internal hole currents have the same relationship as electron flow in the NPN device.

$$I_C = I_E - I_B$$

The electron flow external to the transistor is perhaps more clearly expressed as

$$I_E = I_C + I_B$$

Of course these two expressions are mathematically identical since one can be derived from the other by simple algebraic manipulation.

If the emitter current is 4 ma and the collector current is 3.85 ma, what is the base current? I_B = _____

18. (.15 ma or 150μA) The base current is the difference between the emitter and collector currents or

$$I_B = I_E - I_C$$
$$I_B = 4 - 3.85 = .15 \text{ ma}$$

The collector current is less than the emitter current by the amount of the base current.

The ratio of the collector to emitter current is approximately one because in most cases the collector current is very nearly equal to the emitter current. This ratio is called the forward current gain (α or alpha).

$$\alpha = \frac{I_C}{I_E} \approx 1 \text{ since } I_C \approx I_E$$

($\approx$means approximately equal to)

Practical values of alpha run in the .95 to .99 range. The higher the gain the better the transistor.

Using the values in the previous example ($I_E = 4$ ma, $I_C = 3.85$ ma) what is the current gain alpha? _____

19.

$$\left(\alpha = \frac{I_C}{I_E} = \frac{3.85}{4} = .9625\right)$$

While alpha is always less than one, we still refer to this current ratio as a gain.

Figure 2-14 below shows another way of connecting the bias to a transistor.

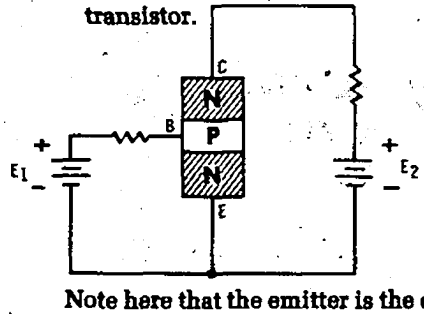


Figure 2-14
Biasing an NPN transistor with a common emitter connection

Note here that the emitter is the common element for the supply voltages rather than the base in the previous examples.

Will this transistor conduct? _____

20. (Yes) The transistor will conduct. Figure 2-15 shows the current paths.

The transistor conducts because the E-B junction is forward biased and the B-C junction is reverse biased. This reverse bias condition can be more readily seen if you consider the voltage on the base. With the E-B junction forward biased the base is .7 volts more positive than the emitter. The collector is more positive than the base with respect to the emitter because E_2 is usually much greater than .7 volts. For this reason the base is less positive or more negative than the collector, thus the reverse bias.

In Figure 2-15 what is the relationship between the various currents flowing?

a. $I_E = I_C + I_B$
b. $I_C = I_E + I_B$
c. $I_C = I_B + I_E$

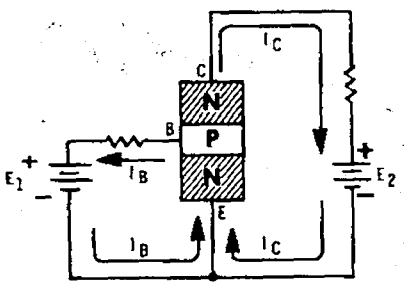


Figure 2-15
Current flow in
a common emitter biasing circuit

21. (a. $I_E = I_C + I_B$) The base and collector currents combine at the emitter to form the emitter current. The relationship expressed below

$$I_E = I_C + I_B$$

holds true for *any* transistor in *any* bias circuit configuration.

Since both bias voltages E_1 and E_2 are positive with respect to the emitter as shown in Figure 2-15 then they can be replaced by a single supply battery as shown in Figure 2-16. The result is proper bias for conduction at a considerable savings in the power supply.

The values of R_B and R_C are adjusted to provide the desired current levels. The bias voltage is labeled V_{CC} and is called the collector supply.

In Figure 2-16
a. $I_B > I_C$
b. $I_B = I_C$
c. $I_B < I_C$

Note: > means greater than
< means less than

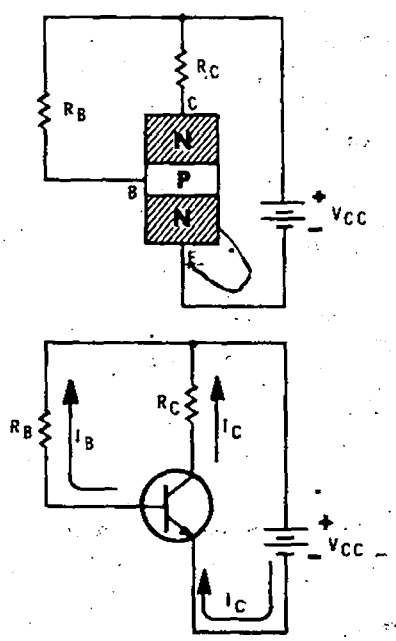


Figure 2-16 Simple voltage biasing of an NPN transistor

22. (c. $I_B < I_C$) The base current is always less than the collector current. But they are related as you learned earlier.

$$I_E = I_B + I_C$$

The ratio of the collector current to base current is another way of defining the gain of a transistor.

This is known as the dc forward current gain designated as β (beta) or h_{FE}.

$$h_{FE} = \beta = I_C/I_B$$

The higher this ratio, the higher the gain.

If I_C = 3.85 ma and I_B = .15 ma the gain is _____.

23. $$\left(h_{FE} = \frac{I_C}{I_B} = \frac{3.85}{.15} = 25.67\right)$$

This current gain figure actually tells us how much control the base current has over the collector current. Remember that if no base current flows due to a lack of forward bias on the E-B junction, then no collector or emitter current flows. It is also true that the amount of collector current flowing depends upon the amount of base current. The collector current is directly proportional to the base current. The I_C/I_B ratio is essentially constant for a given transistor so increasing I_B increases I_C by a factor equal to h_{FE}.

If I_C is 4 ma, h_{FE} = 20 I_B = _____ ma.

24. (.2 ma or 200 μA) Since $h_{FE} = I_C/I_B$ then $I_B = I_C/h_{FE}$ so I_B = 4/20 = .2

If we decrease I_B by .05 ma the new I_C will be _____ ma.

25. (3 ma) $I_C = I_B\ h_{FE} = (.2 - .05)\ 20 = 3$ ma.

Note that we decreased I_B by .05 ma while I_C decreased by 1.0 ma, a 20 to 1 ratio (h_{FE}). Therefore you can see that the smaller base current can control the larger collector current.

As you change the base current to control the collector current the transistor acts essentially as a variable resistor. A high collector current means a low emitter to collector resistance and a low collector current represents a high emitter to collector resistance.

An increase in base current causes the emitter-collector resistance to _____. (increase/decrease)

26. (decrease) Increasing I_B increases I_C so that the transistor conducts more and appears as a lower resistance.

In amplifier applications a small signal such as a sine wave varies the base current to produce a larger collector current variation of the same shape.

The transistor can also be used as an on-off switch. If no base current is applied, no collector current flows so the transistor is cut off. It acts as an open switch. If a high base current is applied, the transistor conducts and acts like a very low resistance. The transistor appears as a closed switch. In this program on digital techniques, the transistor will be considered a switch.

Self Test Review

1. Current flow in N-type semiconductor material is by
   a. holes
   b. electrons
   c. positive ions
   d. negative ions

2. Current (electron) flow in a PN junction diode is from
   a. P to N
   b. N to P
   c. either a. or b.

3. To cause a current to flow in a PN junction it must be
   a. forward biased
   b. reverse biased
   c. connected to a source of ac
   d. subjected to an electric field

4. Current will flow in a PN junction diode if
   a. P is −, N is +
   b. the cathode is positive with respect to the anode.
   c. the cathode is negative with respect to the anode.
   d. P is +, N is −

5. Majority carrier flow through a transistor is from _____ through the _____ to the _____.

6. A conducting NPN transistor has which of the following bias conditions?
   a. base positive with respect to emitter and collector negative with respect to base.
   b. base negative with respect to emitter and collector negative with respect to collector.
   c. base negative with respect to emitter and collector positive with respect to collector.
   d. base positive with respect to emitter and collector positive with respect to base.

7. The gain of the common emitter transistor circuit is
   a. I_E/I_B
   b. I_C/I_E
   c. I_C/I_B
   d. I_E/I_C
8. Which expression below accurately describes the relationship between the various transistor currents?
   a. $I_C = I_E + I_B$
   b. $I_C = I_E - I_B$
   c. $I_E = I_C - I_B$
   d. $I_B = I_C + I_E$
9. The collector current is controlled by varying the _____ current.
10. The emitter-collector resistance
    a. increases
    b. decreases

    when the collector current decreases.

Answers

1. (b) electrons
2. (b) N to P or cathode to anode
3. (a) forward biased
4. (c) cathode is negative with respect to the anode and (d) P is + and N is −
5. emitter, base, collector
6. (d) base positive with respect to emitter and collector positive with respect to base
7. (c) $I_C/I_B = h_{FE}$
8. (b) $I_C = I_E - I_B$
9. base
10. (a) increases

Learning Activity A2

The Bipolar Transistor as a Switch

Objective: To be able to relate transistor action to logic levels.

There are two basic types of transistor switches used in the implementation of digital integrated circuits, the bipolar transistor and the metal oxide semiconductor field effect transistor (MOS-FET).

In digital applications the transistor operates as an off/on switch. When the transistor is conducting full on (saturated) it operates like a closed switch. When the transistor is cut off it operates like an open switch.

A cutoff transistor is equivalent to a logic 0 and a saturated transistor is equivalent to a logic 1.

Learning Activity A3

Design of a Switching Transistor

Objective: To design a transistor driver for a LED.

Use the following first approximations:

a, Vce = 0 when the transistor is on

b, Vce = Vcc when the transistor is off

c, Collector to base leakage ignored

Design Considerations.

1 Define the load, usually voltage and current.

2 Specify supply voltage.

3 Select transistor maximum Ic, must be 2xIc required and voltage breakdown must be 2xIc.

4 Determine series dropping resistor.

$$Rc = \frac{Vcc - VL}{Ic}$$

5 Calculate Ib

$$hfc = \frac{Ic}{Ib} \qquad Ib = \frac{Ic}{hfe}$$

to insure saturation half hfe

$$Ib = \frac{Ic}{hfe/2} = \frac{2Ic}{hfe}$$

6 Calculate Rb

$$Rb = \frac{Vi - Vbe}{Ib} = \frac{Vi - .7}{Ib}$$

Application

Design a transistor driver for a LED

No. 1 Ic = 20 milliampre
VL = 1.7 Volts

No. 2 Vcc = + 5

No. 3 Use transistor 2N3904

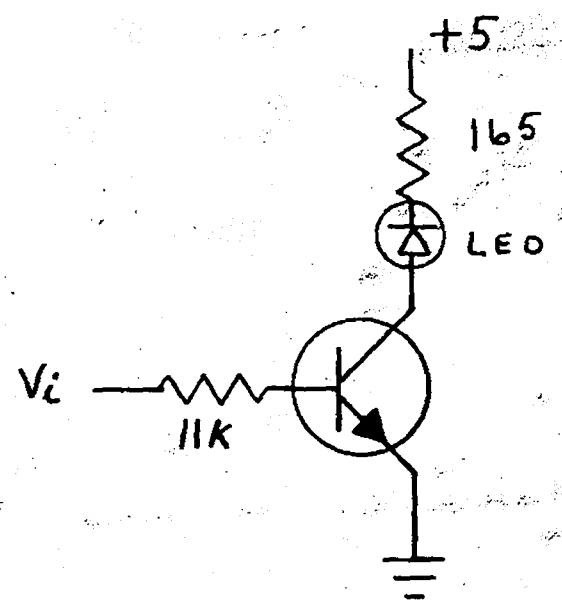
No. 4 $Rc = \frac{Vcc - VL}{Ic} = \frac{5 - .7}{20E-3} = 165$ ohms

No. 5 $Ib = \frac{2xhfe}{100} = \frac{2(.02)}{100} = \frac{.04}{100} = .4$ milliamps

No. 6 Vi = 5 Volts

$Rb = \frac{5 - .7}{.4E-3} = \frac{4.3}{.4E-3} = 11E+3$ ohms

Typical circuit



B Number Systems

B1 Decimal System

B2 Binary Numbers

B3 Octal Numbers

B4 Hexadecimal

B5 Binary Addition

B6 Binary Subtraction

B7 Multiplication of Binary Numbers

B8 Division of Binary Numbers

B9 Binary Coded Decimal (BCD)

B10 Adding Octal Numbers - Adding Hexadecimal Numbers

Learning Activity B1

Decimal System

The number system most familiar to us is the decimal system, in which the characters have ten possible states.

0		= 0
0	+ 1	= 1
1	+ 1	= 2
2	+ 1	= 3
3	+ 1	= 4
4	+ 1	= 5
5	+ 1	= 6
6	+ 1	= 7
7	+ 1	= 8
8	+ 1	= 9
9	+ 1	= 10
10	+ 1	= 11
	"	
	"	
	"	
98	+ 1	= 99
99	+ 1	= 100
100	+ 1	= 101

The decimal system has the concept of place value.

That is; $71 = 7 \times 10^1 + 1 \times 10^0$
$= 70 + 1 = 71$

Depending upon the position of the numeral, with respect to the decimal point, that numeral is multiplied by some power of 10.

Another example:

$728 = 7 \times 10^2 + 2 \times 10^1 + 8 \times 10^0$
$= 700 + 20 + 8 = 728$

Learning Activity B2

Binary Numbers

Binary number system is similar to the decimal system except that there are only two possible states, 1 and 0.

Decimal		Binary		
0	=	0		= 0
1	=	0	+ 1	= 1
2	=	1	+ 1	= 10
3	=	10	+ 1	= 11
4	=	11	+ 1	= 100
5	=	100	+ 1	= 101
6	=	101	+ 1	= 110
7	=	110	+ 1	= 111
8	=	111	+ 1	= 1000
9	=	1000	+ 1	= 1001
10	=	1001	+ 1	= 1010

Converting from Binary to Decimal is similar to place value in the Decimal system, i.e.

$$\langle 1010 \rangle_2 = \langle 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 \rangle_{10}$$
$$= 8 + 0 + 2 + 0 = \langle 10 \rangle_{10}$$

To develop skill in this area, convert the following Binary numbers to decimal:

101 = ...

1011 = ...

1100 = ...

1111 = ...

10,000 = ...

10,110 = ...

Can you think of a way to convert decimal numbers to binary numbers?

See Introduction to Microprocessors: Software, Hardware, Programming by Lance A. Leventhal, P.503-504.

Learning Activity B3

Octal Numbers

The octal number system is similar to the decimal number system except the characters have only eight possible states.

Decimal	Octal	Binary
0	0 = 0	0000
1	0 + 1 = 1	0001
2	1 + 1 = 2	0010
3	2 + 1 = 3	0011
4	3 + 1 = 4	0100
5	4 + 1 = 5	0101
6	5 + 1 = 6	0110
7	6 + 1 = 7	0111
8	7 + 1 = 10	1000
9	10 + 1 = 11	1001
10	11 + 1 = 12	1010
11	12 + 1 = 13	1011

Converting from octal to decimal is similar to the place value system used in decimal:

$$13_8 = (1\times8^1 + 3\times8^0)_{10}$$

$$= 8 + 3 = 11_{10}$$

Practice converting the following octal numbers to decimal:

11 = ...
16 = ...
17 = ...
20 = ...
117 = ...

Can you think of a way to convert decimal numbers to octal?

The following method can be used to convert from octal to binary.

$$(1\ ,\ 7)_8 = (001,111)_2$$

Separate the numerals and supply the correct binary number for each numeral. Note use 3 binary bits for each octal numeral.

Example $(2,0)_8 = (010,000)_2$

Convert the following octal numbers to binary.

11	16 .....
17	24 .....
117	377 .....

Learning Activity B4

Hexadecimal

The Hexadecimal (Hex for short) number system is similar to the Decimal number system except the characters have 16 possible states.

Decimal	Hexadecimal	Binary
0	0 = 0	0000
1	0 + 1 = 1	0001
2	1 + 1 = 2	0010
3	2 + 1 = 3	0011
4	3 + 1 = 4	0100
5	4 + 1 = 5	0101
6	5 + 1 = 6	0110
7	6 + 1 = 7	0111
8	7 + 1 = 8	1000
9	8 + 1 = 9	1001
10	9 + 1 = A	1010
11	A + 1 = B	1011
12	B + 1 = C	1100
13	C + 1 = D	1101
14	D + 1 = E	1110
15	E + 1 = F	1111
16	F + 1 = 10	10000
17	10 + 1 = 11	10001

Converting from Hexadecimal to Decimal is similar to the decimal place value system.

Example:

$$11_{16} = (1 \times 16^{1} + 1 \times 16^{0})_{10}$$

$$= 16 + 1 = 17_{10}$$

To improve your skill in converting from Hex to Decimal, convert the following Hex numbers to decimal.

12 = ...

19 = ...

1A = ...

1B = ...

1F = ...

Can you think of a way to convert Decimal numbers to Hexadecimal?

The following method can be used to convert from Hex to Binary:

$$1B_{16} = (1,B)_{16} = (0001,1011)_2$$

Separate the Hex numerals and supply the correct binary number for each numeral. Note: use 4 binary bits.

Example: $20_{16} = (0010,0000)_2$

Convert the following Hex numbers to Binary:

12 .....

19 .....

1B .....

1F .....

2E .....

2F .....

AF .....

A6 .....

FF .....

Learning Activity B5

Binary Addition

Rules: 0 + 0 = 0
0 + 1 = 1
1 + 1 = 10

Example:

```
  101 = 5
+ 011 = 3
---------
        8/10
```

```
 11  carry
 101
 011
----
1000 2 = 8 /10
```

Add the following and check your answer by converting to decimal.

```
  1011     1001     0001     0110,0111
+ 0110   + 1011   + 0011   + 1100,1001
------   ------   ------   -----------
```

Learning Activity B6

Binary Subtraction

Rules: 0 - 0 = 0
1 - 0 = 1
1 - 1 = 0
0 - 1 = 1 with a Borrow

Example: A = 1011 A - B 1011
B = 0110 - 0110

```
A     1 0 1 1
B    -0 1 1 0
         ------
C    =0 1 0 1
```

Check by adding , i.e. B + C = A

```
B     0 1 1 0
C   + 0 1 0 1
         ------
A   = 1 0 1 1
```

Possibly a better method would be to guess the value of C that must be added to B to equal A.

This method should be demonstrated by the teacher.

```
A    1 0 1 1
B  - 0 1 1 0
         ------
C  =
```

The following method must be demonstrated. It involves finding the 2's complement of the subtrahend and adding it to the minuend.

A = 1 0 1 1 A - B = ...
B = 0 1 1 0

```
  1 0 1 1    implies     1 0 1 1
- 0 1 1 0               + 1 0 1 0
                        ---------
                       (1) 0 1 0 1
```

Drop Extra Bit

How did I get the answer on the right?

Step I
Find the 1's complement of B (0110) 1001
Step II + 1
Add + 1 to form the 2's complement of B 1010
Step III
Add 2's complement of B to A

```
                          A = 1011
two's complement of B       = 1010
                            -----
                           (1)0101
```

Drop Overflow

Perform the following subtraction by forming the 2's complement of the subtraction and adding.

```
 0110    10100110    00111100    1011
-0011   -01101100   -00011010   -1010
-----   ---------   ---------   -----
```

Convert the following decimal numbers to binary and subtract.

```
 16     22     7
-12    -11    -6
```

Why use this method to subtract?

These operations can be performed quite easily by digital circuits. Also computers like to do things such as complementing and adding.

Learning Activity B7

Multiplication of Binary Numbers

This can be accomplished by repeated addition.

```
Example:    1011          1011
          x 011         + 1011
                        ------
                         10110
                        + 1011
                        ------
                        100001
```

Another method:
Shifting a number to the left is the same as multiplying by 2.

```
 A    1011
    x  010
    ------
     10110
```

Note: A is shifted one place to the left to get result. Check result by converting to decimal.

To multply by 4, shift the number left ..... places.

This interesting concept will be explored with the computer under programming.

Learning Activity B8

Division of Binary Numbers

This can be accomplished by repeated subtraction.
Method not shown:

```
  1100  =  0100
 ------
    11
```

Another method:

Shift the dividend right one place for each divide by 2 to find the quotient.

```
Example:  1100  =  (Shift right 2 places)  11
          ----
           100
```

This is a very interesting concept and will be explored in the programming section.

Learning Activity B9

Binary Coded Decimal (BCD)

In BCD four bits are used to represent each number between 0 and 9.

Example:

Decimal	BCD
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

Numbers between 9 and 99 are represented in the following manner.

09	0000,1001
10	0001,0000
11	0001,0001
69	0110,1001
99	1001,1001

Convert the following decimal numbers to BCD using eight bits.

5 .....

33 .....

77 .....

16 .....

ASCII (American Standard Code for Information Interchange) numbers can easily be converted to BCD, can you see how this can be done?

Learning Activity #10

Adding Octal Numbers

Probably the best way to add octal numbers is to use a number line.

Example: 5 + 6 =13

number line 0 1 2 3 4 5 6 7 10 11 12 13 14 15 16

Example subtraction 5 - 6 = -1

number line -3 -2 -1 0 1 2 3 4 5 6 7 8 9

Another example 2 + 13 = 15

number line 0 1 2 3 4 5 6 7 10 11 12 13 14 15 16 17 18

More practice may be required, add the following octal numbers:

13 + 10 = ...

16 + 06 = ...

67 + 06 = ...

Adding Hexadecimal Numbers

The same system that was used to add octal numbers can be applied to hexadecimal numbers.

Example: 8 + 6 = E

number line 0 1 2 3 4 5 6 7 8 9 A B C D E F 10 11 12 13

Use the Hexadecimal number line to add the following Hexadecimal numbers.

07 + 0A = ...

0F + 0A = ...

1B + 07 = ...

2A + 0E = ...

Answer all the above correctly and you get a star.

C Clock Pulses and Integrated Circuits

1 Clock Pulses

2 Application of clock pulses

3 Construction of an integrated circuit

4 Development of integrated circuits

5 Characteristics of integrated circuit

Learning Activity C1

Clock Pulses

Objective: To introduce the concept of clock pulses.

Clock: Reference timing source in a system, typically a microprocessor. A clock provides regular pulses that trigger or synchronize events.

A clock pulse can be described as a transition from a logic 0 to a logic 1 and back to a logic 0. In TTL positive logic a logic 1 = +5 volts, and a logic 0 = 0 volts, see figure 1. A logic 1 is also called a Hi and a logic 0 a LO.

The concept of a clock pulse with leading and trailing edge must be thoroughly understood and committed to memory. An illustration is presented in figure 1.

TTL Positive Logic

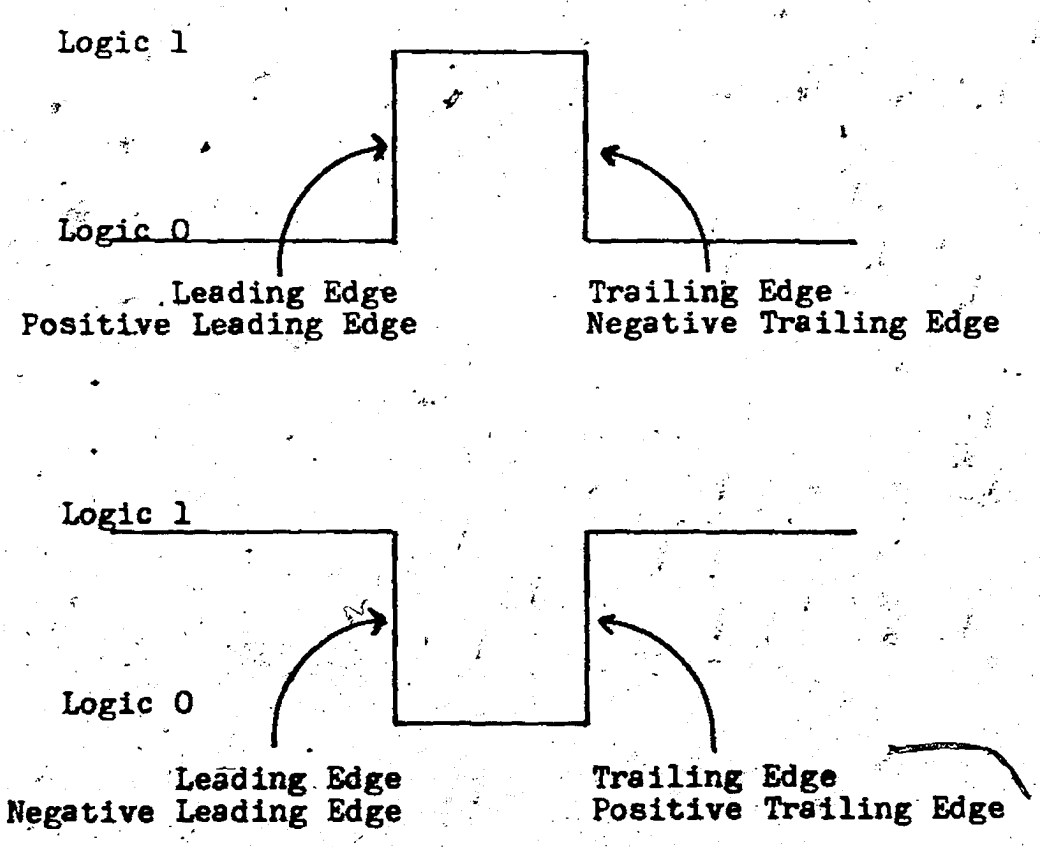


Figure 1

Figure 2 is an illustration showing typical clock pulses entering and leaving a TTL logic circuit. The important characteristics of these circuits are:

t(r) - Rise time, the time it takes the pulse to rise from 10 percent to 90 percent of it's maximum value.

t(f) - Fall time, the time it takes the pulse to fall from 90 percent to 10 percent of it's maximum value.

Propagation Delay: The time delay between input transition and output transition.

t(phl) - Propagation delay occuring while the output changes from high to low, usually measured at the 50 percent level.

t(plh) - Propagation delay occuring when the output changes from low to high, usually measured at the 50 percent level.

t(w) - Pulse width, usually measured between the 90 percent levels of the pulse.

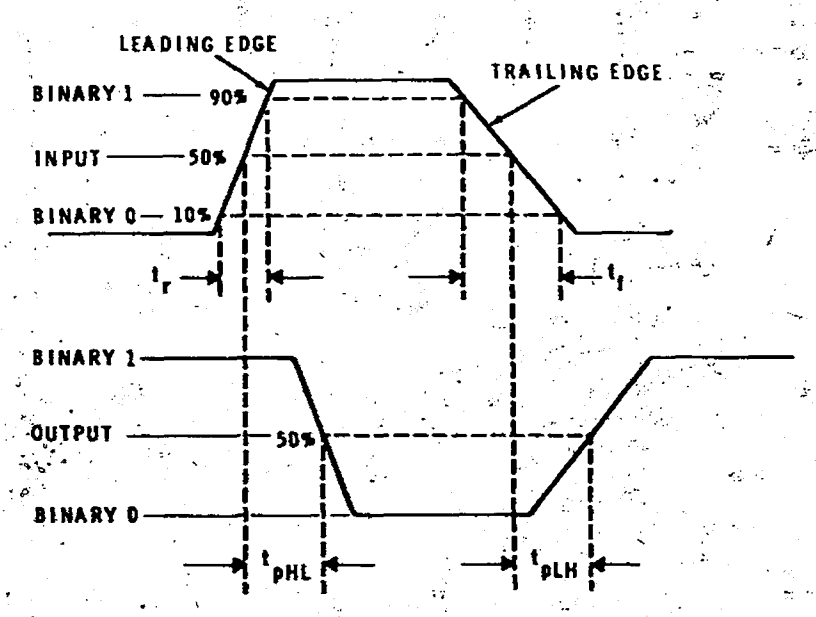


Figure 2

Learning Activity C2

Application of Clock Pulses

Objectives: To show the relationship between clock pulses, control signals, data bits and address bus levels.

Figure 3 shows the relationship (time) between clock pulse 1 (Ø1), clock pulse 2 (Ø2), Read/Write (R/W) line, Address Bus level, Valid Memory Address (VMA) line and Data from the microprocessor.

This illustration indicates that when the R/W line is low, VMA line is high and the Ø2 clock goes through a positive to negative transition, Data will be transferred from the Microprocessor to a specific location in memory. This location is determined by the address bus.

For further information see R6500 Hardware Manuel pages 1-15, 1-16, and 5-8.

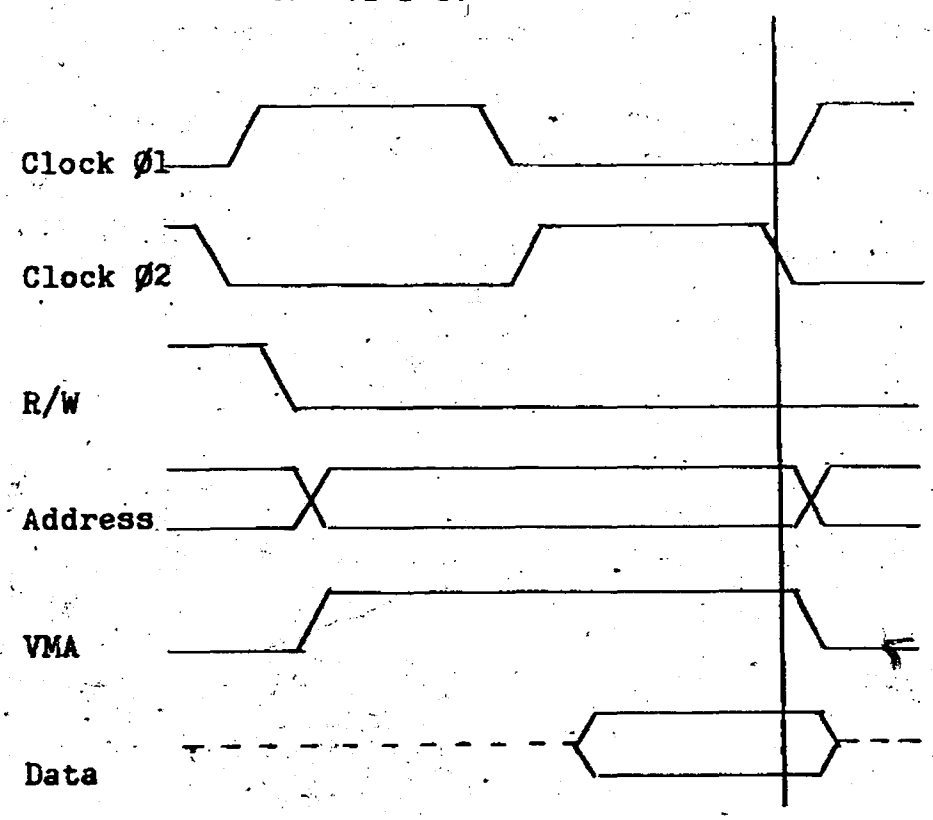


Figure 3

Figure 4, as in figure 3, shows the relationship (time) between clock pulse 1 (Ø1), clock pulse 2 (Ø2), Read/Write (R/W) line, Address Bus level, Valid memory Address (VMA) line and Data on the Data Bus.

When the R/W line is high, the VMA line is high and Ø2 goes through a positive to negative transition Data will be transferred from memory to the microprocessor. The specific location in memory is determined by the Address bus.

Note: In both figure 3 and 4 the transitions take place on the trailing edge of the Ø2 clock pulse.

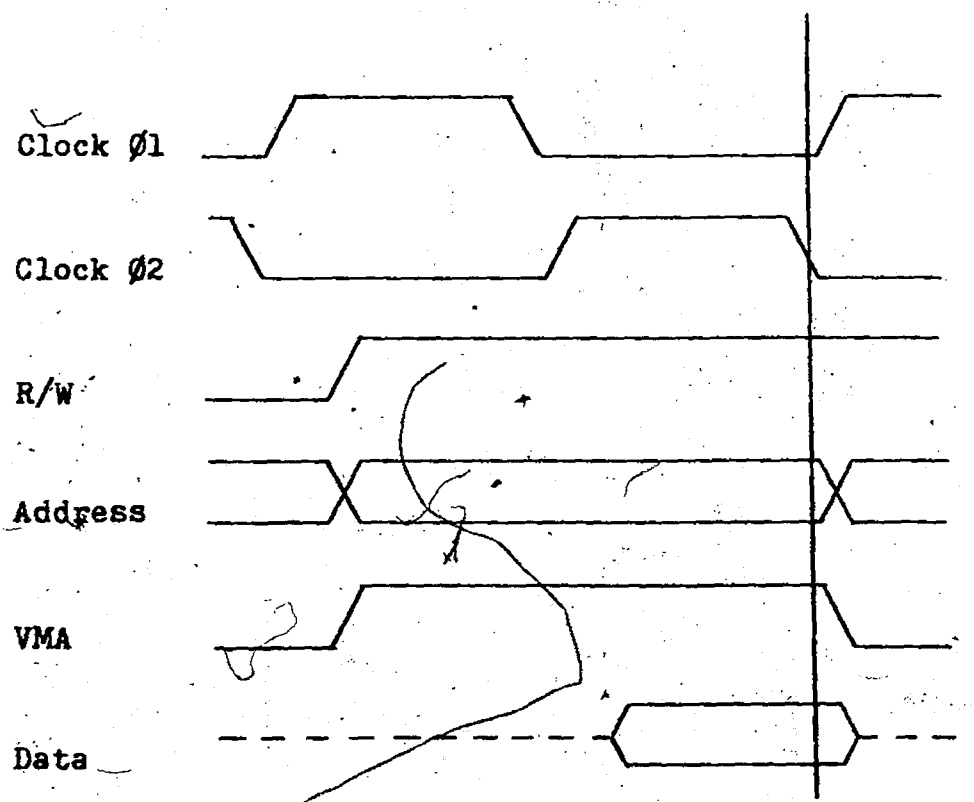


Figure 4

Learning Activity C3

Fabrication of an integrated circuit

Integrated circuits are constructed by selectively etching and diffusion of a silicon wafer. The method used to accomplish the selective etching is called photolithography. "Photolithography is the process by which a microscopic pattern is transferred from a photomask to a material layer in an actual circuit."

The following illustration shows the process of photolithography.

Silicon wafer with oxide coating on top

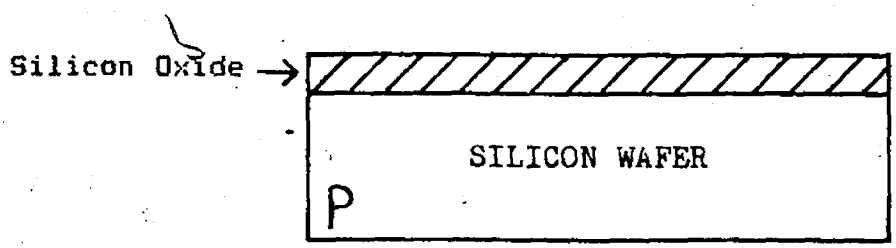


Photo resist applied to top of the silicon oxide

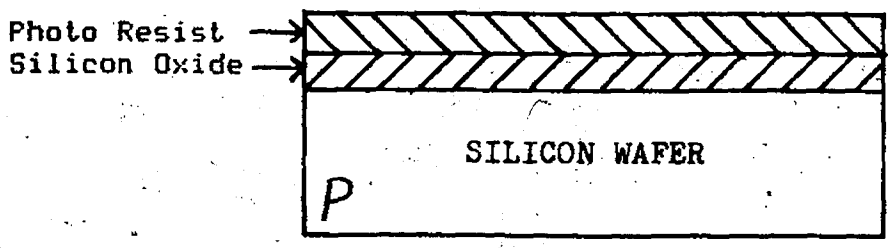
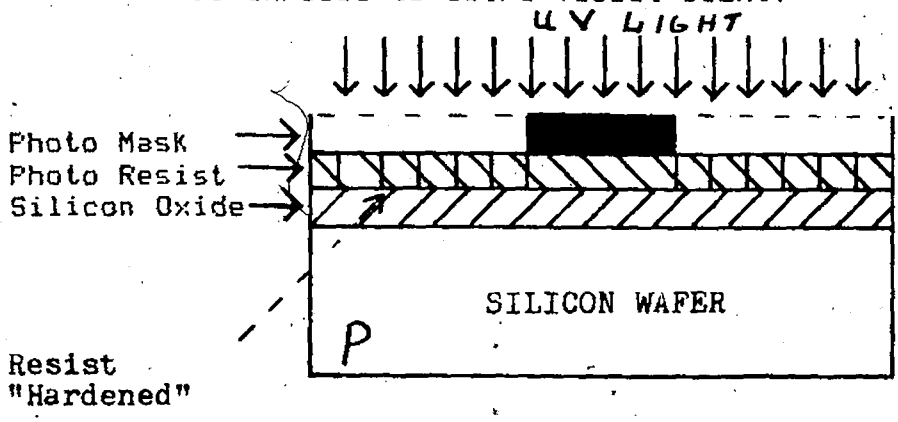
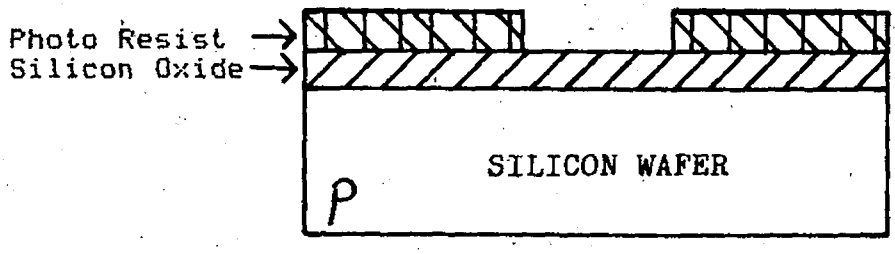


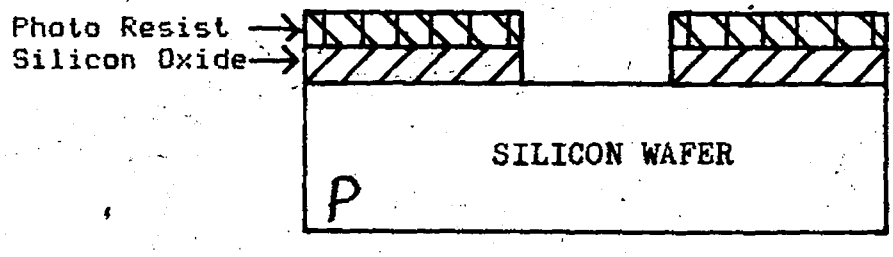
Photo mask placed on top of resist and the wafer is exposed to ultra violet light.



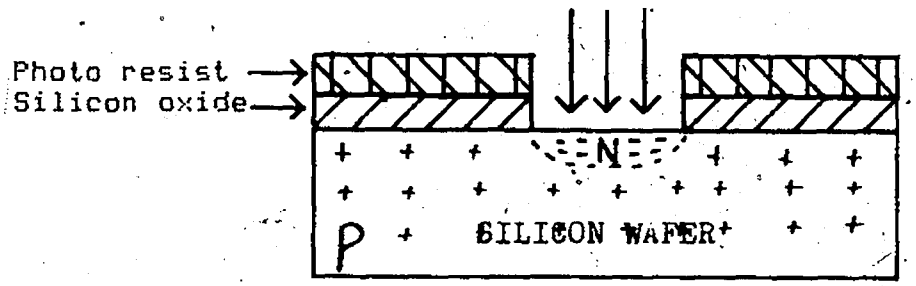
Wafer placed in developer solution: Window opened



Wafer immersed in hydrofloric acid, silicon oxide etched away.



Gaseous diffusion by N material to form an NP Junction.



In a similar manner other Junctions are formed in the silicon wafer. Finally metal contacts are attached to the Junction material. These contacts then become the transistor leads.

Learning Activity C4

Development of integrated circuits

Objective: To trace the development and fabrication of integrated circuits.

Read the following articles from Scientific American, September 1977, theme issue on Microelectronics.

Article 1
Microelectronics, by Robert N. Noyce, page 62.

Article 2
Microelectronic Circuit Elements, by James D. Meindl, page 70.

Article 3
The Fabrication of Microelectronic Circuits, by William G. Oldham, page 110.

Additional articles on computers are contained in TIME, February 20, 1978. This journal contains a special section on the Computer Society.

Learning Activity C5

Characteristics of Integrated Circuits

Objective: To compare semiconductor technologies.

TTL logic will be used as the standard for comparing other technologies.

Characteristics of TTL logic are:

NAND gate logic

High level Z out varies from 10 to 70 ohms

Low level Z out = R(sat)

Power supply is typically +5 volts

Power dissipation per gate is from 12 to 22 milliwatts

Propagation delay varies from 12 to 22 nanoseconds, depending upon circuity

Very good noise immunity

Maximum fan-out = 10

The following semiconductor technologies will be compared to TTL logic. TTL is used for comparison purposes because most digital circuits employ TTL logic.

PMOS (P-channel MOS). A relatively dense, cheap, but slow technology.

NMOS (N-channel MOS). A dense, cheap, medium-speed technology.

CMOS (Complementary MOS). A low-power, high noise immunity technology.

Schottky TTL. A high speed, high power, fully compatable technology that is not as dense as MOS.

Low-Power Schottky TTL. A low power version of the Schottky TTL.

ECL (emitter-coupled logic). An ultra high speed, high power technology.

IIL. A new technology that has many of the best characteristics of the other technologies. Theoretical predictions imply that IIL could eventually be denser and cheaper than NMOS, faster than TTL, and as low in power consumption and as high in noise immunity as CMOS.

Characteristics used for comparison purposes are:

1 Speed: The delay of a logic gate is a measure of it's switching time, short delays mean high switching speed.

2 Density: Typical gate size is a measure of the technology density. Very dense technology can produce single-chip microprocessors.

3 Cost: A measure of cost is the typical cost per gate.

4 Power consumption: A measure of power consumption is the power dissipated in a gate.

5 Noise Immunity: A measure of noise immunity is the variations permitted in voltage levels before a logic transition occurs.

6 Ruggedness: Ruggedness refers to the ability to withstand extreme conditions or variations in such factors as temperature, pressure, humidity, shock, torque, vibration, chemical conditions (such as acidity and salt build up), and nuclear radiation.

7 TTL Compatability: TTL compatability is important because most electronic systems are built with standard TTL circuits.

8 Maturity: Use of a mature technology makes system implementation simpler and avoids many of the difficulties that are always associated with state-of-the-art technology.

The technologies favoured by the various charcteristics are:

1 Speed: ECL and Schottky TTL technology is the fastest.

2 Density: PMOS and NMOS have the highest density and produce single chip microprocessors.

3 Cost: PMOS and NMOS are currently the cheapest per gate.

4 Noise Immunity: CMOS technology has the highest noise immunity. CMOS, however, may be damaged by large current variations or static changes. IIL has considerable potential here.

5 Power consumption: CMOS technology consumes the least power, ECL and Schottky TTL the most. IIL could challenge CMOS in this area.

6 Ruggedness: CMOS technology is the most rugged.

7 TTL Compatability: Schottky TTL technology are completely TTL compatable. Some of the newer NMOS and CMOS processors are also TTL compatable.

8 Maturity: NMOS is the most common technology used with microprocessors and CMOS and TTL are the most common technology used with digital circuits.

The above information is from Appendix 4, Introduction to Microprocessors: Software, Hardware, Programming. For further information see the above and Chapter 2 in Digital Programming.

Which semiconductor technology would you select to interface the data and address lines of a 6502 microprocessor to an external device that uses TTL logic? Justify your decision.

Learning Activity D1

Construction of a Logic Tester

Objective: To construct a logic tester.

The learning activity will outline a method that can be applied to construct a logic TTL logic tester.
The logic tester will have 7 individual circuits.

1. Power supply
2. 8 bounceless switches
3. 8 LED test monitors
4. One Hz oscillator
5. One hundred KHz oscillator
6. 2 seven segment readouts
7. Speaker with a TTL driver

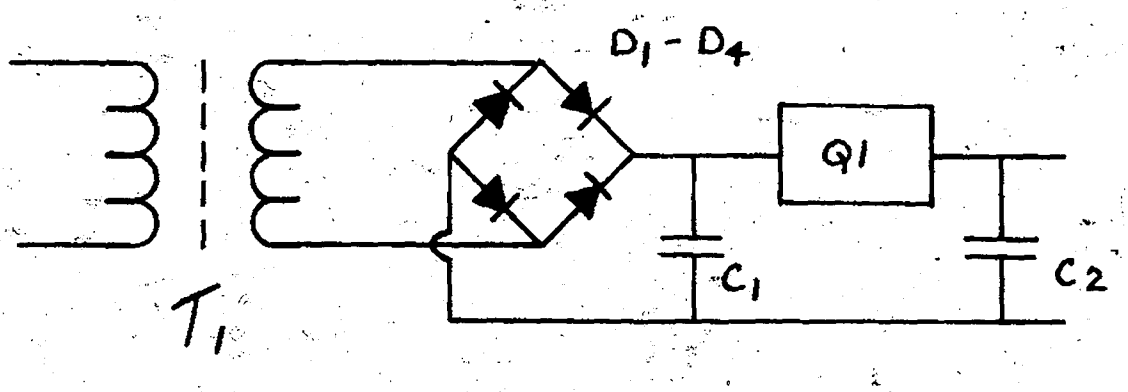
Mount the following hardware as per figure 1.

8 SPDT switchers
8 Tip jacks next to the switches
8 LEDs
8 Tip jacks next to the LEDs
2 Seven segment readouts
14 Tip jacks around the 7 segment readouts
1 Jack for the 1 Hz output
1 Jack for the 100 KHz output
1 2 1/2 inch speaker

Printed circuit boards will be required for each of the following circuits. These circuits could be combined to form 1 large printed circuit board or several smaller boards.

1. Power Supply

Schematic diagram 9



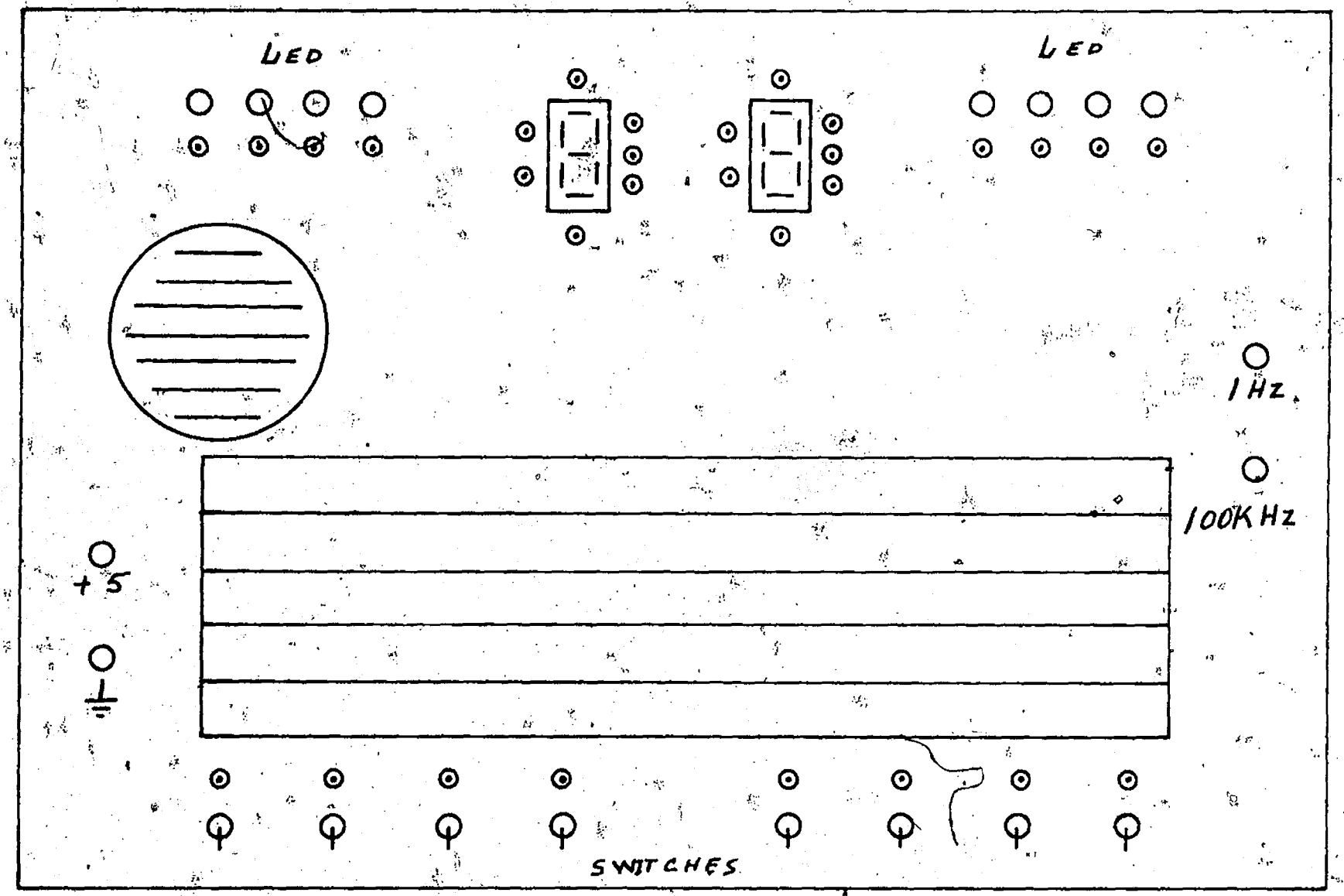
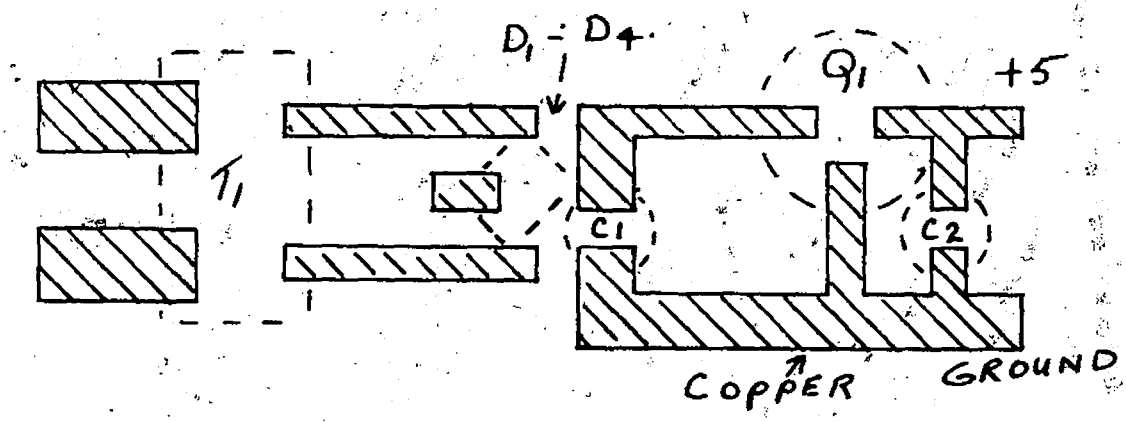


FIGURE 1

Parts:

T1	120 - 6/12 volt transformer
D1-D4	1A, 100V silicon diodes
C1	3000 microfarad 25 volt capacitor
C2	10 microfarad 25 volt capacitor
Q1	LM 309K 5V voltage regulator
	Power cord
	SPST Off-ON switch

Printed Circuit Board layout.



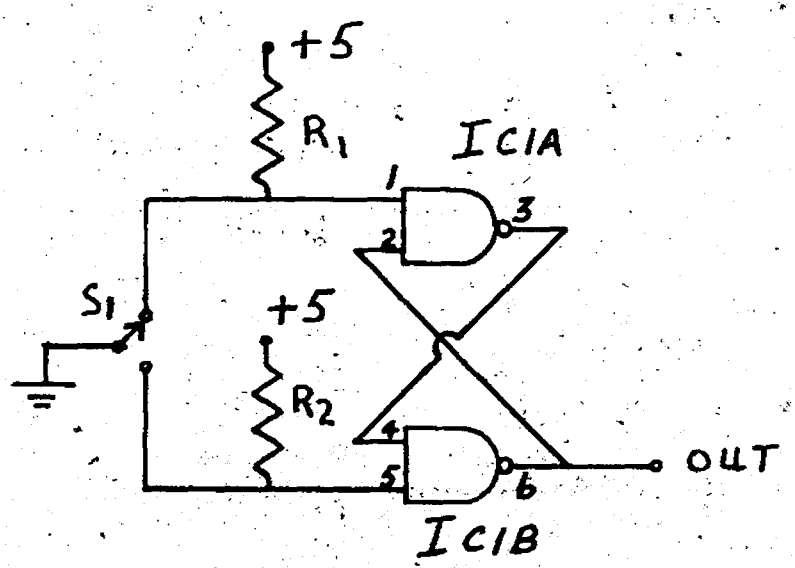
Directions:

Construct printed circuit (PC) board, see Learning Activity D2

Mount the transformer
Wire up bridge rectifiers
Wire up C1 and C2
Install Q1, LM 309K
Install power cord and OFF-ON switch

2. Bounceless switches

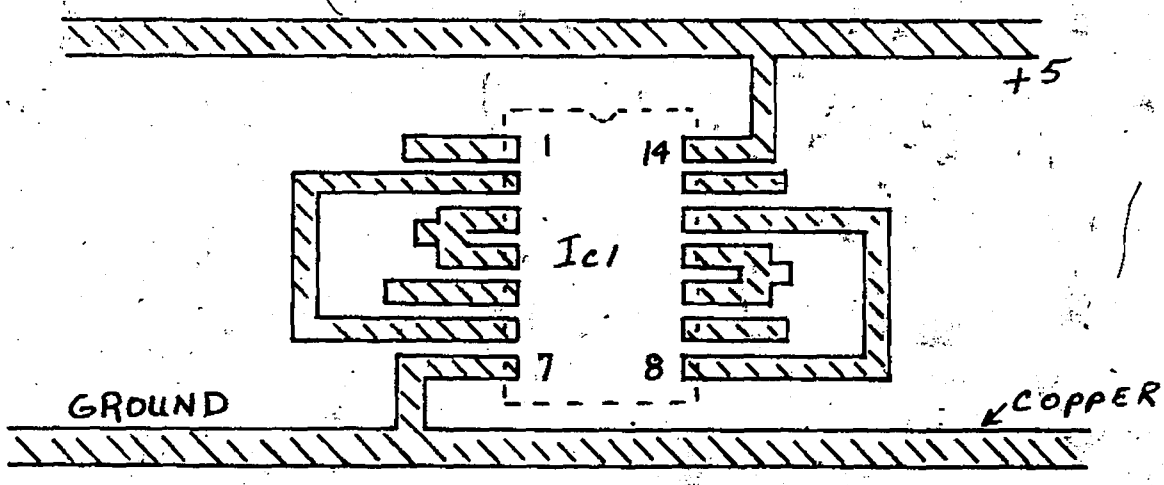
Schematic diagram



Parts:

S1	SPST
R1	4700 ohms 1/2 watts
R2	4700 ohms 1/2 watts
IC1	7400 integrated circuit

Partial PC board layout, similar layout for the other switches.

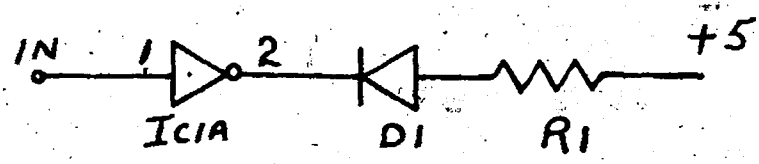


Directions:

Construct the PC board, for directions see D2
Connect R1 from pin 1 to +5
Connect R2 from pin 5 to +5
Wire S1 to pin 1 and pin 5, see diagram
Wire up the other side of IC1 in a similar manner see PC board layout and schematic diagram

3. LED test monitors

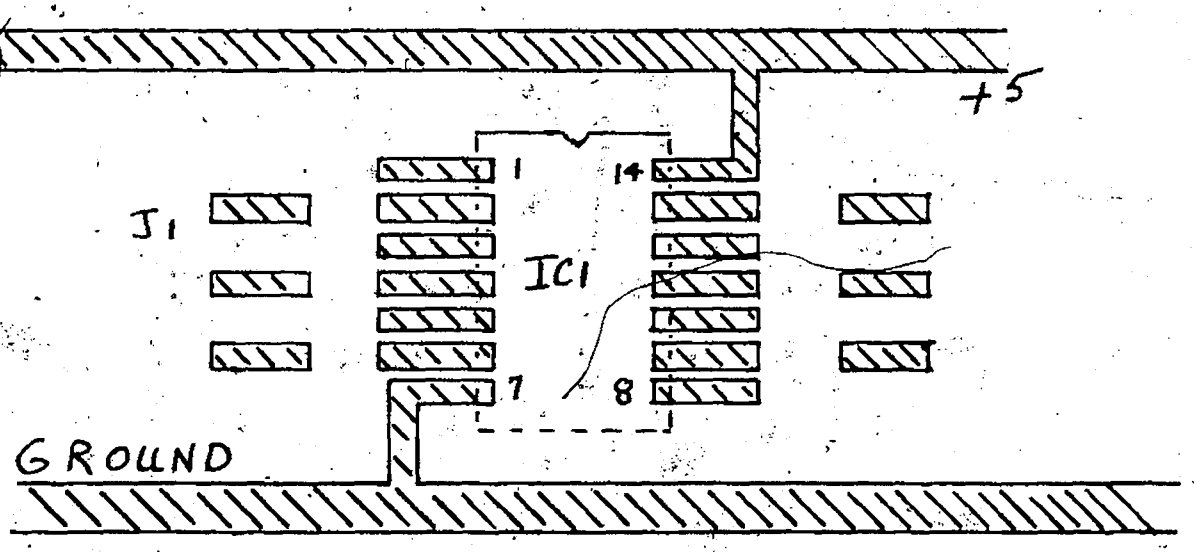
Schematic diagram



Parts:

IC1	7404
D1	LED (Light Emitting Diode) max I 15 ma
R1	200 ohms 1/2 watts

Partial PC board layout

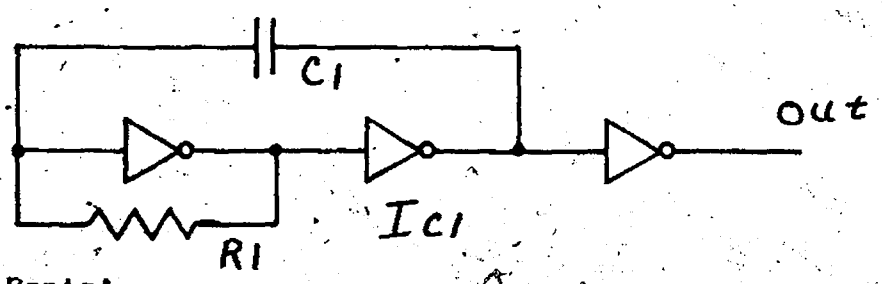


Directons

Construct PC board, see D2
Connect LED between pin 2 and the junction 1
Connect R1 from junction 1 to +5
Pin 1 is the input
Install components in a similar manner for the rest of the inverters in the 7404.

4 and 5 Oscillators

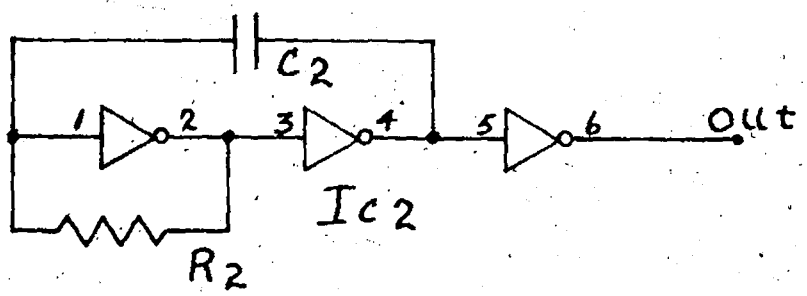
Schematic diagram for the 1 Hz oscillator



Parts:

IC1	7404
C1	1500 microfarad capacitor
R1	150 ohm resistor

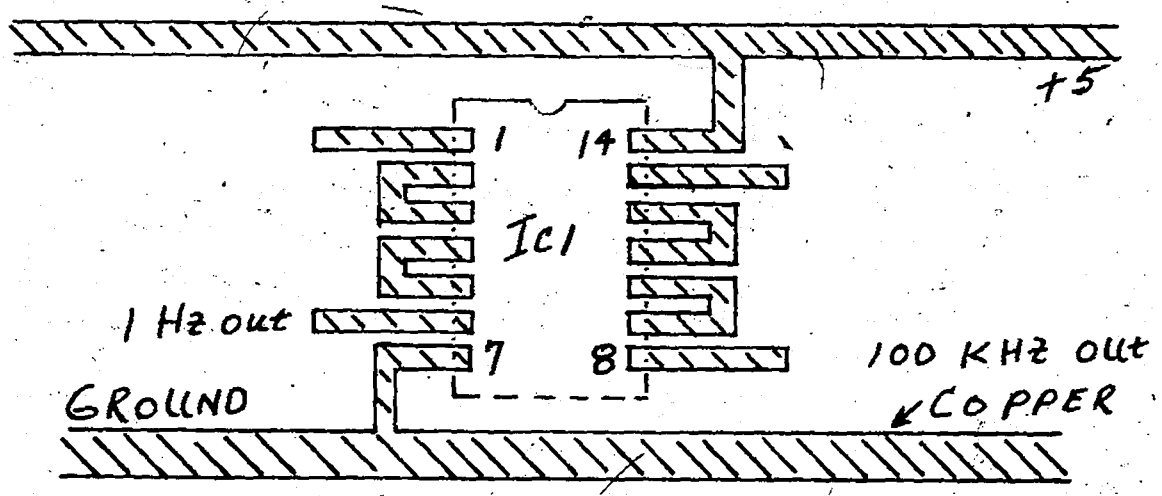
Schematic Diagram for 100 KHz oscillator



Parts:

IC2	7404
C2	.022 microfarad capacitor
R2	150 ohm resistor

PC board layout



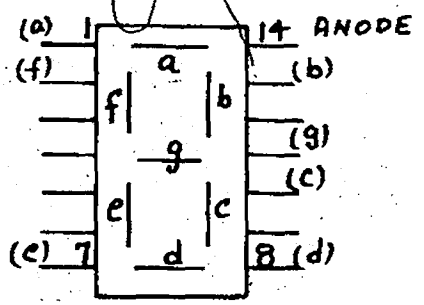
Directions

Construct PC board, see D2 for instructions
Connect C1 between pin 1 and pin 4
Connect R1 between pin 1 and pin 2
Connect C2 between pin 13 and pin 10
Connect R2 between pin 13 and pin 12

6 Seven segment readouts

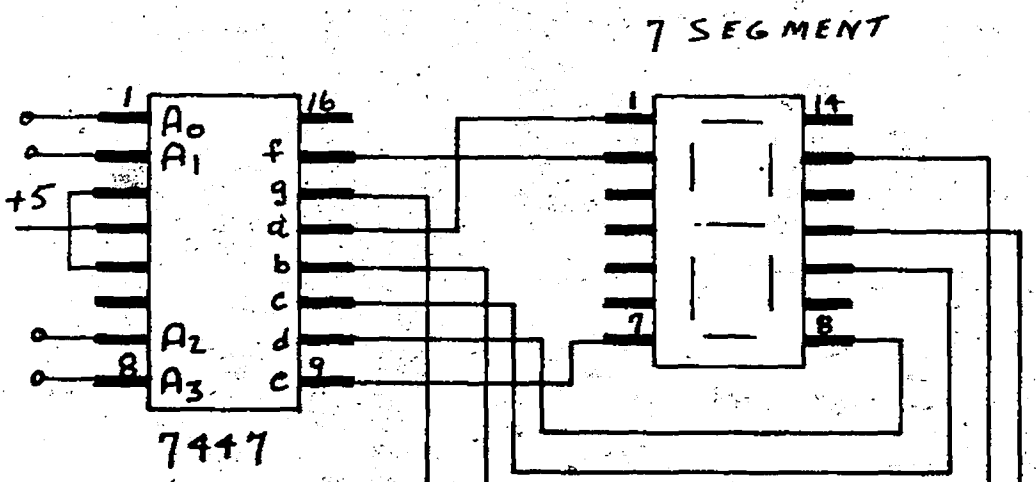
See Hewlett-Packard optoelectronics designers catalogue, 1979, page 41 for additional information.

The seven segment display should be mounted at the top centre of the logic tester, see figure 1.
Pictorial layout for the 7 segment readout



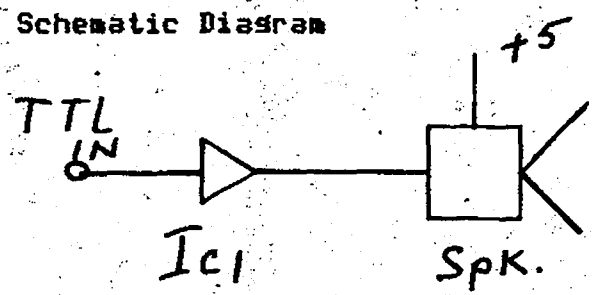
In order to have the 7 segment display indicate decimal values from BCD code a 7447 BCD to seven segment decoder driver must be interfaced with the 7 segment readout.

Interconnection diagram between 7 segment display and 7447 decoder driver.



7. Speaker

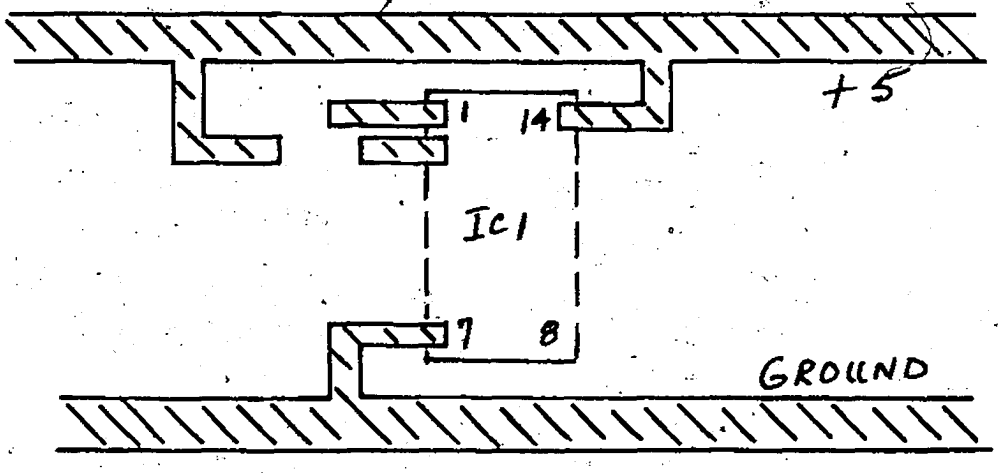
Schematic Diagram



Parts:

IC1 7407
SPK 2 1/2 inch low impedance speaker

PC board layout



Directions:

Construct PC board, see D2
Connect speaker between +5 and pin 2
TTL input to pin 1

Learning Activity D2

How to make a printed circuit board

Objective: To construct a printed circuit board.

Most printed circuits consist of etched copper foil wiring patterns bonded to any of several insulating bases. The best bases are made of glass-epoxy board, glass polyester is next, and phenolic is the cheapest.

The bases are lamenated with 1-ounce and 2-ounce copper foil on one or both sides. To form a printed circuit the copper foil must be etched.

Steps in constructing a printed circuit board.

1. Master Artwork.

a) After deciding on your basic circuit design, layout your components on a grid so you can determine the interconnecting lines etc.

b) Select a 1/10 inch grid pattern.
Place a piece of clear acetate over the grid pattern.

c) Using Bishop graphics pressure sensitive electronic component drafting aids lay out your circuit on the acetate overlay.

First locate all your terminals and IC patterns.
Then join your terminals and patterns using Bishop graphics tapes.

Bishop graphics patterns are supplied in 1X, 2X, and 4X actual size. It may be to your advantage to use the 2X or 4X scale and photographically reduce your artwork.

2. Producing the Negative.

Have a photographer photograph your artwork and supply you with the negative. If your artwork was 2X the photographer will have to reduce the picture size by 1/2 to give you the correct negative.

3. Select a printed circuit board.

The size of the printed circuit board will be determined by the size of the negative.

4. Clean the copper side of the board.

Clean the copper on the board with coppertone or similar copper cleaner. When the board is clean, running water will bead and run off the board in a manner similar to water on a newly polished car.

5. Dry the surface.

Either blow dry the surface or dry in an warm oven (150 degrees F) for ten minutes.

IMPORTANT: The following steps must be done using a safe light, for eample, a yellow bug light.

6. Apply the photo resist.

Use Kodak KPR-4 photo resist.

Two methods that can be used to apply the photo resist are:

a) Use a small brush and apply a thin even coat of KPR-4 on the copper side of the board.

b) Apply a fine spray of photo resist on the copper using an air brush. When spraying keep the air brush about 8-10 inches from he board. Apply light even strokes starting at the upper left hand corner and finishing at he extreme opposite corner. Be sure to overspray on each edge. As soon as the board is completed lay it flat, face up for a couple of minutes.

Be sure to clean up with laquer thinner after using the air brush.

It is important that the coating is even and will dry without runs. If the coating is uneven remove it with thinners and start over.

7. Dry the photo resist.

Three methods are possible:

a) Place the sensitized board in an oven set at 115 degrees F for 20 minutes.

b) Let the board dry overnignt at room temperature.

c) Force dry the board with a heat gun or spin dry.

REMEMBER: Still under the safe light.

8. Exposing the sensitzed board.

Place the board in a contact frame, sensitized surface up.

Place the negative, with the pattern showing the way it will finally appear, on top of the copper and close the glass frame top.

Expose the board to ultra violet light for 4 to 5 minutes.

Position the ultra violet light 6-10 inches away from the contact printer.

9. Develop the sensitized board.

Put 1/2 inch of Kodax KRP developer in an aluminum tray. Place the board in the developer for 1 minute, slightly agitating the tray during this time. This step will remove the photo resist that was not hardened by exposure to the ultra violet light, because of blockage by the negative.

NORMAL LIGHT NOW OK

10. Harden the resist.

Remove the board from the developer. Stand the board in a nearly vertical position, allow the resist to harden for 3 to 5 minutes.

Rince the board under gently running water.
Dry the board.

Errors in the printed circuit can be corrected by painting the faulty section with a fibre tipped resist pen.

11. Etch the board.

Immerse the board in a tray containing liquid ferric cloride. Be careful handling ferric cloride, keep it in a glass or plastic container.

Slightly agitate the tray while the board is being eteched by the ferric cloride. This process may take from 20 minutes to 2 hours.

Several methods are available to speed up the process.

a) Heat the solution to 140 degress F.

b) Spray the ferric cloride on the board.

c) Pump bubbles into the solution, this increases the asitation of the solution and speeds up etching.

Remove the board from the resist when the etching is complete. Remaining on the board is the printed circuit you designed.

12. Clean the board.

Use laquer thinner and a soft cloth to remove the resist remaining on the board. Polish the copper printed circuit with copper cleaner and fine steel wool.

Experience is the best teacher, if at first you don't succeed,try again. You too can make professional looking circuit boards.

For further information see

Printed Circuit Handbook by GC Electronics
Printed Circuit Handbook by Jana
Printed Circuit Handbook by Clyde F. Coombs published by McGraw-Hill Book Co.
Printed Circuit Boards for Microelectronics by J.A. Scarlett, published by Van Nostrand Reinhold.
73 Magazine, November 78, p. 240
June 77, p. 178
March 77, p. 136
April 77, p. 58

E Logic Gates

The purpose of this section is to learn how the basic gates operate. The method used will be to observe and verify the TTL logic gates under actual operating conditions.

E1 AND Gate

E2 OR Gate

E3 NAND Gate

E4 NOR Gate

E5 Buffer

E6 Inverter

E7 Exclusive-OR

E8 Exclusive-NOR

E9 Tri-State Buffer

Learning Activity E1

AND Gate

Objective: To verify the operation of an AND Gate.

Select a 7408 Integrated circuit.
Look up Pin connections in Fairchild TTL Data Book.

Wire the 7408 on the Logic Board.
+ 5 on Pin 14
Ground on Pin 7
Apply Logic A to Pin 2
Apply Logic B to Pin 1
Connect LED Monitor to Pin 3

Complete the following Truth Table for the 7408

Hi = 1
Lo = 0

A	B	Q
0	0	
1	0	
0	1	
1	1	

Draw the symbol for the AND Gate.

Conclusion: The output from a TTL AND Gate is high only when .....

Select a 3 input AND Gate and record it's Truth Table.

Learning Activity E2

OR Gate

Objective: To verify the operation of an OR Gate.

Select a 7432 integrated circuit.
Look up pin connections in the Fairchild TTL Data Book.

Wire the 7432 on the Logic Board.
+ 5 on Pin 14
Ground on Pin 7
Connect Logic A to Pin 1
Connect Logic B to Pin 2
Connect LED Monitor to Pin 3

Complete the following Truth Table for the 7432

Hi = 1
Lo = 0

A	B	Q
0	0	
1	0	
0	1	
1	1	

Draw the symbol for the OR Gate.

Conclusion: The output from a TTL OR Gate is high when

.....

Test the other 3 OR Gates in the 7432.

Learning Activity E3

NAND Gate

Objective: To verify the operation of a NAND Gate.

Select a 7400 integrated circuit.
Look up pin connections in the Fairchild TTL Data Book.

Wire the 7400 on the Logic Board,
+ 5 on Pin 14
Ground on Pin 7
Connect Logic A to Pin 1
Connect Logic B to Pin 2
Connect LED Monitor to Pin 3

Complete the following Truth Table for the 7400

Hi = 1
Lo = 0

A	B	Q
0	0	
1	0	
0	1	
1	1	

Draw the symbol for the NAND Gate.

Conclusion: The output from a TTL NAND Gate is high when

.....

Test the other 3 NAND Gates.

Learning Activity E4

NOR Gate

Objective: To verify the operation of a NOR Gate.

Select a 7402 integrated circuit.
Look up pin connections in the Fairchild TTL Data Book.

Wire the 7402 on the Logic Board
+ 5 on Pin 14
Ground on Pin 7
Connect Logic A to Pin 2
Connect Logic B to Pin 3
Connect LED Monitor to Pin 1

Complete the following Truth Table for the 7402

Hi = 1
Lo = 0

A	B	Q
0	0	
0	1	
1	0	
1	1	

Draw the symbol for the NOR Gate.

Conclusion: The output from a TTL NOR Gate is high when

.....

Select a 3 input NOR Gate and record it's Truth Table.

Learning Activity E5

BUFFER

Objective: To verify the operation of a BUFFER/Driver.

Select a 7407 integrated circuit.
Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 7407 on the Logic Board.
+ 5 on Pin 14
Ground on Pin 7
Logic switch A on Pin 1
LED Monitor on Pin 2

Since this Buffer has open collector A "pull up", resistor must be installed between Pin 2 and VCC, use 1K 1/4 watt.

Complete the following Truth Table for the 7407.

Hi = 1
Lo = 0

A	Q
0	
1	

Draw the symbol for a BUFFER.

Give at least one purpose a buffer can serve.

Learning Activity E6

INVERTER

Objective: To verify the operation of an INVERTER.

Select a 7404 integrated circuit.
Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 7404 on the Logic Board.
+ 5 on Pin 14
Ground on Pin 7
Logic switch A on Pin 1
LED Monitor on Pin 2 (Q)

Complete the following Truth Table for the 7404.

Hi = 1
Lo = 0

A	Q
0	
1	

Draw the symbol for an INVERTER.

Give at least two uses an inverter can serve.

Learning Activity E7

Exclusive-OR

Objective: To verify the operation of an Exclusive-OR Gate.

Select a 7486 integrated circuit.
Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 7486 on the Logic Board.
+ 5 on Pin 14
Ground on Pin 7
Logic A on Pin 1
Logic B on Pin 2
LED Monitor on Pin 3 (Q)

Complete the following Truth Table for the 7486.

Hi = 1
Lo = 0

A	B	Q
0	0	
0	1	
1	0	
1	1	

Draw the symbol for an Exclusive-OR Gate.

Conclusion: When A and B are Low, Q is .....

When A and B are High, Q is .....

When either A or B, but not both, is high, Q is .....

The output of an Exclusive-OR Gate is only high when either ..... or ..... is high.

Learning Activity E8

Exclusive-NOR

Objective: To verify the operation of an Exclusive-NOR Gate

Select a 74266 integrated circuit.
Look up Pin connections in the Fairchild TTL Data Book.
(Note: Open Collector)

Wire up the 74266 on the Logic Board.
+ 5 on Pin 14
Ground on Pin 7
Logic A on Pin 1
Logic B on Pin 2
LED Monitor on Pin 3 (Q)

1K 1/4 watt pull up resistor between Pin 3 and VCC
(required for open collector)

Complete the following Truth Table for the 74266.

Hi = 1
Lo = 0

A	B	Q
0	0	
0	1	
1	0	
1	1	

Draw the symbol for the TTL Exclusive-NOR Gate.

Conclusion: The output from an Exclusive-NOR Gate is low only when either ..... or ..... is high but ..... when both are high.

Learning Activity E9

Tri-State Buffer

Objective: To verify the operation of a Tri-State Buffer.

Tri-State Buffers are normally used where more than 1 data bus line feeds the same point. The output from the active bus can be high or low while the output from the non-active bus will be in a high impedance state and have no effect on the other data bus outputs.

Select a 74126 integrated circuit.
Look up pin connections in the Fairchild TTL Data Book.

Wire up the 74126 on the Logic Board.
+ 5 on pin 14
Ground on pin 7
Logic A on pin 2
Logic B on pin 1 (E)
Hewlett Packard logic probe to monitor pin 3 (Q)

In order for the Tri-State Buffer to output data E must be high.

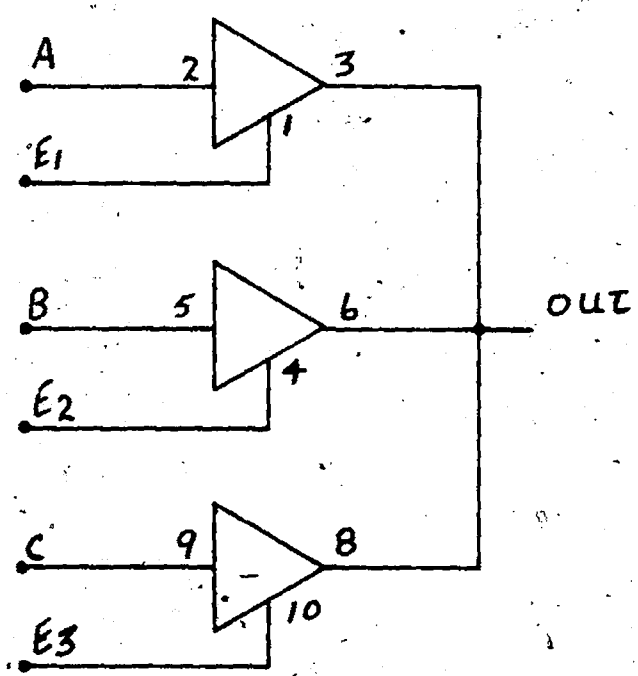
Complete the following Truth Table for the 74126

Hi = 1
Lo = 0
X = does not matter

E	A	Q
1	0	
1	1	
0	X	

You should observe that the buffer cannot process data when E is Lo. The buffer is in the high impedance state.

This following circuit allows several buffers to be connected to a point but only the buffer with E enabled controls the line. Wire up the circuit and test the above hypothesis.


A
2
3
1
E1
B
5
6
4
E2
C
9
8
10
E3
OUT

F BOOLEAN ALGEBRA

Suppose you are given the equation X↑2=X, you would probably say "That equation is false." However, if you limit the value of X to only 0 and 1 then the equation is valid.

George Boole (1815-1864), an English mathematican, developed a system of logic based on these two binary numbers, where 1 is true and 0 is false. There are two operations in this system + or *, they represent the logical operations OR and AND.

Digital logic uses only two digits 1 and 0. Mathematicans and circuit designers were quick to adopt Boole's operations and laws. The subject is now known as Boolean Algebra and provides a method to mathematically represent digital logic circuits.

Fundamental Boolean Identities

1 true
0 false
A and B are variables, that is they may represent 1 or 0.

$\overline{A}$ means not A * means AND + means OR

Identities	Comments	
1 $A * A = A$	A AND A = A	$1 * 1 = 1$
2 $A + A = A$	A OR A = A	$1 + 1 = 1$
3 $A * \overline{A} = 0$	$1 * 0 = 0$	
4 $A + \overline{A} = 1$	$1 + 0 = 1$	
5 $A + 1 = 1$	$1 + 1 = 1$	$0 + 1 = 1$
6 $A * 1 = A$	$1 * 1 = 1$	$0 * 1 = 0$

Commutative Laws: Same as in algebra except + means add and * means multiply.

$A + B = B + A$

$A * B = B * A$

Associative Laws: Same as in algebra except + means add and * means multiply.

$(A * B) * C = A * (B * C)$

$(A + B) + C = A + (B + C)$

Distributive Laws:

A * (B + C) = A * B + A * C Same as algebra

A + B * C = (A + B) * (A + C) Unique to Boolean algebra

Proof is supplied below using logic circuits and truth tables. Compare the results of the truth tables.

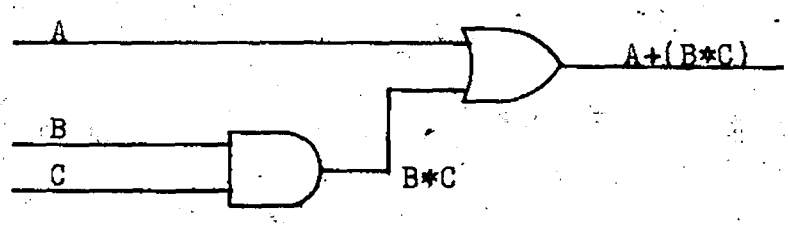


Figure 1

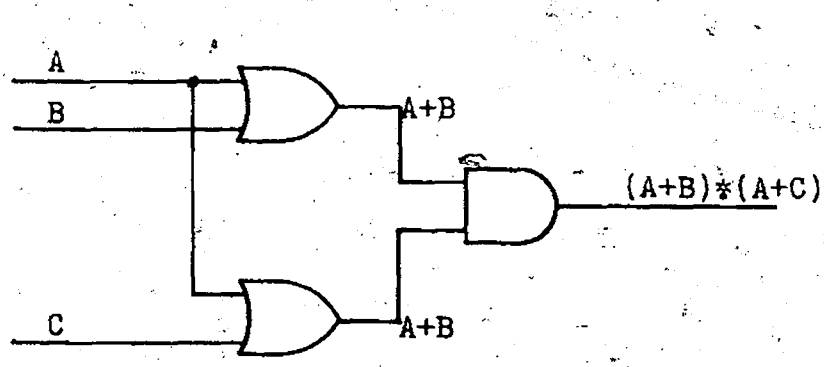


Figure 2

Truth table for figure 1

A	B	C	B*C	A+B*C
0	0	0	0	0
0	0	1	0	0
0	1	0	0	0
0	1	1	1	1
1	0	0	0	1
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Truth table for figure 2

A	B	C	A+B	A+C	(A+B)*(A+C)
0	0	0	0	0	0
0	0	1	0	1	0
0	1	0	1	0	0
0	1	1	1	1	1
1	0	0	1	1	1
1	0	1	1	1	1
1	1	0	1	1	1
1	1	1	1	1	1

The following exercises are intended to give you the necessary skills in applying Boolean Algebra and to provide for the student an introduction to using Boolean Algebra to simplify circuit design.

Refrence text: Practice Problems in Number Systems, Logic and Boolean Algebra by Bukstein.

Problem Number

31. Theorems, assignment applying Boolean theorems or identities.

32. Removing common factors, factoring using Boolean theorems.

33-35. The truth table, applications using truth tables. (The above should be completed before combinational or sequential logic circuits.)

36-42. Converting block diagram to truth tables and Boolean equation or vice versa.

46-53. Developing skill in manipulating Boolean equations.

54. Developing skill in manipulating minterm and maxterm equations.

The remainder of the exercises 55-61 involve construction of Karnaugh maps and their application. This section should be considered optional.

The teacher should check all work sheets when the assigned exercises are completed.

G Sequential Logic Circuits

Sequential logic circuits are used in a variety of timing , sequencing and storage functions. There can exist an almost infinite variety of sequential logic circuits. A representative sample of the most common types are included in these learning activities.

The method used will be to introduce you to various logic circuits and have you perform operations on them. The amount of information supplied is limited therefore you will have to investigate these circuits in more detail than that shown on your learning activity sheet.

1 Astable Multivibrator
2 Monostable multivibrator
3 Schmitt trigger
4 RS Flip Flop
5 JK Flip Flop
6 D Flip Flop (Latch)
7 Binary counters
8 Synchronous counters
9 Up/Down counters
10 Decade counters
11 BCD counters
12 Shift register 1
13 Shift register 2
14 Ram memory
15 Construction Job (decade counter)

Learning Activity G1

ASTABLE MULTIVIBRATOR

Objective: To construct an Astable Multivibrator.

To observe the operation of an Astable Multivibrator.

The Astable Multivibrator is a free running oscillator whose frequency is determined by the time constant in the base of each transistor. Construction of the Astable is accomplished by connecting two transistor amplifiers back to back.

Select 2N 4401 transistors to construct the oscillator.

Complete Lab # 25, (Basic Techniques in Electronic Instrumentation, by Diefenderfer, P.229.)

Draw a schematic diagram of an astable multivibrator. Give a brief explaination of it's operation.

Is it possible to construct an Astable Multivibrator using Logic Circuits?

Some research required.

Learning Activity G2

MONOSTABLE MULTIVIBRATOR

Objective: To learn the operating characteristics of a Monostable Multivibrator.

Select a 74121 integrated circuit.
Look up 74121 in the Fairchild TTL Data Book.

Complete Experiment # 12 (Experiments in Digital Principles, by Leach. P.59.).

Draw a schematic diagram of a monostable multivibrator.

How is output pulse width determined?

When triggering occurs, can the Monostable be re-triggered?

Give two applications for a Monostable Multivibrator.

Learning Activity G3

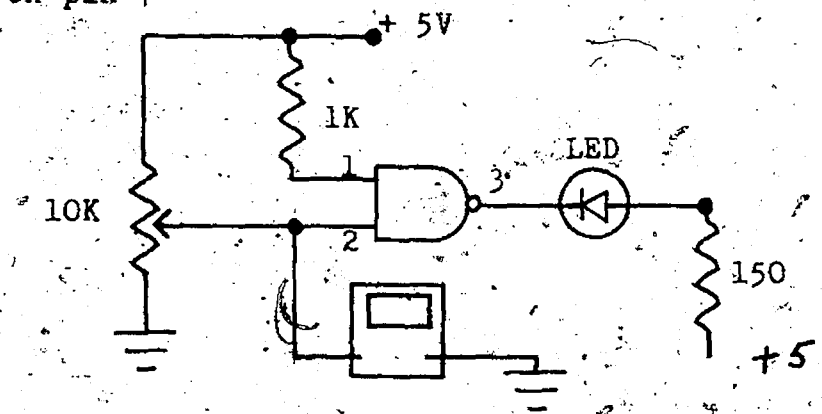
SCHMITT TRIGGER

Objective: To observe the operation of a Schmitt Trigger.
To record the Hysteresis Voltage.

The input to Schmitt Trigger is usually a nonlinear voltage and the output is a rectangular voltage.

Select a 74132 integrated circuit.
Look up Pin connections in Fairchild TTL Data Book.

Wire up the 74132 according to the following diagram.
+ 5 on pin 14
Ground on pin 7



Set R(1) at 0 resistance. Light .....

Increase resistance of R(1) until light is on.

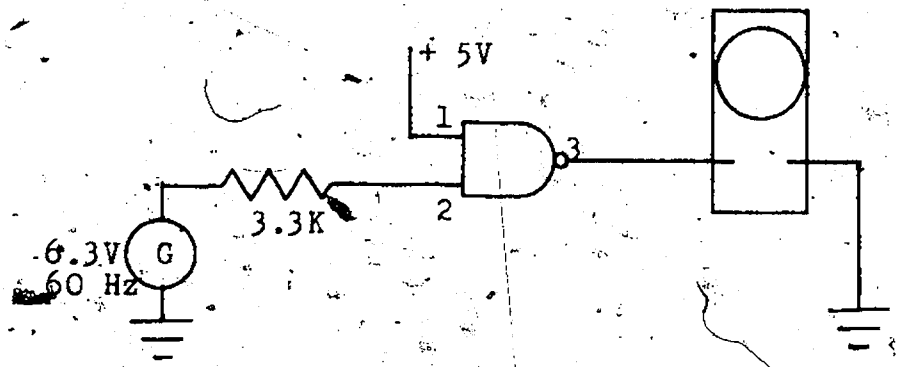
V(A) voltage at Pin 2 .....

Decrease resistance of R(1) until light is off.

V(B) voltage at Pin 2 .....

Hysteresis = V(A) - V(B) = ...

Wire up the following circuit using a 74132.
+ 5 on pin 14
Ground on pin 7



Record waveform at Pin 2 .....

Record waveform at Pin 3 .....

Make sure the input and output time are in line.

Can you suggest a use for a Schmitt Trigger?

Learning Activity G4

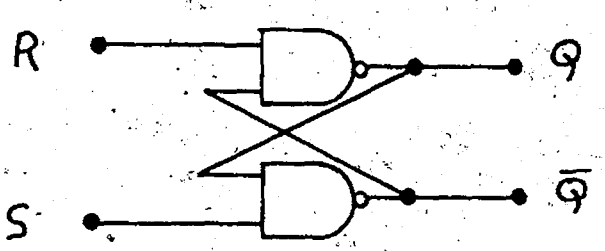
RS FLIP FLOP

Objective: To verify by experiment an RS Flip Flop using Nand Gates.

A Flip Flop is a fundamental digital circuit whose output is either a 1 or 0.

Select a 7400 integrated circuit.
Look up Pin connections in Fairchild TTL Data Book.

Wire up the following circuit.
+ 5 to Pin 14
Ground to Pin 7



Complete the following Truth Table.

R	S	Q	$\overline{Q}$
0	0		
0	1		
1	0		
1	1		

Observe the above sequence when switching R and S.

Conclusion: When R is Hi and S is Lo then Q .....

When S is Hi and R is Lo then Q .....

Note: Q is not always opposite $\overline{Q}$.

A RS Flip Flop can be used to construct a bounceless switch. See: (Basic Techniques in Electronic Instrumentation, by Diefenderfer, P.317).

What are the limitations of the RS Flip Flop? (See text).

Draw a schematic diagram of a clocked flip flop, (RS).

Learning Activity G5

JK FLIP FLOP

Objective: To observe the operation of the JK Flip Flop.

Most digital systems operate in a synchronous mode, i.e., are synchronized with a system clock, and the flip flops are required to change state in synchronism with a clock signal.

Select a 7473 JK Flip Flop integrated circuit.
Look up Pin connections in Fairchild TTL Data Book.

Wire a 7473 on Logic Board.
+ 5 on Pin 4
Ground on Pin 11
LED Monitor on Q
Hi to C(D1)

Using Logic Tester, complete the following Truth Table. A clock pulse (a logic level going from Lo-Hi-Lo) must be applied to C(P1) in order to observe the output.

Before clock *Input		After clock Output
J	K	Q
0	0	
1	0	
0	1	
1	1	

Conclusion: If J and K are low, application of the clock pulse has no effect on the output.

If J is high and K is low after the clock pulse is applied, Q .....

If K is high and J is low after the clock pulse os applied, Q .....

If K is high and J is low after the clock pulse is applied, Q .....

Figure 1 is a summary of the 7473 JK Flip Flop. Note: Transfer of data takes place on the negative (Hi-Lo) transition of clock.

With J and K High, apply 10 clock pulses.

Record the number of output pulses (Q) .....

State your conclusion.

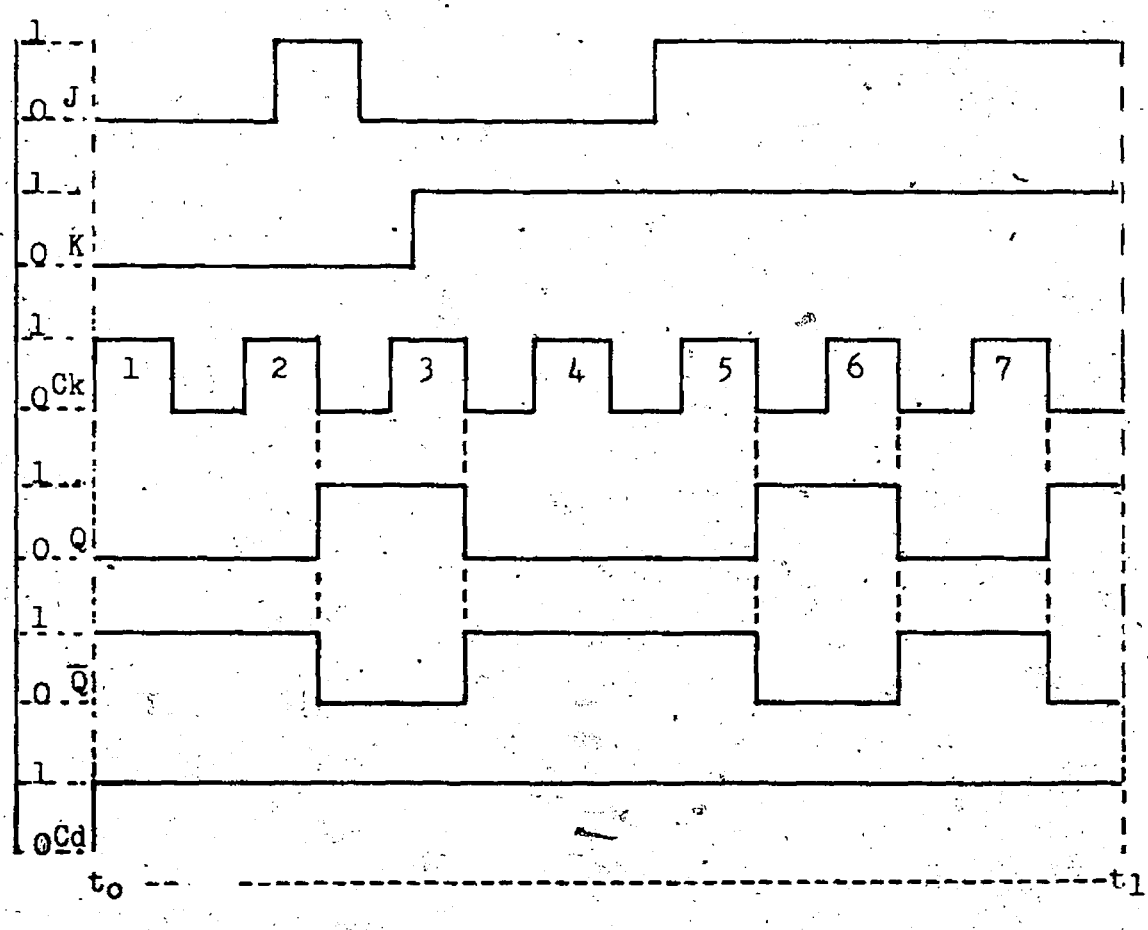


FIG. 1

Learning Activity G6

D FLIP FLOP

Objective: To observe the operation of the D Flip Flop.
To learn how data can be stored.

Select a 7474 D Flip Flop integrated circuit.
Look up Pin connections in Fairchild TTL Data Book.

Wire a 7474 on Logic Board.
+ 5 on Pin 14
Ground on Pin 7
Hi on both S(D) and C(D)

A clock pulse must be applied to C(P) in order to correctly observe the output.

Complete the following Truth Table.

Hi = 1
Lo = 0

Before clock	After Clock
D	Q
1	
0	

Repeat several times to check results.

What happens to Q if the clock is not applied?

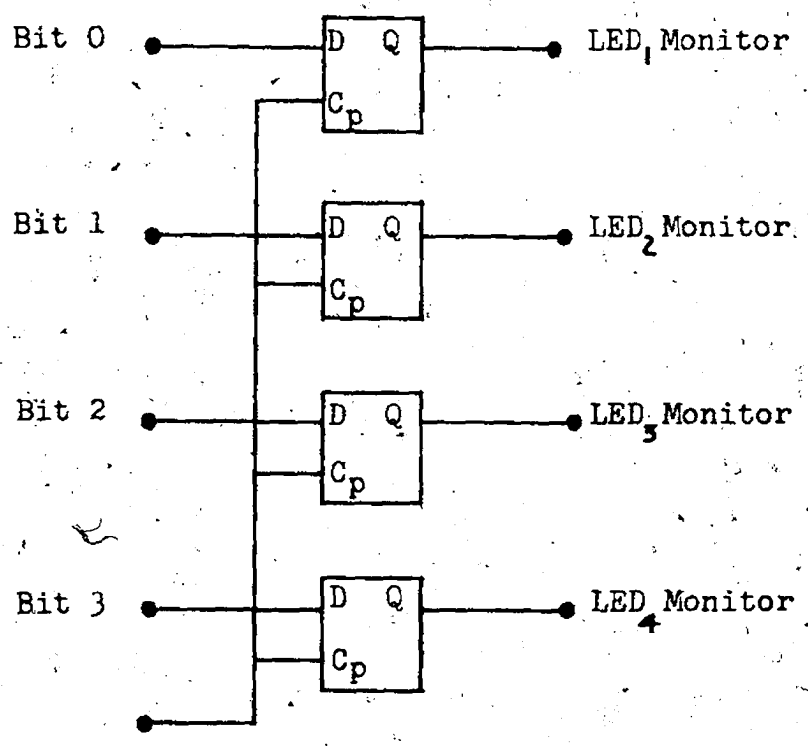
Conclusion: If D is low and a clock is applied, Q = ...

If D is high and a clock is applied, Q = ...

Note: Transfer of data takes place on the negative to positive (Lo-Hi) transition of the clock.

Construct the following circuit and record your results on the output for various inputs. Note: After clock pulse, data is locked in the output.

Logic Hi on C(D)
Logic Hi on S(D)



What is the purpose of C(D) and S(D)?
This is an example of a 4 Bit storage resister.

Optional: Construct and test an 8 Bit storage resister.

Show finished product to your teacher and be prepared to explain it's operation.

Learning Activity G7

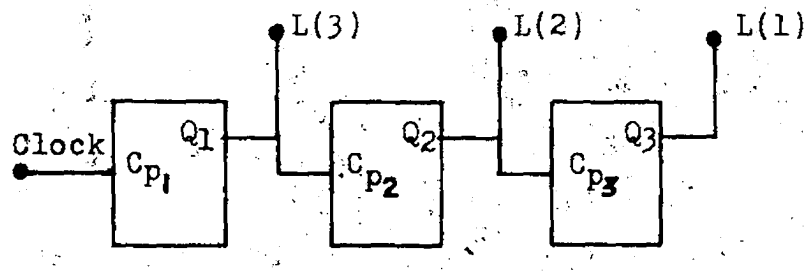
BINARY COUNTERS

Objective: To construct a divide by 2, 4, 8, 16 Counter from JK Flip Flops.

Previously we learned that one JK Flip Flop will divide the clock pulse by 2, that 4 clock pulses in C(P) gives 2 pulses out at Q.

Select two 7473 JK Flip Flop integrated circuits. Look up Pin connections in Fairchild TTL Data Book.

Wire up the following circuit
+ 5 on Pin 4
Ground on Pin 11
Hi on C(D) . Switch Lo to Hi
LED Monitor on Q(1) and Q(2) and Q(3)
Logic Hi on J and K



Clock	Output			Decimal
	L(1)=4	L(2)=2	L(3)=1	
0	0	0	0	0
1				
2				
3				
4				
5				
6				
7				
8				

Complete the above table. Rem., clock requires Hi to Lo transition.

Conclusion: Q(1) divides by .....

Q(1) with Q(2) divides by .....

Q(1) with Q(2) and Q(3) divides by .....

Using the above information, Design and Test a circuit that will divide by 16.

It is possible to divide by 10 or some other base.

Design a circuit that will divide by 10.

For more information see H.P. Video tape binary on counters.

Learning Activity G8

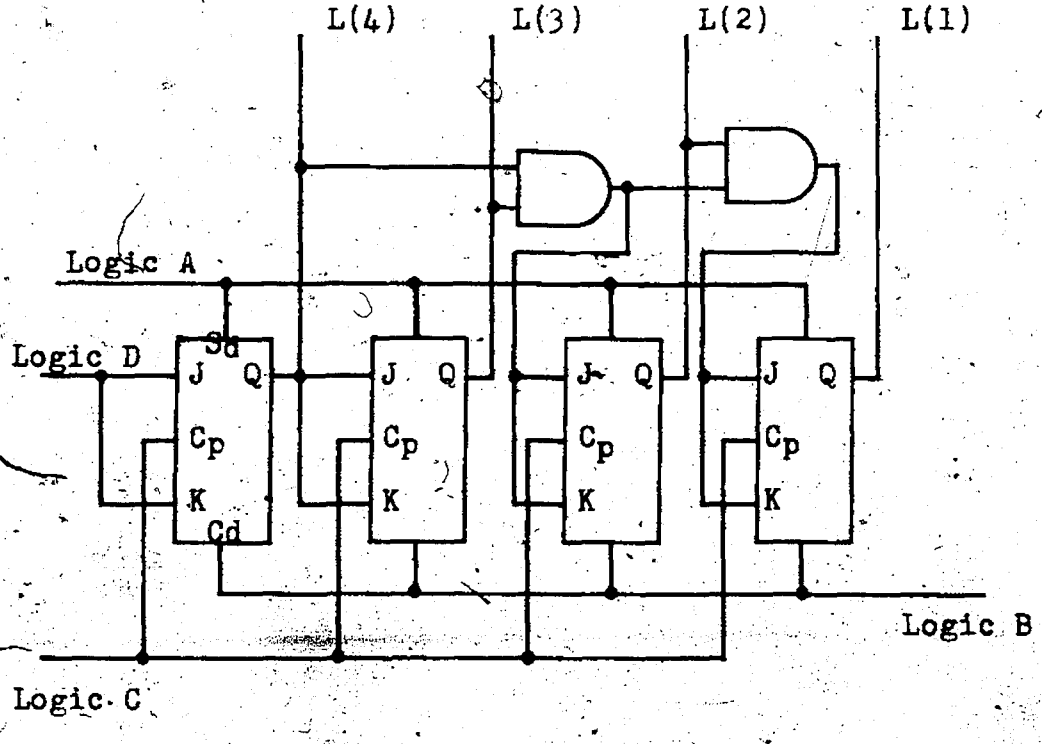
SYNCHRONOUS COUNTERS

Objective: To observe and record the operation of a Synchronous Counter.

A synchronous counter (74160) changes the state of all flip-flops simultaneously providing a much higher frequency capability. An asynchronous counter (7490) requires the output of one flip-flop to change state to trigger the next flip-flop. In asynchronous counters the maximum input frequency is limited by the time it takes for the pulse to ripple through the flip-flops.

Select two 7476 and one 7408 integrated circuits.
Look up Pin connections in the Fairchild TTL Data Book.

Wire up the circuit in Fig. 1.
+ 5 on Pin 5, 7476 and Pin 14, 7408
Ground on Pin 13, 7476 and Pin 7, 7408
LED Monitors on Q(1) - Q(4)
Logic A on Set S(D)
Logic B on Clear C(D)
Logic C on clock C(P)
Logic D on J(1) and K(1), switch to Hi



Switch Logic A, Lo - Hi. Note: All LED's on.

Switch Logic B, Lo - Hi. Note: All LED's off.

Switch Logic C, Hi - Lo.

This is clock pulse no. 1. Record the outputs, complete Table 1.

Clock	Output				Decimal
	L(1)=8	L(2)=4	L(3)=2	L(4)=1	
0	0	0	0	0	0
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					

You should observe that the synchronous counter works similar to the asynchronous counter except that the change in Q levels is instantaneous.

Learning Activity G9

UP/DOWN BINARY COUNTERS

Objective: To observe and record the operation of an Up/Down Binary Counter with parallel load.

Select a 74193 integrated circuit.

Look up pin out in Fairchild TTL Data Book

Wire up the 74193 on the Logic board.

+ 5 on Pin 16

Ground on Pin 8

LED Monitors on Q(0) - Q(3)

Logic switches on P(0) - P(3)

Clock on count up Pin 5

Clock on count down Pin 4

LED Monitors on terminal count down

LED Monitors on terminal count up

Logic switches on master reset

Logic switches on parallel load

Set P(0) = 0

P(1) = 1

P(2) = 1

P(3) = 0

MR Hi - Lo

PL Hi - Lo - Hi

Complete the following chart.

Count up Parallel Load

C(Pd)	C(Pu)	Q(0)	Q(1)	Q(2)	Q(3)	Carry
1	0	0	1	1	0	
1	∫					
1	∫					
1	∫					
1	∫					
1	∫					
1	∫					
1	∫					
1	∫					
1	∫					

∫ CLOCK

Count Down Parallel Load

C(Pd)	C(Pu)	Q(0)	Q(1)	Q(2)	Q(3)	Borrow
0	1	0	1	1	0	
∫	1					
∫	1					
∫	1					
∫	1					
∫	1					
∫	1					
∫	1					
∫	1					
∫	1					
∫	1					
∫	1					

List four functions the 74193 can perform.

Learning Activity G10

DECADE COUNTERS

Objective: To observe and record the operation of a Decade Counter.

Select three 7490 Decade Counters intesrated circuits.

Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 7490.

+ 5 on Pin 5

Ground on Pin 10

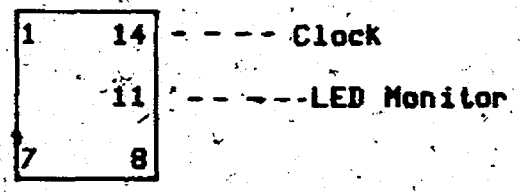
Losic A on pin 2 and 3

Lo on pin 6 and 7

LED Monitor on Pin 11

Pin 1 tied to Pin 12

Switch Losic A Hi to Lo



7490

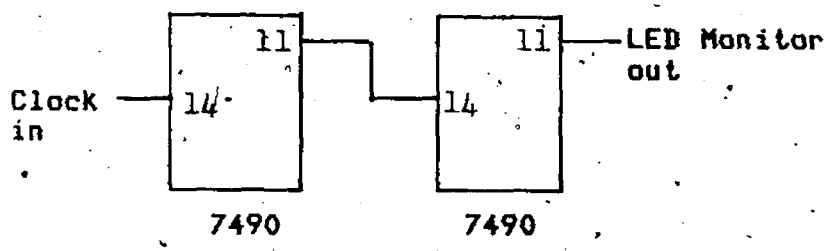
Apply 20 clock pulses to Pin 14.

Count pulses out at Pin 11.

A Decade Counter divides by .....

Cascade Decade Counters

Pins 1, 5, 10, 3, 6, 2, 7, as before.



Apply 200 clock pulses to input .....

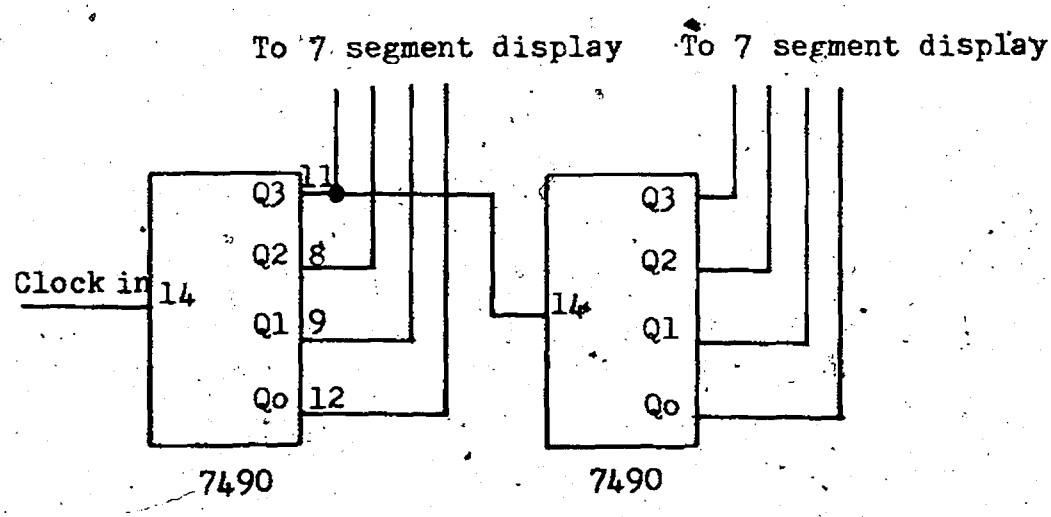
Count pulses out .....

2 Decade Counters in Cascade (Series) divide by .....

3 Decade Counters in Cascade divide by .....

Wire up the following circuit.

Pins 1, 5, 10, 3, 6, 2, 7 as before.



Apply clock pulses to input.

What do you observe?

Learning Activity G11

Binary Coded Decimal (BCD) COUNTER

Objective: To observe the operation of a Binary Coded Decimal Counter.

A BCD Counter is a sequential circuit that counts by tens. That is, the counter will cycle from 0000 to 1001 (9) and back to 0000.

Select a 7490 integrated circuit.
Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 7490 on the Logic Board.
+ 5 on Pin 5
Ground on Pin 10
LED Monitors on Q(0) - Q(3)
Pin 1 tied to Pin 12
Logic A on pin 2 and 3
Lo on 6 and 7
Switch Logic A Hi to Lo
Apply clock pulse to Pin 14

Complete the following table.

Clock		Q_0	Q_1	Q_2	Q_3		Decimal
⌊		0	0	0	0		0
⌊							1
⌊							2
⌊							3
⌊							4
⌊							5
⌊							6
⌊							7
⌊							8
⌊							9

Learning Activity G12

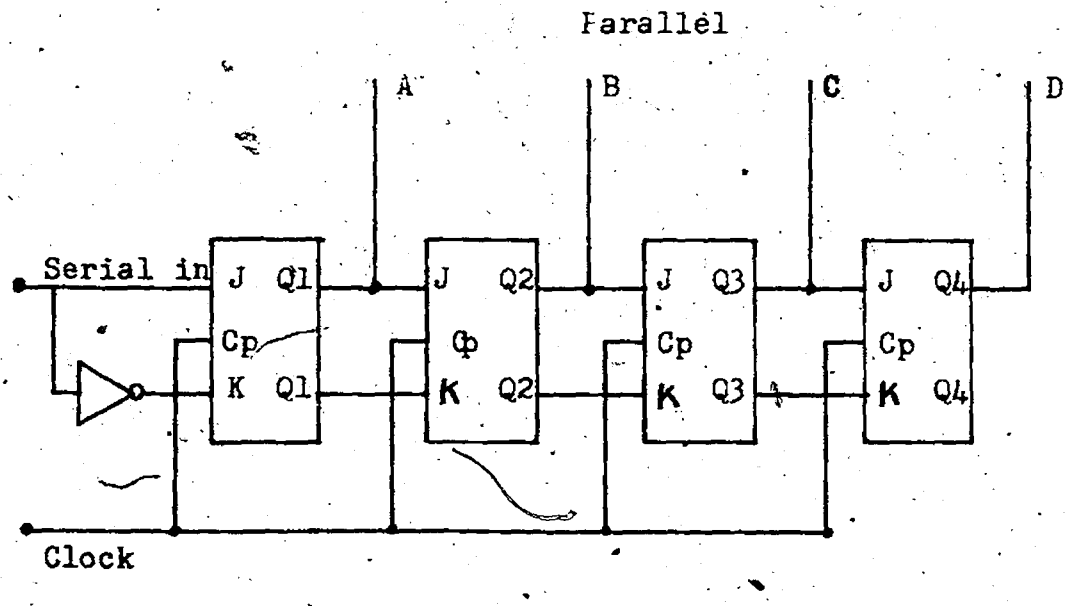
SHIFT REGISTERS I

Objective: To be able to construct a Shift Register from flip flops.

Registers are used to store binary information and to perform digital arithmetic operations.

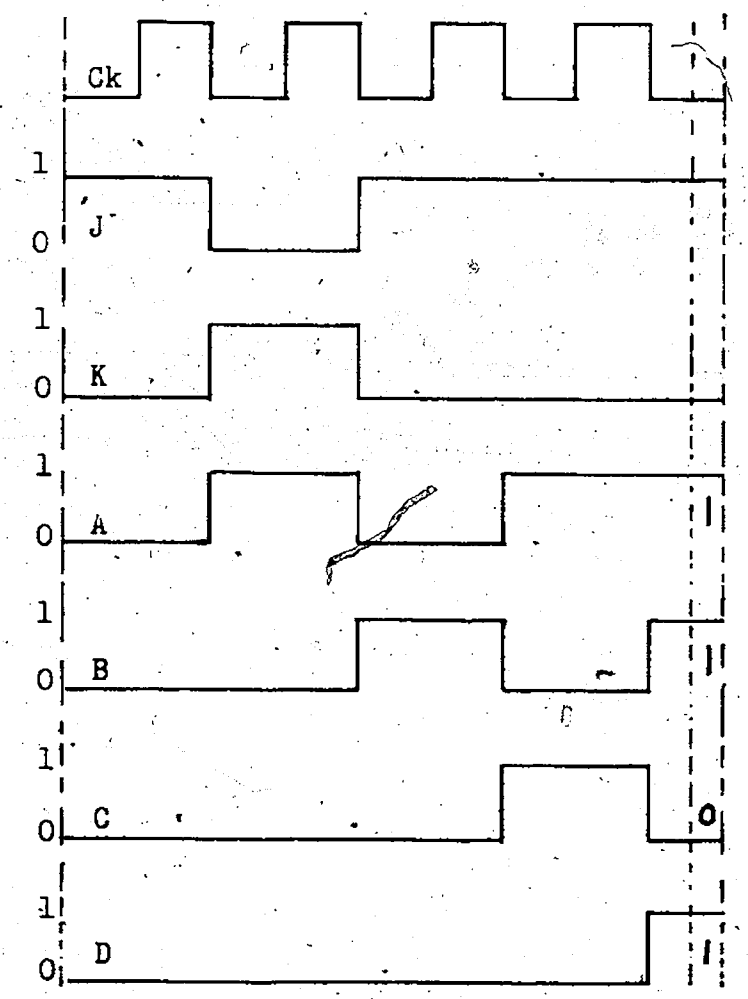
Select two 7473 JK Flip Flops integrated circuits.
Select one 7400 Nand Gate integrated circuits.
Look up pin out in the Fairchild TTL Data Book

Wire up the following Shift Register using 7473 Flip Flop and 7400 Nand Gate.
+ 5 on Pin 14, 7400 and Pin 4, 7473
Ground on Pin 7, 7400 and Pin 11, 7473
Logic Hi on C(D)
LED Monitor on A, B, C, D



Apply 1101 to the input, LSB first
(MSB) (LSB)

Ladder Chart for the shift register



Serial in (J) (MSB)1101(LSB)
Parallel out A,B,C,D 1101

Repeat the above and observe the serial input is shifted through the JK flip flops. Note 4 clock pulses are required to shift in a 4 bit word.

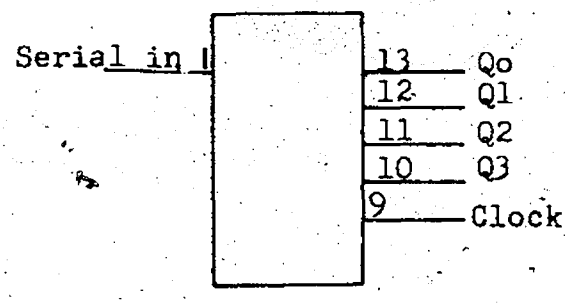
Learning Activity G13

SHIFT REGISTERS II

Objective: To observe the operation of a 4-Bit Right/Left Shift Register.

Select a 7495A Shift Register integrated circuit. Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 7495A for synchronous, shift right serial input.
+ 5 on Pin 14
Ground on Pin 7
Logic Lo to Pin 6



Construct a Ladder Chart to compare clock, serial in (4 Bit) and Q(0), Q(1), Q(2), Q(3).

How would you reset the output to 0?

Learning Activity G14

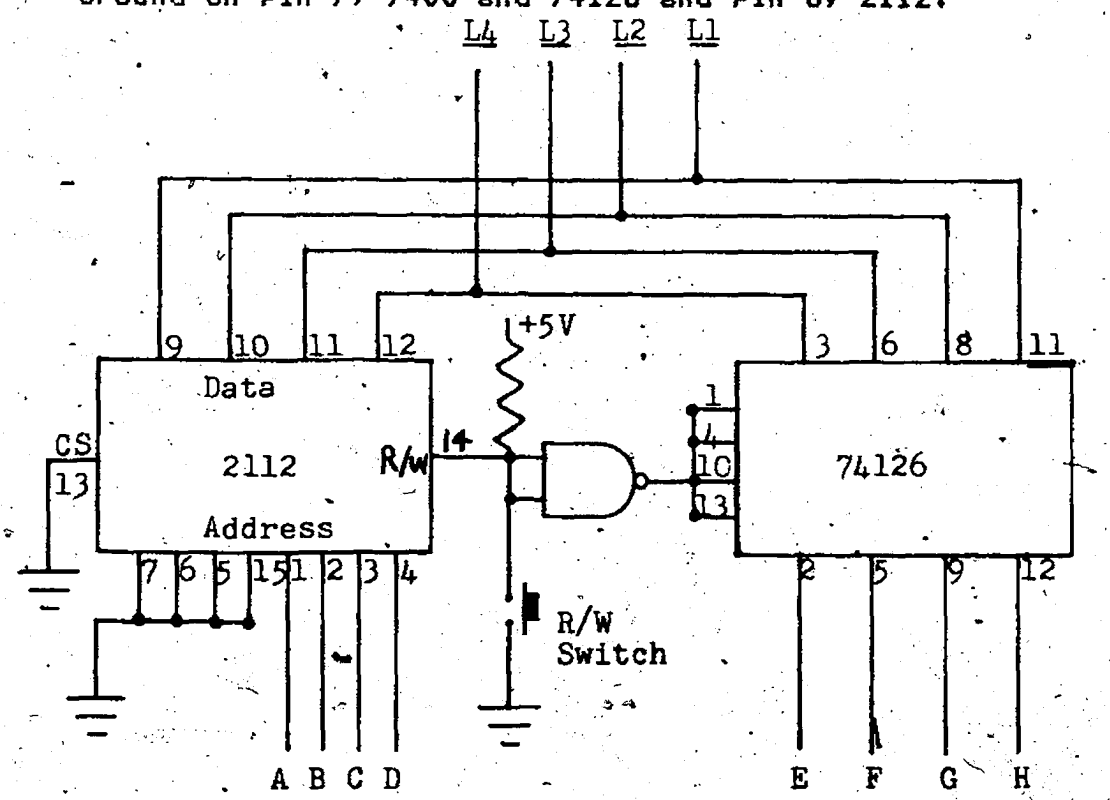
RAM Memory

Objective: To write data into a Random Access Memory (RAM).
To read data that has been previously written into a RAM.

Reading data from a RAM or writing data into a RAM involves setting up the data lines, address lines, chip select line and the R/W line. For more information on the RAM, read page 7-20 to 7-31, (Microprocessors, Heathkit) or similar text.

Select a 7400, 7426 and 2112 integrated circuit.
Look up pin out in Fairchild TTL Data Book and MOS Data Book.

Wire up the following circuit.
+5 on pin 14, 7400 and 74126 and pin 16, 2112
Ground on pin 7, 7400 and 74126 and pin 8, 2112.



Logic switches A,B,C and D select the RAM address.

Logic switches E,F,G and H select the RAM data.

R/W logic switch controls the read/write data.

Switch A=0, B=0, C=0, D=0

Data on the LEDs= .....

This is the contents of memory location 0000.

Switch E=0, F=0, G=1, H=1.

Press R/W switch

Data on the LEDs=.....

Note: the contents of address 0000 have changed.

Why has this happened?

Select address 0001 by setting

A=0, B=0, C=0, D=1

Set input data by selecting

E=0, F=0, G=0, H=1

Press R/W switch

Data on the LEDs= .....

What is the contents of address 0001?

Describe how a RAM can read and write data.

Note: A ROM is simply a RAM that cannot normally be written into.

Complete the following table by writing data into the indicated address and then recording that data.

Address	Data
0000	0011
0001	0001
0010	
0011	
0100	
0101	
0110	
0111	
1000	
1001	
1010	
1011	
1100	
1101	
1110	
1111	

Define the following: RAM.....

ROM.....

PROM.....

EPROM.....

Extra: Wire up two 2112 integrated circuits to form a RAM with 8 bits of data. Read and write into the RAM. For assistance see (Microprocessors, Heathkit, page 10-12).

Learning Activity G15

Sequential Circuits

Objective: To construct a basketball scoreboard that will register scores from 0 to 99. It must be able to reset to 0 and read out on the logic board (seven segment).

See Teacher for:
Special instructions for constructing P.C. board.
Special instructions for wire wrap.

You will complete the readout part of this project when you complete combinational logic circuits.

H Combinational Logic Circuits

Combinational Logic Circuits are digital circuits that are made up of gates and inverters. The output of a combinational logic circuit is a function of the state of it's inputs, the type of gates used, and how they are connected.

H1 Decoders I

H2 Decoders II

H3 Encoders

H4 Multiplexers

H5 Demultiplexers

H6 Multiplexers Demultiplexers

H7 Half Adder

H8 Full Adder

H9 Parity Generator

H10 Parity Checker

H11 Four-Bit Arithmetic Logic Unit

H12 Construction Job

Learning Activity H1

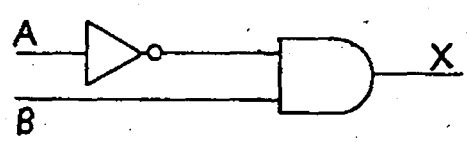
DECODERS I

Objective: To verify the operation of the AND Decoders.
To construct and test several decoders.

The input to a decoder is a parallel binary number and the output is a binary signal that indicates if a specific binary number is present at the input.

Basic Decoder is the AND Gate.

Wire up the following circuit and confirm the Truth Table.

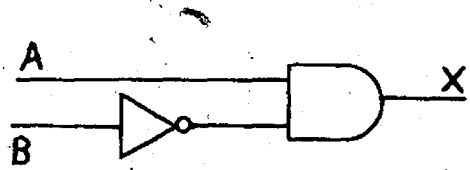


A	B	X
0	0	0
0	1	1
1	0	0
1	1	0

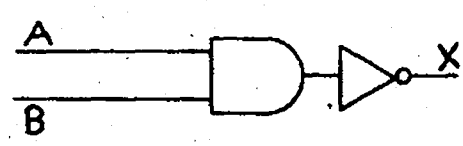
$X = \bar{A}.B$

It is very important that you understand the above Boolean equation.

What is the Boolean equation for the following Decoders?



X = ...



X = ...

Learning Activity H2

DECODERS II

Objective: To construct several decoders and record their operation.

A Decoder is a Logic circuit that will detect the presence of a specific binary number or word and output a specific binary number or word.

Select a 7442 integrated circuit.
Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 7442.
+ 5 on Pin 16
Ground on Pin 8
Logic switches on A(0), A(1), A(2), A(3)
LED Monitors on Q(0) - Q(9)

Complete the following Truth Table.

Binary in output

A3	A2	A1	A0	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
0	0	0	0	0	1	1	1	1	1	1	1	1	1
0	0	0	1										
0	0	1	0										
0	0	1	1										
0	1	0	0										
0	1	0	1										
0	1	1	0										
0	1	1	1										
1	0	0	0										
1	0	0	1										
1	0	1	0										
1	0	1	1										
1	1	0	0										
1	1	0	1										
1	1	1	0										
1	1	1	1										

Which output goes low for the binary word 1001? .....

Select a 74154, 1 of 16 Decoder, integrated circuit.
Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 74154.
+ 5 on pin 24
Ground on pin 12
E(0) and E(1) to Lo
LED Monitors to Q(0) - Q(15)
Logic switches to A(0), A(1), A(2), A(3)

Complete the following Truth Table.

A0	A1	A2	A3	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0	0	0	1																
0	0	1	0																
0	0	1	1																
0	1	0	0																
0	1	0	1																
0	1	1	0																
0	1	1	1																
1	0	0	0																
1	0	0	1																
1	0	1	0																
1	0	1	1																
1	1	0	0																
1	1	0	1																
1	1	1	0																
1	1	1	1																

Which output goes low for the binary word 1010? .....

How can you wire up the above circuit to give you a 1 of 8 Decoder?

Select a 7447 BCD to 7 segment Decoder.
Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 7447.
+ 5 on pin 16
Ground on pin 8
LT and RB1 and B1/RBO to Hi
LED Monitors on a - g
150 ohm load resistors between a - g and +5
Logic switches A(0), A(1), A(2), A(3)

Complete the following Truth Table.

A3	A2	A1	A0	a	b	c	d	e	f	g
0	0	0	0	0	0	0	0	0	0	1
0	0	0	1							
0	0	1	0							
0	0	1	1							
0	1	0	0							
0	1	0	1							
0	1	1	0							
0	1	1	1							
1	0	0	0							
1	0	0	1							
1	0	1	0							
1	0	1	1							
1	1	0	0							
1	1	0	1							
1	1	1	0							
1	1	1	1							

Why was it necessary to connect 150 ohm resistors from a - g to +5 ?

This integrated circuit is designed to drive a 7 segment readout. Wire up a 7 segment readout and check that it can count from 0 - 9.

Learning Activity H3

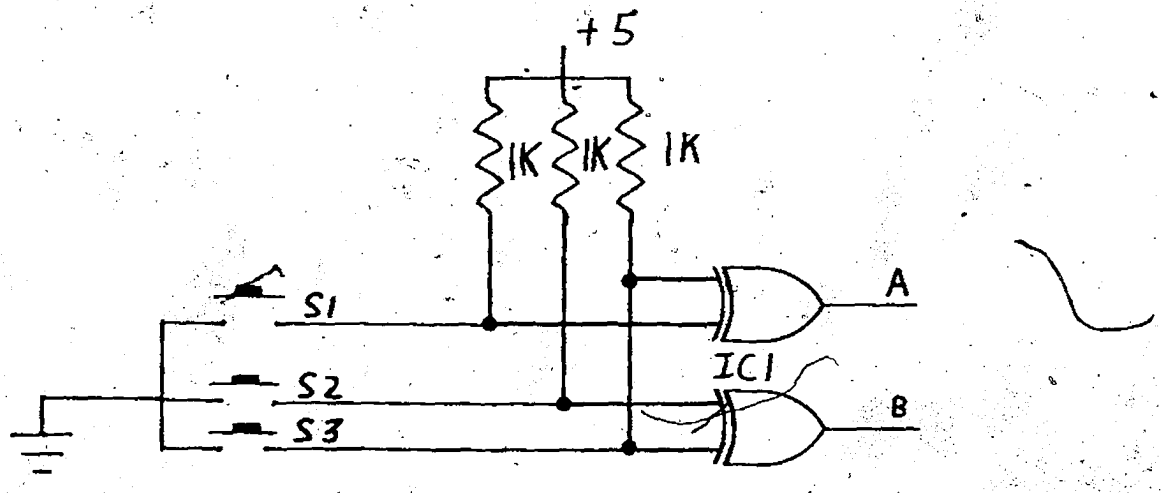
ENCODERS

Objective: To observe and record the operation of an encoder.

An Encoder is a combinational logic circuit that accepts one or more inputs and generates a multi-bit binary output code.

Select a 7486 integrated circuit.
Look up Pin connections in Fairchild TTL Data Book.

Wire the following Encoder.
+5 on pin 14
Ground on pin 7
LED Monitor on A and B



IC1 = 7486
R = 1K
S(1), S(2), S(3) are Push Button switches.

Complete the following Truth Table.

input	output A	output B
S1 closed		
S2 closed		
S3 closed		

Encoders can be used by a computer to determine if a specific switch combination has been closed.

Can you think of any other applications for Encoders?

Learning Activity H4

MULTIPLEXERS

Objective: To observe and record the operation of a Multiplexer.

A Multiplexer is an electronic circuit that is used to select and route any one of a number of input signals to a single output.

Select a 74151 integrated circuit.
Look up Pin connections in the Fairchild TTL Data Book.

Wire up the 74151.
+ 5 on pin 16
Ground on pin 8
Lo on E
Led Monitor on Z
I(0) = 1, I(1) = 0, I(2) = 0, I(3) = 1
I(4) = 0, I(5) = 1, I(6) = 0, I(7) = 0

Complete the following Truth Table.

Input			Output	
S(2)	S(1)	S(0)	Z	
0	0	0		I(0)
0	0	1		I(1)
0	1	0		I(2)
0	1	1		I(3)
1	0	0		I(4)
1	0	1		I(5)
1	1	0		I(6)
1	1	1		I(7)

Did you confirm that input was routed to the output? Explain.

Multiplexers are used to convert parallel data to serial data.

Design a circuit that will cause this to happen? (Consider an 8 Bit Multiplexer, see Heathkit, Digital Techniques, Experiment no. 20).

Learning Activity H5

DEMULTIPLEXERS

Objective: To observe and record the operation of a Demultiplexer.

A Demultiplexer has a single input and multiple outputs. A Demultiplexer is also called a data router.

Select a 74154 integrated circuit.
Look up Pin connections in Fairchild TTL Data Book.

+ 5 on Pin 24
Ground on Pin 12
LED Monitors on Q(0) - Q(15)
Logic switch on A(0) - A(3)
Lo on E(0) and E(1)

By selecting address inputs A(0) - A(3), a Lo applied to the enable inputs can be routed to any one of the outputs.

Complete the Truth Table.

Address				Outputs															
A0	A1	A2	A3	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15
0	0	0	0																
0	0	0	1																
0	0	1	0																
0	0	1	1																
0	1	0	0																
0	1	0	1																
0	1	1	0																
0	1	1	1																
1	0	0	0																
1	0	0	1																
1	0	1	0																
1	0	1	1																
1	1	0	0																
1	1	0	1																
1	1	1	0																
1	1	1	1																

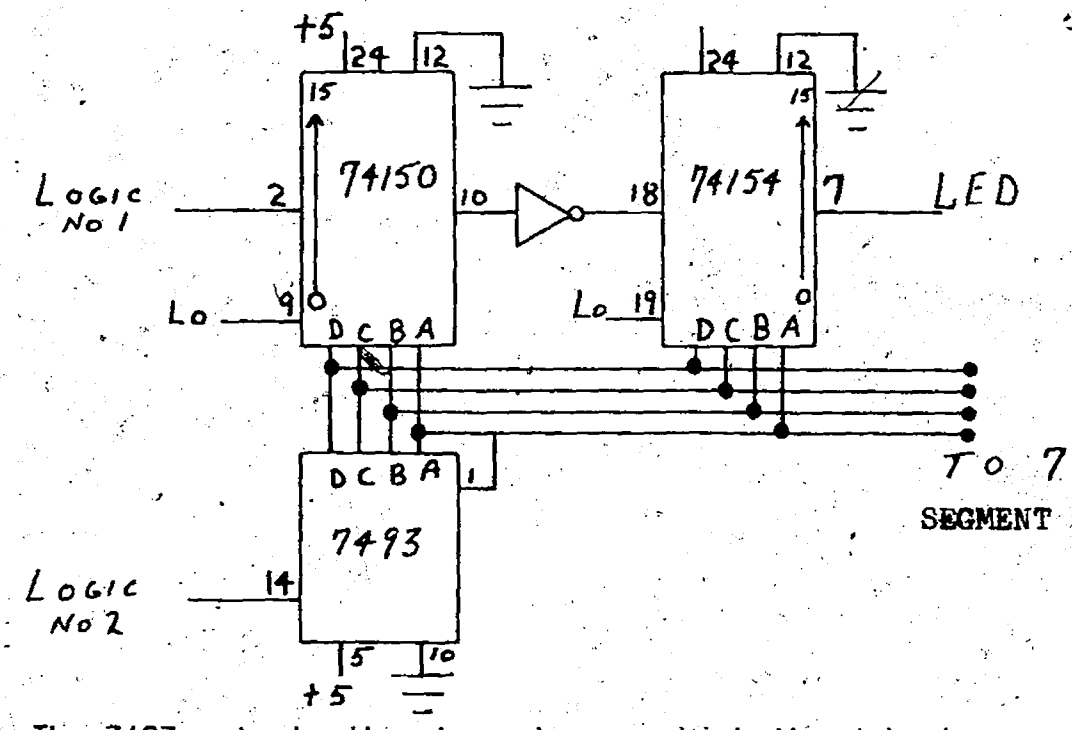
Learning Activity H6

MULTIPLEXER DEMULTIPLEXER

Objective: To construct and test a 1-of-16 Multiplexer and a 1-of-16 Demultiplexer.

Select 74150, 74154, 7493, 7404 integrated circuits. Look up Pin connections in the Fairchild TTL Data Book.

Wire up the following circuit.
+ 5 on pin 24, 74150 and pin 24, 74154 and pin 5, 7493 and pin 14, 7404.
Ground on pin 12, 74150 and pin 12, 74154 and pin 10, 7493 and pin 7, 7404.



The 7493 selects the channel over which the data is transmitted. Select channel 6 by pulsing logic switch no.2. Logic switch 1 inputs data.

LED reads out data. Make sure data is being transmitted and received. Pulse logic switch 2 and change to channel 1.

Locate input and output channel.

Input pin ..... on IC .....

Output pin ..... on IC .....

Can you input data on any input channel and output it on the same channel? Show your teacher.

Conclusion: If this is true, then one could transport 16 channels with only 5 wires.

Learning Activity H7

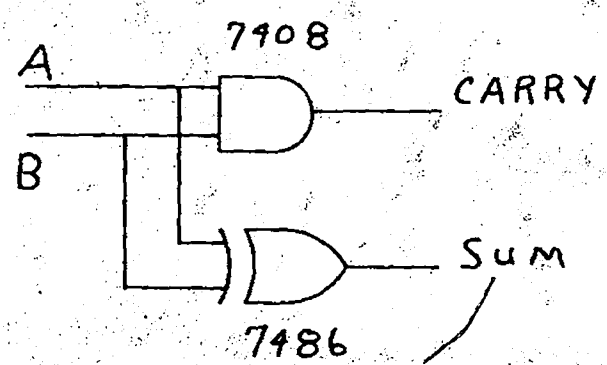
HALF ADDER

Objective: To construct and observe the operation of a Half Adder.

A Half Adder is a binary adder that adds two binary bits.

Select two integrated circuits; no. 7486 and no. 7408. Look up Pin connections in Fairchild TTL Data Book.

Wire up the following circuit:
+ 5 on pin 14, 7486 and pin 14, 7408
Ground on pin 7, 7486 and pin 7, 7408
Logic switch on A and B
LED Monitor on Sum and Carry



Complete the following Truth Table.

Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1		
1	0		
1	1		

Write the Boolean equation for Sum and Carry.

.....

Learning Activity H8

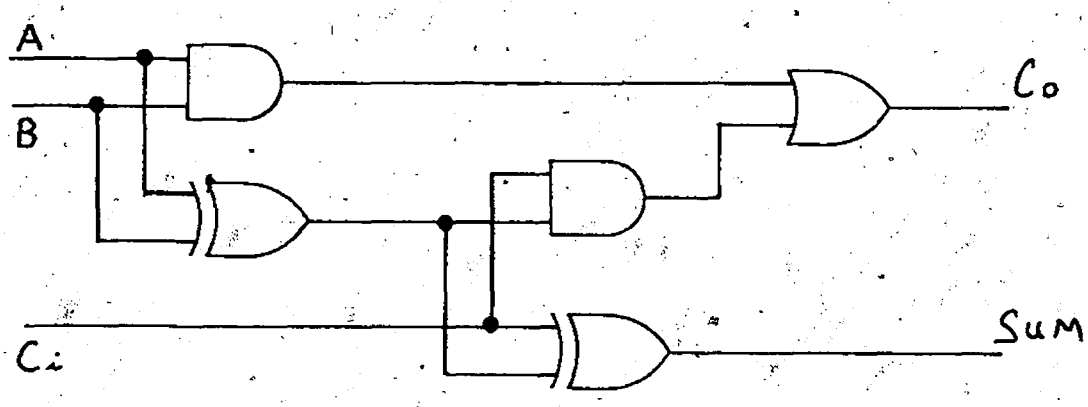
FULL ADDER

Objective: To construct and observe the operation of a Full Adder.

A Full Adder is a binary adder that adds two binary bits and the carry from a previous addition.

Select three integrated circuits; No. 7486, no. 7408 and no. 7432.
Look up Pin connections in Fairchild TTL Data Book.

Wire up the following circuit:
+ 5 on pin 14, 7486 and pin 14, 7408 and pin 14, 7432
Ground on pin 7, 7486 and pin 7, 7408 and pin 7, 7432
Logic switches on A, B and C
LED Monitor on Sum and Carry



Complete the following Truth Table.

Input			Output	
A	B	C	Sum	Carry
0	0	0	0	0
0	1	0		
1	0	0		
1	1	0		
1	1	1		

Write the boolean equation for Sum and Carry.

.....

Activity: Design a 4 Bit Adder. (That is an Adder that will add 4 Bit words with carry out).

Learning Activity #9

PARITY GENERATOR

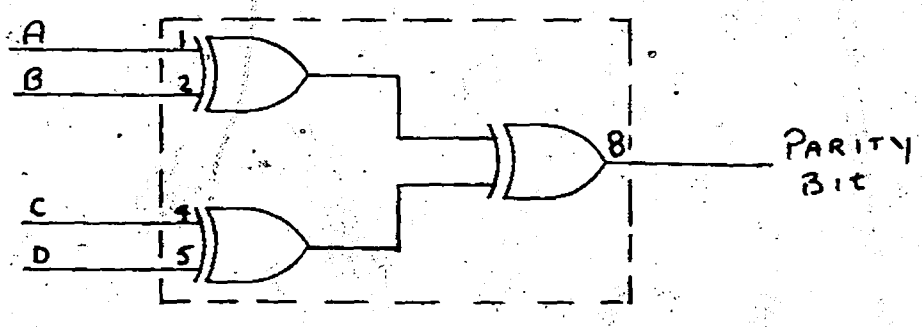
Objective: To construct and record the operation of a 4 Bit Parity Generator.

A Parity Generator is a combinational logic circuit that observes the 1 bits in the word and generates the appropriate parity bit. For even parity, the number of 1 bits in the word plus the parity bit, will be even.

Select integrated circuit no. 7486.
Look up Pin connections in Fairchild TTL Data Book.

Wire up the following circuit:
+ 5 on pin 14
Ground on pin 7
Logic switches on 1, 2, 4, 5
LED Monitor on Parity Bit output

4 Bit Even Parity Generator



Complete the following Truth Table.

Input				Output
A	B	C	D	Parity Bit
0	0	0	0	0
0	0	0	1	1
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	
1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

For odd Parity, the number of 1 bits in the word plus the Parity bit, will be odd.

Design and test an odd Parity bit generator.

Learning Activity H10

Parity Checker

Objective: To observe and record the operation of an 8 bit parity checker.

A parity checker checks for the number of 1 bits in the word, generates a parity bit and compares the generated bit with the word parity bit. If the word parity bit is incorrect a Hi is generated at the output.

Select a 74180 integrated circuit, parity generator checker.
Look up pin connections in Fairchild TTL Data Book.

Wire up the 74180
+ 5 on pin 14
Ground on pin 7
Logic switches on I(0) - I(7)
Logic switches on E(I) and E(0)
Note: E(0) must be the inverse of E(I)
LED Monitor on Σ(E) to check parity bit for even parity
A Hi on Σ(E) means the even parity bit is incorrect

Complete the Table

Inputs										Output
E(I)	O(I)	I(0)	I(1)	I(2)	I(3)	I(4)	I(5)	I(6)	I(7)	Σ(E)
0	1	1	0	1	0	1	0	1	0	
1	0	1	0	1	0	1	0	1	0	
0	1	1	1	1	1	1	1	1	1	
0	1	0	0	0	0	0	0	0	0	

Identify those combinations that have incorrect parity bits.

Show your teacher that you can set up the parity bit for a word different from those above.

Learning Activity H11

FOUR-BIT ARITHMETIC LOGIC UNIT

Objective: To use an Arithmetic Logic Unit to perform binary addition and subtraction.

The 74181 integrated circuit, 4 Bit Arithmetic Logic Unit provides: 16 Arithmetic operations
16 Logic operations on two variables

Select integrated circuit no. 74181.
Look up Pin connections in Fairchild TTL Data Book.

Wire up the 74181
+ 5 on pin 24
Ground on pin 12
For addition:
Logic switches on A(0) - A(3) and B(0) - B(3)
S(0) - Hi
S(1) - Lo
S(2) - Lo
S(3) - Hi
M = Lo
C(n) = Hi
LED Monitor F(0) - F(3)
LED Monitor C(n + 4)

Complete the truth table for a 4 bit binary adder

A3	A2	A1	A0	B3	B2	B1	B0	F3	F2	F1	F0	C(n+4)
0	1	1	0	0	1	1	0					
1	0	1	0	0	1	1	1					
0	1	1	1	1	0	0	1					

Convert the binary input and output to decimal to check the accuracy of the 4 bit adder

For Subtraction:

S(0) - Lo
S(1) - Hi
S(2) - Hi
S(3) - Lo
C(n) = Hi
All others as above.

Complete the following truth table for
A minus B minus 1

A3	A2	A1	A0	B3	B2	B1	B0	F3	F2	F1	F0	C(n+4)
0	1	1	1	0	1	0	1					
1	0	0	0	0	0	1	0					
1	0	1	0	0	0	1	0					

Convert the binary input and output to decimal to check your answer. Remember you are subtracting A-B-1.

You may wish to investigate some of the other 30 operations. See TTL Logic book for information.

Parallel two 74181 integrated circuits to add two 8 Bit words with carry.

Learning Activity H12

Construction Job

Objective: To construct a decoder and display for a decade counter.

To construct an interface between the TRS-80 computer and a Centronics printer with parallel interface.

Design and construct a circuit that will decode data from a decade counter and display it in decimal form from 0 to 99. This circuit will interface with the decade counter you constructed in Learning Activity G15.

Design and construct a circuit that will interface the TRS-80 computer to a Centronics printer with parallel interface. For assistance see Microcomputing January 1980.

III Computer Architecture

All computers, whether micro or maxi, will require certain basic elements. The important thing is to learn how to identify these elements in any computer you will use. Once this is done you can analyze variations on the basic theme. To fail to learn these basic elements will leave students vulnerable to those "new and improved" computers which always seem to be coming along. As with all engineering activities, there are a few truly basic ideas. Most build on existing activities. Learn the basics and the details will take care of themselves.

This chapter is presented in two parts:

A Introduction to the Microcomputer

B A Real Computer

A Introduction to the Microcomputer

1 Microcomputer Architecture

2 Basic Computer Operation

3 Fetch/Execute of a Computer Program

4 Fetch/Execute Using Zero Page Addressing

5 A Symbolic method to Illustrate Microcomputer Operation

Learning Activity A1

Microcomputer Architecture

The purpose of this unit is to introduce the student to the computer via the microprocessor. Therefore all references to a computer are essentially references to microcomputers.

The microcomputer has three essential components:

1 The Microprocessor Unit (MPU), also called the Central Processor Unit (CPU). The MPU is considered to be the heart of the computer.

2 Memory, either Random Access Memory (RAM) or Read Only Memory (ROM).

3 Input/Output (I/O), provides a means of connecting the computer to the outside world.

A microcomputer is similar to a mini or mainframe computer, it does similar things but much slower. Essentially it receives data from the outside world via the input port, processes this data, and delivers the resultant data to an output port.

Figure 1 is a block diagram of a Microcomputer.

Basic computer operation

The microcomputer uses the stored program concept. That is a program is stored in sequential memory locations and the MPU selects the address that contains the program data. This is done by sending pulses out the address bus and control lines to a specific address. The contents of this address are output to the MPU via the data bus. This data is then processed and the MPU selects a new address for data.

Figure 2 will be used to explain the operation of an elementary microprocessor. This is a pseudo microprocessor, however, the ideas learned here are sufficient for you to understand the operation of most microprocessors. This is an eight bit microprocessor, data words are eight bits wide and each memory location holds eight bits of data.

Block Diagram of a Microcomputer

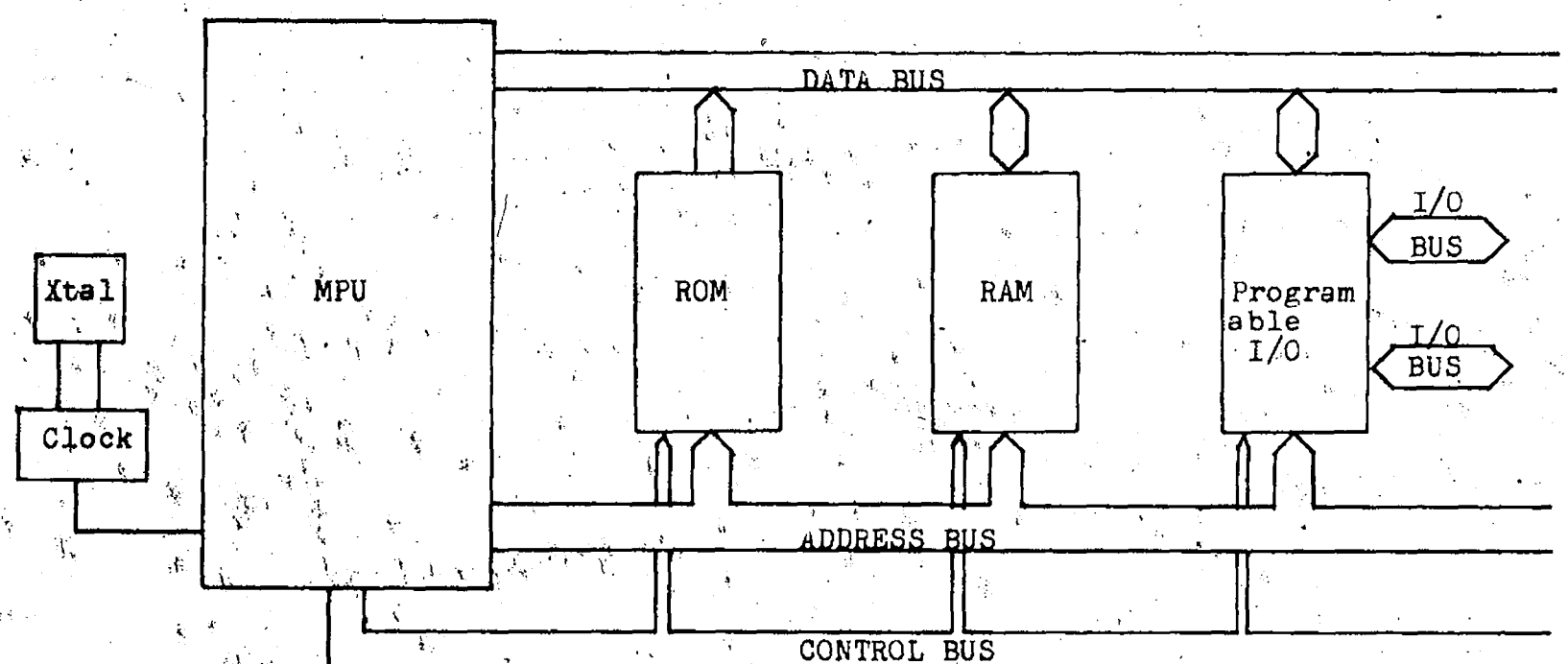


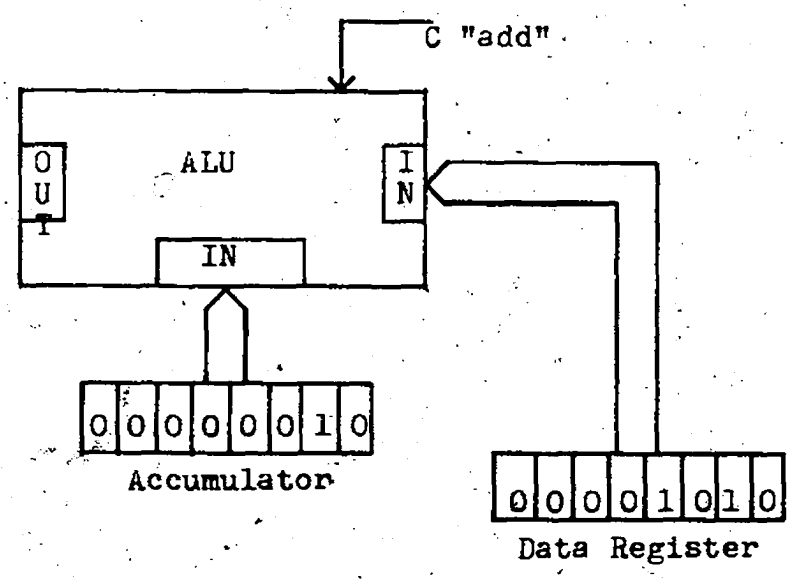
Figure 1

Following is a list of terms, including definitions, that are used with this microprocessor:

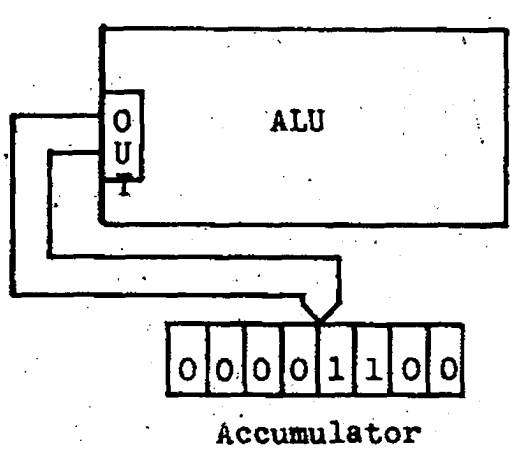
Accumulator: A special purpose register associated with the Arithmetic and Logic Unit (ALU), which temporarily stores sums and other arithmetical and logical resuts of the ALU.

Arithmetic Logic Unit (ALU): One of the three essential components of a microprocessor. The other two are the registers and the control block. The ALU performs various forms of additions, subtraction and logic operations, such as ANDing the contents of two registers and masking the contents of a register.

Operation of an ALU utilizing the ADD instruction.



The control lines tell the ALU to add the two inputs. The result is stored in the accumulator.



Address Bus: Set of wires (typically 16) used to transmit an address from the microprocessor to memory or I/O device.

Address Register: Temporary storage register, it holds the address of the memory location presently under access.

Condition Code Register: This register consists of a series of flag bits whose settings reflect the state of the MPU after an operation has been performed.

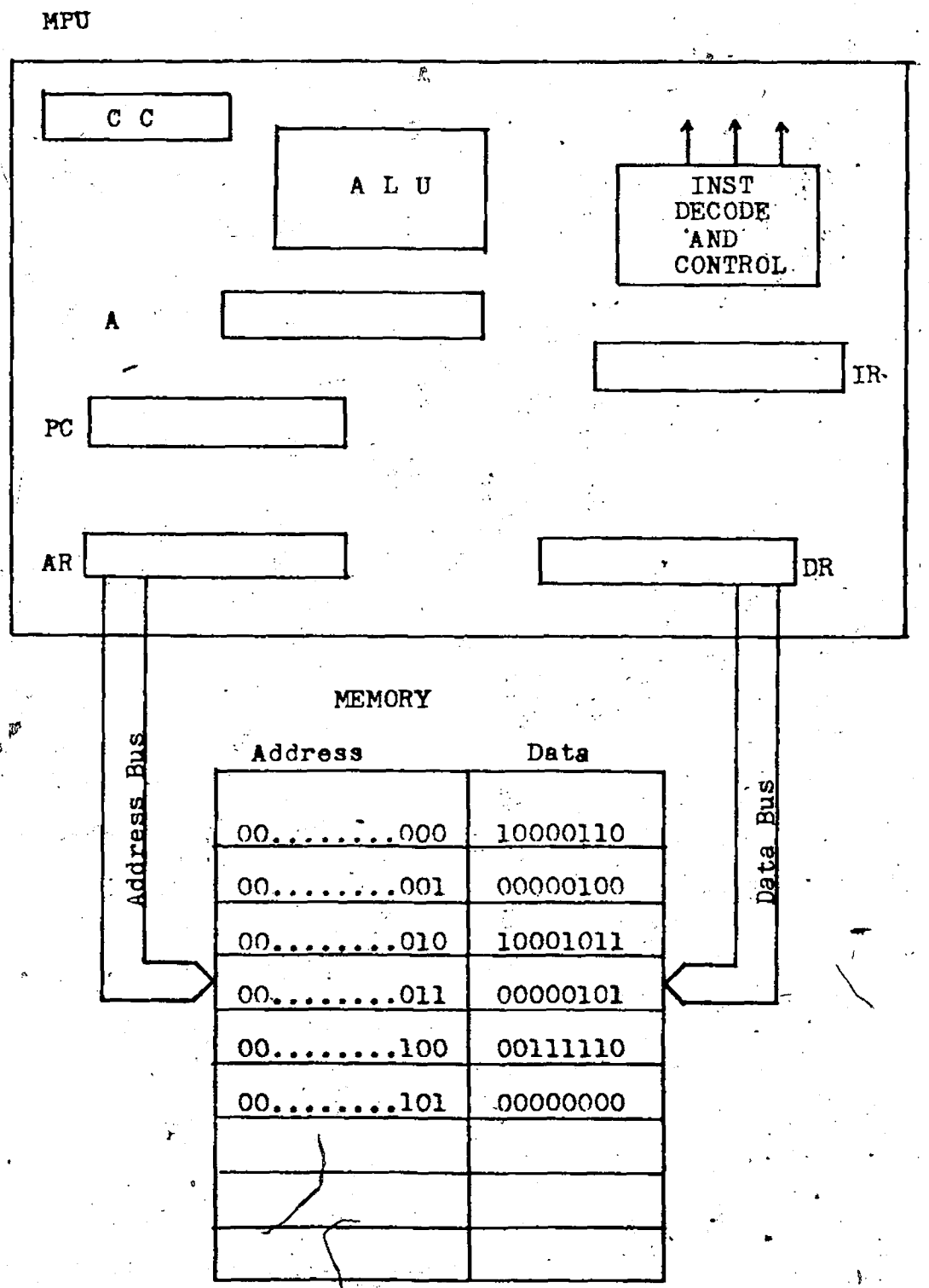
Data Bus: Set of lines carrying data. The data bus is usually bidirectional and tri-state.

Data Register: Temporary storage register for data going to or coming from the data bus.

Instruction decode and control: As it's name implies it decodes instructions and generates control signals for the MPU and external devices such as memory and I/O.

Instruction Register: Contains the opcode for the instruction being executed.

Program Counter: Register which contains the address of the next instruction to be executed.



A Accumulator
ALU Arithmetic Logic Unit
AR Address Register
CC Condition Code Register
DR Data Register
IR Instruction Register
PC Program Counter
Instruction Decode and Control

Figure 2

Learning Activity A3

Fetch/Execute of a computer program

A microcomputer sequences through a program by following a series of fetch, execute operations.

During the fetch phase an instruction is read from memory and decoded by the MPU. During the execute phase the instruction operations are carried out. This sequence is repeated for the next set of instructions etc.

The following are examples of MPU instructions, LDA, ADD, HLT, with a description of each:

	Mnemonic code	Operation code
Load the accumulator with the contents of the next memory location	LDA	10000110
Add the contents of the accumulator to the contents of the next memory location	ADD	10001011
Halt, stop the program	HLT	00111110

(Mnemonic code, operation code see glossary)

A program could look like this.

Mnemonic code	Number	Operation code	Operand
LDA	04	10000110	00000100
ADD	05	10001011	00000101
HLT		00111110	

(Operand see glossary)

This program is loaded into sequential memory locations starting at memory location 0

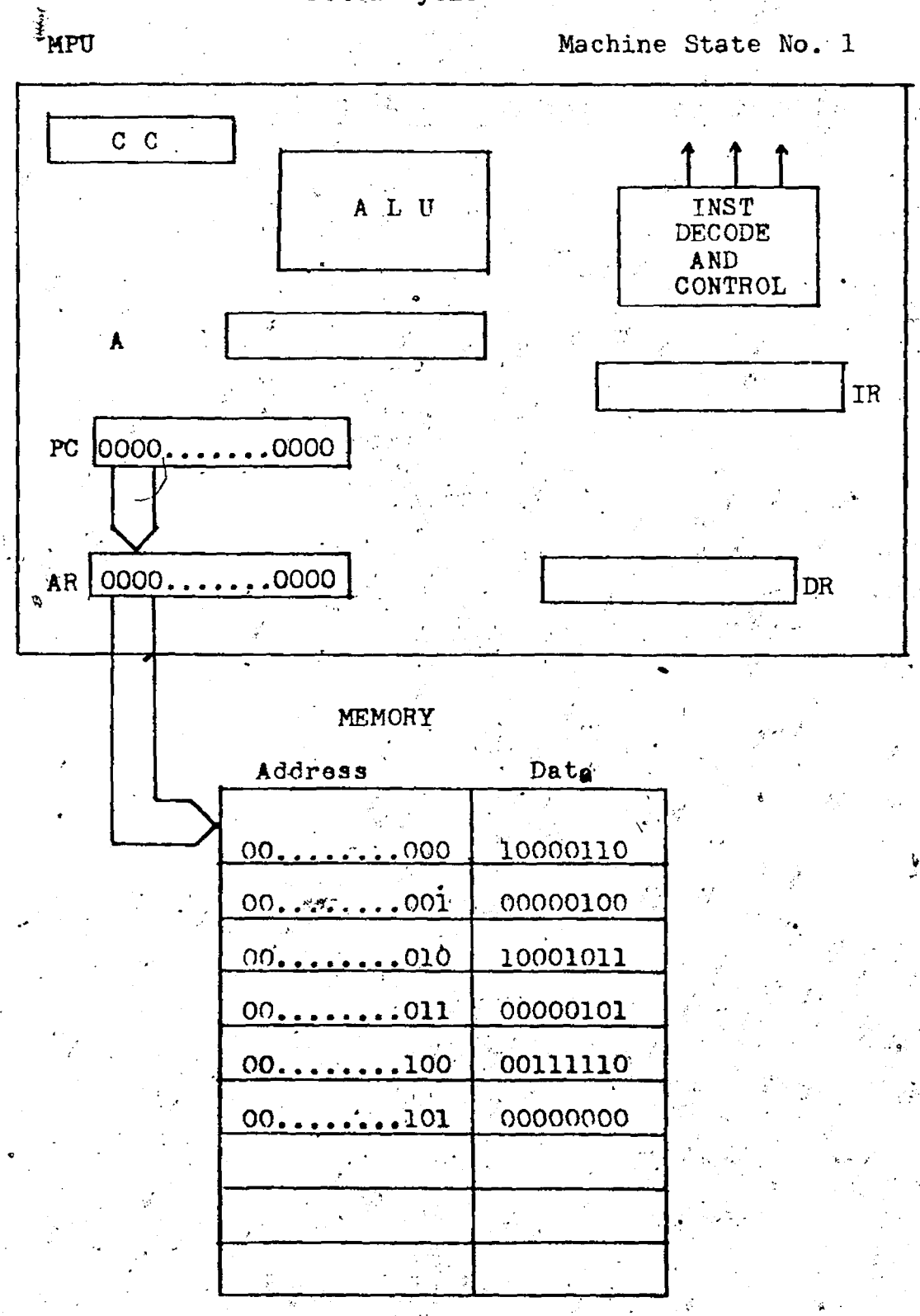
Address 16 Bits	Data 8 Bits
0000000000000000	10000110
0000000000000001	00000100
0000000000000010	10001011
0000000000000011	00000101
0000000000000100	00111110

To run this program the program counter is set to zero and the MPU sequences through the program.

The following pages are a series of illustrations that show MPU operation as the MPU sequences through a program.

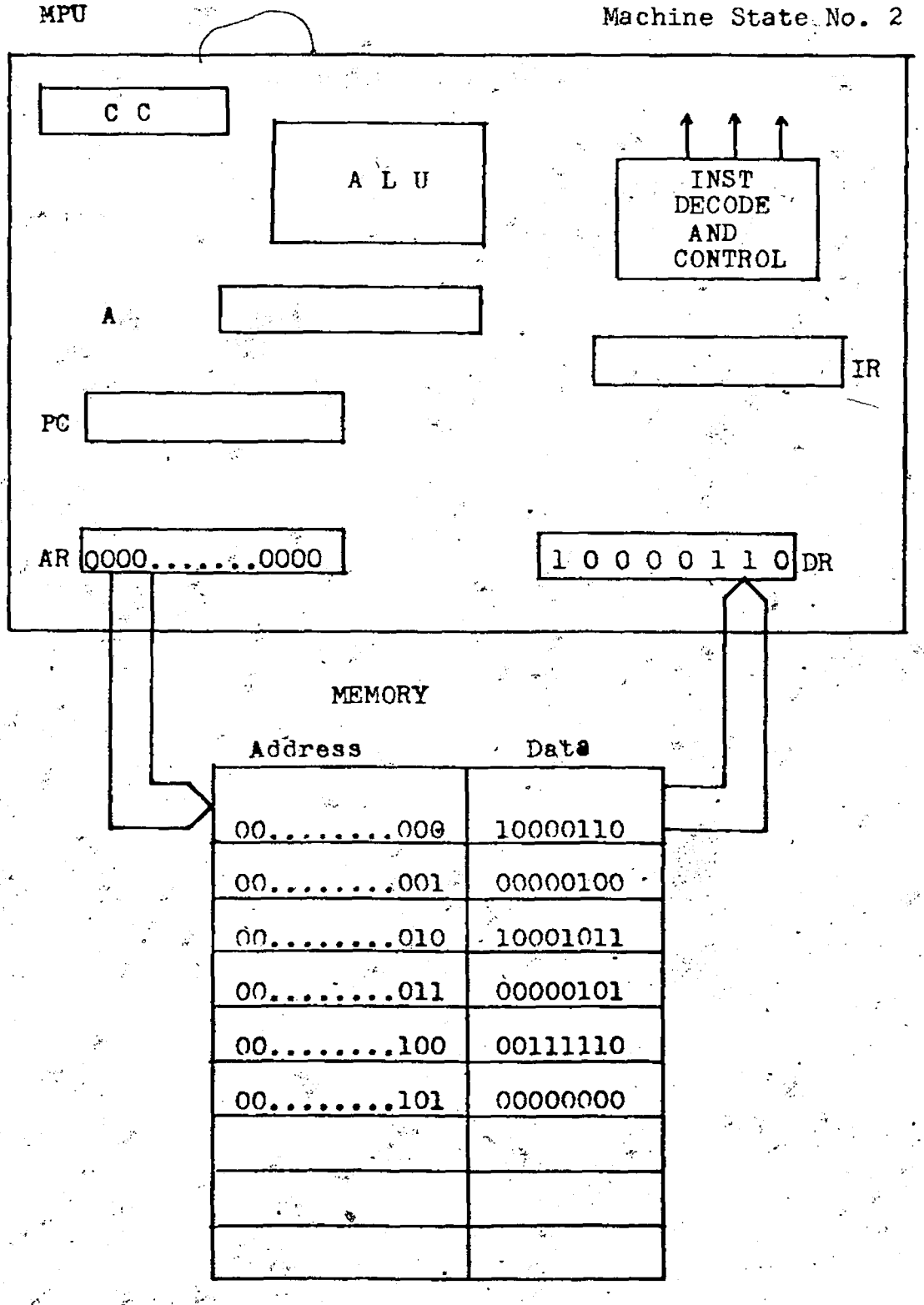
Run the Program

Fetch Cycle



The contents of the program counter are loaded into the address register and placed on the address bus.

Fetch Cycle

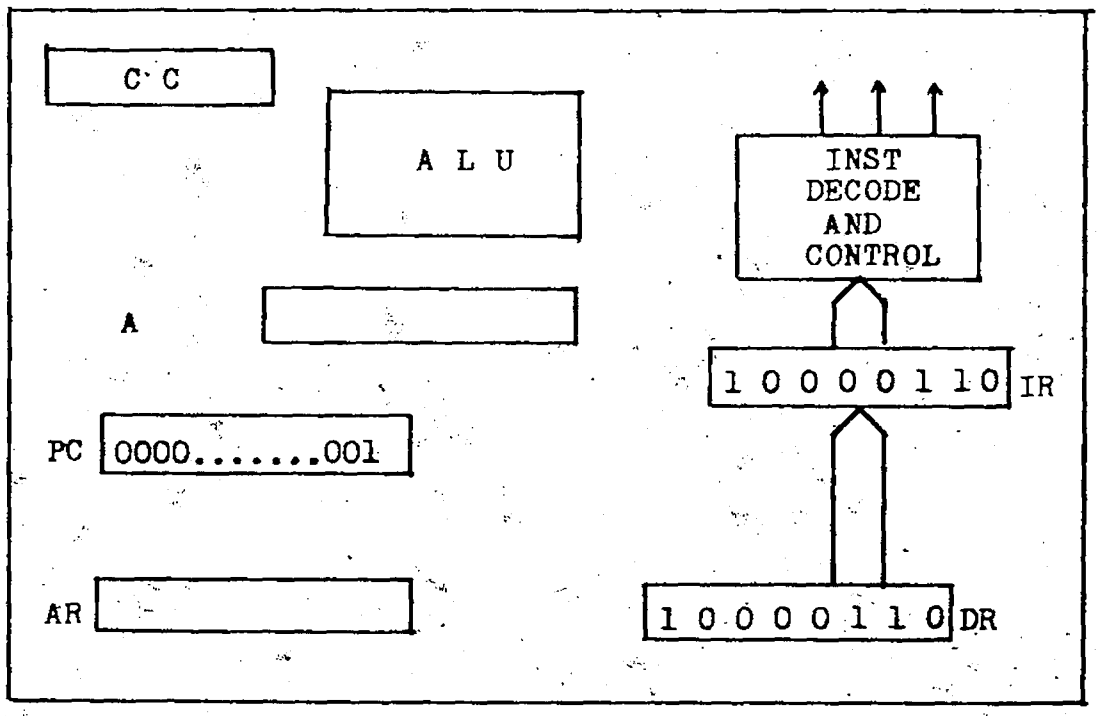


The contents (instruction) of the first address are placed on the data bus and read into the data register

Fetch Cycle

MPU

Machine state No. 3



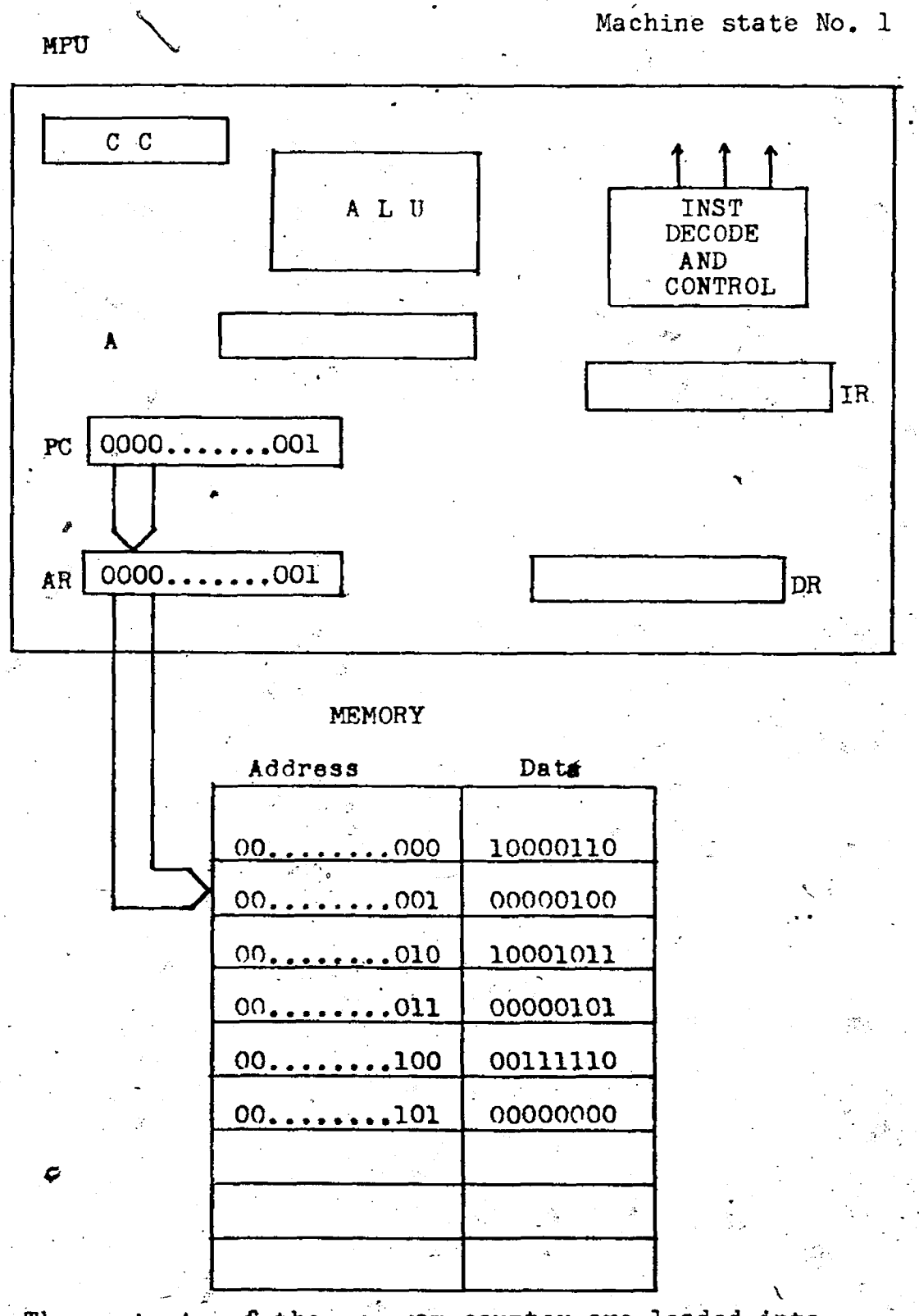
MEMORY

Address	Data
00.....000	10000110
00.....001	00000100
00.....010	10001011
00.....011	00000101
00.....100	00111110
00.....101	00000000

The instruction in the data register is transfered to the instruction register for instruction decode and execution
The program counter is incremented by 1
This completes the fetch cycle.

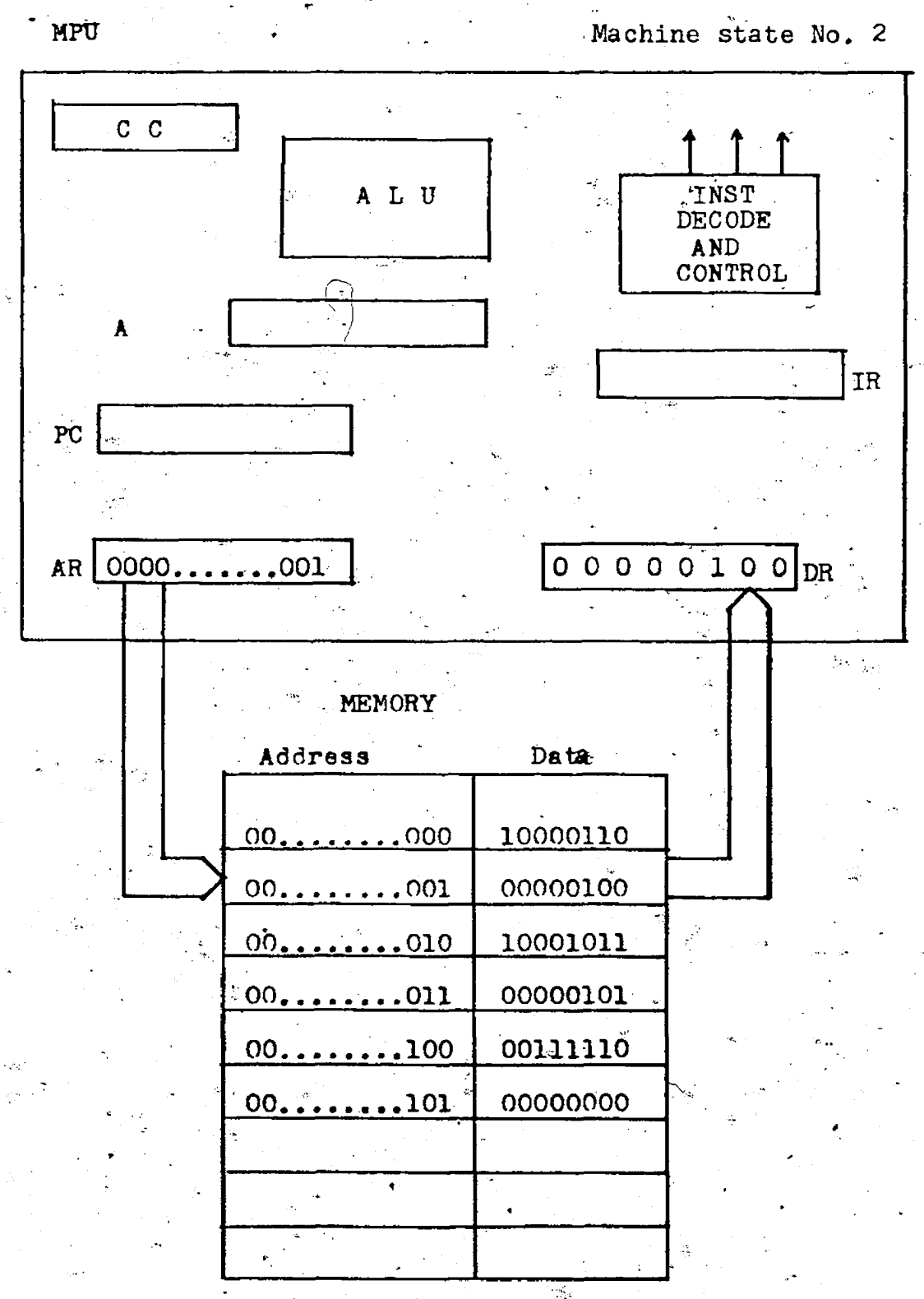
Execute Cycle

Machine state No. 1



The contents of the program counter are loaded into the address register and then placed on the address bus

Execute Cycle

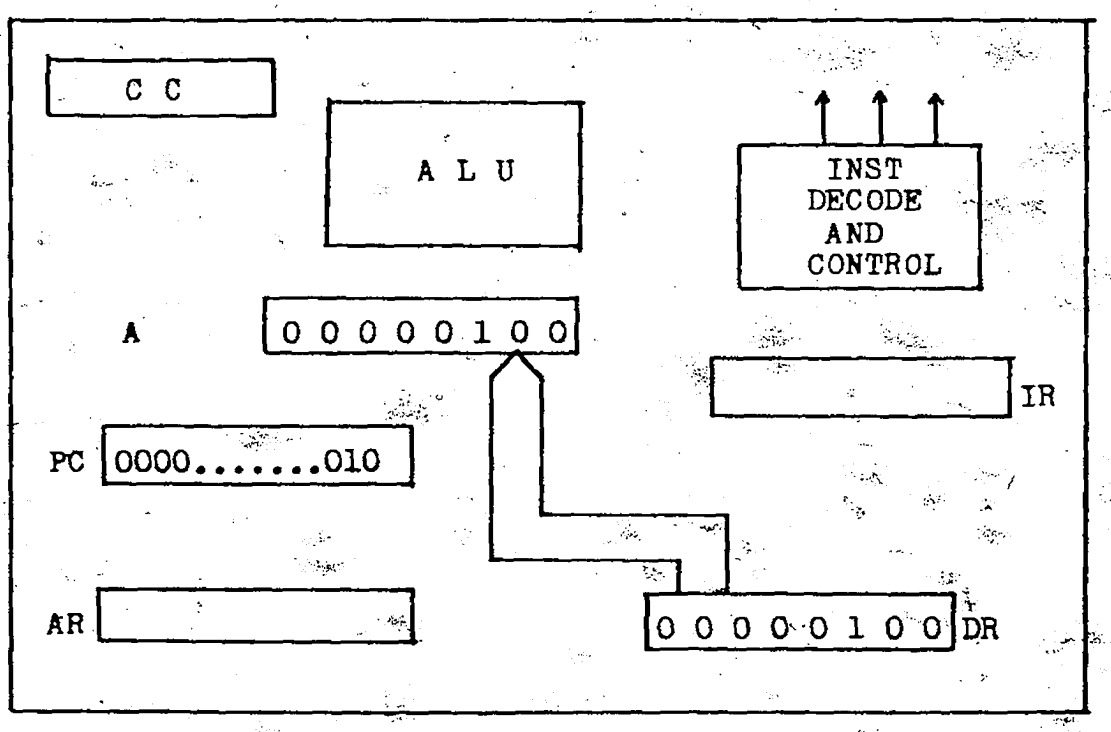


The contents (operand) of the selected address are placed on the data bus and read into the data register

Execute Cycle

MPU

Machine state No. 3



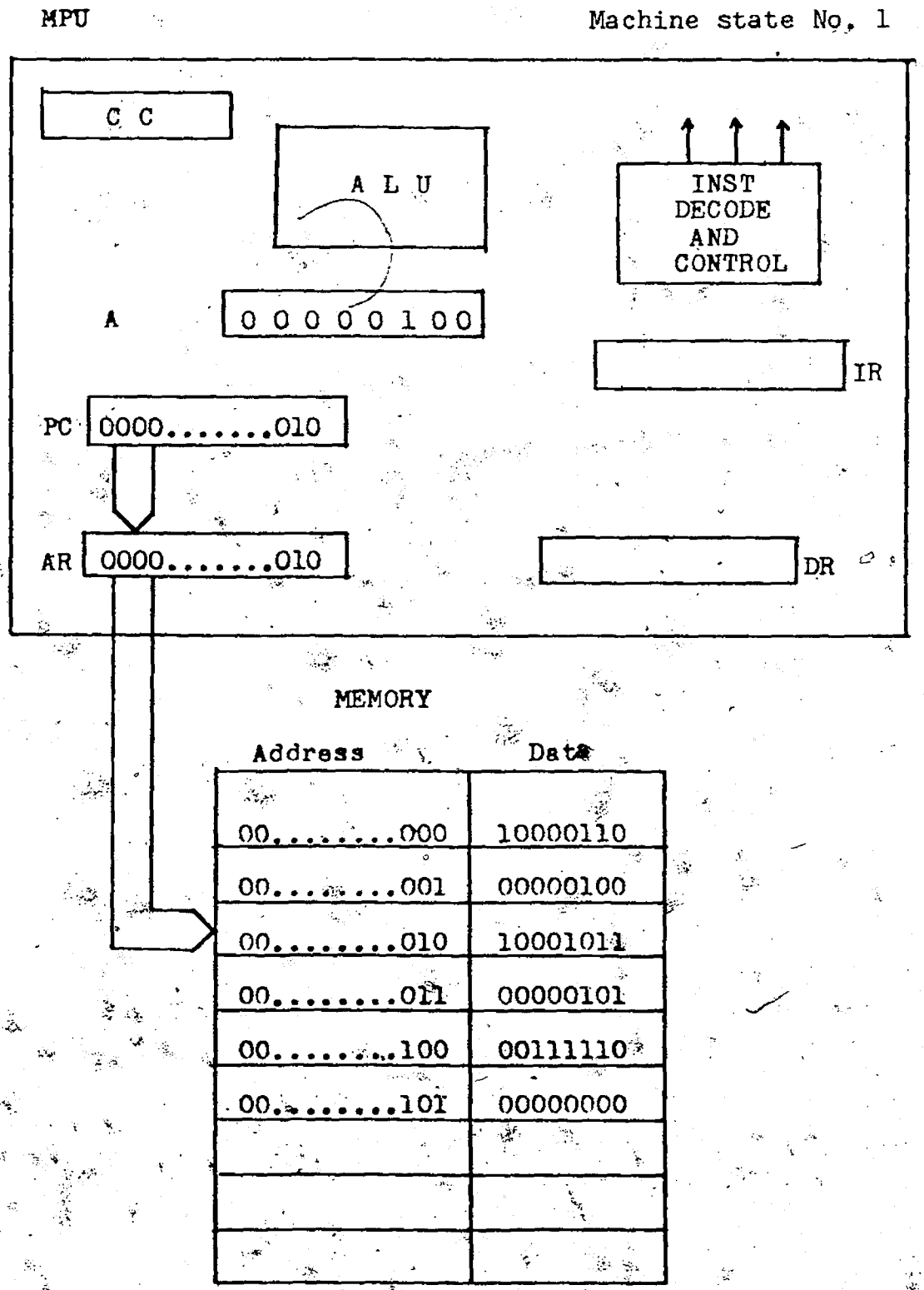
MEMORY

Address	Data
00.....000	10000110
00.....001	00000100
00.....010	10001011
00.....011	00000101
00.....100	00111110
00.....101	00000000

The contents (operand) of the data register are transferred to the accumulator.

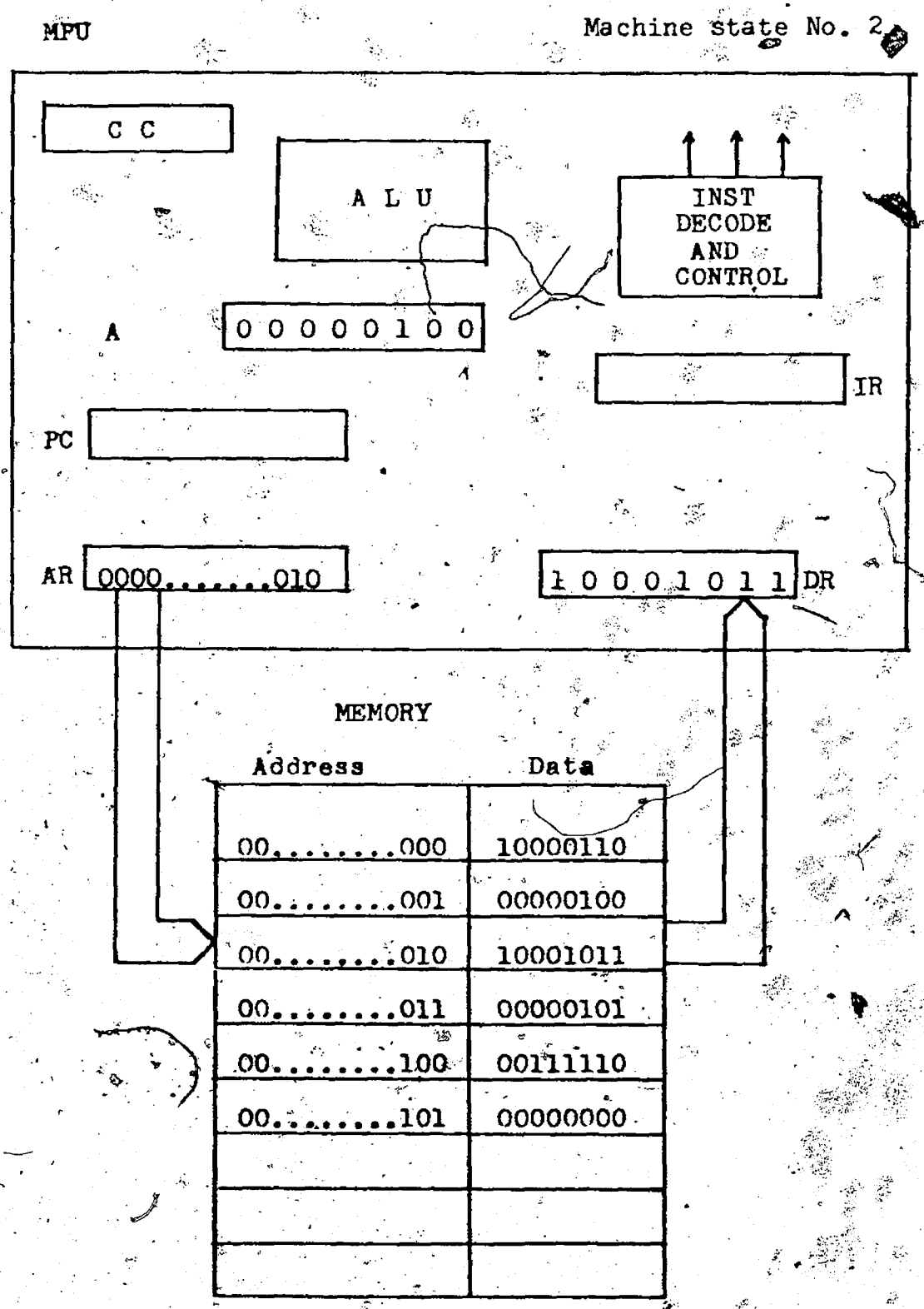
The program counter is incremented by 1.

Fetch Cycle



The contents of the program counter are loaded into the address register and then placed on the address bus

Fetch Cycle

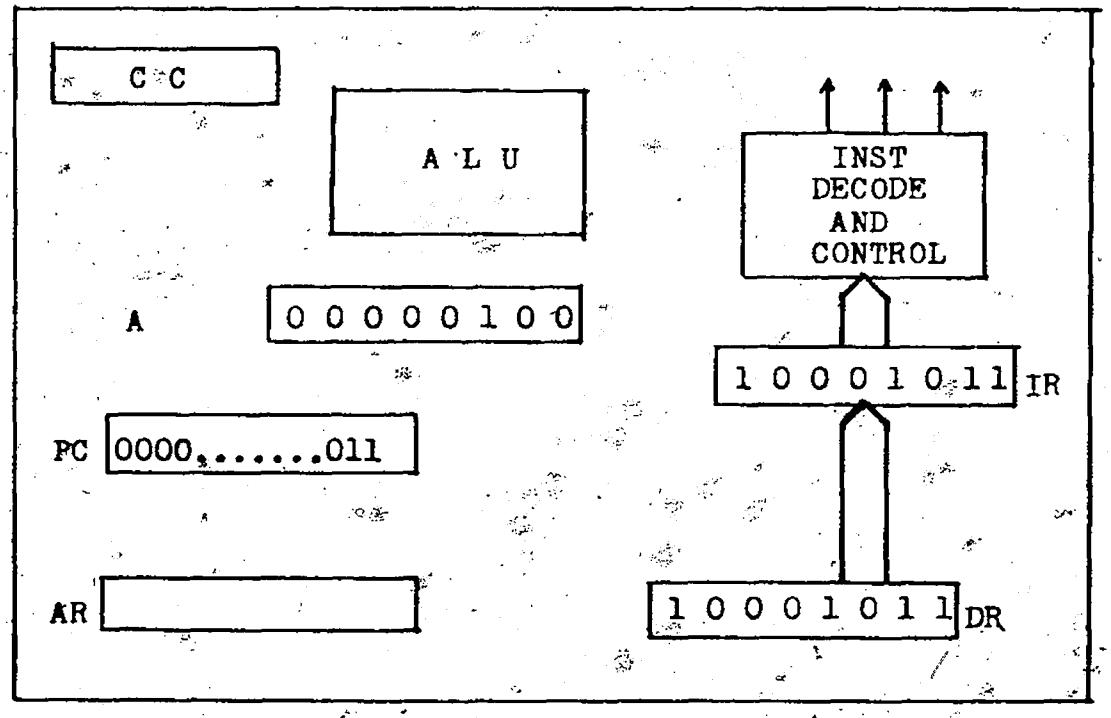


The contents (instruction) of the selected address is placed on the data bus and read into the data register

Fetch Cycle

MPU

Machine state No. 3

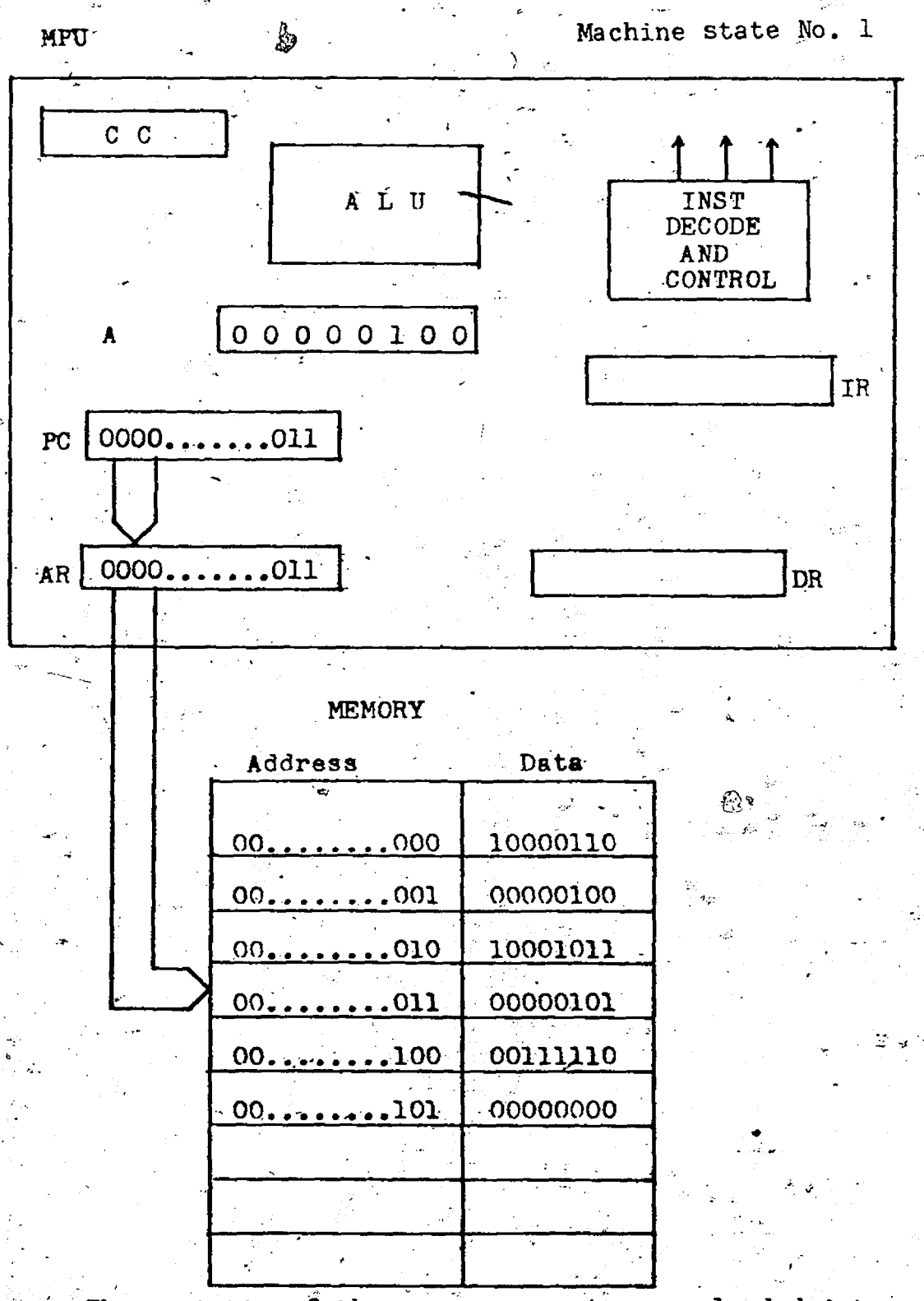


MEMORY

Address	Data
00.....000	10000110
00.....001	00000100
00.....010	10001011
00.....011	00000101
00.....100	00111110
00.....101	00000000

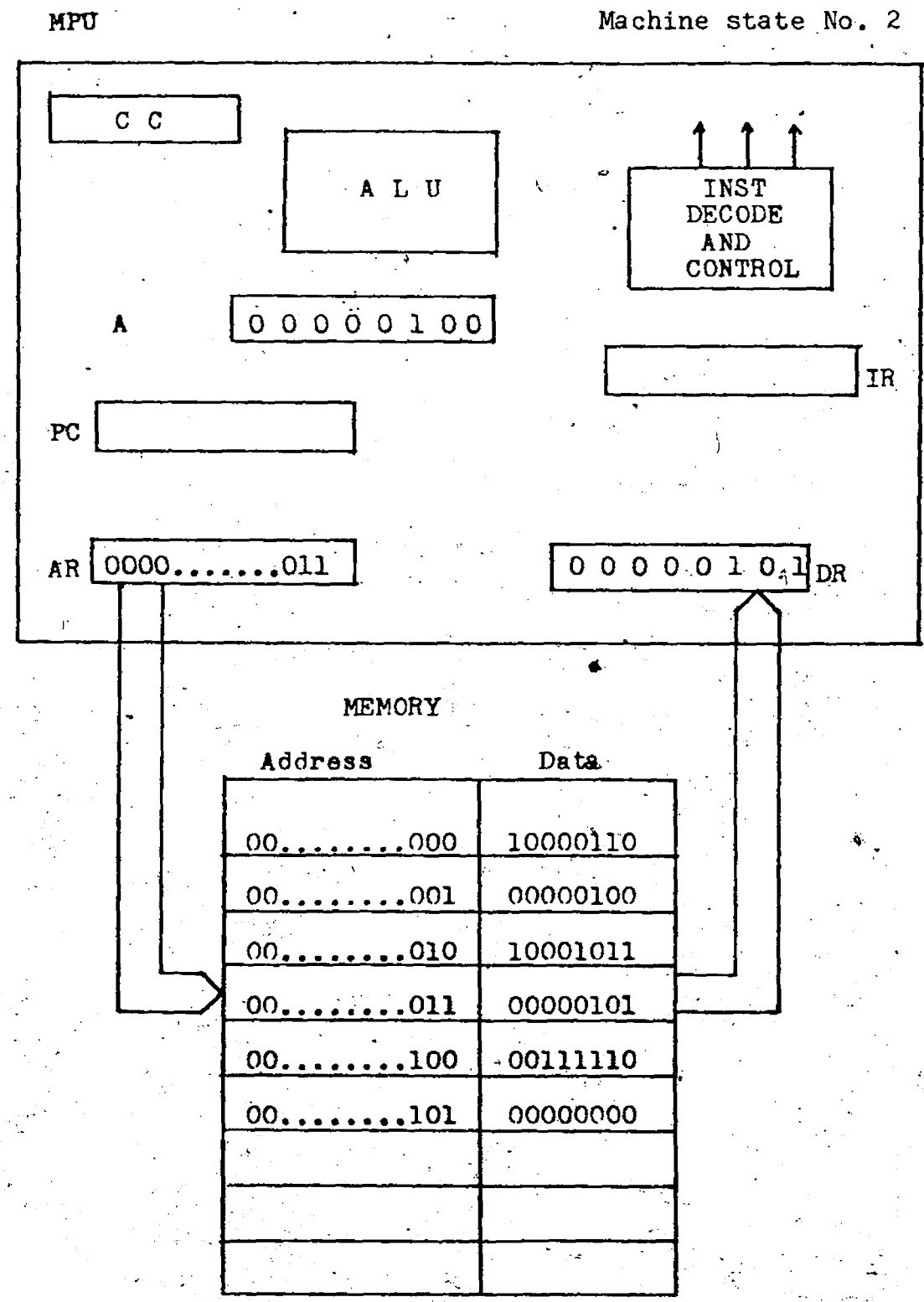
The instruction in the data register is transferred to the instruction register for instruction decode and control. The program counter is incremented by 1. This completes the fetch cycle.

Execute Cycle



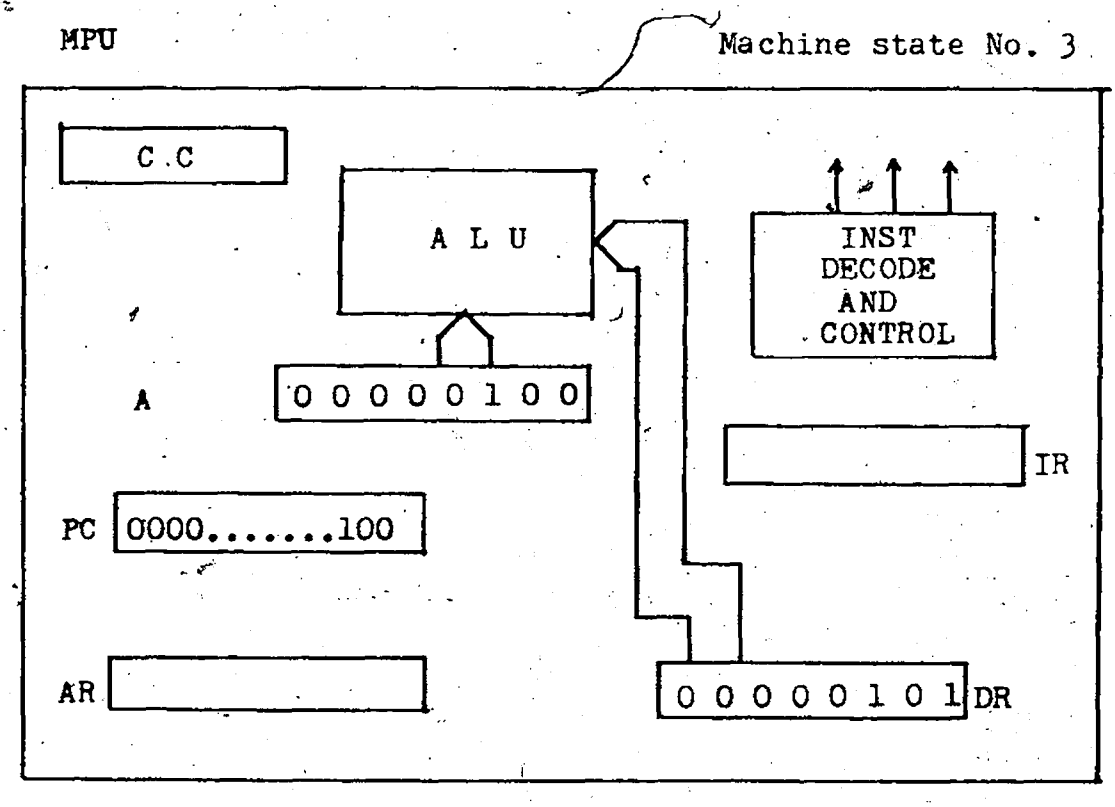
The contents of the program counter are loaded into the address register and then placed on the address bus.

Execute Cycle



The contents (operand) of the selected address are placed on the data bus and read into the data register.

Execute Cycle

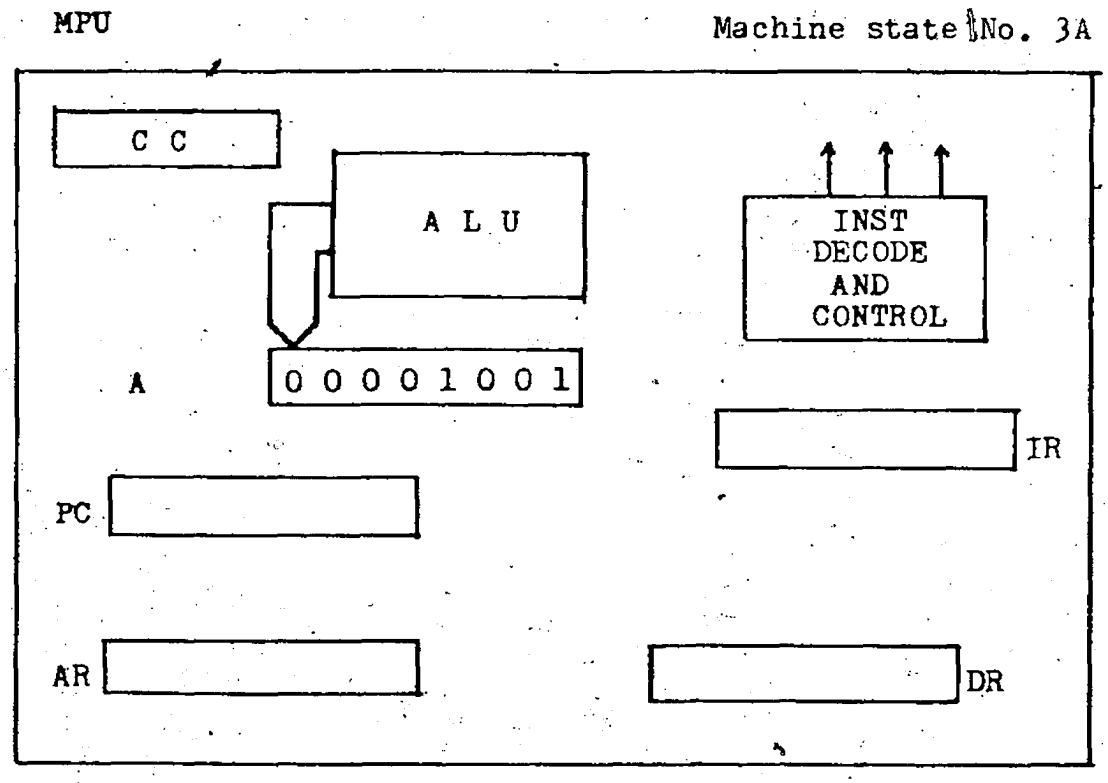


MEMORY

Address	Data
00.....000	10000110
00.....001	00000100
00.....010	10001011
00.....011	00000101
00.....100	00111110
00.....101	00000000

The contents of the data register are loaded into the ALU
The contents of the accumulator are loaded into the ALU
The program counter is incremented.

Execute Cycle

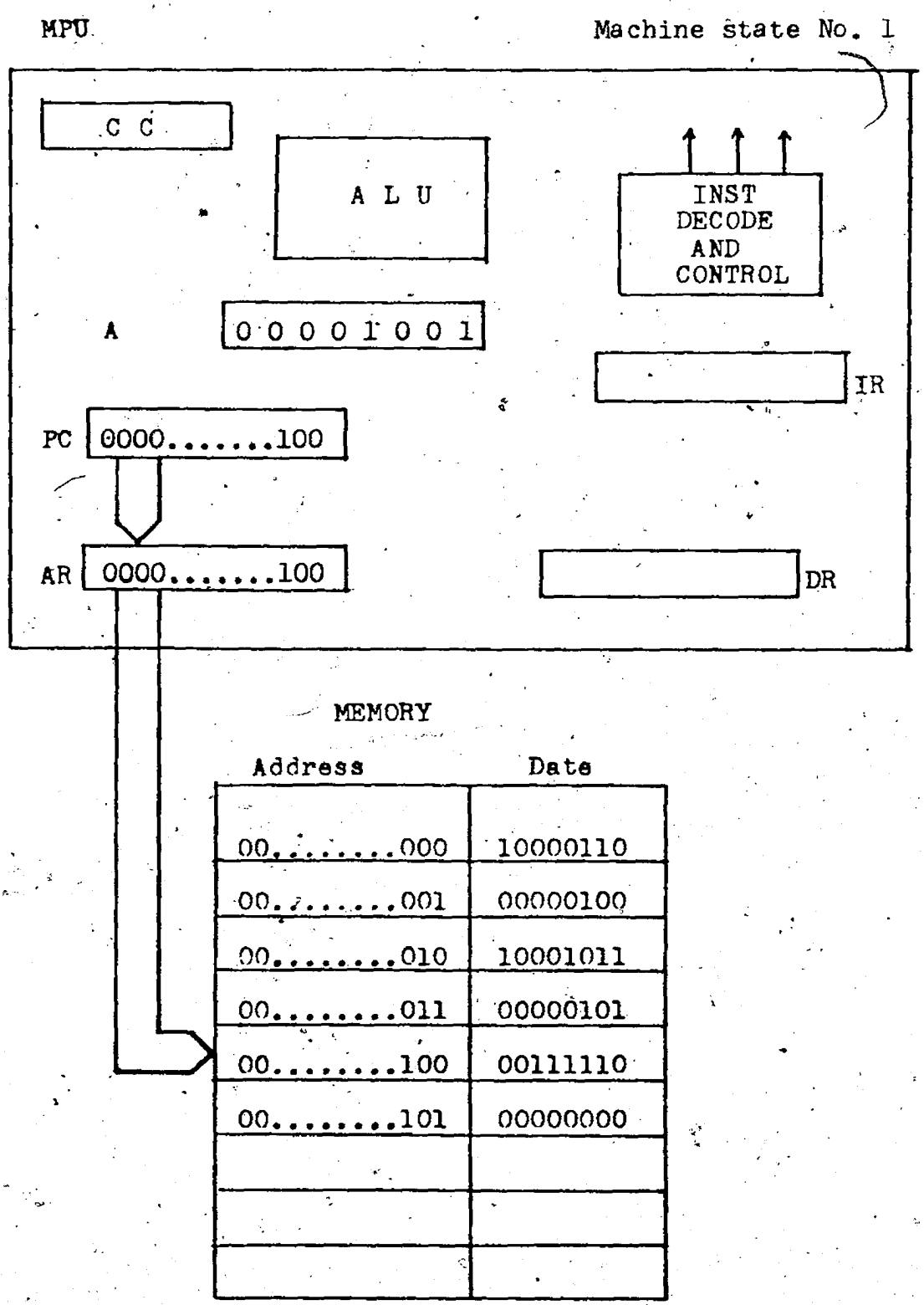


MEMORY

Address	Data
00.....000	10000110
00.....001	00000100
00.....010	10001011
00.....011	00000101
00.....100	00111110
00.....101	00000000

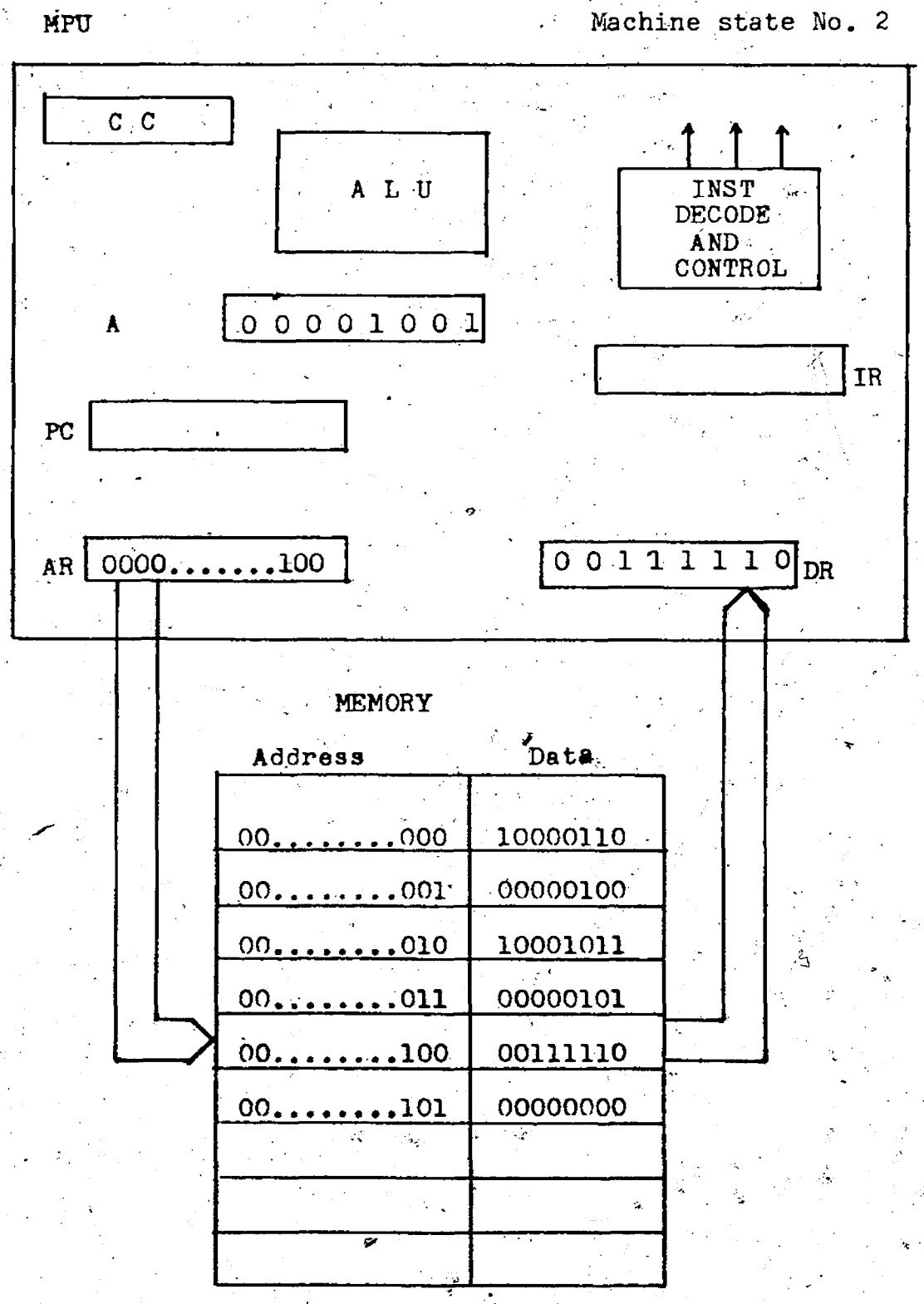
The sum of the accumulator and the data register are loaded into the accumulator.
This is the end of the execute cycle.

Fetch Cycle



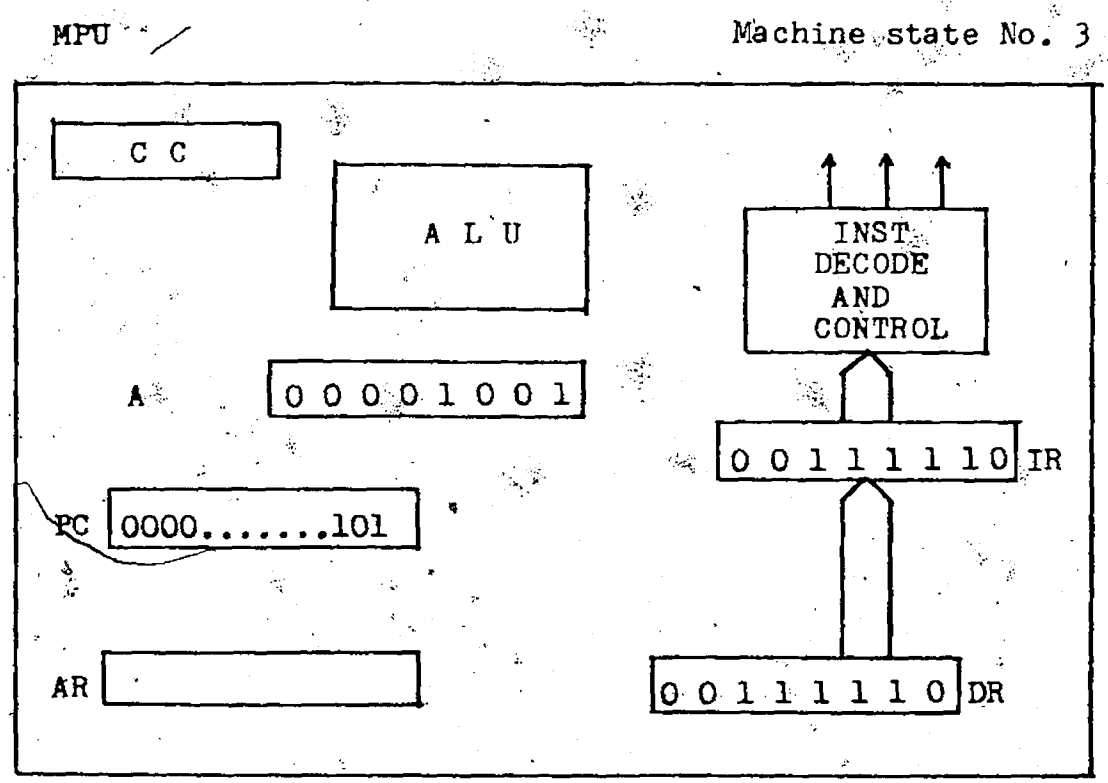
The contents of the program counter are loaded into the address register and then placed on the address bus.

Fetch Cycle



The contents (instruction) of the selected address are placed on the data bus and read into the data register.

Fetch Cycle



MEMORY

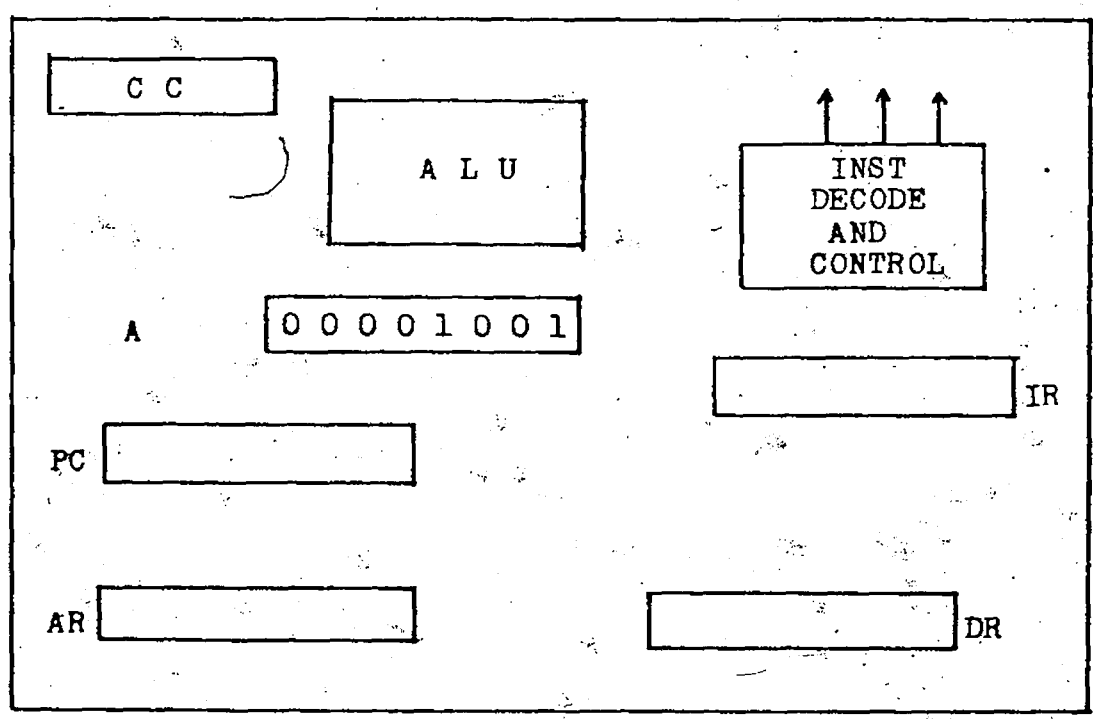
Address	Data
00.....000	10000110
00.....001	00000100
00.....010	10001011
00.....011	00000101
00.....100	00111110
00.....101	00000000

The instruction in the data register is transferred into the instruction register for instruction decode and execution. The program counter is incremented.

Execute Cycle

MPU

Machine state No. 1



MEMORY

Address	Data
00.....000	10000110
00.....001	00000100
00.....010	10001011
00.....011	00000101
00.....100	00111110
00.....101	00000000

The control stops producing control signals, all computer operation stops.

Learning Activity A4

Fetch/Execute using Zero Page Addressing

The program in the previous illustration used the immediate mode of addressing. That is the operand was located in the address following the instruction.

Another common method of addressing is direct or zero page addressing. In zero page addressing, the address of the operand is contained in the second byte of the instruction.

This address must be between 00 Hex and FF hex.

The following program uses zero page addressing:

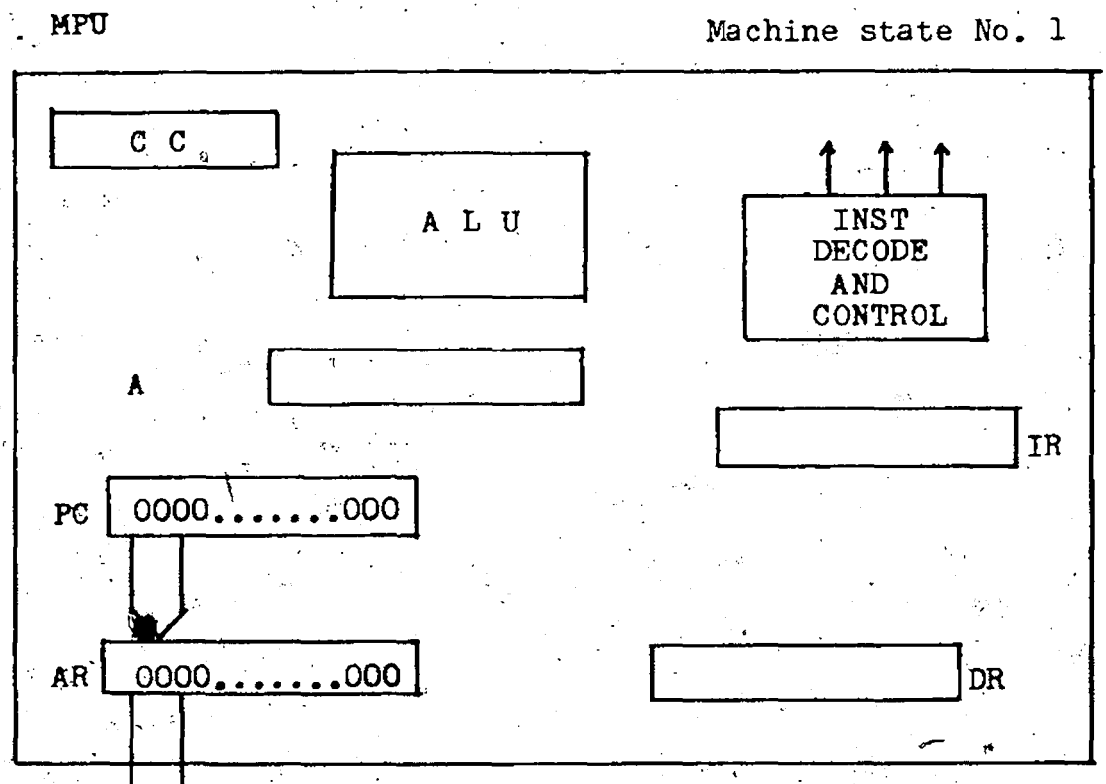
Mnemonic code	Address LSB of the operand	Operation code	Address LSB of the op-nd
LDA	07	10010110	00000111
ADD	08	10011011	00001000
HLT		00111110	

In words, the contents of memory location 07 will be loaded into the accumulator and the contents of memory location 08 will be added to the contents of the accumulator. The result will be stored in the accumulator.

The following pages are a series of illustrations that show MPU operation as the MPU sequences through a program using zero page addressing.

Run the Program

Fetch Cycle

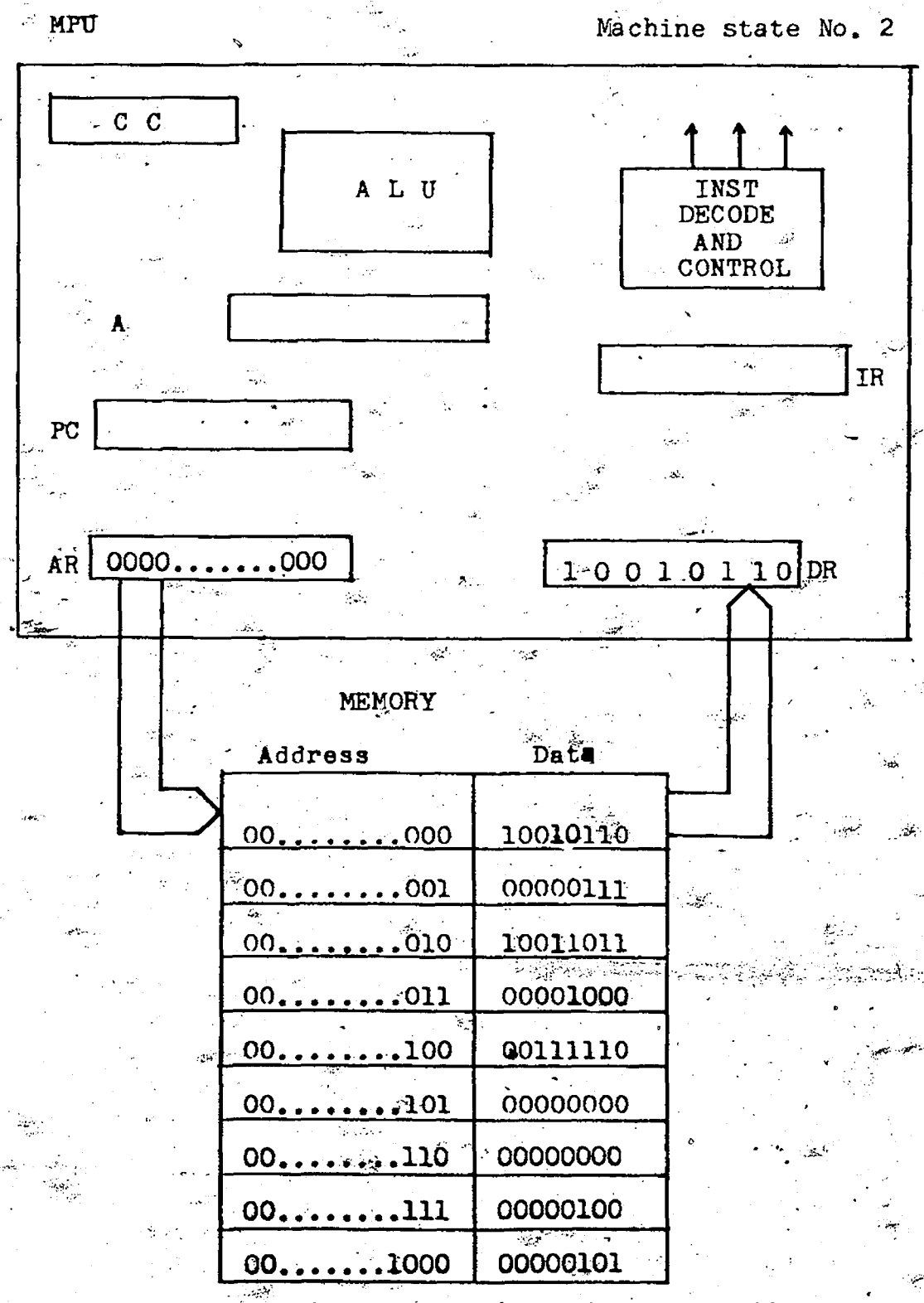


MEMORY

Address	Data
00.....000	10010110
00.....001	00000111
00.....010	10011011
00.....011	00001000
00.....100	00111110
00.....101	00000000
00.....110	00000000
00.....111	00000100
00.....1000	00000101

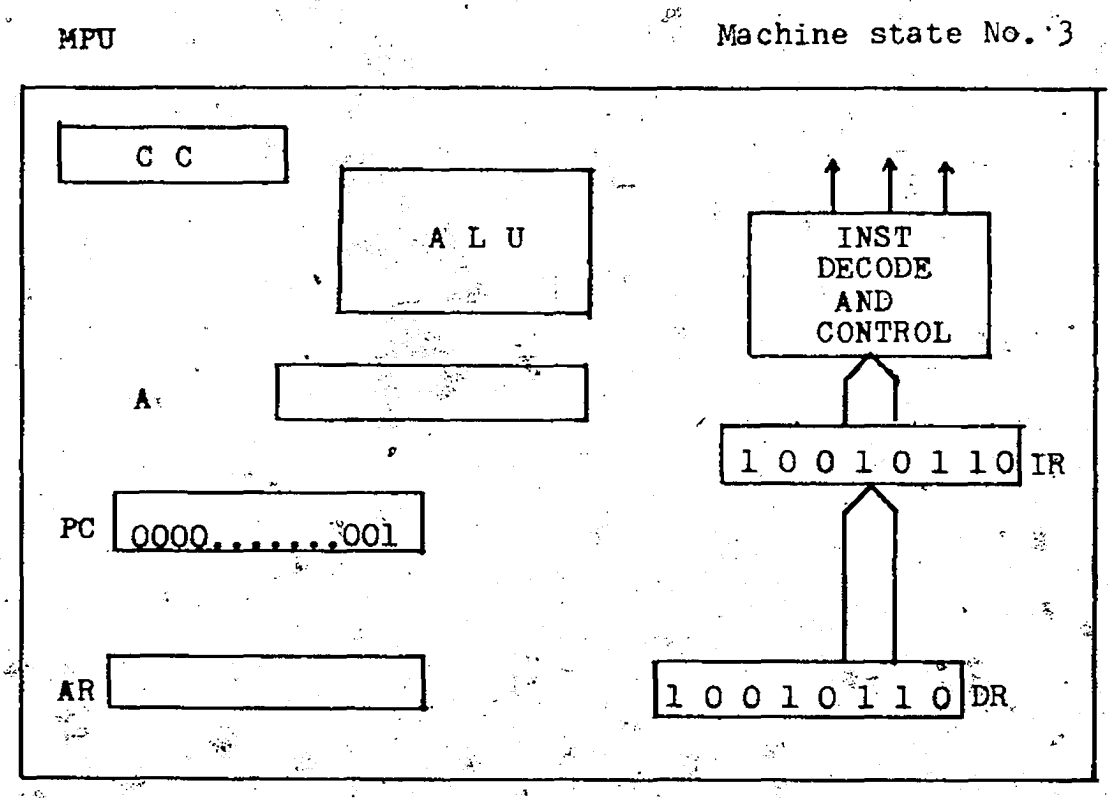
The contents of the PC are put on the address bus.

Fetch Cycle



The contents (instruction) of the first address are placed on the data bus and read into the address register.

Fetch Cycle

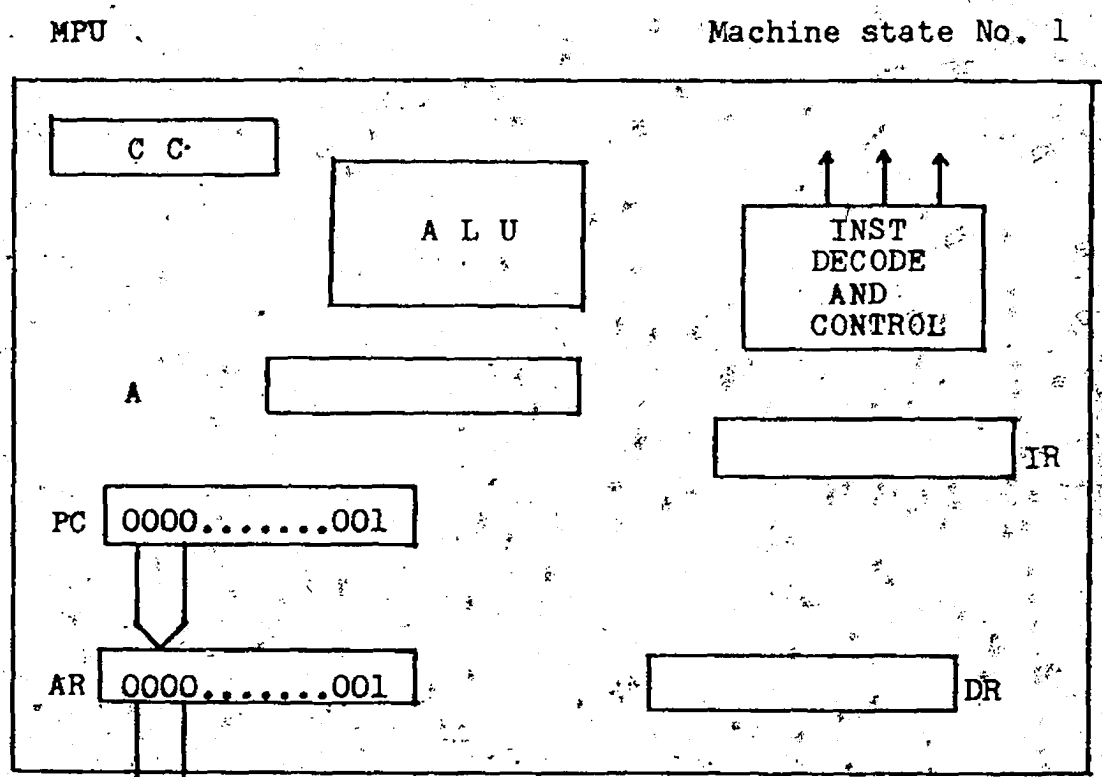


MEMORY

Address	Data
00.....000	10010110
00.....001	00000111
00.....010	10011011
00.....011	00001000
00.....100	00111110
00.....101	00000000
00.....110	00000000
00.....111	00000100
00.....1000	00000101

The instruction in the data register is transferred to the instruction register for instruction decode and execution
The program counter is incremented.
This completes the fetch cycle.

Execute Cycle

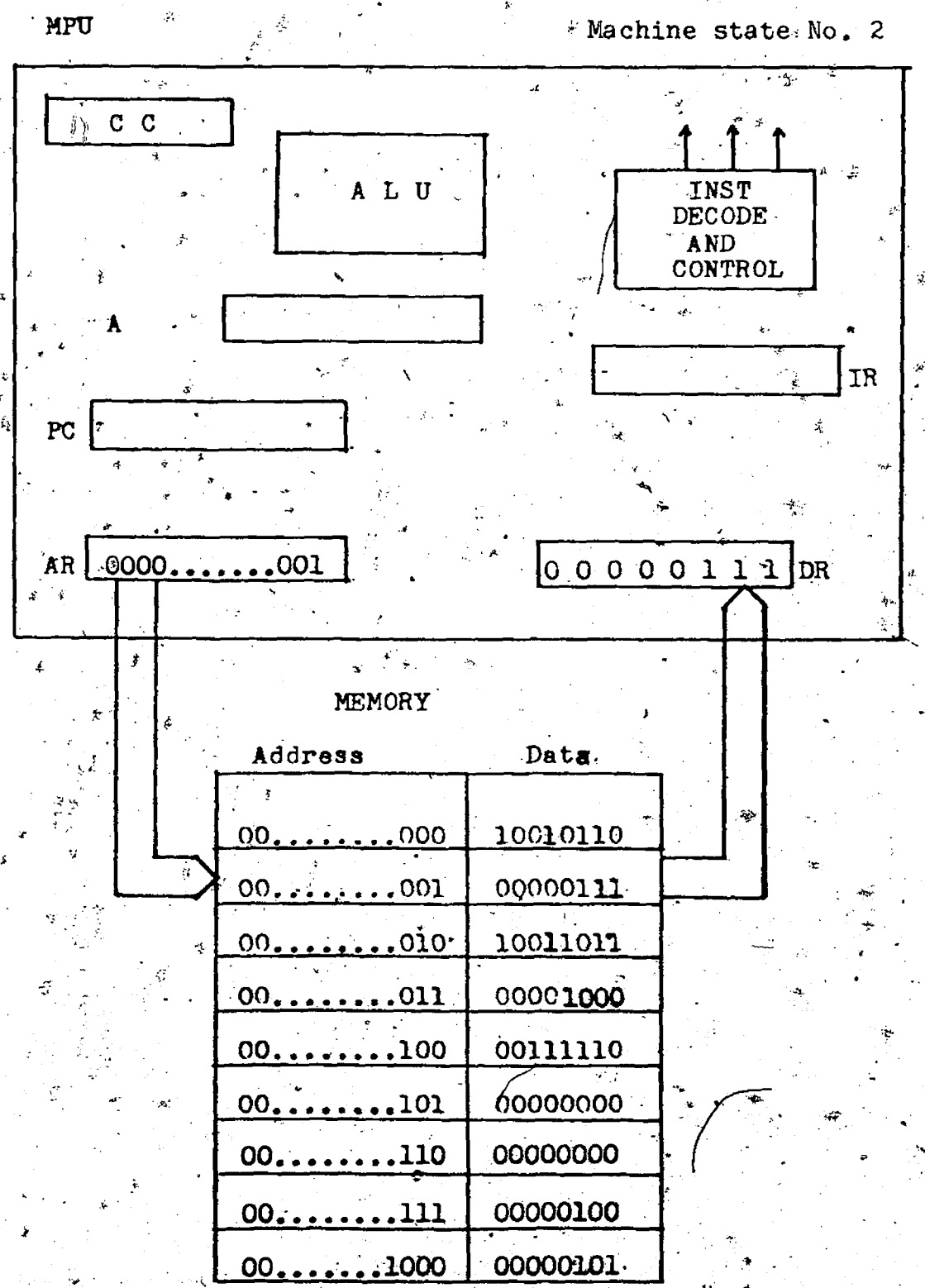


MEMORY

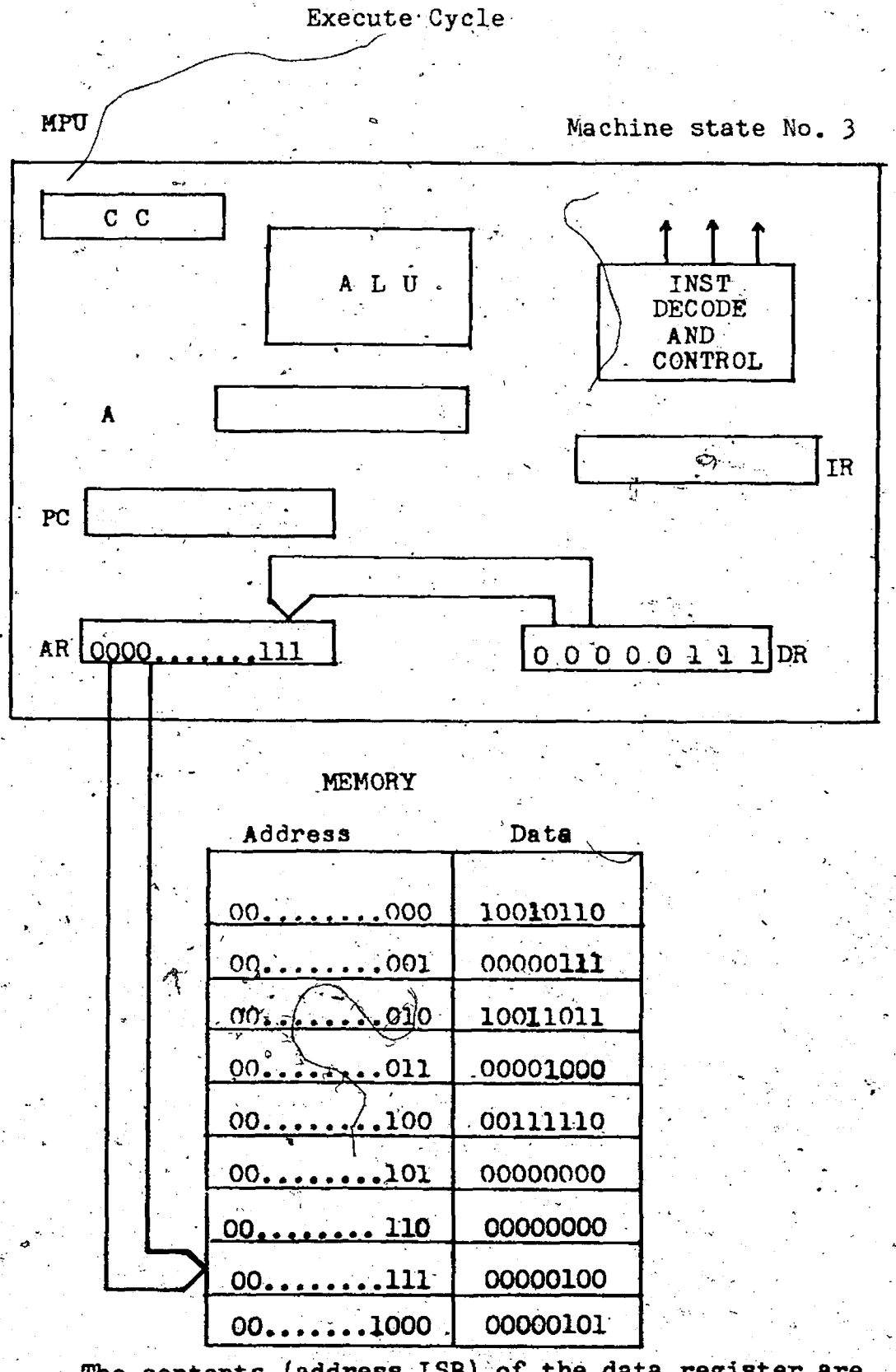
Address	Data
00.....000	10010110
00.....001	00000111
00.....010	10011011
00.....011	00001000
00.....100	00111110
00.....101	00000000
00.....110	00000000
00.....111	00000100
00.....1000	00000101

The contents of the PC is placed on the address bus.

Execute Cycle

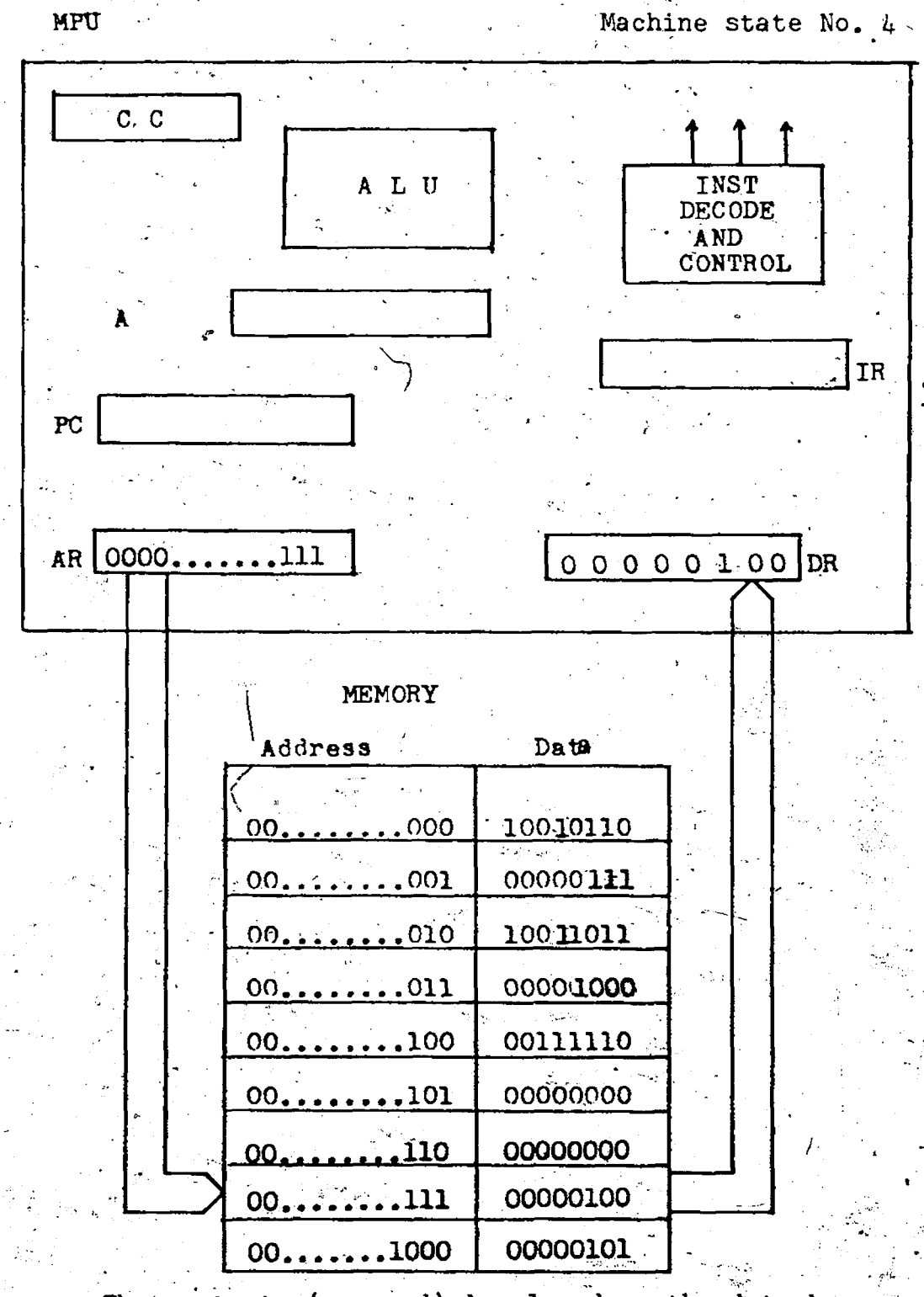


The contents (address LSB) of the selected address are placed on the data bus and read into the data register



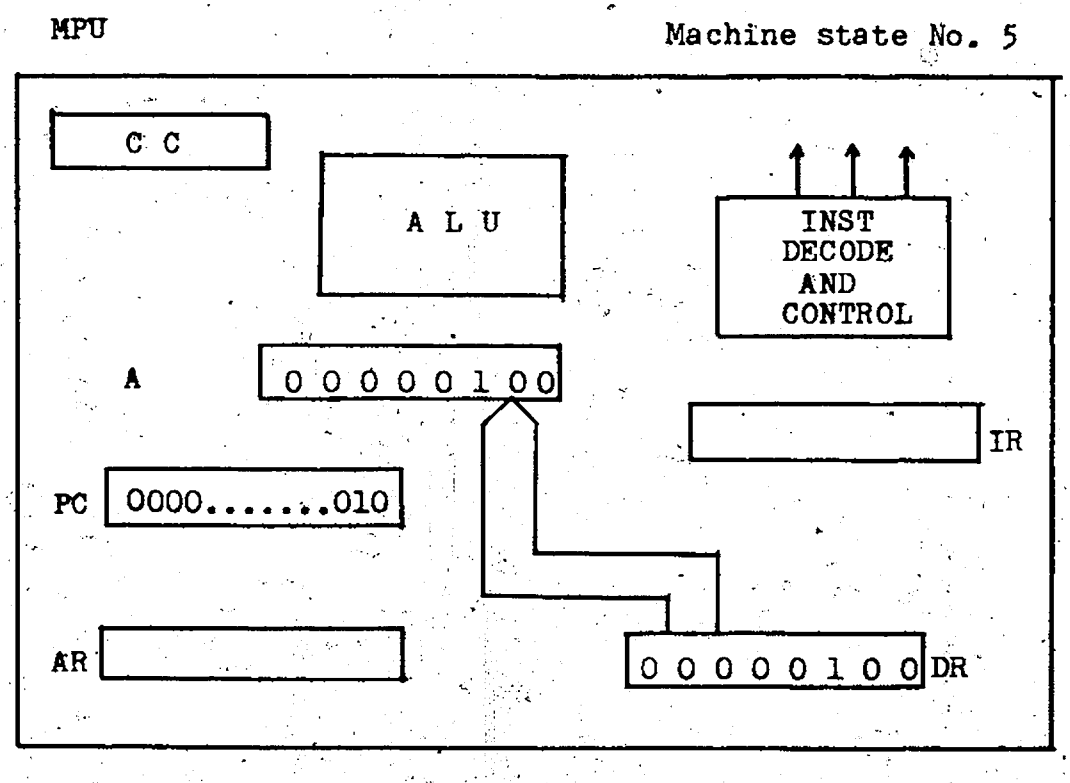
The contents (address LSB) of the data register are read into the address register and placed on the address bus.

Execute Cycle



The contents (operand) is placed on the data bus and read into the data register.

Execute Cycle



MEMORY

Address	Data
00.....000	10010110
00.....001	00000111
00.....010	10011011
00.....011	000C1000
00.....100	00111110
00.....101	00000000
00.....110	00000000
00.....111	00000100
00.....1000	00000101

The contents of the Data Register are transferred to the accumulator.
The PC is incremented

To complete the program the MPU must sequence through the following:

Fetch

1 The contents of the program counter (00000000000000-10) are placed on the address bus.

2 The contents (opcode) of the selected address are transferred to the data register.

3 The contents of the data register are transferred to the instruction register for instruction decode and execution. The program counter is incremented.

Execute

1 The contents of the program counter (00000000000000-11) are placed on the address bus.

2 The contents (address LSB) of the selected address are transferred to the data register.

3 The contents (address LSB) of the data register are transferred to the address register. The address contained in the address register is placed on the address bus.

4 The contents (operand) of the selected address is transferred to the data register.

5 The contents of the data register are transfered to the Arithmetic Logic Unit. The ALU adds the contents of the accumulator and the data register and stores the result in the accumulator. The program counter is incremented.

The program will fetch and execute the HLT instruction as before.

Learning Activity A5

A Symbolic method to illustrate microcomputer operation

Symbols have been devised so that we can illustrate microcomputer operation in a shorter form, they are:

```
----------->  Data transfer
<---------->  Data exchange
(         )   Contents of a register
[  ADDR   ]   Memory location address
([ ADDR  ])   Contents of a memory location address
    MA        Address pointed to by the address register
```

Example (PC) --------> AR would represent the transfer of the program counter into the memory address register.

The instruction fetch cycle could be represented as:

```
State 1       (PC)  -----> AR
State 2     ([ MA ]) ----> DR
State 3       (DR)  -----> IR
              (PC)+1 ----> PC
```

Compare the above method to the previous illustrations for the fetch cycle.

Instruction execution (immediate addressing)

```
State 1       (PC)  -----> AR
State 2     ([ MA ]) ----> DR
State 3       (DR)  -----> Acc
              (PC)+1 ----> PC
```

Instruction execution using zero page addressing

```
State 1       (PC)  -----> AR
State 2     ([ MA ]) ----> DR
State 3       (DR)  -----> AR
              (PC)+1 ----> PC
              Contents of the data register are
transferred to the LSB of the address register and placed
on the address bus. The program counter is incremented.
State 4     ([ MA ]) ----> DR
State 5       (DR)  -----> Acc
```

Compare the above method to the previous illustrations showing the MPU sequence for execute using zero page addressing.

Learning Activity A6

Introduction to the STA (store) instruction using zero page addressing and symbolic notation.

	Mnemonic Code	Operation code
Store the accumulator at the address determined by the contents of the next location	STA	10010111

Fetch

State 1 (PC) -----> AR
State 2 ([MA]) ----> DR
State 3 (DR) -----> IR
(PC)+1 ----> PC

Execute

State 1 (PC) -----> AR
State 2 ([MA]) ----> DR
State 3 (DR) -----> AR
(PC)+1 ----> PC
Contents of the data register are placed on the address bus via the address register.
State 4 (A) -----> DR
Contents of the accumulator are transfered to the data register
State 5 (DR) -----> [AR]
The contents of the data register are transferred to the address specified by the memory address register.

This completes your introduction to fetch/execute operation of the microprocessor. The next section involves a real microcomputer. You will be asked to perform a series of operations. These are designed to teach you MPU operation as well as the MPU instruction set.

B A Real Computer

1 Advanced Interactive Computer AIM 65

2 The AIM 65 Instruction set

3 AIM 65 Addressing Modes

4 AIM 65 Course Objectives

5 Writing and Executing the AND Program on the AIM 65

6 OR (IMMEDIATE)

7 Zero Page Addressing

8 OR Zero Page Addressing

9 ADC (Addition)

10 SBC (Subtraction)

11 Printing out the result of an Addition

12 Input Data from the Keyboard

13 I/O

14 - 15 Instruction Entry (I), Disassembly (K)

16 Using input and output ports

17 The AIM 65 as an AND gate

18 AIM 65 Simulating Logic Gates

19 Subroutines

20 Stack

21 The Traffic Light Problem

22 Simulation of a Monostable Multivibrator

23 Simulation of a D Latch

24 Simulation of a BCD to 7 Segment Decoder

25 Hardware Interrupts

26 Break

27 Interrupts using the VIA

28 Computer as a Shift Register

29 Alarm Program

30 Bell Program

Learning Activity B1

AIM 65 Course Objectives

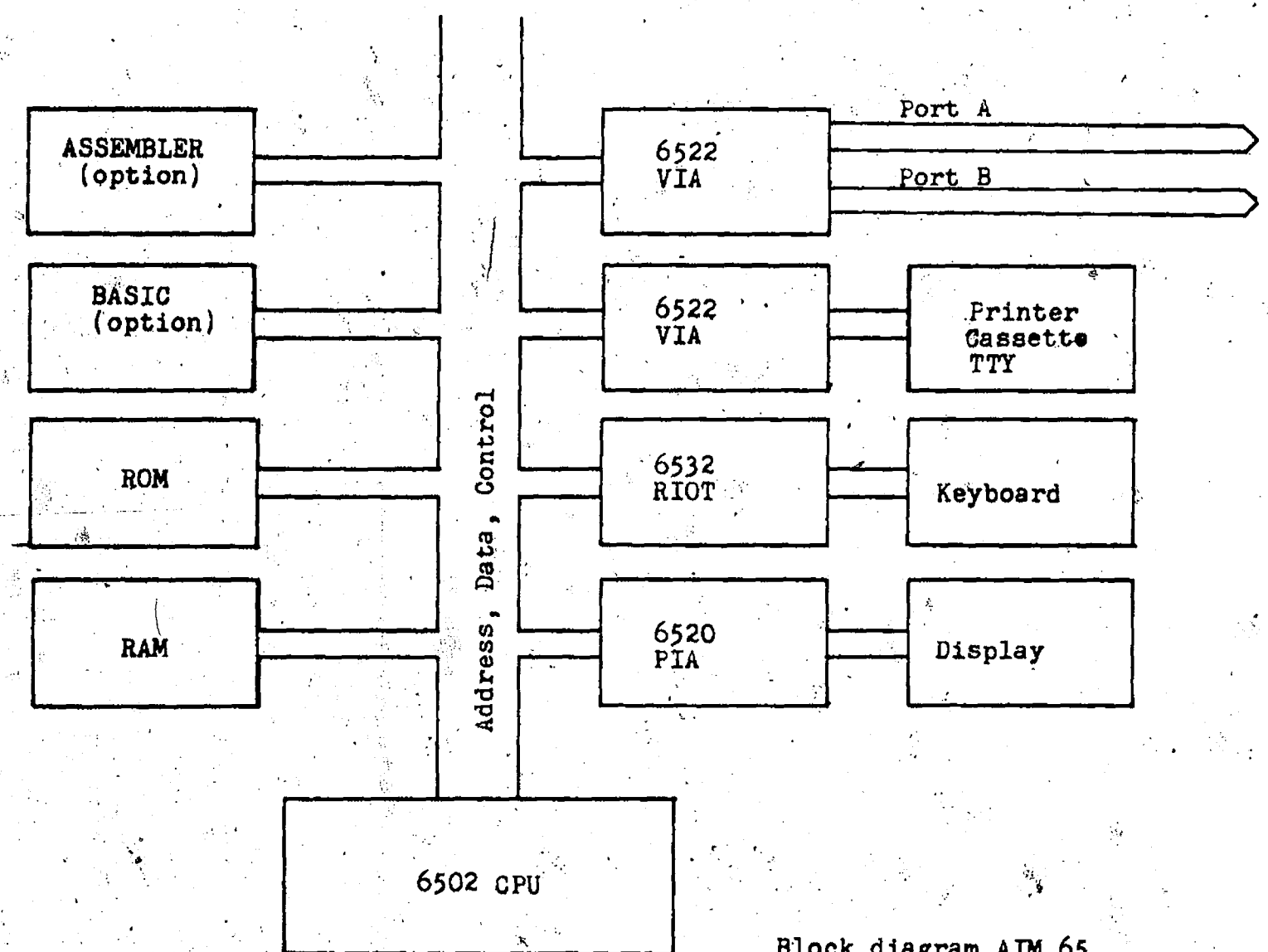
In order to demonstrate how the AIM 65 works we will write a number of programs using the machine instructions and initially only immediate and zero page addressing. These programs are designed to:

1) Give you experience in using the computer.
2) Familiarize you with the AIM 65 instruction set.
3) Develop skill in converting instructions (given in mnemonic code) to opcode (machine code).
4) Give you experience in writing machine level programs.
5) Develop skill in using the extensive AIM 65 operating system.
6) Develop skills in interfacing the AIM 65 with the real world.
7) Help you understand how a microcomputer works.

You will require the following:

The 6502 instruction set, part of the AIM 65 instruction set. See also learning activity B3 and B4.
AIM 65 Microcomputer.
AIM 65 monitor program listing.
AIM 65 users guide.
R 6500 microcomputer system hardware manual.
R 6500 microcomputer system programming manual.

A block diagram of the AIM 65 is included to illustrate the relationship between the different parts of the microcomputer. See figure 1.



Block diagram AIM 65

Learning Activity B2

Advanced Interactive Computer
Rockwell AIM 65

The AIM 65 is a complete general purpose microcomputer featuring advanced hardware and software.

The heart of the AIM 65 is the 6502 microprocessor. The cycle time (1 microsecond) of the 6502 is controlled by a 4 Mhz crystal.

The computer contains a full keyboard, twenty character display module, two software controlled I/O ports and a twenty character on board printer. In addition it is switch selectable to operate a TTY.

An 8K ROM is included in the system, this provides powerful software features that allow you to enter and debug programs from the keyboard. In addition 4K RAM workspace is provided for the user.

Other 6500 devices in the AIM 65 are, 6520 Peripheral Interface Adapter (PIA) for the display, 6532 RAM-I/O Timer for the keyboard interface, and the Versatile Interface Adapter (VIA) for interface with the real world.

Although the AIM 65 is an ideal educational system it is a full fledged computer with power to handle any of the following:

1 Factory data collection terminal
2 Integrated circuit tester
3 Automatic service monitor
4 Process control
5 Motor control
6 Navigation calculator
The list can be very extensive.

To use the AIM 65, simply switch on the computer and it will cycle through a reset and print out ROCKWELL AIM 65 on the printer and the display. To switch the printer off or on simply press CTRL/PRINT, try it, the display will indicate the condition of the printer.

(CTRP/PRINT means press both keys at the same time)

Learning Activity B3

The AIM 65 Instruction Set

The power of the AIM comes from it's extensive instruction set combined with an extensive set of addressing modes.

The following is the Machine instructions set for the AIM 65 with a brief explanation of their function:

MACHINE INSTRUCTIONS

ADC Add memory to accumulator with carry
AND AND memory with accumulator
ASL Shift left one bit (memory or accumulator)

BBC Branch on carry clear
BSC Branch on carry set
BEQ Branch on result zero
BIT Test bits in memory with accumulator
BMI Branch on result minus
BNE Branch on result not zero
BPL Branch on result not plus
BRK Force break
BVC Branch on overflow clear
BVS Branch on overflow set

CLC Clear carry flag
CLD Clear decimal mode
CLI Clear interrupt disable bit
CLV Clear overflow flag
CMP Compare memory and accumulator
CPX Compare memory and index x register
CPY Compare memory and index y register

DEC Decrement memory by one
DEX Decrement index x register by one
DEY Decrement index y register by one

EOR Exclusive-OR memory with accumulator

INC Increment memory by one
INX Increment index x register by one
INY Increment index y register by one

JMP Jump to new location (return address not saved)
JSR Jump to new location (return address saved)

LDA Load accumulator with memory
LDX Load index x register with memory
LDY Load index y register with memory
LSR Shift right one bit (memory or accumulator)

NOP No operation

ORA Or memory with accumulator

PHA Push accumulator on stack
PHP Push processor status on stack
PLA Pull accumulator from stack
PLP Pull processor status from stack

ROL Rotate one bit left (memory or accumulator)
ROR Rotate one bit right (memory or accumulator)
RTI Return from interrupt
RTS Return from subroutine

SBC Subtract memory from accumulator with borrow
SEC Set carry flag
SED Set decimal mode
SEI Set interrupt disable status
STA Store accumulator in memory
STX Store index x register in memory
STY Store index y register in memory

TAX Transfer accumulator to index x register
TAY Transfer accumulator to index y register
TXA Transfer index x register to accumulator
TSX Transfer stack pointer to index x register
TXS Transfer index x register to stack pointer
TYA Transfer index y register to accumulator

Learning Activity B4

AIM 65 Addressing Modes

The following are the addressing modes for the AIM 65 with a brief explanation of their function. This course will concentrate on the first seven.

ADDRESSING MODES

IMM -Immediate addressing- The operand is contained in the second part of the instruction.

ABS -Absolute addressing- The second byte of the instruction contains the 8 low order bits of the effective address. The third byte contains the 8 high order bits of the effective address.

Z Page -Zero page addressing- Second byte contains the 8 low order bits of the effective address. The 8 high order bits are zero.

A -Accumulator- One byte instruction affecting the accumulator.

IMP -Implied addressing-One byte instruction affecting registers in the MPU.

REL - Relative addressing- Two byte instruction, the second byte is an offset from the program counter that determines the address of the next instruction. See manual for method to determine the offset.

Absolute Indirect- This is a three byte instruction used exclusively with the JMP instruction.

Z Page,X -Z Page,Y -Zero page indexed- The second byte of the instruction is added to the index register to form the low order byte of the effective address. The high order byte of the effective address is all zeros.

ABS,X- ABS,Y -Absolute indexed- The effective address is formed by adding the index to the second and third byte of the instruction.

INX,X -INDEXED INDIRECT-The second byte of the instruction is added to the X index. The result points to a location on page zero which contains the low order 8 bits of the effective address. The next byte contains the 8 high order bits.

IND,Y -Indirect Indexed-The second byte of the instruction points to a location in page zero. The contents of this location is added to the Y index, the result being the low order 8 bits of the effective address. The carry from this operation is added to the contents of the next page zero location, the result being the 8 high order bits of the effective address.

Learning Activity B5

Writing and Executing the AND program on the AIM 65

Objective: To write and execute a program on the AIM 65.

A program is a detailed list of instructions that tell the microprocessor what to do step by step.

The program is usually written in assembly language first, using mnemonics, then disassembled to get the computer opcode and data.

The opcode and data in Hex (converted to binary by the computer software) is entered into the computer memory in sequential memory locations.

Following is an example of a program written in assembler language, disassembled to opcode and data. This program will AND the data stored in two words and store the result in a third word. Unless otherwise indicated all numbers are hexadecimal.

Assembler Program	Comments
LDA #17	Load the accumulator in the immediate mode
AND #05	AND the accumulator in the immediate mode
STA 08	Store the result in memory location 08, this instruction uses zero page addressing
BRK	Stop program execution

Use the 6502 instruction set to verify the following opcode.

Mnemonic Code	Opcode
LDA	A9
AND	29
STA	85
BRK	00

Assembled Program	Disassembled Program
LDA #17	A9
	17
AND #05	29
	05
STA 08	85
	08
BRK	00

Load the program into sequential memory locations starting at memory location 0.

Press (in sequence) ESC M 0 Return

This sequence will cause the bottom four memory locations to be displayed.

Press / to change memory contents.

Enter (in sequence) A9 17 29 05 /(to continue)
85 08 00 Return

Press M 0 Return

Check that memory corresponds to:

0000 A9
0001 17
0002 29
0003 05
(Press space bar for next set of memory data)
0004 85
0005 08
0006 00

To set the program counter to zero
Press ESC * 0 Return

To execute the program
Press G Return

To check results:
Press M 08 Return

Data in memory location 0008 = ...
00010111 AND 00000101 = ...

If the above are not equal repeat.
Can you explain why they should be equal?

Review the above program very carefully. You will be asked to write your own programs in the future. Be sure you understand and can execute this program.

Learning Activity B6

OR (Immediate)

Objective: To write and execute a program that will OR two computer words (numbers).

The method used to AND two words in B5 can be used to OR the same two words (17,05).

1 Write a program in assembler language (using mnemonic code) that will OR the contents of two memory locations.

2 Disassemble the code. Same method as with AND.

3 Load it in sequential memory locations.

When entering code in memory you can type space for any value you want to leave unchanged, again same method used with AND.

4 Execute the program, i.e. set program counter to zero and press G, again same method used with AND.

The mnemonic for OR is ORA, opcode is 09.

```
Program     LDA #17
            ORA #05
            STA  08
            BRK
```

This is the program for number 1 above, now disassemble, load and execute.

Calculate the OR of the two words ---------

Computer OR of same two words ---------

If the above answers are not equal, do again!

Congratulations, you have just written your first program in machine code.

Learning Activity B7

Zero Page Addressing

Objective: To write and execute a program using zero page addressing.

Mnemonic codes that use the first 256 locations in memory are said to use zero page addressing. These are two byte instructions i.e. the first byte is an opcode and the second byte is an address.

Example: LDA AD means load the accumulator with the contents of memory location 00AD (173 in decimal).

Write a program that will AND the contents of two memory locations using zero page addressing.

Assembly Program	Opcode and Data
LDA AD	A5
	AD
AND AE	25
	AE
STA 0F	85
	0F
BRK	00

The disassembled program is also shown above.

Use your 6502 instruction set to check the opcode. If you have a problem ask for a demonstration!

Load data (17,05) into memory locations 00AD and 00AE

Press ESC or Reset

Press M 00AD
Display reads: 00AD -- -- -- --
To indicate data in memory locations AD, AE, AF, B0

To change data in AD to 17 and AE to 05
Press / 17 05 Return

Load your AND program, starting at memory location 0, check memory, execute.

Data in memory location 000F = ---------

00010111 AND 00000101 = ---------

If the above answers are not equal repeat!

You have just completed an exercise using zero page addressing. Already you have extended the power of the computer.

Learning Activity B8

OR Zero Page Addressing

Objective: To write and execute a program that will OR two computer words (numbers). Use zero page addressing.

Write and execute a program using zero page addressing that will OR the same two words used in B7.

Opcode for OR using zero page addressing is 05.

Here is a chance for you to really show your stuff. Use the last program as a model.

Summary

	Opcode		
Mnemonic	immediate	zero page	implied
LDA	A9	A5	
AND	29	25	
ORA	09	05	
STA		85	
BRK			00

Learning Activity B9

ADC (Addition)

Objective: To write and execute a program that will add two computer words (numbers).

Using immediate and zero page addressing write a program that will;

1) add 17 and 05
2) and store the result.

Before an add (Mnemonic code ADC) or subtract (mnemonic code SBC) can be carried out the carry flag must be cleared using a CLA instruction. This should be the first instruction in your program. In addition we will have to clear the decimal flag (mnemonic code CLD) to make sure our addition takes place in binary and not BCD.

Program in assembler language

```
CLC
CLD
LDA   0D
ADC   0E
STA   0F
BRK
```

Opcode for CLC	18
Opcode for CLD	D8
Opcode for ADC(zero page)	65

Disassemble and load the machine code program into sequential memory locations starting at 0000. Use the memory change function (/) to load 17 into memory location 000D and 05 into memory location 000E.

Execute your program.

The results will be contained in memory location 000F.

Compare your results with normal addition.

Learning Activity B10

SBC (Subtraction)

Objective: To write and execute a program that will subtract two numbers.

Write and execute a program to subtract the two numbers (17 and 05 Hex).

Compare results with normal subtraction.

SBC is mnemonic for subtract. What is the opcode?

Use the program you wrote in B9 as a model.

Learning Activity B11

Printing out the result of an addition

Objective: To be able to print out the result of an operation performed by the AIM 65.

The computer has a program at EA46 that will print out the contents of the accumulator. We can use this program by simply jumping (mnemonic code JSR) to EA46.

Example:

Load 17 in memory location FD
Load 04 in Memory location FE

Program

```
CLC
CLD
LDA   FD
ADC   FE
JSR   EA46
JSR   EA13    output carriage return
BRK
```

Disasembled program

```
18    D8    A5    FD
65    FE    20    46
EA    20    13    EA
00
```

Load the program starting at memory location zero.

Press CTRL/PRINT, to turn the printer on.

```
Press    *    0    Return
Press    G    Return
```

The contents of the accumulator (i.e. the sum of 17 and 04 Hex) will be printed at the left side of the tape.

Remember ((A0)) means contents of memory location A0

```
Check  ((FD))+((FE))= ----------
          17 + 04   = ----------
```

NOTE addition takes place in Hex

Learning Activity B12

Input Data from the Keyboard

Objective: To be able to input data from the keyboard, process it, and output data to the printer.

With the addition of one more bit of information we can use the full power of the computer, that is we can;

1 input data

2 process it

3 output the result.

The computer has a program at memory location E3FD that will input two Hex digits from the keyboard and pack them in the accumulator. Each time this program is called up we can get two more Hex numbers. These numbers can then be stored in memory for later use.

Example;

```
CLC
CLD
JSR   E3FD    Input two hex digits to the accumulator
STA   FD
JSR   E3FD    Get two more
STA   FE
```

We could now use previous information obtained in B11 to complete the program, that is, add the two numbers and print out the sum.

Example complete program

```
JSR   E3FD    Get two digits
STA   FD
JSR   E3FD    Get two more
STA   FE
JSR   EA13    Output carriage return
CLC           Remember clear carry flag
CLD           Remember clear decimal flag
LDA   FD
ADC   FE
JSR   EA46
JSR   EA13    Output carriage return
BRK
```

Program Disassembled, check for error with your instruction set.

20	FD	E3	85
FD	20	FD	E3
85	FE	20	13
EA	18	D8	A5
FD	65	FE	20
46	EA	20	13
EA	00	00	

Load into sequential memory locations starting at 0000.

Press CTRL/PRINT, Remember to turn the printer on.

Press * 0 Return
Press G Return

Input data 17 05

Data recorded on the printer just below input data = -----

Is this data correct i,e. does it = 17Hex + 05Hex?

Write a similar program using mnemonic code that will input two Hex numbers from the keyboard, subtract them and print out the result.

Disassemble the code

Load it into sequential memory locations starting at 0000

Execute the program

Record results, compare wih normal subtraction

Repeat if not correct.

You should now feel confident to write many machine language programs.

Learning Activity B13

I/O

Objective: To be able to set up the input and output ports.

The AIM 65 uses memory mapped I/O, that is the CPU will address input/output in exactly the same way it normally address memory.

For example, to read the data on an input port you simply LDA (load the accumulator) with the contents of the input port. That port has a specific address (A001) just as a specific memory location has a specific address.

To send data out a port just STA (store the accumulator) at a specific port (A000).

Before A001 can become an input port and A000 an output port a short program must be written to "set up" the input and output ports. We will name this program "set up", and it will be stored starting at memory location 0400.

Address	Program	Comments
0400	LDA #00	Put 0 in the accumulator
0402	STA A003	Put 0 in data direction resister to make A001 an input port
0405	LDA #FF	Put all 1's in the accumulator
0407	STA A002	Put 1's in the data direction resister to make A000 an input port
040A	LDA #00	
040C	STA A00B	Housekeeping
040F	STA A00C	
0412	STA A00E	
0415	RTS	

Learning Activity B14-15

Instruction Entry (I), Disassembly (K)

Objective: To be able to enter data using the instruction entry function.
To be able to disassemble data in the computer memory to symbolic 6502 instructions.

The AIM 65 has a really neat feature, called instruction entry (I), that lets you write programs in assembly language and enter them directly into the computer without disassembly. That is the mnemonics and data can be entered directly from the Keyboard.

For example, to enter the program "set up" into the computer starting at memory location 0400 proceed as follows:

Press	CTRL/PRINT	Turns printer on
Press	ESC or Reset	
Press	* 0400 C/R(return)	
Press	I	

0400 will appear in the display

Enter the program "set up" in the following manner:

```
LDA #00     C/R
STA A003    C/R
LDA #FF     C/R
STA A002    C/R
LDA #00     C/R
STA A00B    C/R
STA A00C    C/R
STA A00E    C/R
RTS
```

To see if the program is entered correctly you can use the disassemble memory function (K).

Press	ESC or Reset	
Press	CTRL/Print	Turns printer on
Press	K	
Display reads K *=		Press 0400 C/R
Press 09		To disassemble 9 instructions and print them out

Compare the printout with the program entered and if an error appears, correct it using the I function.

This program can also be checked using the M (examine memory) function.

Press ESC or Reset

Press CTRL/Print Turn printer on

Press M

Display reads M = , Press 0400 C/R

The contents of the first four memory locations starting at 0400 will be displayed. i.e.

```
M =  0400     A9     00     8D     03
Press space bar for 4 more
     0404     A0     A9     FF     8D
Press space bar
     0408     02     A0     A9     00
Press space bar
     040C     8D     0C     A0     8D
Press space bar
     0410     0B     A0     8D     0E
Press space bar
     0414     A0     60     --     --
```

The contents of the above memory locations contain your original program disassembled.

It should be clear that you have two ways to enter a program into the computer.

1 Write the program in assembler language using mnemonics code, disassemble by hand and use the memory alter (/) function to enter the machine code.

2 Use the Instruction Enter Function (I) to enter the program into the computer.

You should become completely familiar with the I and K function. Go back to five of your previous programs, Use the I function to enter them, then use the K function to see if they are entered correctly.

Use the M function to see how good you were at hand disassembly. (You probably will never do it again!)

Execute each of your programs and show the printed result to your teacher.

Learning Activity B16

Using Input and Output Ports

Objective: To set up Port A as an input port and Port B as an output port.

The subroutine "set up" sets up Port A (pins 14,4,3,2,5,6,7,8) on the edge connector J1 as an input port and Port B (pins 9,10,11,12,13,16,17,15) on the edge connector J1 as an output port.

By calling up the subroutine, i.e. JSR to the address of the subroutine, we can use the computer to simulate logic gates, combinational and/or sequential logic circuits.

Subroutine "set up"

```
0400  LDA #00
0402  STA A003
0405  LDA #FF
0407  STA A002
040A  LDA #00
040C  STA A00B
040F  STA A00C
0412  STA A00E
0415  RTS
```

This subroutine can be relocated, i.e. it could start at 0420 or at any location available in your memory.

Learning Activity B17

The AIM 65 as an AND Gate

Objective: To simulate an AND Gate with the AIM 65.

Load in subroutine "set up" at 0400. See B16.

Connect logic A to Port A bit 0, pin 14 on J1

Connect logic B to Port A bit 1, pin 4 on J1

Connect Led monitor to Port B bit 0, pin 9 on J1

Following is an assembly program that will simulate an AND gate:

Address	Program	Comments
0200	JSR 0400	Subroutine to set up ports, must be in the computer
0203	LDA A001	Get data at port A
0206	AND #01	Clear all but bit 0
0208	STA 04FF	Save bit 0
020B	LDA A001	Get data at port A
020E	AND #02	Clear all but bit 1
0210	LSR A	Line up bit 1 with bit 0
0211	AND 04FF	
0214	STA A000	Output result of AND operation
0217	BRK	Stop

Enter the above program into the computer using the I function.

Remember in order to execute this program:

Press ESC * 0200 C/R
Press G C/R

Use the logic switches to change the inputs to your simulated logic gate. After each change execute the program and complete a truth table for your AND gate.

Learning Activity B18

AIM 65 Simulating Logic Gates

Objective: To write and execute programs that will allow the AIM 65 to simulate the basic gates.

Write and execute a program that will simulate the OR gate. Use the AND gate simulation as a model. Complete the truth table for your simulated gate.

Write and execute a similar program that will simulate an EX-OR gate. Complete the truth table for your simulated gate.

The NAND, NOR, and EX-NOR gates can be simulated by inverting respectively the AND, OR, and EX-OR gates.

Following is a program to simulate a NAND, print out the result, as well as output data to port B.

```
0200    JSR 0400
0203    LDA A001
0206    AND #01
0208    STA 04FF
020B    LDA A001
020E    AND #02
0210    LSR A
0211    AND 04FF
0214    EOR #01      Inverts the simulated AND gate.
0216    STA A000
0219    JSR EA46
021C    JSR EA13
021F    BRK
```

Press ESC or Reset
Press CTRL/Print set printer

Execute the program

Data at port B will be printed under the G on the printer tape.

Complete a standard truth table for the NAND gate

Write and execute a program to simulate the NOR and EX-NOR gates and have the result printed out on the tape.

This completes your exercises using the computer to simulate basic gates

SUMMARY

You have gained experience in using some of the following machine instructions. For the instructions listed below give the operation each one performs.

ADC-

AND-

BNE-

BRK-

CLC-

DEC-

DEX-

DEY-

EOR-

INC-

INX-

INY-

JMP-

JSR-

LDA-

LDX-

LDY-

LSR-

NOP-

ORA-

PHA-

PLA-

RTS-

STA-

Learning Activity B19

Subroutines

Objective: To illustrate the JSR (Jump to subroutine) and RTS (return from subroutine) instructions

A Subroutine is a group of instructions that perform some limited but frequently required tasks. In many cases the easiest way to write a program is to break the overall job down to many simple operations, each of which can be performed by a subroutine.

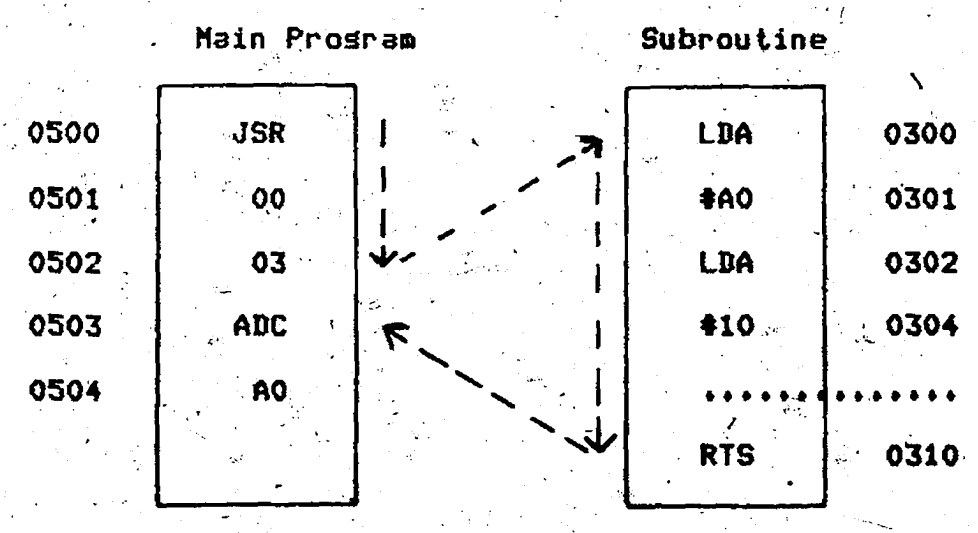
The microprocessor has special instructions to handle subroutines.

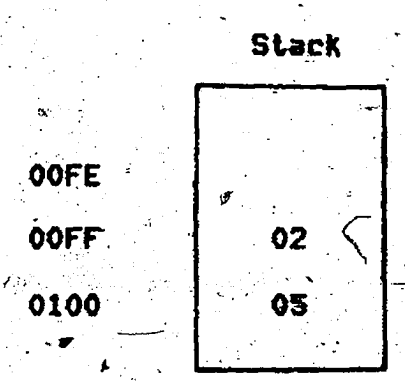
JSR - Jump to the subroutine
RTS - return from the subroutine

JSR The contents of the program counter +2 are pushed into the stack. The subroutine address is then loaded into the program counter. This is called a subroutine "call".

RTS Restore the program counter from the stack and increment it by 1. Adjust the stack pointer.

The following illustration shows how the flow of the program is interrupted by a JSR (Jump to subroutine) instruction.





Before JSR the stack pointer was set at 0100
After JSR the stack pointer is decremented to 00FE
After RTS the stack pointer is incremented to 0100

Once the JSR instruction is decoded the contents of the program counter are incremented by 2 and stored in the stack. The stack pointer is also incremented by 2, one for each byte.

Control of the program is then transferred to the address contained in the operand of the JSR instruction. The subroutine resides at this address.

The MPU then sequences through the subroutine until it encounters an RTS (return from subroutine) instruction.

New contents for the program counter are "pulled" from the top of the stack and incremented by 1. This returns the MPU to the main program and normal fetch/execute continues.

Learning Activity B20

Stack

Objective: To introduce the concept of the stack, and explain how data can be dumped into the stack.

The stack is an area in memory set aside to store data from the MPU. Data is stored in the stack when an interrupt in the normal flow of the program occurs. Instructions used by the stack are:

PHA - The contents of the accumulator are pushed into the stack, the stack pointer is decremented.

PLA - The top word of the stack is pulled back into the accumulator, the stack pointer is incremented.

PHP - The contents of the processor status register are pushed into the stack, the stack pointer is incremented.

PLP - The top word of the stack is pulled back into the processor stack, the stack pointer is incremented.

The stack pointer, a register in the CPU, or just stack contains the address of the stack. When an interrupt occurs data from the Accumulator, or other CPU registers, is stored in the stack (PHA instruction) at the address indicated by the stack pointer. The address in the stack pointer is decremented by 1.

To restore the data to the accumulator from the stack a PLA instruction is used. The address in the stack pointer is incremented.

Other registers can be pushed into the stack by transferring their data to the accumulator first and then dumping the accumulator into the stack.

Learning Activity B21

The Traffic Light Problem

Objective: To write a program that will allow the AIM 65 to operate a set of traffic lights.

When designing a complex program you should follow an orderly sequence. The following is a suggested procedure:

1 Define the problem
2 Design the solution
3 Flowchart the program
4 Write the program
5 Test and debug the program (execute it)

Define the problem:

A traffic light controller must sequence a series of lights according to the following:

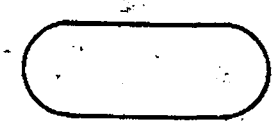
1 Light A is red, light B is green
2 Wait green time
3 Change light B to yellow
4 Wait yellow time
5 Change light B to red, light A to green
6 Wait green time
7 Change light A to yellow
8 Wait yellow time
9 Go to step 1 and repeat the process

Design the solution.:

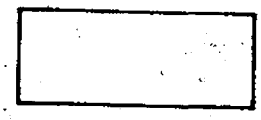
It looks like a linear program would work quite nicely, and subroutines could be used to call up the delay.

These groupings are required to set up the lights.

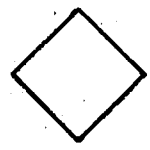
	A	B
Condition 1	Red on Yellow off Green off	Green off Red off Yellow off
Condition 2	Red on Y,G off	Yellow on G,R off
Condition 3	Green on R,Y off	Red on Y,G off
Condition 4	Yellow on G,R off	Red on Y,G off
Condition 5	Same as condition 1	



Program Name



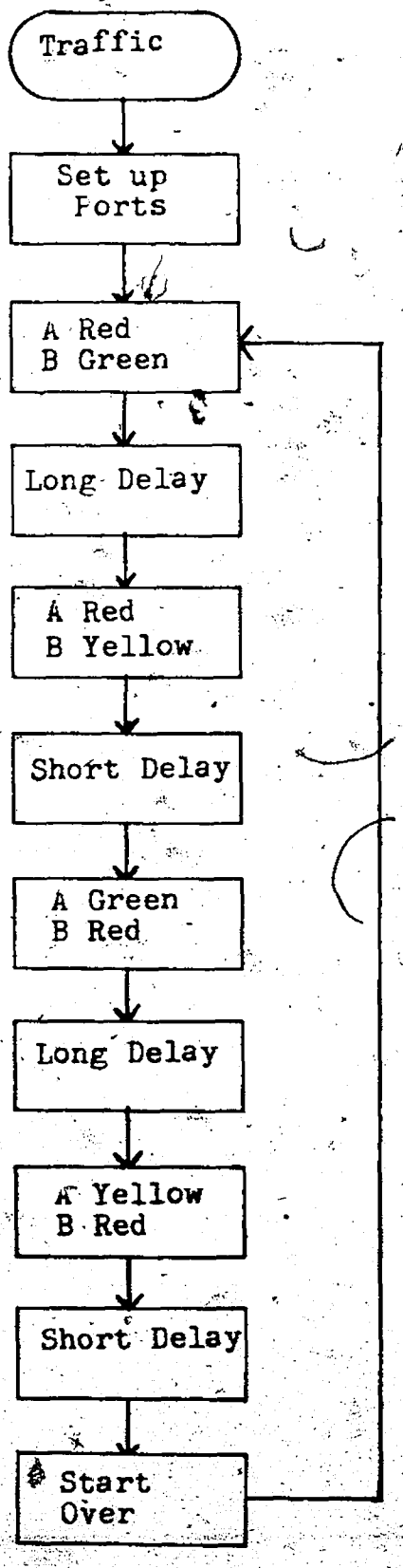
Action



Decision

Flow chart symbols used in the following activities

Flow Chart


Traffic
Set up Ports
A Red
B Green
Long Delay
A Red
B Yellow
Short Delay
A Green
B Red
Long Delay
A Yellow
B Red
Short Delay
Start Over

Program:

In order to output data we have to set up the ouptut ports.

This can be done with the program "set up", see B16.
Locate the "set up" program at memory location 0420.

Port A will become the output port and the pin connections on edge connector J1 are as follows:

Pin 9 - bit 0 - will light A green
Pin 10 - bit 1 - A yellow
Pin 11 - bit 2 - A red
Pin 12 - bit 3 - Not used
Pin 13 - bit 4 - B green
Pin 16 - bit 5 - B yellow
Pin 17 - bit 6 - B red
Pin 15 - bit 7 - Not used

Connect these outputs to the LED monitors on the logic board.

Main Traffic Light Program

Address	Program	Comments
0200	JSR 0420	Set up ports
0203	LDA #14	A red, B green
0205	STA A000	Output data
0208	JSR 0400	Green delay
020B	LDA #24	A red, B yellow
020D	STA A000	
0210	JSR 0405	Yellow delay
0213	LDA #41	A green, B red
0215	STA A000	
0218	JSR 0400	
021B	LDA #42	
021D	STA A000	
0220	JSR 0405	
0223	JMP 0200	Go to address 0200
0226	BRK	

Enter the <u>Delay program</u> listed below.

Execute:

Load the above program into the computer.

Press ESC 0200 C/R
Press G C/R

Check the light sequence, if you have a probem recheck the main program and subroutines.

Delay Program

Address	Program	Comments
0400	LDA #A0	Start green delay
0402	JMP 0407	
0405	LDA #10	Start yellow delay
0407	STA 00	
0409	LDY #FF	Delay
040B	LDX #FF	Delay
040D	DEX	
040E	BNE 040D	
0410	DEY	
0411	BNE 040B	
0413	DEC 00	
0415	BNE 0409	
0417	RTS	

The following exercise is designed to help you understand how the delay times are determined.

1 List, in a vertical column, all the instructions used in the delay program.

2 Look up, in the R6500 programming manual, the number of cycles required for each instruction.

3 Manually follow through the delay program to determine how many times each instruction is used.

4 Multiply the number of cycles used for each instruction by the number of times that instruction is used by 1E-6 to get total time in seconds.

5 Total all individual times to get total delay time.

Example using yellow delay

Instruction	Cycles	Times used	Total time
DEX implied mode	2	(256*256*17)	2.28 sec
BNE relative	3	(257*257*17+17)	3.36 sec

Complete the table using the other instructions. If you do not understand how the above times were calculated ask your teacher for assistance.

Change the delay time for the yellow and green lights and execute the traffic light program.

Write a program that will have different delay times for A and B green.

Do the same for A and B yellow.

Can you think of a better way to write this program?

How about asking the operator to input the delay program time before the main program is executed.

Do you think you could sell this program to the Town of Truro? They seem to have a big problem with their traffic light timing.

Most of the skills you have learned in this exercise are directly applicable to industry.

Some examples are: Process control, Burgular alarms, Electronic locks etc.

Learning Activity B22

Simulation of a monostable multivibrator

Objective: To simulate a monostable multivibrator with the AIM 65.

A monostable multivibrator (MMV) is a switching circuit that has one stable state and one quasi stable state. That is, the MMV is normally in one condition, i.e. the output is Lo. When an input pulse is received, it's output switches to a Hi for a predetermined time and then switches back to a Lo. It stays in this condition until another input pulse is received.

Monostable multivibrators can also be designed so that the output is normally Hi and it's quasi stable state is Lo.

Problem:

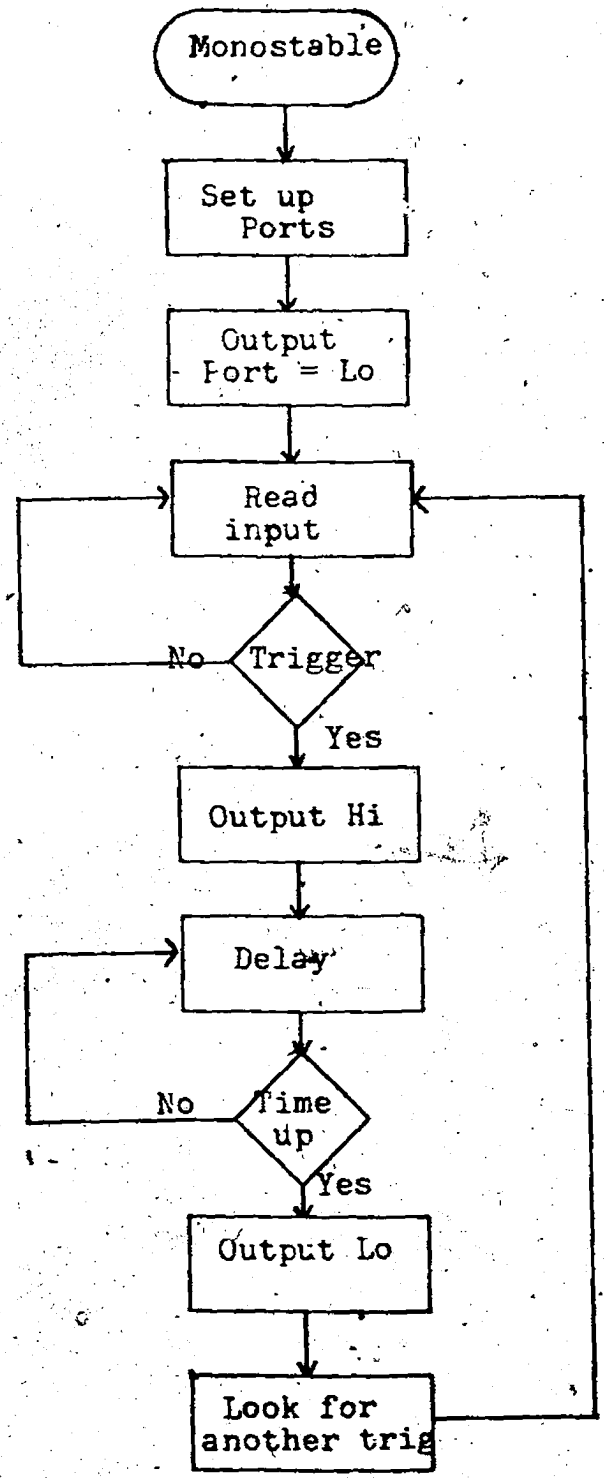
Write a program to simulate a monostable multivibrator with output normally Lo and it goes Hi for 2 milliseconds when it receives an input pulse or trigger.

Solution:

```
Output Lo Port B bit 0
Trigger pulse Port A bit 0
Output Hi Port B bit 0
Delay 2 milliseconds
Output Lo port B bit 0
```

Flow chart on the next page.

Flow Chart:


Monostable
Set up Ports
Output Port = Lo
Read input
No
Trigger
Yes
Output Hi
Delay
No
Time up
Yes
Output Lo
Look for another trig

Program:

```
0200  JSR 0420
0203  LDA #00
0205  STA A000      Port B bit 0 Lo
0208  LDA A001
020B  AND #01
020D  BEQ 0208      Look for trigger
020F  LDA #01
0211  STA A000      Output Hi
0214  JSR 0500      Timing routine
0217  LDA #00
0219  STA A000      Output Lo
021C  LDA A001
021F  AND #01
0221  BNE 021C      Trigger gone?
0224  JMP 0208
```

"Set up" subroutine

```
0420  LDA #00
0422  STA A003
0425  LDA #FF
0427  STA A002
042A  LDA #00
042C  STA A00B
042F  STA A00C
0432  STA A00E
0435  RTS
```

"DELAY" subroutine

```
0500  LDY #02
0502  LDX #C0
0504  DEX
0505  BNE 0504
0507  DEY
0508  BNE 0502
050A  RTS
```

Execute:

Execute the program, use Hewlett Packard logic probe to monitor Port B bit 0. Use a logic switch or pulse to input trigger on Port A bit 0.

Change the monostable width to .5 milliseconds and execute the program. Show the results to your teacher.

Learning Activity B23

Simulation of a D LATCH

Objective: To simulate a D Latch with the AIM 65.

Operation of a D-type flip flop (latch):

Information at the input is transferred to the output on the positive edge of a clock pulse. After data has been clocked in, further input data is blocked until another clock pulse is received.

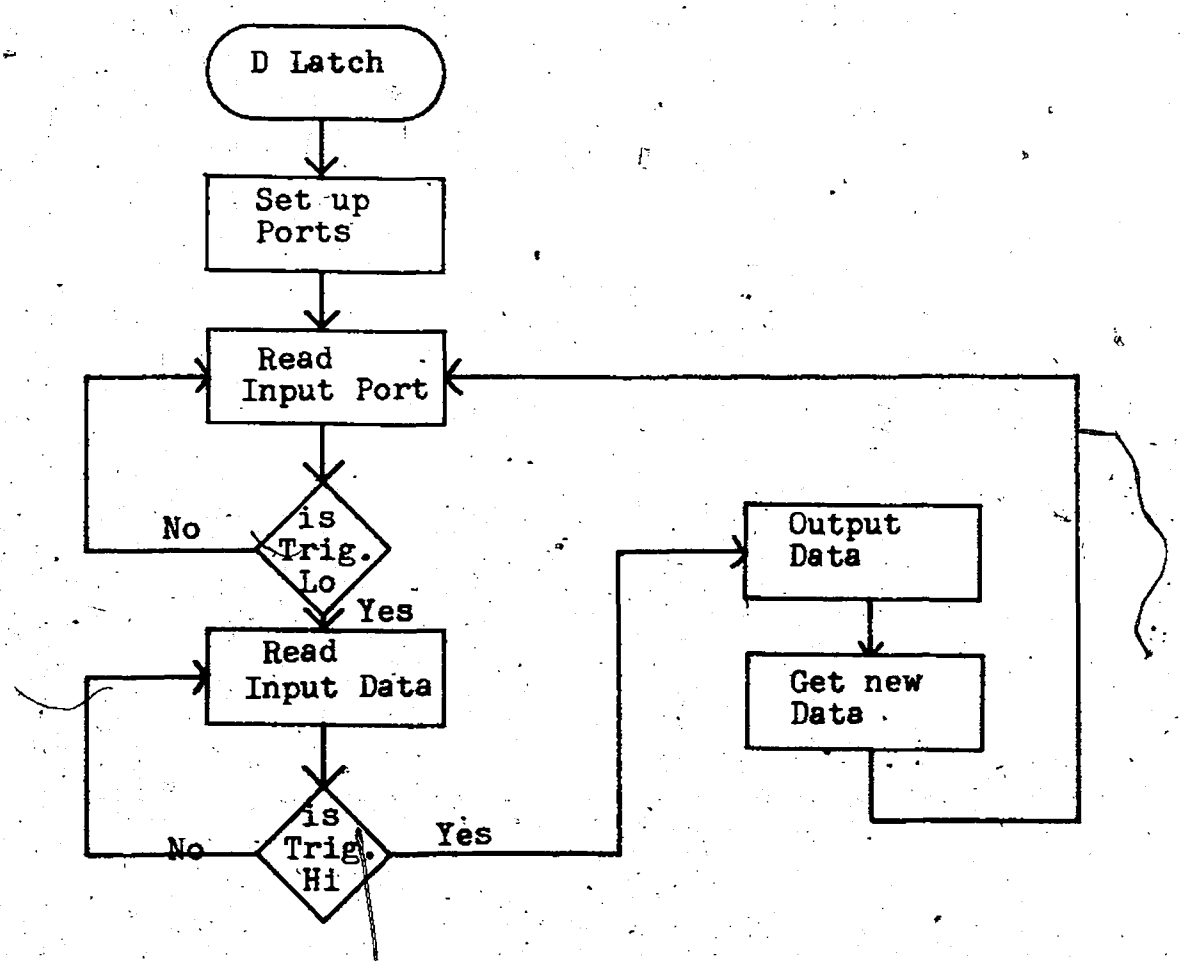
Problem:

Write a program to simulate a D-type flip flop. Data must be transferred when a clock pulse rises from Lo - Hi.

Solution:

Clock pulse in on Port A, Bit 0
Data pulse in on Port A, Bit 1
Data transfer when pulse goes from Lo to Hi
Data out on Port B, Bit 0
Look for new clock pulse.

Flow Chart:



Program:

```
0200  JSR 0420    Set up ports
0203  LDA #01
0205  STA 01
0207  LDA A001
020A  AND 01      Isolate Bit 0
020C  BNE 0207    Is Bit 0 Lo?
020E  LDA A001
0211  STA 02
0213  AND 01      Isolate Bit 0
0215  BEQ 020E    Is Bit 0 Hi?
0217  LDA 02
0219  AND #02     Isolate Bit 1
021B  STA A000    Output data on Port B, Bit 1
021E  JMP 0207    Get new data
```

"Set up" subroutine:

```
0420  LDA #00
0422  STA A003
0425  LDA #FF
0427  STA A002
042A  LDA #00
042C  STA A00B
042F  STA A00C
0432  STA A00E
0435  RTS
```

Execute program, use LED to monitor Port B, Bit 1. Logic switches on Port A, Bit 0 and 1 to input data and trigger.

Eight D latches in parallel can serve as a register. It may appear difficult to simulate an 8 bit register (eight lines plus trigger are required); however, a 4 bit register should be relatively easy.

Write a program that will allow the AIM 65 to simulate a 4 bit register.

Learning Activity B24

Simulation of a BCD to 7 segment DECODER

Objective: To write a program that will allow the AIM 65 to simulate a 7 segment Decoder.

For more information on a BCD to 7 segment Decoder, see Fairchild Data Book, Page 4-44.

Problem:

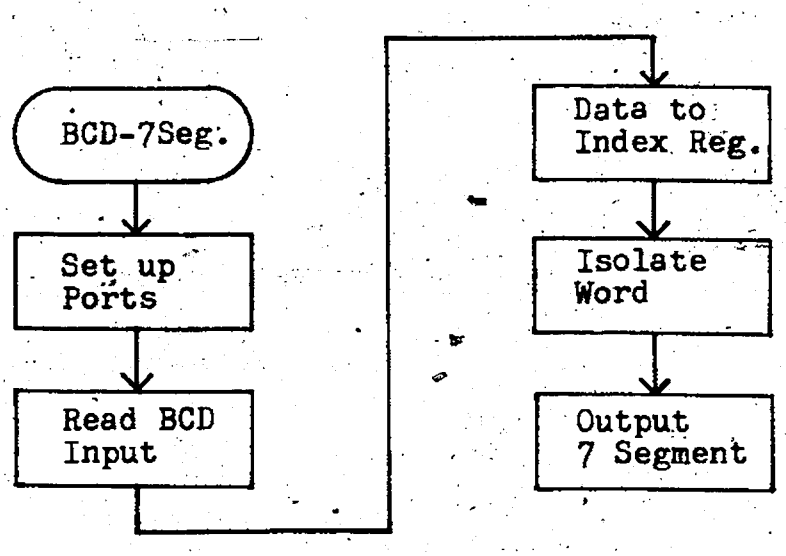
Write a program to decode BCD to seven segments.

Solution:

Set Port A input and Port B output
For 0 input, output 01
For 1 input, output 47
For 2 input, output 12
For 3 input, output 06
For 4 input, output 4C
For 5 input, output 44
For 6 input, output 60
For 7 input, output 0F
For 8 input, output 00
For 9 input, output 0C

Port B output connections for 7 segment	Port A input connections for BCD
Bit 0 = g	Bit 0 = LSB
Bit 1 = f	Bit 1 = LSB + 1
Bit 2 = e	Bit 2 = LSB + 2
Bit 3 = d	Bit 3 = MSB
Bit 4 = c	
Bit 5 = b	
Bit 6 = a	

Flow Chart:



Program:

```
0200  LDA #00
0202  STA A003
0205  LDA #FF
0207  STA A002
020A  LDA A001
020D  AND #0F
020F  TAX
0210  LDA 10, X
0212  STA A000
0215  BRK
```

Set up the following memory locations with the corresponding 7 segment data:

```
0010  01
0011  47
0012  12
0013  06
0014  4C
0015  44
0016  60
0017  0F
0018  00
0019  0C
```

Execute:

Connect LED on Port B
Connect Logic switch on Port A

Run the program.

Compare the output to the Truth Table you developed in Combinational Logic Circuits for a BCD to 7 segment Decoder.

Activity: The output of Port B will not sink enough current to drive a 7 segment readout. Interface Port B output and the 7 segment readout with a buffer driver. See TTL Data Book for buffer selection.

Demonstrate your simulated BCD to 7 segment readout to your teacher.

Learning Activity B25

HARDWARE INTERRUPTS

Objective: To write and execute a program illustrating the use of interrupts.

INTERRUPT:

Attention signal sent from an I/O device or chip to the MPU to obtain service. When accepted, the Interrupt results in halting the MPU which preserves its internal registers and branches to the appropriate interrupt service routine. Program execution resumes upon completion of the interrupt service routine.

INTERRUPT SERVICE ROUTINE:

Program that is executed when an interrupt occurs.

INTERRUPT MASK:

Register that has one bit to control each interrupt. Used to selectively disable specific interrupts.

POLLING:

Scheduling techniques for I/O devices, where the program interrogates in turn (the status of) each peripheral, and gives service when required.

INTERRUPT VECTOR:

An Interrupt Vector is simply an address that is loaded into the program counter when an interrupt occurs.

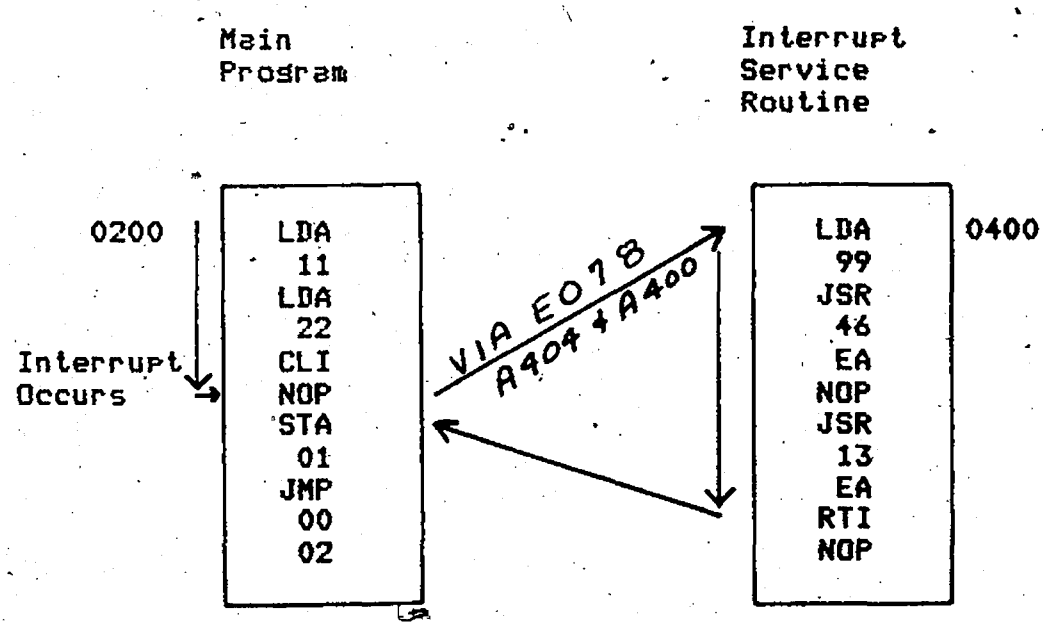
A computer normally sequences through a program responding to the MPU instruction set. Since the computer can handle only one instruction at a time, how is it possible that it can play a game and check and adjust the room temperature at the same time?

One method is to periodically jump from the main routine and check the port monitoring the temperature. This could be done several times a second; however, if no change has taken place, it should be clear that this method wastes CPU time.

A preferred method would be to have the I/O device "interrupt" the main program only when a correction to the temperature is required. The MPU will then branch to a routine to adjust the temperature and then return to the main program.

Interrupts allow the computer to seemingly perform several operations at the same time.

The following sequence of events illustrate how Interrupts affect the flow of the program:



A Low appears on the Interrupt request line (IRQ).

1. The current instruction being processed by the CPU is completed.

2. The interrupt flag is checked and if set, no interrupt occurs; if not set, an interrupt occurs.

3. The CPU registers are pushed into the stack.

4. The interrupt request flag in the processor status register is set.

5. The lower byte of the program counter (PC) is loaded with the contents of FFFE (78).

6. The upper byte of the program counter is loaded with contents of FFFE (E0).

7. E078 contains a jump to A404.

8. The computer proceeds to service the routine starting at the address in location A400.

9. The last instruction in the routine is a RTI (return from interrupt). This causes the contents of the stack, previously dumped from the CPU, to return to the CPU.

10. Normal programming continues.

Load "Do Nothing" program into memory starting at location 0200:

```
0200        LDA #11
0202        LDA #22
0204        LDA #33
0206        LDA #44
0208        LDA #55
020A        STA  00
020C        STA  01
020E        STA  02
0210        CLI      (Clear interrupt flag, set
                 ready for interrupt).
0211        JMP 0200
```

Set A400 to 0400.

Load interrupt routine into memory starting at location 0400.

```
0400        LDA #99
0402        JSR EA46    (output data)
0405        NOP
0406        JSR EA13    (output CR to print data)
0409        RTI
```

Set PC to 0200. Press G to execute the program.

Momentarily Ground IRQ Pin 4 on J3. The computer will service the interrupt and print at 99.

Review the above program several times, execute it, make sure you understand each instruction and its purpose.

Write a program that will print out the data on port A when the IRQ in momentarily grounded.

The NMI (non maskable interrupt) and the RES (reset) lines perform as similar operation to the IRQ. For more information see your manual.

Is there some way for Port A to signal the MPU that it has data ready and that it should be read in?

Learning Activity B26

BREAK

Objective: To write and execute a program using the BRK instruction.

The BRK instruction operates like an interrupt. When a BRK is encountered the program counter and the processor status register, after the B flag has been set, are pushed into the stack. The program then branches to the same address as the IRQ interrupt, that is the address (E154) at location A404.

The status register is pulled from the stack, pushed back into the stack, and flag B is tested to differentiate a BRK from an IRQ.

The normal break program (at location E163) dumps the program counter-1 and the associated instruction onto the display and returns to the monitor. The following program illustrates the BRK instructions:

```
0200    LDA #FF
0203    STA 00
0204    BRK
0205    LDA 00
0207    JSR EA46
020A    JSR EA13
020D    JMP 020D
```

Note: Make sure A404, A405 contains 54, E1.

Set program counter. Press G. Program runs, dumps PC and the associated instruction.

Press G. Program continues until complete.

The BRK is normally used when debugging a program.

Execution continues until it reaches a BRK, the PC and associated instructions are dumped. Pressing G causes program execution to continue.

Write a program that includes 3 BRK instructions. Run the program.

Modifying the address contained at A404 will allow you to write your own service routine. This routine will start at the memory location you load into A404, A405.

The BRK instruction is sometimes referred to as a software interrupt.

Learning Activity B27
INTERRUPTS using the VIA

Objective: To interrupt program sequence using the VIA Interrupt.

As stressed previously, computers essentially do two things, process data and input/output data. The AIM 65 inputs and outputs data via the VIA (versatile interface adapter). The VIA also allows the computer to communicate to other computers using the VIA and handshaking signals.

Handshaking:

Control signals at an interface in which the sending device generates a signal indicating that new information is available, and the receiving device that responds with another signal indicating that the data has been received.

The interrupt line IRQ is an essential part of the handshaking signals. The sending device sends out a signal to CA1 on the VIA, see Fig.1. In order to interpret this signal, the "interrupt enable register", bit 1, must be set to a 1. The peripheral control register, bit 0, must be set to a 1 so that a negative transition on CA1 will set bit 1 of the interrupt flag register and a low will be placed on the IRQ line.

The low on the interrupt line will cause an interrupt in the flow of the main program and the interrupt routine will be serviced.

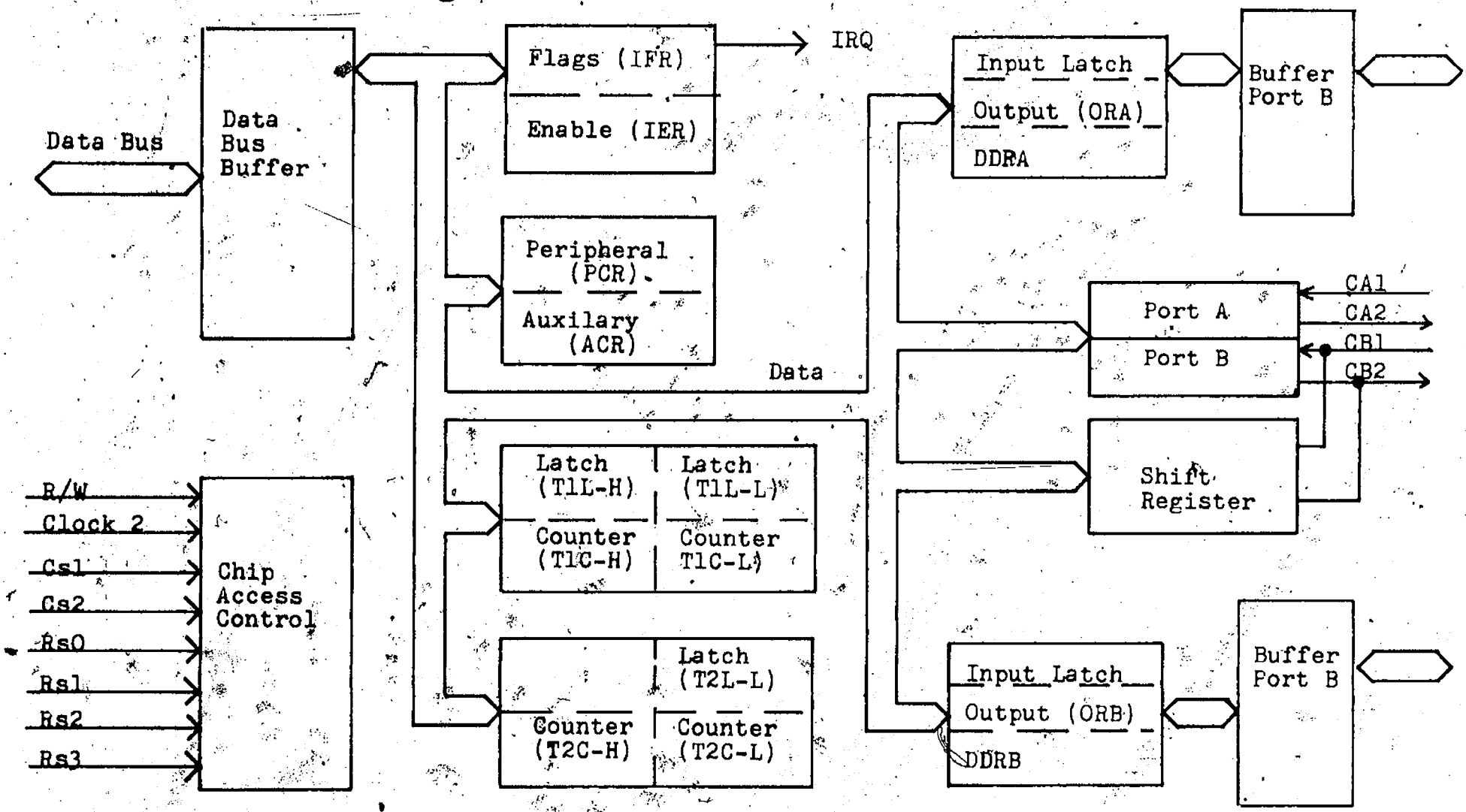

Data Bus
Data Bus Buffer
Flags (IFR)
Enable (IER)
IRQ
Peripheral (PCR)
Auxilary (ACR)
Data
Input Latch
Output (ORA)
DDRA
Buffer Port B
Port A
Port B
CA1
CA2
CB1
CB2
Shift Register
R/W
Clock 2
Cs1
Cs2
Rs0
Rs1
Rs2
Rs3
Chip Access Control
Latch (T1L-H)
Latch (T1L-L)
Counter (T1C-H)
Counter T1C-L)
Latch (T2L-L)
Counter (T2C-H)
Counter (T2C-L)
Input Latch
Output (ORB)
DDRB
Buffer Port B

Figure 1

The following program will set up Port A to recognize a "data ready" signal, initiate an interrupt, read the data at Port A, and generate a "data taken" signal. The "data taken" signal is automatically generated by a read statement to Port A if bit 1, 2 and 3 of the peripheral control register are set to 1, 0, 1 consecutively.

The address of the Peripheral Control Register (PCR) is A00C. The address of the Interrupt Enable Register (IER) is A00E. Port A is automatically set as an input on reset.

```
0200  LDA #82
0202  STA A00E    Set Bit 1 on the IER
0205  LDA #0A
0207  STA A00C    Set Bit 1 and 3, clear Bit 0 and 2
                  on the PCR
020A  CLI         Clear interrupt flag in the
                  accumulator
020B  NOP
020C  NOP
020D  NOP         Waiting for interrupt
020E  NOP
020F  JMP 020B
```

Load 0400 at address A400. This sets the location of the interrupt routine.

```
0400  LDA A001    Get data at Port A
0403  JSR EA46    Print data
0406  JSR EA13    C/R
0409  RTI         Go back and set up Port A
```

Ground CA1, pin 20 on J1.

Load the program into memory.

Set the PC to 0200 Press G C/R

Each time CA1 is interrupted, i.e., removed from ground and replaced, the interrupt routine is serviced and the data at Port A is printed on the Display and Tape. Try changing the data at Port A (grounding some of it's input) and running the program again.

You can observe the "data taken" signal at CA2, Pin 21 on J1, with a logic probe containing memory. This pulse, "data taken", is only one microsecond long.

Write a similar routine but rather than printing out the data, output it to Port B. Connect Port B output to the LED Monitor on your logic tester.

To set Port B as an output port:

```
LDA #FF
STA A002
```

This could be written in your program just before you output the data to Port B.

Perform the same operation using Port B as an input Port and Port A as the output Port. Read pages 6-22 to 6-28 in the R6500 Hardware Manual that accompanies the AIM 65 or interpret the information for the PCR and IER on the Rockwell AIM 65 Summary Card.

Learning Activity B28

COMPUTER as a SHIFT REGISTER

Objective: To simulate a Shift Register with the AIM 65.

The VIA has a shift register which can be used to convert data between serial and parallel forms.

Auxiliary control register, address A00B, bits 2, 3 and 4 control this register as shown in the following table.

Bit 4	Bit 3	Bit 2	Mode
0	0	0	Shift register disabled
0	0	1	Shift in under control of timer 2
0	1	0	Shift in under control of clock 2
0	1	1	Shift in under control of external clock
1	0	0	Free running output at rate determined by timer 2
1	0	1	Shift out under control of timer 2
1	1	0	Shift out under control of clock 2
1	1	1	Shift out under control of external clock

Program to shift out 8 bits from memory location 40 under control of clock 2.

```
041A  LDA #00
041C  STA A00B    Shift register disabled
```

```
041F  LDA #18
0421  STA A00B    Set shift resister for shift out
                  under clock 2
0424  LDA 40      Load accumulator with contents of
                  memory location 40
0426  STA A00A    Start shift out of data
0429  LDA A00D    Check for flag to indicate data
                  shifted out
042C  AND #04
042E  BEQ 0429
0430  JMP 041A
```

Load AA into memory location 0040.

Connect oscilloscope to CB2.

Execute program.

Observe a series of AA pulses on CB2. These are shifted out from memory locaton 40 by the AIM 65 shift resister.

Change data in memory location 40 and execute again.

Program to shift in 8 bits under control of Phase 2 clock and store the data in memory location 40.

```
03B8  LDA #00
03BA  STA A00B    Shift resister disabled
03BD  LDA #08
03BF  STA A00B    Shift resister in under clock 2
03C2  LDA A00A
03C5  LDX #04
```

```
03C7  DEX          Set up timing loop
03C8  BNE 03C7
03CA  LDA #00
03CC  STA A00B     Disable shift register
03CF  LDA A00A     Get data
03D1  STA 40
03D3  BRK
```

Lo on CB1.

Execute program.

Check memory location 40. Data .....

Hi on CB1.

Execute program.

Check memory location 40. Data .....

Is the above data correct? Explain.

Learning Activity B29

Alarm Program

Objective: To write a program that will monitor 14 inputs and control 2 outputs.

Write a computer program that will allow the AIM 65 to monitor 8 smoke detectors, 2 bursular alarms and 4 temperature regulators. The AIM 65 will output a logic 1 on Port B bit 0 for an alarm. A logic 1 on Port B bit 1 will turn the furnace on and a logic 0 on the same port will turn the furnace off.

Design an interface for the smoke detector and bursular alarm so that the AIM 65 input port will sense a logic 1 for an alarm.

Design an interface so that an output from port B bit 0 will sound an alarm.

Design an interface so that the output from port B bit 1 can turn off and on a furnace.

Learning Activity B30

Bell Program

Objective: To write a computer program for the AIM 65 that will simulate the bell ringing in your school.

Write a program that will output a pulse on port B bit 0 for 3 seconds each time the bell should ring in your school. This program should repeat every 24 hours.

Design an interface so that the data at port B bit 0 will close a relay as long as bit 0 is a logic 1.

IV INTERFACING

A Serial

B Parallel

C Uart

D Analog - Digital and Digital - Analog

E Software interface (Z-80)

F PIA, VIA (6502)

A computer performs essentially two functions,

1. Process data, 2. input/output data. This section deals with input/output data.

Almost all microprocessors use the same busses for both memory and input/output transfers. Two methods are used to distinguish memory data and address from input/output data and address.

1. Isolated input/output (I/O) in which memory and I/O addresses are decoded separately. Typical microprocessors that use isolated I/O are Intel 8080 and Zilog-Z80.

2. Memory - mapped input/output in which I/O ports are treated exactly the same as memory locations. Typical microprocessors that use memory - mapped I/O are Motorola 6800 and Rockwell 6502.

Figure 1 shows a microcomputer with an isolated I/O section. The signal select line will select I/O address and data when the line is driven Hi; a Lo will select memory address data for the microcomputer.

Features of isolated I/O:

1. Short addressing instructions (OUT 01, A)

2. Programs are clearer using I/O instructions

3. Requires extra decoding and machine instructions for I/O

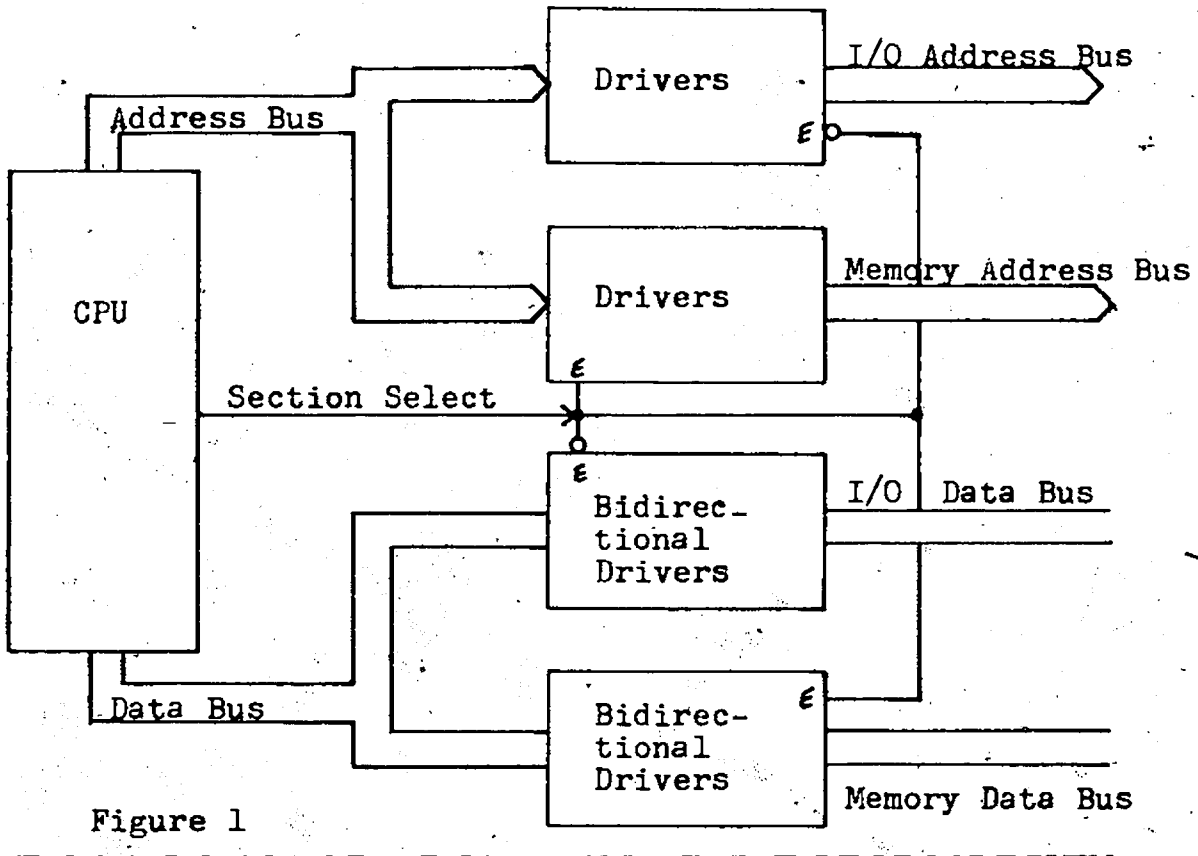


Figure 1

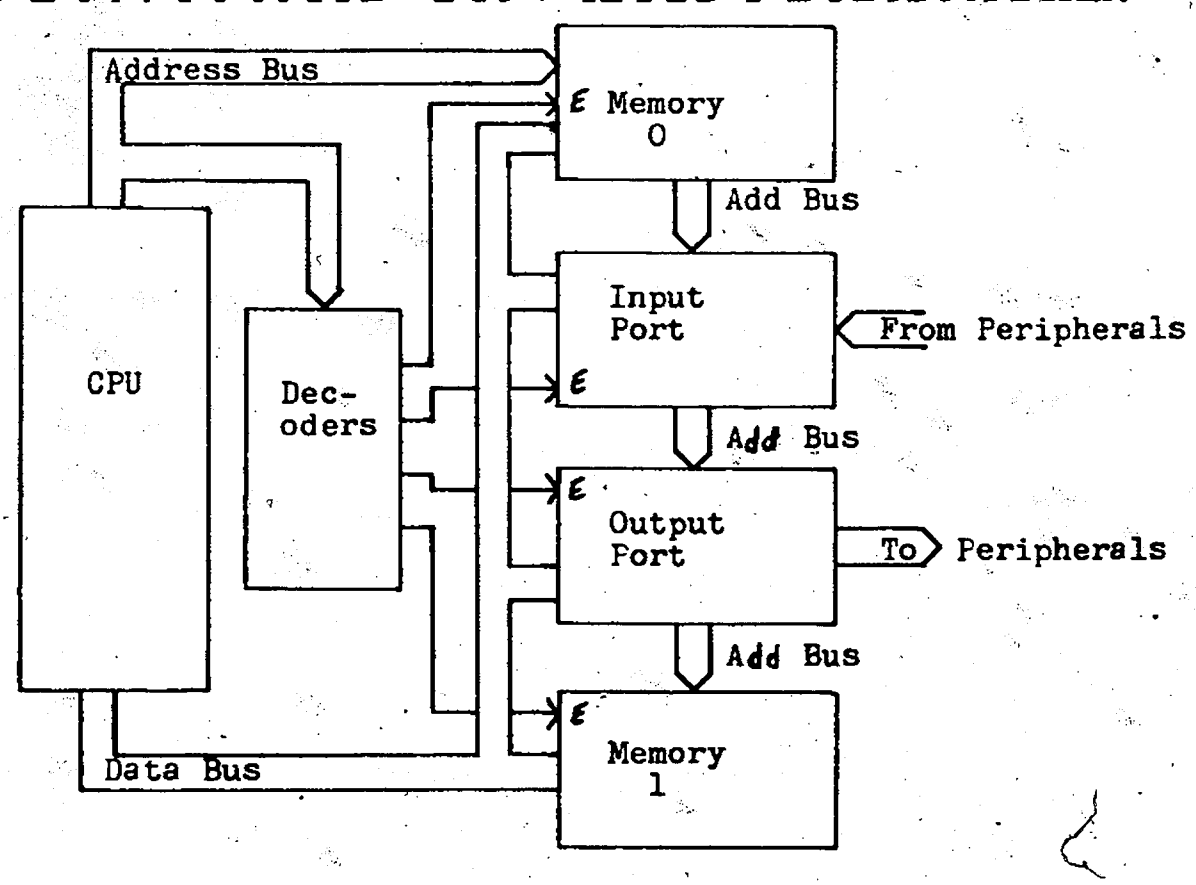


Figure 2

Figure 2 shows a microcomputer with memory - mapped I/O. The processor uses the same instructions for memory and I/O transfers. Specific addresses must be set aside for I/O transfer of data. The AIM 65 reserves A000 to AFFF for I/O.

Features of memory - mapped I/O:

1. Difficult to distinguish between complex I/O instructions and memory instructions.

2. I/O ports use up memory address space. 4 K of memory is used up for addressing in the AIM 65.

3. Incorporates LSI devices (PIA, VIA) that aid in interfacing external devices.

4. I/O chips may be difficult to program.

Usually memory - mapped I/O is best suited to systems that use complex interface chips, whereas isolated I/O is better for systems that use small and medium - scale integrated circuits.

A Serial

A1 Serial Interface

A2 EIA Standard RS-232C Interface

A3 20 Milliamp Current Loop

Learning Activity A1

SERIAL INTERFACE

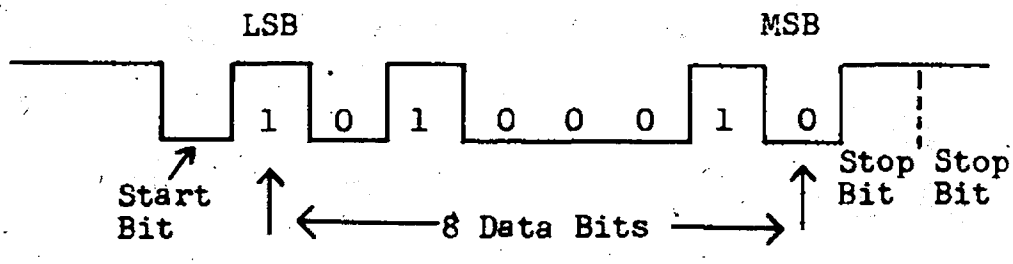
Objective: To write a program to convert serial data to parallel.

The computer processes data in parallel, in fact the data bus is 8 bits wide. Suppose you wish to output data to a printer; there are two possibilities:

1. Eight parallel lines between the printer and the computer plus a common line.
2. One line between the printer and the computer plus a common line.

In order for the eight bits to be transmitted or received over 1 line, the data must be converted from parallel to serial.

Normal serial transmission is as follows:



Each 8 bit character is converted to 11 serial bits, 1 start bit, 8 data bits, 2 stop bits.

Suppose the data transmission rate is 10 characters per second and there are 11 bits per character, then the transmission rate is 10 x 11 = 110 bits per second or 110 Baud. The width of each bit is 1/110 = 9.1 milliseconds.

This is standard TTY transmission.

For a computer to properly receive data from a TTY, the following procedures are necessary.

1. Locate the start bit. (Logical 0)

2. Centre reception on the start bit by waiting 1/2 time, i.e., 4.5 milliseconds.

3. Wait 9.1 milliseconds, i.e., centre on first data bit, the Least Significant Bit (LSB).

4. Shift that bit into the carry flag.

5. Wait 9.1 milliseconds, shift second bit into carry flag and first bit into computer word.

6. Continue shifting all 8 bits into the computer word via the carry flag.

7. When all 8 bits are shifted into a word, return from subroutine.

The following program will fetch serial data from a TTY via bit 7, Port A on the VIA and place the data in memory location 0060.

```
0400  LD #00
0402  STA A00C    All control lines input
0405  STA A003    Port A input
0408  LDA A001    Looking for start bit
040B  BMI 0408
040D  JSR 0500    4.5 millisecond delay
0410  LDA #80     MSB = 1
0412  JSR 0504    9.1 millisecond delay
0415  ROL A001    Load data in Bit 7 of A001, into
                  carry flag
0418  ROR A       Transfer data bit from carry flag to
                  the accumulator bit 7, Hi in bit 7
                  transferred to bit 6
0419  BCC 0415    Get another bit if flag = 0
                  Continue when the Hi that started in
                  bit 7 is shifted into the flag.
041B  STA 60
041D  RTS
```

Delay Routine:

```
0500  LDY #05
0502  BNE 0506
0504  LDY #0A
0506  LDX #B4
0508  DEX
0509  BNE 0508
050B  DEY
050C  BNE 0506
050E  RTS
```

Remember: Bit 0 of data is received first at Pin 7 on Port A.

Define the following:

BMI .....

ROL .....

ROR .....

BCC .....

The signals used in the above illustration are all TTL logic levels.

This program will be required for learning activity A3, be sure you understand the program logic.

Modify the above program so that the received data is displayed on the AIM 65 Display.

Learning Activity A2

EIA Standard RS-232C Interface

Objective: To interface TTL logic to RS-232.

The EIA standard interface between computers and peripherals consists of the following signals and circuits "from" and "to" data communications equipment (DCE) at the terminals:

Ground (as a basic refrence)
Common return
Transmitted data (to)
Received data (from)
Request to send (to)
Clear to send (from)
Data terminal ready (to)
Ring indicator (from)
Receive line signal detector (from)
Signal Quality detector (from)
Two Data-Rate selectors (to data terminal equipment or DTE source, and from data communications equipment or DCE source)
Two transmitter signal element timings (to DTE source and from DCE source)
Receiver signal element timing (from)
Secondary transmitted data (to)
Secondary received data (from)
Secondary request to send (to)
Secondary clear to send (from)
Secondary received line signal detector (from)

The various circuits determine the "hand shaking" needed to establish interface connections.

Usually we will be concerned with only the first eight, for example to send data from a computer to a printer only the following signals are required at the printer:

Received data
Data terminal ready
Signal common (may be attached to signal ground)

RS-232C signals are designed for serial transmission over long distances, up to 100 feet. These signals have the following voltage levels, compared to TTL logic, a high is -12 and a low is +12.

Since most computers and peripherals work at TTL logic levels the following circuits can be used to convert RS-232 to TTL logic levels and TTL to RS-232.

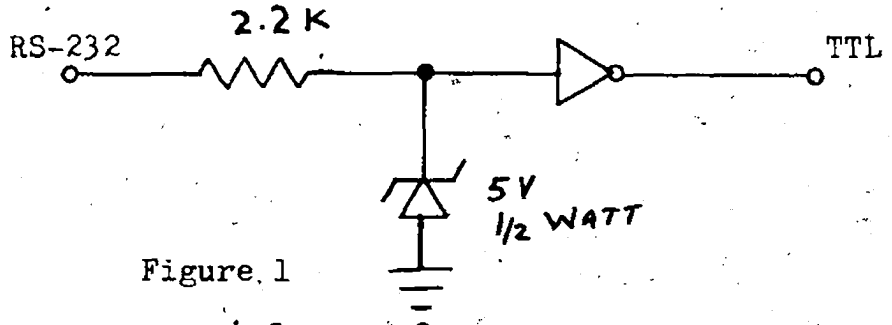


Figure 1

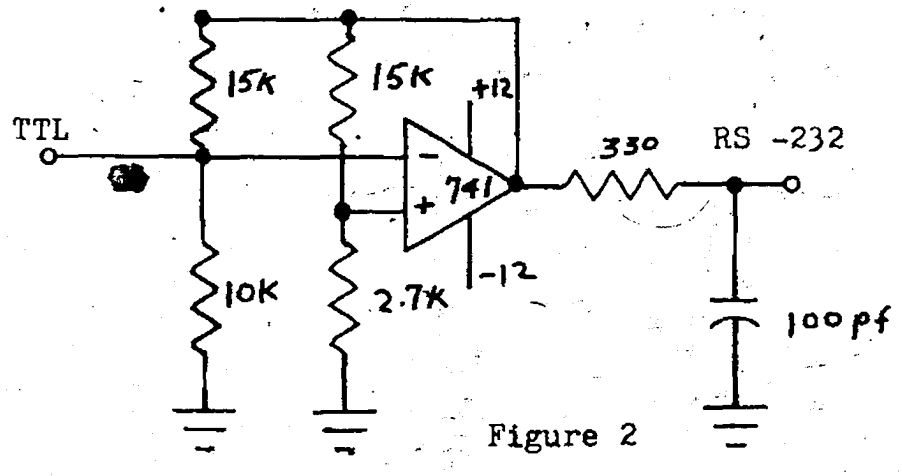


Figure 2

Wire up the above circuits, connect Fig. 2 RS-232 output to Fig. 1 Rs-232 input.

Place a logic 0 on the TTL input in Fig. 2

Fig. 2 RS-232 output = ...

Fig. 1 TTL output = ...

Place a logic high on the TTL input in Fig. 2

FIG. 2 RS-232 output = ...

Fig. 1 TTL output =.....

Design a circuit that will :

1 Accept data from the AIM 65 in serial form and convert it to EIA RS-232, this signal should be able to drive any RS-232 device (printer).

2 Accept a data terminal ready (DTA) signal from the RS-232 device and change it to TTL logic.

3 Write a program that will take data from sequential memory locations starting at 0200, output it to a RS-232 device and halt the output when it receives a DTR signal from the RS-232 device.

For additional information see Introduction to Microprocessors, Software, Hardware, Programming by Lance A. Leventhal, page 424-427.

Show the completed project to your teacher. He will have an RS-232 device you can use to test your program and interface.

Learning Activity A3

20 MILLIAMP CURRENT LOOP

Objective: To interface 20 Milliamp Current Loop to TTL.

The Model 33 ASR Teletype (TTY) communicates with computers through current loops. Current loops have the following characteristics:

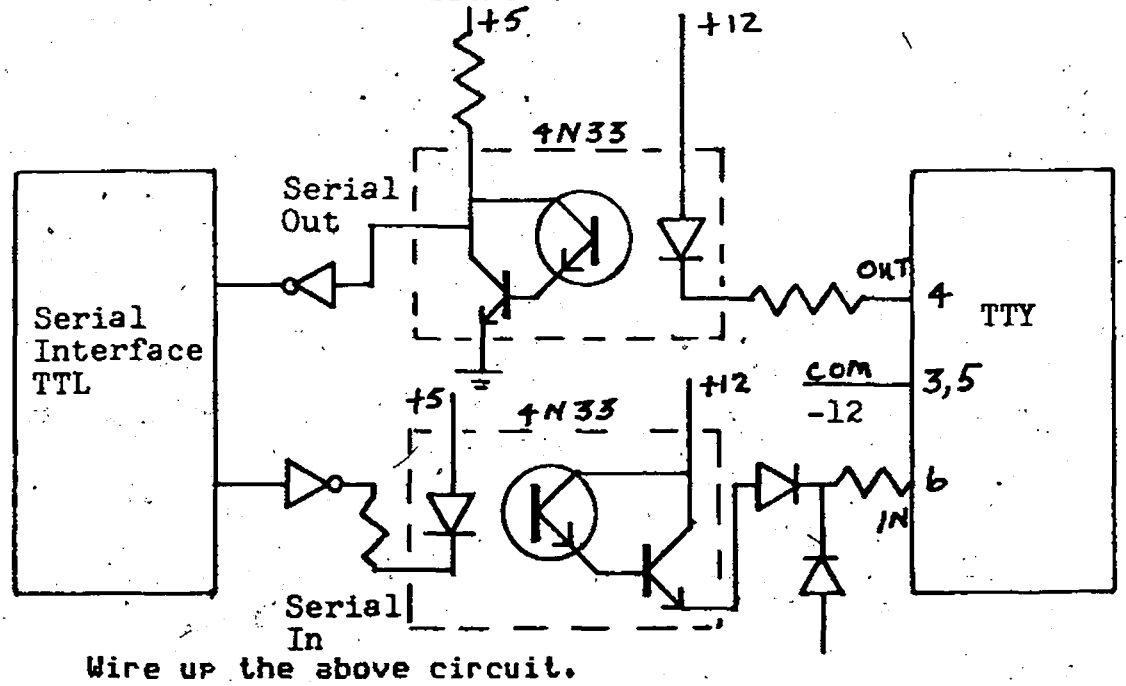
1. Logic 1 state: presence of current of approximately 20 mA in the current loop.

2. Logic 0 state: absence of current in the current loop.

Current loops are low impedance transmission lines that are highly resistant to noise. Digital signals can be transmitted via current loops for distances of up to a mile with no loss of information.

Interfacing the TTY 20 mA operation with TTL logic requires the use of opto-isolators.

Figure 1 is an illustration of such a circuit.



Wire up the above circuit.
Connect the serial out to the AIM 65.

Load a serial print program. (See learning activity A1)

Output data from the TTY to the AIM 65.
Show the communication to your teacher.

Learning Activity B1

PARALLEL INTERFACE (8 bit words)

Objective: To parallel interface the AIM 65 to a centronics printer.

Probably the easiest way to interface the AIM 65 is parallel. All the data lines are parallel, the memory is parallel and the VIA has two parallel output ports.

To parallel interface the AIM 65 to a printer, the following signals are required:

1. Eight data bits (Bit 7 is not normally used).

2. Strobe (Active on Hi to Lo transition).

3. Busy (Indicates the printer is not ready to receive data).

4. Signal ground.

Several other control lines may be used; however, these are considered a minimum.

Program to parallel interface printer to the AIM 65.

```
0400  LDA #00
0402  STA A00C    Set CB2 in pulse mode output
0405  LDA #00
0407  STA A003    Set up Port A as input
040A  LDA #FF
040C  STA A002    Set up Port B as output
040F  LDA A001
0412  AND #01     Mask off bit 0, to check busy
0414  BNE 040F    Check busy
0416  JSR E93C    Get data
0419  STA A000    Output data and strobe on CB2
041C  JSR E97A    Data to Display
041F  JMP 0400
```

Wire up the centronics printer to the AIM 65.
Use buffers to insure no damage to the printer. You will have to determine pin connections from the centronics manual and the above interface program.

Have the teacher check your wiring.
Load the program into the AIM 65 .
Execute the program.
Data printed on your display should now be printed on the external printer.

Improve the above interface by including the ACK (acknowledge) signal in your program.

Learning Activity B2

PET and the IEEE-488 Bus

PET and the IEEE-488 Bus is a book available from Osborne/McGraw-Hill, 630 Bancroft Way, Berkeley, California 94710.

This is the only complete guide available on interfacing PET to the IEEE-488 Bus.

From this book you will learn how to program the PET Interface to control power supplies, signal sources, signal analyzers and other instruments. It is full of practical information, as one of the authors assisted in the original design of the PET IEEE-488 interface.

The output ports of the PET are identical to the output ports on the AIM 65. Both use the VIA and PIA for interfacing. Unfortunately, this book, ordered January 1980 has not been received at the time of the publication of this document.

PET and the IEEE-488 will provide extensive interface learning activities for the student.

Learning Activity C1

UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER (UART)

Objectives: Demonstrate how to send an 8 bit binary word serially from the transmitter section to the receiver section of the UART.

Examine the behavior of the outputs at pins 19, 22, 24 and 25 on the UART chip.

Demonstrate how you can control the number of bits in the asynchronous character transmitted by a UART.

UART stands for Universal Asynchronous Receiver/Transmitter . Essentially it can receive parallel data and transmit it serial or receive serial data and transmit it parallel.

Complete experiments No. 1, 2 and 3 in the Busbook II by Larsen.
These exercises could be shared by a group of students.

Activity:

1. Wire up a UART (AY-5-1013) so that it will accept parallel data from the AIM 65 and transmit it serially.

2. Feed the serial data to another UART, at a remote location, output parallel data to drive a set of LED'S. Input characters from the AIM 65 keyboard and monitor the ASCII code on the LED'S.

Special Instructions:
One group set up number 1.
Second group set up number 2.

Demonstrate the finished product for your teacher!

For additional information see Microcomputing, April 1969, page 62.

Learning Activity D1

Digital to Analog

Objective: To construct a digital to analog converter.
To interface the digital to analog converter with the AIM 65.

A digital device deals with discrete voltage levels, either a Hi or a Lo.

Analog signals have voltage levels that vary over a wide range and can be either positive or negative.

A digital to analog converter (DAC) converts digital voltages to analog voltages or current.

Schematic diagram of a digital to analog converter.

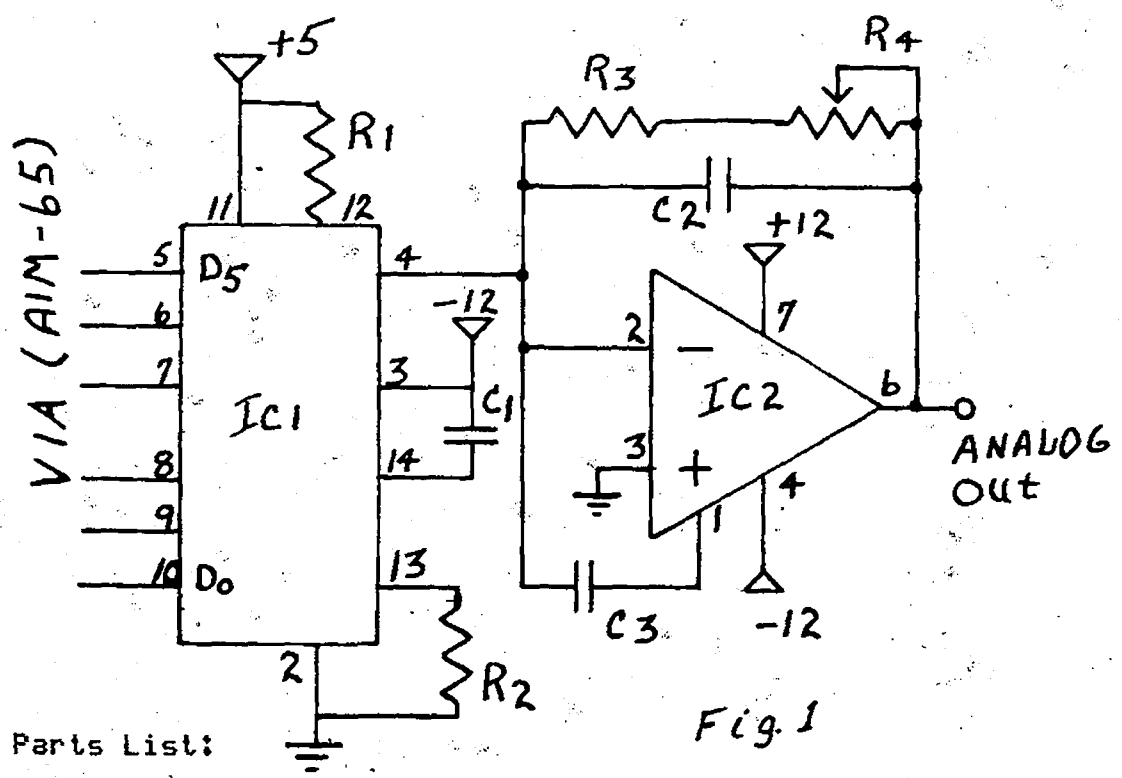


Fig. 1

Parts List:

IC1	MC1406 DAC
IC2	301 op amp
R1	2700 1/2 watt resistor
R2	2700 1/2 watt resistor
R3	2000 1/2 watt resistor
R4	1000 ohm varible resistor 1/2 watt
C1	100 picofarad capacitor
C2	47 picofarad capacitor
C3	100 picofarad capacitor

Wire up the circuit in figure 1.

Interface the AIM 65 and IC1 via the versatile interface adapter port B.

The following program is designed to control the output voltage of IC2, i.e. the analog voltage.

The maximum voltage will be set at +5 volts. This will occur when the data from the VIA is 00, remember only 6 bits will be used by the digital to analog converter.

Minimum output will occur when the data at port B is 3F, i.e. all 1's.

Program

0200	JSR 0400	Set up port B as an output port. See learning activity B 16.
0203	JSR E3FD	Input 2 characters from the Keyboard to the accumulator.
0206	STA A000	Output data to the DAC
0209	JMP 0203	Get 2 more characters

Connect a DC voltmeter between IC2 pin 6 and ground.

Execute the program.

Input 00 from the Keyboard.
Adjust R4 so that the voltmeter reads 5 volts.

Complete the following table.

Keyboard	Voltmeter
00	.....
10	.....
20	.....
30	.....
38	.....
3F	.....

Write up a short summary on how the data from the Keyboard is converted to an analog voltage.

Learning Activity D2

Digital to Analog

Objective: To interface the the digital output of the AIM 65 with an analog voltage.

The output of the digital computer in parallel mode is normally 8 bits. A simple digital to analog converter uses specifed data, usually in parallel form, to switch off or on AC analog devices.

Figure 1 is a typical AC power controller manufactured by DIEGO, Inc. Box 3009, Boulder, Colorado 80307.

Description of the circuit:

Data is applied, in parallel, to IC8 and IC11. When the correct address and a strobe is applied to IC1 and IC5 a trigger is developed at IC5 pin 8 which latches the data into IC8 and IC11.

If any one of the outputs of IC8 or IC11 is a logic 1 current will flow through the corresponding optoisolator and switch on the triac. An AC load must be placed in series with 120 VAC and across pins 1 and 2 of the triac.

Application

Place a 50 watt 120 V light bulb in series with 120 VAC. Connect one side of the AC to pin 1 of Q1 and connect pin 2 of Q1 to the opposite side of the AC load.

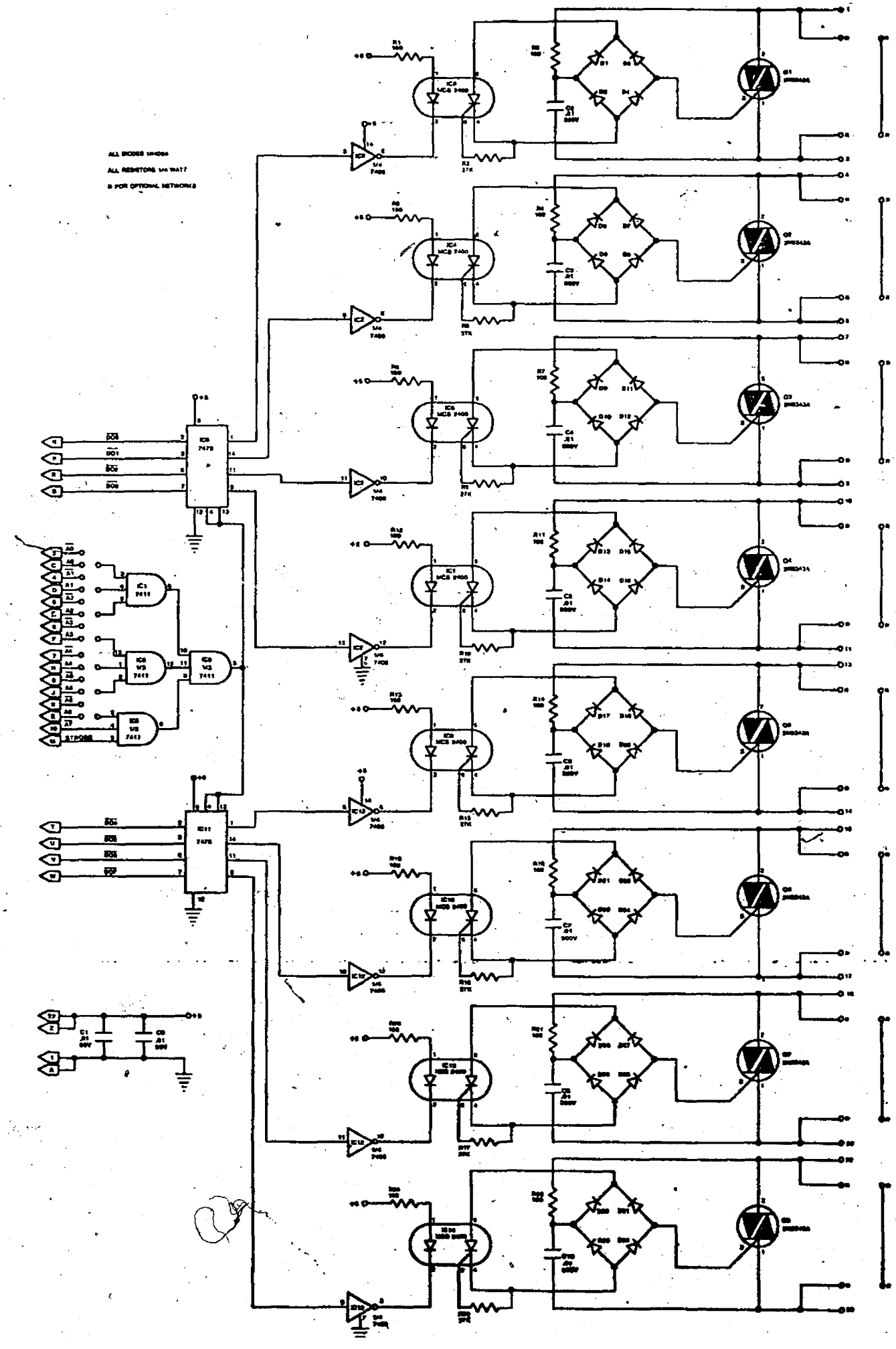
In figure 1 connect the following jumpers:

$\overline{A0}$ to pin 3 IC1
$\overline{A1}$ to pin 4 IC1
$\overline{A2}$ to pin 5 IC1
$\overline{A3}$ to pin 13 IC5
$\overline{A4}$ to pin 1 IC5
$\overline{A5}$ to pin 2 IC5
$\overline{A6}$ to pin 3 IC5

Connect the AIM 65 to the AC power controller.

Wire up Port B to the data lines
Wire up Port A to the address lines
Connect an inverter between CA2 and pin M (the strobe line)

AC POWER CONTROLLER



Program

```
LDA  #0A
STA  A00C      CA2 pulse mode output
LDA  #FF
STA  A003
STA  A002      Set up ports
LDA  #00
STA  A000
LDA  #01       Clear optoisolators
STA  A001
LDA  #01
STA  A000      Data to turn on Q1
LDA  #01
STA  A001      Latch data to optoisolators
```

Write a program that will alternately switch off and on Q1 and Q2. Connect a 50 watt light bulb in series with 120 VAC across each.

Execute the program. Show the result to your teacher.

Can you make the lights flash?

Learning Activity D3

Analog to Digital

Objective: To construct an analog to digital converter.

A. The 6502 Applications Book by Rodney Zacks contains a circuit that converts an analog voltage, change in thermistor resistance, to a digital voltage.

See page 206 to 215

Construct the circuit in figure 5-47, page 206.

Load in the program on page 210 and 211.

Execute the program.

Describe the program operation.

B. Microprocessors by Heathkit Continuing Education contains a program and circuit diagram that will convert an analog signal to a digital signal.

This circuit (figure 10-84) could be used to emulate a digital voltmeter, for additional information see Microprocessors page 10-108.

Construct the circuit in figure 10-84 and write a program so that the AIM 65 can simulate a digital voltmeter. See figure 10-85 for additional information. Being able to read the 6800 assembler program on page 10-109 will assist you in writing a program for the AIM 65 using mnemonic codes.

Learning Activity E1

SOFTWARE INTERFACE

Objective: To introduce a software interface.

Software program to output the contents of accumulator to a printer via port 3 and a strobe output on bit 3, port 1. This program can be used with a Intel 8080 or Zilog-80 MPU that employs isolated input/output.

```
0618  Push AF
0619  Push BC
061A  Push DE
061B  Push HL
061C  Push AF
061D  IN   01
061F  Bit  7,A     Check Port 1 Bit for busy signal
0621  JR   NZ,FA   If busy, read port again *061D*
0623  NOP
0624  POP  AF      Accumulator from stack
0625  OUT  03      Output data port 3
0627  XOR  A       Clear accumulator
0628  OR   F7      Clear bit 3
062A  OUT  01      Output Lo on bit 3, all other Hi
062C  SET  3,A     Bit 3 Hi
062E  OUT  01      Output strobe
0630  POP  HL      Pop stack
0631  POP  DE      Pop stack
0632  POP  BC      Pop stack
0633  POP  AF      Pop stack
0634  RET          Return from print subroutine
```

This subroutine will output data to drive a parallel printer. The requirements are:

1. When a print is required, the data to be printed is in the accumulator.

2. The main program calls up a subroutine at memory location 0618.

Write a similar subroutine for the AIM 65 so that it will print data on an external parallel printer whenever a special print is called, i.e., F1 on the Keyboard.

Learning Activity F1

Peripheral Interface Adapter (PIA)

Objective: To be able to set the Control Registers and Data Direction Registers of the PIA.

The PIA is an I/O device which acts as an interface between the microprocessor and peripherals such as printers, displays, keyboards, etc. It is comprised of two almost identical sections, A and B, each capable of receiving or transmitting eight bits of data.

See figure 1

CRA --> Control Register A
CRB --> Control Register B
ORA --> Peripheral Output Register A
ORB --> Peripheral Output Register B
DIR --> Data Input Register

For further information on the PIA, see Rockwell's R6502 Microcomputer data sheet on the PIA. Available from Rockwell International, Anaheim, CA 92803 USA.

Because of pin limitations the Data Direction Register (DDRA) and port A both have the same address, AC00. For DDRB and port B the address is AC02.

In order to read data from port A the following sequence of events must occur.

1 Control Register A (CRA) address AC01 bit 2 is reset to a logic zero, this means that address AC00 serves DDRA. If bit 2 is set to a logic 1 then AC00 serves Port A.

2 Reset all 8 bits of DDRA to logic 0 makes port A an input port.

3 Set CRA bit 2 address AC01 to logic 1, causes AC00 to serve port A.

4 Read AC00 i.e. read data at port A.

Typical assembler program to read data at port A.

```
LDA  #FB
AND  AC01
STA  AC01        CRA bit 2 logic 0
LDA  #00
STA  AC00
LDA  #04         Set data direction
ORA  AC01
STA  AC01        Port A active
LDA  AC00
```

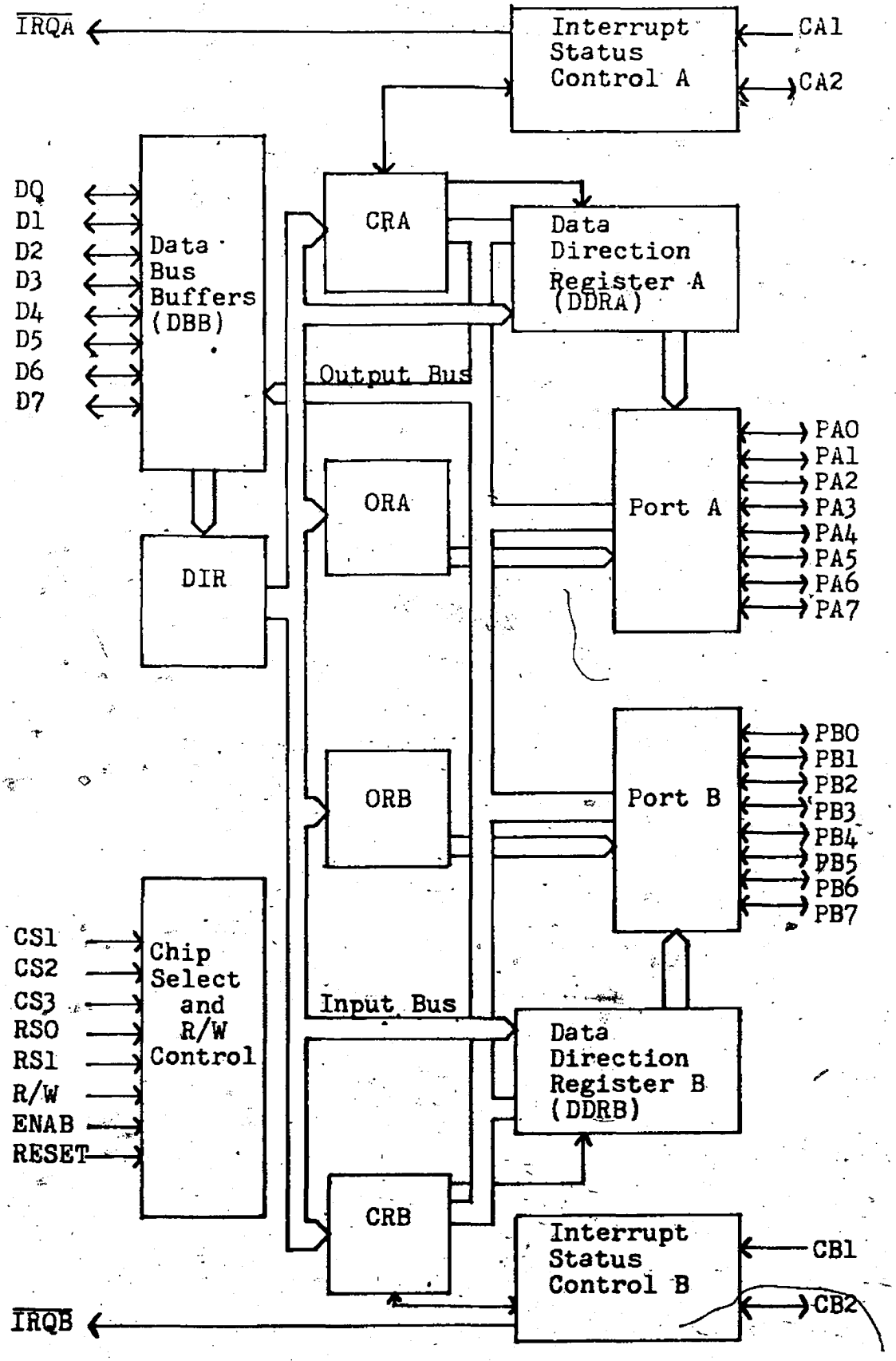

IRQA
Interrupt Status Control A
CA1
CA2
DQ
D1
D2
D3
D4
D5
D6
D7
Data Bus Buffers (DBB)
CRA
Data Direction Register A (DDRA)
Output Bus
ORA
Port A
PA0
PA1
PA2
PA3
PA4
PA5
PA6
PA7
DIR
ORB
Port B
PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
CS1
CS2
CS3
RS0
RS1
R/W
ENAB
RESET
Chip Select and R/W Control
Input Bus
Data Direction Register B (DDRB)
CRB
Interrupt Status Control B
CB1
CB2
IRQB

FIG. 1

Write an assembler program to read the data at port B.

AC02 is the address of DDRB and Port B
AC03 is the address of CRB

Write an assembler program that will output data to port B i.e, put all logic 1's in DDRB.

The following example will be used to illustrate some of the features of Control Register A (CRA) and Control Register B (CRB).

Example

1 Port A will be used for input

2 Port B will be used for output

3 Set up control register A and B for the following When data appears on port A and the peripheral signals the PIA that data is ready, the PIA (via CA1) will interrupt the CPU, read port A and generate a data taken signal.

4 After the CPU processes data it outputs the data via port B. Port B will generate a data available signal.

Solution

1 Write an assembler program to set port A as input.

2 Write an assembler program to set port B as output.

3 Bit 0 on CRA must be set to a logic 1 to generate an interrupt (IRQA) when CA1 goes Lo.
Bit 3 and 5 on CRA must be set to a logic 1 to generate a data taken signal on CA2 after port A is "read".

```
LDA  $29
STA  AC01      set bit 0,3,5 clear other bits
```

4 Bits 3 and 5 CRB must be set to logic 1 to generate a data available on CB2.

Combine 1,2,3, and 4 to write a program that will set CRA, CRB, DDRA and DDRB to accomodate the example above.

Use the memory examine function to check data in CRA, CRB, DDRA, DDRB.
Show your teacher the result.

Rockwell

R6500 Microcomputer System

DATA SHEET

PERIPHERAL INTERFACE ADAPTER (PIA)

DESCRIPTION

The R6520 Peripheral Interface Adapter is designed to solve a broad range of peripheral control problems in the implementation of microcomputer systems. This device allows a very effective trade-off between software and hardware by providing significant capability and flexibility in a low cost chip. When coupled with the power and speed of the R6500 family of microprocessors, the R6520 allows implementation of very complex systems at a minimum overall cost.

Control of peripheral devices is handled primarily through two 8-bit bidirectional ports. Each of these lines can be programmed to act as either an input or an output. In addition, four peripheral control/interrupt input lines are provided. These lines can be used to interrupt the processor or for "hand-shaking" data between the processor and a peripheral device.

Ordering Information

Order Number: R6520 — — —

Temperature Range:
- No suffix = 0°C to +70°C
- E = -40°C to +85°C (Industrial)
- MT = -55°C to +125°C (Military)
- M = MIL-STD-883, Class B

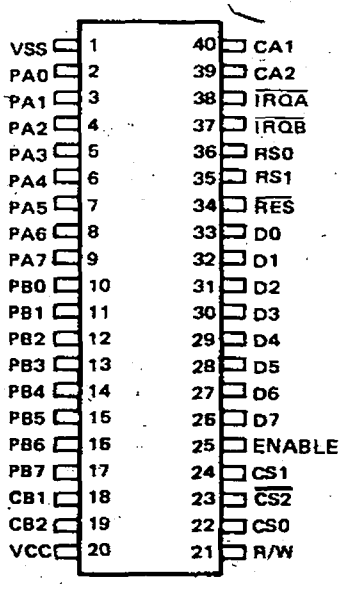
Package:
- C = Ceramic
- P = Plastic
- (Not Available for M or MT suffix)

Frequency Range:
- No suffix = 1 MHz
- A = 2MHz

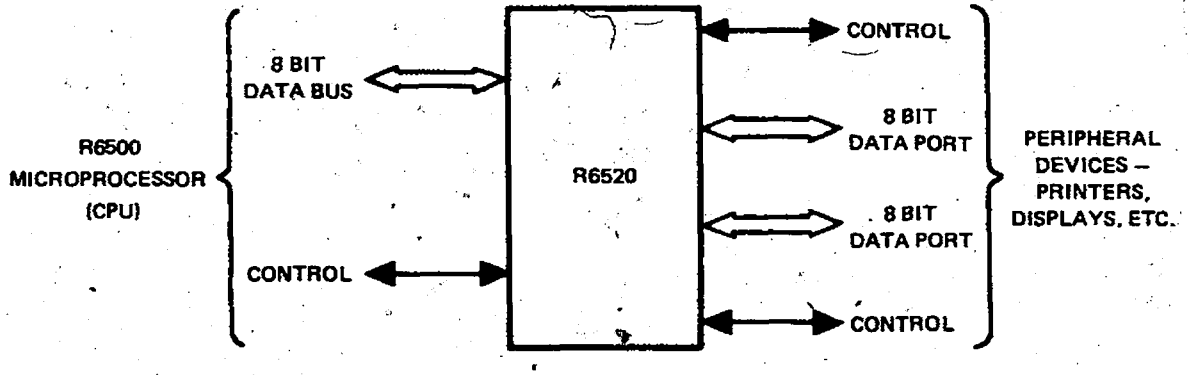
NOTE: Contact your local Rockwell Representative concerning availability.

FEATURES

- High performance replacement for Motorola/AMI/MOSTEK/Hitachi peripheral adapter.
- N channel, depletion load technology, single +5V supply.
- Completely Static and TTL compatible.
- CMOS compatible peripheral control lines.
- Fully automatic "hand-shake" allows positive control of data transfers between processor and peripheral devices.
- Commercial, industrial and military temperature range versions.



Pin Configuration



Basic R6520 Interface Diagram

SUMMARY OF R6520 OPERATION

See Rockwell Microcomputer Hardware Manual for detailed description of R6520 operation.

CA1/CB1 Control

CRA (CRB) Bit 1	CRA (CRB) Bit 0	Active Transition of Input Signal*	IRQA (IRQB) Interrupt Outputs
0	0	Negative	Disable – remain high
0	1	Negative	Enable – goes low when bit 7 in CRA (CRB) is set by active transition of signal on CA1 (CB1)
1	0	Positive	Disable – remain high
1	1	Positive	Enable – as explained above

*Note: Bit 7 of CRA (CRB) will be set to a logic 1 by an active transition of the CA1 (CB1) signal. This is independent of the state of Bit 0 in CRA (CRB).

CA2/CB2 Input Modes

CRA (CRB) Bit 5	CRA (CRB) Bit 4	CRA (CRB) Bit 3	Active Transition of Input Signal*	IRQA (IRQB) Interrupt Output
0	0	0	Negative	Disable – remains high
0	0	1	Negative	Enable – goes low when bit 6 in CRA (CRB) is set by active transition of signal on CA2 (CB2)
0	1	0	Positive	Disable – remains high
0	1	1	Positive	Enable – as explained above

Note: Bit 6 of CRA (CRB) will be set to a logic 1 by an active transition of the CA2 (CB2) signal. This is independent of the state of Bit 3 in CRA (CRB).

CA2 Output Modes

CRA Bit 5	CRA Bit 4	CRA Bit 3	Mode	Description
1	0	0	"Handshake" on Read	CA2 is set high on an active transition of the CA1 interrupt input signal and set low by a microprocessor "Read A Data" operation. This allows positive control of data transfers from the peripheral device to the microprocessor.
1	0	1	Pulse Output	CA2 goes low for one cycle after a "Read A Data" operation. This pulse can be used to signal the peripheral device that data was taken.
1	1	0	Manual Output	CA2 set low
1	1	1	Manual Output	CA2 set high

CB2 Output Modes

CRB Bit 5	CRB Bit 4	CRB Bit 3	Mode	Description
1	0	0	"Handshake" on Write	CB2 is set low on microprocessor "Write B Data" operation and is set high by an active transition of the CB1 interrupt input signal. This allows positive control of data transfers from the microprocessor to the peripheral device.
1	0	1	Pulse Output	CB2 goes low for one cycle after a microprocessor "Write B Data" operation. This can be used to signal the peripheral device that data is available.
1	1	0	Manual Output	CB2 set low
1	1	1	Manual Output	CB2 set high

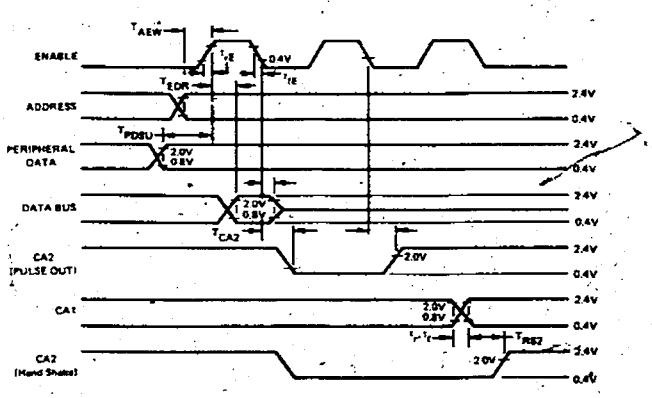
A.C. CHARACTERISTICS

Read Timing Characteristics (Loading 130 pF and one TTL load)

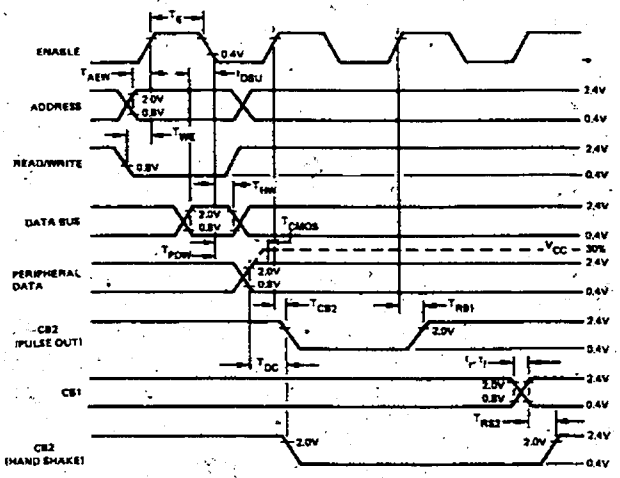
Characteristics	Symbol	1 MHz		2 MHz		Unit
		Min	Max	Min	Max	
Delay Time, Address valid to Enable positive transition	T_{AEW}	180	–	90	–	ns
Delay Time, Enable positive transition to Data valid on bus	T_{EDR}	–	395	–	190	ns
Peripheral Data Setup Time	T_{PDSU}	300	–	150	–	ns
Data Bus Hold Time	T_{HR}	10	–	10	–	ns
Delay Time, Enable negative transition to CA2 negative transition	T_{CA2}	–	1.0	–	0.5	µs
Delay Time, Enable negative transition to CA2 positive transition	T_{RS1}	–	1.0	–	0.5	µs
Rise and Fall Time for CA1 and CA2 input signals	t_r, t_f	–	1.0	–	0.5	µs
Delay Time from CA1 active transition to CA2 positive transition	T_{RS2}	–	2.0	–	1.0	µs
Rise and Fall Time for Enable input	t_{rE}, t_{fE}	–	25	–	25	ns

Write Timing Characteristics

Characteristics	Symbol	1 MHz		2 MHz		Unit
		Min	Max	Min	Max	
Enable Pulse Width	T_E	0.470	25	0.235	25	µs
Delay Time, Address valid to Enable positive transition	T_{AEW}	180	–	90	–	ns
Delay Time, Data valid to Enable negative transition	T_{DSU}	300	–	150	–	ns
Delay Time, Read/Write negative transition to Enable positive transition	T_{WE}	130	–	65	–	ns
Data Bus Hold Time	T_{HW}	10	–	10	–	ns
Delay Time, Enable negative transition to Peripheral Data valid	T_{PDW}	–	1.0	–	0.5	µs
Delay Time, Enable negative transition to Peripheral Data valid CMOS (V_{CC} - 30%) PA0-PA7, CA2	T_{CMOS}	–	2.0	–	1.0	µs
Delay Time, Enable positive transition to CB2 negative transition	T_{CB2}	–	1.0	–	0.5	µs
Delay Time, Peripheral Data valid to CB2 negative transition	T_{DC}	0	1.5	0	0.75	µs
Delay Time, Enable positive transition to CB2 positive transition	T_{RS1}	–	1.0	–	0.5	µs
Rise and Fall Time for CB1 and CB2 input signals	t_r, t_f	–	1.0	–	0.5	µs
Delay Time, CB1 active transition to CB2 positive transition	T_{RS2}	–	2.0	–	1.0	µs



Read Timing Characteristics



Write Timing Characteristics

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 to +7.0	Vdc
Input Voltage	V_{in}	-0.3 to +7.0	Vdc
Operating Temperature Range	T		°C
Commercial		0 to +70	
Industrial		-40 to +85	
Military		-55 to +125	
Storage Temperature Range	T_{STG}	-55 to + 150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages, however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this circuit.

Static D.C. Characteristics

(V_{CC} = 5.0V ± 5%, V_{SS} = 0, T_A = 25°C unless otherwise noted)

Characteristic	Symbol	Min	Typ	Max	Unit
Input High Voltage (Normal Operating Levels)	V_{IH}	+2.0	–	V_{CC}	Vdc
Input Low Voltage (Normal Operating Levels)	V_{IL}	-0.3	–	+0.8	Vdc
Input Threshold Voltage	V_{IT}	0.8	–	2.0	Vdc
Input Leakage Current	I_{in}				μAdc
V_{in} = 0 to 5.0 Vdc R/W, $\overline{Reset}$, RS0, RS1, CS0, CS1, $\overline{CS2}$, CA1, CB1, Φ2		–	±1.0	±2.5	
Three-State (Off State Input Current) (V_{in} = 0.4 to 2.4 Vdc, V_{CC} = max) D0-D7, PB0-PB7, CB2	I_{TSI}	–	±2.0	±10	μAdc
Input High Current (V_{IH} = 2.4 Vdc) PA0-PA7, CA2	I_{IH}	-100	-250	–	μAdc
Input Low Current (V_{IL} = 0.4 Vdc) PA0-PA7, CA2	I_{IL}	–	-1.0	-1.6	mAdc
Output High Voltage (V_{CC} = min, I_{Load} = -100 μAdc)	V_{OH}	2.4	–	–	Vdc
Output Low Voltage (V_{CC} = min, I_{Load} = 1.6 mAdc)	V_{OL}	–	–	+0.4	Vdc
Output High Current (Sourcing) (V_{OH} = 2.4 Vdc)	I_{OH}	-100	-1000	–	μAdc
(V_O = 1.5 Vdc, the current for driving other than TTL, e.g., Darlington Base) PB0-PB7, CB2		-1.0	-2.5	–	mAdc
Output Low Current (Sinking) (V_{OL} = 0.4 Vdc)	I_{OL}	1.6	–	–	mAdc
Output Leakage Current (Off State) $\overline{IRQA}$, $\overline{IRQB}$	I_{off}	–	1.0	10	μAdc
Power Dissipation	P_D	–	200	500	mW
Input Capacitance (V_{in} - 0, T_A = 25°C, f = 1.0 MHz)	C_{in}				pF
D0-D7, PA0-PA7, PB0-PB7, CA2, CB2		–	–	10	
R/W, $\overline{Reset}$, RS0, RS1, CS0, CS1, $\overline{CS2}$,		–	–	7.0	
CA1, CB1, Φ2		–	–	20	
Output Capacitance (V_{in} - 0, T_A = 25°C, f = 1.0 MHz)	C_{out}	–	–	10	pF

NOTE: Negative sign indicates outward current flow, positive indicates inward flow.

ROCKWELL INTERNATIONAL — MICROELECTRONIC DEVICES

REGIONAL SALES OFFICES

SOUTHWEST REGION, U.S.A.*
3310 Miraloma Avenue
P.O. Box 3669
Anaheim, California 92803
(714) 632-0950
TWX: 910-591-1698

NORTHWEST REGION, U. S. A.
Rockwell International
1601 Civic Center Drive, Suite 203
Santa Clara, CA 95050
408/984-8070
Telex: 171135 mission snta

CENTRAL REGION, U.S.A.
Contact Robert O. Whitesell & Associates
8891 East Washington Street
Indianapolis, Indiana 46219
(317) 359-9263
Attn: Milt Gamble, Mgr. (Acting)
TWX: 810-341-3320

EASTERN REGION, U.S.A.*
Carolier Office Building
850-870 U.S. Route 1
North Brunswick, New Jersey 08902
(201) 246-3630
TWX: 710 480 6261 RECTC NBR

MIDWEST REGION, U.S.A.
1011 E. Touhy Avenue, Suite 245
Des Plaines, Illinois 60018
(312) 297-8862
Telex: 726353 Rockwell

EUROPE
Rockwell International GmbH
Microelectronic Devices
Fraunhoferstrasse 11
D-8033 Munchen-Martinsried
Germany
(089) 859-9575
Telex: 0521/2650

FAR EAST
Rockwell International Overseas Corp.
Itohpia Hirakawa-cho Bldg.
7-8 Hirakawa-cho 2-chome
Chiyoda-ku, Japan
(03) 265-8806
Telex: J22196

*Also Applications Centers

Rockwell International

YOUR LOCAL REPRESENTATIVE

Learning Activity F2

Versatile Interface Adapter (VIA)

Objective: To be able to set the Peripheral Control Register (PCR), Interrupt Flag Register (IFR), Interrupt Enable Register (IER) and the Data Direction Register (DDR).

The VIA is an I/O device which acts as an interface between the microprocessor and peripherals such as printers, displays, keyboards etc. It is comprised of two almost identical sections, A and B, each capable of receiving or transmitting eights bits of data. In addition it contains a powerful interface timer, serial-to-parallel and parallel-to-serial shift register and input data latching on the peripheral ports.

See figure 2

The VIA will be set up using the same example as the PIA. For further information on the VIA see Rockwell's R6522 Microcomputer data sheet on the VIA. Available from Rockwell International, Microelectronic Devices, Anaheim, CA 92803 USA.

Port A Data Direction Register (DDRA) is located at address A003 and port B DDRB is located at A002.

To set port A or B for output load FF into the DDR and to reset port A or B as input load 00 into the appropriate DDR.

The following example will be used to illustrate some of the features of PCR, IFR, IER and the DDR's.

Example

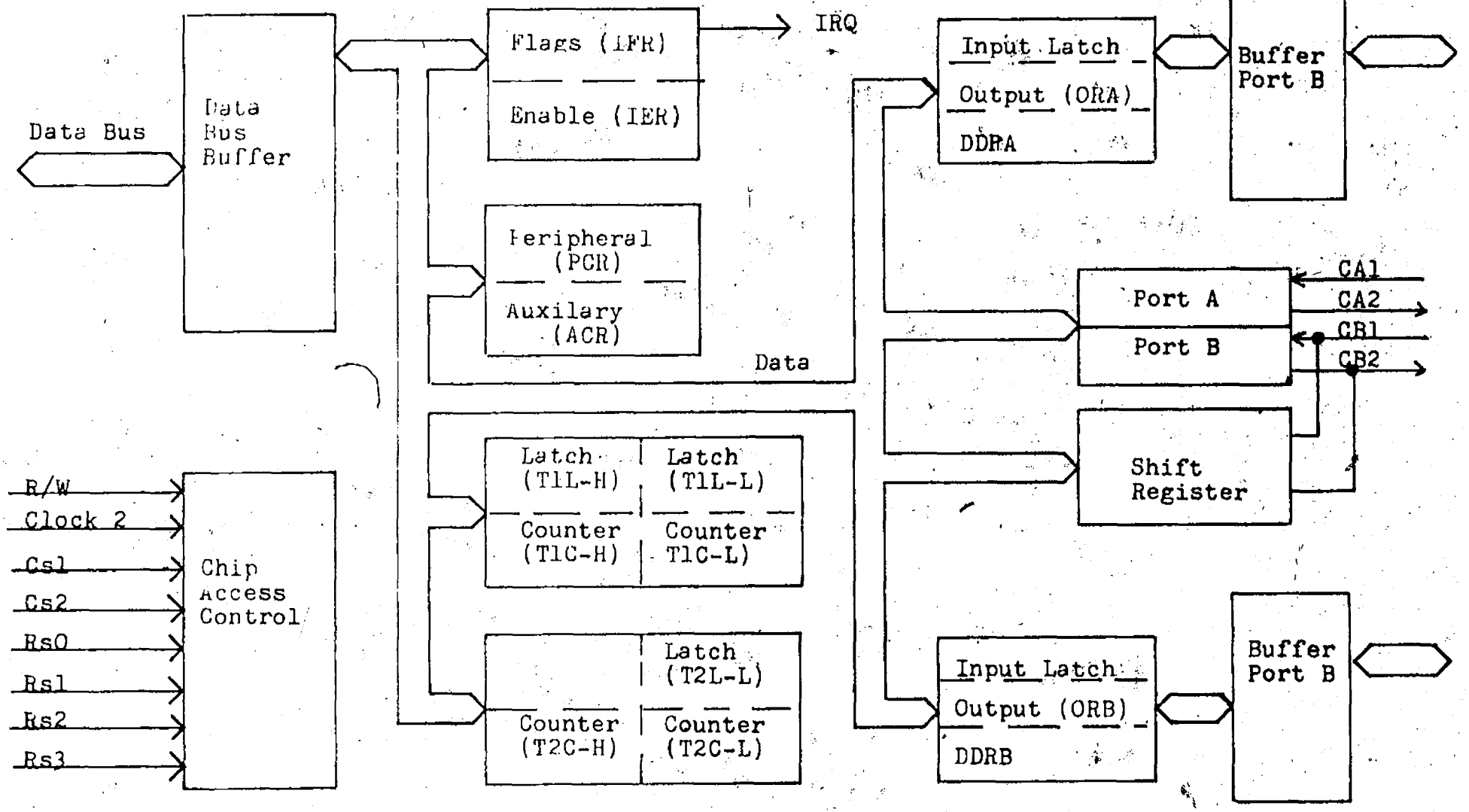
1 Port A will be used for the input

2 Port B will be used for the output

3 Set up PCR, IFR and IER for the following
When data appears on Port A and the peripheral signals the VIA via CAI (negative transition) that data is ready, the VIA will interrupt the CPU, read port A and generate a data taken signal.

4 After the CPU processes the data it will output it via port B. Port B will then generate a data available signal.

Figure 2


Data Bus
Data Bus Buffer
Flags (IFR)
Enable (IER)
IRQ
Peripheral (PCR)
Auxilary (ACR)
Data
Input Latch
Output (ORA)
DDRA
Buffer Port B
Port A
Port B
CA1
CA2
CB1
CB2
Shift Register
R/W
Clock 2
Cs1
Cs2
Rs0
Rs1
Rs2
Rs3
Chip Access Control
Latch (T1L-H)
Latch (T1L-L)
Counter (T1C-H)
Counter T1C-L)
Latch (T2L-L)
Counter (T2C-H)
Counter (T2C-L)
Input Latch
Output (ORB)
DDRB
Buffer Port B

Solution

1 Address of port A is A001
Address of DDRA is A003
For port A input

```
LDA  #00
STA  A003
```

2 Address of port B is A000
Address of DDRB is A002
For port B output

```
LDA  #FF
STA  A002
```

3 Reset bit 0 on PCR to logic 0
Set bit 1 on IER to a logic 1 so that a negative transition on CA1 will cause an interrupt (IRQ) to be generated.
Set bit 1 and 3 on PCR to a logic 1 so that CA2 will generate a data taken pulse after the CPU reads port A.

```
LDA  #0A      set bit 1 and 3 clear all other bits
STA  A00C     address of PCR
LDA  #FF      clear IFR
STA  A00D     address of IFR
LDA  #82      CA1 interrupt enable
STA  A00E     address of IER
```

4 Set bit 5 and 7 to a logic 1 and clear bit 6 on the PCR so that after data is written into port B a data available signal will be generated by CB2.

```
LDA  #BF
AND  A00C
ORA  #AD
STA  A00C      clear bit 6 and set bit 5 and 7 of PCR
```

4 could be combined with 3 above to form

```
LDA  #AA      set bit 1,3,5,and 7 clear all other
              bits
STA  A00C
```

Combine 1,2,3 and 4 to write a program that will set PCR, IFR, IER, DDRA and DDRB to accomodate the above example.

Rockwell

R6500 Microcomputer System
DATA SHEET

VERSATILE INTERFACE ADAPTER (VIA)

DESCRIPTION

The R6522 Versatile Interface Adapter (VIA) features two 8-bit bidirectional I/O data ports, four I/O control lines, two independent 16-bit timers and an 8-bit serial-to-parallel/parallel-to-serial Shift Register. Seven detectable I/O conditions are indicated in an Interrupt Flag Register. These conditions may be programmed to issue an interrupt request to the processor to allow polled and/or immediate processor response to selective I/O line, timer, and Shift Register operation.

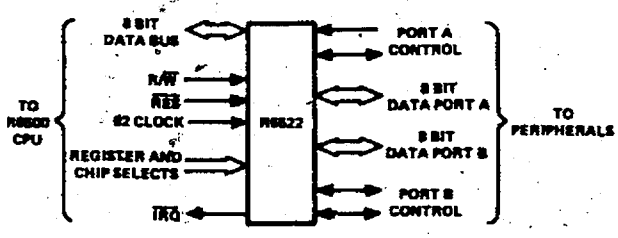
Control and monitoring of peripheral devices is handled primarily through the two 8-bit I/O ports. Each I/O line can be programmed to latch input data. The four I/O control lines provide an expanded handshaking capability which allows control of data transfer between the R6522 and interfacing peripheral devices or between separate VIAs in multiple processor systems. Programmable negative or positive edge detection capability on the I/O control lines allows the R6522 to be easily included in a variety of existing and new control applications. Each control line may be programmed to interrupt the processor upon detection of a rising or falling edge.

One I/O line can be selectively controlled by a timer to generate a programmable-frequency square wave or a variable-width rectangular wave. Another I/O line can be configured to count externally generated pulses using the other timer.

Positive programmable control of the R6522 VIA is achieved through its internal register organization: the Interrupt Enable Register, the Interrupt Flag Register and two function/peripheral control registers, the Auxiliary Control Register, and the Peripheral Control Register.

Ordering Information

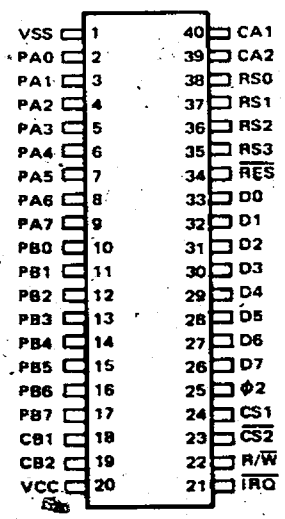
Order Number	Package Type	Frequency	Temperature Range
R6522P	Plastic	1 MHz	0°C to +70°C
R6522AP	Plastic	2 MHz	0°C to +70°C
R6522C	Ceramic	1 MHz	0°C to +70°C
R6522AC	Ceramic	2 MHz	0°C to +70°C
R6522PE	Plastic	1 MHz	-40°C to +85°C
R6522APE	Plastic	2 MHz	-40°C to +85°C
R6522CE	Ceramic	1 MHz	-40°C to +85°C
R6522ACE	Ceramic	2 MHz	-40°C to +85°C
R6522CMT	Ceramic	1 MHz	-55°C to +125°C



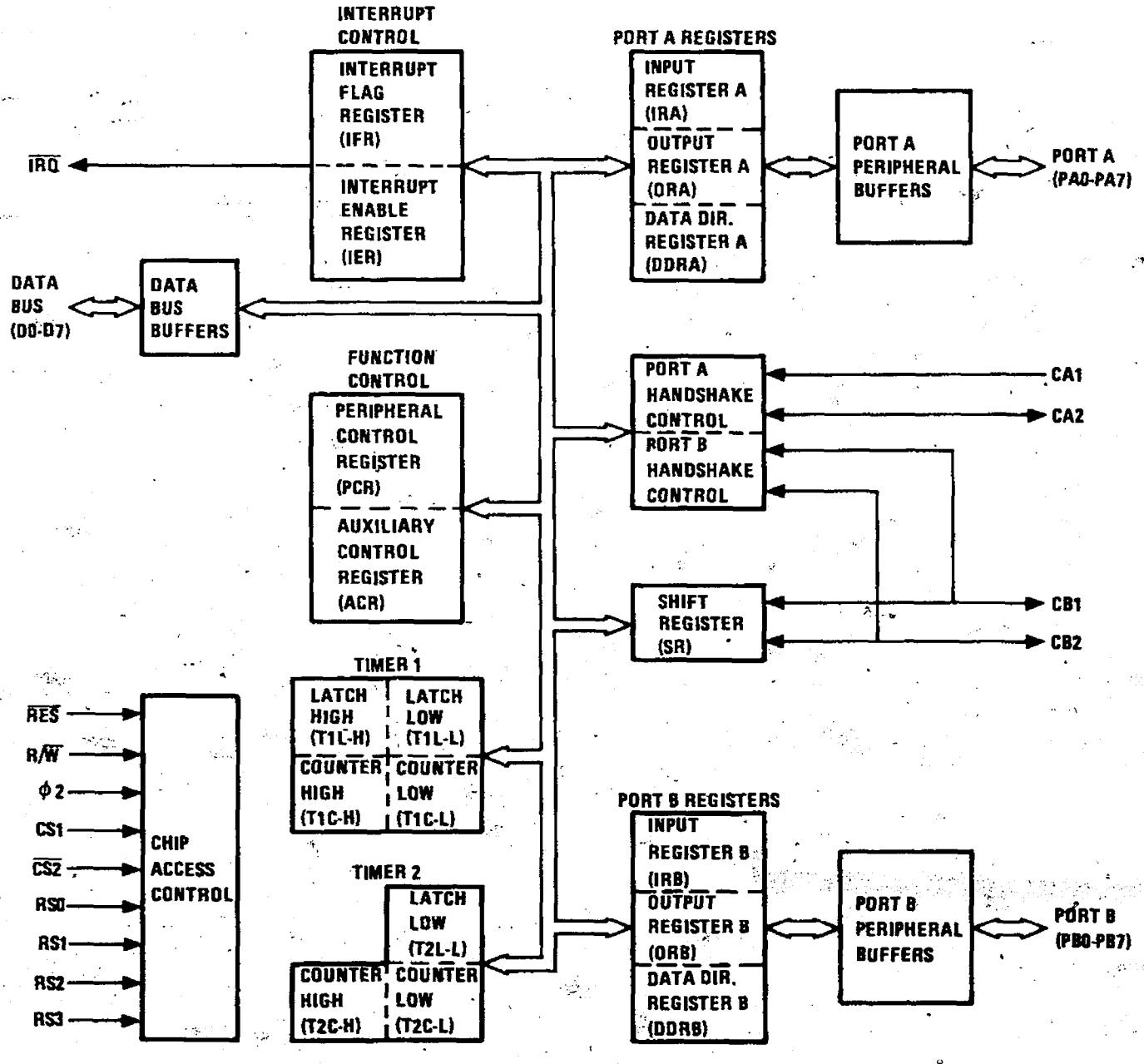
Basic R6522 Interface Diagram

FEATURES

- Organized for simplified software control of many functions
- Compatible with the R650X and R651X family of microprocessors (CPUs)
- Bi-directional, 8-bit data bus for communication with microprocessor
- Two Bi-directional, 8-bit input/output ports for interface with peripheral devices
- CMOS and TTL compatible input/output peripheral ports
- Data Direction Registers allow each peripheral pin to act as either an input or an output
- Interrupt Flag Register allows the microprocessor to readily determine the source of an interrupt and provides convenient control of the interrupts within the chip
- Handshake control logic for input/output peripheral data transfer operations
- Data latching on peripheral input/output ports
- Two fully-programmable 16-bit interval timers/counters
- Eight-bit Shift Register for serial interface
- Forty-pin plastic or ceramic DIP package
- Timer 1 with four modes:
  - One shot interval timer
  - Free running mode
  - Both above modes with toggle output to PB7 enabled or disabled
- Timer 2 with three modes:
  - One shot interval timer
  - Counts external pulses on PB6
  - Clock serial shift register



Pin Configuration



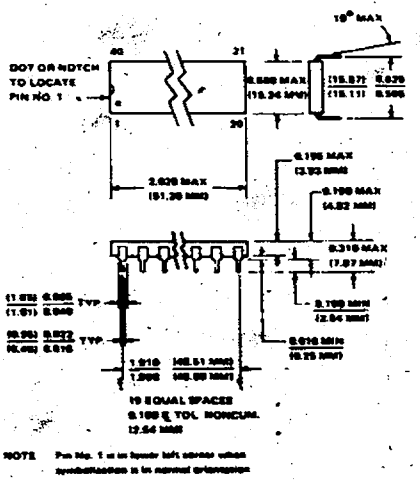
R6522 Block Diagram

REGISTER ADDRESSING

The four Register Select Lines are normally connected to the processor address bus lines to allow the processor to select the internal R6522 register which is to be accessed. The sixteen possible combinations access the registers as follows:

Register Select				Address	Addressed Register	R/$\overline{W}$ = L	R/$\overline{W}$ = H	Notes
RS3	RS2	RS1	RS0					
L	L	L	L	0	IRB/ORB	Write ORB Clear CB2 and CB1 Interrupt Flags (IFR3 and IFR4)	Read IRB Clear CB2 and CB1 Interrupt Flags (IFR3 and IFR4)	1 = High, 0 = Low
L	L	L	H	1	IRA/ORA	Write ORA Clear CA2 and CA1 Interrupt Flags (IFR0 and IFR1)	Read IRA Clear CA2 and CA1 Interrupt Flags (IFR0 and IFR1)	1 = High, 0 = Low Controls CA2 Handshake
L	L	H	L	2	DDRB	Write DDRB	Read DDRB	0 = Input, 1 = Output
L	L	H	H	3	DDRA	Write DDRA	Read DDRA	0 = Input, 1 = Output
L	H	L	L	4	T1	Write T1L-L	Read T1C-L Clear T1 Interrupt Flag (IFR6)	
L	H	L	H	5	T1	Write T1L-H & T1C-H Transfer T1L-L to T1C-L Clear T1 Interrupt Flag (IFR6) Initiate T1 Counting	Read T1C-H	
L	H	H	L	6	T1	Write T1L-L	Read T1L-L	
L	H	H	H	7	T1	Write T1L-H Clear T1 Interrupt Flag (IFR6)	Read T1L-H	
H	L	L	L	8	T2	Write T2L-L	Read T2C-L Clear T2 Interrupt Flag (IFR5)	
H	L	L	H	9	T2	Write T2C-H Transfer T2L-L to T2C-L Clear T2 Interrupt Flag (IFR5) Initiate T2 Counting	Read T2C-H	
H	L	H	L	A	SR	Write SR Clear SR Interrupt Flag (IFR2)	Read SR Clear SR Interrupt Flag (IFR2)	
H	L	H	H	B	ACR	Write ACR	Read ACR	
H	H	L	L	C	PCR	Write PCR	Read PCR	
H	H	L	H	D	IFR	Write IFR	Read IFR	1 = Detected, 0 = Not Detected
H	H	H	L	E	IER	Write IER	Read IER	1 = Enable, 0 = Disable
H	H	H	H	F	IRA/ORA	Write ORA Clear CA2 and CA1 Interrupt Flags (IFR0 and IFR1)	Read IRA Clear CA2 and CA1 Interrupt Flags (IFR0 and IFR1)	1 = High, 0 = Low No Effect on CA2 Handshake

Note: L = 0.4 VDC, H = 2.2 VDC



40-Pin Plastic Package

FUNCTION CONTROL

Control of the various functions and operating modes within the R6522 is accomplished primarily through two registers, the Peripheral Control Register (PCR), and the Auxiliary Control Register (ACR)

PERIPHERAL CONTROL REGISTER (PCR)

The PCR is used primarily to select the operating mode for the four peripheral control pins (CA1, CA2, CB1 and CB2). The Peripheral Control Register is organized as follows:

Bit No.	7	6	5	4	3	2	1	0
Bit Designation	PCR7	PCR6	PCR5	PCR4	PCR3	PCR2	PCR1	PCR0
Function	CB2 Control			CB1 Control	CA2 Control			CA1 Control

CA1 Control

PCR0 = 0 The CA1 Interrupt Flag (IFR1) will be set by a negative transition (high to low) on the CA1 pin.
= 1 The CA1 Interrupt Flag (IFR1) will be set by a positive transition (low to high) on the CA1 pin.

CA2 Control

PCR3	PCR2	PCR1	Mode
0	0	0	CA2 negative edge detect (IFR0/ORA clear) mode — Set CA2 interrupt flag (IFR0) on a negative transition of the CA2 input signal. Clear IFR0 on a read or write of the ORA or by writing logic 1 into IFR0.
0	0	1	CA2 negative edge detect (IFR0 clear) mode — Set IFR0 on a negative transition of the CA2 input signal. Clear IFR0 by writing logic 1 into IFR0.
0	1	0	CA2 positive edge detect (IFR0/ORA clear) mode — Set CA2 interrupt flag (IFR0) on a positive transition of the CA2 input signal. Clear IFR0 on a read or write of the ORA or by writing logic 1 into IFR0.
0	1	1	CA2 positive edge detect (IFR0 clear) mode — Set IFR0 on a positive transition of the CA2 input signal. Clear IFR0 by writing logic 1 into IFR0.
1	0	0	CA2 handshake output mode — Set CA2 output low on a read or write of the Peripheral A Output Register. Reset CA2 high with an active transition on CA1.
1	0	1	CA2 pulse output mode — CA2 goes low for one cycle following a read or write of the Peripheral A Output Register.
1	1	0	CA2 low output mode — The CA2 output is held low in this mode.
1	1	1	CA2 high output mode — The CA2 output is held high in this mode.

CB1 Control

PCR4 = 0 The CB1 Interrupt Flag (IFR4) will be set by a negative transition (high to low) on the CB1 pin.
= 1 The CB1 Interrupt Flag (IFR4) will be set by a positive transition (low to high) on the CB1 pin.

CB2 Control

PCR7	PCR6	PCR5	Mode
0	0	0	CB2 negative edge detect (IFR3/ORB clear) mode — Set CB2 interrupt flag (IFR3) on a negative transition of the CB2 input signal. Clear IFR3 on a read or write of the ORB or by writing logic 1 into IFR3.
0	0	1	CB2 negative edge detect (IFR3 clear) mode — Set IFR3 on a negative transition of the CB2 input signal. Clear IFR3 by writing logic 1 into IFR3.
0	1	0	CB2 positive edge detect (IFR3/ORB clear) mode — Set CB2 interrupt flag (IFR3) on a positive transition of the CB2 input signal. Clear IFR3 on a read or write of the ORB or by writing logic 1 into IFR3.
0	1	1	CB2 positive edge detect (IFR3 clear) mode — Set IFR3 on a positive transition of the CB2 input signal. Clear IFR3 by writing logic 1 into IFR3.
1	0	0	CB2 handshake output mode — Set CB2 output low on a write of the Peripheral B Output Register. Reset CB2 high with an active transition on CB1.
1	0	1	CB2 pulse output mode — CB2 goes low for one cycle following a read or write of the Peripheral B Output Register.
1	1	0	CB2 low output mode — The CB2 output is held low in this mode.
1	1	1	CB2 high output mode — The CB2 output is held high in this mode.

AUXILIARY CONTROL REGISTER (ACR)

The ACR selects the operating mode for the two interval timers (T1 and T2) and the Serial Register (SR). The Auxiliary Control Register is organized as follows:

Bit No.	7	6	5	4	3	2	1	0
Bit Designation	ACR7	ACR6	ACR5	ACR4	ACR3	ACR2	ACR1	ACR0
Function	Timer 1 Control		Timer 2 Control	Shift Register Control			Port B Latch Enable	Port A Latch Enable

Port A Latch Enable

ACR0 = 1 Port A latch is enabled to latch input data when CA1 Interrupt Flag (IFR1) is set.
= 0 Port A latch is disabled, reflects current data on PA pins.

Port B Latch Enable

ACR1 = 1 Port B latch is enabled to latch the voltage on the pins for the input lines or the ORB contents for the output lines when CB1 Interrupt Flag (IFR4) is set.
= 0 Port B latch is disabled, reflects current data on PB pins.

Shift Register Control

ACR4	ACR3	ACR2	Mode
0	0	0	Shift Register Disabled.
0	0	1	Shift in under control of Timer 2.
0	1	0	Shift in under control of Ø2.
0	1	1	Shift in under control of external clock.
1	0	0	Free-running output at rate determined by Timer 2.
1	0	1	Shift out under control of Timer 2.
1	1	0	Shift out under control of Ø2.
1	1	1	Shift out under control of external clock.

Timer 2 Control

ACR5 = 0 T2 acts as an interval timer in the one-shot mode.
= 1 T2 counts a predetermined number of pulses on PB6.

Timer 1 Control

ACR7	ACR6	Mode
0	0	T1 one-shot mode — Generate a single time-out interrupt each time T1 is loaded. Output to PB7 disabled.
0	1	T1 free-running mode — Generate continuous interrupts. Output to PB7 disabled.
1	0	T1 one-shot mode — Generate a single time-out interrupt and an output pulse on PB7 each time T1 is loaded.
1	1	T1 free-running mode — Generate continuous interrupts and toggle the output on PB7.

INTERRUPT CONTROL

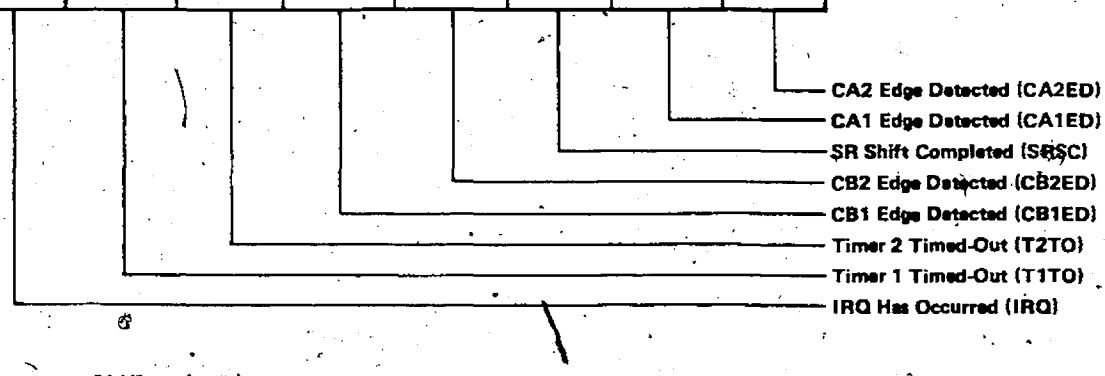
The interrupt control is performed with two registers, the Interrupt Flag Register (IFR), and the Interrupt Enable Register (IER).

INTERRUPT FLAG REGISTER (IFR)

The IFR indicates detection of up to seven I/O conditions associated with the two interval timers (T1 and T2), the control lines (CA1, CA2, CB1, and CB2) and the Shift Register (SR). In addition, whenever any bit from IFR0 through IFR6 is set to logic 1, if the corresponding bit in the IER is set to logic 1, the $\overline{IRQ}$ interrupt output line is driven low and IFR7 set to logic 1 to indicate that an IRQ interrupt has been generated.

The Interrupt Flag Register is organized as follows:

Bit No.	7	6	5	4	3	2	1	0
Bit Designation	IFR7	IFR6	IFR5	IFR4	IFR3	IFR2	IFR1	IFR0
Function	IRQ	T1TO	T2TO	CB1ED	CB2ED	SRSC	CA1ED	CA2ED

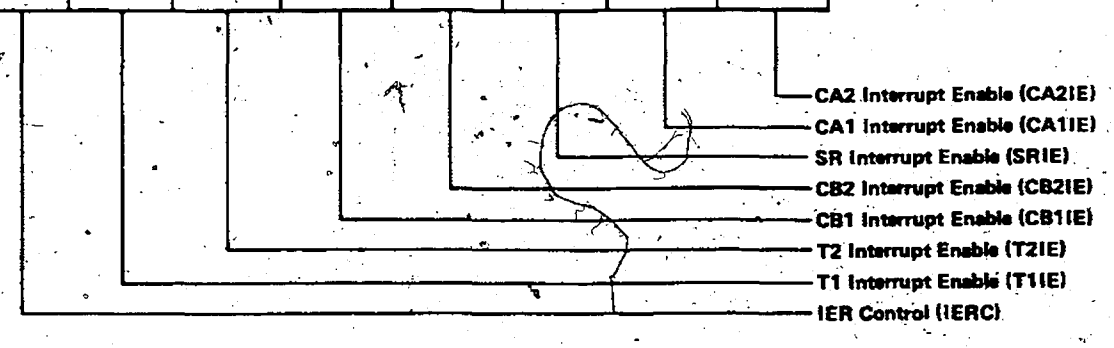


INTERRUPT ENABLE REGISTER (IER)

For bits 0 to 6 in the IFR there is a corresponding bit in the IER. If one of these bits is set to logic 1 in the IER, an IRQ interrupt will be generated if the corresponding bit in the IFR is set to logic 1.

The Interrupt Enable Register is organized as follows:

Bit No.	7	6	5	4	3	2	1	0
Bit Designation	IER7	IER6	IER5	IER4	IER3	IER2	IER1	IER0
Function	IERC	T1IE	T2IE	CB1IE	CB2IE	SRIE	CA1IE	CA2IE



Interrupt Enable Bits (IER0-6)

IERn = 0 Disable Interrupt
= 1 Enable Interrupt

IER Control (IER7)

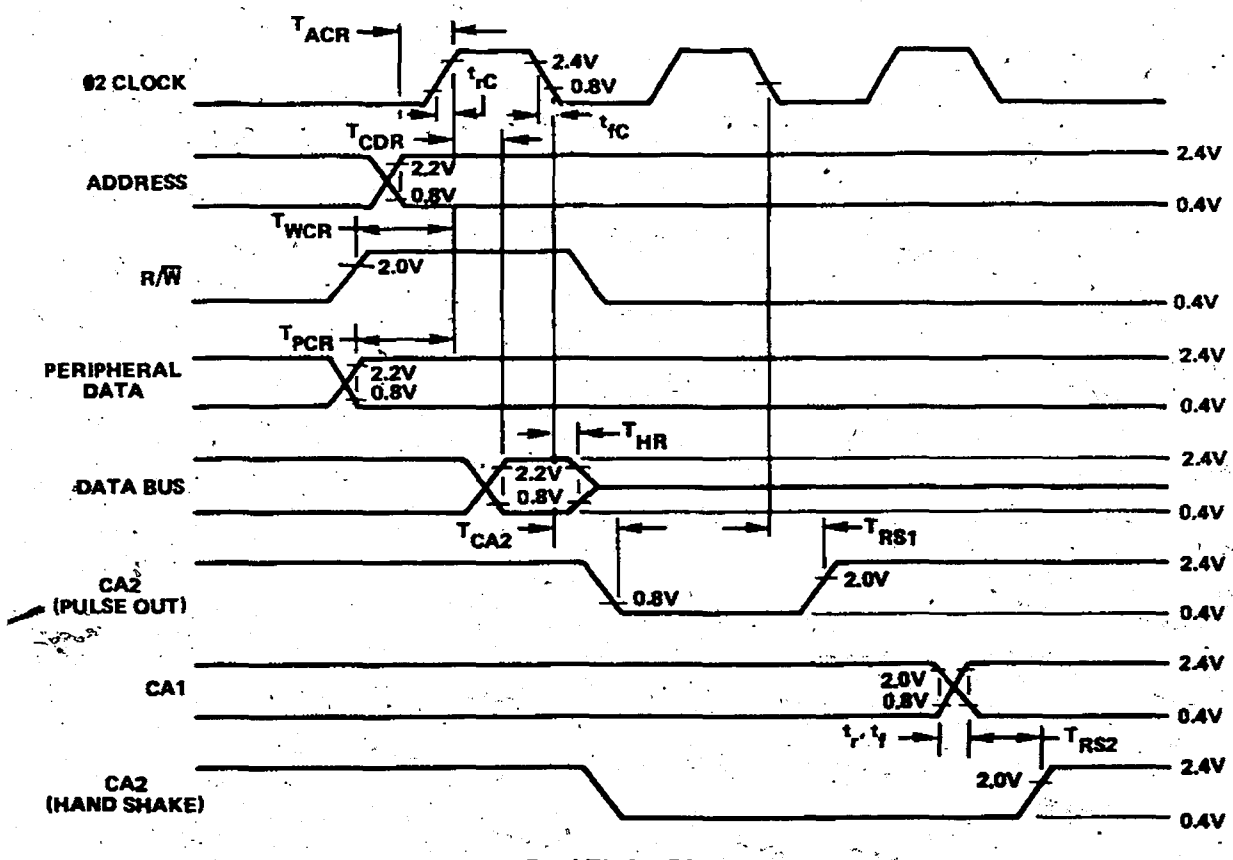
IER7 = 0 For each data bus bit set to logic 1, the corresponding IER bit is cleared
= 1 For each data bus bit set to logic 1, the corresponding IER bit is set

Note: IER7 is active only when $R/\overline{W}$ = L; when $R/\overline{W}$ = H, IER7 will read logic 1.

READ TIMING CHARACTERISTICS

(Loading 130 pF and one TTL load)

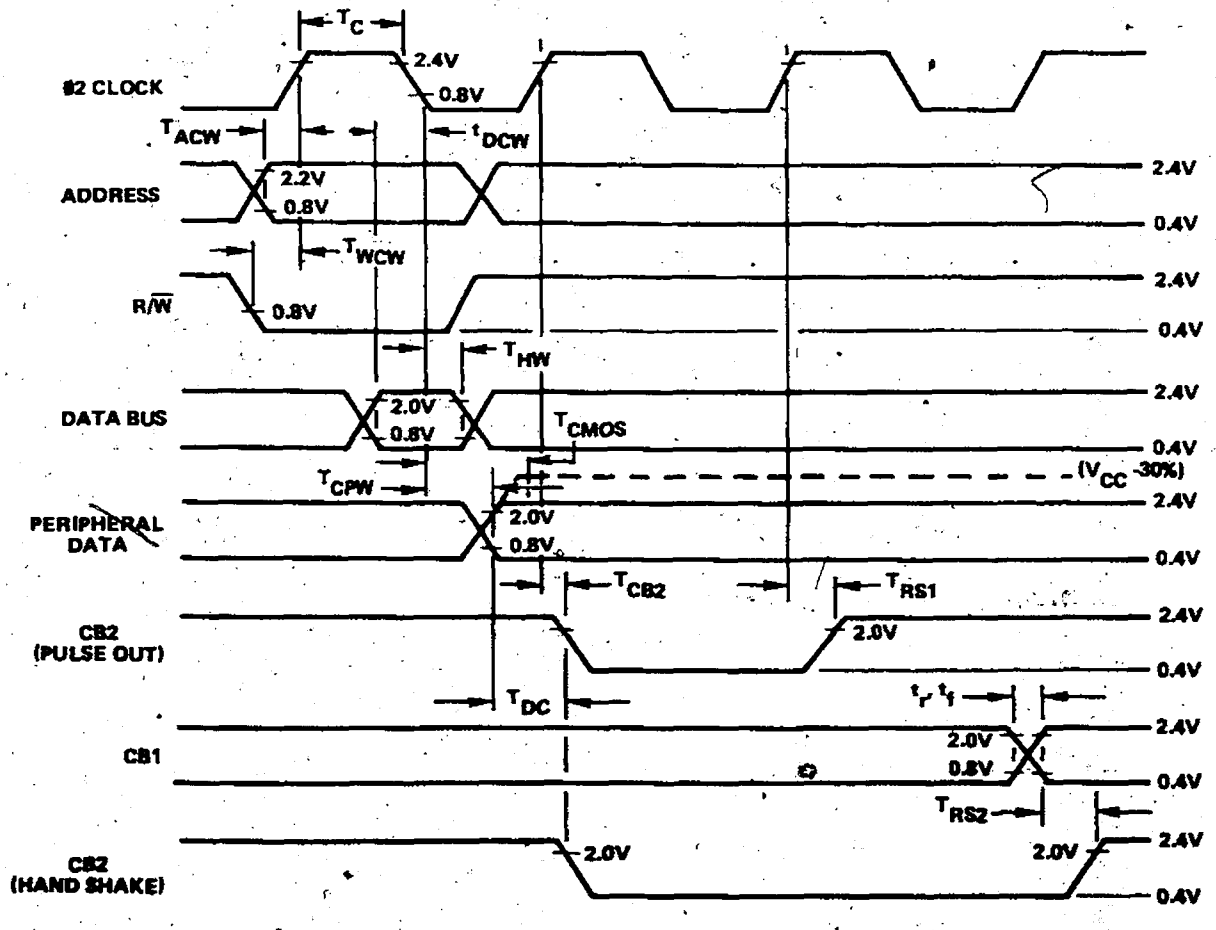
Characteristics	Symbol	1 MHz		2 MHz		Unit
		Min	Max	Min	Max	
Delay Time, Address valid to Clock positive transition	T_{ACR}	180	–	90	–	ns
Delay Time, Clock positive transition to Data valid on bus	T_{CDR}	–	395	–	190	ns
R/$\overline{W}$ valid before positive edge of clock	T_{WCR}	180	–	90	–	ns
Peripheral data valid before positive transition of clock	T_{PCR}	300	–	150	–	ns
Data Bus Hold Time	T_{HR}	10	–	10	–	ns
Delay Time, Clock negative transition to CA2 negative transition	T_{CA2}	–	1.0	–	0.5	µs
Delay Time, Clock negative transition to CA2 positive transition	T_{RS1}	–	1.0	–	0.5	µs
Rise and Fall Time for CA1 and CA2 input signals	t_r, t_f	–	1.0	–	0.5	µs
Delay Time from CA1 active transition to CA2 positive transition	T_{RS2}	–	2.0	–	1.0	µs
Rise and Fall Time for Clock input	t_{rC}, t_{fC}	–	25	–	25	ns



Read Timing Diagram

WRITE TIMING CHARACTERISTICS

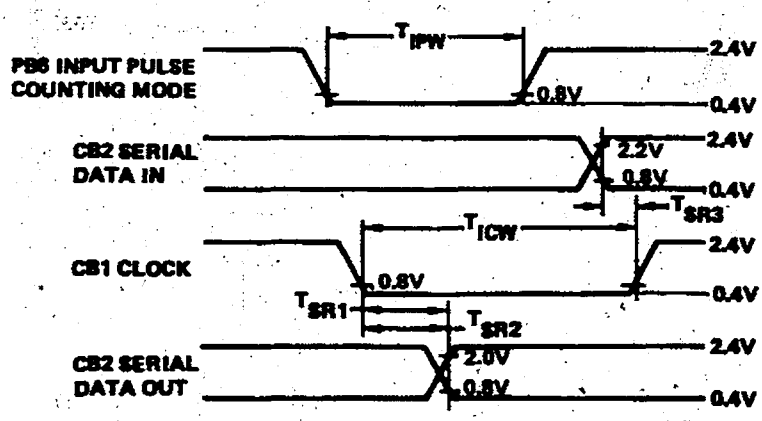
Characteristics	Symbol	1 MHz Min	1 MHz Max	2 MHz Min	2 MHz Max	Unit
Clock Pulse Width	T_C	0.470	10	0.235	10	µs
Delay Time, Address valid to Clock positive transition	T_{ACW}	180	–	90	–	ns
Delay Time, Data valid to Clock negative transition	T_{DCW}	300	–	150	–	ns
Delay Time, Read/Write negative transition to Clock positive transition	T_{WCW}	130	–	65	–	ns
Data Bus Hold Time	T_{HW}	10	–	10	–	ns
Delay Time, Clock negative transition to Peripheral Data valid	T_{CPW}	–	1.0	–	0.5	µs
Delay Time, Clock negative transition to Peripheral Data valid CMOS (V_{CC} - 30%), PA0-PA7, PB0-PB7, CA2	T_{CMOS}	–	2.0	–	1.0	µs
Delay Time, Clock positive transition to CB2 negative transition	T_{CB2}	–	1.0	–	0.5	µs
Delay Time, Peripheral Data valid to CB2 negative transition	T_{DC}	0	1.5	0	0.75	µs
Delay Time, Clock positive transition to CB2 positive transition	T_{RS1}	–	1.0	–	0.5	µs
Rise and Fall Time for CB1 and CB2 input signals	t_r, t_f	–	1.0	–	0.5	µs
Delay Time, CB1 active transition to CB2 positive transition	T_{RS2}	–	2.0	–	1.0	µs



Write Timing Diagram

I/O TIMING CHARACTERISTICS

Characteristic	Symbol	1 MHz		2 MHz		Unit
		Min	Max	Min	Max	
Rise and fall time for CA1, CB1, CA2 and CB2 input signals	T_{RF}	–	1.0	–	0.5	µs
Delay time, clock negative transition to CA2 negative transition (read handshake or pulse mode)	T_{CA2}	–	1.0	–	0.5	µs
Delay time, clock negative transition to CA2 positive transition (pulse mode)	T_{RS1}	–	1.0	–	0.5	µs
Delay time, CA1 active transition to CA2 positive transition (handshake mode)	T_{RS2}	–	2.0	–	1.0	µs
Delay time, clock positive transition to CA2 or CB2 negative transition (write handshake)	T_{WHS}	–	1.0	–	0.5	µs
Delay time, peripheral data valid to CB2 negative transition	T_{DC}	0	1.5	0	0.75	µs
Delay time, clock positive transition to CA2 or CB2 positive transition (pulse mode)	T_{RS3}	–	1.0	–	0.5	µs
Delay time, CB1 active transition to CA2 or CB2 positive transition (handshake mode)	T_{RS4}	–	2.0	–	1.0	µs
Delay time, peripheral data valid to CA1 or CB1 active transition (input latching)	T_{IL}	300	–	150	–	ns
Delay time CB1 negative transition to CB2 data valid (internal SR clock, shift out)	T_{SR1}	–	300	–	150	ns
Delay time, negative transition of CB1 input clock to CB2 data valid (external clock, shift out)	T_{SR2}	–	300	–	150	ns
Delay time, CB2 data valid to positive transition of CB1 clock (shift in, internal or external clock)	T_{SR3}	–	300	–	150	ns
Pulse Width – PB6 Input Pulse	T_{IPW}	2	–	1	–	µs
Pulse Width – CB1 Input Clock	T_{ICW}	2	–	1	–	µs
Pulse Spacing – PB6 Input Pulse	T_{IPS}	2	–	1	–	µs
Pulse Spacing – CB1 Input Pulse	T_{ICS}	2	–	1	–	µs



I/O Timing Diagram

SPECIFICATIONS

Maximum Ratings

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	-0.3 to +7.0	Vdc
Input Voltage	V_{IN}	-0.3 to +7.0	Vdc
Operating Temperature Range	T		°C
Commercial		0 to +70	
Industrial		-40 to +85	
Military		-55 to +125	
Storage Temperature Range	T_{STG}	-55 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages. However, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages.

Electrical Characteristics

(VCC = 5.0V ±5%, VSS = 0)

Characteristic	Symbol	Min	Max	Unit
Input high voltage (normal operation) except Ø2	V_{IH}	+2.2	VCC	Vdc
Input high voltage (normal operation) Ø2	V_{IHC}	+2.4	VCC	Vdc
Input low voltage (normal operation)	V_{IL}	-0.3	+0.8	Vdc
Input leakage current — V_{in} = 0 to 5 Vdc R/W, $\overline{RES}$, RS0, RS1, RS2, RS3, CS1, $\overline{CS2}$, CA1, Ø2	I_{IN}	—	±2.5	μAdc
Off-state input current — V_{in} = 0.4 to 2.4V VCC = Max, D0 to D7	I_{TSI}	—	±10	μAdc
Input high current — V_{IH} = 2.2V PA0-PA7, CA2, PB0-PB7, CB1, CB2	I_{IH}	-100	—	μAdc
Input low current — V_{IL} = 0.8 Vdc PA0-PA7, CA2, PB0-PB7, CB1, CB2	I_{IL}	—	-1.6	mAdc
Output high voltage VCC = min, I_{load} = -100 μAdc PA0-PA7, CA2, PB0-PB7, CB1, CB2	V_{OH}	2.4	—	Vdc
Output low voltage VCC = min, I_{load} = 1.6 mAdc	V_{OL}	—	+0.4	Vdc
Output high current (sourcing) V_{OH} = 2.4V V_{OH} = 1.5V, PB0-PB7, CB1, CB2	I_{OH}	 -100 -1.0	 — —	 μAdc mAdc
Output low current (sinking) V_{OL} = 0.4 Vdc	I_{OL}	1.6	—	mAdc
Output leakage current (off state) $\overline{IRQ}$	I_{off}	—	10	μAdc
Input Capacitance — T_A = 25°C, f = 1 MHz R/W, $\overline{RES}$, RS0, RS1, RS2, RS3, CS1, $\overline{CS2}$, D0-D7, PA0-PA7, CA2, PB0-PB7, CB1, CB2 Ø2 input	C_{in}	 — — —	 7.0 10 20	pF
Output capacitance — T_A = 25°C, f = 1 MHz	C_{out}	—	10	pF
Power dissipation	P_d	—	750	mW

V Troubleshooting Digital Circuits and Microcomputers

A Tools used to troubleshoot digital/computer systems.

B Troubleshooting microprocessor systems.

The purpose of this chapter is to introduce the student to different tools used in troubleshooting digital/computer circuits and some general techniques in troubleshooting.

Each manufacturer has developed troubleshooting methods for his products; therefore, rather than concentrating on specific techniques the following problem solving method is presented. It has general application and can be modified to apply to specific products.

Steps of thinking in problem solving
(Basic Principles of Curriculum and Instruction by Tyler)

1 Sensing a difficulty or question that cannot be answered at present.
2 Identifying the problem more clearly by analysis.
3 Collecting relevant facts.
4 Formulating possible hypothesis, that is developing possible solutions to the problem.
5 Testing the hypothesis by appropriate means.
6 Drawing conclusions - that is solving the problem.

A Tools used to Troubleshoot Digital/Computer Systems

1 Logic Probes

2 Logic Pulser

3 Current Tracer

4 Logic Comparator

5 Oscilloscopes

6 Logic State Analyzer

7 Signature Analysis

The circuit in figure 1 will be used to illustrate the application of these tools.

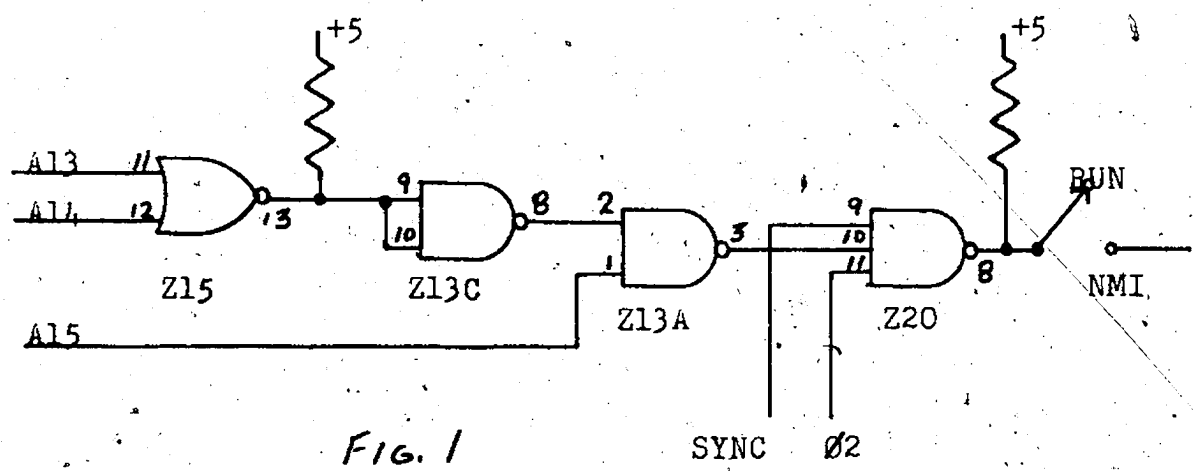


FIG. 1

Functional operation of the circuit in figure 1.

If address lines A13 or A14 are Lo and address line A15 is Lo the output of Z-13A will be Hi, and if Ø2 and the sync line both so Hi as well then the output of Z-20 goes Lo.

With the Single Step switch on Run this operation has no effect on the computer, however if the switch is on single step the NMI line is pulled low and the interrupt is serviced.

Address lines A13, A14 and A15 are used by the ROM therefore Single Step will not work while a ROM program is being run.

To make sure the AIM 65 is not addressing ROM you can load in this short program and execute it before each application of the above tools if applicable.

```
0000    JMP
0001     01
0002     00
```

Learning Activity A1
The Logic Probe
Typical Logic Probe Hewlett Packard (HP) 545A

Objective: To apply the Logic Probe to digital circuits.

The logic probe is a digital state indicator (Hi or Lo) which provides, via a lamp, an indication of a high level, low level or bad level signal.

Brightly lit lamp indicates a high.
Lamp off indicates a low.
Dimly lit lamp indicates a bad signal or high impedance state.

The logic probe also has pulse stretching capability so that it can detect pulses as narrow as 10 nanoseconds, and blink on and off for .1 seconds.

In addition the logic probe will detect pulse trains (clocks) up to 50 MHZ. When detecting these high frequency pulse trains the lamp will blink off and on at a 10 HZ rate.

Application of Logic Probes

Equipment Required: Aim 65 Computer

HP 545A Logic Probe

Connect the logic probe to pin 10, Z-16 same as pin 11, Z-20 (see fig.1) and observe the Ø2 clock. Probe action.....

(while performing the following test switch off and on your short program)

Connect the logic probe to pin 7, Z-9 same connection as pin 9, Z-20. Probe action.....

Connect probe to pin 10, Z-13C. Probe action.....

You have observed the three different actions of the probe:
1 pulse train Ø2
2 short pulse at pin 7, Z-9
3 Logic Hi or Lo at pin 10, Z-13C

Learning Activity A2

The Logic Pulser

Typical HP 546A

Objective: To apply the Logic Pulser to digital logic circuits.

The logic pulser injects the circuit with a single 500 nanosecond wide pulse of proper amplitude and polarity each time the button is pushed.

In addition pulse streams of 1, 10 and 100 Hz and pulse bursts of 1, 10 and 100 Hz can be selected.

Node: Each point in the circuit is called a node. All points that are wired together are part of the same node.

The pulser is capable of sourcing or sinking .5A for 500 nanoseconds to insure that the node is pulsed.

Stimulus-response testing is an effective technique to locate troubles in digital circuits. The logic pulser serves as the stimuls and the logic probe is used to monitor the result.

Application of the Logic Pulser

Equipment required: Aim 65 Computer
Logic Pulser
Logic Probe

Connect the logic pulser to pin 9, Z-13A

Connect the logic probe to pin 8, Z-13A

Press the button on the pulser

Record the output on the logic probe .....

You should have observed that the output changed when the button in the pulser was pressed.

To check multiple input gates they will have to be tied together and pulsed with the logic pulser.

Learning Activity A3

The Current Tracer

Typical HP 547A

Objective: To apply the current tracer to digital circuits.

The current tracer detects current activity on logic nodes by means of an inductive pick-up at it's tip. In order to use the current tracer it must be positioned perpendicular to the line being traced and the small hole in the tip aligned with the direction of the current being traced. The sensitivity control is adjusted for half intensity on the current tracer lamp.

Application of the Current Tracer

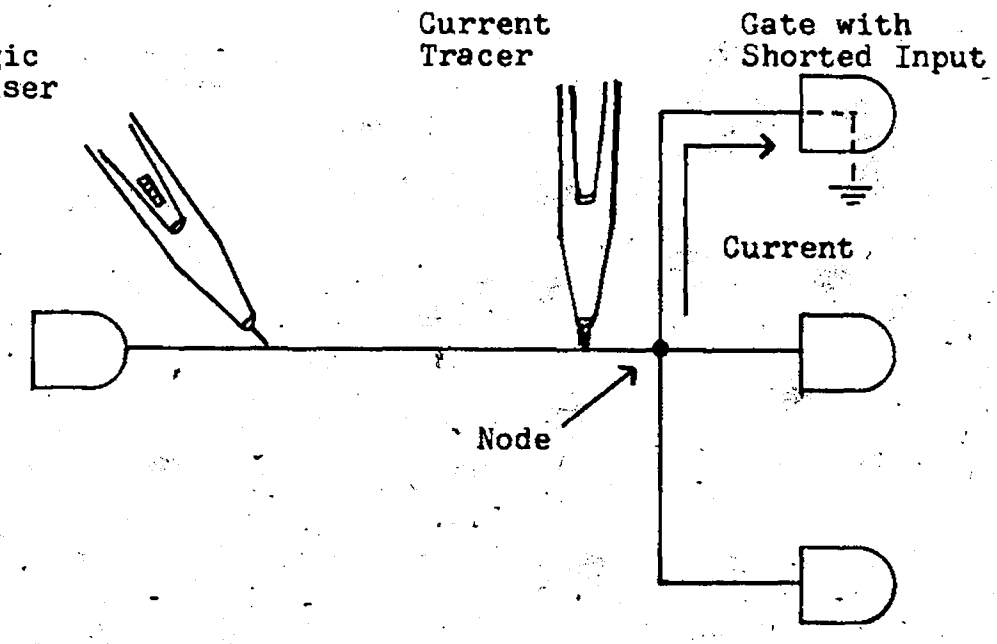
Equipment required: AIM 65 Computer
Logic pulser
Current tracer

Place the current tracer over the line going to pin 5, Z-13B. Adjust the sensitivity control for half brilliance.

You are observing the activity on Ø2 clock line.

Using the current tracer, trace the line back to it's origin i.e. pin 10, Z-16.

You can use the logic pulser to inject current pulses into a shorted line and use the current tracer to locate the short. The following diagram illustrates how this is done.



The current tracer can trace the pulse from the logic pulser to the shorted gate.

Learning Activity A4

Logic Comparator
Typical HP 10529A Logic Comparator

Objective: To apply the Logic Comparator to digital circuits.

The logic comparator compares the operation of a TTL IC, under actual operation, to a known reference IC.

In operation the comparator clips onto a powered IC and the same type of IC is placed in the comparator. The outputs of both IC's are compared and if they are not the same a LED is lit corresponding to the location of the error.

For troubleshooting purposes the logic comparator is simply clipped across a powered IC and a refrence IC is placed in the comparator. This sequence is continued until the bad IC is located.

Learning Activity A5

Oscilloscopes

Objective: To apply the Oscilloscope to digital circuits.

Stimulus response techniques, using generators and oscilloscopes, normally associated with analog circuits will not normally work with microcomputers. Signal lines are tri-state, data and address lines are multiplexed and only the CPU really knows the purpose of the Bus activity. However, there are applications suited for an oscilloscope with at least 50 MHZ bandwidth.

1 Monitor clock activity: all CPU operations are timed by a systems clock. The clock's waveform can be observed on an oscilloscope and it's frequency determined.

2 The first 12 address lines, the data lines and the R/W line can be observed on the oscilloscope by writing a short routine and having the computer continually loop through it. This gives the data and address lines a repetative frequency necessary for oscilloscope observation. The oscilloscope can be synchronized by the clock or read write lines.

3 The I/O ports can be observed on an oscilloscope if a short looping routine is used to continually address the I/O ports.

Application of the Oscilloscope

Equipment required: AIM 65 Computer

50 MHZ Dual Channel Oscilloscope

1 Record the voltage on Ø1 clock .....

Determine it's frequency

2 Record the voltage on Ø2 clock .....

Determine it's frequency

3 Connect Ø1 to channel 1

Connect Ø2 to channel 2

Observe that the two clocks are phase shifted by 180 degrees.

4 Load the following program into the AIM 65

```
03FD    JMP
03FE    FD
03FF    03
```

Execute the program

Connect the oscilloscope sync (trigger) to the Ø2 clock.

The address lines should repeat after 3 data bits.

Sketch the waveform you observed on address line 0.

Indicate time versus amplitude

Show it to your teacher for conformation.

Observe the data on all twelve address lines and the data lines. Sketch a graph of the data on data bit 2, show amplitude versus time.

Learning Activity A6

Logic State Analyzers
Typical HP 1602A Logic State Analyzer

Objective: To apply the Logic State Analyzer to digital circuits.

The logic state analyzer captures up to 64, 16 bit words at clock speeds up to 10 MHZ. Thse words are stored in the logic analyzers memory and may be displayed in Hex, Octal, Decimal or Binary. The logic state analyzer can be used to monitor the address bus, data bus, control lines or input/output activity.

Application of the Logic Analyzer

Equipment required: AIM 65 Computer
1602A Logic State Analyzer

Setting up the logic state analyzer to observe the data lines.

Connect the 8 LSB probes to the AIM 65 data lines
Clock to O2
Ground to Computer ground
Select positive logic polarity
Select negative clock edge
Select Hexadecimal
Word width = 8 (8 bits only on data bus)
Press Trigger = (used to set trigger word)

Pressing Trace instructs the analyzer to start looking for the trigger word.

Load in the traffic light program (see Learning Activity III B21)

Set trigger word, Press trigger = 20 (Hex for JSR)

Press Trace

Run traffic light program

64 data words will now be stored in the logic analyzer.

The data in the analyzers memory may be viewed on the display using the four keys in the display block.

They are: Next Word - view next word in memory

Prior word - go back one word

At trigger word - in this case 20

Word number - display a specific word

Compare the traffic light program to the data stored in the logic analyzer.

Set up the logic state analyzer to trace address lines.

Connect 16 probes to address lines
Clock to 02
Ground to Computer ground
Positive logic polarity
Negative clock edge
Select Hexadecimal
Word width = 16
Trigger = 0200

Load traffic light program

Press trace

Execute program

64 data words 16 bits wide will now be stored in the logic analyzer.

Use the 4 display keys to observe address words stored in memory. Compare to the traffic light program.

Learning Activity A7

Signature Analysis

Objective: To introduce the student to Signature Analysis.

Signature Analysis is an easy to use and highly accurate technique for identifying faulty logic nodes. The signature analyzer can convert the long complex serial data streams present on microprocessor system logic nodes into four-digit "signatures".

In order to use the signature analyzer the product under test must have been designed for test by the signal analyzer. Essentially the instrument under test must contain a ROM that generates a series of signals.

To test a computer or digital instrument switch on the signature analyzer and place it in the diagnostic mode. Place the signature analyzer probe on a designated node. Compare the signature on the node to that on the schematic diagram of the instrument under test. Each node on the schematic is marked with a specific signature. Once a bad signature is identified the faulty component can be easily located.

Component problems can be identified by detecting good signatures going into a component and bad signatures at the output. Bad components on a node can also be isolated by applying the current or logic probe in conjunction with the logic pulser.

B Troubleshooting Microprocessor Systems

The following lesson is from the "Hewlett-Packard Practical Microprocessor Textbook" and is included with the permission of Hewlett-Packard (Canada) Ltd.

LESSON 18

Troubleshooting Microprocessor Systems

INTRODUCTION

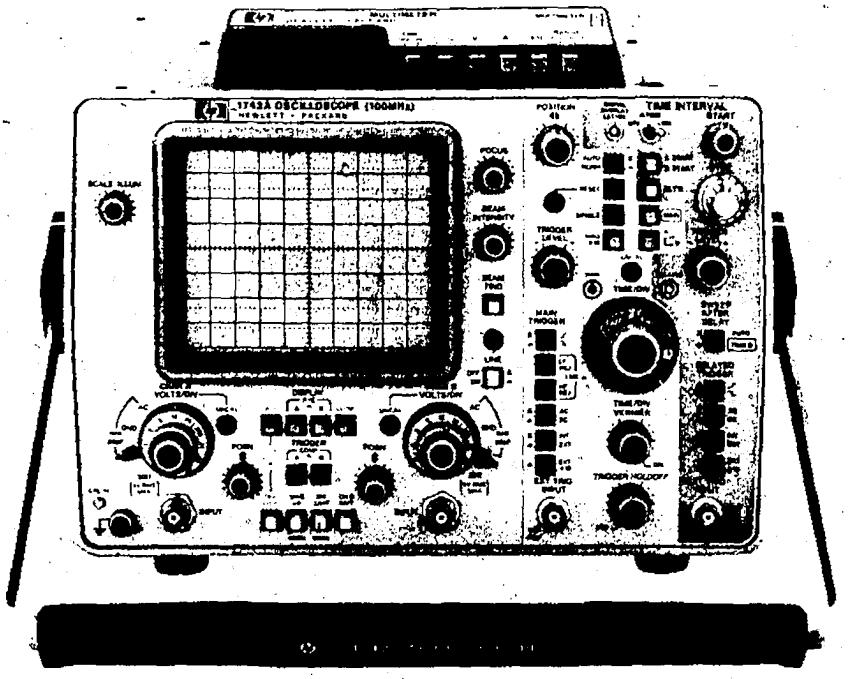
The troubleshooting philosophy for microprocessor-based products is fundamentally no different than for standard digital designs. As with any circuit you are trying to analyze or troubleshoot, it is helpful to first become familiar with the circuit. Studying the theory of operation, the block diagram, and the schematic provides a base of knowledge from which to work. In this lesson, problems relating to microprocessor systems and the troubleshooting techniques for dealing with them are discussed.

MICROPROCESSOR TROUBLESHOOTING PROBLEMS

There are a number of testing problems somewhat unique to microprocessor systems. For one thing, most of the control is in the software, so that signal flow is hard to trace. Another difficulty is that everything happens too rapidly to see in real time. In most cases, a microprocessor system, unlike many logic circuits, cannot be stopped and manipulated. Measurements must be taken while the microprocessor is running. This requirement reduces the effectiveness of the logic probe and pulser but enhances the usefulness of the current tracer, oscilloscope, signature analyzer, and logic analyzer because these instruments rely on circuit activity for their measurements.

Microprocessor bus structures pose additional difficulties. Data on these buses is often unstable or meaningless because of three-state outputs, multiplexing, and switching transients. These conditions cause no problems for the system itself, since it is synchronous and knows when the bus lines contain stable signals. The signature analyzer and the logic analyzer also know when these lines are valid, because of clock signals provided to them. The oscilloscope does not have this capability. It provides little quantitative information, but is useful for examining qualitative factors, such as general activity, logic levels, waveform timing, and bus conflicts.

Since bus structures also make it possible for many devices to be connected together on a single node, finding the one bad device on such a node can be difficult. The current tracer is useful for this purpose. The data bus also acts as a digital signal feedback path and tends to propagate errors through good



Oscilloscope Helps Identify Problems in Microprocessor Systems

circuits and then back to the fault source. The best way to deal with this problem is to open the feedback path when possible. Techniques for doing so are discussed in this lesson.

Complex devices are often connected to the microprocessor buses. It is difficult to test these devices using simple stimulus-response testing. The correct operation of these devices can be verified by swapping them with a known good chip, or by observing that the function they perform for the system is being performed correctly.

Microprocessors are sequential machines. Program flow depends on a long sequence of instructions and events. If even a single bit of information is incorrect, the whole system can go awry. Noise glitches and bad memory bits are the most common sources of single-bit errors. Others are also discussed in this lesson. These failures are difficult to pinpoint because the entire system may appear to be operating incorrectly.

Experience gained from doing the Microprocessor Lab troubleshooting experiments in Lesson 19 will provide you with a good foundation for troubleshooting other microprocessor-based products. Such experience can prevent the really difficult troubleshooting problems from being thrown under your bench (or worse). New things always seem more difficult at first, and the same is true of microprocessors. Designed-in serviceability and good documentation by the manufacturer can make troubleshooting much easier. The use of signature analysis and other high-level servicing aids can greatly reduce the task of troubleshooting.

Dozens of different microprocessors exist, and hundreds of people design products and service procedures for them. Since the μLab is specifically designed for educational purposes and for teaching troubleshooting, the concepts developed using the μLab should be applicable to many classes of microprocessor systems. It is as close to a typical (but small) system as is practical.

PROBLEMS SPECIFIC TO MICROPROCESSOR SYSTEMS

Clocks

Bad clocks can cause fouled, but "running," systems. There are a number of malfunctions that can result in system clocking problems. Clock problems can show up as a failure of the system to function at all (no activity), the ability to function only open-loop (free-running), or semifunctional activity (a meaningless and undefined program sequence). Some microprocessors are sensitive to clock speed. Since many systems run "at spec," even a small variation in clock rate (too fast) can cause system failures. If the system runs too slow, dynamic storage cells on ICs in the system may fail. Both of these problems are more likely to occur when resistor and capacitor (RC) clock circuits are used instead of the more accurate and stable crystal-controlled circuits. However, crystals can sometimes break into their third overtone oscillation mode, causing a much higher than expected clock rate. In addition, some processors require multiphase and nonoverlapping clocks with very stringent timing requirements. Also, clock voltage levels are not necessarily TTL compatible, but may be much wider in voltage swing. Microprocessor clock specs can be found on the device data sheets and can be checked using conventional frequency counters and oscilloscopes.

Power-Up Reset

The microprocessor's power-up reset circuit can also cause fouled, but running, system operation. A reset pulse that is nonexistent, too short, too noisy, or too slow in transition can start everything off on the wrong foot, resulting in out-of-sequence, partial, or no reset activity. Problems can also occur in reset circuits that are susceptible to power supply glitches. Even when Schmitt input circuits are used, slow edges can cause reset timing skew from one device to another within some systems. This will cause some of the devices to power-up before the others, resulting in erroneous behavior. A too rapid ON-OFF-ON system power sequence will fail to restart many systems (e.g., the μLab). It may then be necessary to increase the OFF time to allow the power supplies and restart circuits to discharge.

None of these reset failures will necessarily prevent the system from running. It may run for a short time and then stop, or lock up in a meaningless program loop, or even perform most of its normal operations. The key point to remember is that the system must complete the power-up reset sequence to insure that all of the test, control, and initialization operations necessary to bring the system up have been performed.

Power-up reset circuits are normally operative only when the system is initially powered-up. They can be monitored at that time with storage oscilloscopes, logic analyzers, and in some cases, signature analyzers. They can also be manually overdriven and controlled externally for testing purposes.

Interrupts

Stuck or noisy interrupt lines can cause faulty system operation. The system may work with a stuck line but it will do so very slowly (spending most of its time servicing the "phantom" interrupt). Noisy interrupt lines can cause sporadic system changes to occur, or peripheral inputs or outputs may take place at improper times. Sometimes the system will not respond at all to certain I/O devices, which can occur when a higher priority interrupt has disabled the lower ones.

Interrupt line activity can be monitored with a logic probe, logic analyzer, or oscilloscope. Interrupts are asynchronous in nature and can often be manually controlled (enabled or disabled) for testing purposes.

Signal Degradation

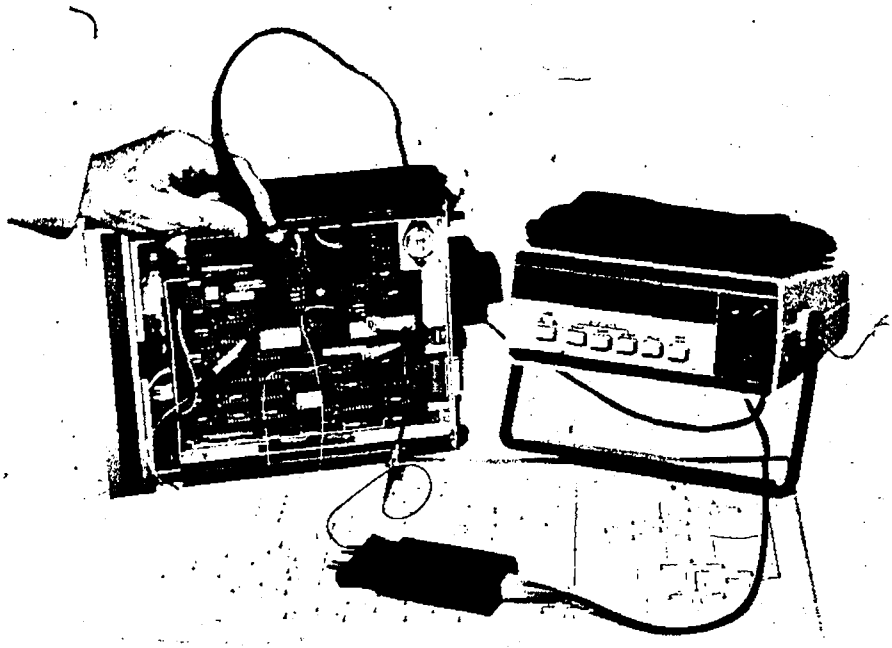
The long parallel bus and control lines present in medium-to-large microprocessor systems are sometimes susceptible to crosstalk and transmission line problems on critical lines (such as clocks and enables). These problems can show up as glitches on adjacent signal lines or ringing on the driving line (causing multiple transitions through a logic threshold). Either of these situations can inject faulty data or control signals that are very difficult to detect. This problem is most common when signal lines are long and already taxing the timing and noise margins of the system. When extender cards are added to these systems or high-humidity conditions exist, failures may occur. Cross-coupling of lines on extender cards can be a problem when fast signal transition lines (such as Schottky gate outputs) run alongside other signal lines, even when they are on opposite sides of a PC board.

Memories

Memory failures in microprocessor systems can produce deviant system behavior in a number of ways. Anything from a total system failure to a single faulty bit of stored data can occur. Most memory failures can be found during the power-up self-test program, unless the memory failure prevents this program from running. If the system doesn't do a RAM verification test and no RAM test service fixtures or procedures are provided, it is nearly impossible to test the RAM. You will probably need to resort to substitution techniques when a RAM becomes suspect.

RAM failures occuring in the area of the memory used for the stack will usually cause the system to crash, even for a single-bit error. Otherwise, RAM failures may cause soft errors that result in unreliable system operation. Faulty dynamic RAM refresh circuitry is another factor to consider in diagnosing apparent RAM failures.

ROMs can also fail. Such failures are more frequent when nonmask programmable types are used. A single bad bit could crash the system or, even worse, 99 percent of it could work and 1 percent could produce erroneous results. ROMs can be effectively tested during power-up self-test, if such tests are



Using Signature Analyzer to Troubleshoot Microprocessor-Based Product

designed in. But, unlike RAMs, ROMs can also be tested by other techniques if no self-test is provided. One such technique involves free-running the system and then using the signature analyzer to either verify documented signatures or compare the outputs of a suspected ROM with that of a ROM in a known good system (see Experiment 17-2).

SELF-TEST PROGRAMS

The programmability of microprocessor-based systems can be used to great advantage in assisting system testing. Programs stored in the system's ROM can test ROMs, RAMs, and the processor itself. Often the I/O can be tested to some extent. Software can also be used to provide stimulus for an external test instrument, such as a signature analyzer.

ROM Testing

The most common technique for testing ROMs uses a *checksum*. When the ROM is programmed, all of its words are added together, ignoring any carries that result. This number is complemented and stored in the last (or sometimes the first) word of the ROM, so that when all the words are added together (including the checksum stored in the last byte), the result is zero. If the total is not zero at the end of the test sequence, then something is wrong with the ROM. (In actual practice, the checksum is usually calculated to make the total a specific number other than zero.)

Unfortunately, the checksum is not totally reliable. It detects any single-bit error and most multiple-bit errors; however, there are many combinations of two or more errors that still produce the correct checksum. Thus, a ROM that passes a checksum test is probably good. If the test fails, something is definitely wrong (though it might not be the ROM itself).

RAM Testing

RAMs are tested by writing a pattern into the memory, reading it back, and then verifying that has changed. Of the many different patterns that can be used, a common one is the checkerboard. In this pattern, all the bits are set to alternating ones and zeros. Once all memory locations have been tested, the pattern is repeated with each bit reversed, verifying that each bit of the RAM can store a one and a zero. Many other patterns used to test RAMS are specifically aimed at detecting various failure mechanisms within the RAM.

No memory test can guarantee 100 percent accuracy, even though it may show that each bit can store a one or a zero. RAMs can be pattern sensitive. For example, one location might correctly store 01010101 and 10101010 but fail when 01111000 is stored. Even for a small RAM, it would take an extremely long time to test every possible pattern sequence. For this reason, RAM test credibility is generally much lower than that of ROMs. As with the checksum test, if a RAM passes the system self-test program, it is probably good. If it fails the test, something is definitely wrong.

MULTIPLEXED I/O

Multiplexed keyboards and displays often share some of the same scanning circuits (as does the μLab). In these situations a stuck key can appear to make the display fail. Likewise, a bad display driver input could cause a keyboard error. The interaction between common scan circuits must be considered in making a diagnosis.

INTERFACES

Many microprocessor systems interface with other systems through external communication lines (e.g., IEEE-488, RS-232C, telephone modem). These lines are frequently long and are often exposed to sources of electrical interference, such as relays, transformers, motors, solenoids, and even lightning. Electromagnetic interference (EMI) emanating from these sources can cause the transmission of faulty data, overstressing of interface circuits, and, especially in the case of lightning, gross component failures. Generally, output line driver circuits tend to have higher-than-average failure rates, due both to EMI stressing and to the high transition currents that result from driving capacitive interfacing cables.

TROUBLESHOOTING TREES

A *troubleshooting tree* is a graphical means of showing the sequence of tests performed on a product under test. These trees are often drawn as flowcharts in which the results of each test determine what step is taken next. The use of troubleshooting trees for repairing microprocessor-based products can save considerable time and effort.

Figure 18-1 shows a portion of the troubleshooting tree for the HP 3455A Digital Voltmeter. Theoretically, it should lead you to the product's fault by means of the actions taken and decisions made along the tree. Unfortunately, such is not always the case. A perfect troubleshooting tree must consider all possible failures, a difficult criterion for the person writing the troubleshooting tree to meet. Also, troubleshooting trees tend to be fairly generalized, lacking the specifics desired for making tests and decisions. Few troubleshooting trees provide practical information about how a specified test or measurement relates to what the circuit does or is supposed to do. If the troubleshooting tree fails to direct you to the actual fault, you may be left at a dead-end, with no idea of where to go next. However, the troubleshooting tree will often be your best guide (at least to begin with).

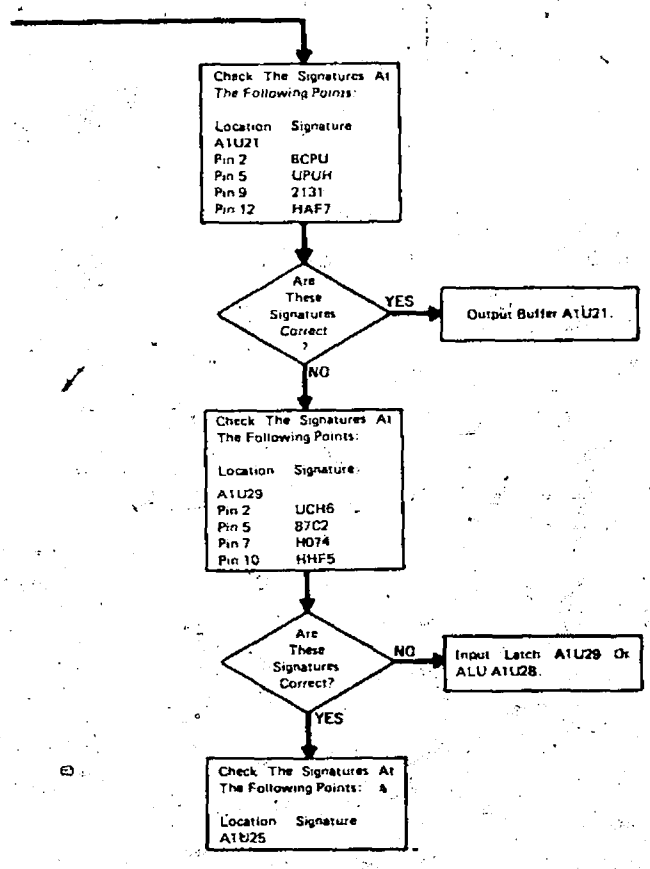


Figure 18-1. Typical Troubleshooting Tree for Product Incorporating Signature Analysis

There are good troubleshooting trees and there are bad troubleshooting trees. The good ones seldom lead to a dead-end and provide a logical, well-directed sequence of tests and measurements, requiring a minimum level of understanding of the circuit under test. Often they include advanced techniques such as signature analysis to simplify the procedure. In troubleshooting a product, even the poorer troubleshooting trees can be useful for localizing a failure area in the system and can save considerable time and effort.

OTHER DOCUMENTATION

For many experienced troubleshooters, working from product *block diagrams* can supply the right amount of information to understand how the different parts of the circuit work together. A product's theory of operation and its troubleshooting trees do not relate as closely to the hardware. The schematics often provide too much detailed information, making it difficult to see the "big picture."

The remaining portion of this lesson outlines a loose sequence of general steps that you can take to troubleshoot a microprocessor-based product. Numerous servicing techniques and "tricks of the trade" are interspersed with the descriptions.

IS THERE REALLY A PROBLEM?

It is important to have a general understanding of the defective product so that you can be sure that a problem really exists. To some degree, you should know what it does and how it operates. Microprocessors allow designers to design products that are not only complex in function, but sometimes complex to operate as well. Be sure the apparent problem is not a user error, but a real product malfunction. Few things are more frustrating than trying to fix something that is not broken. In some situations, it appears that a product should do something it was not actually designed to do. For example, a DVM AC select switch may work on VOLTS but not on AMPS. This "design limitation" can usually be verified in the operating manual and does not constitute a product malfunction; it is only a shortcoming.

Design "bugs" in the firmware (ROM) can sometimes cause failures when used under operating conditions that were not anticipated during the product design. These are more likely to occur in early production runs and can best be verified (if suspected) by contacting the manufacturer. At the other extreme, a problem may actually exist but not show up because the product is not adequately exercised. These kinds of problems are often very simple to detect (e.g., observing a burnt out OHMs LED indicator when pushing the OHMs button on a DVM). They can also be complex problems. For example, errors can occur when an unusual sequence of operations is performed. Because the complex problems are much more difficult to test for, extensive test procedures are used to test products at the factory. The customer, bringing in a product for repair, has no trouble pointing out a problem. It is up to the troubleshooter to solve it.

WHAT CAN BE LEARNED FROM THE FRONT PANEL?

A great deal of diagnostic information can often be obtained without even removing the product's covers. Most microprocessor-based products have some sort of front panel. On it there may be switches and indicators, inputs and outputs. *Milking* the front panel is a process in which the switches, buttons, and other inputs are used to solicit responses from the product that can be observed using its indicators and other outputs. For instance, if the indicators are all dead when the power is turned on, you might suspect a bad switch, fuse, power cord, battery connection or power supply. If one segment of a display is dead, the problem is probably the display itself or the circuit that drives it. If the only failure of a DVM is in the 1-10 VOLT range, the problem area can be narrowed down to a relatively small portion of the circuit (the attenuator).

Always take advantage of any designed-in performance verification or power-up test modes and diagnostic messages that are available. These are specified in the product manual.

At this point you may have some idea of where the problem is or you may have even fixed it. But in all likelihood, neither has taken place.

WHAT DOES THE MANUAL SAY?

"If all else fails, look at the manual." This rather poor (but prevalent) attitude makes even less sense for microprocessor-based products than for conventional ones. There may be a bonanza of service aids and procedures in the manual just waiting for you to try out. Special service switches, jumpers, test fixtures, indicators, and test techniques can make the job much easier.

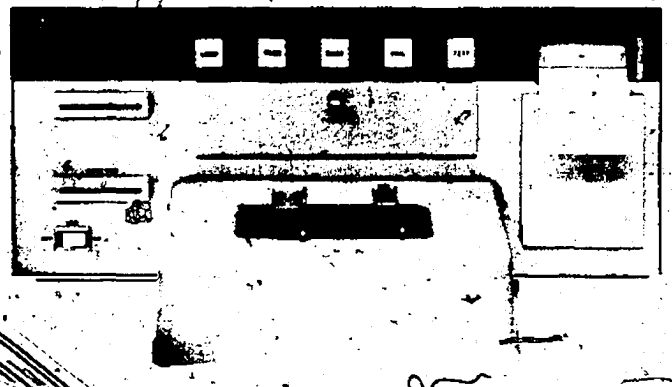
Try to understand the circuits and figure out where things are. Check out the manual's theory of operation section, the block diagrams, and the schematics.

You do not have to do this in great detail but just enough to have some idea of what is going on. Identify the microprocessor, ROM, RAM, I/O, address decoder, clock, bus, control, and interrupt portions of the system.

PRODUCTION VERSUS FIELD FAILURES

The life of an IC is generally a sequence of predictable events. It is born in the IC factory and is sent to a product manufacturer. There it is inserted into a circuit board, which in turn is inserted into a product. Then the product goes into service, and the IC remains there for the rest of its useful life. Needless to say, not all ICs live a long and healthy life.

Product manufacturers estimate that approximately 2 percent of all incoming ICs are defective. Testing incoming ICs on an IC tester will detect most of these. The effective cost of finding a defective IC at this point is about 10 cents. Once ICs are loaded into circuit boards, the bad ones cost about $1 to find. If they are not detected until the boards are assembled into the end product, this "in-situ" troubleshooting and repair costs about $10 at the factory. Replacing a bad IC in the field is even more expensive: a typical bill for finding and replacing a faulty IC in a customer's product is about $100. Clearly, it makes sense to find and eliminate the defective ICs as early in the cycle as possible.



HP 5045A Digital IC Tester Used to Perform Incoming Inspection

Types of Failures

Common fault sources and the best troubleshooting techniques for finding them depend on the history of the product and the environment in which it is tested. When a new product is first turned on at the factory, almost anything might be wrong with it. Products that fail in the field have all worked at one time. Assembly errors, such as misloaded components and miswired circuits, generally need not be considered in field failures. Also, the likelihood of solder shorts and multiple faults is much greater on the production line than in the field. Field failures are usually caused by components or connections that have failed.

Automatic Testers

Because of the volume of identical products tested at the factory, specialized testing and troubleshooting equipment and techniques can be justified. Automatic board testers and test fixtures are often used to minimize the time it takes to locate faults. In general, they provide fast, economic, and accurate verification and fault diagnosis. Do not, however, fall victim to overconfidence in computer-controlled automatic board testers. Occasionally, boards passed by a production board tester are actually defective as a result of deficiencies (timing, loading, or component exercising) in the tester or the test program being run. However, newer board testers that perform more sophisticated dynamic, functional, and parametric tests have greatly increased credibility.

HP 3060A Board Test System Performs Fast, Thorough, and Efficient Testing in Production Environment

WHAT ARE THE EASY THINGS TO TEST?

It makes sense to look first at the things that can be tested and repaired easily. The simple things are as likely to fail as the complicated ones. A case in point is the power supply: it is actually one of the more failure-prone portions of most products. It is also one of the easiest to test and is usually simple to troubleshoot. An out-of-spec voltage can cause erratic circuit performance. If the voltages are not checked first, it could take considerable time to find the problem.

A mechanical inspection can also be fruitful. Poor PC board and cable connectors, broken wires, and loose parts can usually be found either visually or by touch.

COMMON PRODUCTION-LINE TROUBLESHOOTING PROBLEMS

A number of common sources of failure in a manufacturing environment can be found through careful visual inspection of a product's circuit assemblies. It is easy to check for improperly set switches and jumpers, misloaded components (wrong ones and backward ones), and cold solder joints. Backward resistor packs can be particularly hard to diagnose electrically because they can cause interaction between unrelated logic nodes, but they are easy to check visually.

Two of the more common failures in production are solder and gold (copper) shorts on printed circuit boards. These can usually be removed with a sharp knife. When the precise location of the short is not known, there is a rather novel technique for removing it that often works. It is also useful for situations in which the location of the short is not accessible (such as inner layer shorts on multilayer boards). The procedure involves charging a 100,000 μF (or larger) capacitor

to five volts (a safe voltage for logic circuits). Then, with cables solidly connected to the two shorted nodes and proper polarity observed, discharge the capacitor into them and listen for a snapping sound on the board. Check continuity to see if the short has been opened and, if not, try again. This technique should be used with caution since it will open the weakest link of the current path, which may not always be the fault source, but may be a fine trace or a plate-through. The current tracer provides a much safer means for finding shorts, as demonstrated in Experiment 16-4.

A relatively new problem in production is the occurrence of bent-under IC pins caused by automatic component insertion equipment. These can result in an open electrical connection between the IC and the PC board, an intermittent connection, or shorts to traces near or under the IC. The bent-under pin is often difficult to spot visually because it may look as though it is properly soldered in place. The best way to tell is to look at the bottom of the board for the ends of any IC pins or along the plane of the board to see under the ICs.

PC board edge connectors are commonly used. They may cause problems in production when their borders are cut off center or when they are accidentally covered with solder resist or board sealing spray. Visual inspection can reveal such problems.

Multilayer PC boards suffer from all of the problems of regular boards plus some of their own. Misregistration and contamination of inner layers (which can cause high frequency or leakage problems) can often be observed by holding the board up to the light. Since repair of the inner layers is often impossible, the entire board may have to be scrapped.

Wire-wrap boards are prone to bent posts that cause shorting. Other common production problems include 14-pin ICs loaded into the wrong end of a 16-pin socket, miswiring, wire shorts between pins, and signal coupling (crosstalk) due to closely bundled wires.

MECHANICAL FIELD FAILURES

Visual inspection of a product that fails in the field can reveal such things as loose wires, broken traces, cracked ceramic ICs and resistor packs, bent wire-wrap posts, and dirty connectors. A "calibrated fist" on the side of the cabinet can often be used to detect loose or intermittent connections and stuck relays. Mechanically stressing boards and connectors (by twisting and flexing) can often help to locate some of these problems. You might suspect the PC board edge connectors when a product is "D.O.A." (dead on arrival) or fails in a hostile physical environment. You may want to try reseating all of the assemblies and circuit board connections to determine if the problem is poor connector contact. A pencil eraser is useful for cleaning dirty edge connectors.

GENERAL TROUBLESHOOTING TECHNIQUES

Board Swapping

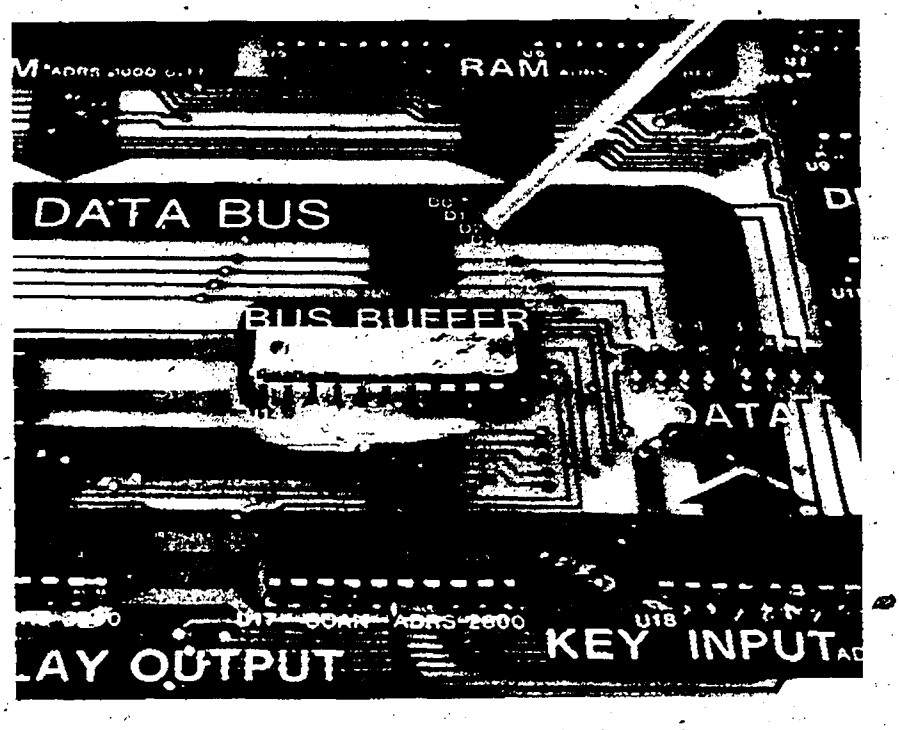
If any of the PC boards are easy to remove and replace and known good ones are at hand, you can try swapping them. When duplicates of the same board or assembly are used in one product, they can be swapped with each other. The risk involved in board swapping is that you could damage a good board because of the same electrical overload that damaged the bad one when it was installed. In any case, power to the product should be turned off when removing or installing boards or assemblies.

If an identical product is available, functional comparisons can sometimes be informative. This comparison can be especially useful in situations in which it is not clear that there is actually a hardware problem (it may be a product idiosyncrasy or design limitation).

If a device in a socket is suspect, try tapping it first to see if there is a loose connection and then try substituting a known good one. Note, however, that one of the last devices you should suspect, but that is most often the first to be replaced, is the microprocessor. The actual failure rate for microprocessors is very low. However, because they are complex and their correct operation is difficult to verify, they are often the first to be plucked from a PC board. This is also true of the LSI chips used with them.

Stress Testing

A technique referred to as *stress testing* can be very effective in dealing with marginal or intermittent failures. Stress testing can often cause these types of failures to temporarily improve or deteriorate; either case is beneficial in locating a fault. Boards are stressed physically by tapping or twisting them, thermally by heat (air gun or hair dryer) or by cooling them (from an aerosol freeze can), and electrically by varying the supply voltage. Thermal stressing can be used to isolate a fault in a specific device on a board more precisely than the other methods because heat or cold can be applied directly to a single component. Intermittents can result from marginal chips, lead bonds, solder joints, connections, and drive and timing circuits.



Cold Spray Helps Identify Faulty and Marginal Devices

Briefly touching each device on a circuit board can pinpoint a component that is running hot (much hotter than the others). When a particular device runs significantly hotter than others of the same type, a problem may exist. A faulty device

can sometimes be hot enough to burn your finger, so use this technique with caution. Be aware also that some good devices may run hotter than you expect during normal operation, and that temperatures may vary widely from one device to another.

Power Supply Shorts

There are some effective ways of dealing with shorts across the power supply. The first thing to do in a multiboard system is to try to localize the short to a single board. This can be done by removing one board at a time until the power supply is no longer shorted. The last board to be removed is the shorted one.

One technique for finding the short on a faulty board is to inject current through the two shorted lines with the logic pulser. The current tracer is then used to follow this current to the short. Keep in mind that capacitors (especially electrolytics) will have some current going into them because of the pulsing current. Shorted capacitors can be found by using the current tracer to compare the current levels going into identical capacitors on the same board. The capacitor that shows a much higher level than the others is likely to be shorted. This technique is particularly useful for finding shorted ceramic bypass capacitors.

Another technique for locating power bus shorts is to supply a relatively high current (about 3–5 Amps) into the short. Be sure to maintain the same voltage polarity and not to exceed the supply voltage normally present. The current path to the short can often be determined by using a DVM with high resolution (.01 mV) to look at voltage drops on the power bus traces. Voltages are developed across the traces that are in the path going to the short, and not elsewhere (see Figure 18-2).

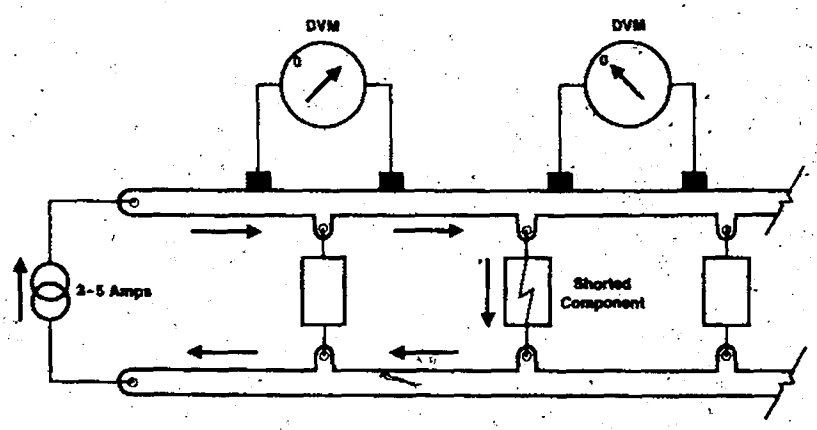


Figure 18-2. Using Sensitive Voltmeter for Locating Power Bus Short

A less scientific, but much more dramatic, technique for finding power bus shorts is to freeze the entire board (to about -10 degrees C), allow moisture to condense on it, and then power it up with a 3–5 Amp supply. As it warms up and defrosts, the current path becomes visible and, in many cases, will pinpoint the short.

HOW CAN THE FAULT BE ISOLATED?

Once the easy things have been tried unsuccessfully, it is time to get down to business. At this point individual troubleshooting skills, intuition, and knowledge of the product really make a difference.

First, be sure to take advantage of any designed-in and documented circuit isolation features, such as selected board removal, service jumpers, and special test modes and procedures. It can be very useful to separate the microprocessor system from the peripheral circuits to allow you to diagnose each portion independently.

An important troubleshooting concept is *half-splitting*. Although the term may be new to you, you've probably been using the process for years without even knowing it. Half-splitting involves choosing a point roughly in the middle of the circuit. It is just as likely that a fault exists before as after that point. If the performance is correct up to that point, the fault lies after it. If not, then the fault is before that point. This process works best in circuits that have clear, unidirectional signal paths without large feedback loops. Even with microprocessor-based systems, this approach can be effective because the circuits outside the microprocessor portion often fit these guidelines.

In a typical product, the first half-split is generally done at the digital-to-analog interface, if possible. Analog circuits often have higher failure rates (due to higher demands made on speed, power, temperature, sensitivity, accuracy, adjustment, external overloads, and reduced component safety margins). The contribution of a product often relates to its analog circuits. These are often the circuits that represent the "high-technology" contribution and that may be operating near their limits. They may also outnumber the digital ones. Be aware also of the possibility of the electrical interaction of clock and TTL power bus lines with analog circuits, which can cause serious system noise problems.

DIGITAL FAILURE MODES

When suspicion falls on the digital portion, the first thing to look for is signal activity. With a logic probe you can examine activity on the clock signals, bus lines, chip enables, and control lines. Absence of activity on any of these nodes indicates a possible problem. You may wish to refer back to Experiment 16-1 to refresh your memory about troubleshooting with the logic probe.

The most common failure mode for digital ICs is open lead bonds inside the package. There are thin wires connecting the package pins to the IC chip. If an output lead bond opens, the output pin floats and the logic probe will probably indicate a constant floating logic level because of other device inputs connected to that node. If an input lead bond opens, one or more of that IC's outputs will usually appear to malfunction (stuck high, low, or executing its logic function incorrectly). If any of these outputs goes to a three-state bus, it can cause bus conflicts (more than one output on at a time), and the current tracer can be used to find these. Bus conflicts are often observed on an oscilloscope by the presence of bad, but solid, logic levels on bus lines, but the scope provides no information as to the source of the fault (see Figure 18-3). Good bus lines can also appear to have solid, bad levels present when all devices on the bus are off.

Another common digital IC failure is a shorted input pin to ground. This fault is often caused by a bad input protection diode on the chip. It usually appears as a stuck low level, which can be seen with a logic probe. An oscilloscope connected to a node with this type of problem shows a voltage level near ground being pulled up, perhaps a few hundred millivolts, whenever a logic 1 output on that node turns on (see Figure 18-4). The current tracer provides an excellent means of pinpointing shorted input pins.

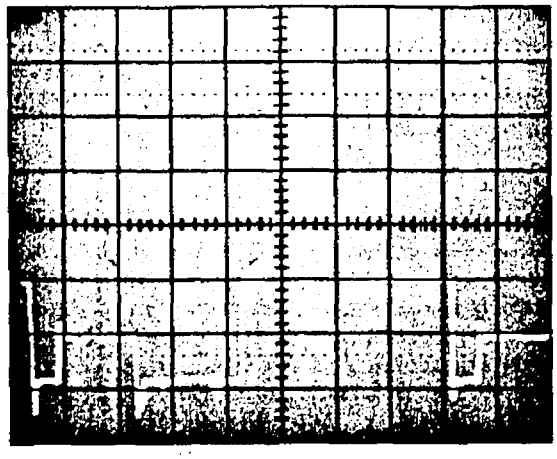


Figure 18-3. Bus Conflicts Cause Bad, But Solid, Logic Levels

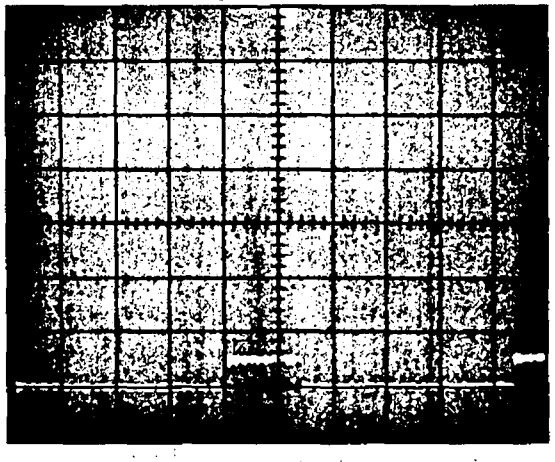


Figure 18-4. Shorted Substrate Diode on Gate Input Pin Clamps Node to Ground

If a current tracer is not available, another means for locating stuck inputs and outputs involves the use of a sensitive (high resolution) DVM and a can of cold spray. Connect the DVM to the stuck node and select the most sensitive DC voltage range available. Then, while monitoring the voltage, spray each IC connected to the stuck node, one at a time, to change its temperature. Any noticeable change in voltage (more than 10 mV) on the node indicates that the IC being sprayed is drawing current. If a freeze can is not available, a heat source can be used instead. This technique relies on the properties of the semiconductor material used in the IC that relates voltage to temperature.

ISOLATION TECHNIQUES

Once a particular input or output pin is suspected, it is useful to isolate it from the rest of the circuit. A quick, nondestructive way to do so is to suck the solder away from the area between the pin and the PC board pad, using a vacuum desoldering tool or solder wicking braid. Then bend the pin so that it is centered in the pad's hole, not touching it at any point. Use a continuity tester to verify that the pin is no longer in electrical contact with the board.

The techniques that you can use to isolate the digital blocks of a microprocessor-based product are entirely dependent upon its electrical and mechanical architecture. If some of the digital boards can be removed and still allow the kernel to operate, this procedure can be useful. If the kernel can be allowed to run open-loop (no feedback from the data bus), a free-run mode can sometimes be used to check the kernel and address bus activity.

An extender board with switches on bus and signal lines can be used to break selected signals between a PC board and the rest of the system. In this manner, feedback paths and stuck buses can be removed from the main system.

An even simpler way to open selected signals going through a board edge connector is to place a piece of tape or stiff paper on the PC board edge fingers that you wish to isolate. Be careful to note to which board(s) you have done this to so that you will remember to remove the tape or paper later.

A somewhat unconventional, but often effective means of detecting bus line problems is to measure the resistance to ground (with the power off) of each of the bus lines in a particular bus (e.g., data bus, address bus). The resistance of each of these lines is usually the same. If any one differs substantially, you may suspect a problem on this line. If two lines show the same (lower) resistance, the two lines may be shorted together. In either case, check the schematic to see if the arrangement of circuits connected to these lines could explain the differences before going further.

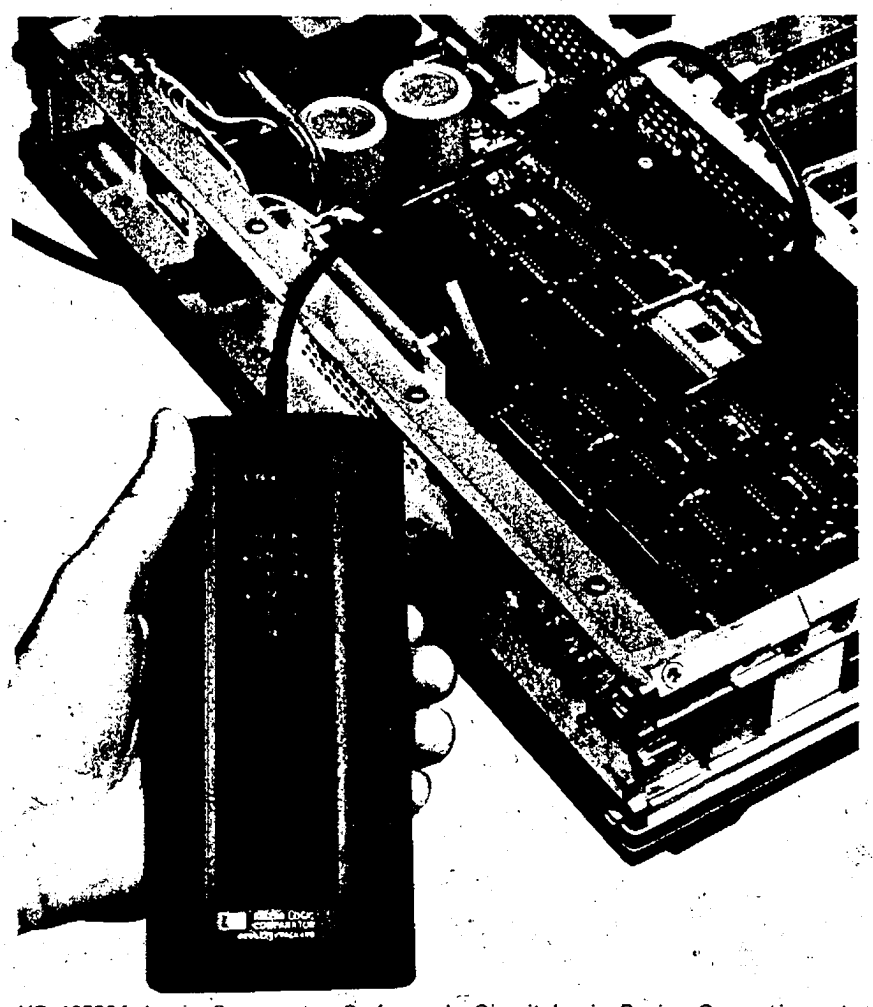
Overriding interrupt lines and chip enable pins on suspected devices can be used to verify that the IC is functioning correctly. This can be done by momentarily shorting the appropriate pin high or low, or by using a logic pulser (refer to Experiment 16-2).

FEEDBACK LOOPS

Digital feedback loops are often difficult to troubleshoot because errors propagate around and around. A feedback loop with a faulty output signal sends this signal back to the input to produce more bad outputs. Opening this feedback path prevents the faulty output signals from going back to the input. Then, if controlled inputs to the loop can be generated, the signal flow from the input to the output can be observed. Often, however, it is not easy to provide this input (many lines may need to be controlled). It may also be difficult to predict correct circuit operation. If another working product (or board with the same circuitry) is available, it is sometimes practical to allow the output of the good circuit to

control the inputs of both circuits. In this manner, you know that the circuit under test is getting the correct input signal. It is then a matter of comparing the nodes of the two circuits and looking for differences. A signature analyzer can be useful for doing this comparison.

Piggy-backing ICs is a technique that can sometimes be used to locate defective ICs. It involves looking at suspicious IC outputs with an oscilloscope or signature analyzer and then placing an identical IC package directly on top of it. The pins should be bent slightly, if necessary, so they are all in contact. A signal change can indicate problems with that device. If no change is observed and the output is not stuck, it can generally be assumed that the IC is not the problem. Be cautious of sequential circuits (such as counters and shift registers) that may cause output differences because of start-up conditions. A better way of performing this test is to use an IC comparator, such as the HP 10529A Logic Comparator.



HP 10529A Logic Comparator Performs In-Circuit Logic Device Comparisons to Known Good Reference.

CONCLUSION

No amount of knowledge and experience can totally compensate for inadequate service documentation. In some cases, shotgunning (replacing components until the problem disappears) may be the only solution. Most microprocessor-based products, fortunately, do not fall into this class. Future products will probably incorporate advanced service techniques, such as signature analysis, as more designers realize that the old troubleshooting methods and tools used for random logic are not very effective in dealing with microprocessors.

Occasionally Shotgunning Produces Unfavorable Results

REVIEW

Lesson 18

Microprocessor systems can be thought of as an extension of traditional digital logic. Many of the components, circuit designs, and troubleshooting tools and techniques are the same. However, there are some differences. Microprocessor systems are bus structured, and many of the devices on the bus are complex LSI devices. The signal activity between the devices on the buses is constant and complex. It is often useful to break the data bus, which is the system's main feedback path, to help isolate a fault that causes the entire system to malfunction.

Although troubleshooting trees provide an orderly approach for locating system faults, they are not always adequate. There are numerous techniques, procedures, and tricks that can be effective in diagnosing, isolating, and locating faults in microprocessor-based products. Many of these were discussed.

QUIZ

Lesson 18

1. In microprocessor buses, oscilloscopes are least effective when looking for:
   a. improper data.
   b. faulty logic levels.
   c. timing problems.
   d. bus conflicts.

2. The most effective tool for finding the defective device on a stuck bus is the:
   a. signature analyzer.
   b. logic analyzer.
   c. oscilloscope.
   d. current tracer.

3. A potential problem with troubleshooting trees is that:
   a. they are hard to follow.
   b. they have termites.
   c. they require too much knowledge of the product.
   d. they may lead you to a dead-end.

4. The first troubleshooting step should be to:
   a. read the product service manual.
   b. check the fuse and power cord.
   c. shake the product and listen for rattles.
   d. determine the nature of the problem.

5. A key requirement for *half-splitting* is:
   a. unidirectional signal paths.
   b. SA test modes.
   c. board swapping.
   d. having a good comparison product available.

6. The most common failure mode for digital ICs is:
   a. a wrong chip in the package.
   b. a shorted input diode.
   c. an open lead bond.
   d. a bad output logic level.

VI Programming in Assembler Language

Programming the 6800 microprocessor is a self-instructional workbook for assembly language and machine code programming for the 6800 family of microprocessors and peripherals.

The 6502 is essentially an upgrade of the 6800. The 6502 and it's peripherals are very similar or the same as those of the 6800 family. Enough similarity exists between the two that a student can, by completing the workbook, become skillful in programming a computer using the 6800 and at the same time develop skills that are transferable to the 6502.

The following table compares the registers and control signals of the 6800 and the 6502:

	6800	6502
Program Counter	16 Bits	16 Bits
Accumulator	Accumulator A Accumulator B	Accumulator A
Condition Code Register	6 Flags	7 Flags status register
Index Register	X register 16 Bits	X register Y register 8 bits each
Stack Pointer	16 Bits located anywhere in memory	8 Bits located in zero page only
Arithmetic Logic Unit	works primarily with Accumulator A	works primarily with Accumulator A
Instruction decode and control	72 instructions	56 instructions
Abbress Bus	16 lines	16 lines
Data Bus	8 lines	8 lines
Clocks	01 and 02	01 and 02
Interrupts	IRQ NMI Reset	IRQ NMI Reset

CPU control	Halt	RDY
Control lines	R/W VMA DBE BA TSC	R/W Sync Set overflow

Control lines or registers that are not similar in the two processors are discussed below.

Accumulator B, the 6800 has one additional accumulator however most operations are carried out using accumulator A.

RDY (ready), delays execution of any cycle during which the RDY line is pulled low. The RDY function will not stop the processor during a write cycle.

HALT, a low on the HALT line causes the CPU to stop.

VMA (Valid Memory Address), this signal is output high whenever a valid address has been output on the address bus.

SYNC, a signal is provided to identify those cycles in which the processor is doing an opcode fetch.

DBE (Data Bus Enable), will enable the bus drivers when in the high state.

BA (Bus Available), the bus available signal will normally be in a low state; when activated, it will go to a high state indicating that the microprocessor has stopped and that the address bus is available.

Set Overflow, not normally used.

TSC (Three State Control), this input is used to float the address bus and the read/write control output.

Common addressing modes:

	6800	6502
Accumulator	either A or B	only A
Immediate	same as 6502	same as 6800
Direct	6800 terminology	called zero page
Extended	6800 terminology	called absolute high and low byte order reversed

Implied	same as 6502	same as 6800
Relative	same as 6502	same as 6800
Indexed	uses all of memory	only page zero is used

The 6502 has 6 other addressing modes not found on the 6800.

Probably the main difference between the two is the pipelining concept, that is, the 6502 will address new data while it is still processing opcode. This gives the 6502 a look ahead feature that speeds up execution time.

Programming the 6800 Microprocessor

The titles of the chapters are listed with comments if applicable. It is important to do all the exercises, if you have a problem kindly ask for assistance.

1 Review of binary and Hexadecimal numbers.

2 Accumulator operations: Similar to the 6502 except two accumulators are used.

3 Symbolic Addressing.

4 Index Register: Intoduces addressing via the index register.

5 Branching-assembly language.

6 Branching-machine code.

7 Asynchronous Communication Interface Adapter: Probably this chapter should be done last.

8 Peripheral Interface Adapter: Similar to the Versitale Interface Adapter.

9 Subroutines: Similar to the 6502.

10 Stack Operation: Similar to the 6502.

11 Interrupts: Similar to the 6502.

After you complete the workbook you will have gained considerable skill in writing programs in 6800 assembler language and machine code and you will also be able to program in 6502 assembler code.

Switch on the AIM 65, type N, you are now ready to program the AIM 65 in assembler language. For more information see page 5-1 in the AIM 65 Users Guide. Happy Programming!

APPENDIX B

GLOSSARY

Accumulator: A special purpose register in which the results of Arithmetic Logic Unit operations are placed.

Acoustic Coupler: Device for connecting the telephone handset to the computer input port.

A/D : Analog to digital. Conversion from sensor's analog voltages to the digital representation used by computer systems. This is so computers can sense the "real world".

Address: Number indicating the position of a word in the memory. Typial addresses range from 0 to 64K.

Algorithm: A set of well-defined instructions to carry out a process in a finite number of steps.

Alphanumeric: The set of all alphabetic characters and numeric chracters.

ALU : Arithmetic Logic Unit.

Analog: Having a continuous range of voltage or current values.

ANSI : American National Standards Institute.

Arithmetic Logic Unit: Element which can perform the basic data manipulations in the central processor. Usually the ALU can add, subtract, complement, negate, rotate, AND and OR.

ASCII : America Stndard Code for Information Interchange. Character code used for representing information in most non-IBM or Western Union equipment.

Assembler: A program that translates assembly language into machine language.

Assembly Language: A computer language that uses mnemonic names to stand for one or more machine language instructions.

BASIC: An acronym for Beginners All Purpose Symbolic Instruction Code. A high-level conversational, interpretive, programming language in wide use.

Baud: Bits Per Second. Actually binary units of information per second. Teletypes transmit at 110 baud. Each character is 11 bits, and the TTY transmits 10 characters per second.

Binary Number: Representation of a decimal number in the binary system, using a sequence of 0's and 1's.

Bit: Contraction of Binary Digit. A bit is a "0" or a "1".

Breadboard: A breadboard refers to a prototype circuit. Comes from when radios were made on mother's breadboard.

Boolean Logic: Named after George Boole who defined binary arithmetic and logical operations such as AND, OR, NOT, and XOR.

Bootstrap: Program used for starting the computer. Usually clears memory, sets up I/O devices, and loads the operating system from ROM, disk or cassette.

Breakpoint: Software or hardware device which will stop a program and dump the current machine status.

Bug: A mistake. Getting out the mistakes is known as debugging.

Bus: Path for signals having a common function. Every "standard" MPU creates three buses: the data bus, address bus, and control bus.

Byte: Set of 8 bits. A byte is universally used to represent a character. Microcomputer instructions require one, two or three bytes.

Call: Instruction used to transfer the program execution sequence to a subroutine or subprogram.

Carraige Return: Standard typewriter key causing the printing element to move back to the beginning of the line.

CCD : Charge Coupled Device. Serial storage technology that uses MOS capacitors.

Chracter Generator: Circuit which forms the letters or numbers on a display or printer.

Checksum: Method used to verify the integrity of data loaded into the computer.

Chip: Rectangular silicon die cut from the wafer. By extension, every LSI package is commonly called a chip.

Combinational Logic: Circuit arrangement in which the output state is determined only by the present state of the input.

Comment Field: Field within an instruction, reserved for comments, which is ignored by the compiler or the assembler.

Compiler: Translation program which converts high-level instructions into a set of binary instructions (object code) for execution. Each high-level language requires a compiler or an interpreter. A compiler translates the complete program which is then executed.

Computer: General-purpose computing system incorporating a CPU, memory, I/O facilities and power supply.

Control Bus: Set of control lines in a computer system. Provides the synchronization and control informtion necessary to run the system.

Core: Small magnetic toruses of ferrite which are used to store a bit of information.

CPU : Central Processing Unit. Computer module in charge of fetching, decoding and executing instructions. It incorporates a control unit, an ALU and related facilities (registers, clocks, drivers)

Crash: Hardware or software malfunction that causes the system to halt or become lost in a loop.

Crosstalk: Interference between two signals.

CRT Terminal: Computer terminal using a CRT display and keyboard, usually connected to the terminal by a serial link.

Current Loop: Means of communicating data via presence or absence of a two-wire cable.

D/A : Digital to analog. Conversion from the digital representation used in computers to the analog signals used to drive speakers, motors, etc.

Data Bus: Set of lines carrying data. In a "standard" 8 bit MPU, the data bus is bidirectional, tristate and has 8 lines.

Debouncing: Elimination of the accidental bounce signals characteristic of mechanical switches. Debouncing may be performed by hardware (latch) or software (delay).

Decoder: Logic device that decodes binary inputs. A 3-bit decoder (e.g. 74138) will have 2↑3 = 8 outputs because a 3-bit number can have 8 different values.

Development system: Microcomputer system with all the facilities required for efficient hardware and software development for a given microprocessor. It includes at least a microcomputer box, plus a CRT display (or TTY), printer, mass-storage (usually dual floppies), PROM programmer, paper-tape reader (as a back up), and in circuit emulator.

Diagnostics: Set of routines used to diagnose system malfunctions.

Digital: Having discrete states. Most logic is binary logic, with two states, on or off.

Diskette: Floppy disk. A circular mylar substrate coated with a magnetic oxide, rotating inside a special jacket which internally cleans the surface.

Directory: Table of contents signed to allow convenient access to specific files.

DMA: Direct memory access method used to provide high speed data transfers between a peripherial and memory

DOS: Disk Operating System integrating disk-file facilities.

Dot Matrix: A method of forming characters by using many small dots.

Double Density: Techniques used to double bit density on a magnetic storage medium, such as MFM, M2FM.

Dynamic Memory: MOS RAM memory using dynamic circuits. Each bit is stored as a charge on a single MOS transistor. This results in very high density (only one transistor per bit)

EBCDIC: 8-bit code used by IBM to encode alphanumeric symbols. It is essentially analogous to ASCII, with a different sequence.

Editor: Program designed to facilitate the entry of text in a computer system. Typical facilities include: insert/line, append, search for "string", substitute (from...to...)

EIA: Electronic Industries Association.

EIA-RS232C: Serial interface standard for asynchronous communications. Data are sent in 10 or 11 bit long serial bundles. The first is a start bit indicating the beginning of the data. The next is the LSB of the data. After the last bit comes the stop bit or bits.

EPROM: Erasable Programmable Read Only Memory. A PROM that can be reused. Most EPROMs can be erased by exposing them to ultraviolet light.

Fairchild: The oldest semiconductor manufacturer in Silicon Valley.

Fan-in: Electrical load presented to an output by an input.

Fan-out: Electrical load that an output can drive. Usually expressed as the number of inputs that can be driven.

Fetch: Reading an instruction from memory.

Firmware: Program stored in ROM. Normally, firmware designates any ROM-implemented program.

Floppy Disk: Mass-storage device that uses a flexible (floppy) diskette to record information.

Flowchart: Graphical representation of of program logic. Flowcharts enable the designer to to visualise the procedure necessary for each item in the program. A complete flowchart leads directly to the final code.

GIGO: Garbage in, garbage out. Implies that misinformation applied to the CPU will result in misinformation output.

Glitch: A pulse or burst of noise.

Half-duplex: Communication technique where data may travel in only one direction at a time.

Handassemble: Translate a program from assembly language to machine code without the assistance of an assembler program.

Hard-copy: Computer output on paper.

Hardware: The bolts, nuts, boards, chips, wires, transformers, etc. The physically existing components of a system.

High-level Language: Programming languge resembling "natural language", with powerful instructions. Examples are Fortran, Basic, APL, ALGOL, COBOL, PL/M. All require a compiler, or an interpreter.

Impact Printer: Any mechanical imprinting device where the characters are formed by striking the ribbon onto the paper.

Input/Output: Lines or devices used to obtain or to display information outside.

Instruction: Single command within a program. Instructions may be arithmetic or logical, may operate on registers, memory, or I/O devices, or may specify control operations. A sequence of instructions is a program.

Integrated circuit: A circuit which is fabricated on a single chip of silicon.

Interrupt: Involves suspension of the normal program that the microprocessor is executing in order to handle a sudden request for service (interrupt).

Interrupt Vectoring: Providing a device ID number or an actual branching address in response to the interrupt acknowledge signal. Allows each interrupt to be serviced by a different routine.

Kansas City Standard: Standard for cassette tape recording and playback of EIA-RS232-C data. Uses frequency-double fequency encoding techniques where a 1 is represented by 8 cycles of 2400 hertz, and a 0 by 4 cycles of 1200 hertz.

Line Printer: High speed printer capable of printing simultaneously a complete line (80 to 120 characters).

LSB: Least Significant Bit.

Machine Language: Set of binary codes, representing the instructions which can be directlyexecuted by a processor.

Memory: Storage area for binary data programs.

Microprocessor: LSI implementation of a complete processor (ALU + Control Unit) on a single chip.

Mnemonic: Symbolic representation, generally an opcode.

Modem: Modulator-demodulator. Used to interface a digital device to a telephone line. Encodes and decodes serial bits into frequencies.

Object code: Code produced by a translator program, such as compiler or assembler, which can be executed by the processor.

Operand: Second part of an instruction, usually data or an address.

Operating System: Software required to manage the hardware resources of a system, and its logical resources, including scheduling and file management.

Operation Code (opcode): Used to describe the segment of the machine language or assembly language instruction specifying the operation to be performed.

Pascal: A high level programming language.

Port: Physical I/O connection. Usually involves 8 bits, for 8-bit microprocessor.

Pull-up Resistor: Used to provide the source current for open-collector logic gates or a termination for unused high inputs.

RAM: Random Access Memory. Denotes in fact Read/Write LSI memory.

ROM: Read Only Memory.

S100: Popular hobbyist standardized bus characterized by 100 pins, and suited to an 8080 type system.

Scratchpad: Group of general purpose registers without specific function providing a high speed workspace. Usually, an internal RAM.

Sector: Triangular section of a disk surface. A block of data is addressed by its track and sector numbers. A typical disk sector has 128 bytes of data.

Silicon Valley: Area around Sunnyvale, California where most of the semiconductor manufacturers are installed. Also called Silicon Gulch.

Source Code: User written program, once entered in the system, usually in ASCII code.

Static Memory: MOS memory which uses a flip-flop as a storage element. It does not need to be refreshed and does not require a clock. It does not lose its contents as long as power is applied.

TTY: Teletype or teletypewriter.

Variable: Symbolic entity which may assume a number of values.

VLSI: Very Large Scale Integration. In practice, over 100,000 transistors per chip.

Word: Logical unit of information. May have any number of bits, but is usually 4, 8, or 16 for MPU's.

APPENDIX C

LETTERS OF PERMISSION

REPLY TO REQUEST FOR INFORMATION ON COMPUTER TRAINING PROGRAM

HEATH

January 14th, 1980.

Mr. John R. Balcom,
356 Pleasant St.,
TRURO, NS
B2N 3T4

Ref: # 008292 G.

Dear Mr. Balcom:-

Thank you for your recent letter concerning the Heathkit digital techniques' training program.

Please accept this letter as our permission for you to reproduce pages 2-5 to 2-18 of the Heathkit digital techniques' program, as part of your thesis. The only stipulation we place on the reproduction of this material is that you clearly indicate the material is produced with permission of the Heath Company and reference be made to the fact that it was obtained from the Heathkit model EE-3201, digital techniques learning program.

Thank you for contacting us in this matter and we trust that the foregoing will be satisfactory.

Yours truly,

HEATH COMPANY, a division of
ZENITH RADIO CANADA LIMITED

G. A. Harris,
Manager, Marketing Services.

GAH/ah

HEATH COMPANY
A DIVISION OF [redacted]
1480 DUNDAS HWY. EAST • MISSISSAUGA, ONTARIO L4X 2R7 • TEL. (416)277-3191
TELEX 06-961114

HEWLETT hp PACKARD

HEWLETT-PACKARD (CANADA) LTD. • 800 Windmill Road, Dartmouth, Nova Scotia B3B 1L1, Telephone 469-7820

January 25, 1980

Mr. John R. Balcom,
356 Pleasant Street,
Truro, Nova Scotia
B2N 3T4

Dear John,

Thank you for your letter of January 23rd regarding your Thesis that you will be submitting to the University.

Please consider this letter your authority to include Pages 295 to 313 of the "Hewlett-Packard Practical Microprocessor Textbook" in your Thesis.

Should you have any additional questions please feel free to contact me at your convenience.

Yours truly,

HEWLETT-PACKARD (CANADA) LTD.,

Roger A. LeBlanc,
Electronic Instruments
Sales Representative

RAL:lcb

Rockwell International

...where science gets down to business

Here's the literature you requested on Rockwell's Microelectronic Products

To get more detailed literature, Applications Engineering or other assistance, please contact your nearest Rockwell representative or distributor, shown on the included list.

If, for any reason, you do not get the immediate and complete service you need, please call this number:

(714) 632-3729

We appreciate your interest in our microelectronic products, and look forward to satisfying your requirements.

ROCKWELL INTERNATIONAL
Microelectronic Devices
P.O. Box 3669
Anaheim, CA 92803, U.S.A.
Marketing Phone: (714) 632-3729

NORTHERN ALBERTA
INSTITUTE OF TECHNOLOGY
Department of
Advanced Education
and Manpower

403/477-4111

11762 - 106 Street
Edmonton, Alberta, Canada T5G 2R1

November 8, 1979

Faculty of Education
Saint Mary's University
Halifax, Canada
B3H 3C3

Attention: Mr. John R. Balcom

Dear Sir:

In response to your request of October 25, 1979 (just received), I have enclosed copies of the following:

1. 1979/81 NAIT Calendar.
2. Electronics Engineering Technology Course Outlines - Year 1 and 2.
3. Electronics Engineering Technology student book/materials lists.

While our two-year Diploma program in Electronics Engineering Technology is not specialized solely to train computer technicians/technologists, employers have expressed a high degree of satisfaction in using our graduates in this capacity.

I hope the information supplied will be useful to you. I would more than welcome information relating to the outcome of your investigation.

Yours very truly,

R. M. Kachman, Head
Electronics Department

RMK*bk

Enclosures

APPENDIX D

COURSE OUTLINES FOR PRACTICAL MICROPROCESSORS AND INDIVIDUAL LEARNING PROGRAM IN MICROPROCESSORS

PRACTICAL MICROPROCESSORS by HEWLETT-PACKARD

Course Objectives and Outline

Course Objectives

a) Acquire a practical knowledge of microprocessor system hardware.

b) Gain a basic understanding of the software that is used to control a microprocessor system.

c) Learn how the system uses this software to perform a wide variety of operations.

d) Use this information to learn practical troubleshooting techniques that are applicable to any microprocessor system.

Course Outline

Section 1. Microprocessor Fundamentals, contains three lessons that provide a basic introduction to microprocessor systems.

Lesson 1. Introduction to Microprocessor Systems.
Lesson 2. Number Systems.
Lesson 3. Software Fundamentals.

Section 2. Introduction To Programming, contains three lessons that provide an introduction to programming and instructions for using the Microprocessor Lab.

Lesson 4. Using the Microprocessor Lab.
Lesson 5. Software Concepts.
Lesson 6. Inside the Microprocessor.

Section 3, Microprocessor System Hardware, contains four lessons that describe microprocessor hardware in detail.

Lesson 7. Basic Hardware Concepts.
Lesson 8. Address Decoding.
Lesson 9. Memories and Peripherals.
Lesson 10. Control Circuits.

Section 4. Programming Microprocessors, contains five lessons that cover some of the more advanced concepts and techniques for programming microprocessors.

Lesson 11. Registers and Breakpoints.
Lesson 12. The 8085 Instruction Set.
Lesson 13. Software Design Techniques.
Lesson 14. Software Control of Peripherals.
Lesson 15. Number Representations and Algorithms.

Section 5. Troubleshooting Microprocessor Systems, contains four lessons that deal with the theory of troubleshooting and the new tools and techniques that have been developed to troubleshoot microprocessor systems.

Lesson 16. Hand-Held Troubleshooting Tools.
Lesson 17. Signature and Logic Analyzers.
Lesson 18. Troubleshooting Microprocessor Systems.
Lesson 19. Troubleshooting the Microprocessor Lab.

Section 6. Other Microprocessors, contains only one lesson. It provides a survey of several currently available microprocessors.

Lesson 20. Microprocessor Survey.

INDIVIDUAL LEARNING PROGRAM IN MICROPROCESSORS
by
HEATHKIT Continuing Education

Course Objectives and Outline

Course Objectives:

When you have completed this course, you will be able to do the following:

1. Program a representative microprocessor.
2. Interface a representative microprocessor with the "outside world."

Course Outline:

Unit 1. Number Systems and Codes
- a) Introduction
- b) Decimal Number System
- c) Binary Number System
- d) Octal Number System
- e) Hexadecimal Number System
- f) Binary Codes
- g) Experiment

Unit 2. Microcomputer Basics
- a) Introduction
- b) Terms and Conventions
- c) An Elementary Microcomputer
- d) Executing a Program
- e) Addressing Modes
- f) Experiment

Unit 3. Computer Arithmetic
- a) Introduction
- b) Binary Arithmetic
- c) Two's Complement Arithmetic
- d) Boolean Operations
- e) Experiment

Unit 4. Introduction to Programming
- a) Introduction
- b) Branching
- c) Conditional Branching
- d) Algorithms
- e) Additional Instructions
- f) Experiment

Unit 5. The 6800 Microprocessor - Part 1
- a) Introduction
- b) Architecture of the 6800 MPU
- c) Instruction Set of the 6800 MPU
- d) New Addressing Modes
- e) Experiment

Unit 6. The 6800 Microprocessor - Part 2
- a) Introduction
- b) Stack Operations
- c) Subroutines
- d) Input-Output (I/O) Operations
- e) Interrupts
- f) Experiment

Unit 7. Interfacing - Part 1
- a) Introduction
- b) Interfacing Fundamentals
- c) Interfacing With Random Access Memory
- d) Interfacing With Displays
- e) Experiment

Unit 8. Interfacing - Part 2
- a) Introduction
- b) Interfacing With Switches
- c) The Peripheral Interface Adapter (PIA)
- d) Using the PIA
- e) Experiment

Unit 9. Programming Experiments
- a) Introduction
- b) Experiment 1. Binary/Decimal Training Program
- c) Experiment 2. Hexadecimal/Decimal Training Program
- d) Experiment 3. Straight Line Programs
- e) Experiment 4. Arithmetic and Logic Instructions
- f) Experiment 5. Program Branches
- g) Experiment 6. Additional Instructions
- h) Experiment 7. New Addressing Modes
- i) Experiment 8. Arithmetic Operations
- j) Experiment 9. Stack Operations
- k) Experiment 10. Subroutines

Unit 10. Interfacing Experiments
- a) Introduction
- b) Experiment 1. Memory Circuits
- c) Experiment 2. Clock
- d) Experiment 3. Address Decoding
- e) Experiment 4. Data Output
- f) Experiment 5. Data Input
- g) Experiment 6. Introduction to the Peripheral Interface Adapter (PIA)
- h) Experiment 7. Audio Output
- i) Experiment 8. Key Matrix and Parallel-to-Serial Conversion
- j) Experiment 9. Digital-to-Analog and Analog-to-Digital Conversion